An Adaptive Mechanism for LTE P-GW Virtualization using SDN and NFV

Luciano Jerez Chaves[*][†], Islene Calciolari Garcia[†], and Edmundo Roberto Mauro Madeira[†]
[*] Computer Science Department, Federal University of Juiz de Fora (UFJF), Brazil.
[†] Institute of Computing, University of Campinas (Unicamp), Brazil.
Emails: {lchaves, islene, edmundo} @ic.unicamp.br

***Abstract*—Software Defined Networking (SDN) and Network Function Virtualization (NFV) paradigms have been widely used to redesign the traditional mobile networks. Despite several proposals on the literature, researchers have drawn limited attention to the virtualization of user-plane functions that demand high traffic volume processing, as the case of Long Term Evolution (LTE) mobile gateways. This paper introduces an adaptive mechanism for the user plane virtualization of the LTE Packet Data Network (PDN) GateWay (P-GW), running entirely on top of OpenFlow switches. Using both SDN and NFV concepts, the proposed mechanism employs elastic computing notions to dynamically activate or deactivate the infrastructure switches so the virtualized gateway can adjust to workload changes. This work addresses both software and hardware OpenFlow infrastructure platforms, and simulation results highlight the benefits that can be achieved by the presented mechanism.**

I. Introduction

Nowadays, the Long Term Evolution (LTE) networks have been de facto employed for high-speed wireless communication, providing an efficient packet-optimized service. However, as stated by the Open Networking Foundation (ONF), network operators are facing challenges as demand for bandwidth expands, and meeting current market requirements is almost impossible with traditional network architectures [1]. Among current limitations, the proprietary gateways in the LTE core network play a significant role. They are designed to handle millions of users at the same time with high availability. So, they are prone to be complex, expensive, and most of the time it is not possible to combine capabilities from different vendors [2]. Also, centralized data processing functions (e.g. billing) force all traffic to go through the gateway, reducing the system agility and creating a single point of failure [3].

Endeavoring to address these issues, many works on the literature attempt to virtualize and "softwarize" existing mobile architecture. In this context, Software Defined Mobile Networking (SDMN) makes the case that Software Defined Networking (SDN) [1] and Network Function Virtualization (NFV) [4] can be used to redesign cellular networks. Some works use NFV concepts to virtualize LTE network functions traditionally implemented in dedicated hardware into the cloud. The main idea of NFV is the decoupling of physical equipment from the functions that run on them [5]. So, the LTE architecture could be fragmented into several Virtual Network Functions (VNFs) that are implemented in software and run on Commercial Off-The-Shelf (COTS) physical servers. The major advantage of using NFV is to reduce middle-boxes deployment to benefit from cost savings and flexibility [6].

Besides, SDN has emerged as a promising paradigm designed to decouple the control and user planes of network elements. The network intelligence is logically centralized, while the underlying infrastructure consists of simplified COTS network devices that provide packet switching. Thus, decision-making is facilitated based on a global (or domain) view, enabling more agile and cost-effective networks. The OpenFlow protocol [7] is the first standard SDN interface, providing high-performance and granular traffic control across multiple switch devices. It uses the concept of flows to identify network traffic based on pre-defined match rules. The switch consists mainly of flow tables, which perform packet lookups and forwarding, based on rules configured by the controller. SDN can be seen as complementary to NFV paradigm. By enabling these technologies, it is viable to deploy more services with fewer costs and resources, assuring them a long-term solution compared to the current shortages [2].

Motivated by the above discussion, this paper presents a mobile architecture that employs NFV and SDN to virtualize the LTE gateways and configure the OpenFlow backhaul network at the same time. The main contribution of this work is the virtualization model for the PDN GateWay (P-GW) — the gateway responsible for connecting the LTE network to the Internet. The SDN concept is used to split the P-GW control and user planes, modeling each segment as an independent VNF. Since the P-GW user plane (P-GWu) imposes significant challenges due to the high traffic volume, its virtualization model is designed to run on top of an OpenFlow switch collection that can be instantiated using hardware or software platforms. This work also employs elastic computing notions to propose an original adaptive mechanism to dynamically activate or deactivate the switches on the virtualization infrastructure to ensure that the P-GWu can adapt to workload changes. Discussions about whether using hardware or software switches are addressed, along with simulation results that highlight the benefits of the proposed mechanism.

This paper is organized as follows: Section II presents background concepts on LTE networks, whereas Section III reviews how SDN and NFV have been used to enhance mobile networks. Section IV shows the adopted architecture, and Section V presents the P-GWu VNF. Performance evaluation is unveiled in Section VI, followed by the conclusions.

978-3-901882-98-2 © 2017 IFIP

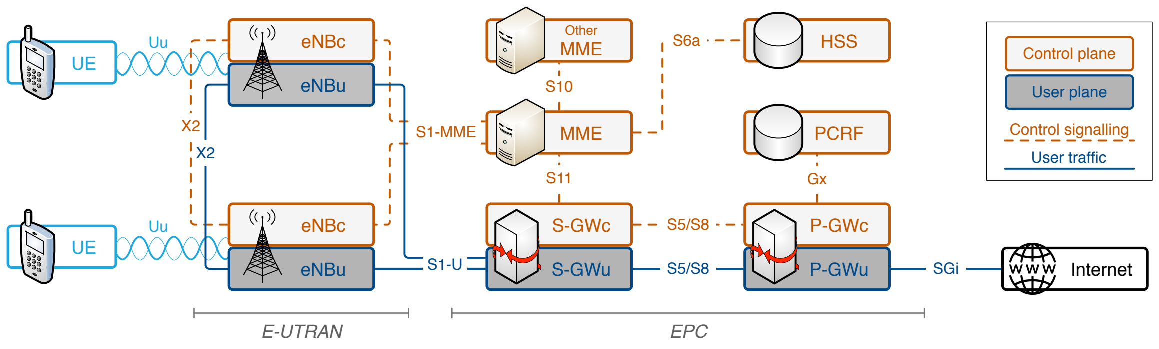


Fig. 1. The Evolved Packet System (EPS) network architecture.

II. Long Term Evolution

LTE is a 3rd Generation Partnership Project (3GPP) collection of standards for high-speed wireless communication [8]. There are three components: The User Equipment (UE), representing the mobile device; the Evolved Universal Terrestrial Radio Access Network (E-UTRAN), composed by the Evolved Node B (eNB) responsible for all radio-related functions; and the Evolved Packet Core (EPC). Together, they comprise the Evolved Packet System (EPS), as illustrated in Fig. 1.

The EPC consists of control and user-plane nodes. The main control-plane nodes are:

- The *Mobility Management Entity (MME)* processes the signaling between the UE and the EPC. Functions supported by the MME are related to bearer management, connection management (security, gateway selection, mobility, and handover), and roaming with other networks.
- The *Policy Control and Charging Rules Function (PCRF)* is responsible for Quality of Service (QoS) decision-making and data flow authorization on user-plane nodes.
- The *Home Subscriber Server (HSS)* maintains a database of subscriber-related information, including the user profile and information like roaming, security, and location.

The EPC gateways connect the eNBs to the Internet. They implement both user and control-plane functions:

- The *PDN GateWay (P-GW)* is the point of contact with the outside world. IP address allocation for UE and accounting for inter-operator charging are examples of functions handled by the P-GW control plane (P-GWc). On the P-GW user plane (P-GWu), the heavy burdens are the QoS policy enforcement and the downlink IP packets filtering into bearers (described in detail below).
- The *Serving GateWay (S-GW)* connects the eNB to the P-GW, acting as a local mobility anchor for UE handover. The S-GWc monitors the traffic for charging purposes and the S-GWu can be used for lawful interception.

The LTE uses the concept of bearers to route IP traffic from the P-GW to the UE [9]. An EPS bearer is mapped into lower-level tunnels as it crosses multiple EPC interfaces. On S1 and S5/S8, the General Packet Radio Service (GPRS) Tunnelling Protocol (GTP) is used to encapsulate the bearer within the GTP / UDP / IP protocols. The IP addresses and the Tunnelling End ID (TEID) are used to identify each tunnel on the network.

An EPS bearer uniquely identifies packet flows from the same UE that receives a common QoS treatment. Bearers can be classified into *Guaranteed Bit Rate (GBR)* bearers, which have an associated bit rate for which dedicated transmission resources are permanently allocated; and *Non-Guaranteed Bit Rate (Non-GBR)* bearers, which do not assure any particular bit rate. Services on top of GBR bearers can assume that congestion-related packet losses will not occur. When the UE attaches to the network, the EPC establishes the Non-GBR UE default bearer. Additional GBR or Non-GBR dedicated bearers can also be created during or after the initial attach procedure.

IP packet filtering into different EPS bearers is based on Traffic Flow Templates (TFTs). The TFTs use IP and TCP/UDP header information (mainly source/destination addresses and port numbers) to filter packets so each one can be sent to the respective bearer with the appropriate QoS. For the downlink traffic, this heavy task is assigned to the P-GWu that has to identify the target S-GW IP address and the TEID for the GTP encapsulation processes. On the uplink direction, the analog process is performed by the eNB.

III. Integrating SDN and NFV into Mobile Networks

A general SDMN architecture called MobileFlow is proposed by Pentikousis et al. [11], moving the entire control plane to software while the user plane is composed only of simple elements with tunnel handling support. Kempf et al. [12] present a study on the evolution of cloud-based EPC, where all the control functions of EPC elements are moved into the cloud, and the user plane is shifted into OpenFlow switches with GTP support. Hampel et al. [10] introduce the Vertical Forwarding concept to handle tunnels in mobile networks, using only a few switches that tunnel and de/encapsulate user-plane data. On a different direction, Jin et al. [13] introduce SoftCell, a disruptive solution that removes GTP tunnels and replaces all user plane EPC elements by simple OpenFlow switches and a set of middle-boxes. The controller directs traffic over the network and middle-boxes based on the traffic service. A new OpenFlow-based control plane for EPC is presented by Said et al. [14], with the focus

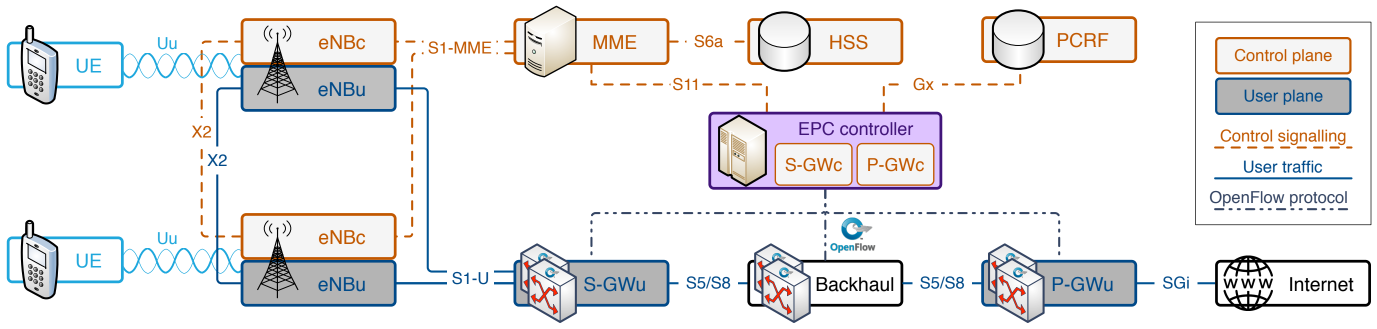


Fig. 2. The adopted Software Defined Mobile Networking (SDMN) architecture.

on resiliency and load balancing. The proposed architecture ensures the on-demand connectivity service even in critic situation such as network equipment failure and overload conditions. Mahmoodi et al. [15] realize the SDN concept into the backhaul network through a distributed control plane including the MME, HSS, and PCRF elements. They show that the new architecture reduces the UE power consumption and the signaling overhead between in backhaul entities.

When it comes to NFV, Hawilo et al. [4] discuss the challenges and requirements of its use in mobile networks. They propose a criterion to bundle multiple functions of a virtualized EPC in a single physical device or a group of nearby devices. Jain et al. [16] present the design and evaluation of two open-source implementations of the LTE EPC, one based on SDN principles and the other based on NFV. Their results show that a NFV-based implementation is better suited for handling signaling traffic on the control plane, while a SDN-based design is better suited for user plane traffic because SDN switches are often more optimized for packet forwarding than virtualized software appliances.

On the gateway virtualization context, Basta et al. [17] made an analysis of mobile gateway functions and mapped them into alternative deployment frameworks, presenting the pros and cons of each deployment decision. They continued their work in [18], addressing the function placement problem based on two broad categories: gateway virtualization and gateway decomposition. The virtualization requires directing the data traffic from the transport network to the data center, which imposes an additional load and increases the expected traffic delay. When decomposing gateway functions, only the control plane is shifted to the data center, and enhanced SDN network elements process the user plane. To find the most optimal deployment solution, the authors formed a model by taking the control-plane load and user-plane latency into account and then tried to minimize these parameters. An et al. [3] focus on the P-GW virtualization and propose an architecture that applies SDN to decouple the control and user planes, and NFV to implement the gateway's user-plane forwarding function on low-cost COTS hardware. A cluster model is used to overcome the performance limitations of a single server, and OpenFlow switches together with an enhanced OpenFlow controller act as the load balancer for the proposed architecture.

IV. The Adopted SDMN Architecture

Built upon strength concepts from the literature, Fig. 2 illustrates the adopted SDMN architecture for this paper. SDN and NFV principles are used to split S/P-GW control and user planes, following the same ideas discussed in [14], [18]. Even so, different from other proposals, this architecture also encompass the backhaul network under the management of the same EPC controller.

The backhaul network is built upon enhanced OpenFlow switches, which were slightly modified so they can handle the tunneled traffic. This architecture preserves the GTP tunnels, and the OpenFlow switches use only the TEID field for traffic routing. Previous work from the same authors of this paper addresses different aspects related to this backhaul network implementation, aiming to enforce traffic QoS requirements [21].

The EPC controller becomes the central control-plane element. It is responsible for proactively configuring the backhaul network and for handling all S/P-GW control plane procedures. The EPC controller must comply with the standard interfaces for communication with other elements (MME and PCRF) and, since the communication between the P-GWc and S-GWc is internal to the controller, it reduces the number of messages on the network. The EPC controller also coordinates the use of equal TEID values for the same bearer on both S1 and S5/S8 interfaces. Because of that, there's no need for a new downlink packet filtering and TEID mapping at the S-GW, reducing the number of installed OpenFlow rules. As indicated by Kiess et al. [19], the EPC control plane requires high amounts of computing and storage resources, but comparatively low throughput. Therefore, the proposed EPC controller can be safely modeled as a VNF and instantiated on the cloud.

Finally, virtualizing the S/P-GWu functions is a challenging task due to the high amount of the traffic load from the user plane [19]. Even with the advantages of recent technologies for packet processing on commodity CPUs, like Data Plane Development Kit (DPDK) [20], a network operator may still face performance limitations when compared to hardware platforms, especially because transferring packets between kernel and userspace is a costly operation [16]. This constraint motivated the decision to implement S/P-GWu using OpenFlow switches rather than generic VNF in software. On the

S/P-GWu, OpenFlow logical ports are used to de/encapsulate the GTP traffic so that the gateways can match and perform required action on top of encapsulated IP user packets. On this SDMN architecture, all S/P-GWu operations are implemented using standard OpenFlow instructions, actively supported by the set-field action and by the OpenFlow eXtensible Match (OXM) metadata field carried between logical ports.

With this approach, the gateways can still be modeled as VNFs running on top of virtual OpenFlow switches (like the Open vSwitch, which offers an optimized implementation for packet forwarding in kernel space [22]). However, as an alternative, it is possible to deploy this same architecture using hardware OpenFlow switches, which are designed to ensure packet processing at line rate. On the other hand, hardware switches have limited number of entries in its internal pipeline tables. This restriction is usually associated with higher costs for implementing Ternary Content-Addressable Memory (TCAM) on hardware. So, selecting a hardware or software switch platform is a trade-off that depends on the network requirements and available infrastructure.

The central point of this paper is the adaptive mechanism for P-GWu virtualization that can meet different traffic load requirements while overcoming previously discussed limitations, regardless of using software or hardware OpenFlow switches.

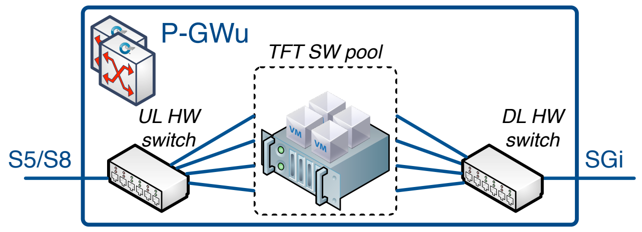
V. P-GW USER PLANE VIRTUALIZATION

On the user plane, the heaviest P-GW task is the filtering of downlink user IP packets into the different QoS-based bearers using TFTs. When using OpenFlow protocol to implement TFT filtering, at least one OpenFlow entry must be installed at the P-GWu for each active bearer. This flow rule will match the 5-tuple header fields to identify the packet flow and get the proper bearer TEID and S-GW IP address. These two values will be combined and saved on the OpenFlow metadata field, and the packet will be sent to the logical port connecting the P-GWu to the backhaul network. This logical port will encapsulate the user IP packet within the GTP tunnel. From this point on, all routing decisions at backhaul OpenFlow switches will be performed solely based on TEID values.

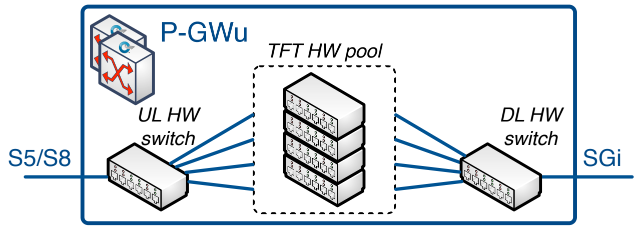
Considering that a UE can have some dedicated bearers other than the default bearer, and considering that each bearer can convey the traffic from different applications, the total number of flow entries on the P-GWu is prone to grow quickly. Despite the flexibility offered by the adopted SDMN architecture, implementing the P-GWu with a single OpenFlow switch is not realistic. Therefore, the P-GWu virtualization employs the elastic computing concept, so computing resources can be scaled up and down easily whenever required.

A. *The P-GWu internal design*

Fig. 3 shows the P-GWu internal design. No matter whether using hardware or software platform, a pool of OpenFlow TFT switches is required to attend the intense traffic filtering demand. When using software-based TFT switches (Fig. 3a), the table size may not be the problem, but a single virtual appliance probably will not be able to handle all the traffic load. When using hardware-based TFT switches (Fig. 3b), individual flow tables will get full soon, even when there is available processing capacity for use.



(a) Software-based TFT switches.



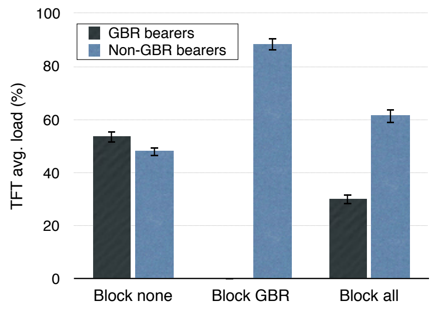
(b) Hardware-based TFT switches.

Fig. 3. The P-GWu internal design.

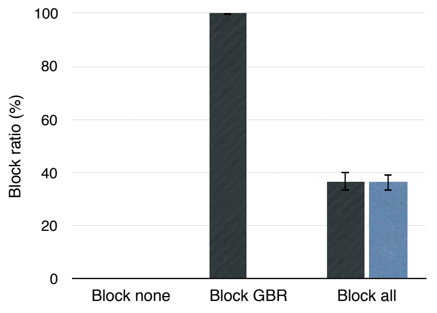
To implement this pool of TFT switches, the P-GWu design also requires a pair of uplink (UL) and downlink (DL) OpenFlow switches with the single purpose of redirecting the traffic of each UE to one of the available TFT switches. Ensuring that all the traffic of a single UE is processed at the same TFT switch allows the P-GWc to query for traffic statistics via OpenFlow protocol for charging purposes. So, the load balancing among TFT switches is based solely on UE IP address masked matching. Because of that, the number of flow entries installed on UL/DL switches will be negligible (mainly, one rule for each active TFT switch), and these entries will only be updated when the number of active TFT switches changes. However, the traffic load on the UL/DL switches can be very high and the processing capacity of these switches may become a potential design bottleneck. Because of that, using a high-performance hardware-based platform for implementing these elements is the best option in this case.

B. *The adaptive mechanism for elastic computing*

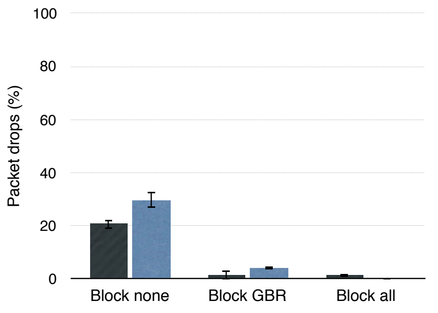
The proposed adaptive mechanism dynamically configures the number of active TFT switches to meet current traffic demand. For software platforms, this means deploying and starting more virtual switches on the service provider infrastructure. For hardware platforms, the hardware switch must be already installed and connected, but they may be turned on/off based on current needs. Every time the number of active TFT switches is adjusted, the EPC controller must move some OpenFlow flow entries to ensure a proportional load balancing among active TFT switches. It means updating the UL and DL switches, installing new rules on some TFT switches and removing old rules from other switches (or let them expire, depending on the adopted configuration).



(a) P-GWu traffic load.



(b) P-GWu overloaded block ratio.



(c) P-GWu overloaded packet drops.

Fig. 4. Simulation results for three different blocking strategies.

Two different threshold values were defined for the adaptive mechanism operation. The first one is the *split threshold*, used to trigger the increase in the number of active TFT switches. This threshold value should be close to the maximum capacity, to avoid wasting resources. The second one is the *join threshold*, used to reduce the number of active TFT switches whenever fewer elements can accommodate the current traffic requirements. These threshold values are defined as a percentage to indicate the relative usage of the monitored resource.

The adaptive mechanism monitors two resources on its current implementation: the flow table usage and the current traffic processing load. The adaptive mechanism will increase its level by doubling the number of active TFT switches when either the flow table usage or the traffic load exceeds the split threshold value. On the other hand, the mechanism will decrease its level to half of active TFT switches whenever both monitored resources fall below the join threshold.

The current implementation uses a factor of 2 for increasing and decreasing the number of active TFT switches. This approach was adopted because it simplifies the OpenFlow matching processes at UL/DL switches when using the UE IP address for traffic load balancing. Considering 2^n the number of active TFT switches at any moment, a maximum of 2^n flow entries on both UL/DL switches is enough for a masked match on the n Least Significant Bits (LSBs) from UE IP address to equally distribute the traffic among the active TFT switches. Different OpenFlow matching strategies can be developed and evaluated, as long as they keep the number of flow entries at UL/DL switches fixed to a constant value. If the number of flow entries grows along with the number of users, then the flow table size of the UL/DL switches may become a bottleneck, reducing the applicability of the proposed design.

C. The blocking strategy

If for some reason the adaptive mechanism reaches its maximum level and no more TFT switches can be activated, the EPC controller may start blocking new bearer requests until resources are released. For hardware platforms, this circumstance may occur when all physical switches are already in use. Situations like that are not expected to happen in software platforms, but in a restrictive context, the physical servers may run out of computing or networking resources, preventing the deployment of new VNF instances.

For the case of flow tables filled up to maximum capacity, no more entries can be added to the TFT switch, so the network must block subsequent requests to avoid OpenFlow error messages or the eviction of installed rules. However, when the traffic load exceeds the maximum switch processing capacity, it is still possible to accept new bearer requests without QoS guarantees. So, three different blocking strategies were compared for the case of an overloaded switch:

- *Block none* requests, allowing overloaded switches to drop random packets when exceeding its internal processing capacity;
- *Block GBR* bearer requests only, ensuring that traffic with QoS requirements can only be accepted if there are enough processing resources at the OpenFlow switches;
- *Block all* bearer requests, regardless of the bearer type.

Initial simulations were performed to study the outcomes of each blocking policy (Sec. VI will describe the simulation scenario used in this paper). Fig. 4 shows the average simulation results calculated from 24 different simulation seeds with 95% of confidence interval. Each graph shows the values for both GBR and Non-GBR bearer traffic, considering all blocking strategies when the network traffic and bearer requests are already in the steady state. Fig. 4a shows the TFT switch average traffic load ratio, expressed as a percentage of the maximum processing capacity. Fig. 4b displays the bearer request block ratio, and Fig. 4c shows the packet drop rate due to the overloaded TFT switch.

It becomes evident that the *Block GBR* strategy converges to a condition where the EPC controller blocks all GBR requests because the Non-GBR traffic completely consumes the P-GWu processing capacity. In turn, the *Block none* strategy can equally share the processing capacity but at the price of a very high packet drop ratio. Both situations are not acceptable for handling GBR traffic with strict QoS requirements, and its use may not be viable in real network deployments. It is possible to conclude that the *Block all* policy can bring better results for these simulations, with minor packet drop ratio and balanced block ratio. Because of that, subsequent simulations in this paper adopt this strategy.

VI. P-GWu PERFORMANCE EVALUATION

A. Simulation scenario

The simulation scenario was implemented in the Network Simulator 3 (ns-3) using the `OFSwitch13` module, which enhances the ns-3 with OpenFlow 1.3 technology. This module was developed by the authors of this paper and it is described in [23]. Due to CPU and memory limitations for performing large-scale LTE simulations on ns-3, a reduced simulation scenario was adopted, as illustrated in Fig. 5. The backhaul network is composed of six OpenFlow switches in a ring topology, connected by Gigabit Ethernet full-duplex links. The ring was chosen as most of the legacy backhaul networks have a ring access topology, and it is one of the most efficient approaches regarding protection and costs. The P-GWu and five LTE sites are also connected to the ring switches over Gigabit Ethernet links. Each site has three eNBs and its own S-GWu. The eNBs are laid out on a hexagonal grid with an inter-site distance of 500 meters. Fig. 6 shows the REM, which represents the Signal-to-Interference-plus-Noise Ratio (SINR) in the downlink with respect to the eNB that has the strongest signal at each point. The UEs, varying from 100 to 400 on this scenario, are randomly scattered over all the coverage area. The EPC controller was entirely implemented from scratch by the authors of this paper inside ns-3. So, it is tailored to the adopted simulation scenario, taking advantage of the global view of the network.

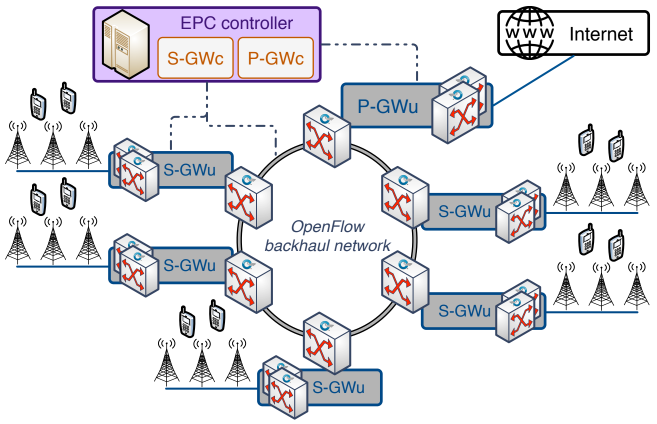


Fig. 5. The simulation scenario topology.

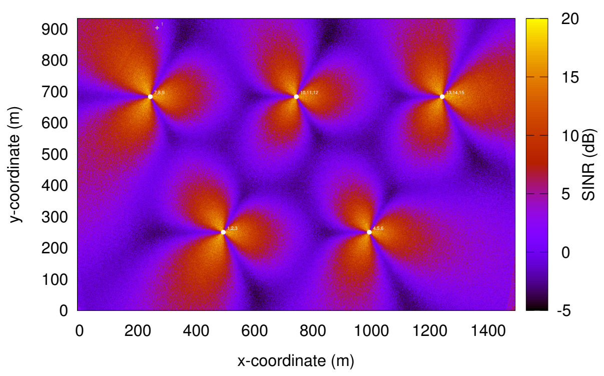


Fig. 6. The Radio Environment Map (REM) for the simulation scenario.

As LTE networks focus on Voice over IP (VoIP) and multimedia applications, four applications are used during the simulations to provide five different types of traffic flows with different QoS requirements. They are: a HyperText Transfer Protocol (HTTP) application for web page access mapped to QoS Class Identifier (QCI) 9; a VoIP conversation with call length expected to 100 seconds over UDP protocol and mapped to QCI 1; a live MPEG-4 video streaming over UDP protocol and mapped to QCIs 2 or 7; and a buffered MPEG-4 video streaming over TCP protocol and mapped to QCI 6 (videos with expected length of 90 seconds).

Apart from the established default bearer, any UE can request for randomly dedicated bearer context activation. Note that multiple bearers can be simultaneously established with different QCI values for a unique UE. The individual bearer requests generated by each UE follows a Poisson Process with rate $\lambda = 0.\bar{3}$ requests per minute while the traffic length is determined by the application. In this approach, the more active users on the network, the greater is the aggregated Poisson process rate.

Table I summarizes the LTE parameters that were settled in ns-3 for the simulations, while Table II shows the parameters used for the P-GWu OpenFlow switch modeling, adjusted to reflect the proportionality of the scenario. The P-GWu virtualization considers both software and hardware OpenFlow platforms. Hence, simulations use different flow table size and traffic processing capacity values for each configuration. These values were defined based on empirical observations from information available at [24]–[28]. It was noticed that software switch processing performance is near 10 to 15 times lower when compared to COTS hardware switches. Although, software switches usually support up to 150 times more entries on its pipeline tables.

TABLE I
LTE PARAMETERS ADJUSTED IN NS-3.

Parameter	**Value**
System frequency	2100 MHz
System bandwidth	20.0 MHz
eNB TX power	46 dBm
UE TX power	23 dBm
SRS periodicity	320 ms
Propagation model	`OhBuildingsPropagationLossModel`
MAC scheduler	`CqaFfMacScheduler`

TABLE II
P-GWu OPENFLOW SWITCH MODELING PARAMETERS.

Parameter	**Hardware**	**Software**
Flow table size	1200	4096
Processing capacity	1 Gbps	100 Mbps
TFT switches	1, 2 or 4	
Split threshold	90%	
Join threshold	30%	

B. The P-GWu adaptive mechanism evaluation

To better understand how the P-GWu adaptive mechanism operates, simulations were performed for a scenario with 500 UEs and an OpenFlow software-based P-GWu virtualization. On this scenario, the flow table size can hold all TFT entries, so the switch processing capacity becomes the only performance constraint. Fig. 7 simultaneously shows the number of active TFT switches (right Y-axis), the gateway traffic throughput (left Y-axis), and the average individual TFT processing load (bottom chart) on the course of 500 seconds of simulation. It is possible to observe how the increasing traffic throughput at the beginning of the simulation reflects on the TFT switches activation. Whenever the average TFT processing load hits the split threshold (90% on the bottom chart), the number of active TFT switches increase twofold. This happens two times before 100 seconds of simulation. After 150 seconds, the traffic throughput begins to slow down, and the adaptive mechanism reduces the number of active TFT switches every time the average TFT processing load hits the join threshold (30%). This figure highlights how the P-GWu adaptive mechanism can adjust the number of active TFT switches based on workload changes. Note that the conservative join threshold value used on simulations reflects in the delayed reduction of active TFT switches. However, it helps to prevent the ping-pong effect in the case of unstable traffic throughput.

For evaluating the P-GWu design under different configurations, Fig. 8 shows the steady-state average simulation results of several metrics, calculated from 24 different simulation seeds with omitted confidence interval due to negligible values. Both software and hardware platforms were evaluated, and the case where only a single TFT switch is available is contrasted to the situation where the adaptive mechanism can activate 1, 2, or 4 TFT switches. Fig. 8a shows the number of TFT switches required to accommodate the flow entries and

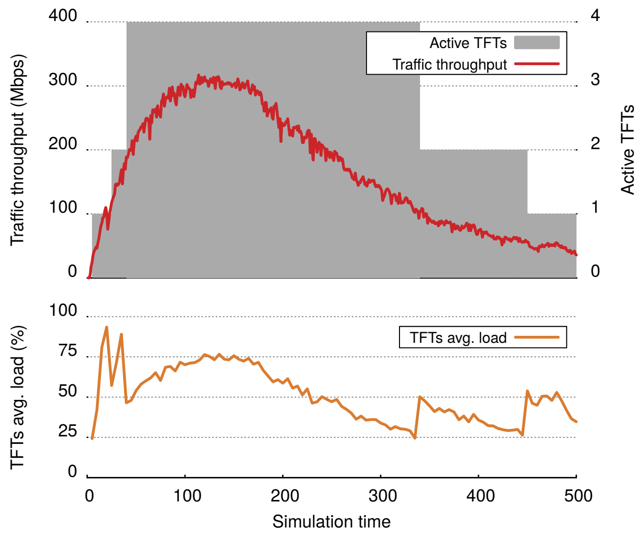


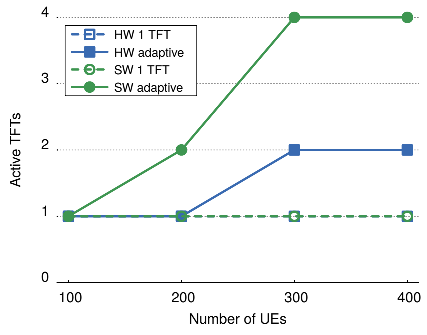
Fig. 7. The P-GWu adaptive mechanism into action.

traffic throughput generated by the increasing number of UEs. The dashed lines, for both software and hardware platforms, remains stuck to the single TFT switch available. However, when the adaptive mechanism comes into action, the solid line shows that the number of TFT switches increases alongside with the number of UEs. This is expected, as more UEs on the network directly reflects on more traffic and flow entries at the P-GWu. It is important to note that the software platform has a very limited processing capacity when compared to the small table size of hardware switches. As a consequence, the software-based infrastructure requires more TFT switches. Fig. 8b and Fig. 8c show the average flow table usage and traffic load for TFT switches, respectively. As the hardware OpenFlow switch has a limited flow table size, it is possible to observe the higher average flow table usage in Fig. 8b. When a single TFT switch is available, the flow table will get full for 300 UEs or more, and the controller must block new bearer requests as shown in Fig. 8e. Nonetheless, the adaptive mechanism can avoid this by activating more TFT switches and distributing the flow entries among them. A similar behavior can be observed for the software platform with respect to the traffic load. The high traffic load for 200 UEs or more (Fig. 8c) results in blocked requests as shown in Fig. 8f, but the adaptive mechanism eliminates these blocks. Finally, Fig. 8d displays the network aggregated throughput for each configuration. The adaptive mechanism on both platforms can adapt the number of active TFT switches improving overall network performance.

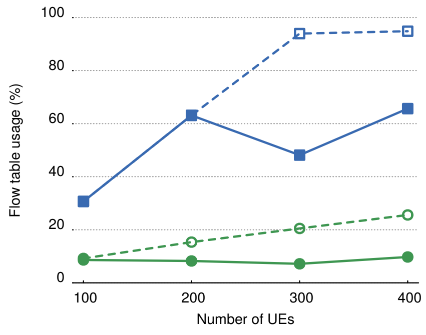
C. Final remarks for a real-scale scenario

Simulations carried out in this paper were realized on a scaled-down scenario. As previously observed, the P-GWu traffic load and flow table usage are directly proportional to the number of UEs for the adopted traffic model. Results in Fig. 8 show that 400 UEs generate near 250 Mbps of traffic throughput and install almost 1600 flow entries on TFT switches when the network reaches its steady state. Hence, on average, a single UE is responsible for 625 Kbps of traffic throughput and 4 TFT flow rules.

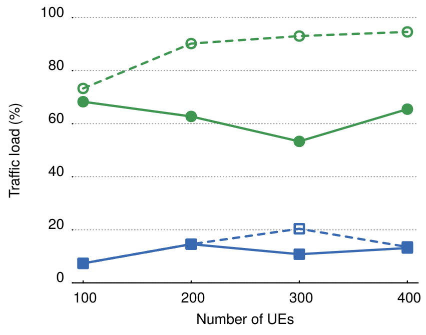
For the proposed P-GWu design shown in Fig. 3, a potential bottleneck may be processing capacity of the UL/DL switches (the number of flow entries on these switches can be negligible, as discussed in Sec. V). As the processing capacity of COTS hardware OpenFlow switches ranges from 90 to 240 Gbps [26]–[28], a UL/DL switch device with 240 Gbps could process the traffic of almost 400K active UEs. Based on this maximum P-GWu theoretical capacity, Table III shows the required number of TFT switch for both software and hardware platforms. The flow table size on hardware switches can vary from 1K to 16K entries. So, a first-class TFT hardware switch [28] can handle up to 4K UEs, limited by the number of flow entries on internal pipeline tables. For the Open vSwitch in software platform, the flow tables size can reach up to 750K entries, but the processing capacity is limited to 10 Gbps on conventional servers [24]. Because of that, a TFT software switch can handle 1.6K UEs. Based on



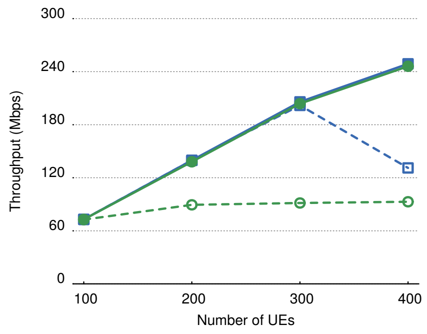
(a) Number of active TFT switches.



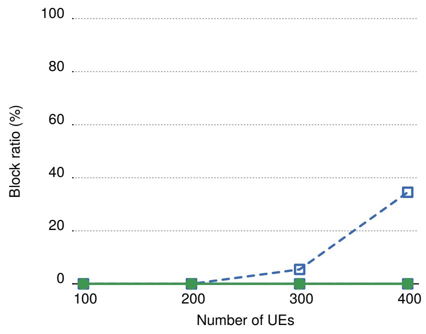
(b) Average P-GWu flow table usage.



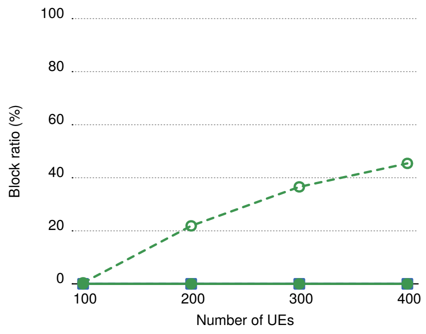
(c) Average P-GWu traffic load.



(d) Network aggregated throughput.



(e) Block ratio caused by full tables.



(f) Block ratio caused by overloaded switches.

Fig. 8. The P-GWu adaptive mechanism performance evaluation.

these numbers, it would be necessary to have up to 128 TFT hardware switches or 256 TFT software switches instances so the P-GWu adaptive mechanism can accommodate the network load of 400K UEs.

TABLE III
P-GWu TFT switch specifications.

Parameter	**Hardware**	**Software**
Flow table size	16K	750K
Processing capacity	240 Gbps	10 Gbps
UEs supported	≈ 4K	≈ 1.6K
Required TFT switches	128	256

VII. Conclusions and Future Work

Rethinking the current LTE architecture is mandatory for the evolution toward the next generation of cellular networking. In this context, SDMN makes the case that SDN and NFV can simplify mobile networks and lower management costs. This paper discusses the virtualization of LTE gateways, with considerable attention dedicated to the P-GW. As the high traffic volume on P-GWu imposes significant challenges, the adopted solution employs the OpenFlow protocol for user plane virtualization. Besides, the adaptive mechanism uses elastic computing notions to dynamically adjust the number of switches on the OpenFlow infrastructure, ensuring that the P-GWu can adapt to workload changes.

Performance evaluation on a scaled-down scenario was realized using the ns-3. Based on the analysis of the adaptive mechanism behavior throughout the simulation, it is possible to conclude that the adopted approach ensures the de/activation of OpenFlow switches as expected. Discussions about the pros and cons when using software and hardware OpenFlow switches are also presented. Considering different flow table sizes and processing capacity for both platforms, it becomes noticeable that software platforms require more virtual switches instances than hardware devices to support the same workload. However, using a software-based infrastructure offers more flexibility for mobile operators to deploying VNFs. Finally, this paper presents a numerical analysis for a real-scale scenario.

As future work, the authors aim to develop other TFT load balancing strategies that guarantee a minimum number of flow entries on UL/DL switches, allowing the use of arbitrary (non-power of 2) TFT switches. They also intend to employ the same adaptive mechanism for S-GWu virtualization.

Acknowledgment

This work has received funding from the European Union's Horizon 2020 for research, technological development, and demonstration under grant agreement no. 688941 (FUTEBOL), as well from the Brazilian Ministry of Science, Technology and Innovation (MCTI) through RNP and CTIC. The authors also would like to thank the Brazilian funding agencies CNPq, CAPES, and FAPESP.

References

[1] Open Networking Foundation, "Software-Defined Networking: The new norms for networks," ONF White Paper, 2012.

[2] L. J. Chaves, V. M. Eichemberger, I. C. Garcia, and E. R. M. Madeira, "Integrating OpenFlow to LTE: some issues toward Software-Defined Mobile Networks," in *International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE, Jul 2015, pp. 1–5.

[3] X. An, W. Kiess, and D. Perez-Caparros, "Virtualization of cellular network EPC gateways based on a scalable SDN architecture," in *Global Communications Conference (GLOBECOM)*. IEEE, Dec 2014, pp. 2295–2301.

[4] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, "NFV: state of the art, challenges, and implementation in next generation mobile networks (vEPC)," *IEEE Network*, vol. 28, no. 6, pp. 18–26, Nov 2014.

[5] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 236–262, Firstquarter 2015.

[6] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[7] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, Apr 2008.

[8] S. Sesia, I. Toufik, and M. Baker, Eds., *LTE: The UMTS Long Term Evolution: From Theory to Practice*, 2nd ed. Wiley, 2011.

[9] H. Ekstrom, "QoS control in the 3GPP evolved packet system," *IEEE Communications Magazine*, vol. 47, no. 2, pp. 76–83, Feb 2009.

[10] G. Hampel, M. Steiner, and T. Bu, "Applying software-defined networking to the telecom domain," in *Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, Apr 2013, pp. 133–138.

[11] K. Pentikousis, Y. Wang, and W. Hu, "MobileFlow: Toward software-defined mobile networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 44–53, Jul 2013.

[12] J. Kempf, B. Johansson, S. Pettersson, H. Luning, and T. Nilsson, "Moving the mobile evolved packet core to the cloud," in *International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, Oct 2012, pp. 784–791.

[13] X. Jin, L. E. Li, L. Vanbever, and J. Rexford, "SoftCell: scalable and flexible cellular core network architecture," in *Conference on Emerging Networking Experiments and Technologies (CoNEXT)*. ACM, 2013, pp. 163–174.

[14] S. Said, M. R. Sama, K. Guillouard, L. Suciu, G. Simon, X. Lagrange, and J.-M. Bonnin, "New control plane in 3GPP LTE/EPC architecture for on-demand connectivity service," in *International Conference on Cloud Networking (CloudNet)*. IEEE, Nov 2013, pp. 205–209.

[15] T. Mahmoodi and S. Seetharaman, "Traffic jam: Handling the increasing volume of mobile data traffic," *IEEE Vehicular Technology Magazine*, vol. 9, no. 3, pp. 56–62, Sep 2014.

[16] A. Jain, S. N. S., S. K. Lohani, and M. Vutukuru, "A comparison of SDN and NFV for re-designing the LTE packet core," in *Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, Nov 2016, pp. 74–80.

[17] A. Basta, W. Kellerer, M. Hoffmann, K. Hoffmann, and E.-D. Schmidt, "A virtual SDN-Enabled LTE EPC architecture: A case study for S-/P-Gateways functions," in *SDN for Future Networks and Services (SDN4FNS)*. IEEE, Nov 2013, pp. 1–7.

[18] A. Basta, W. Kellerer, M. Hoffmann, H. J. Morper, and K. Hoffmann, "Applying NFV and SDN to LTE mobile core gateways, the functions placement problem," in *Workshop on All Things Cellular: Operations, Applications, & Challenges (AllThingsCellular)*. ACM, Aug 2014, pp. 33–38.

[19] W. Kiess, X. An, and S. Beker, "Software-as-a-service for the virtualization of mobile network gateways," in *Global Communications Conference (GLOBECOM)*. IEEE, Dec 2015, pp. 1–6.

[20] Linux Foundation, "Data Plane Development Kit (DPDK)," Online, 2017, available at https://dpdk.org.

[21] L. J. Chaves, I. C. Garcia, and E. R. M. Madeira, "OpenFlow-based mechanisms for QoS in LTE backhaul networks," in *Symposium on Computers and Communication Workshops (ISCC)*. IEEE, Jun 2016, pp. 1233–1238.

[22] "Open vSwitch," Online, 2017, available at http://http://openvswitch.org.

[23] L. J. Chaves, I. C. Garcia, and E. R. M. Madeira, "OFSwitch13: Enhancing ns-3 with OpenFlow 1.3 support," in *Workshop on ns-3 (WNS3)*. ACM, Jun 2016, pp. 33–40.

[24] M. Challa, "Open vSwitch performance measurements & analysis," Online, 2014, Open vSwitch 2014 Fall Conference. Available at http://openvswitch.org/support/ovscon2014/.

[25] P. Emmerich, D. Raumer, F. Wohlfart, and G. Carle, "Performance characteristics of virtual switching," in *International Conference on Cloud Networking (CloudNet)*. IEEE, Oct 2014, pp. 120–125.

[26] L. C. Costa, A. B. Vieira, E. de Britto e Silva, D. F. Macedo, G. Gomes, L. H. A. Correia, and L. F. M. Vieira, "Performance evaluation of OpenFlow data planes," in *Symposium on Integrated Network Management (IM)*. IEEE, May 2017, pp. 1–6.

[27] D. Y. Huang, K. Yocum, and A. C. Snoeren, "High-fidelity switch models for software-defined network emulation," in *SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN)*. ACM, 2013, pp. 43–48.

[28] "NoviSwitch: Switching Made Smarter," Online, 2017, available at https://noviflow.com/products/noviswitch/.