Software-Defined Networking for Low-Latency 5G Core Network

Jérémy Pagé and Jean-Michel Dricot

Abstract—Mobile communications have grown exponentially these last few years leading to an increasing demand. Software-Defined Networking appears to be a promising technology in order to provide flexible Quality of Service and very low latency. By modifying the 4G LTE architecture and integrating OpenFlow (SDN protocol) towards the 5G architecture, this new architecture promises to fulfill the requirements and to achieve better performances while participating to the network functions virtualization.

Index Terms—Core Network, 4G, 5G, SDN, OpenFlow

I. INTRODUCTION

Over the last decade, mobile cellular subscriptions have increased exponentially leading to a higher demand in availability, faster transmissions and lower response times.

The evolution of the mobile communications technologies is presented by [1]. The first generation introduced the basic mobile voice service, while the second generation (GSM) improved by adding capacity and coverage. The third generation (3G/WCDMA) has brought data at higher speeds. The fourth and current generation (4G/LTE) integrates all the services (voice, data, . . .) in an Internet Protocol (IP) environment. Over these generations, the number of components and the number of protocol layers have been significantly reduced. These reductions allowed to reach higher speeds and faster data processing.

The data bandwidth expected for the last generations (3G and 4G) and the new coming (5G) is presented by [2]. In order to increase this data bandwidth, we are reducing the stack of protocols which leads to a simpler architecture and a reduced network traversal delay. The stack can be reduced horizontally, i.e., by reducing the number of components to traverse (routers, switches, . . .), but also vertically, i.e., by reducing the number of protocol layers. We will present a further optimization in this paper by introducing Software-Defined Networking (SDN), and integrating it in the mobile architecture.

Several different approaches have been proposed. For instance, the authors of [3], [4] integrate SDN on the radio part. The authors of [5], [6] present a new architecture by integrating SDN in the core network. The authors of [7]–[10] integrate the SDN in the backhaul, between the Evolved NodeB (eNodeB) and the Packet Data Network Gateway (P-GW). In [11], SDN is used in conjuction with the GPRS Tunneling Protocol (GTP) tunnels. The authors of [12] integrate SDN in a new mobile architecture, and the authors

of [13], [14] present several approaches to integrate SDN in a mobile architecture.

Other works are using OpenFlow, an SDN protocol (see Section III), to improve the mobile architecture. In [15], OpenFlow is used to create a new control plane, whereas in [16], OpenFlow is integrated in IP Multimedia Subsystem (IMS), and in [17], OpenFlow is used on the access and aggregation nodes.

In this paper, SDN is proposed in the 4G architecture, and changes are proposed to improve the architecture in order to move towards the new coming 5G. These changes comprise the reduction of the components by removing the Serving Gateway (S-GW), and the reduction of the number of protocol layers. The goal is to improve the 4G Long-Term Evolution (LTE) architecture and to make it possible to virtualize the network. By introducing SDN, the virtualization is made possible and the routes can be optimized. For instance, the Quality of Service (QoS) can be handle optimally by setting specific rules in the switches along the data path.

The paper is organize as follows. The next section presents the 4G LTE architecture and describes the attach procedure of a terminal. Section III explains the concept of SDN and presents the OpenFlow protocol. Section IV presents the integration of SDN into the mobile architecture. Section V presents the migration of GTP messages to OpenFlow messages for the new presented architecture. The last section presents the conclusion and future works.

II. THE 4G LTE ARCHITECTURE

The main functional entities of the 4G LTE architecture are presented on Fig. 1.

It can be observed that the architecture is divided in two sets: 1) the Evolved Universal Terrestrial Radio Access Network (E-UTRAN) from LTE and 2) the Evolved Packet Core (EPC) from System Architecture Evolution (SAE). Together they comprise the Evolved Packet System (EPS). The E-UTRAN is the radio access network and EPC is the core network. More precisely, E-UTRAN is essentially made of one node, the eNodeB (i.e., the antenna) and the EPC consists in four main components, the Mobility Management Entity (MME), the Home Subscriber Server (HSS), the S-GW, and the P-GW. The MME is responsible of the mobility and the network setup procedures in the control plane. The S-GW is serving several eNodeBs on the user plane to connect them to the P-GW. The P-GW is the anchor point for reaching the Packet Data Networks (PDNs). The HSS is the users database providing user information to the MME in order to authenticate the users. Further details and explanations can be found on [18], [19].
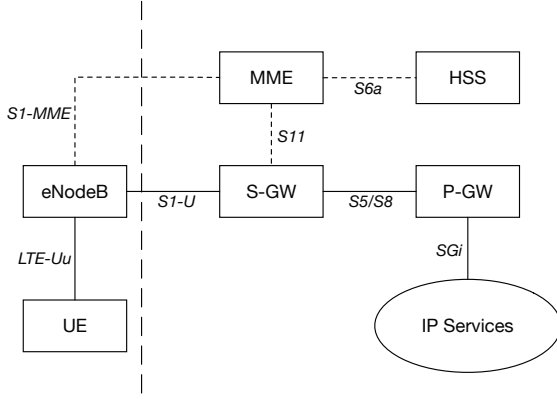
Fig. 1. Main Components of the 4G LTE Architecture

Fig. 3. 4G LTE User Plane Protocols Stack

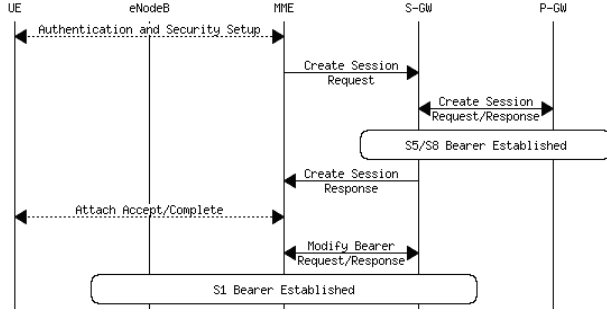Fig. 4. OpenFlow Controller and Switches

Fig. 2. 4G LTE Attach Procedure

When an arriving User Equipment (UE) roams in the network, the eNodeB it is connected to contacts the MME. The MME authenticates the UE with the help of the HSS and setups the bearers by exchanging control messages with the S-GW and the eNodeB. The steps involved in the attach procedure are presented on Fig. 2. More precisely:

1) UE sends the attach request to the MME through the eNodeB.
2) The MME authenticates the UE using the HSS.
3) The MME selects a S-GW and forwards to it the attach request.
4) The S-GW creates a new bearer entry in its table. It sends its Tunnel Endpoint Identifier (TEID) to the P-GW.
5) The P-GW creates also a new bearer entry and sends back its TEID in order to finish the creation of the S5/S8 bearer.
6) The S-GW creates an other bearer entry (for the eNodeB) and sends its TEID to the eNodeB through the MME.
7) The MME sends to the UE through the eNodeB the attach accept response.
8) The eNodeB sends back to the S-GW through the MME its TEID to finish the creation of the S1 bearer.

Once the attach procedure is complete, the bearers are established and the UE is connected to the network. Note that the network is IP-based from one end to the other, and uses GTP tunnels from the eNodeB to the P-GW. Those tunnels are the underlying technology used to implement the bearers.
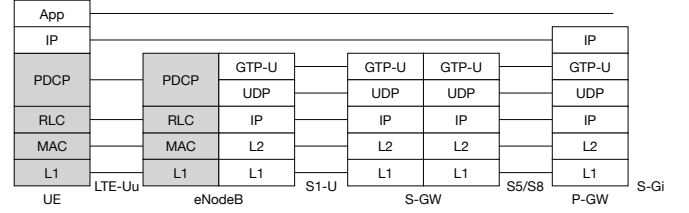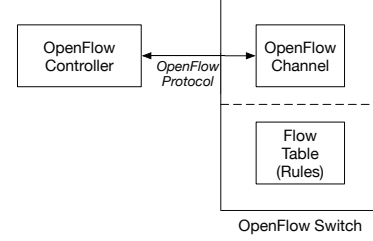
Finally, the user plane protocols stack is detailed on Fig. 3. It can be observed that the extensive amount of protocols in the stack introduces overhead processing delays. By reducing this number the processing time can be reduced. Therefore, the GTP tunnels, which are implemented using IP-in-IP encapsulation are no longer needed.

III. SDN AND OPENFLOW

A. SDN Architecture

SDN [20] is a radically novel approach that decouples the control plane (routing logic) from the data plane (packet forwarding).

The Open Networking Foundation (ONF) has published a white paper [21] of SDN and [22] presents a survey of the past, the present, and the futur of SDN. In a classical network approach, the control plane is included in every router and acts independently, i.e., each router makes a decision in order to forward the packets. A distributed algorithm is implemented and, asymptotically, it converges.

On the opposite, in SDN, a single controller will centralize all relevant information (e.g., the network topology) in order to make routing decisions. Therefore, the SDN switches will not take any decision, but will be told by the controller on how to forward the packets.

Noticeably, SDN is an architecture and does not define how the controller communicates with the SDN switches. For instance, OpenFlow [23] is a protocol defining this exact behavior. In OpenFlow, the network is composed by OpenFlow switches (instead of routers) and an OpenFlow controller (see Fig. 4). The switches have flow tables where each entry defines a matching rule for every incoming packets and a set of actions to do when there is a match. The flow tables are populated by the controller, either proactively, reactively, or both.

The rules of the flows can match the packets over several fields, e.g., the MAC addresses, the IP addresses, the QoS
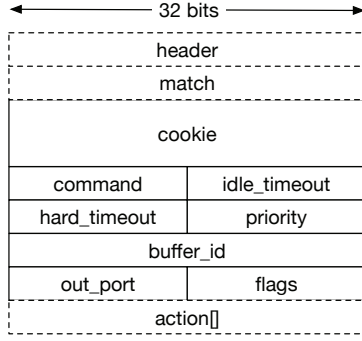
Fig. 5. FlowMod Message Structure (OpenFlow version 1.0)

Fig. 6. 4G LTE User Plane Protocols Stack with OpenFlow

field, the Transmission Control Protocol (TCP) ports, etc. As it will be demonstrated later, the concept of a flow can be used to implement a bearer or a tunnel. Based on a set of matching rules injected in all switches, a flow forwards a packet from one ingress point to the egress point in the topology. This process is very flexible and is entirely defined by the central controller which, in turn, allows a global and flexible optimization of the entire network.

B. OpenFlow Protocols

The OpenFlow Protocol [24] defines several messages to interact with the switches. The creation of the flows is made using the *FlowMod* message. The structure of this message is presented on Fig. 5. The interesting parts are: *match, command, action*.

The *match* part is the information which will be used to match the incoming packets. Among the fields there are the Media Access Control (MAC) source/destination address, the IP source/destination address, the TCP/User Datagram Protocol (UDP) source/destination port, the Virtual Local Area Network (VLAN) identifier, the VLAN Priority Code Point (PCP), and the Type of Service (ToS).

The *command* part defines whether a new flow has to be added, or an existing flow has to be modified or deleted. To add a new flow, the value 0 must be used.

The *action* part is a set of actions to execute when there is a match. A basic action is to output the packet to a specific port.

IV. OpenFlow Integration in a Mobile Architecture

SDN can be integrated in the 4G LTE architecture by means of OpenFlow switches and a controller implementing common IP routing algorithms and protocols. First we start by integrating OpenFlow and removing the S-GW in the architecture then we demonstrate how to improve the architecture towards the 5G.

A. OpenFlow Integration in Legacy 4G LTE Architecture

The Section II introduced the 4G LTE architecture and explained the attach procedure. OpenFlow provides good
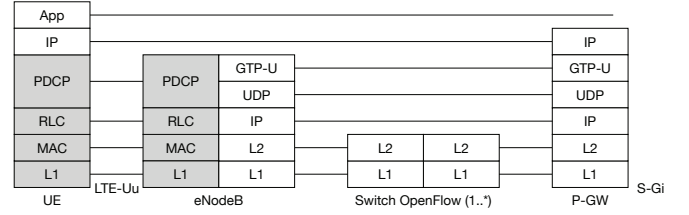
flexibility and good performances [25] but its scalability is still being investigated [26]. The solution proposed in this paper is to integrate OpenFlow in the LTE architecture as follows. First, the S-GW entity is removed from the architecture in order to reduce the stack horizontally. Indeed, its role to connect eNodeBs to S-GWs will be intrinsic to the new architecture. The eNodeB now communicates directly with the P-GW (data plan) over OpenFlow switches (layer 2 switching). In order to provide the needed backward compatibility, the stack is not reduced vertically, i.e., the GTP tunnels are required. Nevertheless, in future architectures, such as 5G, it would be a further opportunity as it will be demonstrated in the next section.

In order to establish a bearer, the request and response messages sent from the MME to the S-GW must be passed to the OpenFlow Controller. This controller then communicates with the P-GW. The two original tunnels (S1 and S5/S8 bearers) are merged into one single flow. The user plane protocols stack becomes as shown on Fig. 6.

To merge the two bearers in a single flow, the TEID of the eNodeB must be mapped to the TEID of the P-GW to make one GTP tunnel. However, regarding the attach procedure steps, it is not possible to map them directly. Indeed, the exchange of TEIDs follows this sequence:

1) The S-GW sends its TEID to the P-GW.
2) The P-GW sends back its TEID.
3) The S-GW sends its TEID to the eNodeB through the MME.
4) The eNodeB sends back its TEID.

Therefore, the OpenFlow Controller acting as the S-GW is not able to send the TEID of the eNodeB to the P-GW and is not able to send the TEID of the P-GW to the eNodeB.

The contribution of OpenFlow is possible but limited in this case since none of the current versions of OpenFlow can match packets based on GTP headers. Therefore, the forwarding decisions cannot be further optimized to avoid the tromboning[1] effect. The integration of OpenFlow is nevertheless possible by keeping the S-GW while using OpenFlow switches as hardware components.

B. OpenFlow Integration towards 5G Architecture

In the previous section, the stack has been reduced horizontally by removing the S-GW, but could not be reduced

[1]Tromboning is when a media traffic originating at a certain point, follows a path out into the network and back to a destination close to the originating point, leading to an inefficient routing.
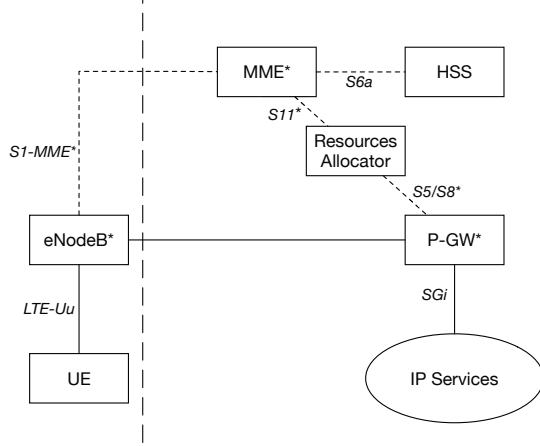
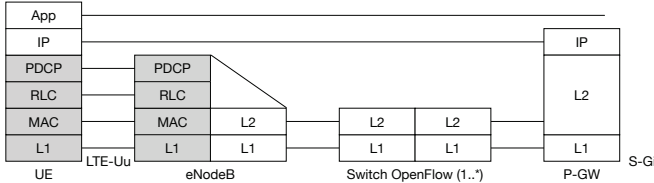Fig. 7. Proposed 5G Architecture with Fast Layer 2 Switching (i.e., Open-Flow)

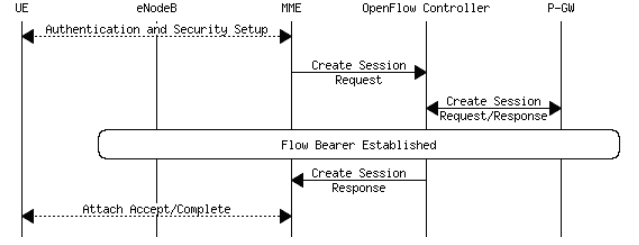
Fig. 8. 5G User Plane Protocols Stack with OpenFlow

Fig. 9. 5G Attach Procedure using OpenFlow

between the eNodeBs and the P-GWs.

The attach procedure is modified and is presented on Fig. 9. More specifically, the new steps are:

1) UE sends the attach request to the MME through the eNodeB.
2) The MME authenticates the UE using the HSS.
3) The MME forwards the attach request to the OpenFlow controller.
4) The OpenFlow Controller sends to the P-GW information about the UE.
5) The P-GW sends back the IP address of the UE.
6) The OpenFlow controller computes a route from the UE to the P-GW and add flows into the OpenFlow switches.
7) The MME sends to the UE through the eNodeB the attach accept response.

The OpenFlow controller must only be traversed by some of the packets on the signaling part in order to create the flows. When the flows are created, the packets are directly going from the eNodeB to the P-GW.

C. Advantages of OpenFlow Integration

So far, we have seen three different architectures (4G LTE + OpenFlow, 4G LTE without S-GW + OpenFlow, 5G without S-GW + OpenFlow) where the second one is impossible to achieve. The advantages of each architecture are:

4G LTE + OpenFlow	5G without S-GW + OpenFlow
Virtualization	Virtualization
Network flexibility	Network flexibility
Easy deployment	Easy deployment
	Protocols stack reduced
	Possible route optimization
	No S-GW to traverse
	QoS can be proactive

V. MIGRATION FROM GTP S11 MESSAGES TO OPENFLOW

In this section, a practical example is explained to show the migration from the GTP bearers establishment to the OpenFlow flows creation. The 4G architecture example is presented on Fig. 10. On this figure, the S1 and S5/S8 bearers are represented (data path). These bearers are implemented by means of GTP tunnels. In order to create those tunnels, the messages exchanged between the MME, the S-GW and the P-GW are using the protocol GTP-C (control plane).

The protocol GTP-C defines several messages. Among them, there are the *Create Bearer Request/Response*

vertically due to backward compatibility. In a more advanced architecture, such as the forthcoming 5G, we foresee that it will be required to reduce the stack vertically by removing the GTP tunnels and performing pure layer 2 switching, i.e., by setting up flows with the OpenFlow Controller to the different entities. As it will be demonstrated, these flows play an equivalent role to the GTP tunnels found in 4G.

Our novel architecture is presented on Fig. 7. Note the introduction of a new entity: the *Resources Allocator* (playing the role of the Policy Decision Function (PDF) in IMS) which can be in our case an OpenFlow Controller. Note also that the stared-entities (with an asterisk) are the same as in 4G but without the GTP part. More precisely, the eNodeB and the P-GW are not creating any tunnel, nor encapsulating packets inside one and the MME is not using TEIDs in its messages. The interfaces S1-MME*, S11* and S5/S8* are not using TEIDs anymore and GTP Control (GTP-C) is not mandatory for the last two.

The new stack of protocols using OpenFlow is presented on Fig. 8. It can be observed that the number of protocol layers is significantly reduced by two levels in the core network, hence improving the network traversal time and the end-to-end delay of all communications.

At the registration of a UE, the OpenFlow Controller is contacted by the MME to create the adequate flows (corresponding to the bearers) on the OpenFlow switches located
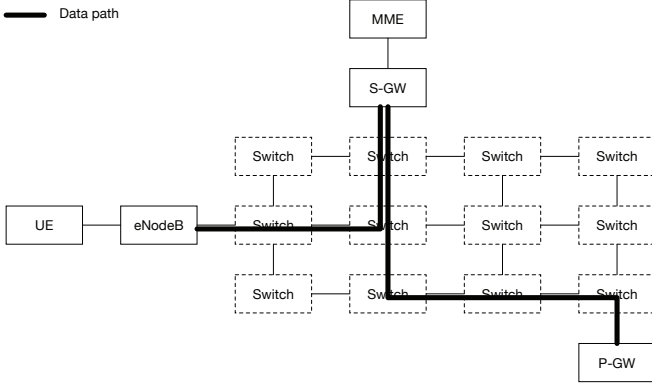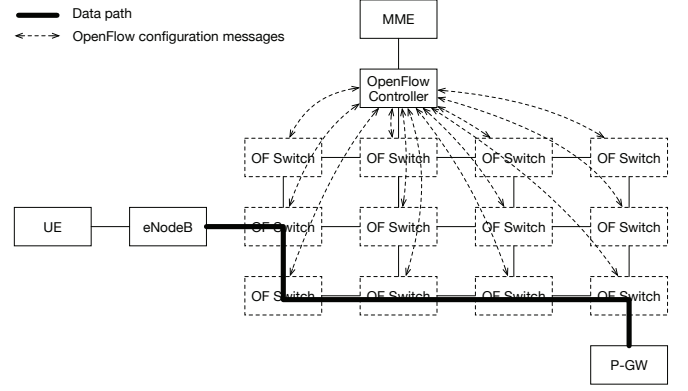
Fig. 10. 4G LTE Example

Fig. 11. The New 5G EPC Example Including OpenFlow

(*Create Session Request/Response*), *Modify Bearer Request/Response*, *Delete Bearer Request/Response* (*Delete Session Request/Response*).

A. Create a Bearer (Session)

The different steps and exchanged messages to create a bearer (when creating a session) are describe in the following:

1) On the request made by the UE, the MME assigns an EPS bearer ID and selects the P-GW.
2) The MME sends a *Create Session Request* to the S-GW with the following information:
   - International Mobile Subscriber Identity (IMSI)
   - EPS bearer ID
   - P-GW IP
   - Access Point Name (APN)
   - Subscribed Profile (received by the HSS)
   - E-UTRAN Cell Global Identifier (ECGI)
   - Tracking Area Identity (TAI)
3) The S-GW allocates the S5/S8 S-GW TEID.
4) The S-GW forwards the *Create Session Request* to the P-GW with the following information:
   - IMSI
   - EPS bearer ID
   - S5/S8 S-GW TEID
   - APN
   - Subscribed Profile (received by the HSS)
   - ECGI
   - TAI
5) The P-GW allocates an IP address for the UE, determines the policies by exchanging messages with the Policy and Charging Rules Function (PCRF) and enforces them.
6) The P-GW sends back to the S-GW a *Create Session Response* with the following information:
   - UE IP address
   - EPS bearer ID
   - S5/S8 P-GW TEID
   - Authorized QoS Profile
   - Traffic Flow Template (TFT)
   - Change Reporting Action

7) The S-GW forwards this response to the MME with the following information:
   - UE IP address
   - EPS bearer ID
   - S5/S8 P-GW TEID
   - S1 S-GW TEID
   - Authorized QoS Profile
   - TFT
   - Change Reporting Action
8) The MME sends to the eNodeB an *Initial Context Setup Request* with the following information:
   - UE-Aggregate Maximum Bit Rate (AMBR)
   - e-Radio Access Bearer (E-RAB) ID
   - E-RAB QoS
   - S1 S-GW TEID
   - K_{eNodeB} (generated by MME from K_{ASME}, and used by eNodeB for derivation of Access Stratum (AS) security keys)
   - UE Security Algorithm
   - Non-Access Stratum (NAS) message (*Attach Accept* for the UE)
9) The eNodeB responds to the MME with an *Initial Context Setup Response* with the S1 eNodeB TEID.
10) The MME sends a *Modify Bearer* to the S-GW with the S1 eNodeB TEID.

On Fig. 11, the created flows joining the eNodeB to the P-GW are represented (data path). The GTP tunnels created on the previous example are replaced by flows created with the OpenFlow protocol.

The steps introduced in Section V-A for the bearer creation are modified as follows:

1) *Idem.*
2) *Idem* but to the OpenFlow Controller.
3) *Nothing.*
4) *Idem* but from the OpenFlow Controller and without the S5/S8 S-GW TEID.
5) *Idem.* (Enforcement of the policies is different and involves the OpenFlow Controller.)
6) *Idem* but to the OpenFlow Controller and without the S5/S8 P-GW TEID.

6*) [openflow-specific step] The OpenFlow Controller creates flows in the OpenFlow switches from the P-GW to the UE (using the IP addresses) and from the UE to the P-GW. The flows make the connection from the eNodeB to the P-GW using OpenFlow switches connecting the UE to the PDN.

7) *Idem* but from the OpenFlow Controller and without the TEIDs.

8) *Idem* but without the S1 S-GW TEID.

9) *Nothing.*

10) *Nothing.*

The GTP-C messages can be used but with the GTP-C header ignored or stripped. These messages are translated by the OpenFlow Controller into OpenFlow messages. In this example, two flows need to be created (see Section III-B) in each OpenFlow switch in the path from the eNodeB to the P-GW. These flows are defined as follows:

- the *match* part will match the packets based on the UE IP address as source (eNodeB to P-GW) or as destination (P-GW to eNodeB),
- the *command* is *add*,
- the *action* is a set of one action that outputs the packet to the right port.

Note that if the QoS needs to be defined, the action must be changed. The packet is not outputted directly on a port, but is enqueued in a queue (selected from the specified QoS).

The OpenFlow specific step (creating the flows in each switch) may add an additional delay but this delay only impacts the start of the session. When the flows are created and the session is established, no additional delay is suffered besides the switching delays present in the switches. Moreover the creation of the flows can be done while terminating the session creation which leads to the diminution of the delay or to no delay at all (if the flows creation delay is lower than the rest of the session creation delay).

B. Modify a Bearer

To modify a bearer, a GTP-C *Modify Bearer Request* is used to modify the GTP tunnel. In the new architecture, the corresponding flows need to be modified as well. The OpenFlow *FlowMod* message, previously presented, is also used in this case. However, the *command* part is changed to *modify_strict* in order to modify an existing flow. The other parts are completed to reflect the desired changes.

C. Delete a Bearer (or a Session)

To delete a bearer in the new architecture, the corresponding flows must be removed and it is done using also the OpenFlow *FlowMod* message but with the *command* set to *delete_strict*. If the session must be removed in addition to the bearer, the MME must be informed.

VI. CONCLUSION & FUTURE WORKS

The forthcoming 5G architecture is going to a fast switching, peer-to-peer architecture where virtualization can be employed. The SDN concept is a promising and powerful approach in order to achieve the objectives.

In this article, it has been demonstrated how to modify the 4G architecture towards the 5G architecture, by using the SDN concept with OpenFlow as protocol. Likewise, a migration from the bearer based GTP-C messages to the lightweight OpenFlow messages has been demonstrated. The gain, in terms of end-to-end delay reduction comes from a significant reduction of the protocol stack and the use of a layer 2, full switched architecture.

Future works will focus on two aspects. First, the implementation of a real life network including the presented modifications and propositions in order to measure the gain, in terms of delay and session setup time. Second, a performance and benchmark analysis to compare the proposed architecture with the current 4G LTE architecture.

REFERENCES

[1] Divya, A. Kumar, Y. Liu, and J. Sengupta, "Evolution of Mobile Wireless Communication Networks: 1G to 4G," *International J. of Elect. and Comm. Tech*, vol. 1, no. 1, pp. 68–72, 2010.
[2] S. Hossain, "5G Wireless Communication Systems," *American Journal of Engineering Research (AJER) e-ISSN*, pp. 2320–0847, 2013.
[3] H. Cho, C. Lai, T. Shih, and H. Chao, "Integration of SDR and SDN for 5G," 2014.
[4] A. Gudipati, D. Perry, L. E. Li, and S. Katti, "SoftRAN: Software Defined Radio Access Network," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 25–30.
[5] X. Jin, L. E. Li, L. Vanbever, and J. Rexford, "SoftCell: Taking Control of Cellular Core Networks," *arXiv preprint arXiv:1305.3568*, 2013.
[6] ——, "Softcell: Scalable and Flexible Cellular Core Network Architecture," in *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. ACM, 2013, pp. 163–174.
[7] J. Costa-Requena, "SDN Integration in LTE Mobile Backhaul Networks," in *2014 International Conference on Information Networking (ICOIN)*. IEEE, 2014, pp. 264–269.
[8] L. E. Li, Z. M. Mao, and J. Rexford, "Toward Software-Defined Cellular Networks," in *2012 European Workshop on Software Defined Networking (EWSDN)*. IEEE, 2012, pp. 7–12.
[9] T. Mahmoodi and S. Seetharaman, "On Using a SDN-based Control Plane in 5G Mobile Networks," in *Wireless World Research Forum, meeting*, vol. 32.
[10] X. Jin, L. E. Li, L. Vanbever, and J. Rexford, "CellSDN: Software-Defined Cellular Core Networks," *Open Networking Summit (Research Track)*, April 2013.
[11] B. Liu, "Software Defined Networking and Tunneling for Mobile Networks," Ph.D. dissertation, KTH Royal Institute of Technology, February 2012.
[12] K. Pentikousis, Y. Wang, and W. Hu, "Mobileflow: Toward Software-Defined Mobile Networks," *Communications Magazine, IEEE*, vol. 51, no. 7, pp. 44–53, 2013.
[13] M. Yang, Y. Li, L. Hu, B. Li, D. Jin, S. Chen, and Z. Yan, "Cross-Layer Software-Defined 5G Network," *Mobile Networks and Applications*, pp. 1–10, 2014.
[14] S. Tomovic, M. Pejanovic-Djurisic, and I. Radusinovic, "SDN Based Mobile Networks: Concepts and Benefits," *Wireless Personal Communications*, vol. 78, no. 3, pp. 1629–1644, 2014.
[15] S. Ben Hadj 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 *2013 IEEE 2nd International Conference on Cloud Networking (CloudNet)*. IEEE, 2013, pp. 205–209.
[16] C. Tranoris, S. Denazis, N. Mouratidis, P. Dowling, and J. Tynan, "Integrating OpenFlow in IMS Networks and Enabling for Future Internet Researchand Experimentation," in *The Future Internet*, ser. Lecture Notes in Computer Science, A. Galis and A. Gavras, Eds. Springer Berlin Heidelberg, 2013, vol. 7858, pp. 77–88. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38082-2_7

[17] D. P. Venmani, D. Zeghlache, and Y. Gourhant, "Demystifying Link Congestion in 4G-LTE Backhaul using OpenFlow," in *2012 5th International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE, 2012, pp. 1–8.

[18] T. Ali-Yahiya, *Understanding LTE and its Performance*. Springer Science & Business Media, 2011.

[19] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-Advanced for Mobile Broadband*. Academic press, 2013.

[20] K. Ahokas, "Software-Defined Networking." [Online]. Available: http://kimia.fi/papers/sdn.pdf

[21] O. N. Foundation, "Software-Defined Networking: The New Norm for Networks," *ONF White Paper*, 2012.

[22] B. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *Communications Surveys Tutorials, IEEE*, vol. 16, no. 3, pp. 1617–1634, Third 2014.

[23] 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, 2008.

[24] OpenFlow Switch Specification, "Version 1.0.0 (Wire Protocol 0x01)," December 2009. [Online]. Available: http://archive.openflow.org/documents/openflow-spec-v1.0.0.pdf

[25] A. Bianco, R. Birke, L. Giraudo, and M. Palacin, "Openflow Switching: Data Plane Performance," in *2010 IEEE International Conference on Communications (ICC)*. IEEE, 2010, pp. 1–5.

[26] F. Benamrane, M. B. Mamoun, and R. Benaini, "Short: A case study of the performance of an openflow controller," in *Networked Systems*. Springer, 2014, pp. 330–334.