SDN-based Local Breakout for Mobile Edge Computing in Radio Access Network

Yifan Yu
Intel Labs China
Beijing, P.R.China
Email: yifan.yu@intel.com

***Abstract*—The local breakout (LBO) is one of the crucial functionalities in the Mobile Edge Computing (MEC) infrastructure. It enables the traffic offloading in the radio access network to reduce end-to-end latency and save core network load. One of the essentials regarding LBO in MEC is to allow the traffic steering in diverse granularity without impacting the ongoing traffic. We present the LBO solution based on the Software-defined networking (SDN) approach to meet the requirement. The proposed solution is evaluated in the testbed built via extending the Open vSwitch (OVS). The experimental results measured in the emulation of the Long Term Evolution (LTE) network disclose that our solution outperforms the existing MEC platform known as Network Edge Virtualization Software Development Kit (NEV SDK) by $10\% - 20\%$ in the average packet processing latency. The additional features including the QoS provisioning and the charging are supported in our solution as well.**

I. Introduction

Today's mobile network is compelled to evolve in the innovative ways to tackle the increasing traffic volume stemming from diverse domains in 5G era. Mobile Edge Computing, which is currently known as Multi-access Edge Computing (MEC) [1], is the promising way to tackle the challenge in optimizing the overall user experience. MEC provides IT and cloud-computing capabilities within the RAN in close proximity to mobile subscribers to enable the significantly reduced end-to-end latency and the alleviated network congestion.

The local breakout (LBO) [2] gains the significant importance in the service delivery with MEC. It is a technique for the traffic offloading in the RAN. As specified in the 3rd Generation Partnership Project (3GPP) standard [3], LBO has the specific IP packets bypass the mobile core network and offloaded to the local network within the user plane (UP) built with GTP-U protocol. European Telecommunications Standards Institute (ETSI) also specifies that the MEC should support the LBO in the diverse granularity including the IP packet and the Packet Data Network (PDN) connection [4].

The LBO should not only handle the traffic routing but also take into account the critical functions such as policy, charging and rating for offloaded traffic. Both of 3GPP and ETSI specify that the LBO for MEC should support the tunable traffic offloading. In particular, the external application can tune the policy of traffic offloading through the control plane (CP) of the 5G network in the light of its service requirement.

It is promising to leverage the Software-defined networking (SDN)-based approach [5] to implement the LBO in MEC due to SDN's capability in programmatic initialization, control, change, and dynamic management of network behavior via open interfaces. To the best of our knowledge, there is little effort in the current SDN approaches focusing on the LBO for the MEC in the 3GPP network.

The remainder of the article is organized as follows. The following section elaborates on the overview of LBO in MEC by briefly reviewing the specifications by European Telecommunications Standards Institute (ETSI) and 3GPP as well as the relevant efforts taken in the industry field. Then the solution to implement the LBO based on the Open vSwitch (OVS) [6] is presented in section III. Following that the experimental results obtained from the testbed are discussed. Finally, we conclude the article.

II. Related Work and Motivation

A. Related Work

In ETSI's specification relevant to MEC [4], the LBO is conducted in Data Plane following the steering policy notified by the Mobile Edge Platform. The LBO triggered by the local service delivery has no impact on the core network. The LBO policy is defined as the traffic filters at the PDN and the packet levels. The PDN filters are based on the Subscriber Profile ID (SPID), Quality Class Indicator (QCI) and Allocation Retention Priority (ARP). The packet filters are based on the 3-tuple (UE IP address, network IP address and IP protocol). The additional filters may be specified in future releases of the initiative.

3GPP specifies that 5G core network should introduce the UP Function (UPF) close to the UE to support the MEC [3]. An functionality termed as "UL CL" (Uplink classifier) can be inserted in the UPF of a PDU session for LBO following the traffic filters provided by the CP. The insertion and removal of an UL CL can be conducted either during or after the PDU session establishment. The traffic filters in UL CL may be based on the UE's subscription data, UE location, the information from Application Function (AF), policy or other related traffic rules.

In [7], an LBO system in base station with MEC infrastructure is proposed. It is built on the commercial LTE based testbed showing that about 58% LBO ratio is properly enabled. The system should be configured with predefined information of IP address and port number requested to breakout.

978-1-5386-1734-2/18/$31.00 ©2018 IEEE

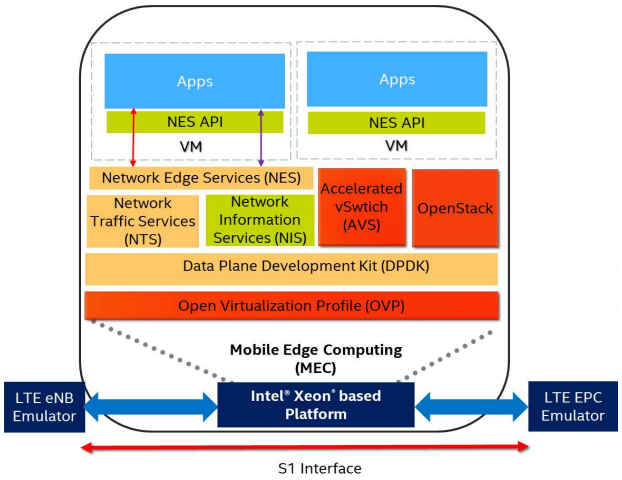

Fig. 1: NEV SDK Architecture

The NEV SDK is released to assist application development based on MEC [8]. It runs on the general purpose multi-core processor enabling a scalable solution for different cellular network deployment scenarios including small cells, aggregation sites, macro cells, and Cloud radio access network (C-RAN). As in figure 1, the LBO is conducted by the NTS which should follow the predefined routing policy prior to the MEC platform launching. The inter-VM communication is enabled by the AVS which supports the on-the-fly routing policy update.

OVS [6], one of the popular implementation of SDN approaches, is a virtual switch enabling the communication between the virtual machines. The incoming packets retrieved from the Network Interface Card (NIC) are processed with the flow entries configured from the OpenFlow [5] controller to the OVS. Matching packets are routed and forwarded to the proper outgoing interfaces. The flow entry is composed of two components: matching field and action. The matching field can be treated as the traffic filter at the packet level, while the action indicates how the packet should be handled. OVS relies on tuple search involving a chain of flow entries that belong to the diverse flow tables to filter the matched packet. To enable the efficient filtering, OVS first attempts to find an exact match for the incoming packets with the cached flow entry. It is the fastest way to determine what to do with an incoming packet in OVS. Given the unmatched packets in the first attempt, they are passed along to a userspace daemon for wildcard matching.

B. Motivation

Both of the specifications from ETSI and 3GPP raise the requirements on the LBO for MEC as following.

- **Flexibility:** The policy for LBO should be tunable by the external application in diverse granularity.
- **Programmability:** The LBO steering should be on-the-fly. In other words, there should be no interruption on the normal service session through the mobile core given the update on the LBO policy.

It is intuitive to leverage OVS to address the issue of flexibility. However, the existing SDN approach can hardly be used for LBO in MEC because it solely supports the flexible routing at the packet level. For example, given the LBO policy oriented to the specific APN (Access Point Name), the OVS is unable to identify the matched packet because APN is notified in the network signaling delivered in the CP rather than the packets transported in the UP. On the other hand, the packet level identifier (i.e. GTP TEID) associated with APN is unavailable in the external application and it is even unknown to the RAN before the UP establishment. Therefore, it is unfeasible to implement the LBO in MEC simply with the traditional OVS.

Although the solutions both in [7] and NEV SDK enable the LBO at the bearer level, they violates the requirement of on-the-fly programmability. Since the LBO policy must be predefined, there has to be the platform rebooting given the updated LBO policy is meant to be applied to the UP. The service interruption is thereby encountered.

III. SDN-Based LBO Solution

Motivated by the above problems, we propose the SDN-based LBO solution for the MEC platform in which the UP is implemented with the SDN switch. The routing policy in the SDN switch is defined following the context of the signaling message delivered in the CP of the mobile network. On the other hand, thanks to the excellent programmability from SDN, our solution inherently supports the LBO policy update without impacting the ongoing traffic through the mobile core. In other words, our solution supports the LBO policy update without the platform rebooting.

A. System Architecture

In our solution, the SDN controller is extended to be with the build-in signaling parser that is responsible for parsing the signaling in CP. The entire system resides over the interface between the RAN and mobile core wherein both of the UP and the CP for the mobile network are established.

The SDN switch clones each passing packet within the CP and forwards them to the SDN controller. Then the signaling parser in the SDN controller retrieves the context of each UE by parsing the cloned signaling packets.

With the UE context sniffed from the CP, the SDN controller may establish the mapping between diverse identifers in the mobile network. For example, the APN can be mapped into the mapped into the GTP TEID. Since GTP TEID is included in each UP packet, the LBO policy oriented to the specific APN can be generated. Thus, the external application may simply use the APN to set corresponding LBO policy in the UP.

B. Execution in SDN switch

The proposed LBO execution in the UP is implemented with the extended OVS. Since the UP traffic is encapsulated in the GTP tunnel, as in table I, there should be the new matching

TABLE I: Extension on OVS

Matching Field	Description
nw_proto_in_gtp	Network protocol identifier in the packet encapsulated with GTP header
tp_dst_in_gtp	Destination port in the packet encapsulated with GTP header
tp_src_in_gtp	Source port in the packet encapsulated with GTP header
nw_dst_in_gtp	Destination address in the packet encapsulated with GTP header
gtp_flag	Flag in GTP header
gtp_msgtype	Message Type in GTP header
gtp_teid	TEID (Tunnel Endpoint Identifier) in GTP header
Action	**Description**
pop_gtp_header	To remove the GTP header from the packet
push_gtp_header	To add the GTP header to the packet

```
union fields l5_fields;

union fields {
   struct gtp_u_mf gtp_u;
   //gtp_u matching fields
};
```

Fig. 2: New Matching Fields Definition in Enahcned OVS

fields and new action items corresponding to the GTP header introduced in the OVS.

Specifically, with the extended matching fields, the OVS is able to identify the UP traffic following the diverse LBO policies which can be yielded with the traffic filters including the various matching fields. To enable the traffic offloading to the local server, the action for the GTP header pop and push is introduced in the OVS as well. The action of GTP header pop is used to extract the UP traffic over the uplink in the mobile network so that the local server is able to process the data from the UE. Given the local server is to respond to the UE, the action of GTP push is to be applied to its data in the OVS. Then the UP traffic is transported over the downlink of the mobile network and finally reaches the UE.

In our implementation, only the userspace components based on DPDK (Data Plane Development Kit) are updated to support the above extension. Specifically, the *dpif-netdev* component belonging to *ovs-vswitchd* process is extended to support the new actions relevant to GTP-U processing. It is implemented as the header pop/push action corresponding to the GTP-U encapsulation and decapsulation, respectively. Since the matching fields introduced in our proposal reside in the payload of the UDP packet, we define a new structure in *struct flow* to generally denote the matching field in the packet payload as described in figure 2.

Figure 3 illustrates the flowchart for the packet processing inside the enhanced OVS wherein the incoming traffic is firstly identified to be forwarded in the CP or UP. Then the downlink (DL) UP traffic and uplink (UL) UP traffic are captured, while the CP traffic is cloned and forwarded to the SDN controller.

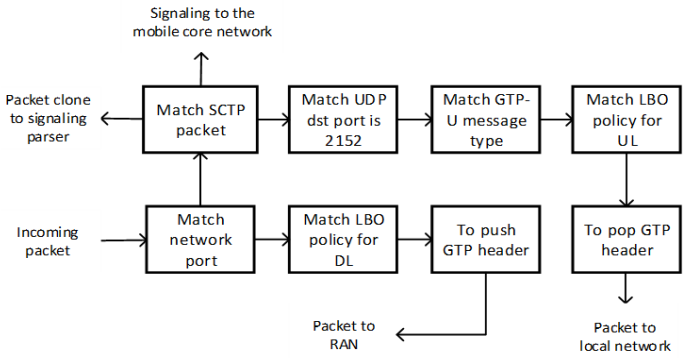

Fig. 3: Pakcet Processing Flowchart in Enahcned OVS

Especially, once the packet is filtered as the UP traffic, the LBO policy can be applied to it. The default action to the UP traffic over UL is to be forwarded to the mobile core.

C. Execution in SDN Controller

The execution in SDN controller should refer to the data recorders associated with the UE-specific context. It is deduced by parsing the signaling messages delivered in the CP. In this paper, we are focused on the S1AP signaling in the LTE network. Specifically, the signaling messages that need to be parsed are as following [10]:

- ***S1*:** E-RAB Setup Response
- ***S2*:** Initial Context Setup Response
- ***S3*:** Path Switch Acknowledge
- ***S4*:** S1AP UE Context Release Complete
- ***S5*:** E-RAB Modify Response

Figure 4 shows the data recorder representing the UE context. Each recorder is indexed with the MME UE S1AP ID and associated with the state which may be either ACTIVE or PENDING. Only the recorder with the state of ACTIVE can be used to generate the traffic filter executed in the SDN switch.

The procedure of processing the LBO request from the external application is following. Given no recorder matching the request, it should be pended until the matched recorder is created. In other words, the SDN controller should check whether there is the pended request for serving upon the creation or the update of the UE context recorder. Given the matched recorder, the GTP TEID is extracted and the LBO request is converted to the packet-level traffic filter where the corresponding GTP TEID is indicated. Finally, the converted traffic filter is notified to the SDN switch.

How the SDN controller updates the data recorder upon the retrieval of the above signaling messages is elaborated as following.

1) S1: This message indicates the bearer establishment in the RAN. If there is the recorder indexed with MME UE S1AP ID same as that in the message, the SDN controller should check whether the recorder is PENDING. If so, the recorder is updated as ACTIVE. Otherwise, the SDN controller should

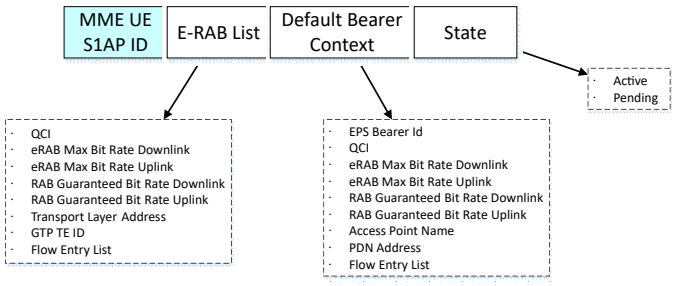

Fig. 4: Data Recorder for UE Context

check whether there is the E-RAB context in the recorder same as that in the message. If so, the message is omitted. Otherwise, the E-RAB context is updated.

2) S2: This message indicates an new UE has attached to the network. If there is the recorder indexed with MME UE S1AP ID same as that in the message, the SDN controller should check whether the recorder is PENDING. If so, the recorder is updated as ACTIVE. Otherwise, the message is omitted.

3) S3: This message indicates the UE has moved to an new eNB. The controller should check whether the new eNB is still covered by the current switch. If so, the message is omitted. Otherwise, the controller should delete the corresponding flow entries in the current switch and add the same flow entries in the new switch covering the new eNB.

4) S4: This message indicates the UE has detached from the network. The controller should delete the recorder indexed with MME UE S1AP ID in the message. Meanwhile, the corresponding flow entries in the switch should be removed as well.

5) S5: The message indicates the bearer is modified. The controller should update the corresponding E-RAB information given the recorder is ACTIVE. Otherwise, it is omitted.

Upon the reception of the LBO policy steering request from the external application, the SDN controller firstly checks whether there is the bearer level keys in the request such as the APN or the E-RAB ID. If the request is at the bearer level, the recorders for the UE context should be checked. Otherwise the request can be converted to the packet-level traffic filter which is then notified to the SDN switch.

The SDN controller should modify the corresponding flow entry in the switch given the update in the UE context recorder. The modification on flow entry includes deletion or addition. The SDN controller should keep the mapping between each bearer and its corresponding flow entry in each UE context recorder. The E-RAB entry in the recorder should keep the list of the yielded flow entries. As for each flow entry in the list, the specific bearer context item such as GTP TEID etc. and the IP address of the configured switch should be kept. Any update in the associated context item may lead to the flow entry update in the switch.

D. Supported Use Cases

The versatile use cases for MEC can be supported with the above extensions. Two examples are herein presented.

1) TCP Optimization: The algorithm in [9] is presented to improve the utilization of the mobile network in the presence of TCP traffic. The TCP server outside of the mobile network is guided for the data transmission with the information on the radio channel capacity inserted into the option field of the TCP packet from the radio network.

With our solution, the TCP packet for the information insertion can be identified with the traffic filter set with "$nw\_proto\_in\_gtp$", "$nw\_dst\_in\_gtp$" and "$tp\_dst\_in\_gtp$". Then the filtered packet is offloaded to the local server in which the radio information is inserted. Next the TCP packet with the radio information is sent back to the OVS which push the GTP header into the packet and further forward it through the mobile core. The TCP packet forwarded back to the mobile core can be identified with the OVS port where the local server connects.

2) DNS Caching: DNS Caching leveraging the MEC platform can reduce the time spent on DNS resolutions. The popular URLs and correlating IP addresses are stored locally. Thus, the webpage load time acceleration and the user experience improvement are achieved due to the short distance that DNS requests travel.

With our solution, the DNS request can be identified with the traffic filter set with "$nw\_proto\_in\_gtp$" and "$tp\_dst\_in\_gtp$". Then the DNS request is offloaded to the local DNS cache after being processed with GTP header pop. The DNS response from the local cache is identified with the OVS port where the DNS cache connects in the OVS. It is returned to the UE after the processing of GTP head push.

E. Impact of LBO Policy Update on Ongoing Traffic

As in section III-C, the flow entry in the OVS can be updated only when the UE context data recorder is changed which is triggered by the signaling messages indicating the change in the ongoing user traffic. The following cases are discussed to assess how the ongoing traffic is affected by the LBO Policy update.

1) UE Mobility: Some of the above signaling messages causing the flow entry update in the switch are generated due to UE's mobility between the eNBs. Given the old eNB and new eNB are covered by the same switch for LBO, our solution takes no modification on the traffic filters in the switch. Given the new eNB is unreachable to the current switch, the controller is to select another switch to conduct LBO. In this case, the ongoing session is to be interrupted. As for the short-lived service such as web browsing, such interruption imposes the trivial effect on the user experience. However, as for the long-lived service such as video streaming, it may lead to user experience deterioration. It can be addressed by establishing the tunnel between the old switch and new switch following the configuration from the controller.

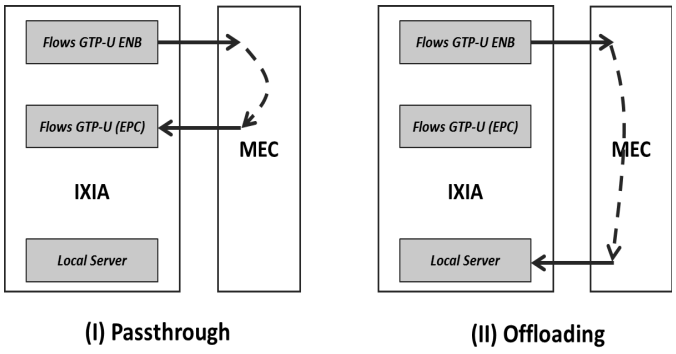

Fig. 5: Test Scenarioes

2) Bearer Modification: The other signaling messages taken into account in our proposal are relevant to the bearer management. Given the new bearer establishment, our solution have the traffic filter added to the switch as long as the CP procedure is completed. At this instant, there is generally no UP traffic yielded by the UE. Therefore, such update has no impact on the user experience. Given the bearer termination, our solution removes the traffic filter in the switch after the completion of CP procedure. In such case, the ongoing traffic has been terminated by the UE.

IV. EXPERIMENTS AND DISCUSSION

A. Testbed Setup

The testbed for the experiments is composed of the X86 server and IXIA's test platform. Table II lists the details of the testbed. The experimental results are measured with the IXIA platform in which the LTE network and the local data network for MEC are emulated. The LBO entity for MEC is built in the X86 server. The UP traffic originates from the IXIA platform and retrieved in the X86 server. After being processed by LBO entity, the UP traffic is sent back to the IXIA platform. Two types of LBO entities are evaluated in the experiment. One is the NTS in NEV SDK, the other is the OVS with the extension proposed in section III.

Two scenarios, as illustrated in figure 5, are taken into account in the evaluation. One is the passthrough scenario wherein the UP traffic is forwarded through the mobile core given no traffic matching. The other is the offloading scenario in which the UP traffic is offloaded to the local network given the traffic matching. The processing latency of each UP packet in the LBO entity, namely X86 server, is measured. The performance comparison between NEV SDK and our proposal are presented in this section.

B. Experimental Results

Since the packet processing is implemented with the software package both in our enhanced OVS and NEV SDK, the measured processing latency is to fluctuate in the certain range. Therefore, the experiment results are collected in three aspects:

TABLE II: Parameters of Testbed

X86 Server	- 2 x Intel(R) Xeon(R) CPUs E5-2680 v3 @ 2.50GHz - 8 x 8GB DDR4 2133Mz - 2 x Intel 82599 10Gigabit Ethernet with SFP - 480GB SSD
IXIA Platform	- IXIA XGS12 [11]
Network Configuration	- 7 Gbps GTP-U traffic - 8 UEs

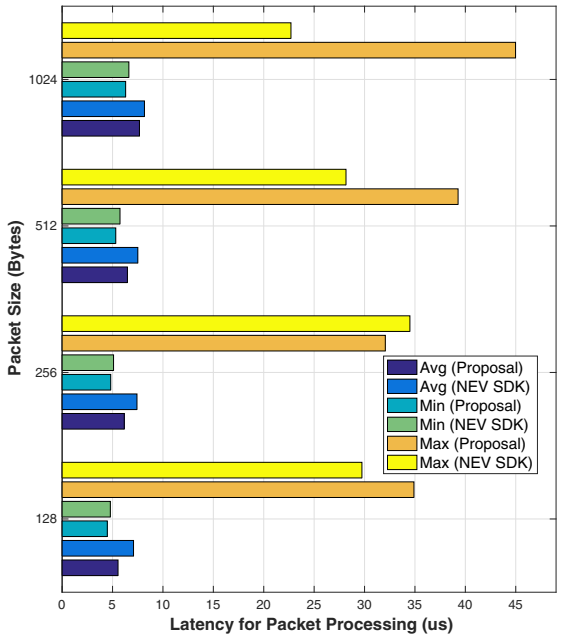

Fig. 6: Latency in Traffic Passing Scenario I

minimum latency, maximum latency and average latency. The IXIA platform emulates UEs and mobile core.

Figure 6 shows the latency measured in the passingthrough scenario. It is observed that the gap between our solution and NEV SDK in terms of the minimum latency is marginal. The maximum latency in the enhanced OVS is higher than NEV SDK. However, our solution achieves the average latency lower than NEV SDK by $10\% - 20\%$. It is additional shown that the larger packet may lead to the longer average latency both in the enhanced OVS and NEV SDK.

The higher maximum latency in our solution is due to the time consumption for the flow fetching caused by the flow cache missing upon the arrival of the first packet. As the first packet is retrieved, the flow cache enabling the fast packet processing is empty. Then the additional lookup occurs in the datapath classifier with the full flow recorders which incurs a performance penalty. However, the processing on the first packet yields the entry in the flow cache. It thereby enables the cache hit in the processing of the sequent packets thereby reducing the time consumption. As for NEV SDK,

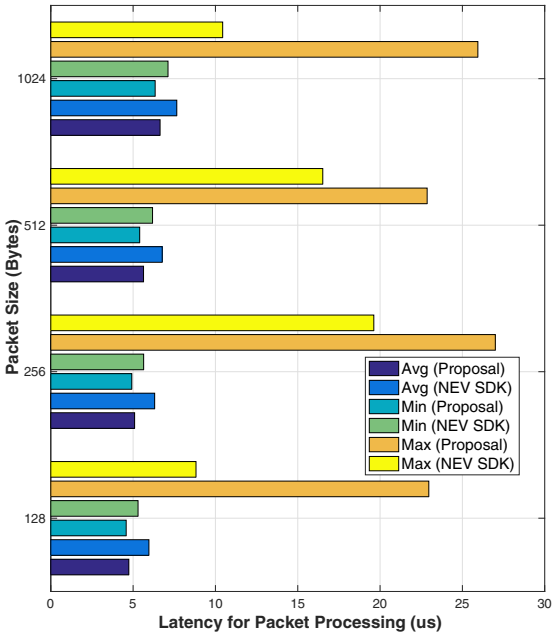

Fig. 7: Latency in Traffic Offloading Scenario II

since it processes the packet with the single classifier, there is no additional lookup which therefore yields the less time consumption in processing the first packet.

The reason for the lower average latency achieved in our solution is that we build the more efficient classifier than NEV SDK. Thanks to the new matching fields dedicated for the UP traffic encapsulated with the GTP header, our solution can identify the packet for LBO without poping the GTP header. On the contrary, NEV SDK has to extract the user packet from the GTP encapsulation prior to classification. It therefore causes the additional latency in forwarding.

The longer average latency in the presence of larger packet is attributed to the more CPU cycles on average spent in the memory manipulation for the packet fetching. There is less variation in the latency for the NEV SDK than our solution. It reflects that the portion of the time consumed by the packet fetching is smaller than our solution.

Figure 7 illustrates the latency results in the traffic offloading. It presents the results similar to those in figure 6. Especially, the gain in the latency reduction amounts to $10\% - 20\%$ in this scenario as well.

It should be noted that our solution allows the on-the-fly UP traffic steering for LBO meanwhile achieving the lower processing latency. It is due to the forwarding actions stepping from the inherent architecture of the classical OVS in which the flow entry cache allows the flow update without impacting other traffic flows.

Additional advantages on our solution are the enhancements on the LBO including QoS provisioning and charging that are currently unavailable in NEV SDK. In particular, these features are enabled without additional efforts on updating OVS.

The QoS provisioning can be implemented with the action of meters supported in OVS. Meters allow for the rate-monitoring of traffic prior to output, thus the LBO entity based on our solution can process the packet in the light of the ingress rate associated with the specific traffic filter. The counters offered in OVS can be leveraged to assist the charging for LBO. With counter associated with the specific traffic filter, the statistics counted as Received Bytes are collected by the SDN controller with the Read-State messages. Thus, the charging on the traffic volume for the specific offloaded traffic flow is enabled by referring to this statistics.

V. Conclusion

This paper investigates the LBO solution in the MEC platform. The requirements on the LBO in MEC are identified via the investigation on the relevant specifications from 3GPP and ETSI. It indicates that the on-the-fly programmability on the UP for the traffic steering is essential to the LBO in MEC. However, the study on the existing industrial LBO approach known as NEV SDK discloses that such requirement is unable to be met. We therefore propose the SDN-based solution via extending the existing OVS. The experimental evaluation is conducted in the testbed with the emulation of practical LTE network. The results disclose that our solution achieves the smaller processing latency in the typical scenario for LBO in MEC as compared with NEV SDK. Meanwhile, the other advantages including on-the-fly programmability in the UP, the QoS provisioning and the charging to the offloaded traffic are available in our solution as well.

References

[1] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta and D. Sabella, "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," in IEEE Communications Surveys Tutorials, vol. 19, no. 3, pp. 1657-1681, 2017.

[2] Y. Ye, N. Vora, J. P. Sairanen and M. Sahasrabudhe, "Enabling local breakout from eNB in LTE networks," 2012 IEEE International Conference on Communications (ICC), Ottawa, ON, 2012, pp. 6982-6986.

[3] 3GPP TR 23.501, Technical Specification Group Services and System Aspects: System Architecture for the 5G System, Stage 2(Release 15), V1.3.0, 2017.

[4] ETSI, Mobile Edge Computing Introductory Technical White Paper, 2014-09-01/2015-10-26.

[5] N. McKeown et al., "OpenFlow: Enabling Innovation in Campus Networks, ACM SIGCOMM Comp. Commun. Review, vol. 38, no. 2, Apr. 2008, pp. 69C74

[6] Open vSwitch, http://openvswitch.org/.

[7] S.-Q. Lee and J.-u. Kim, Local breakout of mobile access network traffic by mobile edge computing, in Information and Communication Technology Convergence (ICTC), 2016 International Conference on, pp. 741C743, IEEE, 2016.

[8] Intel, Intel Network Edge Virtualization, https://networkbuilders.intel.com/network-technologies/nev.

[9] A. Jain et al., "Mobile Throughput Guidance Inband Signaling Protocol", https://www.ietf.org/archive/id/draft-flinck-mobile-throughput-guidance-03.txt

[10] 3GPP TR 36.413, Evolved Universal Terrestrial Radio Access Network (E-UTRAN): S1 Application Protocol (S1AP), 2017.

[11] IXIA, 'XGS12-SD Chassis", https://www.ixiacom.com/products/xgs12-sd-chassis