Improving performance of LTE control plane

Bane Megha Raman[#1], Rane Prajakta Santosh[*2], Bane Raman Raghunath[*3], Warawdekar Bhalchandra C. [*4]

[#]Electronics and Telecommunication Engg. Department , [*]Computer Engg. Department, SSPM CoE, Kankavali.

[1]meghabane2004@gmail.com

[2]praju19978@gmail.com

[3]banerr1999@gmail.com

***Abstract*— This Technological advancement in mobile networks like 4G has given us better quality as well as better bandwidth. This attracted more number of users or customers. But to satisfy the demands of these increased users, it generates higher load on core networks which becomes difficult to handle. It is difficult for mobile operators to scale the current Long Term Evolution (LTE) system resources because of complexity in the architecture and it is not flexible to modify. To address this issue this is proposal to improve the performance of the Evolved Packet Core (EPC) component of LTE network. We took help of Software Defined Networks (SDN) to solve this problem. SDN decompose the mobile data networks into control-plane and data-plane functions. In this paper, we are proposing the SDN-based performance improvement of Evolved Packet Core (EPC) component of Long Term Evolution (LTE).**

The control-plane and data-plane functionalities of EPC components are identified and OpenFlow protocol is used to establish a communication between them. We have implemented the core components of EPC namely, MME, HSS, SGW, PGW as control applications running over SDN controller and data plane of the network is build with Openflow.

***Keywords*— Long Term Evolution, Software Defined Networks**

I. INTRODUCTION

The number of mobile users and electronic gadgets has increased at higher rate due to tremendous improvement in mobile network data technology. It grows the the volume of data traffic. The Telecommunication companies are finding it difficult to balance this load and are looking for new alternatives to overcome the challenge. An increase of this proportion will put a very high load on the radio and packet core networks and will make it difficult for cellular operators to provide and maintain quality service to their customers.

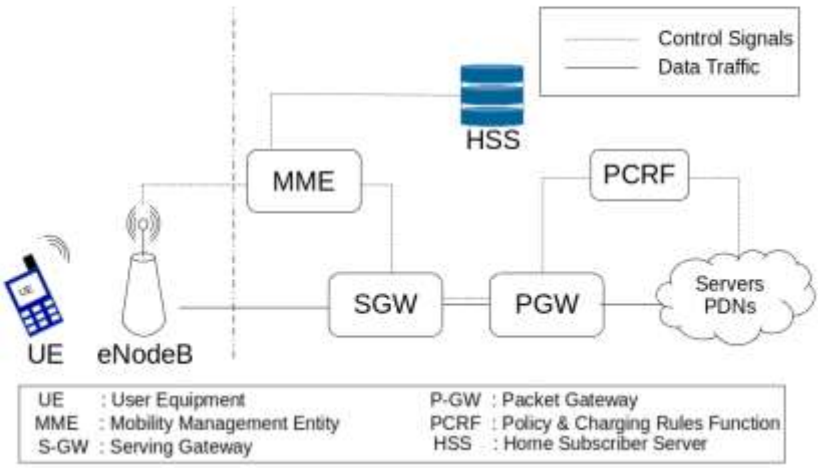

Fig. 1 LTE Architecture

Current architecture of LTE (Long Term Evolution) core packet network is very inflexible and complex, making it difficult to adapt new services [2]. It consists of variety of specialized hardware equipments that have high cost and are generally difficult to upgrade. Also, there is the issue of inter box compatibility, operators cannot mix and match capabilities of different boxes from different vendors. For deploying any new functionality, operators are forced to replace the current system with the new one, even though the system is still capable of serving most of their needs. This factor increases the cost of operating a cellular network and results in low revenue [4]. Figure 1 shows the architecture of typical LTE EPC (Evolved Packet Core).

The EPC is composed of different functions like MME (Mobility Management Entity), HSS (Home Subscriber Server), SGW (Serving Gateway) and PGW (Packet Data Network Gateway). Each of these devices performs lots of complex functions which makes very expensive and inflexible towards any change. Scaling in current architecture is achieved by procuring more expensive and powerful devices. This type of vertical scaling is very inefficient because it forces the operators to replace their entire current equipments, even though they success most of their needs. EPC components are very complex since lots of functionalities packed into a single box. This makes them very inflexible towards any change and requires replacement of current equipment to incorporate new services. Reduced Signalling Overheads: The current architecture of LTE EPC incurs a lot of control signalling among its components. This adds considerable costs in terms of bandwidth and processing time [3].

These shortcomings motivated to improve the performance of the LTE packet core network. Many solutions have been proposed for making it more flexible, scalable and cost effective. Software Defined Networking (SDN) aims to decompose the EPC functionality into control-plane and data-plane functions, enabling cheaper packet gateways in the data plane, and an intelligent core network controller to handle the signalling and management functions. The goal of is to build an SDN-based LTE EPC so that higher data rate can be sustained.

II. THE LTE ARCHITECTURE

The architecture of current LTE and functionalities of its components are described in this section. We then briefly discuss the different procedures involved in the EPC.

The Radio Access Network consists of two components:

- User Equipment: UE represents all gadgets (mobile phones, tablets, laptops, etc) which are capable of services such as voice calls, internet, IP multimedia subsystem, etc.
- eNodeB: Evolved NodeB is the base station responsible for all radio access related procedures with the UE. It is the first hop for the communication from UE to EPC. Some of the functionalities include Radio Resource Management, Mobility Management, EPS bearer setup, encapsulation/decapsulation of user data, encryption /decryption, etc. The eNodeB uses the S1 interface to communicate with the EPC, and the X2 interface to connect with an adjacent eNodeB.
- MME: The Mobility Management Entity is purely a control entity and monitors several eNodeBs in a particular area. At a time, a UE connects to only one MME. The primary functionalities of MME include authentication of UEs with HSS, mobility management (handover and roaming), session management, tracking area update for UEs, selection of P-Gateway and S-Gateway, etc. It communicates with the HSS over the S6a interface to fetch and update UE details.
- S-Gateway: The Serving gateway is the first point of connection to eNodeB for user data transfer. A UE in connected state is always associated with a single S-Gateway at a time. Major functionalities of S-GW include EPS session setup, uplink/downlink data transfer, management of inter-eNodeB handovers, etc. It connects with the MME over the S11 interface.
- P-Gateway: The Packet Data Network Gateway provides connectivity of UEs to a Packet Data Network (PDN). A UE may establish connections with multiple P-Gateways in case it connects simultaneously with multiple PDNs. However, a PGW is always connected with a single SGW. Its functionalities include allocation of IP address to UEs, EPS session setup, routing and forwarding UE data (uplink /downlink), packet filtering, policy charging of UEs with the help of PCRF, etc. PGW connects with the SGW over the S5 interface, and with the PCRF over the Gx interface.
- HSS: The Home Subscriber Server is a central database which contains all information about the users including authentication keys, UE capabilities, current location, etc.
- PCRF: The Policy and Charging Rules Function enforces the policy charging and control decisions for Service Data Flows (SDFs). It informs the PGW about the QoS (Quality of Service) parameters associated with a particular user, both for the default bearer and dedicated bearers.

The connection establishment of a UE (initially disconnected) to a PDN takes place in the following sequence:

1. UE Attach
   - UE Authentication
   - EPS bearer setup
2. Uplink/Downlink UE data transfer
3. UE Detach

A. *UE Attach procedure*

The Attach procedure starts when the user equipment is turned on and it sends an Attach request to MME via eNodeB. But, prior to this, the air interface radio connection is established between the UE and eNodeB. The UE sends its IMSI (International Mobile Subscriber Identity) as the UE ID as well as its network capabilities along with the Attach request. Having received these, the MME proceeds towards authentication and security setup procedures.:

- UE Authentication: During this step, the MME sends an authentication request to HSS, which generates authentication keys of the UE and sends back to MME. After this, some part of the keys is sent to the UE and the remaining is retained.The UE then generates its own authentication keys and authenticates the network by matching its key with that received from the MME. On successful network authentication, it sends its keys to the MME via eNodeB. The MME then authenticates the UE by comparing its retained keys with that received from the UE. Thus, there is a mutual authentication between the UE and MME (i.e. the network). After the authentication is complete, the MME selects encryption and integrity algorithms and corresponding keys, and sends to UE. This ensures secure communication between UE and MME. Simultaneously, the MME sends an Update location request along with UE ID and MME ID to HSS. The HSS updates the tracking area of the UE and responds to MME with the APN (Access Point Name) and the subscriber details (QoS parameters, etc). After all these, the MME proceeds to setup the EPS bearer. Page Style All paragraphs must be indented. All paragraphs must be justified, i.e. both left-justified and right-justified.

- EPS bearer setup: The MME creates an EPS bearer ID and contacts the DNS server to get the IP address of PGW in order to access the APN. Besides, it selects the SGW through which it can connect to the PGW. In this step, a tunnel is setup between eNodeB and the selected PGW via SGW based on GPRS Tunnelling Protocol (GTP). GTP is a UDP/IP based protocol used in LTE for encapsulating /decapsulating user data during the entry/exit to/from the core network. For uplink data, encapsulation takes place at eNode and decapsulation at PGW, whereas for downlink data, encapsulation takes place at PGW and decapsulation at eNode. We have uplink and downlink tunnels for both S1-U and S5 interfaces. A tunnel is formed based on a pair of Tunnel Endpoint Identifier (TEID), each corresponding to one end of the tunnel. A UE may have more than one EPS bearer depending on its subscription. The tunnel setup request is initiated by MME to the SGW. The SGW forwards the request to PGW, which allocates an IP address to the UE and sends it with the tunnel setup response to SGW. Meanwhile, PGW also contacts PCRF to get the actual QoS parameters according to the subscription policy and network condition. SGW forwards the response received from PGW to MME, which then conveys an AttachAccept message along with UE IP address to the UE via eNodeB. During these steps, the SGW, PGW and eNodeB generate their respective TEIDs and exchange with each other and thus, the EPS bearer is setup.

B. Uplink/ Downlink UE data transfer

Once the EPS bearer setup is complete, the UE can send data to the PDN via eNodeB and EPC. For uplink data transfer, we have the following sequence of operations:

- UE to eNodeB: The UE sends IP data packets through the air interface to eNodeB with source IP as the UE IP and destination IP as an IP from the PDN domain (say, www.google.com).
- eNodeB to SGW: The eNodeB encapsulates the received IP packet in a GTP header along with UDP/IP headers, and then forwards to SGW. The outer IP header has source IP as eNodeB IP and destination IP as SGW IP.
- SGW to PGW: The SGW strips off the outer IP header of the received encapsulated packet, adds GTP/UDP/IP headersof its own, and then forwards to PGW. The outer IP header has source IP as SGW IP and destination IP as PGW IP.
- PGWto PDN: The PGW decapsulates the received packet by stripping off the outer headers (IP/UDP/GTP), and then forwards to the PDN.

Similarly, the downlink data transfer takes place from PDN to UE. At every hop from eNodeB to PGW (for uplink) and from PGW to eNodeB (for downlink), appropriate TEIDs are used in the GTP header which are unique for a particular bearer.

C. UE Dettach procedure

In this step [8], the EPS bearer(s) and the state associated with the UE are released. In this thesis, we deal with UE initiated detach, i.e. the Detach Request is initiated by UE. The MME clears the EPS bearer context and forwards the received request to SGW. The SGW in turn, clears the EPS bearer context and forwards the received request to PGW. The PGW clears the EPS bearer context and acknowledges the SGW, which forwards the acknowledgement to the MME. Finally, MME sends a Detach Accept to the UE via eNodeB. Simultaneously, the radio resource is released between UE and eNodeB.

III. SDN BASED EPC

Software Defined Networking decouples the network control functions (i.e. control plane) and the forwarding functions (i.e. data plane) and, thus enables network administrators to program the network in a dynamic and flexible manner [1]. The control plane is built in software as applications running on top of an SDN controller, whereas general purpose switches (instead of proprietary boxes) constitute the data plane. The control plane and data plane communicate with each other via OpenFlow which is a standard protocol and forms the basis of SDN implementation. The controller installs flow rules into the OpenFlow switches according to the enforced policy and thus, the data plane forwarding happens.

A. Proposed Design for SDN based EPC

The actual implementation of SDN-based EPC has been done in accordance with Figure 2. The SDN controller that has been used is Floodlight (Java based) and Open vSwitch (a virtual OpenFlow switch) has been used instead of hardware OpenFlow switches, for the data plane of the Gateways. The database used for HSS is MySQL. Some of the differences from the typical SDN-based architecture are:

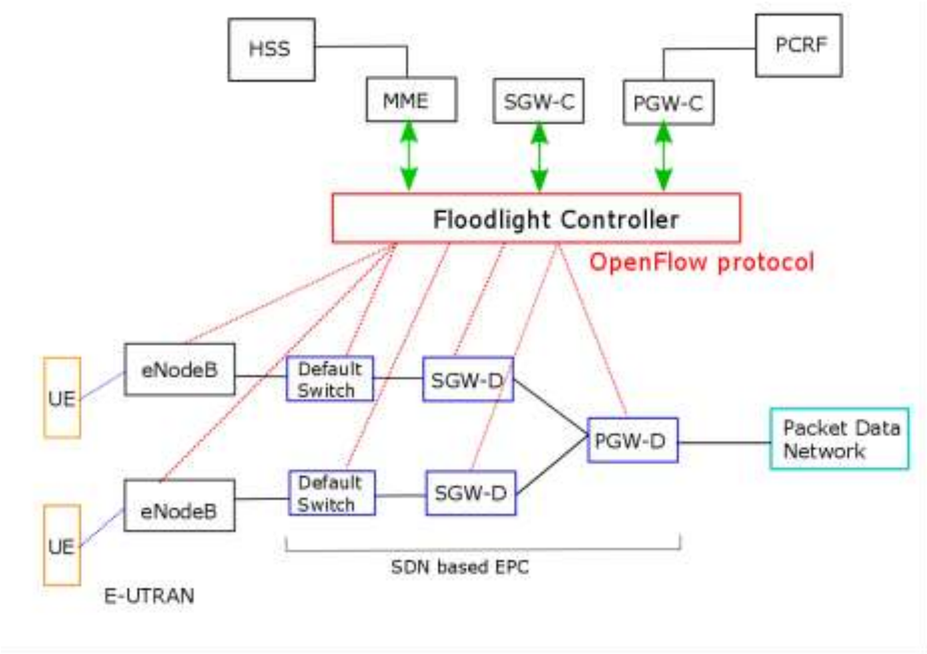

Fig. 2 Proposed SDN based LTE EPC

- The UE and eNodeB have been clubbed together as a single RAN simulator for generating registration requests (i.e. Attach) and data requests. For generating data packets (TCP traffic), we have used iperf3 in the RAN simulator. We do not consider the radio procedures that take place between UE and eNodeB. This has been done since we are concerned with the EPC and not RAN.
- We have a program which represents the PDN and receives/sends UE data packets from/to PGW.
- An additional OpenFlow switch (which we call the default switch) has been used in the data plane between RAN and SGW. This has been done to forward the Attach request from RAN to MME since the RAN itself cannot generate a PACKET IN request to the controller. So, we need an OpenFlow switch that will automatically forward the Attach request to the controller since no flow rule for the UE is installed in the switch initially. The flow rules are installed into this default switch by MME during the EPS bearer setup. Also, having the default switch is computationally equivalent to the typical SDN-based architecture where the eNodeB contains an OpenFlow switch to forward uplink/downlink data.
- One important thing to note is that we have not used the GTP header while encapsulating the UE data packets, since the Open vSwitch (OVS) does not support GTP headers. Instead, we used the VLAN ID field of the Ethernet header (IEEE 802.1Q) as the TEID.

The S-Gateway (control \& data), P-Gateway (control \& data) and Default switch (data) have been implemented in this work.

B. Implementation of S-Gateway and P-Gateway

The functionalities of SGW and PGW are very much correlated with each other. This is the reason we will discuss the procedures as a whole instead of explaining the functionalities of the two separately. The SGW and PGW play a major role in the GTP tunnel setup (i.e. EPS bearer setup), uplink/downlink UE data transfer and UE Detach. We discuss the SDN-based implementation of these procedures in the following sections.

1. EPS Session setup : After the UE is successfully authenticated and registered on the network, the MME initiates the EPS Session setup. The entire procedure of EPS Session setup is depicted in Figure 3. The EPS session setup proceeds as follows:

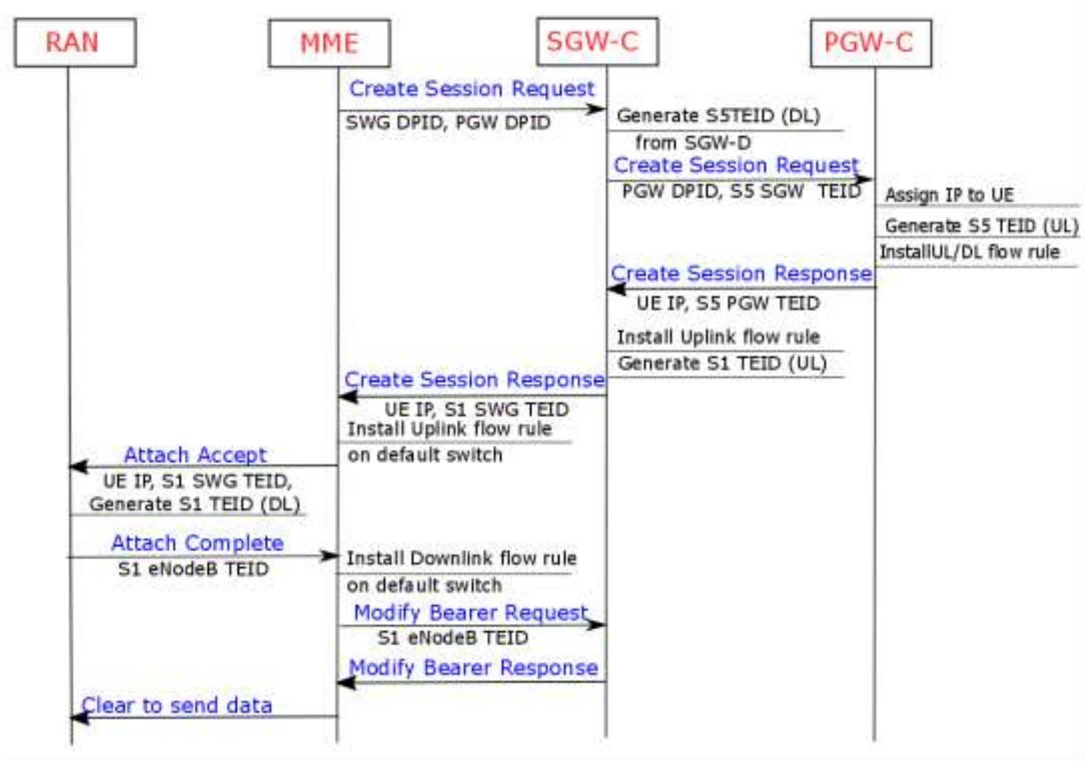

Fig. 3 Procedure for Session Setup

I. The MME selects the SGW and PGW, and sends a Create Session Request to SGWC. It passes the DPIDs (Data Path ID) of the OpenFlow switches corresponding to the selected SGW and PGW along with the request.
II. The SGW-C selects a TEID for the S5 downlink path. It passes the TEID and PGW DPID to PGW-C.
III. The PGW-C selects an IP address from a pool of IP addresses and assigns it to the UE.
IV. PGW-C selects a TEID for the S5 uplink path. Now that PGW-C has both the UL/DL TEIDs, it issues install flow rule command for both Uplink and Downlink path through PGW-D. Besides, it creates a Create Session Response and replies to SGW-C along with the UE IP and the S5 uplink TEID selected.
V. Now, SGW-C has the S5 uplink TEID and hence, issues install flow rule command for uplink path through SGW-D. Besides, it selects a TEID for the S1 uplink path. It passes the received UE IP and the selected TEID to MME as Create Session Response.
VI. The MME sends the received UE IP and S1 uplink TEID to RAN in the Attach Accept message. Meanwhile, it also installs the uplink forwarding rule into the default switch.
VII. RAN selects a TEID for the S1 downlink path and sends it to the MME in the Attach Complete message.
VIII. MME installs the downlink forwarding rule into the default switch and forwards the received S1 downlink TEID to SGW-C in a Modify Bearer Request message.
IX. SGW-C, having received the S1 downlink TEID, issues install flow rule command for downlink path through SGW-D. Finally, it sends a Modify Bearer Response message to MME.
X. The MME sends Clear to send data message to RAN, after which the UEs start generating data traffic.

2. An overview of Uplink/Downlink data transfer
   - Uplink-
     I. RAN: It is responsible for generating TCP data traffic, encapsulating the TCP/IP packets in UDP/IP/Ethernet headers and then, forwarding the encapsulated packets to the Default switch. Encapsulation has been achieved by using an interface, wherein the iPerf client writes data packets on the interface, which is received by a program. The program encapsulates the received IP packet in UDP/IP/Ethernet headers (Outer Src IP = UE IP, Outer Dest IP = PDN IP) and finally, writes the encapsulated packet on the physical interface (eth0) using a raw socket.

II. Default switch: It receives the encapsulated packet, selects a flow rule by matching the outer Src IP, sets the VLAN ID as S1 uplink TEID (specified by the rule) in the packet ethernet header, sets the outer Src IP as the RAN IP (specified by the rule), and then, forwards the packet to SGW-D.
III. SGW-D: It receives the encapsulated packet, selects a flow rule by matching the VLAN ID, sets the VLAN ID as S5 uplink TEID (specified by the rule), and then, forwards the packet to PGW D.
IV. PGW D: It receives the encapsulated packet, strips the VLAN ID, and then, forwards the packet to Sink.
V. Sink: The inner IP packet is received by a tun program using a UDP socket. The tun program writes the packet to the tun interface and the TCP data is received by the iPerf server.

- Downlink-

I. Sink: The iPerf server writes TCP/IP packets on a tun interface, which is received by a tun program. The tun program encapsulates the received IP packet in UDP/IP/Ethernet headers (Outer Src IP = UE IP, Outer Dest IP = RAN IP) and finally, writes the encapsulated packet on the physical interface (eth0) using a raw socket.
II. PGW-D: It receives the encapsulated packet, selects a flow rule by matching the outer Src IP, sets the VLAN ID as S5 downlink TEID (specified by the rule) in the packet ethernet header, sets the outer Src IP as the SINK IP (specified by the rule), and then, forwards the packet to SGW-D.
III. SGW-D: It receives the encapsulated packet, selects a flow rule by matching the VLAN ID, sets the VLAN ID as S1 downlink TEID (specified by the rule), and then, forwards the packet to Default switch.
IV. Default switch: It receives the encapsulated packet, strips off the VLAN ID,and then, forwards the packet to RAN.
V. RAN: The inner IP packet is received by a tun program using a UDP socket. The tun program writes the packet to the tun interface and the TCP data is received by the iPerf client.

3. UE Detach Procedure

Detach procedure in LTE can be UE initiated or MME initiated. Among these, we have implemented the UE initiated Detach procedure. After the UE (thread) completes sending data for the specified duration of time, it initiates the Detach process. The entire procedure for UE initiated Detach is depicted in Figure 4. The steps for Detach procedure are as follows:

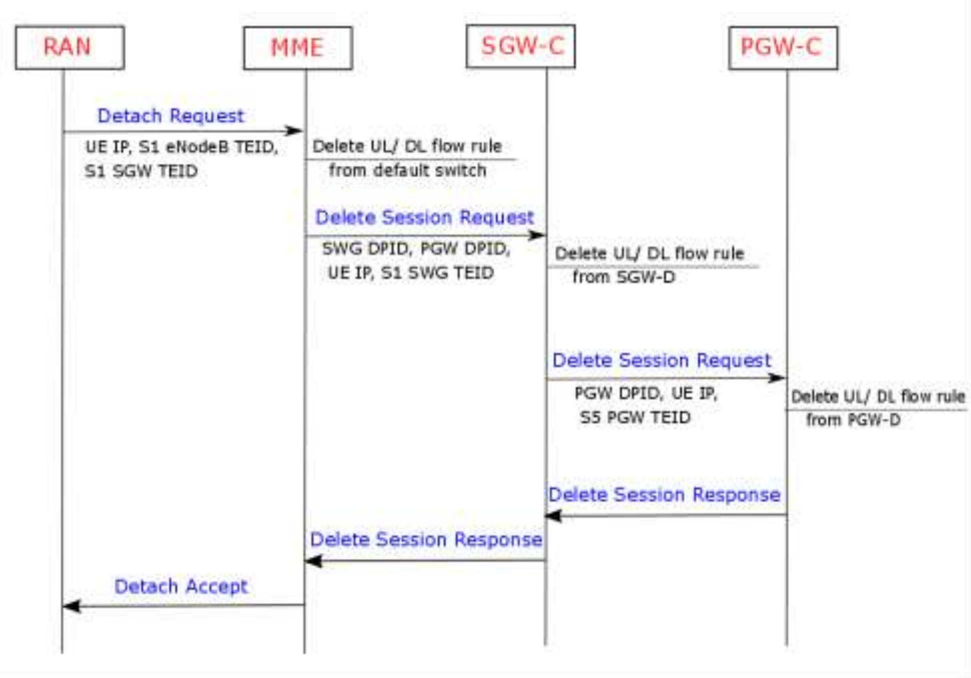

Fig. 4 UE Initiated Detach

IV. EXPERIMENTAL SETUP AND RESULTS

For conducting the experiments, the data plane components of EPC - Default switch, SGW-D and PGW-D, run as Open vSwitch (openvswitch-2.3.2) on different physical machines. The end simulators - RAN and Sink, run on different physical machines. The Floodlight controller (MME, SGW-C and PGW-C) and HSS run on the same physical machine. The common configuration of physical machines used is:

- Processor: Intel Core i5 quadcore processor
- RAM: 16 GB
- Operating System: Ubuntu 16.04 (64 bit)

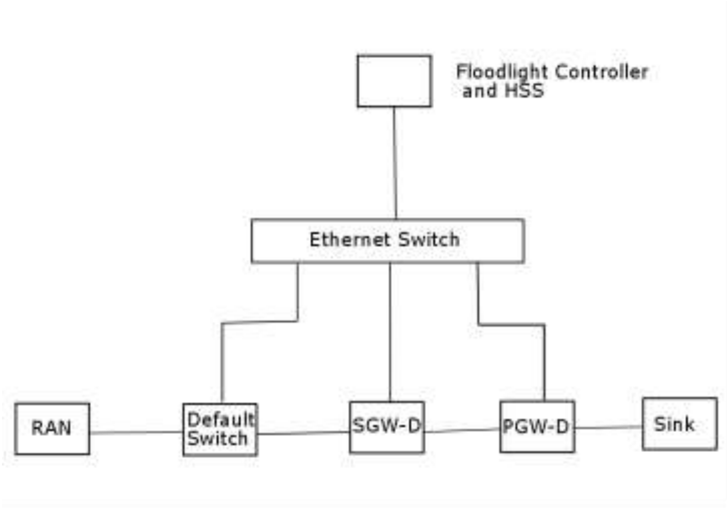

Fig. 5 Experimental Setup

The machines running Default switch, SGW-D and PGW-D have 3 NIC LAN cards (1 Gbps) each, wherein each physical interface has been added as a port to the OVS for connecting them according to EPC setup (see Figure 4). The Default switch, SGW-D and PGW-D are connected to the controller through an Ethernet switch (1 Gbps). Initially, all the OVSes have only a single rule installed by the controller to process ARP messages. For data traffic benchmarking, a certain number of UEs get registered initially and then, send data continuously until the end of experiment. The sequence of procedures that occur are: Authentication/Registration - EPS bearer setup - Data transfer. The entire experiment was run for 300 sec and link speed of 100 Mbps. We conducted two types of experiments: (i) varying the number of concurrent UEs keeping the data rate per UE constant, and (ii) varying the data rate per UE keeping the number concurrent UEs constant. For each of the experiments, we recorded the average data throughput and average latency in terms of round trip time.

A. *With Varying number of UEs:*

In this case, we kept the sending data rate of each UE to be constant at 1 Mbps, and then varied the number of concurrent UEs generating data traffic. The throughput and latency (RTT) are shown in Figure 6 and Figure 7 respectively. We observed that the throughput increases with the increase in the number of concurrent UEs but flattens out when the network becomes the bottleneck. The throughput flattens out close to 90 Mbps because of the overheads of packet encapsulations as well as the physical layer headers. We also monitored the CPU utilizations of Default switch, SGW-D and PGW-D, and they were found to be nominal (20\%). Besides, the utilization of RAN was not bottlenecked. Similarly, the RTT increases with the increase in the number of UEs. There is a sharp increase in RTT as soon as the network starts becoming the bottleneck and, then flattens out when the number of UEs are close to 90.

B. *With Varying data rate per UE:*

In this case, we kept the number concurrent UEs to be constant (i.e. 10), and then varied the sending data rate per UE. The throughput and latency (RTT) are shown in Figure 8 and Figure 9 respectively. Similar to the previous experiment, we observed that the throughput and RTT increase with the increase in the number of concurrent UEs but flatten out when the network becomes the bottleneck. The maximum data rate each of the 10 UEs can get is about 9 Mbps

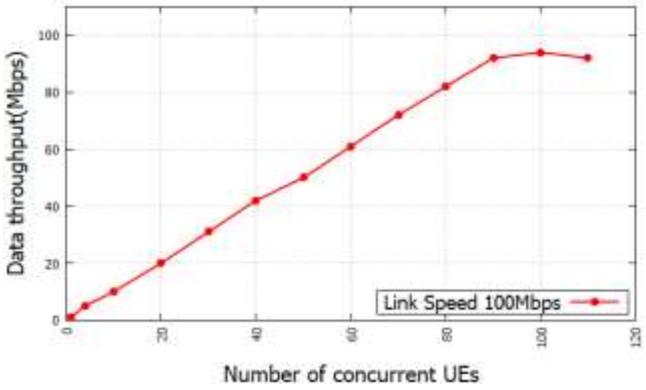

Fig. 6 Throughput vs number of concurrent UEs

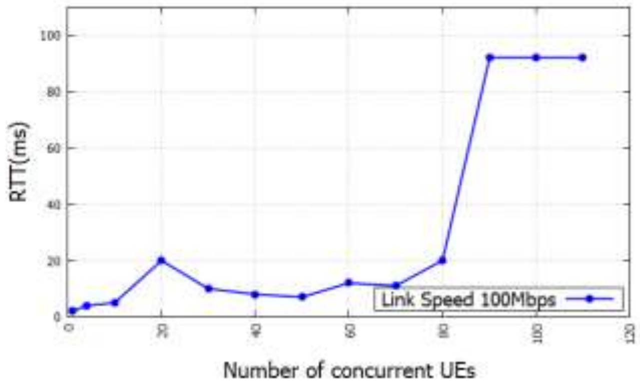

Fig. 7 RTT vs number of concurrent UEs

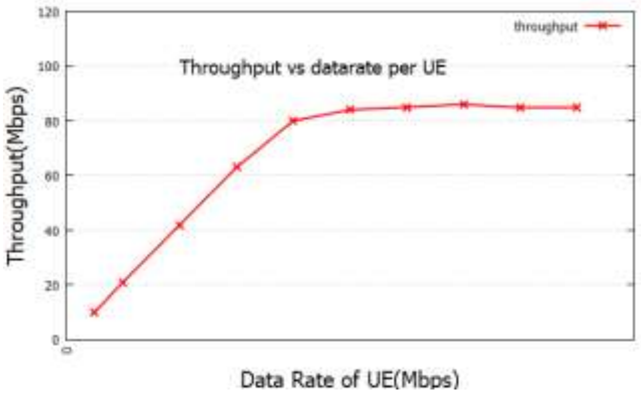

Fig. 8 Throughput vs data rate per UE

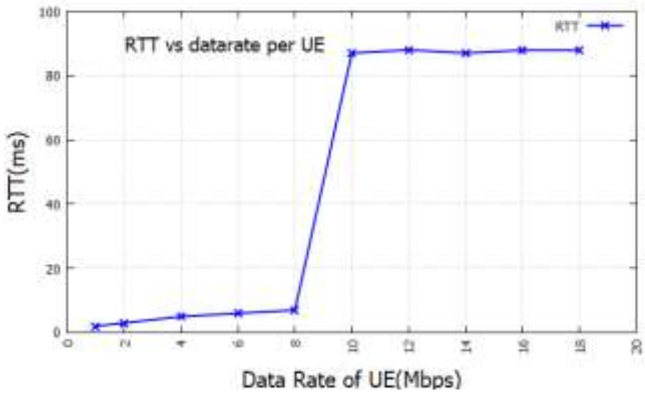

Fig. 9 RTT vs data rate per UE

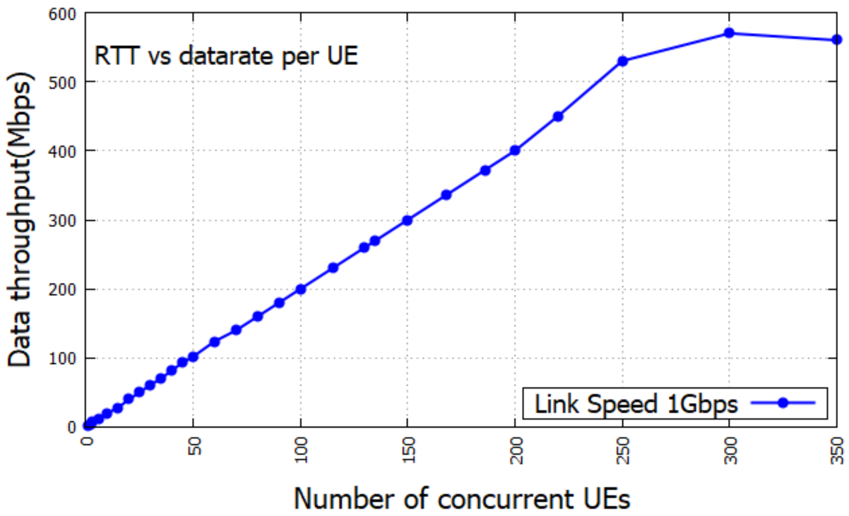

Fig. 8 Throughput vs number of concurrent UEs

C. For 1 Gbps link speed:

We also conducted experiments at 1 Gbps link speed with varying number of concurrent UEs (Figure 10) as well as varying data rate per UE . For the first case, we kept the data rate of each UE to be constant at 2 Mbps, and then varied the number of concurrent UEs. For the second case, we kept the number concurrent UEs to be constant (i.e. 10), and then varied the data rate per UE. Again, we observed that the throughput increases with the increase in the number of concurrent UEs (for first case) and data rate per UE (for second case) but flattens out when the network becomes the bottleneck. We also monitored CPU utilizations of all the systems but none of them were high enough and hence, the network was likely to be the bottleneck. For the first case, the throughput flattens out for about 250 concurrent UEs, whereas in the second case, for a data rate of about 55 Mbps per UE. In both the cases, the maximum throughput achieved is between 500 to 600 Mbps in spite of having 1 Gbps links. This is because of IP-in- IP encapsulations as well as physical layer header overheads. The number of data packets sent by the UEs are large compared to that in case of 100 Mbps links, thus leading to higher overheads.

:

V. Conclusions

We have implemented a prototype of LTE EPC Gateways -S-Gateway and P-Gateway, based on the concepts of SDN. We integrated it with SDN based MME and HSS. For only data traffic the maximum number of concurrent UEs handled by the setup before the network becomes the bottleneck is close to 90 (for 100 Mbps links and 1 Mbps rate per UE) and around 250 (for 1 Gbps links and 2 Mbps rate per UE). For mixed traffic (long duration data), the maximum number of concurrent UEs handled by the setup before the network becomes the bottleneck is about 100 (100 Mbps links, 1 Mbps data rate per UE and data traffic duration of 20 sec).

References

[1] Van Giang Van Giang Nguyen and Younghan Kim, "*Proposal and evaluation of sdn based mobile packet core networks*," EURASIP Journal on Wireless Communications and Networking, 2015(1), 2015.

[2] Basta, Arsany and Kellerer, Wolfgang and Hoffmann, Marco and Morper Hans Jochen and Hoffmann,"*Applying NFV and SDN to LTE Mobile Core Gateways, the Functions Placement Problem*," Proceedings of the 4th Workshop on ALL Things Cellular: Operations, Applications & Challeneges. Pp.33-38

[3] *S. B. H. Said and M. R. Sama and K. Guillouard and L. Suciu and G. Simon and 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), pp. 205-209

[4] L. E. Li, Z. M. Mao, and J. Rexford, "*Toward software-defined cellular networks*," 2012 European Workshop on Software Defined Networking. 2012.

[5] Junguk Cho, Binh Nguyen, Arijit Banerjee, Robert Ricci, Jacobus Van der Merwe, and Kirk Webb, "*Smore: Software-defined networking mobile offloading architecture*," In Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications and Challenges, 2014.

[6] Brent Hirschman, Pranav Mehta, Kannan Babu Ramia, Ashok Sunder Rajan, Edwin Dylag, Ajaypal Singh, and Martin McDonald, "*Highperformance evolved packet core signaling and bearer processing on general-purpose processors*," IEEE Network, vol. 29, no. 3, 2015.

[7] http://www.tutorialspoint.com/lte/lte network architecture.htm

[8] http://www.netmanias.com/en/post/techdocs/6108/detach-emm- lte/emm-procedure-2-detach