Understanding the Bottlenecks in Virtualizing Cellular Core Network Functions

Ashok Sunder Rajan*, Sameh Gobriel*, Christian Maciocco*, Kannan Babu Ramia*,
Sachin Kapur†, Ajaypal Singh†, Jeffrey Erman‡, Vijay Gopalakrishnan‡ and Rittwik Jana‡
*Intel Labs, †Connectem, ‡AT&T Labs

***Abstract*—Network function virtualization (NFV) promises significant cost savings, flexibility and ease of deployment. However, potential challenges in implementing virtualized network elements that can support real-world performance requirements are still an open question. For example, traditional telecom networks have a lot of complex interdependencies that can affect performance. In this paper, we study the potential bottlenecks in virtualizing cellular core network functions. Using a combination of analysis and experimentation, we quantify the impact of software-based EPC elements on various metrics including physical processing, memory, IO, and bandwidth resource requirements. We use production grade, software-based cellular network elements running on general purpose Linux servers, driven by a variety of realistic workloads derived from a real-world cellular network, to examine the combined effects of control and data planes on an LTE enhanced packet core (EPC). In particular, we discover that the SGW handles about 33% of the control plane transactions and is a potential source for performance bottlenecks as a result of the interdependencies between control and data plane processing. Our results indicate that simply replacing existing EPC elements with virtualized equivalents can have severe performance bottlenecks and that virtualized EPC elements need to be carefully designed.**

Keywords*—*Cellular, Packet Core, Virtualization

I. Introduction

Network Function Virtualization (NFV) and Software Defined Networking (SDN) are revolutionizing how networks and network elements are designed, deployed and managed. While NFV allows for separating the software from the underlying hardware, SDN's thrust is to separate control and data planes in order to better manage networks. It would not be an overstatement to say that NFV and SDN are now one of the most critical initiatives in telecom networking [3], [4].

In trying to appease carriers' desire to move to network function virtualization, the common approach that most vendors have taken is to port the legacy network software designed for custom appliances to virtual machines. While the logical option, when the node software is ported from its proprietary platform to a generic hardware platform, it loses the benefits of specialized hardware such as accelerator or fastpath packet processing usually available on the proprietary platform. At the same time, since the software was not designed or optimized for generic hardware, it performs poorly. Finally, typical telecom networks have complex interdependencies (e.g., wait for data from other elements) that can further impact performance. Hence is it is critical to understand the potential bottlenecks with virtual network elements and incorporate that understanding into their design.

978-1-4673-6762-2/15/$31.00

It is in that spirit that this paper carefully looks at cellular network's enhanced packet core (EPC) elements and tries to understand the bottlenecks that need to be addressed if one were to move to a software-based packet core running over commodity servers. There are several reasons why we study potential bottlenecks in cellular EPC. First, the EPC consists of multiple control and data processing elements that are custom to the cellular network making them very specialized and typically implemented on proprietary hardware. This results in lack of economies of scale and makes a cellular EPC very expensive to deploy. Second, unlike traditional networks, the control- and data/user planes in cellular networks are tightly coupled with control plane messages associated with *individual user* or even *application* sessions. The cellular control plane messages have hard response times imposed on them and each message is associated to a specific data plane flow. As a result, the cellular network control plane processing capacity also has to scale according to user activity. Moreover, due to the tight relationship of the control and user planes, elements on both the planes need to be provisioned at once. While NFV allows for computing resources to be allocated dynamically as well as independently, it is still important to understand the interdependency in order to design and allocate resources optimally to user and control plane elements.

To that end, this paper attempts to understand both independent and combined effects of control and user plane processing resources. While previous work [6], [8] has qualitatively highlighted some of the drawbacks with virtualizing individual EPC elements, we use a combination of analysis and experiments using representative cellular traffic load from a large cellular provider on a real-world cellular EPC to quantify multiple bottlenecks that need to be alleviated before moving to a virtualized infrastructure. Specifically, we identify the control and user plane dependencies and show that these dependencies can cripple the overall data plane throughput as the load increases. The approach and experiments detailed in this paper give a detailed understanding of the resource requirements for processing of cellular network loads. We make the following specific contributions in this paper:

- We show that the Serving Gateway (SGW) is a bottleneck and has to handle 33% of the control plane transactions. We provide a detailed breakdown of the protocol transactions for some of the more important 3GPP functions (attach, handoff, relocate, etc.) and quantify the bottleneck at the SGW.
- We establish using both empirical data and a simple queuing model that the user plane packet processing performance is very sensitive to the control plane load.

A 1% increase in the cross-over interference load from the control plane can cause as much as 70% reduction in user plane system capacity. To the best of our knowledge, we are the first to quantify the impact of control plane and user plane interactions.

- We perform operational scale experiments with realistic workloads on a software based EPC using general purpose Linux servers and show the effects on system resources as the control and user plane is scaled.

Our findings indicate that the bottlenecks experienced with blindly moving to software-based EPC nodes are severe enough to warrant a rethink of how to optimally build these elements in software.

II. Background

We will briefly explain the components of the EPC, the interactions between them, and highlight the performance and scalability bottlenecks of the control plane.

A. 3GPP LTE EPC architecture

The LTE cellular network consists of two main components: the LTE Radio Access Network (RAN), and the evolved packet core (EPC). LTE RAN consists of the eNodeB (enhanced NodeB), which communicates with mobile devices via the radio link and then forwards packets to the eventual destination via the EPC. The eNodeB also performs radio resource control and cooperates with the Mobility Management Entity (MME) for mobility management (e.g., handover).

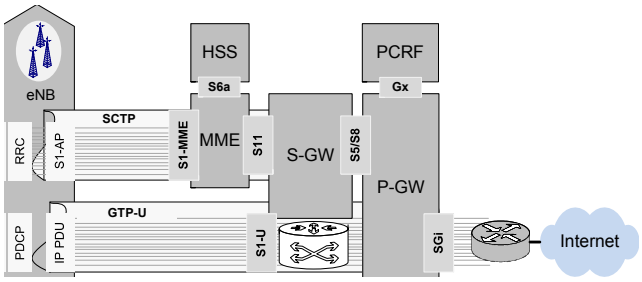

Fig. 1. Traditional cellular EPC with logical interfaces

The main elements of the EPC consist of the MME, the serving gateway (S-GW), and the Packet Data Network Gateway (P-GW). There are auxiliary nodes such as the Home Subscription Server (HSS), Policy Control and Rules Function (PCRF) and Policy Control Enforcement Function (PCEF) providing additional network functions. The MME is responsible for all control plane messaging including user authentication via the Home Subscriber Server (HSS), session establishment and release, and mobility management. The S-GW and P-GW are on the data path, and their main function is packet routing/forwarding, traffic management and accounting, and policy enforcement. The S/P-GW also act as anchor points in the cellular network with the S-GW being the anchor for inter-eNodeB handover, and the P-GW acting as a gateway/anchor to external networks (e.g., the Internet). In a traditional cellular network, almost all of these nodes are implemented on proprietary hardware using proprietary OS and vertically integrated application functions. Figure 1 shows the EPC network elements and the logical interfaces over which these elements interact with each other.

TABLE I. Transaction per NAS Event by EPC Element

Event Type	MME	HSS	S-GW	P-GW	PCRF
Attaches	10	2	3	2	1
Add. Default Bearer Setups	4	0	3	2	1
Dedicated Bearer Setups	2	0	2	2	1
Idle-to-Connected	3	0	1	0	0
Conntected-to-Idle	3	0	1	0	0
X2-based Handovers	2	0	1	0	0
S1-based Handovers	8	0	3	0	0
Tracking Area Updates	2	0	0	0	0
Total	34	2	14	6	3

B. Interactions between EPC elements

Before a device can send or receive data, it has to first establish a GTP (GPRS Tunneling Protocol) tunnel. The GTP tunnel, established between the eNodeB and the P-GW, provides logical point-to-point connectivity per device as it moves around in the network. The GTP tunnel comprises of two halves; one between eNodeB and S-GW and one between S-GW and P-GW. While the latter is retained as long as the device is registered in the network, the former is torn down whenever the device goes idle, and re-created whenever data is exchanged. On successful establishment of a bearer, the application traffic from the UE is transported over IP and encapsulated inside GTP-U tunnels by the eNB. The GTP-U tunnels from each of the UEs connected to the eNB are terminated at the SGW on the S1-U interface and sent on a new tunnel to the PGW. The UE application flows are then decapsulated at the PGW and sent over the SGi interface to the Internet. The return traffic follows a reverse path across the elements and is encapsulated back into the GTP tunnels to be sent over to the correct eNB to which the UE is connected.

The creation, deletion, and modification of radio bearers is handled via a set of control plane protocols called Non-Access Stratum (NAS) and is exchanged between the UE and the MME over the S1-MME interface. These NAS events can cause the MME to trigger subsequent events in the HSS, SGW, PGW and PCRF as required. Consider the bearer establishment event in Table II-B. For a UE to establish a bearer, the MME must authorize the UE with the HSS using DIAMETER protocol. Out of the 10 transactions for one bearer attach event at the MME, there are 2 at the HSS, 3 at the SGW, 2 at the PGW and 1 at the PCRF that are propagated throughout these EPC nodes. Processing of a NAS event at the MME (in this case establishment of one bearer) is blocked as a result of this chaining and can complete only after all these dependent transactions successfully complete.

Table II-B counts the transactions (request-response pair) at each of the EPC nodes for different NAS event types specified in the 3GPP specification [1]. Comparing the number of transactions between the MME and the SGW for the different interactions helps us to quantify the relative distribution of the control plane load across these two EPC nodes. We observe that 41% (14 out of the 34 transactions) of the control plane load incident on the S1-MME interface is propagated to the SGW. However, only 18% (6 out of 34) of the transactions incident on the S1-MME are propagated to the PGW, confirming that the SGW (and not the PGW) is the critical path of the control plane which has a direct impact on the user plane packet processing performance. Note that this result is in contrast to prevailing wisdom that suggests that the MME or the PGW to be bottlenecks [6], [10].

Other NAS events (e.g. 'Idle-to-Connected' and 'Connected-to-Idle' state transitions) are less 'transaction' heavy on the SGW compared to a bearer setup or a S1-based handover, imposing a single transaction . From 3GPP specifications, 'Attach' and 'S1-based handover' events are the most expensive S1-MME loads on the SGW [1]. Ignoring the impact of these single transaction NAS events, we find that 33% of the S1-MME load is propagated onto the SGW.

III. EPC Modeling and Simulation

Workloads of large cellular networks can be very diverse and scale in both control and user plane. Simulating the interaction between the different EPC nodes will provide a first step towards understanding the implications of these dimensions on the system as a whole. The MME capacity is determined by the total number of NAS events it can handle in one second from all the UEs connected to the eNodeBs. Similarly, the S/P-GW capacity is determined by the total number of user plane packets it can handle in one second across all the connected UEs. The total time a control plane event spends in the EPC system compared to a user plane packet gives some indication of the difference in system characteristics.

A. A simple queuing model of the EPC

We assume that a simple D/D/K queuing model (arrival rates and service times are deterministic) provides insight into the performance of the user and control planes for various amount of loading. For a given market (e.g. a city or metro area) the peak number of SCTP terminations and GTP-U tunnels remain relatively constant after system deployment. However, the number of NAS events per second on the control plane and the number of data packets per second on the user plane vary in real-time commensurate with actual demand on the system. From queuing theory, the system utilization ρ is defined as the number of customers per server multiplied by the average service time.

$$\begin{aligned} \rho &= [(\lambda T/m).(1/\mu)]/T \\ \rho &= \lambda/(m\mu) \end{aligned}$$

where λ is the NAS event in the control plane or user plane packet arrival rate, m is the number of servers, $1/\mu$ is the average service time and T is the time interval $(t, t+T)$. For a specific system design, $(1-\rho)$ provides the system blocking probability. $T = 1/\mu + T_w$ is the total time one event or packet spends in the system and T_w is average waiting or blocking time. This total time T a packet or event spends in the system will remain constant as long as the load (i.e. NAS event rate or user plane packet rate) varies within the bounds of the specific design specification.

B. EPC System Utilization

We now provide some insights into the calculations for the control plane and user plane service times. We use an experimental EPC system (explained in Section IV) designed to handle a load of 6400 events/sec on the control plane and 5.533 million packets per second (MPPS) on the user plane. Based on this EPC system, we emperically generate some of the EPC system parameters detailed in Table II. We use these parameters to build some intuition on user plane and control plane processing times.

TABLE II. EPC system parameters

Parameter	Value
CPU clock frequency	2.7 GHz
Measured Clocks Per Instruction (CPI)	0.44
Theoretical System CPI	0.25
Clock cycles per packet	488
# Instructions for processing single packet	1103

Control Plane: System utilization of the control plane can be found by counting the transactions across all the nodes for different events and examining carefully the transaction blocked states in each node. Since 'Attach' and 'S1-based handover' events impose the highest load, we can obtain a fair approximation of the control plane system utilization by tracing the processing of these two events across the EPC nodes. Figure 2 maps the transactions that are triggered in the EPC nodes for these two events. All events arriving on the S1-MME are handled by the MME, the interface termination. The MME may trigger subsequent events in the HSS, SGW, PGW and PCRF as required while processing these events. Transaction on each node is blocked, waiting for the dependent transaction to be processed. Processing of a NAS event can be complete only after all the dependent event triggers have been processed by the required nodes.

The total service time, S_{Tsc} for all the nodes in the event processing chain is $S_{Tsc} = (N_{MME} + N_{HSS} + N_{SGW} + N_{PGW} + N_{PCRF})1/\mu_c$ where N_x is the number of events arriving at node x and $1/\mu_c$ is the control plane service time. The total wait time, S_{Twc} before a NAS event can be completely processed is $S_{Twc} = (N_{HSS} + N_{SGW} + N_{PGW} + N_{PCRF})Twc$. The control plane system utilization can then be calculated from $(1-\rho_c)/\rho_c = S_{Twc}/S_{Tsc}$. Figure 2 shows number of events and the corresponding wait times at each node. Assuming unit wait time and unit service time, the control plane system utilization for these two NAS events (Attach and S1-handover) is $\rho_c = 0.73$ (i.e. 73% utilization or a blocking probability of 0.27 resulting in an average waiting time of $S_{TWc} = 42.9\mu$sec to handle 6400 NAS events/sec. The total time (service + wait) of each event at this load is therefore $T_C = 156.3\mu$sec.

TABLE III. Control and user plane parameters

Dimension	Parameter	Value
Control	λ	6400 per sec
	T_C	156.3μsec
	S_{TWc}	42.9 μsec
User	λ	5.533 MPPS
	T_U	181 nsec
	S_{TWu}	29 nsec

Takeaway: The system will start to drop packets if the user plane or control plane load increases beyond a critical operating point. Our system was specifically designed to handle a sustained load of 6400 NAS events/sec (see Table III). As a result, there is no packet loss experienced in any event processing for $T_C < 156.3\mu$sec. However, for loads greater than 6400 events/sec, S_{Twc} will limit the system capacity, causing response failures and queue build up. The escalating queue build up of the system is shown by the striped bars in Figure 4 when system utilization increases beyond capacity.

User plane: The EPC user plane system utilization is different from the control plane, since it involves processing of packets as they arrive on the S1-U and the SGi interfaces. The

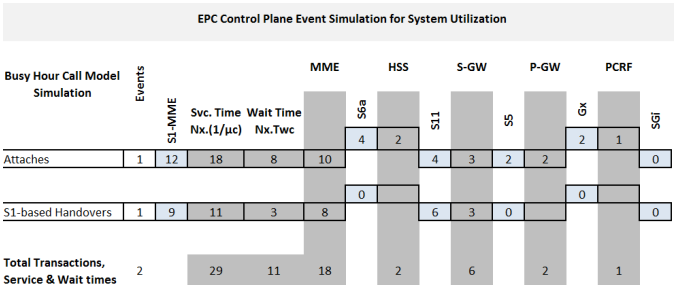

Fig. 2. Control plane message counts and processing times

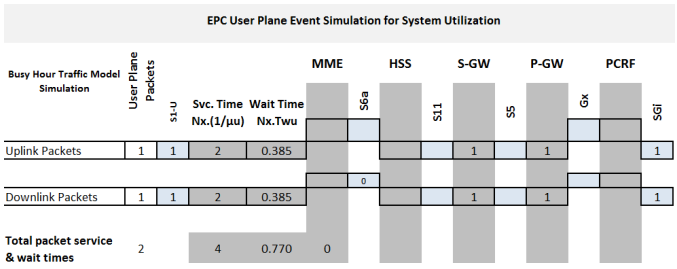

Fig. 3. User plane message counts and processing times

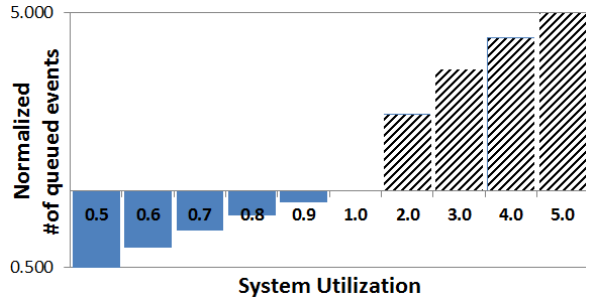

Fig. 4. Control plane event queuing

user plane packet processing should be largely non-blocking by design (i.e. packets should not be gated on the processing of any control plane events). The user plane system parameters are shown in Table II.

The total packet processing service times S_{Tu} through the nodes in the user plane is $S_{Tu} = (P_{SGW} + P_{PGW})1/\mu_u$, where P_x is the number of packets arriving at the SGW and PGW. The packet processing wait time contributing to the user plane system blocking probability will primarily be the system CPU clock wait cycles for cache updates, memory fetch, IO operations, etc.

Using values from our experimental configurations, we now derive the user plane system utilization based on the simple queuing model. The values for the experimental EPC configuration will be detailed further in Section IV. The average waiting time per node for each packet is given by

$$S_{TWu} = [(Measured\,CPI - Theoretical\,System\,CPI) \times \#Clock\,cycles\,per\,packet]/CPU\,clock\,freq.$$

For the user plane to handle a sustained load of 5.533 MPPS, the average service time for each packet is $1/\mu_u = 152$ nsec. The average waiting time per node for each packet $S_{TWu} = 29ns$. The user plane system utilization of $\rho_u = 0.84$ can be calculated from $(1-\rho_u)/\rho_u = S_{TWu}/(1/\mu_u)$. Similar to the control plane, there is a critical operating point for the user plane. Packet arrival rates beyond this design value will cause the total time a packet spends in the system $T_U < 181$nsec. The waiting time S_{TWu} will ultimately limit the user plane performance and packets will begin to flood the system and eventually be dropped.

Takeaway: T_C and S_{TWc} of the control plane is an order of magnitude *larger* than T_U and S_{TWu} for the user plane. The event blocking time S_{TWc} is primarily due to the chaining of event triggers and semaphore software code processing in the control plane. The user plane blocking time on the other hand is mainly dependent on the hardware packet processing code. S_{TWu} is therefore a function of the code execution efficiency on the hardware platform. Optimizing packet processing code for a given hardware will only reduce S_{TWu} to further improve ρ_u. However, software or hardware optimizations will *not* greatly improve the control plane blocking time S_{TWc}.

Sensitivity: We have already established the SGW handles both user and control plane processing. If the implementation of the SGW is sub-optimal and this leads to even a small interference of the control plane processing on the user plane, in theory, the user plane capacity will be significantly impacted. We plot this effect of control plane interference on the user plane capacity in Figure 5. From our model, even a 1% increase in the control plane load interfering with the user plane causes a 70% reduction in user plane capacity.

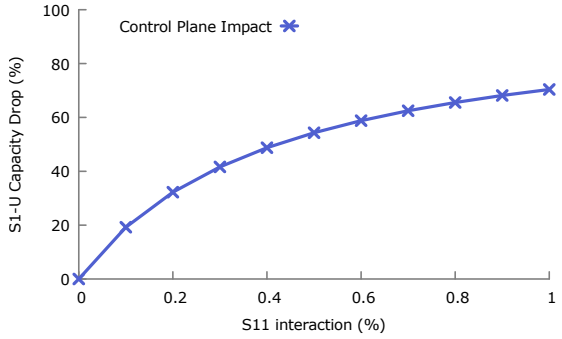

Fig. 5. User plane capacity reduction as control plane interference increases

IV. Experiments

Using an experimental EPC system, we designed experiments to get a sense of the EPC software stack operation and the hardware resource requirements. In particular, we verified: a) the effect of blocking on the control and its impact on the user plane performance, b) the limiting effect of increased system resources in control plane performance and c) the impact of control plane on packet processing. Applied loads were stepped up in each dimension individually to determine the EPC stack performance with related resource impact.

A. Experiment Setup

We used a cluster of 4 server nodes each running an instance of the SGW and PGW (see Figure 6). One of the 4 nodes ran a complete MME, IP manager, Authentication

TABLE IV. USER PLANE DOWNLINK AND UPLINK TRAFFIC PARAMETERS

Direction	Traffic type	Traffic mix %	Pkt size (bytes)	Pkt rate (pkt/sec)
Downlink	VoLTE	5.1	72	2.1M
	Web	52.2	1200	1.3M
	Video	29.5	1440	602K
	Apps	6.7	675	290K
	Email/other	6.7	1440	136K
Uplink	VoLTE	32.3	72	2.1M
	Web	43.1	690	289K
	Video	12.1	240	232K
	Apps	6.5	400	76K
	Email/other	6.2	1000	29K

Center and related functions of the EPC. This node interfaced with the other nodes through an internal 1GbE management interface. Each node delegated multiple CPU cores for each function. A RAN emulator and an application traffic generator were also used to drive traffic on both the uplink and downlink directions. The traffic mix (see Table IV) was generated using models of realistic workloads from a large cellular network. We scaled the traffic to one million UEs as per the traffic mix. The RAN emulator S1-MME interface was connected to the MME instance. Establishment of bearers on the S/PGW of each node were controlled by the MME. Application data packets on the downlink were received at the SGi gateway interface of each node and were directed to the node holding the tunnel association for that particular flow. The downlink user plane flows were directed to the appropriate node through a 10GbE mesh connecting each node of the cluster.

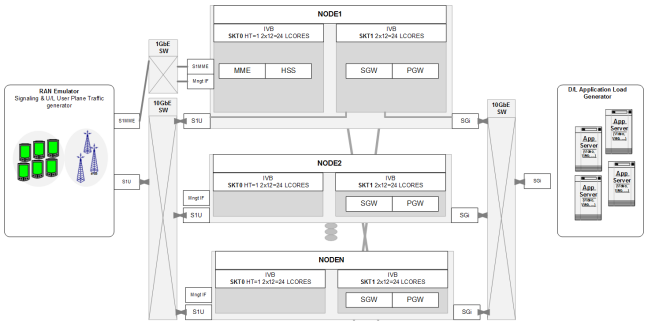

Fig. 6. Experimental EPC setup

B. Empirical results and analysis

The experiments were conducted in three stages. The first experiment measured the user plane performance in an isolated manner without any other overheads including control plane interactions. The second focused on identifying the control plane effects on the user plane under a resource constrained environment. The final experiment studies scalability aspects of system, when there is adequate allocation of resources for both control and user plane. Table V provides the measurements for the different experiments.

Experiment 1 (see Table V row 1) established the maximum performance achievable by the SGW and PGW. Each UE establishes a GTP-U tunnel at the SGW to send packets on the uplink. The tunnels were limited to 32 tunnels per node for a total of 128 tunnels across all 4 nodes of the cluster to focus the experiment on user plane packet processing performance. In order to measure pure packet processing performance of the S/PGW functions no control plane events were generated

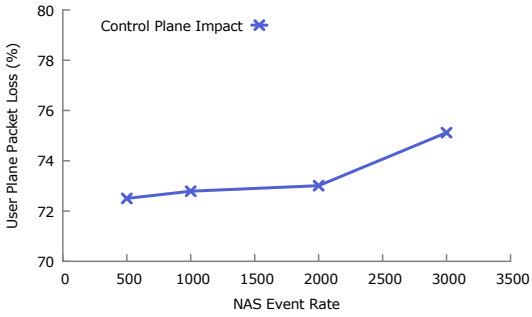

Fig. 7. Increase in the control plane load interfering with the user plane packet processing capacity

and static routes were set up for packets arriving on the 32 tunnels of each node. Packet processing functions were implemented on the DPDK pool mode drivers [2] and functions were optimized to ensure zero loss of the packets processed as the user plane packet arrival rate was steadily increased. The system could sustain a combined uplink and downlink load of 11.307 MPPS on the S1-U interface for each node in the testbed. Approximately 75% of 6.793 MPPS (5.095 MPPS) packets arriving on the SGi interface of a node were forwarded or load-balanced to the other nodes in the system. The maximum sustainable user plane capacity for each node is therefore $(16.402 = 11.307 + 5.095)$ MPPS. With 5 CPU cores dedicated to processing packets on each node the capacity per core is $16.402/5 = 3.28$ MPPS per CPU CORE. This sets the performance ceiling of pure packet processing capacity of the S/PGW functions.

Experiment 2 (see Table V, rows 2A and 2B) was designed to measure effects of the S1-based handover control events on user plane packet processing. A set of two EPC nodes were setup handling 292K tunnels each making up a total of 584K tunnels. In order to study the independent effect of control plane events, packets arriving on tunnels on one node were forwarded out of the same node, eliminating any cross node traffic. First, with no handover events (row 2A), it was observed that the significant increase in number of tunnels to 292K dropped the packet processing performance per node to 1.624 MPPS - an order of magnitude lower than the 11.307 MMPS/node observed previously in experiment 1 for the 32 static tunnels. To accommodate the large number of tunnels, the size of the lookup tables associated with the tunnel end points had increased. This drop in user plane packet processing performance clearly indicates the need for more efficient lookup algorithms as the number of tunnels attached to the EPC system increased.

Keeping the user plane load fixed at 1.624 MPPS, the S1-based handover event rate on the control plane was now steadily increased in steps to 3000 events/sec per server node. The packet processing performance per node dropped to 1.38MPPS. Figure 7 shows a sharp drop in user plane packet processing performance with increasing control plane load on the x-axis. Counting the packets sent/received on the uplink with the packet received/sent on the downlink showed the S/PGW instances were dropping a significant number of packets ($> 50\%$) arriving on the user plane for processing. This reinforces the learning from our simulation in section 3, that the serialization effect of the control plane on the user plane,

TABLE V. EPC PERFORMANCE MEASUREMENTS, MPPS: MILLION PACKETS PER SECOND

Expt.	# Nodes	# Tunnels	# eNB	Control plane rate(event/sec)	U/L + D/L rate(MPPS)	U/L pkt size (bytes)	D/L pkt size (bytes)	# S/PGW CPU cores	MPPS/core
1	4	128	0	0	11.31	256	128	5	3.28
2A	2	584K	800	0	1.62	172	673	1	1.62
2B	2	584K	800	3000	1.38	172	673	1	1.38
3A	2	600K	800	7200	3.78	172	345	4	1.31
3B	4	1000K	800	12000	3.78	172	345	4	1.49

does exist even in an optimized EPC architecture. However this effect can be minimised if the user plane is provisioned with sufficient number of CPU cores. In our experiments this was see to be 4x cores. The above measurements clearly reflect the need to completely isolate the user plane packet processing from the control plane event loads.

Experiment 3 (see Table V, rows 3A and 3B), was designed to measure how the user plane and control plane queues grow with increasing load. Figure 8 plots the user plane packet arrival rate against system resources (number of server nodes with S/PGW instances). The system resources had to be increased as the user plane load increased. Specifically, 4 server nodes had to be used to avoid packet flooding as the load factor was increased by 4x the designed user plane load.

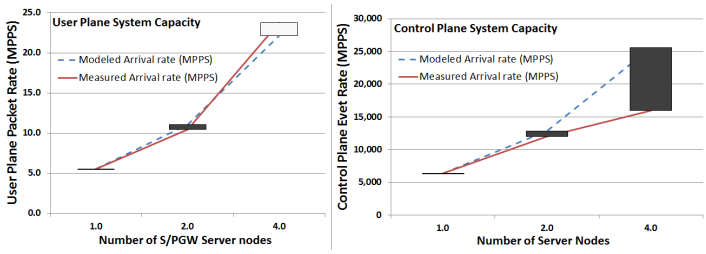

Fig. 8. User plane and control plane queue flooding

Figure 8 also plots the control plane event arrival rate against system resources (server nodes with MME instances). As the event arrival rate increased more system resources needed to be allocated. The modeled rate (blue line) shows the desired scaling effect if there were to be no bottlenecks. However, note that the actual measured control plane performance could only match the modeled rate up to a point (2 server nodes). Doubling the system resources beyond 2 server nodes did not proportionately improve the system response to keep up with the increasing event rate. This non-linearity is due to the chaining of the control plane event triggers and semaphore based processing. This empirical evidence verifies the limiting effect of increased system resources in control plane performance.

V. RELATED WORK

The application of NFV and SDN to telecom networking is gaining increased industry attention [3], [4]. Many vendors have been taking existing LTE core modules (e.g. MME, SGW, PGW) and ported them to a virtual machine. This kind of nodal based approach has several drawbacks and some of which are highlighted in [8], [11]. Kempf et al. [6] investigate how SDN can be applied to the EPC by deploying it in the cloud. By having control and data plane separation, IP flows can be routed from the eNB to the PGW via middle boxes using an OpenFlow controller [9]. While these works provides insight to an SDN based EPC in the cloud, it does not discuss the performance implications of moving real network workloads, especially the sustainability of control plane event rates by the OpenFlow controller. More generally, NFV and SDN is also an area of significant study outside the area of telecom with many applications being proposed for data centers and other contexts [5], [7].

VI. CONCLUSIONS

We explored the problems of virtualizing cellular packet core components. Using a simple model and realistic experiments, we uncovered performance bottlenecks in a software-based LTE core network architecture. We quantified the interference that the control plane packet processing induces on the user plane. In particular, we have shown that 33% of the SGW is busy handling control plane packets, and as the control plane workload increases beyond a certain threshold, the user plane capacity is impacted. We identified the specific protocol interactions that are costly and quantified the performance bottlenecks for these message sets. Our results lead us to conclude that simply replacing existing EPC elements with virtualized equivalents has severe performance bottlenecks and that one needs to carefully think through the design of virtualized EPC elements.

REFERENCES

[1] 3GPP TS 24.301: 3GPP Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS). http://www.3gpp.org/DynaReport/24301.htm.

[2] DPDK. http://dpdk.org/.

[3] AT&T's Cloud Future Takes Shape. http://www.lightreading.com/atandts-cloud-future-takes-shape/d/d-id/707903, 2014.

[4] ETSI. ETSI ISG on Network Functions Virtualization (NFV). http://portal.etsi.org/portal/server.pt/community/NFV/367.

[5] J. Hwang and K. K. Ramakrishnan and T. Wood. Netvm: High performance and flexible networking using virtualization on commodity platforms. In *USENIX NSDI*, 2014.

[6] J. Kempf and B. Johansson and S. Pettersson and H. Luning and T. Nilsson. Moving the mobile Evolved Packet Core to the cloud. In *IEEE Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2012.

[7] J. Martins and M. Ahmed and C. Raiciu, V. Olteanu and M. Honda and R. Bifulco and F. Huici. Clickos and the art of network function virtualization. In *USENIX NSDI*, 2014.

[8] S. Matsushima and R. Wakikawa. Stateless user-plane architecture for virtualized EPC. https://tools.ietf.org/html/draft-matsushima-stateless-uplane-vepc-00, July 2013.

[9] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, Mar. 2008.

[10] M. R. Sama, S. B. H. Said, K. Guillouard, and L. Suciu. Enabling Network Programmability in LTE/EPC Architecture Using OpenFlow. In *Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2014.

[11] G.-H. Tu, Y. Li, C. Peng, C.-Y. Li, H. Wang, and S. Lu. Control-plane protocol interactions in cellular networks. In *ACM Sigcomm*, Aug 2014.