A New Architecture to Offload Network Traffic using OpenFlow in LTE[†]

VENMANI Daniel Philip[*], Yvon GOURHANT[*], and Djamal ZEGHLACHE[**]

Abstract Next generation cellular applications and smart phone usage generate very heavy wireless data traffic. It becomes ineluctable for mobile network operators to have multiple core network entities such as Serving Gateway and Packet Data Network Gateway in 4G-LTE to share this high traffic generated. A typical configuration consists of multiple serving gateways behind a load-balancer which would determine which serving gateway would service a end-users'request. Such hardware is expensive, has a rigid policy set, and is a single point of failure. Another perspective of today's increasingly high data traffic is that besides it is being widely accepted that the high bandwidth LTE provides is creating bottlenecks for service providers by the increasing user bandwidth demands without creating any corresponding revenue improvements, a hidden problem that is also passively advancing on the newly emerging 4G-LTE that may need more immediate attention is the network signaling traffic, also known as the control-plane traffic that is generated by the applications developed for smartphones and tablets. With this as starting point, in this paper, we propose a solution, by a new approach considering OpenFlow switch connected to a controller, which gains flexibility in policy, costs less, and has the potential to be more robust to failure with future generations of switches. This also solves the problem of scaling the control-plane traffic that is imperative to preserve revenue and ensure customer satisfaction. Thus, with the proposed architecture with OpenFlow, mobile network operators could manipulate the traffic generated by the control-plane signaling separated from the data-plane, besides also reducing the cost in installing multiple core-network entities.

Key Words : Traffic Offloading, Infrastructure Sharing, Core Networks, 4G-LTE.

1. Introduction

Telephony networks, including the classical GSM [1] originally evolved to service voice calls. Network operators have gathered decade's worth of statistics describing how many calls typically occur within any specific time window during the day and how long those conversations take. For nearly any voice call, the network signaling resources used in establishing and maintaining the call are orders of magnitude less than the traffic-handling resources used for carrying the conversation. As a result, telecommunications system designs dedicate the bulk of their backhaul, backplane, and processing resources to the user plane that carries the voice and message traffic and only a small percentage to the control plane that sets up calls, manages billing, and the like. The evolution new technologies such the 4G-LTE [2], [3]

lead to the drastic accession of smartphone and mobile tablet users. In the years to come,

† CNRS-SAMOVAR Laboratory, University Pierre and Marie CURIE, Paris, France danielphilip.venmani@orange.com, yvon.gourhant@orange.com, djamal.zeghlache@it-sudparis.eu
* Orange Labs, France Telecom R&D, Lannion, France
** Wireless Networks & Multimedia Services Department, TELECOM & Management SudParis, France

smartphones will become the primary method by which much of the world will consume the internet. This puts mobile operators in a very strong position —a position that should allow them to open up new revenue streams from both downstream content users and upstream content providers. With new services based on IP and with mobile networks rapidly moving to IP technology, it is clear that IP has now become a cornerstone technology in the mobile world on a par with radio frequency (RF). While this sounds compelling, the rise of the smartphone is not without its challenges.It is putting a tremendous strain on mobile operator networks and business models. These devices, which can often be thought of as mobile computers, greatly increase the load on mobile networks. Their impact is found in the user plane where they can generate orders of magnitude more data traffic then a feature phone causing the need to replicate multiple core network entities such as the Serving Gateway (S-GW) to tackle million of end user requests as well as in the control plane where they can generate a great deal more signaling traffic. Business models are also being strained by the smartphone revolution, which has caused a complete decoupling of traffic carried from revenue generated. Therefore, it is an assailable subject that mobile applications and high-end mobile devices are growing in an explosive manner leading to extravagantly increasing network traffic thereby causing tremendous problem within the mobile operator networks without creating any corresponding revenue improvements.

1.1 Problem Statement

Problem 1: A blotted out problem that is creeping up on newly emerging technologies such as the 4G-LTE that may need more immediate attention is the increasing network signaling traffic. As the use of smartphones and connected devices switches from voice traffic to data traffic, the statistics that cellular network operators gathered for classical 2G or 3G technologies turned on their head. Applications for smartphones and tablets increase the network signaling traffic. The trouble with signaling traffic is not its bandwidth demands. It does not require much of the radio spectrum for a smartphone application to set up a data link and query the network looking for page updates. But such queries do occupy resources on the network's control plane, which are much more limited. Applications on smart phones run on top of the TCP/IP protocol suite, and programmers of such applications are oblivious to the fact whether their data is causing a control traffic flow or a data traffic flow on the wireless network. These idiosyncrasies of the underlying network are hidden from the programmers by multiple layers of software and protocol specifications. As a result, mobile network operators are discovering to their immense displeasure that smart phones can generate up to ten times more control traffic than regular phones for the same amount of data traffic. What this means is that the control channels on the mobile network is likely to get overloaded faster than their data channel get overloaded as more and more smart phones come online.

Problem 2: Another perspective of today's increasingly high data traffic in mobile communications systems is the need to replicate several core network entities to handle numerous amounts of requests from the end-users. Typically when an end-user sends a request to a S-GW, which is further processed by the Packet Data Network Gateway (PDN-GW) in a typical LTE network, a response is sent to the end-user from one of potentially many S-GW acting as the logical entity. This leadsthe mobile network operators to replicate multiple S-GW with a load-balancer since the S-GW receives millions of requests each day, where the load-balancer takes care of choosing the appropriate S-GW according to the traffic characteristics handled by that S-GW at that particular instant of time. Further adding, operators want to preserve service availability to premium subscribers which paid the most for the service and are more sensitive to service failure. Without load balancing on the control-plane, operators find it difficult to differentiate between the different types of subscribers although they know that some subscribers are more sensitive to service degradation than others. However, load-balancers are expensive with prices are easily ranging upwards of $50k, highly specialized machines that typically "know" things about the S-GW they are forwarding to. They may route based on the S-GWs' current loads, location of content relative to the request, or some naïve policy such as round-robin. Since load-balancers are not commodity hardware and run custom software, the policies are rigid in their offered choices. Special administrators are required and it is not possible to implement arbitrary policies. Since policy implementer and switch are coupled we are reduced to a single point of failure. This lead to our choice of adapting to a load-balancer based on OpenFlow protocol [4]-[7] within the core network of mobile networks that is void of all such disadvantages.

1.2 Existing Solutions

Thus, the growth of mobile data brings to the forefront one of the basic design choices that network protocol designers had to make during the protocol development, specification and standardization. Under most circumstances, that basic design choice is one between the in-band control and out-of-band control. Mobile network protocols have used out-of-band control, and mobile network operators provision separate channels to carry control and data traffic on wireless links. They can provision more resources for their control traffic, but this is not particularly a choice of preference. The reason for the same is that since end users pay only for data traffic and not for control traffic. That leaves only two possible solutions on the table, (a) have the smart-phone applications be written so that they do not cause too much control traffic overload into the network or (b) deploy a new protocol that is capable of separating the control traffic from the user traffic, that could be easily located within an operator network and protect the older equipment in the mobile network from the onslaught of the control traffic. In order to attain (a), a set of best practices and guidelines to minimize control traffic load on the network can be incorporated into the development process of mobileapplications using software engineering tools. Considering solutions for (b), different types of networks equipments can be imagined located at various points in the mobile network. Some progress towards (b) is already being made by startups such as Genband which provide an ability to mitigate the signaling overhead due to the smart phones. France Telecom is calling for development of "best practice"guidelines for application developers to reduce the amount of network signaling traffic smartphones generate. In Korea, guidelines for signaling traffic have already arisen as the result of third-party application taking one operator's voice-call success rate to down 10% by saturating the control plane through excessive network signaling. In 3GPP there are three parallel efforts to address traffic offloading [3]:

Local IP Access (LIPA): LIPA provides access to a residential/corporate local network interconnected to a femtocell (Home e-Node B Subsystem). Primary motivation is to provide access to a subnet within home or an office for shared resources like printers, file servers, media servers, displays, etc. while a mobile device is attached to the 3GPP operator network. LIPA capability will be introduced with 3GPP Release 10 specifications.

Selected IP Traffic Offload (SIPTO): SIPTO refers to the ability to selectively forward different types of traffic via alternative routes to/from the terminal device. Home network flavor of SIPTO is dependent on using a Home e-Node B. In this mode, specific traffic determined by operator policy and/or subscription transferred to/from e-Node B directly to Internet/Intranet bypassing the mobile operator access/core network. Macro network flavor of SIPTO covers the ability to offload traffic next to a Radio Network Controller (RNC) for 3G or Serving Gateway (S-GW) for LTE as opposed to traversing the operator's core network.

IP Flow Mobility (IFOM): IFOM specifies the terminal device to connect to two access networks (WiFi and 3G/4G) simultaneously and forward/receive packets belonging to different flows through different access networks. Unlike the other methods IFOM does not assume the presence of Home e-Node B.

2. Concept Visualization

Let us assume a brand new LTE base station is placed next to an existing other technology base station and LTE carries only 50% of the total traffic (due to lagging deployment of devices) by 2014. Like any large scale network, wireless service provider networks have fairly long tails in their traffic distribution graphs. Based on the typical Pareto distribution (commonly known as the 80/20 rule), approximately 20% of base stations will be carrying 80% of all traffic. If the same analysis is applied repetitively (assuming self-similarity of traffic distribution patterns), it is possible to demonstrate that top 2.5% of all base stations will carry close to 1/3 of all traffic. Similar to the long-tail geographical distribution of traffic, a significant temporal lumpiness also exists. In other words, traffic is not uniform, instead it increases or decreases with people's sleeping, working, driving, shopping, etc. habits. That is why instead of looking at the average throughput, the Busiest Hour in a given month must be considered as the engineering design target. Based on a conservative figure of 0.05% of monthly traffic within the Busiest Hour, a throughput figure of 20 Mbps can be computed for the unlikely scenario of all base stations being equally loaded. If the analysis is limited to the load of top 20% of base stations (50,000 base stations),then per base station throughput requirement of 80 Mbps is derived. Certainly the top 6% or 2.5% of base stations need to serve even higher levels of traffic. By 2014, LTE with a theoretical sector capacity of 48 Mbps (in 4×4 MIMO downlink with 20+20 MHz deployment scenario) needs multiple base stations and more importantly more spectrum in a significant number of base station locations. In order to achieve such high traffic, the operators not only have to deploy large number of e-Node Bs, but also the equal number of S-GW to handle the traffic with suitable load-balancers. But, doing that would definitely result in heavy investments. Hence, we propose a load-balancer architecture that is enabled by running OpenFlow protocol in the aggregation nodes in the backhaul. This costs an order of magnitude less than a commercial load-balancer that a mobile network operator would use and this provides the flexibility of writing modules for the controller that allow arbitrary policies to be applied.

2.1 OpenFlow Background

The OpenFlow protocol allows programs running on a logically-centralized controller to coordinate a distributed collection of switches. **OpenFlow switches:** An OpenFlow switch has a flow table that stores an ordered list of rules for processing packets. Each rule consists of a pattern (matching on packet header fields), actions (such as forwarding, dropping, flooding, or modifying the packets, or sending them to the controller), a priority (to distinguish between rules with overlapping patterns), and a timeout (indicating whether/when the rule expires). A pattern can require an "exact match" on all relevant header fields (*i.e.*,a *microflow* rule), or have "don't care" bits in some fields (*i.e.*,a *wildcard* rule). For each rule, the switch maintains trafficcounters that measure the number of bytes and packets processed so far. When a packet arrives, a switch selects the highest-priority matching rule, updates the traffic counters, and performs the specified action(s). Switches also generate events, such as a "join" event upon joining the network, or "port change" events when links go up or down. **Centralized controller:** An OpenFlow network has a centralized programming model, where one (or a few) software controllers manages the underlying switches. The controller (un)installs rules in the switches, reads traffic statistics collected by the switches, and responds to network events. A controller application defines a handler for each event (*e.g.*, packet arrival, rule timeout, and switch join), which may installnew rules or issue new requests for traffic statistics. A common idiom for controller applications is to respond to a packet arrival by installing a rule for handling subsequent packets directly in the data plane. Sending packets to the controller introduces overhead and delay, so most applications try to minimize the fraction of traffic that must go to the controller. Most OpenFlow applications are written on the NOX controller platform [9], [10], which offers OpenFlow API for applications written in Python or C++. These controller applications are general-purpose programs that can perform arbitrary computation and maintain arbitrary state.

3. Proposed Solution

Thus stated above, we propose our solution to the above mentioned problems by enabling the OpenFlow protocol in mobile networks. OpenFlow advantages lead to its use beyond research, e.g. in the context of network virtualization and traffic aggregation. At its core, OpenFlow offers a higher fixibility in the routing of network fiws and the freedom to change the behavior of a part of the network without infiencing other traffic. It achieves this by separating the control plane in network switches from the data plane, and allowing for a separate controller entity that may change the forwarding rules in modern switches. This enables the critical manipulation of the control traffic and hence solves our first problem.

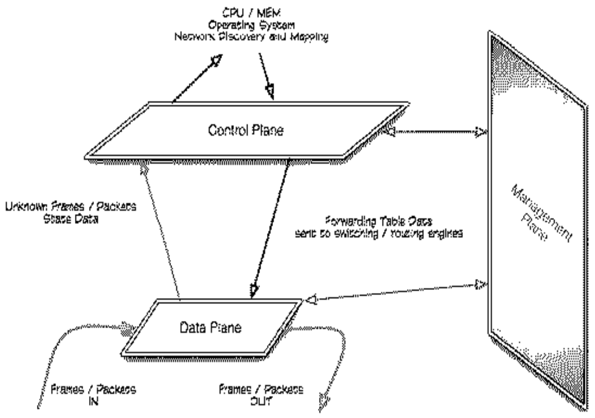


Figure 1. OpenFlow control plane and data plane separation.

Next, to enable mobile network operators to implement a load-balancer architecture which costs an order of magnitude less than a commercial load-balancer and provides the flexibility of writing modules for the controller that allow arbitrary policies to be applied, we propose our load balancing architecture that consists of an OpenFlow switch with an OpenFlow controller such as the NOX and multiple S-GWs connected to the output ports of the OpenFlow switch. The OpenFlow switch can be a node in the access or aggregation network. Each serving gateway has a static IP address and the NOX controller maintains a list of S-GW currently connected to the OpenFlow switch. Each S-GW is further connected to the PDN-GW on another port. Every end-user resolves the hostname of a S-GW to an IP address and sends its request to that IP address on the known port number. Figure 2 illustrates the architectural design for load-balancer with OpenFlow in mobile networks.

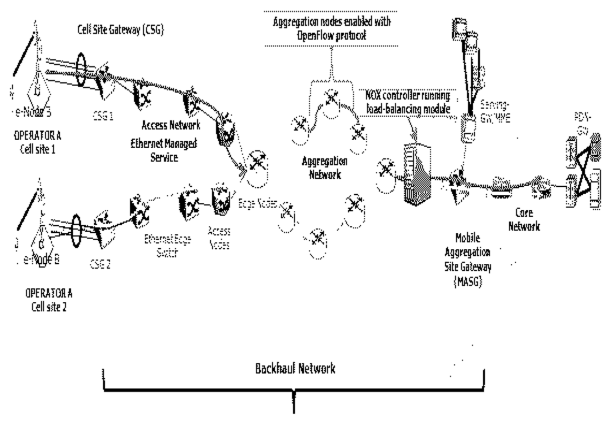
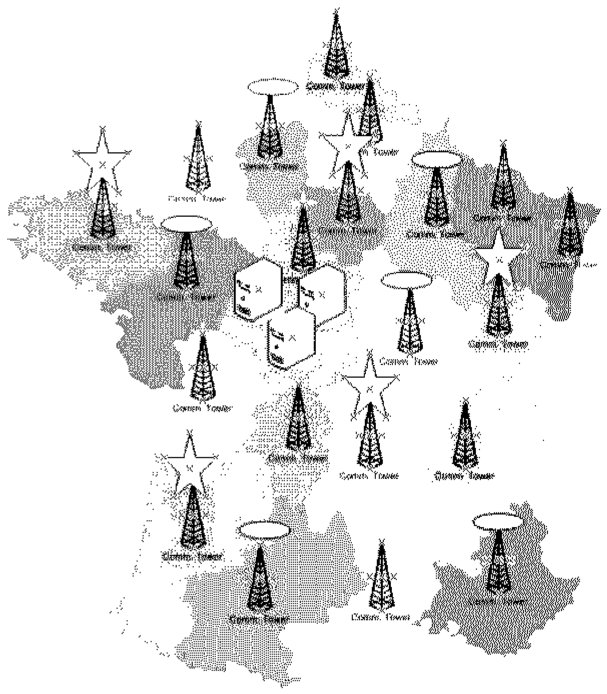


Figure 2. Mobile networks architecture with OpenFlow based load-balancing

When a packet from an end-user arrives at the aggregation network which runs OpenFlow protocol, the header information of the packet is compared with the flow table entries that is already stored in the nodes. If the packet's header information matches a flow entry, the counter for the number of packets and number of bytes is incremented, and the actions associated with the flow entry are performed on the packet. If no match is found, the switch forwards the packet to the NOX. The NOX then decides on how the packet for this flow should be handled by the switch. The NOX then inserts a new rule into the switch's flow table using the OpenFlow protocol. The load-balancing features are executed by the NOX controller. The NOX executes the handle() function declared in the module when a new flow arrives at the switch. This function defines the load balancing policies and adds new rules into the switch's flow table.



<Figure 3> Load-balancing based on geography

Elaborating further, as portrayed in figure 3, considering a country like France, where the PDN-GW is centralized in one region, mobile network operators could have one or more Openflow controllers that act as load-balancer. They could be centralized and could be placed near the core network entities. Based on their preliminary analysis of traffic statistics that had been already collected for traffic distribution during the radio planning and management phase, they can program their centralized load-balancer to choose the appropriate S-GW, at the appropriate instant of time. By performing contextual load balancing based on customer type, service type and system load the operator was able to optimize the service across all subscriber segments and enforce service policies. The algorithm that runs on the NOX will enable to select the one that allowed maximum service quality and service availability to the premium subscribers and allowed occasional users to use the service only in cases where this could be done without impacting the first group.

4. Conclusions and Future Works

Our argument in this paper is that mobile network operators could take advantage of the capability that OpenFlow protocol offers to separate the control-plane and the data-plane by the use of the OpenFlow controllers. By doing so, the resources that are spent exclusively for the control-plane alone could be surpassingly manipulated as well as cost investment involved for the deployment of several core network entities could be effectively avoided. As a primary step, we set up a test bed to evaluate the performance of OpenFlow protocol with the standard VLAN technology to see the throughput performance, since current network sharing is based on VLAN [3]. We noticed that OpenFlow gives much higher throughput performance compared to the existing VLANs. We also experimented on FlowVisor's property to isolate network resources. Our work is on-going. With this insight, we further proceed towards evaluating the performance results for the solution proposed in this paper. All these lead to the conclusion that OpenFlow is an enabler to network virtualization and service virtualization programmability within the context of mobile network architecture. Network & service virtualization for increasing the ARPU while cutting down CapEx, OpEx can increase revenue opportunities for network service providers. Nevertheless, equipment vendors and operators might also contribute to a solution by revising their assumptions as to the occurrence, type, and duration of calls upon network resources and adjusting equipment designs accordingly by adopting to such new paradigms involving OpenFlow. Today's wireless communications will soon make voice-only telephony seem as outdated as those wall phones with hand cranks for ringing the operator.

5. Acknowledgements

The authors would like to express their gratefulness to the FlowVisor team of Stanford University, U.S.A., especially to Rob SHERWOOD, Deutsch Telecom Labs, Palo Alto, U.S.A., and Hardeep Uppal and Dane Brandon of University of Washington, U.S.A. for their ever ending support towards this concept visualization.

References

[1] ETSI TS 100 933 GSM 03.68 version 8.2.0 Release 1999: "Digital cellular telecommunications system (Phase 2+) (GSM); Voice Group Call Service (VGCS); Stage 2.

[2] 3GPP TS 36.300 Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2.

[3] Universal Mobile Telecommunications System (UMTS); LTE; Network sharing; Architecture and functional description (3GPP TS 23.251 version 9.2.0 Release 9).

[4] N. McKeown, T. Anderson., H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, "OpenFlow Enabling innovation in campus networks", Proc. Of ACM SIGCOMM Computer Communication Review, 38(2):69 - 74, April 2008.

[5] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T.Y. Huang, P. Kazemian, M. Kobayashi, J. Naous, S. Seetharaman, D. Underhill, T. Yabe, K.K. Yap, Y. Yiakoumis, H. Zeng, G. Appenzeller, J, N. McKeown, G. Parulkar, "Carving Research Slices Out of Your Production Networks with OpenFlow", Proc. of ACM SIGCOMM, Barcelona, Spain, August 2009.

[6] R. Sherwood, G. Gibb, K.K. Yap, G. Appenzeller, McKeown, G. Parulkar, M. Casado, "Can the Production Network Be the Test-bed?", Proc. Of OSDI 2010.

[7] N. Handigol, S. Seetharaman, M. Flajslik, N. McKeown, and R. Johari, "Plug-n-Serve: Load-balancing web traffic using OpenFlow," Aug. 2009. Demo at ACM SIGCOMM.

[8] R. Wang, D. Butnariu, and J. Rexford, "OpenFlow-Based Server Load Balancing Gone Wild," in Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE), Mar.2011.

[9] NOX-An OpenFlow Controller. http://www.noxrepo.org.

[10] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, McKeown, and S. Shenker, "NOX: Towards an Operating System for Networks, "SIGCOMM Comput. Commun. Rev., vol. 38, pp. 105 - 110, July 2008.