OpenFlow as an Architecture for e-Node B Virtualization

Venmani Daniel Philip[1], Yvon Gourhant[1], and Djamal Zeghlache[2]

[1] Orange Labs, France Telecom R&D, Lannion, France
[2] TELECOM & Management SudParis, Evry, France
{danielphilip.venmani,yvon.gourhant}@orange-ftgroup.com,
djamal.zeghlache@it-sudparis.eu

Abstract. The ability to enable multiple virtual networks on common infrastructure with different network architectures has been gaining critical importance recently mainly because this kind of sharing does not incur any additional equipment cost for operators. An aim of our ongoing research is to take pragmatic approach towards infrastructure sharing applying operator differentiation and provide a solution to improve traffic prioritization primarily for 4G-LTE mobile networks. We propose a novel solution to the same, based on exploring OpenFlow as an architecture for e-Node B virtualization. By demonstrating the feasibility of adapting the existing OpenFlow mechanism to mobile network architecture, we illustrate the evolution of network sharing via an open network approach, based on OpenFlow. With OpenFlow, we seek to define how far it can be gone within the sharing scenarios based on the architecture of LTE/EPC defined in 3GPP, where the key lock is to open facilities to define flexible and extensible policies.

Keywords: e-Node B Virtualization, Infrastructure Sharing, 4G-LTE, OpenFlow.

1 Introduction

The flexibility to manipulate any hardware device with the ability to program leads to innovation and thus virtualization is one of the key technologies for easy innovation. Virtualization which accounts for and results in resource partitioning are similarly heavily used in network infrastructures. As a result, network virtualization is expected to gain higher interest and be a potential solution to change the way that the communication world exists today. Converging towards this topic, within the context of our research, focusing towards cellular communications, we considered the novel idea of virtualizing e-Node Bs for 4G-LTE mobile networks, thereby enabling operators to share them resulting in enormous cost reduction and take greater advantage of the available resources. The idea emerged from the fact that in recent times, cell site sharing that includes sharing of site locations and masts has been widely adopted as a form of passive sharing especially in rural areas, which was not the case few years before. This is mostly due to the fact that mobile network operators, especially in emerging countries have acknowledged that reducing the cost per bit in their backhaul is now their primary objective. Recent developments show further expansion towards the concept of 'resource sharing' i.e. wider network infrastructure sharing and spectrum sharing. Active sharing (e.g., Radio Access Network (RAN) sharing but not limited to this) has been considered by the operators as a way to reduce cost per bit in the backhaul and has been already set up in different ways which includes 3G RAN sharing between T-Mobile & Hutchison 3 UK, Vodafone & Hutchison 3 Sweden, Orange & Vodafone Spain. It is considered seriously for the rapid deployment of 3G, even in urban areas such as the small towns in Spain with a population range of 1000 and 25000 people, since it achieves approximately 43% saving in Capital Expenditure (CAPEX) and 49% in Operating Expenditure (OPEX), in addition to the passive sharing [28]. Similarly, in Long Term Evolution (LTE), Evolved UMTS Terrestrial Radio Access (E-UTRAN) sharing has already been standardized as an agreement between operators towards active network sharing [27]. Besides, infrastructure sharing has a good impact on energy consumption which is primordial in emerging countries. Africa as a whole is characterized by a very low penetration rate of fixed networks (e.g. 0.7% in Senegal, 3% in Cameroon). By contrast, a significant and rising part of the population owns a mobile phone: 25% on average [1]. Both the rurality of the population and its insolvency acts as a brake upon prospective deployment of fixed infrastructures taking into account the huge investments necessary to install wired solutions. In the sub-Saharan African countries like Kenya, Uganda, Nigeria as well as the Eastern European countries, it is undesirable for each cellular operator to replicate expensive telecom infrastructure to reach the subscribers in remote rural areas even if they were able to afford it. Hence, they go for access network sharing. Digging deeper into the existing network sharing policies [27], it is possible to conclude that operators' consensus on sharing agreement is that a User Equipment (UE) may switch from operator A e-Node B to operator B e-Node B in case of failure in operator A network. This kind of sharing policy can have an impact on the inability of the operators to differentiate themselves over a long duration of time, where every operator becomes unique in terms of QoS. This solution will not be valid when all operators will have enough clients to fill their resource. Additionally, with a high degree of shared resources using today's technologies, the stimulation for competition between the operators is gradually reducing. Hence, from a research perspective, we emphasize the way to evolve infrastructure sharing where the policies could enable "Service Differentiation", ex. service priorities, dynamic sharing policies between operators. Hence, we propose our solution which is based on virtualization of e-Node Bs of operators within the LTE/EPC architecture, where more dynamicity and differentiation in access network sharing could be incorporated by OpenFlow [3], [4] mechanisms, especially when the Telecom regulator imposes it. With OpenFlow, we seek to define how far it can be gone within the sharing scenarios based on the architecture of LTE/EPC defined in 3GPP, where the key lock is to open facilities to define flexible and extensible policies.

Therefore, in this paper, the properties, features, and limitations of OpenFlow enabled devices when illustrated within the context of LTE/EPC architecture are clearly described. The mobile network architecture model was prototypically simplified and simulated by employing the currently available virtualization technique, FlowVisor [5] proposed by the OpenFlow group consortium, since our proposal is based on adapting OpenFlow protocol to the LTE/EPC architecture and the performances were evaluated with comparative results. This paper is a proof of concept experimentation where we pictured two scenarios (but not limited to this) to validate the virtualization behavior of FlowVisor on mobile network architecture. One scenario which evaluates the performance of a network based on OpenFlow protocols compared to the standard virtual local area network (VLAN) slicing techniques [13], since current access network sharing techniques [27] are based on VLANs. The second scenario details about virtualizing the e-Node Bs for different traffic classes and allocating one slice per operator depending upon their traffic needs and evaluates how the available bandwidth is isolated efficiently depending upon the traffic. The most interesting feature of networks based on FlowVisor virtualization technique is that, it gives the operators, the possibility to slice or virtualize bandwidth, traffic, topology of any given network to give each slice its own fraction on a link to the sharing operator. The rest of the paper is structured as follows. Section II describes the various virtualization techniques and gives an overview of the various virtualization projects carried out and its stand in today's communication world. Section III details the adaptation of OpenFlow protocols within LTE/EPC architecture with a brief introduction about LTE/EPC architecture and OpenFlow architecture. This is followed by the last section where we evaluated our results that concludes the paper summarizing our future works.

2 Network Infrastructure Virtualization

Within the context of network infrastructure virtualization, the cellular network architecture can be seen as a physical infrastructure heavily deployed with numerous equipments thus allowing to host multiple virtual networks owned by different mobile network operators. This enables each operator to dynamically adjust in switching resources as well as to maintain independent management control and offer differentiated services in support of the competitive landscape of their geographic region. Taking a brief look on the current state of the art on virtualization techniques, it is already a published result that different operators might manage different virtual networks, all hosted on the Internet, but sharing the same physical infrastructure [6]. Also, virtualization for servers, routers and wire-line links in the internet architecture has already been extensively studied in the literature [7-12].This implies applying the knowledge gained from operating system virtualization experience to network components, leading to virtual network resources like virtual routers, virtual base stations. Vanu MultiRAN [35] is a solution that is available which enables multiple operators to virtually share a single physical network. However, the limitation of this solution is that it is restricted to only 3G and there is no central entity for the each virtual operator to mange all of their virtual networks together. Apart from these, a number of research initiatives and projects all over the globe have started focusing on Network Virtualization, e.g. GENI [15] [16], PLANETLAB [17], VINI [18], CABO [19], Cabernet [20] in the United States; 4WARD [21] [22] in Europe, AKARI [23], AsiaFI [24] in Asia and many others. These show that the current direction in designing the Future Internet is going in favor of having multiple coexisting architectures, where each architecture is designed and customized to fit and satisfy a specific type of network requirements rather than trying to come up with one global architecture that fits all.

3 OpenFlow as an Architecture for e-Node B Virtualization

3.1 LTE/EPC Architecture in Brief

From a technical point of view, Long Term Evolution (LTE) [36] is a radio platform technology that is standardized by 3GPP that allows operators to achieve even higher peak throughputs than other existing mobile technologies in higher spectrum bandwidth. LTE uses Orthogonal Frequency Division Multiple Access (OFDMA) on the downlink which is highly flexible in channelization, achieving peak rates in the range of 100 Mbps in high spectrum bandwidth radio channel sizes ranging from 1.4 to 20 MHz. On the uplink, however, a pure OFDMA approach results in high Peak to Average Ratio (PAR) of the signal, which compromises power efficiency and, ultimately, battery life. Hence, LTE uses an approach for the uplink called Single Carrier FDMA (SC-FDMA). LTE evolution calls for a transition to a “flat,” all-IP core network with open interfaces, called the Evolved Packet Core (EPC). The goal of the EPC is higher throughput, lower latency, simplified mobility between 3GPP and non-3GPP networks, enhanced service control and provisioning, and efficient use of network resources. From an investment point of view, it has been revealed the economic reality of LTE migration facing mobile operators around the world and estimated the total CAPEX investment faced by a tier one mobile operator in the first year of roll out [2]. This is listed in Table 1. All these could be envisaged towards infrastructure sharing scenario that would impart savings in equipment costs as well as introduces flexibility in hosting easily configurable virtual networks on a common infrastructure that can be optimized independently by operators to maximize network utility.

Table 1. Cost analysis for LTE investment

Region	Estimated CAPEX investment
US	US $1.78 billion
Europe	US $880 million
Middle East	US $287 million
Asia Pacific	US $227 million

3.2 Choice of Virtualization

Realizing network virtualization technique to the LTE/EPC mobile network architecture means to virtualize the infrastructure of the LTE system. This includes e-Node Bs, routers and even ethernet links and let multiple mobile network operators share a common infrastructure that already exists, by creating their own virtualized network depending on their requirements. From our research prospective, there are primarily two different scopes of virtualization that are foreseen for the LTE/EPC mobile architecture. The first one falls under the scope of virtualization of the air interface between the UE and the e-Node Bs and the second one is to virtualize the physical nodes from the e-Node Bs extending to the backhaul. In [25], the authors carried out virtualization of air interface between the UE and the e-Node Bs by running Hypervisor [44] on the physical e-Node Bs. The simulation results proved that based on the contract configurations and the traffic load of each virtual operator, when the air interface resources are shared among the operators, the overall resource utilization is enhanced and the performance of both network and end-user is better. Although the simulation results are quite specific, the basic findings are representative and show the advantages of applying network virtualization to the LTE/EPC architecture. Their results also demonstrated that the sharing operators benefitted from virtualization mainly by being able to cut costs and providing better performance for the users.

Forecasting such results as the possibility of opening the market to new players especially Greenfield operators that can serve a specific role and have small numbers of users, in this paper, we propose a solution that is based upon virtualization of the physical nodes of the LTE/EPC architecture which particularly includes the e-Node Bs. Each e-Node B is virtually sliced and the resources of physical e-Node Bs owned by an operator are allowed to be controlled remotely by the sharing operator also. Current access network sharing techniques [27] are based on VLANs [13], a common network slicing technique. However, from our research results, we could not be convinced with the advantages that VLANs are offering at the moment. In enterprise and data center networks, VLAN technology is commonplace and continues to evolve. VLANs like IEEE 802.1Q operate mainly on the link layer, subdividing a switched Local Area Network (LAN) into several distinct groups either by assigning the different ports of a switch to different VLANs or by tagging link layer frames with VLAN identifiers and then routing accordingly. When two operators decide to share the same e-Node B with the current VLAN techniques, the operators partition the network by switch port and all traffic is mapped to a VLAN by input port or explicit tag. Nevertheless, these types of partitioning by the VLANs are considered as coarse-grained type of network slicing that complicates IP mobility or wireless handover. On the other hand, in the backbone networks, virtualization in the form of different protocol families utilizing a single Multi Protocol Label Switching (MPLS) core network [34], [35], Virtual Private Networks (VPN) [14] (both layer-2 and layer-3) and tunneling technologies (e.g., IPSec) are widely used and allow some degree of sharing of common physical infrastructures. However, such virtualization approaches are focusing on the virtualization of links and does not allow for traffic differentiation. Accordingly, our solution is based on the idea of having a dedicated OpenFlow network [3], [4] which implements FlowVisor [5] based isolation, which deals with the virtualization of a whole network infrastructure with the ability to control the traffic remotely.

3.3 OpenFlow Architecture in Brief

The fundamental concept behind OpenFlow is that it allows the path of network packets through the network of switches to be determined by software running on a separate server. This separation of the control from the forwarding allows for more sophisticated traffic management than feasible today using Access Control Lists (ACLs) and routing protocols.It works by standardizing the interface between control and data planes and defines atomic behaviors for packet handing within each switching element. The control plane is then moved off-box into a centralized server called the OpenFlow Controller [26], thus enabling users to program their own network behaviors by injecting their own control programs into the controller. FlowVisor is a specialized OpenFlow controller that uses the OpenFlow protocol to control the underlying physical network. It acts as a transparent proxy between OpenFlow-enabled network devices and OpenFlow controllers, using the OpenFlow protocol to communicate with both the controllers and network devices, which are e-Node Bs in our scenario. FlowVisor can logically slice an OpenFlow network and allow multiple controllers to concurrently mange different subsets or different slices of the network resources. Slices can defined by any combination of ten packet header fields [5], including physical layer (switch ports), link layer (src/dst mac addresses, ether type), network layer (src/dst IP address, IP protocol), and transport layer (src/dst UDP/TCP ports or ICMP code). FlowVisor slices can also be defined with negation ("all packets but TCP packets with dst port 80"), unions ("ethertype is ARP or IP dst address is 255.255.255.255"), or intersections ("netblock 192.168.0.0/16 and IP protocol is TCP"). In this way, much like a Hypervisor that acts in a standard machine virtualization, FlowVisor intercepts all control messages to and from the data path and then checks severely and re-writes them to ensure isolation. In an OpenFlow network, when a packet arrives at a switch that does not match any cache flow entries of the switch, the switch generates a message to the controller asking what to do with the packet that has been received of this form. The FlowVisor intercepts this message and makes a policy check to determine which controller is responsible for this packet. This policy check is what we define a slicing definition, i.e. when an OpenFlow switch connects to a FlowVisor, the FlowVisor receives all the slices configured to the OpenFlow switch based on the MAC address. The message is then forwarded to the appropriate controller associated with the slice which makes the forwarding decision. Once the decision is made, the controller sends a corresponding new forwarding rule back down to the switch. The FlowVisor again intercepts the rule and does another policy check, this time, to ensure that the new rule does not infringe on the traffic from other slices. Once the rules are approved by the FlowVisor, it is forwarded onto the switch, cached and then the packet is forwarded on appropriately. Any new packets arriving further, upon matching the cache entry are then forwarded without going through this process again. Thus, all OpenFlow messages, both from switch to the controller and vice versa, are sent through FlowVisor. More explanations about the working of OpenFlow are enumerated in [3-5].

Our scenarios for infrastructure sharing essentially require that typically the e-Node B should have atleast minimal IP support and they must be OpenFlow enabled. With our solution, e-Node Bs are expected to behave as Provider Edge routers or a routing node. With 3GPP's focus on a flat all-IP LTE/EPC architecture, our argument is that, this is realizable. We exploit the capability of FlowVisor based virtualization for virtualizing LTE/EPC architecture because it gives the possibility to slice or

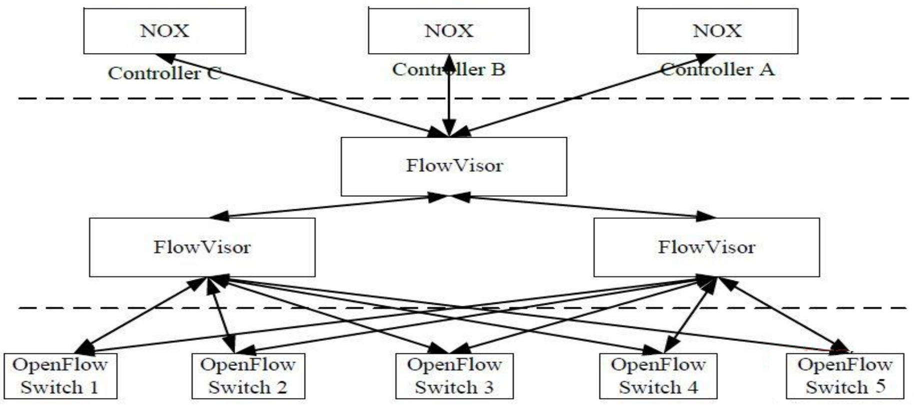

Fig. 1. Block diagram of OpenFlow architecture with FlowVisor and NOX controllers

virtualize bandwidth, traffic, topology of any given network. After virtualization, each operator gets its own portion on a link. As mentioned before, one of the current technologies that is widely used in today's networks as well as a proposed solution for LTE network sharing scenario [27] is based on VLANs. However, VLANs differ from FlowVisor in that rather than virtualizing the network control layer generally, they virtualize a specific forwarding algorithm (L2 learning). FlowVisor, on the other hand, not only supports a much more flexible method of defining networks over set of flows called flow space, it provides a model for virtualizing any forwarding logic which conforms to the basic flow model. Taking advantage of FlowVisor's flexible and fine-grained network slicing technique, with additional capability of hosting multiple OpenFlow controllers with one controller per slice [5], making sure that a controller can observe and control its own slice, while isolating one slice from another, we chose to visualize our proposed solution on network infrastructure sharing based on it.

3.4 Resource Sharing Strategies

Network infrastructure sharing should enable the operators to be able to share the network resources that are already available, without having to invest any further, just by making "slight" modifications to the existing system. This "slight" modification should not result in any additional cost more than it would result in establishing a separate network infrastructure. Our primary solution focuses on the access network sharing extending to the backhaul where the resources from the e-Node Bs until the mobile core network are shared and controlled by operators who have concluded on a sharing agreement. Although, access network sharing has already been standardized in 3GPP [27], there are no solutions proposed to control the resources of e-Node B by another operator other than the one who owns it physically. Now, according to our proposal, each operator will be able to share sufficient amount of its own resource with the other operator(s) who is sharing the infrastructure for the purpose of load sharing as well as to tackle network failure situations of their own network.

As a first step towards this innovative idea, we have elaborated our proposal by considering two scenarios. The first scenario is where the physical equipment, i.e. e-Node B is sliced into two. By this, it is implied that it enforces a policy where there are only two operators who share the same network resources. This is depicted in the Fig. 3. According to this, the entire cellular network resource is divided into two slices by the FlowVisor policy; one for operator A and one for operator B. Each operator operates and controls its own controller(s). Thus, FlowVisor policy slices the network so that operator A's sees traffic from users that have opted-in to his slice. Operators A's slice controller does not know the network has been sliced, so it does not realize it but only sees a subset of only its own traffic. When operator A's controller sends a flow entry to the e-Node Bs, FlowVisor intercepts it, examines operator A's slice policy, and rewrites the entry to include only traffic from the allowed source. Hence the operator A's controller is controlling only the flows it is allowed to, without knowing that the FlowVisor is slicing the network underneath. Similarly, messages that are originating from the e-Node Bs are only forwarded to respective controllers whose flowspace match the message. That is, it will only be forwarded to operator A if the new flow is traffic from a user of operator A that has opted-in to his slice. Thus, FlowVisor enforces transparency and isolation between slices by inspecting, rewriting, and policing OpenFlow messages as they pass. Depending on the resource allocation policy, message type, destination, and content, the FlowVisor will forward a given message unchanged, translate it to a suitable message and forward, or "bounce" the message back to its sender in the form of an OpenFlow error message.

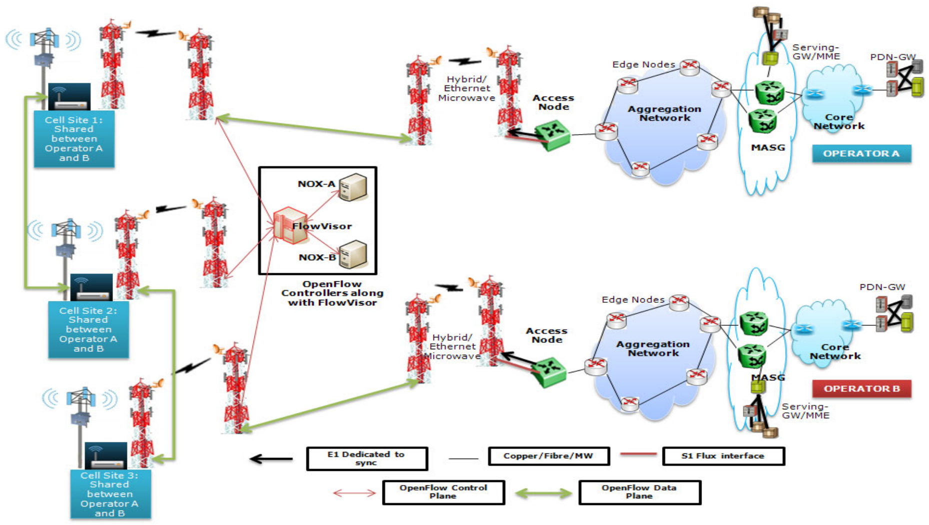

Fig. 2. Access Network sharing between operators using Virtualization (Thanks to OpenFlow FlowVisor)

For a message sent from slice controller to e-Node B, FlowVisor ensures that the message acts only on traffic within the resources assigned to the slice. For a message in the opposite direction (e-Node B to controller), the FlowVisor examines the message content to infer the corresponding slice(s) to which the message should be forwarded. Slice controllers only receive messages that are relevant to their network slice. Thus, from a slice controller's perspective, FlowVisor appears as an e-Node B (or a network of e-Node Bs); from a e-Node B's perspective, FlowVisor appears as a controller. This is one use case by which we trying to elaborate that it is possible to efficiently slice a network according to the needs of the operators.

The second scenario is where the network resources are divided into four different slices. That is each e-Node B is sliced into four for the four different classes of traffic-one optimized for conversational traffic which requires constant bit rate, like voice traffic, one optimized for streaming which is best supported as a variable bit rate service such as audio or video streaming, one optimized for interactive which uses the available bit rate and the last one for background which uses unspecified bit rate like web applications. These four different types of traffic correspond to four different virtual mobile network operators and this is enforced as a policy in the FlowVisor. Fig. 3 shows an example topology that could represent real world OpenFlow mobile network architecture based on our proposal. In Fig. 3, each e-node B in the topology is the connected to a common FlowVisor over a single network path which acts as proxy between the e-node Bs and four different NOX controllers, each operated and controlled by four different operators according to the specified traffic class. Thus, FlowVisor slices every e-Node B of our network and creates multiple logical copies of the same physical network. As explained above, when a controller sends a flow entry to the e-Node B, FlowVisor intercepts it, examines the respective slice policy and rewrites the entry to include only traffic from the allowed source. Thus the bandwidth allocated for each e-Node Bs to carry the traffic towards the core network are isolated virtually and shared among the operators. Thus, operators will be able to control and monitor the resources of a physical e-Node B without really having to take control over it.

The main advantage of this solution are

- Enormous cost reduction: If all the four operators (as in our case) decide to share the cost for deploying the network infrastructure, CAPEX will be greatly reduced for each of them individually.
- Efficient resource utilization: The operators get to optimize their traffic according to the available bandwidth. With our solution we could achieve more optimized use of the available bandwidth according to need of the applications.
- Technically simple solution: Since, the operators do not have to modify the e-Node Bs, it allows for more simplified modification at any time just in the controllers.
- The operators do not have to take care or even pay attention to the traffic of the sharing operator that flows through their own backhaul network infrastructure after the provisioning.
- The operators have the liberty to choose to prioritize the type of traffic that he would want to flow in the sharing backhaul bandwidth. Even better is, the operator can nonetheless care about the traffic priorities and just re-route a part of its own traffic in the shared bandwidth.

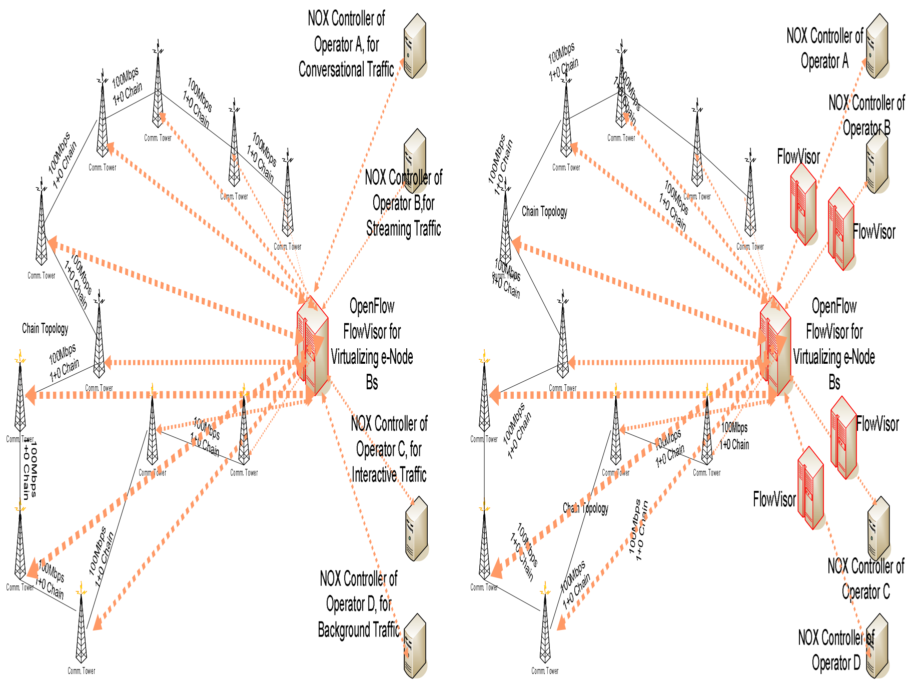
NOX Controller of Operator A, for Conversational Traffic
NOX Controller of Operator B,for Streaming Traffic
OpenFlow FlowVisor for Virtualizing e-Node Bs
NOX Controller of Operator C, for Interactive Traffic
NOX Controller of Operator D, for Background Traffic
100Mbps 1+0 Chain
Comm. Tower
Chain Topology
NOX Controller of Operator A
NOX Controller of Operator B
FlowVisor
NOX Controller of Operator C
NOX Controller of Operator D

Fig. 3. Access Network sharing between operators using Virtualization based on traffic needs

4 Performance Evaluation

The first part of the simulations was to prove that the efficiency of OpenFlow protocol compared to standard layer 2 switching is better, since a part of our argument also involves proving that the current access network sharing techniques for e-Node B based on VLAN could be replaced by OpenFlow architecture. In order to prove the validity of our proposal, we evaluated the performance of OpenFlow protocol against the standard VLANs. As a result of it, we tried to perform the three tests each separately in linux PC. The first one is use to run TCP friendly tests on the PC which had OpenFlow v1.0 running. The second one is to carry out UDP tests for CBR traffic on another PC which was OpenFlow enabled. In addition, we have to add a simple rule in the flow table to forward input packets with a certain destination or source IP address to the output port interface. The third test is to test the traffic throughput with VLAN switching, which was performed by in Linux machine by using the Bridge-tools to set the layer-2 forwarding of the Kernel. The traffic is generated by iperf [33] in TCP mode for the TCP traffic and UDP mode for CBR traffic at a link speed of 1Gbps. As shown in Table 2 and in the graph below, observing from time 60 seconds in the fig. 4, the results prove that the throughput of any OpenFlow network is almost always slightly higher than the usual switching technology. This is due to the better software implementation of packet forwarding method in OpenFlow technology. But, we also observe that the gain decreases when the packet size increases. This is because, the amount of packet that gets dropped increases when the size of the packet increases.

Table 2. Throughput analysis between openflow and vlans

Packet size (bytes)	64	128	256	512	1024	2056
TCP (Mbps) using OF	462.8	656.2	843.5	921.6	954.7	962.8
UDP (Mbps) using OF	507.4	789.3	887.6	935.5	962.5	970.4
VLAN (Mbps)	402.5	596.5	779.9	916.6	934.4	955.2

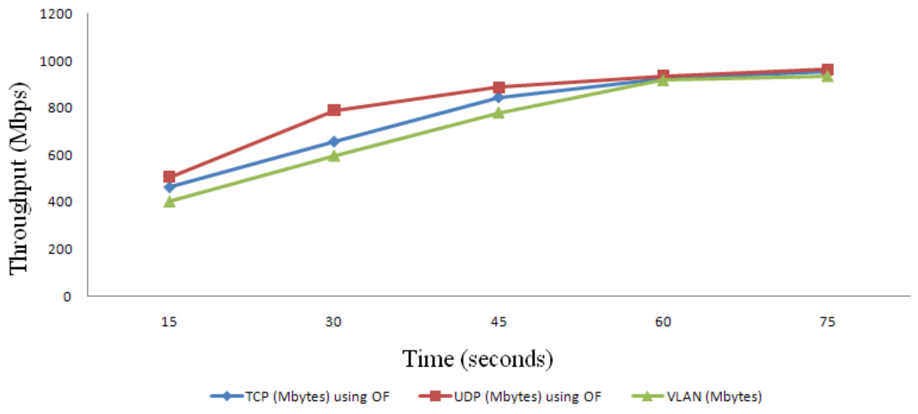

Fig. 4. Throughput analysis between OpenFlow and VLANs for packet size 1024 bytes

The second part of the test is to evaluate the performance gains that could be achieved from virtualizing the LTE/EPC nodes based on OpenFlow implementation exploiting the FlowVisor's bandwidth isolation properties. The fundamental idea is to prove that the network resource that is allocated to a certain physical equipment, which is e-Node B in our case, will be fairly shared among each and every operator who concluded on a sharing agreement based on traffic needs. To demonstrate this, we experimented by considering a simple topology which consists of one OpenFlow Switch connected to four hosts, one FlowVisor Controller and two NOX controllers [26] defining two slices, one is for TCP traffic and the other slice is for UDP traffic. The demonstrated test setup uses two physical machines- one running FlowVisor 0.7.2. configuration [28] the other one runs Mininet simulation tool [30] that helps to populate OpenFlow switches connected to hosts and NOX controllers, running on a virtual LINUX Ubuntu 10.10 [31] as the default OS. Mininet uses the software-based switch type of OpenFlow protocol that use UNIX/Linux systems to implement the entire OpenFlow switch functions. We carried out two sets of experiments. The first one is when the OpenFlow switch is directly connected to the NOX controllers and the second by connecting the switch to FlowVisor, which is inturn connected to the NOX controller. In both the experiments, there are four hosts each connected to two OpenFlow switches on which we carried out TCP and UDP tests using iperf. For the first experiment without FlowVisor connected, when iperf was carried out simultaneously for TCP and UDP traffic on the host machines, we observed that the UDP traffic consumes nearly all the bandwidth and the TCP traffic was only given a part of the bandwidth which averages to 12.28Mbytes of the 1G available link bandwidth. This, in reality means that one operator gets to enjoy more bandwidth than the other when they are sharing a common link. For the second test, where FlowVisor is connected and iperf was carried out simultaneously for TCP and UDP traffic on the host machines, the TCP traffic was able to gain control of the bandwidth ranging to a value of 716Mbytes of the 1G available link bandwidth. This concludes our solution based on FlowVisor isolation where every operator depending upon the contract signed for the specific kind of traffic, will be given a fair share of the network resource. Thus, the FlowVisor does the task of isolating the bandwidth and traffic among the different operators who agreed on sharing. Hence, we could conclude that by adapting FlowVisor based bandwidth isolation features for network infrastructure sharing in LTE/EPC networks, each operator could have its fair share of bandwidth depending upon the traffic needs. Primarily, our emphasis is that with this kind of virtualization technique based on adopting OpenFlow, the configuration of the e-Node B's themselves need not have to be modified in order to change properties of the network infrastructure that is being shared. Also, this scenario allows examination of several aspects of virtualization of e-Node Bs. First, it can be shown that it is possible to migrate one physical network infrastructure entirely into a number of isolated networks just by adding different slice definition in the FlowVisor, without really making many modifications to the existing design of the e-Node Bs. Second, it is possible to share several e-Node Bs in parallel among different operators, sporting different attributes like incorporating different traffic properties for the respective virtually isolated e-Node Bs of the operator. Third, changes within one network can be achieved dynamic during run-time, without any disruption of service in any other virtual e-Node B of another sharing operator. And finally, operators get to control their part of the network without having to be interfered by the sharing operator.

5 Conclusion and Future Works

As the mobile communications sector continues its relentless expansion with more subscribers and more advanced services generating ever-greater volumes of traffic, operators must invest in their infrastructure to provide the bandwidth to meet demand. The LTE/EPC evolution is an evolution towards an all-IP architecture. We believe that OpenFlow opens a door to a new world of virtualization thereby enabling to utilize shared network access. It can be 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. As a part of our proposal towards network infrastructure sharing within the context of LTE/EPC, we have demonstrated in this paper, the adaptability of OpenFlow protocols incorporating the basic additional features to be inculcated into the architecture. With the first phase of results here, we could conclude that network infrastructure sharing by means of virtualization could open new doors not only towards cost reduction but also gives the operators the flexibility they want in terms of traffic prioritization. It allows virtualization of an existing network infrastructure, to start at least between four operators in parallel thus enabling dynamic modification of the properties of one network operator giving fair resource allocation to operators. With such convincing results, our next phase of results would be to extend to prove that with such virtualization technique adapting OpenFlow mechanisms by modifying other parameters to the network infrastructure e.g. adding or removing links, or modifying computing capabilities of virtual e-Node Bs and thus will ease the design of sophisticated network management solutions on top of virtualized networks (e.g. resilient networks). However, at this level, there are legitimate questions to ask about the performance, reliability and scalability of a controller that dynamically adds and removes flows as the number of e-Nodes could increase for a particular operator: Can such a centralized controller be fast enough to process new flows and program the Flow Switches when it comes to running over an entire cellular architecture? What happens when a controller fails? To some extent these questions were addressed in the context of the Ethane prototype, which used simple flow switches and a central controller [37]. Of course, the rate at which new flows can be processed will depend on the complexity of the processing required by the operator trails. But it gives us confidence that meaningful experiments can be run. Scalability and redundancy are possible by making a controller stateless, allowing simple load-balancing over multiple separate devices. If we are successful in deploying OpenFlow networks in the existing mobile network infrastructure, it will lead to a new generation of control software, allowing operators to re-use controllers.

Acknowledgment. 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., for his ever ending support towards the understanding of FlowVisor.

References

1. Digital World Forum, Low cost broadband access and infrastructure, `http://digitalworld.ercim.eu/wp3.html`
2. `http://www.aircominternational.com/the-cost-of-lte-demands-innovation-says-aircom.aspx`
3. OpenFlow Switch Specification v1.0. Brandon Heller (brandonh@stanford.edu), `http://www.OpenFlowswitch.org/documents/OpenFlow-spec-v1.0.pdf`

4. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: OpenFlow Enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review 38(2), 69–74 (2008)
5. Sherwood, R., Gibb, G., Yap, K.K., Appenzeller, G., McKeown, N., Parulkar, G., Casado, M.: FlowVisor: A Network Virtualization Layer: Technical Report
6. Feamster, N., Gao, L., Rexford, J.: How to lease the Internet in your spare time. SIGCOMM CCR 37(1), 61–64 (2007)
7. Williams, D.E., Garcia, J.: Virtualization with Xen:Including Xenenterprise, Xenserver, and Xenexpress. Syngress Publishing, Inc. (May 2007) ISBN-13: 9781597491679
8. Bhatia, S., Motiwala, M., Muhlbauer, W., Valancius, V., Bavier, A., Feamster, N., Peterson, L., Rexford, J.: Hosting virtual networks on commodity hardware. Georgia Tech. University.Tech. Rep. GT-CS-07-10 (January 2008)
9. Kohler, E., Morris, R., Chen, B., Jahnotti, J., Kasshoek, M.F.: The Click Modular Router. ACM Transaction on Computer Systems 18(3), 263–297 (2000)
10. VROUT, http://nrg.cs.ucl.ac.uk/vrouter
11. VMware Server, http://www.vmware.com/products/server/
12. Cisco VN-Link: Virtualization-Aware Networking, white paper, http://www.cisco.com/en/US/solutions/collateral/ns340/ns517/ns224/ns892/ns894/white_paper_c11-525307_ps9902_Products_White_Paper.html
13. Virtual Bridged Local Area Networks, IEEE Standard 802.1Q (May 2003), http://standards..ieee.org/getieee802/download/802.1Q-2003.pdf (accessed December 17, 2008)
14. Kent, S., Seo, K.: Security Architecture for the Internet Protocol. IETF RFC 430 (December 2005), http://tools.ietf.org/html/rfc4301 (accessed December 17, 2008)
15. GENI Planning Group. GENI: Conceptual Design, Project Execution Plan. GENI Design Document 06-07 (January 2006), http://www.geni.net/GDD/GDD-06-07.pdf
16. GENI: Global Environment for Network Innovations, http://www.geni.net/
17. Bavier, A., Bowman, M., Culler, D., Chun, B., Karlin, S., Muir, S., Peterson, L., Roscoe, T., Spalink, T., Wawrzoniak, T.: Operating System Support for Planetary-Scale Network Services (March 2004)
18. Bavier, A., Feamster, N., Huang, M., Peterson, L., Rexford, J.: VINI Veritas: Realistic and Controlled Network Experimentation. In: ACM SIGCOMM 2006 (September 2006)
19. http://www.OpenFlowswitch.org
20. Zhu, Y., Zhang-Shen, R., Rangarajan, S., Rexford, J.: Cabernet: Connectivity architecture for better network services. In: Workshop on Rearchitecting the Internet (December 2008)
21. Niebert, N., Baucke, S., El-Khayat, I., et al.: The way 4WARD to the creation of a Future Internet. In: ICT Mobile Summit, Stockholm (June 2008)
22. 4WARD project page, http://www.4ward-project.eu
23. AKARI Architecture Conceptual Design for New Generation Network (translatedversion1.1), http://akari-project.nict.go.jp/eng/conceptdesign/AKARI_fulltext_e_translated_version_1_1.pdf
24. Asia Future Internet (AsiaFI), http://www.asiafi.net
25. Zaki, Y., Zhao, L., Goerg, C., Timm-Giel, A.: LTE Wireless Virtualization and Spectrum Management. In: IEEE Wireless and Mobile Networking Conference (WMNC), Third Joint IFIP (2010)
26. http://noxrepo.org/wp/

27. Universal Mobile Telecommunications System (UMTS); LTE; Network sharing; Architecture and functional description (3GPP TS 23.251 version 9.2.0 Release 9)
28. Frisanco, T., Tafertshofer, P., Lurin, P., Ang, R.: Infrastructure Sharing for Mobile Network Operators From a Deployment and Operations View. In: Network IEEE Operations and Management Symposium, NOMS 2008 (2008)
29. http://yuba.stanford.edu/git/gitweb.cgi?p=flowvisor.git;a=tags
30. http://yuba.stanford.edu/foswiki/bin/view/OpenFlow/Mininet
31. http://www.ubuntu.com/download/ubuntu/download
32. http://iperf.sourceforge.net/
33. Multiprotocol Label Switching (MPLS) Label Stack Entry: “EXP” Field Renamed to “Traffic Class” Field, RFC 5462
34. Chinni, B.: MR-238, MMBI White Paper on Use of MPLS in LTE (1) (February 2010), info@broadband-forum.org
35. http://www.vanu.com/
36. http://www.3gpp.org/LTE
37. Casado, M., Freedman, M.J., Pettit, J., Luo, J., McKeown, N., Shenker, S.: Ethane:Taking Control of the Enterprise. In: ACM SIGCOMM 2007, Kyoto, Japan (August 2007)