A SDN Controller Enabled Architecture for the IMS

Zeqi Liu, Qichao Wang, and Jae-Oh Lee
Electrical Electronics & Communication Engineering
Korea University of Technology and Education
Cheonan, South Korea
{liuzeqi, qichaowang, jolee}@koreatech.ac.kr

***Abstract*—As the essential technology to realize the open business communication network, the IP Multimedia Subsystem (IMS) is a necessary condition to enhance the mobile internet experience of users. However, with the continuous progress of communication technology, the existing architecture of the IMS has been unable to meet the needs of business development. To better solve these problems, we introduce the concept of Software Defined Network (SDN). SDN is a logically centralized control network architecture. One of SDN mechanisms is OpenFlow, and it is responsible for managing the network resources by using the remote SDN Controller. We argue for network slicing as an efficient solution that reasonably allocates resources by splitting a single physical infrastructure into multiple network resources for various users. Moreover, the SDN architecture provides tools to support network slicing. In this article, we propose and construct a combined architecture model that performs flow processing using the SDN architecture and OpenFlow via the IMS based signaling platform. It not only makes full use of the remote controller to dynamically manage network resources but also supports network slicing through the SDN architecture, which satisfies the broad service requirements of mobile networks. Besides, based on the proposed architecture, we carried out some tentative experiments and described them in detail.**

Keywords—SDN, the IMS, OpenFlow, network slicing

I. INTRODUCTION

The IMS was initially proposed in the 3rd Generation Partnership Project (3GPP), which is a kind of multimedia business forms, and it is an architecture framework for delivering IP multimedia services [1]. Fixed-migration convergence and open business communication networks have always been the goal of operators. As the essential technology carries out this goal, the IMS not only can realize the integration of fixed and mobile networks, which enables operators to provide users with vibrant and colorful multimedia services but also is a necessary condition to enhance the mobile internet experience of users [6].

However, with the continuous progress of communication technology, the existing architecture of the IMS has been unable to meet the needs of business development. It is reported that in the field of the IMS core network, the IMS standard architecture still maintains a rigid "network element chimney" network architecture until now. Specifically, the functions of network elements are allocated statically, the interfaces between network elements are complex, and the network extension uses the network element as the granularity to implement new service deployment by adding service network elements or extending existing network element functions. The existing complex "network element chimney" architecture of the IMS inevitably leads to a long period of the IMS network deployment and new business development. It will not be able to flexibly and efficiently meet the "extremely differentiated" multimedia communication needs of users and vertical industries.

To better solve these problems, we introduce the concept of SDN. SDN is a logically centralized control network architecture. One of SDN mechanisms is OpenFlow, a prominent standard protocol and interface, that is responsible for managing the network resources by using the remote SDN Controller. In the future, some verticals will generate a large number of use cased with decentralized requirements that must be adequately supported by 5G networks. To reasonably allocate resources, we argue for network slicing as an efficient solution to meet service demand. Moreover, the SDN provides tools to support network slicing. Therefore, SDN is also a natural solution to implement network slicing for 5G.

In this article, we propose and construct a combined architecture model that performs flow processing using the SDN architecture and OpenFlow via the IMS based signaling platform. The remainder of this paper is organized as follows. The introductions of the IMS policy control, SDN, and network slicing are given in Section 2. In Section 3, we propose the integrated model for inserting SDN based on the existing IMS architecture. Section 4 is the construction and some experimental attempts of the proposed architecture, then the conclusion and challenging work are in Section 5.

II. KEY TECHNIQUES FOR COMBINED ARCHITECTURE

A. The Operation of Session Establishment for the IMS

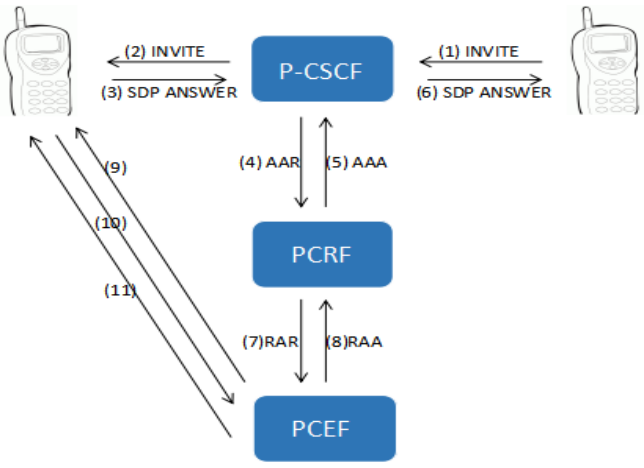

Fig. 1. Session establishment of the IMS

The core feature of the IMS is the adoption of the Session Initiation Protocol (SIP) protocol and the independence of access [4]. The establishment process of the session is, as shown in Figure 1. When The Proxy Call Session Control

Function (P-CSCF) receives an INVITE (1) from one terminal, it is responsible for maintaining the security association and protecting the integrity and confidentiality of the SIP signaling. After receiving the ANSWER (3) from the other terminal, the P-CSCF will send an Authentication Authorization Request (AAR) (4) message to the Policy and Charging Rules Function (PCRF). Based on the description of the session to be established contained in the AAR message, the PCRF selects a set of Policy and Charge Control (PCC) rules that match the session. The PCC rules are responsible for Quality of Service (QoS) and bandwidth-based policy control and flow-based charging.

After that, the PCRF sends an Authentication Authorization Answer (AAA) (5) message to the P-CSCF as a response. The PCC rule needs to be implemented in the Charging Enforcement Function (PCEF) in conjunction with the Re-Authentication Request (RAR) (7) message sent by the PCRF. After the PCEF sends the Re-Authentication Answer (RAA) (8) message, the PCEF instructs the terminal to initiate the context activation process (9). While the context is activated (10) (11), the PCC is applied in the media flow [3].

B. SDN

SDN originated from the Clean Slate Project of Stanford University, intending to the network architecture. SDN has the characteristics of the control layer and infrastructure layer decoupling separation and supports software programming to control the network. As shown in Figure 2, SDN architecture mainly includes the application layer, northbound interface, control layer, southbound interface, and infrastructure layer.

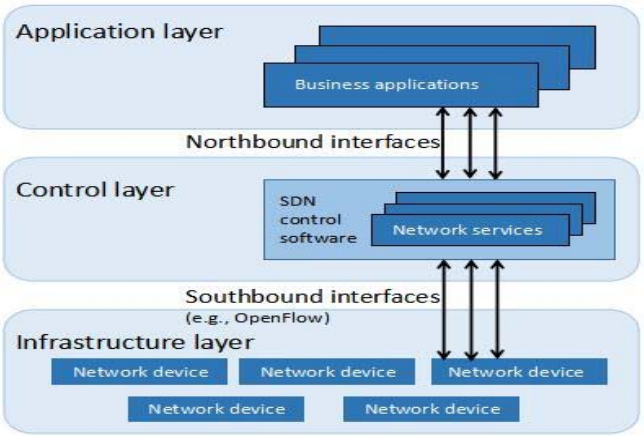

Fig. 2. SDN architecture layer

The network application layer implements the corresponding network function applications. These applications implement the configuration, management, and control of infrastructure layer devices by invoking the northbound interface. The northbound interface is an open interface between the controller layer and the application layer. The SDN Controller is the core part of the architecture and has the functions of providing the programmable capability to the upper layer network application and unified configuration, management, and control of the infrastructure layer. The southbound interface is the interface between the controller layer and the infrastructure layer. The infrastructure layer has a high-performance data forwarding function.

As one of the representative mechanisms used by SDN to communicate between the control layer and the infrastructure layer, the OpenFlow standard is called the OpenFlow Switch Specification, so it is a device specification itself that specifies necessary components and functional requirements of the OpenFlow switches as the SDN infrastructure layer forwarding device, as well as the OpenFlow protocol for controlling switches by the remote controller [10].

C. Network Slicing Supported by SDN

Networking slicing is a particular form of virtualization that allows multiple logical networks to run on shared physical network infrastructure. In other words, it can divide a network device into individually controlled port groups or split the network system into separate administrative domains. The main advantage of the network slicing concept is that it provides an end-to-end virtual network that includes not only the network but also computing and storage functions. Its purpose is to partition the physical network for optimal traffic grouping, isolating other tenants, and allocating resources at the macro level. Specifically, it allows physical mobile network operators to allocate their network resources and allows different users (so-called tenants) to reuse a single physical infrastructure.

As the main component of the SDN architecture, the SDN Controller is a logically centralized entity that performs virtualization and orchestration functions. Because of virtualization, each client context provides a specific Resource Group that can be used by the client associated with that context to realize its services. As the part of SDN Controller conceptual components, the client context provides the complete abstract set of resources (as a Resource Group) and supports control logic to form slices, including the entire collection of related client service attributes. Through orchestration, the SDN Controller optimally dispatches the selected resources to separate Resource Groups. The interaction of both controller functions can meet the various service needs of all customers while maintaining their isolation [11]. Besides, the recursive feature of the SDN architecture in terms of function is also very suitable for network slicing. The SDN control plane can contain multiple hierarchical controllers that extend the client-server relationship to various levels. So it can naturally support a recursive composition of network slicing. As the core technology of SDN, OpenFlow protocol can manage resources by slicing them in a virtualized manner, and this aspect of the OpenFlow protocol can be integrated into the IMS infrastructure. The OpenFlow Controller can identify the network slice which the flows should follow, which in order will configure the OpenFlow Switch to control network resources, to alternate network slices, or other application servers or to provide connectivity [12]. The OpenFlow switch can be a software program or the hardware that forwards packets in the SDN environment.

According to these premises, the SDN architecture satisfies the needs of network slicing to isolate tenants and optimize resource allocation and logic in a flexible, economical, and efficient way.

III. THE SDN ENABLED IMS ARCHITECTURE

In this section, we describe how SDN technology is combined with the IMS architecture and illustrate the functions of each component in this composite architecture in a little more detail. Moreover, the specific implementation of the architecture will be described in the next section. Figure 3 shows an extension based on the existing IMS architecture,

which consists of the central session architecture of IMS, the OpenFlow Controller, and the OpenFlow Switch.

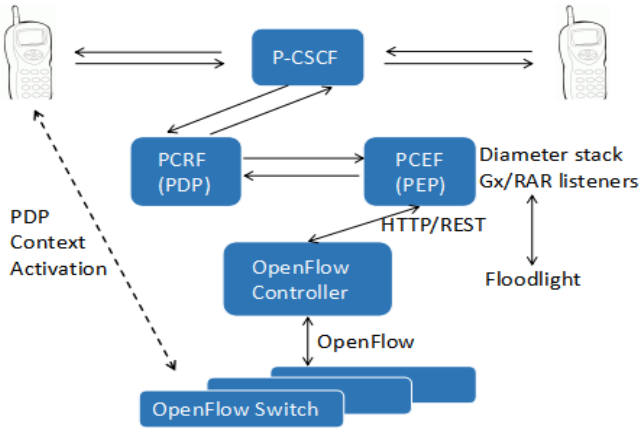

Fig.3. The approach of implementing the integrated architecture

The luxurious multimedia services of IMS require QoS mechanism support. Therefore, in the integrated architecture, two network elements, Policy Decision Point (PDP) and Policy Enforcement Point (PEP), are required. The PDP extracts the corresponding policy from the policy library and translates it into a set of behaviors that can be understood by PEP. The PEP completes the specific operations for the network device according to the policy decision result of the PDP. In the IMS component, the PCRF and the PCEF play the role of the PDP and the PEP, respectively. Individually, in the process of the IMS call flow, the SDP message carrying the QoS parameter is encapsulated in the SIP, and the PCRF checks these parameters and retrieves the appropriate policy, and notify PCEF [5]. The PCEF is responsible for implementing the policy requirements of the PCRF.

In the OpenFlow network, the OpenFlow Controller centrally controls the system to implement the functions of the control layer. The OpenFlow Switch forwards the data layer. OpenFlow Controller can be modified, or injected into modules and rebuilt programmatically, and it communicates with the OpenFlow Switch through the secure channel, which is established by the control plane network and is not affected by the flow meter items in the OpenFlow Switch. Moreover, it can dynamically reroute the traffic to alternate network resources, or a different "network slice" in cases of congestion or applying QoS in the IMS network through the IMS policy network elements [13]. The OpenFlow Switch converts the packet forwarding process, which is entirely controlled by the switch/router, into by the OpenFlow Controller and the Switch [2]. The purpose is to enable the OpenFlow Switch to connect to the external controller using the OpenFlow Protocol through the secure channel, thus separating the data forwarding and routing control [9].

IV. The Experimentation of Model Simulation Setup

A. *The Approach for Implementing Architecture*

As shown in Figure 3, the PCEF plays the role of the PEP. It is implemented in the Java language, communicates with the PCRF through the Diameter Protocol executing policy rules, and monitors the network in real-time through the Gx/RAR listener.

Floodlight is developed based on JAVA. It has better scalability and can load corresponding modules according to different network control requirements. So OpenFlow Controller is implemented through Floodlight, and applications can be controlled remotely. Besides, as the mainstream open interface model, the exposed RESTful API is used to make controller functions more accessible to applications by Floodlight, thereby OpenFlow Switch is accessed or configured, this reduces complexity and simplifies the process. Moreover, the form can achieve the purpose of autonomously managing network resources through the RESTful API. Because PECF communicates directly with the OpenFlow Controller, we can choose to program it on the PECF so that it can dynamically configure Floodlight to identify network slices on-demand and allocate reasonable traffic resources through the OpenFlow Controller. Moreover, to facilitate the PCEF to identify the network slice that the stream should follow and determine the traffic speed, it can be enhanced using the API exposed in the URL, which makes it easy to operate and require the controller.

OpenFlow adopts the architecture of separation of control and forwarding, which means that the learning of the MAC address is implemented by the Controller, the VLAN and the underlying routing configurations are also sent by the Controller to the OpenFlow Switch. For its network devices, all kinds of routers run on the controller, and the controller sends them to the corresponding routers as needed. When an OpenFlow Controller controls multiple OpenFlow Switch at the same time, they look like a sizeable logical switch.

B. *Building Operation of Virtual Simulation Model*

To implement the architecture, we built a virtual simulation experiment environment without OpenFlow hardware. We integrated OpenFlow into the IMS testbed to complete the session establishment. For the OpenFlow part, we provide a simple example. As shown in Figure 4, in the Ubuntu environment, the Virtual Box is used to create the Linux operating system. We choose to use Floodlight, Mininet Open, and Open vSwitch to simulate the controllers, switches, hosts, and related network topologies required for the experiment. Mininet is used to quickly build an SDN network and provide a simple network experiment platform for OpenFlow application testing. Open vSwitch is an open-source, feature-rich virtual switch software running on a virtualization platform that supports the OpenFlow protocol [7].

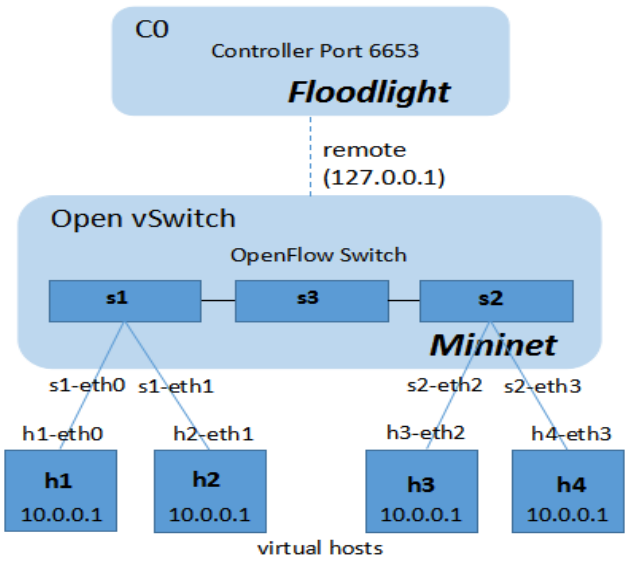

Fig.4. Structure diagram of the proposed model

There are several ways to get access to Floodlight and Mininet. We choose to download the VMs that include both Floodlight and Mininet from the official website of Floodlight. This method can simplify the operation, provide the best workload distribution, and define different scenarios and Isolating resources to increase efficiency. The specified version of Open vSwitch can be installed through Mininet, which avoids many configurations and is a simple and efficient installation method.

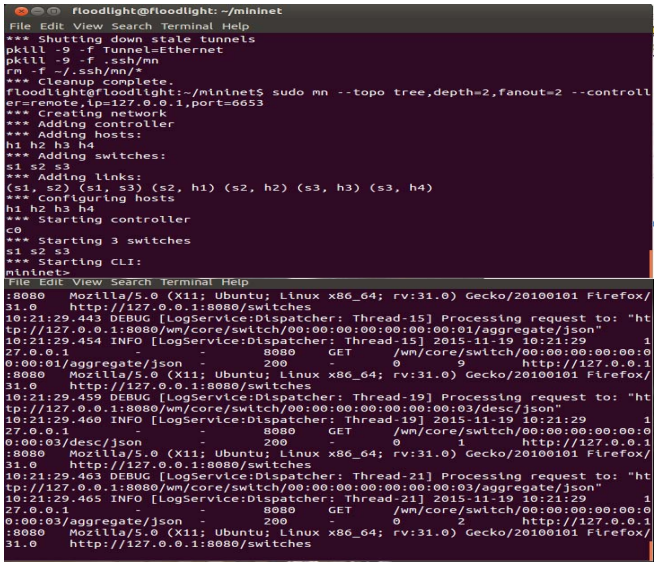

Fig.5. The start and operation interfaces of Mininet term

Mininet allows custom topology, the start and operation interfaces of it are shown in Figure 5. It can be added topology parameters by commands such as "sudo mn" "depth" "fanout" to define the desired topology.

After Floodlight is running, we can view the information in the web management interface provided by Floodlight by logging in to http://localhost:8080/ui/index.html, including dashboard, topology, switched, and hosts, as shown in Figure 6.

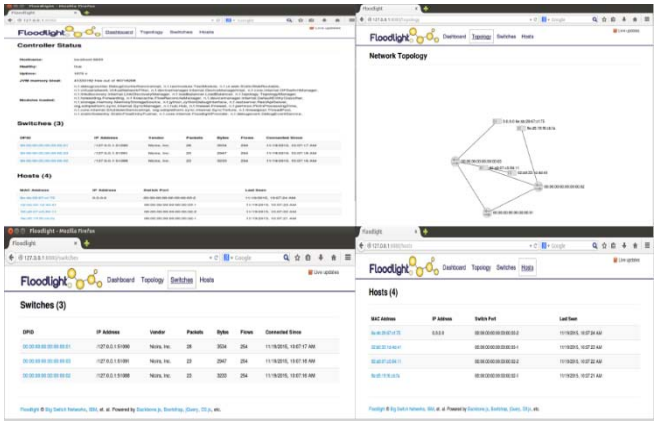

Fig.6. The web management interface of Floodlight

Through the simulation experiment using Floodlight and Mininet, the feasibility of the proposed architecture is positively affirmed.

V. Conclusion and Challenging Work

In this paper, the OpenFlow for the SDN Controller is introduced. Then, we propose a novel integrated architecture model, which is a combination of SDN and the IMS. Under the premise of guaranteeing the reliability and QoS for various services, the proposed framework not only makes full use of a remote controller to manage network resources dynamically [8] but also supports network slicing through the SDN architecture, which satisfies the broad service requirements of mobile networks. This framework may be customized for every user in the system in the 5G era and beyond, and play a vital role in achieving a genuinely flexible mobile network. Also, based on the proposed architecture, some experimental attempts have been made. Moreover, the feasibility of the proposed architecture is affirmed by the effort of simulation experiments.

In the next step, the simulation experiment is completed in-depth and thoroughly, and to improve the compatibility of the framework, we will enrich the experimental data with more experimental tools (such as ONOS). Besides, we will try to make a further study on the issue of service guarantee network slicing, so that it can provide the optimal workload allocation and improve efficiency based on defining different scenarios and isolating resources.

References

[1] J. O. Lee and H. K. Lee, “An Implementation of IMS Based PoC Service Deployment,” Journal of the Korea Academia-Industrial cooperation Society, Vol.16, No.7, pp. 4878-4883, 2015.

[2] Charalampos Rotsos, Nadi Sarrar, Steve Uhlig, Rob Sherwood, and Andrew W. Moore1, "OFLOPS: An open framework for OpenFlow switch evaluation," Passive and Active Measurement. Springer Berlin Heidelberg, 2012

[3] Poikselka. M, Mayer. G, “The IMS, IP Multimedia Concepts and Services,” 20011.

[4] J. H. Kim, J. M. Been, S. C. Kang, and J. O. Lee, “M2M Network Platform Using the MSRP,” Journal of the Korea Academia-Industrial cooperation Society, Vol.17, No.4, pp.752-757, 2016.

[5] W. S. Yang, J. H. Kim, and J. O. Lee, “A Management for IMS Network Using SDN and SNMP,” Journal of the Korea Academia-Industrial cooperation Society, Vol.18, No.4, pp.694-699, 2017.

[6] Seppanen, J. and Martín Varela, "QoE-driven network management for real-time over-the-top multimedia services," IEEE Wireless Communications and Networking Conference (WCNC), 2013.

[7] Pfaff, Ben, and Bruce Davie, "The Open vSwitch Database Management Protocol," RFC 7047, 2013.

[8] C. S. Tang, C. Y. Twu, J. H. Ju, and Y. D. Tsou, “Collaboration of IMS and SDN to enable new ICT service creation,” Proceedings of The 16th Asia-Pacific Network Operations and Management Symposium, pp.1-2, 2014.

[9] Vaughan-Nichols and Steven J, "OpenFlow: The next generation of the network?" Computer 44.8, pp.13-15, 2011.

[10] Z. Yang, C. Li, "Refactoring network, SDN architecture and implementation," 2017.

[11] Jose Ordonez-Lucena, Pablo Ameigeiras, Diego Lopez, Juan J. Ramos-Munoz, Javier Lorca, and Jesús Folgueira, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges," IEEE Communications Magazine, 2017.

[12] Z. Liu, J. Kim, and J. O. Lee, "A model of SDN controllers supporting for processing of flows based on the IMS," Proc. APNOMS 2015, pp. 384-387, Aug. 2015.

[13] Thomas D. Nadeau and Ken Gray, "SDN: Software Defined Networks," 2014.