International Journal of Hybrid Information Technology
Vol.8, No.6 (2015), pp.357-366
http://dx.doi.org/10.14257/ijhit.2015.8.6.34

Expanded Android Application for Openflow-based Visual Interface in Software-Defined Network

Xuejun Tao[(1)], Xueguang Yuan[(2)], Yan Hong[(3)] and Yang' An Zhang[(4)]

State Key Laboratory of Information Photonics and Optical Communication, Beijng University of Posts and Telecommunications, Beijing, 100876, China
wcqs517@126.com

Abstract

In the past year, the northbound interface application has become a hot topic in the Software-Defined Network technology. It offers an opening method and makes people able to develop own business application upper the SDN. In this paper, we propose and evaluate a visual plane based on android phone application to present an explicit watch of the state and working performance of the Openflow-based underlying network topology in Software-Defined Network. The impacts of the designing architecture are evaluated in the form of software simulation and actual operating effects of the android application that we have designed. Through the interface opened by the floodlight controller to the upper business application, we are aimed at making it able to call the underlying network resources and inquiry its status information conveniently. So this paper wants to achieve the unified dispatching by elaborated methods and transplants the idea to the android application. By the way of the software programming, we could make the android phone to probe the southbound network resources through the northbound interface whose connotation we will expand. Between the northbound interface and the android application, we introduce a data center and select the mysql database to implement its function. Through the technology framework we built, we are able to use an android phone to access the status of the underlying devices and monitor the software defined network better aiming at providing better service. And we judge its function from the perspective of overall feasibility and stability not only through the system performance but also sound experiment effects. Finally, we can use any android phone to view about the topology of the network effectively and simply.

Keywords: *android application, Openflow-based, resources, dispatch, conveniently, Software-Defined Network, view, effectively*

1. Introduction

As the rapid development of Software-Defined Network, not only more and more countries have invested a lot of funds to show their great interest and pay much constant attention, but also major well-known telecom corporations put in a lot of human and financial resources to develop the corresponding routing equipments supporting the openflow protocol. Thus the SDN technology gains continuing momentum stream of development contributing to the southbound and northbound interfaces [1]. As is well known that Software-Defined Network is typically composed of the controllers and openflow switches. The controller could use the southbound interface to achieve control of the underlying network and use the northbound interface to achieve spread of the upper business application. But their common goals are just to make the network working more efficiently and enhance the performance of the overall network.

The southbound interface protocol implements the control of the network through the link discovery, topology management, strategic planning, table items dispatching and so on. The link discovery and topology management mainly control the uplink channel of the

ISSN: 1738-9968 IJHIT

southbound interface to monitor and add up the reporting information from the underlying switch devices while the strategic planning and table items dispatching control the downlink channel of southbound interface to monitor the network devices [2]. However, the northbound interface is directly to service the business application and globally control the status of underlying network resources. Therefore, when we want to expand its northbound interface applications, we should considerate the underlying network architecture and devices which must support the openflow protocol. In the meanwhile, the newly research progress of the northbound application by industry peers will impact the market outlook of the SDN corporations which produce the software-defined devices to a certain extent.

To our best knowledge, we propose and evaluate the first northbound application model which enables android application to access the status of underlying network devices via the floodlight controller in software-defined network. We create an android application program and download it to the android phone. This application would use a database as our data transferring station and get the data through the mobile communication network. While the original data is from the underlying network devices which pass the openflow switches and finally the floodlight controller get the data to be transported to the database. And the database saves it at a appropriate way. Under these openflow switches, some terminals like personal computers are connected with them. The network status and other key information of these terminal nodes will be collected together and transferred to the controller. Then the data information via the controller would be saved at the database. In this system, we select the Floodlight as our controller, openflow-1.0.0 as our openflow switches and mysql as our database.

2. The Original Model and Implementation of Software-Defined Network

Since the Stanford University has put forward the software-defined network concept and given a raw model, many applications and theories are excavated and greatly enrich the original connotation of it. Figure 1 illustrates its original model framework. It adopts the splitting thinking of control plane and data plane [3]. That is to say, the controller is responsible for the network configuration and the openflow switches decide the data routing configured by the flow tables from the controller. The evolution of the concept of the software-defined network begins to change the passive status of the network and makes it having greater degree of flexibility in the definition of capacity in which network actively process the traffic and not passively carry the traffic to make the relationship between network and compute not just docking but interacting. Openflow switches expand the underlying network by the southbound interfaces while floodlight controller expands the upper business application by the northbound interfaces. When we run the SDN system, we will get the background data processing procedure like Figure 2. In this picture, we use the wireshark software to acquire the IP address of the source host and the destination host and also get the information of the link topology of the node terminal equipment. Moreover, when we locate the openflow protocol and unfold the packet data information, we will find the data transfer mechanism between the floodlight controller and the openflow switch. And the packages of packet_in and packet_out carry the data stream of the topology information.

In this paper, we adopt the research method and splitting thinking to expand the northbound interface business application. Figure 3 displays our design method and expanded framework. Outside of the original model, we increase a data storage center. It receives the data sent by the floodlight controller and saves it. If the upper business applications issue the data request, the data storage center will satisfy it timely with excellent stability. Firstly, we want to illustrate the procedure in which the floodlight

controller gets the status information from the underlying network and displays it by the intuitive network pages. The floodlight controller obtains the underlying network information and device status through the openflow protocol and mainly uses the background processing capacity of the java program. Then the java program delivers the data to the json which is the newly technology to process data transferring between the background and the network front-page. That is to say, the java rear end is in charge of the business logic processing and then transmits the final result to the json to display the data information in the form of network pages. From the network pages, when we run the SDN system guaranteeing that the openflow switch could connect to the floodlight controller and then open the home page of the network system, we will get the overall underlying device information including the link terminals and the openflow switches just as the Figure 4 shows. Based on this idea, we move the network pages to the android phone for the reason that the android application has better portability and page displaying capacity. We could put in more module functions and design more cool phone interface effects to provide more convenient service to users without restrictions of time or positions except for you just have an android phone. If we want to see the state or the topology of the network, we just open our Android telephone and run the android application. Thus, we can clearly select to see what we want.

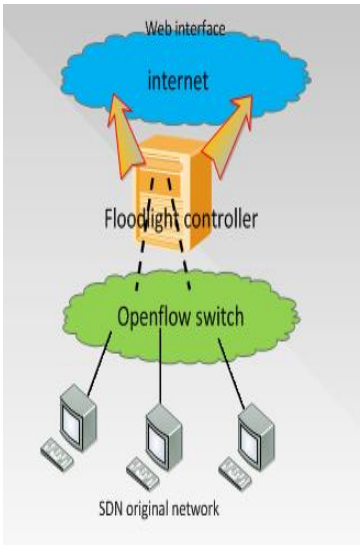

Figure 1. SDN Original Network

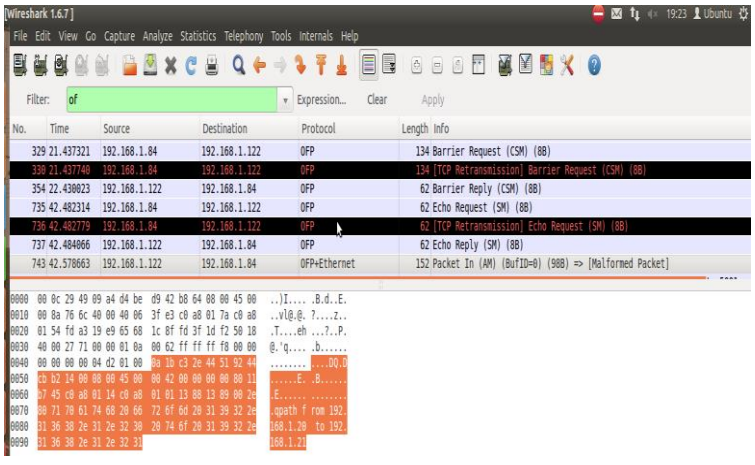

Figure 2. The Result of Experimental Simulation

2.1 The Program to Display the Visual View with the Phone Interface

First of all, we must design the phone interface how the program organizes the data to be appeared to us. Here we should fully considerate the users' experience and the showing way to make people more willing to accept with the aim at enhancing the efficiency of information acquisition. For the sake of security, we have designed the entrance interface. After you input your user's name and corresponding password, you will enter into the home interface which has a background image and four icon buttons above it. The first icon button will guide you into the status information about the floodlight controller and the mysql database. In this interface, we could clearly watch about the controller's name, the version number, location information and much other information as well as the database. The second icon button will guide you into the information of the openflow switches. In this interface, we could clearly watch about the switch's physical address and id port number as well as the flow tables' information. The flow tables could tell us the link routing and the underlying network topology [4]. The third icon button will guide you into the information about the node terminals include the terminals' ip address, the input port number, the connected switch's id number and much other information. The fourth icon button is about the overall topology architecture and gives users the best knowledge of the whole system architecture. Through the four icon buttons, we could monitor the network and better know the network. Thus we have more confidence to manage it perfectly.

The second part is to design the database which plays the most important role between the SDN controller and android application. That is to say, the database we create is playing as an interacting bridge to connect the controller of SDN and Android phone. According to the android phone interface design, we should create a SDN database and in this database we should create five tables. One table is for the admittance interface including two segments about user's name and password. And the left four tables is corresponding to the four android icon buttons and sets the appropriate segments respectively. Thereby, whenever the android application sends a data calling request, the database could query the corresponding table and return the requested data. In terms of how to choose the database, we must considerate at least three aspects as follows: the required volume, access rate, operating costs of the database. And we choose mysql as our database for the reason that it needs smaller volume and has rapid access rate. Most importantly it is the open-source comparing to the other databases like Sql-server, Oracle, Access and so on [5].

The floodlight restful application interface file provides a programmable interface to build a bridge which could connect the controller of SDN and the database to create effective crosstalk with each other [6]. Figure1 shows the original network architecture. In order to display its topology, firstly, we must make the floodlight controller and openflow switch running. After connecting, we could open an internet page and input the united resource location http://ip<floodlight controller>:8080:index.html If all is running normally and corrected, we will get the major page of floodlight topology displayed like Figure4. In this picture, we are shown the information of the switch and hosts' information like identity number and other items. If we have other switches, we could also know about its all information clearly. Openflow switches relocate the communicational path and determine the action of inner link according to the flow-tables from the floodlight controller which adopts TCP/IP protocol to connect the Internet service while adopts openflow protocol to connect openflow switches [7].

With the help of the action routine, the demand flow can be transported to the correct destination and indicate its action. From what we just said, we can transplant its thinking to android application. That is to say, we can use an android phone to know about the status of the SDN network such as the number of the connected switches and its flow-tables information. And we firmly believe we will just use an android phone to control the SDN network in some day.

3. The Implementation of the Data Interacting

We want to illustrate the communicating procedure of the data interacting and split it into two parts.

3.1 The Data Transporting From Controller to Database

As is previously mentioned, we select mysql as our database for the reason that it is more convenient, smaller and efficient to satisfy our requirements [8]. It is no doubt that the database has played an important role in this system. For the android phone, after creating the new database required to save the data from the floodlight controller, our android application could call the corresponding data from it. At last, we must configure the database including the access limits of authority, ip address, the connecting port and others. So, we can use the database for our application. Through the database, we can use it to save the information sent by the floodlight controller.

On the other side, we must increment codes to the floodlight source code to append required functions and prepare the data with appropriate format. These functions include saving data to the mysql database from openflow switches through the floodlight controller, updating data of the database in the real-time and detecting the status between the floodlight controller and openflow switches at least. Since the source code is programmed by the language of java, we must obey the rules of its grammar. First of all, we should considerate how to receive the data from the openflow switch and then how to process it. According to the openflow standard, the structure of packet_in and packet_out is used to transmit the data with the openflow protocol [9]. That is to say, we can take advantage of these structures and make the openflow switch organizing the data as the format of the structure. Then we exploit the openflow protocol to deliver the packaged data to the floodlight controller. After controller receives the packaged data, it must unpack those data information and get the useful portion. When we get the useful data, we must split it into many smaller parts to be able to add it to the database in the proper segments. Of course, before we do it, we must deal with the data passed up from the underlying network and add some business logics into the source code to have business functions. So we add a function into the java relevant class for the purpose of implementing this procedure. Finally, we pass the data information into the mysql database. In this area of coding, we add the mechanism of hibernate which provides some packages to operate the database easily and has higher working performance[4]. Also, we add another function in the same class which is responsible for constantly connecting the database and saving the treated data. In this part, we firstly program some codes to implement the connecting with the database. Secondly, we use some sql sentences and insert the treated data into the database.

After all coding, we could test what we have written and the result tells us that the floodlight controller could connect the database and the database could update the data from the floodlight controller. When we inquiry the database and choose the corresponding table, we will not only find that relevant segments are filled with the data and but also the data is coincided with that from the underlying network which demonstrates our tentative idea. That is to say, we have implemented transporting the data from the floodlight controller to the mysql database. And they work well.

3.2 The Data Transporting From Database to Application

We select eclipse as our developing tool and add android plugins of the software development kit. What effect we want is that when we open our android phone and run the application, we could connect the database and get the data to display in the visual way. For the android program, we first create an android project. In this project, we create some new classes. One of them is implementing the data interaction between the database and the android application. Others are implementing the sound visual effect what the

android phone displays to us. At first, we must create a new class to connect the mysql database. On the basis of it, we should create a method to inquiry the data of mysql database and we will use the structured query language. When we get the data, we deliver it to another method which implements to process the raw data and merge it to the proper format which the android application programming could use it. Another main method is to process the data and combines it to the useful data. Other classes such as the JLabel and JFrame are just to create the framework and buttons as previously addressed. The data that they use is from the processed data. When the android application connects to the database, we adopt the hiberlate mechanism to maintain the connecting. So we will import external classes like the hiberlate. Here, we utilize the well-known view model of MVC to simplify our programming work. In order to constantly maintain the connecting status once we run the application, we use a loop mechanism in the program to realize it and update the data of the android application from the database every five minutes with the consideration of user's experience and system resources. After finishing the application programming in the eclipse developing environment, we should compile the project and generate the executable file. Then we download it to the android phone. After installing the program in the android phone, we can open it and make it running. For the mysql database, since we just use five tables and don't use the complex architecture, the mysql will work very well and we need not to use the index technology and has to maintain at least two connecting requests at the same time. In order to be ready to receive the connecting requests at any time, we should select a separate machine as the database server which has installed the mysql database and is already configured. In one word, through these five dialogs, we could get overall information about the SDN network to make us manage and control SDN more effectively and conveniently.

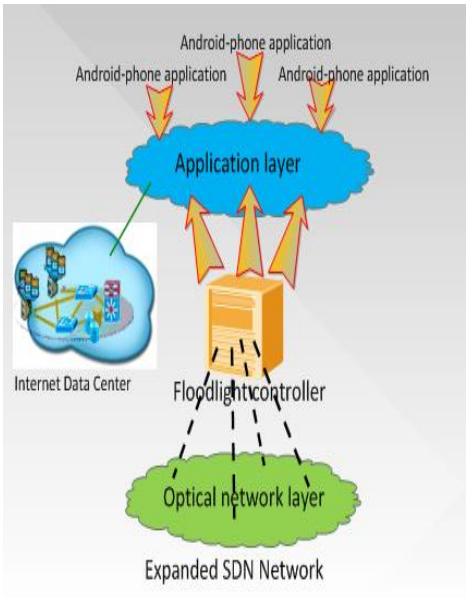

Figure 3. The Overall Designed Architecture

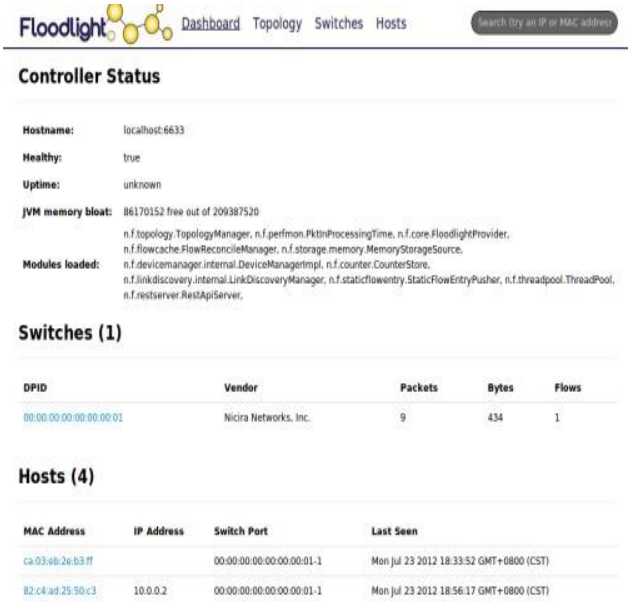

Figure 4. The Effect Of Network Page

Figure 3 shows the overall framework and the expanded architecture. The data center plays an important role as an intermediary connecting the floodlight controller to openflow switches and guaranteeing the data cross-talking normally between them. This paper selects mysql database as the data center. When the android phone connects the data center and requests the service, we could explicitly watch the sound experimental effect on the android phone. Every buttons will guide us to the corresponding interface and display the related information to us. Every step works as our previously expected effects and the responding state is very quick which will give users a good experience. From the global view, we build this application system according to the model shown by Figure3 and design the programming code according to it. Speaking of the architecture design, we expand our application business upper the application layer of the floodlight controller and create the android application which is the upper application expanding. This design is subordinate to the northbound interface expanding application. For this system, we creatively put forward to introduce the intermediary's rule for the first time in Software-Defined Network and give the implementing procedure. With this method, we could make the program more thread-securable to guarantee the effectiveness of data [10] and lower the bit error rate to improve the system reliability and performance.

4. Conclusions

As the study heat of the northbound interface continues to climb, more and more researchers attach much attention to it. After an in-depth study of the northbound interface technology [11], we provide the android application to view and monitor the underlying network in a visual way. In this paper, we successfully demonstrate the upper-layer application about the software-defined network and implement its upper-layer expanding application. After the floodlight controller connects the database and update the data of the database, we could use android phone to run the android application to inquiry the status of SDN network to make us knowlegeable of the built network. Thus we can manage it fully no matter where you are and when it is except for you have no authority to access the network. From the experiment results, Android phone can display clearly the topology and status of SDN network and the system could be running very well. It does not make you waiting too long time in the process to get the data and never make you feel sluggish response when you use your android phone to access the database resource or operate this android application. It also has little data displaying error according to the

actual situation Based on this system and related conclusions, we layout the research fundament of the northbound interface application.

Acknowledgements

In this procedure of writing this paper, many people give a lot of help and support especially in our group members. Our guiding teacher named by Xueguang Yuan urges me to read many papers of high level and often talks to us many recent development of SDN project. He indicates me the forwarding way on the research and makes me walking straightly to the target which is very important. My tutor named as Yang' An Zhang gives me great support and proposes many constructive suggestions when I contribute my paper to the summit conference. Secondly, I must show my deep gratitude to my sincere partner named by "Yan Hong", she helps me collect many paper materials and does much work in the design of the database mainly responsible for the connecting module. Besides, we have taken much time talking about implementing schemes and the technology details. Next, I should thank my dear parents who bring me up and give me the ability to work hard as well as to give me great confidence and active moving energy silently and selflessly behind me. At last, I should represent my appreciation to you, reviewers. It is you to read my paper and gives me suggestions to revise. All of the people mentioned above contribute to make the paper implemented. Although I have no ability to pay all of you back at once, I will bury the memory of your kindness into my heart deeply and do my best to work hard.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L Peterson, J. Rexford, S. Shenker and J. Turner, "OpenFlow: Enabling innovation in campus networks," Comput. Commun. Rev, vol. 38, no. 2, **(2008)**, pp. 69–74.
[2] A. Felix, N. Borges and H. Wu, "Multi-layer SDN on a commercial Network Control Platform for Packet Optical Networks", OFC, **(2014)**.
[3] W. Johnston, "Motivation, Design, Deployment and Evolution of a Guaranteed Bandwidth Network Service," Terena Networking Conference, Prague, Czech Republic, **(2011)** May 15-19.
[4] Y. Yu, Y. Lin, J Zhang, Y. Zhao, J. Han, H. Zheng, Y. Cui, M. Xiao and H. Li, "Field demonstration of datacenter resources migration via multi-domain software defined transport networks with multi-controller collaboration," in Optical Fiber Communication Conf. (OFC), **(2014)** March, pp. 1-3.
[5] A. Louri, "Three-dimensional optical architecture and dataparallel algorithms for massively parallel computing", IEEE Micro, vol. 11, **(1991)**, pp. 24-68.
[6] "The OpenFlow switch consortium," http://www.openflow.org/.
[7] Y. Yu, J. Zhang, Y. Zhao, S. Wang, H. Yang, H. Li and Y. Ji, "Open virtual infrastructure: Implementation framework for integrated provisioning of network and application resources based on software defined networking (SDN)", in 39th European Conf. and Exhibition on Optical Communication(ECOC), **(2013)** September, pp. 1-3.
[8] S. Das, *et al.*, "Parket and circuit network convergence with Openflow," in Proc, of OFC/NFOEC, OTuG1, **(2010)**.
[9] J. Han, H. Yang, Y. Lee and T. Ma, "Experimental demonstration of elastic optical networks based on enhanced software (eSDN) for data center application", Opt. Express, vol. 21, no. 22, **(2013)** November, pp. 26990 –27002.
[10] A. Farrel, "A unified control plane: dream or pipedream," in Proc, of IPOP, K-3, **(2010)**.
[11] X. Zhao, V. Vusirikala, B. Koley. V. Kamalov and T. Hofmeister, "The prospect of inter-data-center optical networks", IEEE Commun. Mag, vol. 51, no. 9, **(2013)** September, pp. 32-38.

Authors

Xuejun Tao, Graduate, Studying in Beijing University of Posts and Telecommunications. Research mainly on Network Management and Optical Communications.

Xueguang Yuan, Lecturer, Working in Beijing University of Posts and Telecommunications. Research mainly on Optical Network and Optical Communications.

Yan Hong, Graduate, Studying in Beijing University of Posts and Telecommunications. Research mainly on Polarized light and Optical Communications

Yang' An Zhang, Associate processor, working in Beijing University of Posts and Telecommunications. Research mainly on Optical Network and Embedded Systems