Journal of Theoretical and Applied Information Technology
15th July 2019. Vol.97. No 13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN SDN TESTBED FOR EVALUATING WIRELESS HETEROGENEOUS NETWORKS

[1]JUNHYUK PARK, [2]JANFIZZA BUKHARI, [3]WONYONG YOON

[1]Researcher. Department of Electronics Engineering, Busan, South Korea

[2]Researcher. National University of Sciences & Technology, Islamabad, Pakistan

[3]Professor. Department of Electronics Engineering, Busan, South Korea

E-mail: [1]88youiju@naver.com, [2]09bicsejbokhari@seecs.edu.pk, [3]wyyoon@dau.ac.kr (corresponding author)

ABSTRACT

This paper presents a proof-of-concept hybrid testbed based on Software-defined networking (SDN) for wireless heterogeneous networks (HetNets) which can be helpful for researchers to assess algorithms and protocols at a large-scale. This prototype uses open source state-of-the-art OpenDaylight controller based on SDN to control simulated OpenFlow switches and Network Simulator NS3 running network topology supporting LTE and WLAN technologies. The proposed testbed can be useful to renovate the HetNets, a feature provided by SDN, given the escalating demand of reliable bandwidth-intensive broadcast services. Targeting multiple radio access technologies (multi-RAT) users, we install Open vSwitch (OVS) on end user devices and use real TCP/IP protocol stack in Linux kernel to generate accurate results. We integrate our testbed to a VLC media server that allows to stream multimedia packets over simulated network and deploy a few use cases over configurable testbed to validate the functionality of our testbed in practice.

Keywords: *SDN, OpenDaylight, LTE, WLAN, Open vSwitch*

1. INTRODUCTION

SDN emerges as an efficient technique to facilitate innovations in cellular network architecture and protocol designs. SDN based wireless networks can be programmed using a software controller according to the requirements of network operators. In next generation wireless networks, multiple radio access technologies (multi-RATs) are anticipated to be deployed together at different frequencies constituting HetNets architectures. For instance, latest wireless technologies including Long-Term Evolution (LTE), WLAN etc. can incorporate together in order to achieve better coverage and connectivity. In SDN based HetNets, reliable broadcast services can be delivered to multi-RAT users through enhanced interworking. A state-of-the-art OpenFlow protocol [1] operate to facilitate such programmable interworking. In an OpenFlow-envisioned wireless network, configurable switches are installed on network forwarding nodes and control logic resides in separate SDN controller. The exchange of OpenFlow based control messages takes place between controller and switches to achieve reliable path among forwarding switches. The internal architecture of these configurable switches is also defined based on OpenFlow protocol. Since the SDN based controller fully manages the operation and intelligence of forwarding switches in wireless network, the accuracy and efficacy of the functions implemented by the controller must be fully validated before its use in a production environment.

For research and learning purposes, there are several tools available using which the performance of networks can be evaluated both in simulation and emulation form. Mininet is a well-known network emulator to prototype SDN based networks. It is capable of creating virtual hosts running standard Linux network software, switches, controllers and forwarding links through process-based virtualization. The links between virtual hosts and virtual switches are implemented by using virtual Ethernet pairs provided by the kernel. Being a simple and inexpensive solution for developing

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

OpenFlow applications, Mininet emulator is utilized in several works. [2] utilizes Mininet for constructing topology for Open Source Hybrid IP/SDN networks. [3] contributes to the extended functions of network topology emulation in Mininet based on Virtualbox, such as Internet connection, independent OS clients, standard routing controller, and automatic flow logging. Without wiring up a physical network, any network topology can be emulated and tested using Mininet, however, it is not enough to support network dynamicity and performance of virtualized hosts. Furthermore, Mininet does not support wireless connection but this constraint can be compensated by integrating it with the simulation feature of Network Simulator 3 (NS3) [4]. However, such integration should be well tested with real SDN controller for its functionality validation. Other than Mininet, popular experimentation tool for networks' performance assessment is NS3 which is an open source project, actively being developed and maintained [5]. Besides other network technologies, NS3 offers OpenFlow module to simulate SDN wireless network, however, it provides an outdated protocol version i.e. 0.8.9. The work in [6] presents 'OFSwitch13' module compatible with NS3 enhancing its capability to support OpenFlow version 1.3. The source code for switch device and a controller application interface is also provided.

In SDN based HetNets, the controller acts as brain of the network, responsible for taking smart routing decisions and pushing these forwarding rules on switches in underlying network. For research and commercial purposes, there are a number of controllers available including Floodlight, NOX, POX, Ryu, Beacon, OpenDaylight, ONOS etc. However, it is essential to choose an open source state-of-the-art controller for making the prototype of the testbed in order to test and debug the SDN based network. For our production environment, we prefer to choose real OpenDaylight controller which is the largest open source SDN controller founded by Linux community [7]. It offers a new networking paradigm which decouples forwarding hardware from control decisions for better managing the topology, traffic flow in switch, port, traffic scheduling and load balancing etc. In this paper, we suggest to implement the OpenDaylight based hybrid experimental testbed with Open vSwitch (OVS) based on Linux in order to virtualize real network nodes, providing end-to-end centralized controllability of flow. OVS is an open source and distributed virtual multilayer switch which provides a switching stack for hardware virtualization environments [8]. We connect OVS to our simulation setup to model and execute the SDN based operations of HetNets nodes and their interactions. Our fully SDN network testbed uses real OpenDaylight controller, simulated HetNets with customized topology connected to real end user devices running OVS, real multimedia server providing video content and therefore, it is fully capable of generating more realistic testing results. Through this work, we intend to fill the research gaps in testing environments for SDN based wireless networks. We describe how an SDN platform is developed and set up to test various network scenarios. It mainly includes performance testing while broadcasting the bandwidth-intensive multimedia data among configurable multi-radio end devices. It also intends to address how network congestion is tackled by offloading traffic to other available network.

The rest of the paper is organized as follows. In the next section, we shall discuss some existing testbeds available to assess the performance of networks; both wired and wireless. Then, we will illustrate the design of our prototype, its architecture and implementation details. At the end, we will discuss a few use cases and possible demonstrations followed by summary of our work.

2. EXISTING SDN SIMULATION TESTBEDS

Utilizing the benefits of open source tools, researchers and developers can easily design and deploy new services and test the SDN control plane solutions with a minimal effort. In past, various platforms are deployed to monitor the performance of SDN based networks. EstiNet is a dedicated SDN suite of simulator and emulator [9]. It enables real TCP connection between controller and simulated switches and between two real applications, exchanged packets are automatically intercepted through tunnel network interfaces and are redirected into the simulation engine. It allows real controllers such as NOX/POX or Floodlight to control thousands of its simulated OpenFlow switches. It provides the benefits of both the simulators and emulators when compared with other tools and found to be more scalable for large networks than Mininet. However, unlike our work, EstiNet is not an open source application. It can neither be extended by altering the source code, nor focuses on the cellular wireless networks. Consider another SDN testbed in [10] implemented with OVS based on Raspberry-pi. It uses Floodlight

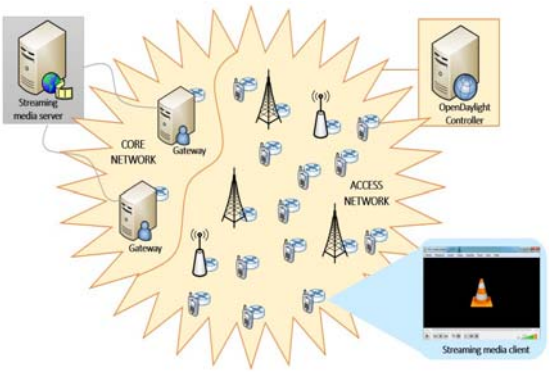
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

controller with OpenFlow specification 1.0, however, like EstiNet, it is also used to evaluate wired networks only. Extending the centralized control model of SDN, WE-Bridge [11] is designed which exchanges virtual network views between different domains. To monitor inter-domain traffic among four real wired SDN networks, two routing applications are deployed. Using these applications, controller abstracts the virtual network view of each individual network. Among different domains, this virtual network view is exchanged, based on which each network constructs a relative global network view. The testbed in [12] provides a physical OpenStack deployment with OpenDaylight controller. A customizable network topology can be emulated using configurable OVS in CORE (Common Open Research Emulator). The realistic background traffic generated using DCT2Gen is injected into the emulator machine. A large number of nodes can be emulated by using virtual containers however, it can create configurable background load into the configurable network topology within CORE.

For SDN based wireless networking, a few efforts developing the experimentation platforms have been done. [13] provides a testbed for wireless mesh network where APs are developed on open source wireless router platform (OpenWRT). The OVS installed on Raspberry Pi devices provide their respective interfaces to the controller. Only one AP is directly connected to the Internet and rest of the APs can access the internet via this particular AP over wireless medium. For high performance gains, these APs can change their routing path only if they have more than one neighbor AP. The throughput at smartphone end is observed when the number of hops is minimum or maximum. Another practical test environment is set up using Raspberry Pi as OpenFlow switch and WLAN AP [14]. It consists of three OVS switches and two WLAN APs. The user groups are classified with different priorities depending on their characteristics and based on this priority, WLAN AP access control is provided. In case of congestion, controller handles the WLAN AP access mode based on the flow table and high priority users can be prioritized. [15] is another testbed deployed in the form of a hypercube mesh containing heterogeneous wireless technologies i.e. mmWave, WiFi etc. in order to test the control of a wireless mesh backhaul. In this testbed, Wireless Backhaul Forwarding Elements carry out the data plane forwarding while the control plane can be wired or wireless depending on the configuration. OpenDaylight is used as a controller in [16] and a real-time testbed is realized using Raspberry Pi with WiFi dongles as OpenFlow Switches. In this work, a traffic aware routing algorithm add-on is implemented to manage the flow with respect to the QoS requirement. Contrary to our work, most of the existing SDN wireless testbed platforms target the WLAN access technology only.

Figure 1: Envisioned software-defined wireless HetNet.

Focusing the LTE/WLAN multi-RAT users, [17] presents a deployable HetNet architecture in which a client side utility runs as a service on UE. The controller maintains UE's RAT-independent IP



address and facilitates UDP-based control and data paths to users. Using this experimental setup, congestion is emulated in LTE data path which is identified by the controller and flow is offloaded from LTE to WiFi. However, this work does not provide the user space implementation of client side utility on smart devices. In [4], an effort has been made to create a network emulator suitable for SDN based wireless meshed network, combining Mininet and WiFi module of NS3. In this work, Mininet provides virtual hosts and mock controller for monitoring and testing the routing over multi-hop meshed sensor network nodes. However, testing with real controller is still lacking in [4]. Table 1 shows the existing testbeds along with the details of their network scenarios and the experimental setups used for network creation. The works in [9], [10], [11] and [12] focus on wired networks, among which [10] an [11] use real network devices instead of simulation nodes, which incurs deployment cost. The work in [13], [14] and [16] target the real WLAN networks, instead of heterogeneous networks, which is the area of our interest. Though [15] and [17] contribute in testing real heterogeneous networks, among which [17] discusses cellular HetNets with LTE and WLAN, however, the controller details are not given. Unlike

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

our work, [4] does not use real controller while emulating LTE and WLAN nodes.

In contrast to the aforementioned platforms, we offer a low-cost testbed which incorporates both the programmable base stations and WLAN access points in a virtualized environment supporting various heterogeneous test scenarios. We believe that using simulated network nodes for testing various network settings and evaluating their effects is cost-effective as compared to the use of real devices in experimentations. We incorporate real OpenDaylight controller to our testbed which can operate on updated version of OpenFlow protocol. This testbed is capable to evaluate the network operations of wireless HetNets with configurable OVS switches, by providing video data through a real-time streaming media server. We explore the applicability of developed SDN wireless testbed and use real video packets for transmission over TCP and UDP for testing purposes. We deploy scenarios of offloading from LTE to WLAN and vice versa in the supervision of real OpenDaylight controller. Moreover, we also investigate the impact of number of users on the performance of heterogeneous cellular networks, in order to validate the practicality of our production environment.

3. PROPOSED SDN SIMULATION TESTBED FOR WIRELESS HETEROGENEOUS NETWORK

Consider Figure 1 depicting fully programmable wireless HetNet composed of LTE and WLAN access technologies. The SDN is expected to provide end-to-end controllability of wireless network where programmatic interfaces are used to interact with the radio data plane consisting of base stations (eNodeBs), access points (APs), and so on. To investigate the virtual network management software solutions and their applicability to real world deployments, we provide the design of a testbed for SDN based wireless heterogeneous network which has the following features:

(1) execution of real OpenDaylight controller code

(2) incorporating OpenDaylight controller in the simulation

(3) integration of NS3 based simulated HetNet with OVS

(4) connection of testbed with real multi-RAT physical end user device

(5) transmission of multimedia data provided by real server to end client node with OVS installed

(6) providing an environment enabled for open source collaboration.

3.1 Architectural Design

The design of proposed testbed is illustrated in Figure 2. The main components of this prototype are i) VLC Streaming media server developed by the VideoLAN project ii) OpenDaylight Controller running on Linux machine iii) NS3 simulator with custom network topology iv) Client nodes with OVS. In NS3 simulator, we design a HetNet with one Remote Host connected to multimedia server for distributing video content over simulated HetNet, one External Controller node, one LTE gateway, one interworking gateway to transmit data over WLAN, one AP, one eNodeB and multiple user nodes (UEs). We connect each simulated multi-RAT UE to separate real Linux machines with OVS switches and install a VLC client application on each of them, representing the physical controllable multimedia clients. The OpenDaylight controller application runs on simulated External Controller node and monitors simulated OpenFlow switches running OpenFlow protocol version 1.3 in NS3. We use Linux TAP interfaces to link real OpenDaylight, multimedia server and physical clients with NS3 simulator. This presented testbed has the potential to re-use the existing protocols and application stacks, modeling the behavior of real wireless networks. Using this prototype, we aim to execute realistic experiments transmitting the broadcast services to SDN enabled configurable end devices.

3.2 Implementation Details

3.2.1 OpenDaylight controller and its visualization

OpenDaylight is a commercial project that exploits model-driven software engineering principles for altering the networks. Its layered infrastructure supports multiple plugins including OpenFlow plugin supporting the implementations of various specifications of OpenFlow protocol. In our prototype, we use protocol version 1.3, however, OpenDaylight can be extended to add support for subsequent OpenFlow specifications. This is why,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

we have chosen it as a controller for our testbed. We run OpenDaylight code version 0.5.3-Boron-SR3 for our production environment without any modifications using karaf. It allows communication with other physical devices running OpenFlow or any other open protocols. However, any latest version of OpenDaylight is equally productive in our setup. As a stand-alone application, it interacts with simulated HetNet in NS3 as discussed below.

3.2.2 NS3 connection with OpenDaylight

We develop an External Controller node within NS3 domain which is connected to real OpenDaylight controller using Linux Tap net device. This Tap Bridge between Linux Tap net device and CsmaNetDevice operates in default operating mode i.e. ConfigureLocal which means this interface is NS3 configuration centric and connection of External Controller node to OpenDaylight is made during simulation time. When the simulation starts, it creates a bridge (i.e. tap-ctrl) on the underlying Linux operating system between real OpenDaylight and simulated External Controller node and assigns the same IP and MAC addresses to this bridge as the simulated External Controller node has. This is how we connect real OpenDaylight to our simulation environment. Within NS3, we realize real controller as an application running at simulated External Controller node.

As discussed earlier, OFSwitch13 module is provided to support NS3 simulations with OpenFlow switch operating on protocol version 1.3. On the network nodes within NS3, we add the code to install OFSwitch13Device to enable configurability in them. These programmable nodes interact with each other through CsmaNetDevice and to interface them to OpenDaylight, we exploit an externally compiled ofsoftswitch13 library [18] which implements the switch datapaths including input/output ports, flow tables and pipeline for packet matching, group table etc. We manually compile and install another dependent library called NetBee which is used to decode and parse the network packets for ofsoftswitch13. In our testbed, we use stable OFSwitch13 module version 3.3.0 and NS3 version 3.28.

3.2.3 Simulated HetNet supporting multi-RAT UEs

Realizing the vision of wireless HetNet based on SDN, we simulate the aforementioned topology along with multi-RAT UE nodes in NS3 environment, supporting LTE and WLAN. For WLAN facility, we add WifiNetDevice on AP and on UE node as well. On same UE, we add LteNetDevice in order to incorporate LTE, making it a multi-RAT node. LteEnbNetDevice is added on eNodeB to disseminate multimedia data among UEs via LTE technology. We add callbacks in NS3 code to send packets to OFSwitch13Device from LteEnbNetDevice/WifiNetDevice and vice versa. We make multi-RAT simulated UE configurable by OpenDaylight controller as explained in the following.

3.2.4 Installation of OVS on client node and its connection to controller

We create an OVS bridge on Machine 3 running Linux, add two virtual ports (vPorts) on it as shown in Figure 2 and assign IP addresses to these virtual interfaces. During its configuration, we attach one of the physical interfaces eth to virtual OVS bridge (consider it as control port) and manually assign IP address to OVS (i.e. 168.115.126.10) as the connected eth interface does not have any IP. We set up OpenDaylight as the controller of this OVS by setting its IP address and port number to the value where OpenDaylight is listening (i.e. 168.115.127.100:6653). We also provide route to access controller by adding an entry in existing routing table in Linux kernel. The GUI of OpenDaylight controller confirms the availability of this switch. Integrating multiple multi-RAT users, we can install multiple OVS switches, each on separate Linux machines and can make their connection to the controller.

3.2.5 Interfacing configurable multi-RAT client to NS3

To interface simulated UE to real physical device running OVS, we use Linux Tap interfaces (tap-w, tap-l) in UseLocal mode and apply network address translation (NAT) rules on them to forward data to corresponding real eth interfaces of the same Linux machine, (Machine 2 as depicted in Figure 2). Each interface showcase a radio access technology and NAT rules are applied accordingly. Using Ethernet crossover cables, we join these interfaces to real eth interfaces of Machine 3, where OVS is already installed. We apply destination NAT routing (DNAT) which modifies the destination address of the incoming packet in order to transmit packets ahead to vPorts (i.e. ovs-br0-p0, ovs-br0-p1). To achieve that, in the NAT rule, we specify the destination IP address (i.e. vPort address) where

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

incoming packets can be forwarded. Moreover, we need to add entries in the kernel's routing table to identify the packets for particular interfaces of Machine 3.

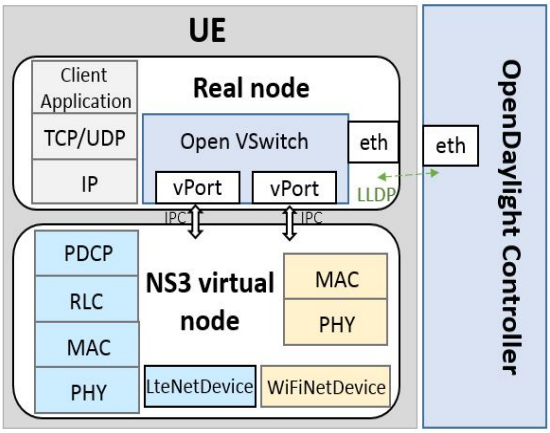


Figure 3: Programmable UE node multi-radio protocol stack in testbed.

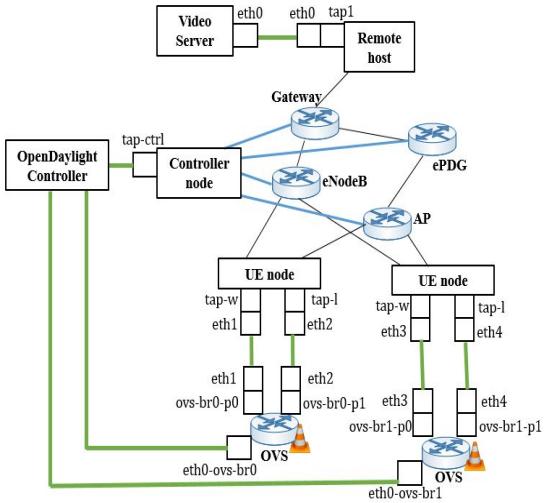


Figure 4: Simulated topology realized by the testbed.

Figure 3 illustrates the programmable UE node and a representation of its multi-radio protocol stack in the testbed. We use TCP/IP stack of real UE node and through inter-process communication via crossover cables, packets are exchanged between PDCP/LLC and IP layer of protocol stack used. The link discovery (LLDP) packets are forwarded through control eth port of OVS (e.g. eth0-ovs-br0) to simulated UE node and vice versa. Figure 4 depicts the exact interfaces, nodes and the overall HetNet topology provided by the proposed testbed.

3.2.6 Real multimedia traffic generation through a streaming server

We attach Linux Tap net device to simulated Remote Host node and apply DNAT routing in order to connect it to the traffic generator. We generate real traffic through VLC Server application running on a separate Windows machine broadcasting to the network at some port via UDP (168.115.127.100:1234). To receive this multimedia traffic in NS3, we install a server application at Remote Host listening to port 1234 which streams the multimedia packets over simulated wireless HetNet. Additionally, we install a client application on simulated UE node in order to receive the incoming packets. We run a VLC client application on end physical device to validate the reception of broadcasted video stream at port 8080 via SDN based wireless HetNet.

4. VALIDATION RESULTS

The connection of real OpenDaylight controller to simulated HetNet and exchange of OpenFlow based messages between controller and switches to visualize the heterogeneous network can be validated by viewing the topology in GUI provided by OpenDaylight. All the switches (both virtual and real) appear in the GUI along with the connections and the assigned addresses. From an experimentation standpoint, we address the gaps in implementation of a low-cost SDN wireless testbed and can showcase its applicability underpinning a variety of use case scenarios.

4.1 Multiple flows

We conduct experiments allowing both the radios of users to stream multimedia data simultaneously, and call it n-casting in this paper. The eNodeB has downlink transmission bandwidth configuration equals to 25, where it is the number of resource blocks available. The data mode of AP is configured as constant using ConstantRateWifiManager and we set the data rate as ofdmRate3MbpsBW10MHz for our simulation.

The video packets are coming from VLC server deployed at Machine 1. Before the start of simulation, multimedia data does not appear at TAP interface attached to Remote Host node, although the server is transmitting the video stream. During simulation time, using DNAT, these packets are forwarded to simulated HetNet with IP of server as source address of packets. The video stream is n-casted via eNodeB and AP switches and received by the simulated UE node in NS3. The simulated UE forwards the incoming packets to IP layer of

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

real protocol stack in Machine 3. The OpenDaylight controllable software switch sends the packets to PDCP layer of virtual node in NS3. We can watch the streamed video in VLC client application in Machine 3 using IP address of ovs-br0-p0 or ovs-br0-p1. Figure 5 is depicting the link utilization of multiple users connected to HetNet via LTE and WLAN. The average throughput in LTE is measured as 59, 32 and 20 packets/second when number of users in HetNet are 1, 3 and 6 respectively. For WLAN, average throughput of user comes out to be 85, 76 and 65 packets/second when connected users are 1, 3 and 6 respectively.

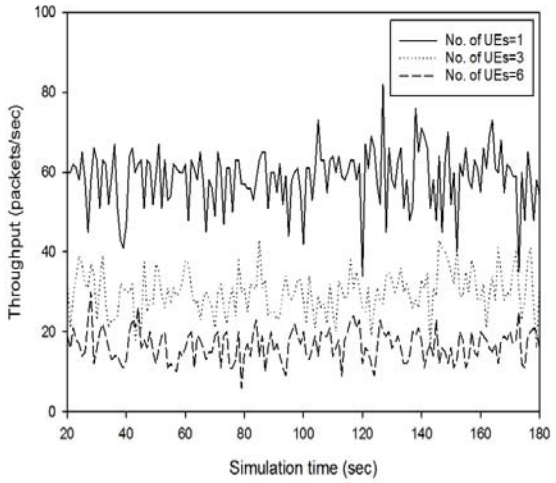


Figure 5(a): Measurement of LTE link utilization when multiple users are connected to the HetNet.

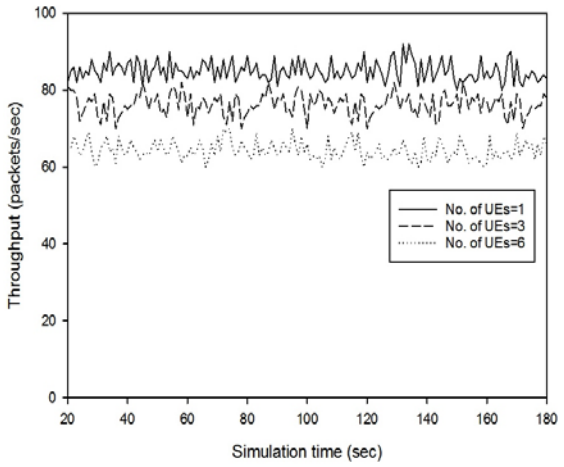


Figure 5(b): Measurement of WLAN link utilization when multiple users are connected to the HetNet.

4.2 Interworking scenario of multi-RAT UEs

Consider the scenario where six multi-RAT UEs are watching the video through LTE interface and congestion is posed on LTE network which is identified by the UE. Similarly, if the distance between UE and base station increases, its throughput is lowered. We program one of the UEs to get the data traffic re-routed to the available WLAN via interworking gateway ePDG switch. We can watch the video in VLC client application running on physical end device (Machine 3) after some interruption caused due to handover to WLAN, as shown in Figure 6. The offloading delay is approximately calculated as 3.4 seconds.

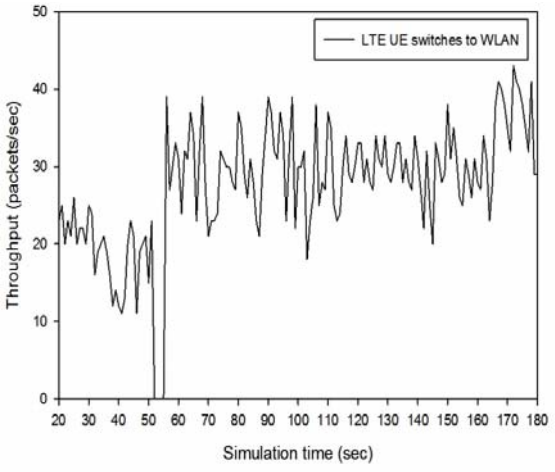


Figure 6: Throughput measurement during offloading.

4.3 Video streaming over TCP protocol

We provide streamed video over TCP instead of UDP in this scenario and demonstrate the throughput of a multi-RAT UE in Figure 7, when six UEs are connected to HetNet. To achieve that, we configure the VLC media server application to make it use RTP over RTSP (TCP) for streaming.

The average user throughput while streaming over TCP protocol comes out to be 54 packets/second whereas in UDP case, average throughput is 56 packets/second. We observe that streamed video is pre-fetched and buffered in the TCP protocol case and a smooth play-out is noted. The multimedia data is successfully broadcasted among configurable UE nodes, confirming the testbed's functionality. Providing broadcast services to users via multiple RATs increases the coverage area and it is useful for better connectivity in case when LTE connection is unfavorable or WLAN AP is inaccessible.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

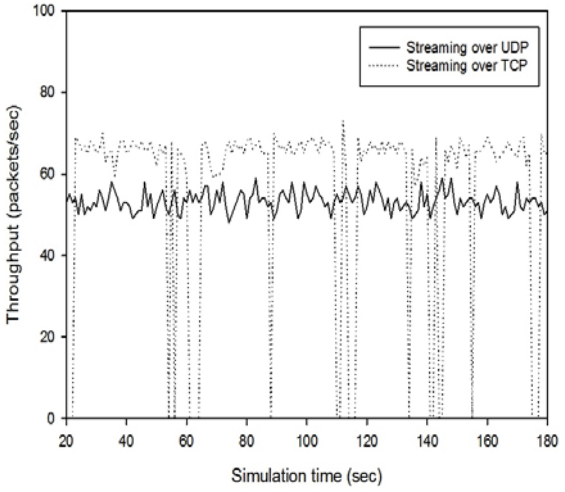


Figure 7: User throughput measurement while streaming the video data packets over different protocols.

5. CONCLUSION

For realistic and reproducible research, we implement a SDN wireless testbed that provides a cost-effective environment in the experimental platform space, leveraging the benefits of simulated network nodes. The real video packets are streamed over TCP and UDP through simulated heterogeneous network in NS3. The real OpenDaylight controller application monitors the delivery of these packets to configurable real end devices. We provide evidence in the form of results to show that this proposed testbed is a completely flexible platform which can be used to evaluate the responsiveness of HetNets according to customers' requirements. However, these network devices need to be carefully and accurately configured within the simulation, avoiding the results to go far from reality.

In this paper, we opt the approach of combining virtual nodes in simulated time and OVS running OpenFlow protocol version 1.3 in real-time. We model the operations of real HetNet nodes and their interactions in simulation environment. We believe it to be a viable approach being low cost, scalable and faster than real time in many cases. For instance, different network settings can be configured on simulated HetNets without extra operational cost. As promising directions, we can construct customized topologies within NS3, realizing network of any size. Moreover, we can integrate lightweight physical devices i.e. smartphones to our testbed, allowing the network operators to directly configure it and feed the decisions based on the network conditions. We can evaluate the performance of OpenFlow standard and its network procedures. Furthermore, we can perform experimental tests by modifying the transmission bandwidths, mobility patterns, change the speed of mobile users during handover to analyze the results and investigate the impact on network performance including data rate, packet loss rate etc.

ACKNOWLEDGMENT:

This work was supported by the Dong-A University Research Fund. J. Park and J. Bukhari are co-first authors who equally contributed to the paper. The corresponding author is Wonyong Yoon.

REFRENCES:

[1] ONF, "OpenFlow Switch Specification Version 1.5.1, ONF Standard TS-025," 2015.

[2] S. Salsano, P. L. Ventre, F. Lombardo, G. Siracusano, M. Gerola, E. Salvadori, M. Santuari, M. Campanella and L. Prete, "Hybrid IP/SDN Networking: Open Implementation and Experiment Management Tools," *IEEE Trans. Network and Service Management*, vol 13, no. 1, Mar. 2016, pp. 138-153.

[3] N.-N. Dao, Q. D. Tran, M. Park and S. Cho, "SDNbox: A portable open-source testbed for SDN study," *Proc. Int. Conf. Information and Communication Technology Convergence (ICTC)*, Jeju Korea, Oct. 2017, Jeju, pp. 829-833.

[4] M. Pieskä, "Emulating Software-Defined Small-Cell Wireless Mesh Networks Using ns-3 and Mininet," *Master's Thesis*, Karlstad University Sweden, June 2018.

[5] "ns-3: a discrete-event network simulator for internet systems," Available from: https://www.nsnam.org/.

[6] L. J. Chaves, I. C. Garcia, and E. R. M. Madeira, "OFSwitch13: Enhancing ns-3 with OpenFlow 1.3 Support," *Proc. ACM Workshop on ns-3 (WNS3 '16)*, USA, June 2016, pp. 33-40.

[7] "OpenDaylight," Available from: https://www.opendaylight.org/.

[8] "Open vSwitch," Available from: http://openvswitch.org.

[9] S.-Y. Wang, C.-L. Chou and C.-M. Yang, "EstiNet openflow network simulator and emulator," *IEEE Communications Mag.*, vol 51, no. 9, Sep. 2013, pp. 110-117.

[10] H. Kim, J. Kim and Y.-B. Ko, "Developing a cost-effective OpenFlow testbed for small-scale Software Defined Networking," *Proc. Int. Conf. Advanced Communication Technology*, Pyeongchang, Feb. 2014, pp. 758-761.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[11] P. Lin, J. Bi, S. Wolff, Y. Wang, A. Xu, Z. Chen, H. Hu and Y. Lin, "A West-East Bridge Based SDN Inter-Domain Testbed," *IEEE Communications Mag.*, vol 53, no. 2, Feb. 2015, pp. 190-197.

[12] C. H. Benet, R. Nasim, K. A. Noghani and A. Kassler, "OpenStackEmu — A cloud testbed combining network emulation with OpenStack and SDN," *Proc. IEEE Annual Consumer Communications & Networking Conf. (CCNC)*, Jan. 2017, pp. 566-568.

[13] W. J. Lee, J. W Shin, H. Y. Lee and M. Y Chung, "Testbed implementation for routing WLAN traffic in software defined wireless mesh network," *Proc. Int. Conf. Ubiquitous and Future Networks (ICUFN)*, Aug. 2016, pp. 1052-1055.

[14] J. W. Shin, H. Y. Lee, W. J. Lee and M. Y. Chung, "Access control with ONOS controller in the SDN based WLAN testbed," *Proc. Int. Conf. Ubiquitous and Future Networks (ICUFN)*, July 2016, pp. 656-660.

[15] J. N.-Martínez, J. Baranda, I. Pascual and J. M.-Bafalluy, "WiseHAUL: An SDN-empowered Wireless Small Cell Backhaul testbed," *IEEE Int. Symp. on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2016, pp. 1-3.

[16] M. Ariman, G. Secinti, M. Erel and B. Canberk, "Software Defined Wireless Network Testbed using Raspberry Pi OF Switches with Routing Add-on," *Proc. IEEE Conf. Network Function Virtualization and Software Defined Networks (Demo Track)*, Nov. 2015, pp. 20-21.

[17] V. Sagar, R. Chandramouli and K. P. Subbalakshmi, "Software defined access for HetNets," *IEEE Communications Mag.*, vol 51, no. 1, Jan. 2016, pp. 84-89.

[18] E. L. Fernandes and C. E. Rothenberg, "OpenFlow 1.3 Software Switch," *Brazilian Symp. on Computer Networks and Distributed Systems (SBRC)*, Brazil, May 2014, pp. 1021-1028.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1: Existing production environments for SDN based networks.

Literature	Controller	Core Network Devices		Network Type
		Type	Simulation/Emulation Software	
[9]	NOX/POX, Floodlight	simulated, emulated	EstiNet	Wired
[10]	Floodlight	real		Wired
[11]	Floodlight	real		Wired
[12]	OpenDaylight	simulated	CORE	Wired
[13]	ONOS	real		Wireless (WLAN)
[14]	ONOS	real		Wireless (WLAN)
[16]	OpenDaylight	real		Wireless (WLAN)
[15]	Ryu	real		Wireless (mmWave, WLAN)
[17]	not mentioned	real		Wireless (LTE, WLAN)
[4]	Mininet	emulated	NS3, Mininet	Wireless (LTE, WLAN)
Our work	OpenDaylight	simulated	NS3	Wireless (LTE, WLAN)

Journal of Theoretical and Applied Information Technology
15th July 2019. Vol.97. No 13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

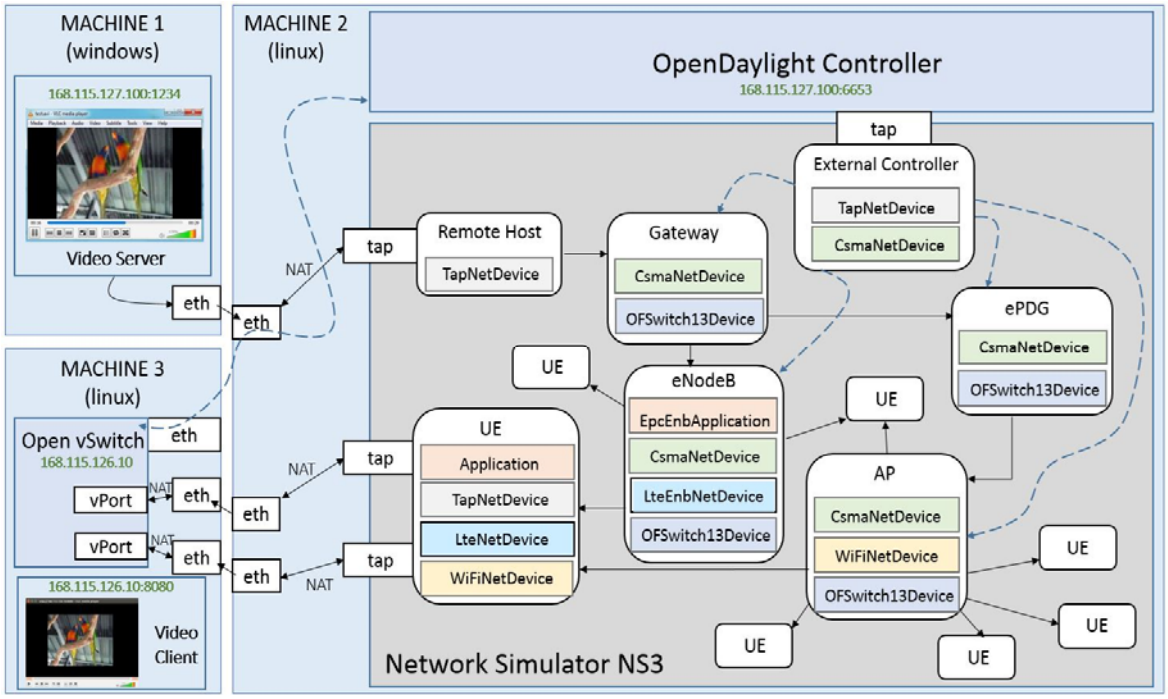


Figure 2: Proposed architecture of our experimentation testbed.