

DEVELOPMENT OF SDN-BASED WI-FI AP USING OPENWRT AND RASPBERRY PI 3

Hai-Chau Le* and Khac-Tuan Nguyen

Posts and Telecommunications Institute of Technology, Hanoi, Vietnam

* Corresponding author: chaulh@ptit.edu.vn

Abstract: We have successfully developed a prototype of SDN-based Wi-Fi access point device for IoT communication. The proposed device, that is based on Raspberry pi 3, uses Linux-based OpenWrt firmware and Open vSwitch software. It is compatible with OpenFlow 1.5 and includes one built-in 100 Mbps Ethernet port. Thanks to taking the advantages of SDN technology, open source software and low-cost mini-computer, the developed SDN-based device has a reasonable cost while it is flexible, scalable and still fully supports advanced network functions. The device also can be deployed as a network device with various functions such as hub, switch, firewall, ... which is determined by a controlling program installed in the SDN controller. The AP configuration and its performance have been verified by testing experiments. The obtained numerical results proved the effective and scalable performance of the developed SDN-based Wi-Fi AP for next generation WLAN and IoT communication infrastructure.

Keywords: Software-defined networking, Wi-Fi access point, OVS, OpenFlow.

I. INTRODUCTION

Recently, the rapid emergence of Internet of Things leverages the promise of many important new benefits but it also poses some potential challenges and issues [1-3]. To support a vast number of connected devices with heterogeneous characteristics, IoT communication infrastructure needs new architectures and advanced device technologies that are able to accommodate the explosively ever-increasing, bandwidth-flexible traffic. New communication technologies are, hence, required to consume less power, need low bandwidth, have low computation power and provide mobility and seamless communication [3-4]. Furthermore, those devices are also required to be cost-effective, flexible and scalable for providing IoT heterogeneous services with various

demands on quality of service. In fact, software-defined networking (SDN) that has been developed to support virtualizing and provisioning dynamically network resources in an on-demand manner are capable of handling those challenges and requirements [5-6]. SDN offers an abstraction of underlying network to applications residing in upper layers and implements the control plane as a software component while locating data plane in the network devices [7-9]. The networking functionalities of SDN devices are replaced by just forwarding devices while the intelligence of where and how to make forwarding is placing in control plane in software form (called controller). These approaches help to transform the network resources into programmable, automation and network control, highly scalable and flexible networks based on the business needs [6, 7]. Hence, SDN technology in the context of IoT communication devices (SD-IoT) have recently attracted some attention [9-11].

On the other hand, wireless local area networks (WLANs), one of the key technologies for IoT communication using unlicensed radio bands, have been used widely in enterprises, campuses, offices, and home environments to support mobile devices through access points (APs) [12-14]. After attaching to an IEEE 802.11 AP, a mobile device can send and receive packets via the wired infrastructure through WLAN. Unlike thin APs which are usually installed with licensed firmware and deployed in large enterprises, fat APs are widely deployed in homes and small offices to provide wireless connection to user devices and offer router-like network services such as dynamic host configuration protocols (DHCP) and network address translation (NAT), ... Existing WLAN unified access platforms are costly and closed systems, and they typically include LAN switches, WLAN controllers, and APs as the entire solution supplied by a single vendor [15, 16]. Up to now, some commercially available software-defined networking (SDN) WLAN solutions have been announced, i.e. HP acquired Aruba Networks,

Tác giả liên hệ: Lê Hải Châu
Email: chaulh@ptit.edu.vn

Đến tòa soạn: 2/2020, chỉnh sửa: 4/2020, chấp nhận đăng: 4/2020

Fortinet acquired Meru Networks, ... [17-19]. However, solutions by vendors are proprietary and it is difficult to add new features.

In this paper, we study a feasible software-defined networking WLAN solution that leverages OpenFlow technique and provides affordable centralized WLAN service. We have developed a prototype of SDN based Wi-Fi AP for IoT communication. The developed SDN-based Wi-Fi AP is based on Raspberry pi 3 integrated with OpenWrt firmware [20] and Open vSwitch software. Because of the use of SDN technology, open source software and low-cost mini-computer, our developed device has a reasonable cost while it is flexible, scalable and still fully supports advanced network functions. We have also verified and tested the designed configuration and performance of the developed device. The experimental results proved the effective and scalable performance of the developed SDN-based Wi-Fi AP system for IoT communication infrastructure.

II. PROTOTYPE OF SDN-BASED WI-FI AP WITH OPENWRT AND RASPBERRY PI

In this work, we target a small-scale, cost-effective and OpenFlow-enable Wi-Fi access point device that is flexible and scalable for IoT communication infrastructure in near future. In order to realize a low-cost and flexible device, we exploit multi-purpose hardware platforms, open-source software and SDN technologies. We build a system with an open source firmware, i.e. OpenWrt [20], and a commercial off-the-shelf hardware, Raspberry pi 3 [21, 22], and develop a mechanism to control wireless bandwidth by using OpenFlow technique [23, 24]. Figure 1 shows the functional design of our developed SDN-based Wi-Fi AP prototype that is based on a Raspberry pi 3, OpenWrt firmware and OVS software.

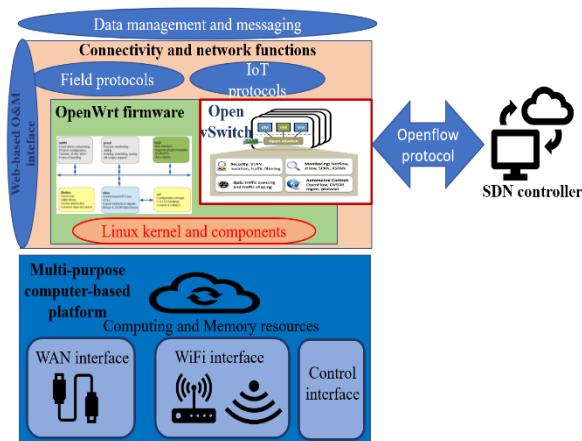


Figure 1. Functional architecture of SDN-based Wi-Fi AP.

Up to now, many multi-purpose mini-computer systems have commercially introduced for a wide range of applications with a reasonable price, such as Raspberry pi, IBM NUC, ... [19, 21]. Among those devices, Raspberry pi 3 is one of the most popular and cheapest devices [22], especially in Vietnam. It supports a great configuration including 1.2 GHz CPU, 1 Gbyte RAM, one built-in Wi-Fi

card 802.11b/g/n [25-27], one built-in 100 Mbps Ethernet port and 4 USB ports. These features meet the requirements of our targeted SDN-based Wi-Fi access point perfectly.

In addition, OpenWrt firmware which is an open source project for embedded operating system based on Linux, primarily used on embedded devices to route network traffic [20]. The OpenWrt project started in January 2004 and the first OpenWrt versions were based on Linksys GPL sources for WRT54G and a buildroot from the uClibc project. This version was known as OpenWrt "stable release" and was widely in use. OpenWrt uses official GNU/Linux kernel sources and only adds patches for the system on chip and drivers for the network interfaces. The developer team tries to re-implement most of the proprietary code inside the GPL tarballs of the different vendors. There are free tools for writing new firmware images directly into the flash, for configuring the wireless LAN chip and to program the VLAN-capable switch via the proc filesystem [20]. Main components of OpenWrt are Linux, util-linux, musl, ... and OpenWrt components have been optimized to be small enough to fit into the limited storage and memory available in home-devices or mini-computers. It can be a great operating system option for small-size and low-cost wireless devices such as Wi-Fi access point, wireless router, ... Hence, in our work, we deploy OpenWrt/LEDE to realize the prototype of our SDN-based Wi-Fi access point. Even security is one of the critical issues for wireless APs, it is out of the paper scope thanks to using open-source OpenWrt/LEDE which is updating continuously.

On the other hand, in order to leverage SDN technology and provide affordable, flexible WLAN Wi-Fi access point, our device has been implemented and deployed in the way where network administrators can exploit the extensibility and flexibility of OpenFlow-based SDN switches and controller and they can also develop customized application modules and required features. An important component to enable SDN technology in the device is open source software named Open vSwitch (OVS). Even there are several open-source SDN switching software including Openflow switch, OVS, ... [24], OVS is one of the most promising ones thanks to its high compatibility and well-developed modules. Open vSwitch that can support OpenFlow protocols is a software switch. It includes a kernel module (*openvswitch.ko*) and a user space daemon (*vswitchd*). The user space daemon deploys the switch and employs the kernel module to perform some time critical switch processing. Open vSwitch is able to operate both as a soft switch running within the hypervisor, and as the control stack for switching silicon [20]. In fact, this open source software has been ported to multiple virtualization platforms as well as many switching chipsets. Table I summarizes key parameters of the developed SDN-based Wi-Fi AP prototype.

TABLE I. SDN-BASED WI-FI AP CONFIGURATION

Parameters	Values
Hardware kit	Raspberry Pi 3 model B
RAM	1 GB

Chipset	Broadcom BCM2837 chipset
Chip rate	1.2 GHz Quad-Core ARM Cortex-A53
Operating system	Linux kernel 4.9 OpenWrt 18.06.4
Ethernet port	1 port / 100 Mbps
OVS version	2.8.5
OpenFlow protocol	version 1.5
Wi-Fi card	Broadcom BCM43438 WLAN 802.11g / 2.4GHz
WAN interface	Ethernet 100 Mbps

Thanks to OpenvSwitch, the developed device, a software switch, offers many advantages. It is capable of not only providing the same functionality as a regular switch, but also being easily extensible and flexibly configurable. The developed router is configured through a simple database (called *ovsdb*) that can be modified by some tools including *ovsdb-client*, *ovsdb-tool* and *ovs-vsctl*. The database utilizes JSON protocol to communicate and store data. In addition, the switch uses *ovs-ofctl* tool to deal with OpenFlow commands.



Figure 2. The developed SDN-based Wi-Fi AP prototype and Web-based GUI.

The developed wireless access point system can work as a fat AP that is managed by an SDN controller over an OpenFlow channel with the optional transport layer security protocol to provide connection services. WLAN architecture with fat Openflow-enable APs and SDN controller provides a flexible way to manage a Wi-Fi wireless network in several aspects, such as bandwidth, antenna power, etc., on the basis of the advantage that the SDN controller can manage APs dynamically and interactively. To control the developed AP equipped with an Open vSwitch, we use POX controller and have also developed a layer 2 switch control program based on Python language. Figure 2 show the real package of the SDN-based Wi-Fi AP prototype which is connected to a POX controller software running on a laptop. A learning switch (layer 2 switch) "brain", the switch controlling component, associated with a single OpenFlow switch. In order to switch a packet to an appropriate output port, a mapping table that maps addresses to ports needs to be built. Actually, we can populate the table by observing traffic to learn about certain output ports of some specific

sources. When a traffic flow requests to be forwarded, the destination is looked up in the updated table. If the output port could not be found, the switch simply sends the message out all ports except the one it came in on. In short, the learning algorithm of a layer-2 switch is described as following.

<OpenFlow control algorithm for layer 2 switch>:

Step 1: Waiting for a new flow

- ✓ Scan input ports.
- ✓ If there is a new flow, go to Step 2, otherwise repeat Step 1.

Step 2: Serving new flow

For each packet from the switch:

- ✓ Use source address and switch port to update address/port table
- ✓ If transparent = False and either Ethertype is LLDP or the packet's destination address is a Bridge Filtered address, then drop packet and return Step 1.
- ✓ If destination is multicast, then flood the packet and return Step 1.
- ✓ If port for destination address in our address/port table, then flood the packet and return Step 1.
- ✓ If output port is the same as input port, then drop packet and similar ones for a while and return Step 1.
- ✓ Install flow table entry in the switch so that this flow goes out the appropriate port and then, send the packet out appropriate port.
- ✓ Go back to Step 1.

III. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, key designed parameters of the developed SDN-Wi-Fi access point device were tested and verified. Our developed device can support typical Wi-Fi standards including 802.11 b/g/n and be compatible with Openflow 1.5. It also can be controlled by both popular interfaces including graphical user interface (GUI) and command line interface (CLI). The device networking functions are deployed and controlled by an SDN controller. Thanks to using OVS, the device is compatible with typical SDN controllers such as Opendaylight, Ryu, ... However, for simplicity, we used POX version 0.5.0 in our experiments. We have also evaluated the performance, in terms of bandwidth, of the device in different experimental scenarios.

A. System Configuration Verification

We have practically tested our developed prototype of SDN-based Wi-Fi AP in order to verify the designed configuration. We set up an experimental testbed including our SDN-based Wi-Fi device controlled by a POX controller which is installed in a laptop and a mobile phone as a Wi-Fi access device. Note that our Open vSwitch can work well with other SDN controllers such as Ryu, NOX, Opendaylight, ...

Figures 3 and 4 describe the OpenFlow configuration of our developed device as designed. It was demonstrated that the AP prototype was successfully connected to the controller at the IP address of 192.168.1.10 through a TCP port (TCP port number of 6633). Moreover, the versions of

the deployed OVS software and the operating OpenFlow protocol were 2.8.5 and 1.5 respectively.

```

root@OpenWrt:~# ovs-vsctl show
ef77b780-2dec-4cb2-b331-05cc9b9e620e
    Bridge "br-wlan0"
        Controller "tcp:192.168.1.10:6633"
            is_connected: true
        fail_mode: standalone
        Port "br-wlan0"
            Interface "br-wlan0"
                type: internal
        Port "wlan0"
            Interface "wlan0"
        ovs_version: "2.8.5"

```

Figure 3. SDN configuration of the Wi-Fi AP prototype.

```

root@OpenWrt:~# ovs-ofctl -V
ovs-ofctl (Open vSwitch) 2.8.5
OpenFlow versions 0x1:0x5

```

Figure 4. Versions of SDN components (OpenFlow and Open vSwitch).

The SDN-based Wi-Fi AP prototype is then activated as a wireless switch by applying the corresponding POX control program (*forwarding.l2_learning*) which has been introduced in the section II. Figure 5 shows that the device worked properly to provide a flow connection. In fact, depending on the applied control program, our device can be deployed with any appropriate networking functional device like switch, router, firewall, ...

```

hunter@DESKTOP-D6N0KEG:~/pox$ ./pox.py --verbose forwarding.l2_learning
POX 0.5.0 (eel) / Copyright 2011-2014 James McCauley, et al.
DEBUG:core:POX 0.5.0 (eel) going up...
DEBUG:core:Running on CPython (2.7.15+/Oct 7 2019 17:39:04)
DEBUG:core:Platform is Linux-4.4.0-18362-Microsoft-x86_64-with-Ubuntu-18.04-bionic
INFO:core:POX 0.5.0 (eel) is up.
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
INFO:openflow.of_01:[b8-27-eb-f3-e1-cd 1] connected
DEBUG:forwarding.l2_learning:Connection [b8-27-eb-f3-e1-cd 1]

```

Figure 5. CLI of POX controller on the laptop.

Moreover, Figure 6 depicts the Web-based configuration and O&M (Operation and Maintenance) interface of the developed device. Similar to those of common Wi-Fi Aps, the device configuration in term of Wi-Fi parameters including channel number, Wi-Fi standard version and associated station information can be set up and controlled. Our device can support three Wi-Fi standards, 802.11b/g/n. As being demonstrated in the Figure 6, one laptop was connected through Wi-Fi 802.11g with the speed of 54 Mbit/s on the channel number 11 with the carrier frequency of 2.462 GHz. Detailed information including MAC address, host name, current signal/noise value and RX/TX rates can be also provided on the Web interface.

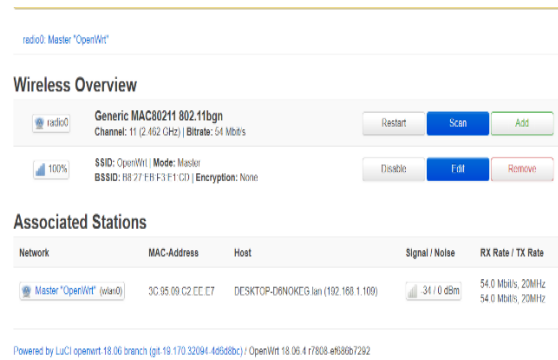


Figure 6. Configurable Web-based interface of the SDN-based Wi-Fi AP prototype.

B. Performance Evaluation

In this part, we have practically tested the performance, in terms of connection speed, of our developed SDN-based Wi-Fi AP. Using a similar experimental setup to that from the previous part, the developed device is controlled by a POX controller installed in a laptop to provide Wi-Fi access while a mobile or a laptop is used as a Wi-Fi station. The Wi-Fi station used Wi-Fi 802.11g which offers the maximum speed of 54 Mbps. Each experimental scenario was repeated five times and its average result was then calculated and summarized.

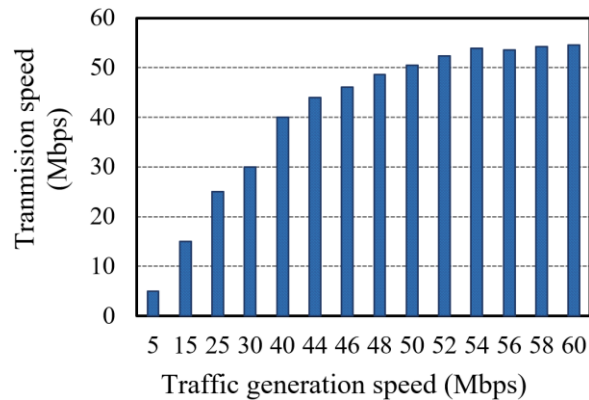


Figure 7. Transmission speed.

Figure 7 shows the obtained speed of the developed prototype when the traffic generated by the mobile station was increasing from 5 Mbps to 60 Mbps. The results confirm that the maximal transmission speed is about 54 Mbps, the limited rate of 802.11g. The received rate is increasing with the generated traffic when it is less than 54 Mbps however, when the transmitted traffic is greater than 54 Mbps, the mobile station can receive only up to 54 Mbps and drop the remaining traffic.

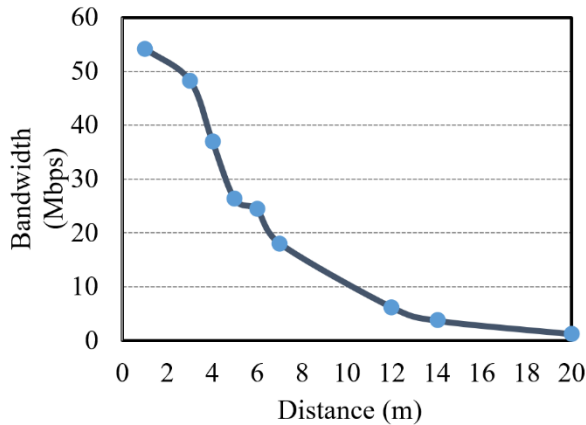


Figure 8. Dependence of bandwidth on distance.

In addition, in order to estimate the cover range of the SDN-based Wi-Fi AP, we tested and measured the transmission speed with different connection distance (the distance between the AP and the mobile station). Figure 8 shows the impact of connection distance on the bandwidth. It informs that the attained bandwidth is decreased when the distance becomes longer. The cover distance of the developed AP is limited due to the use of the built-in Wi-Fi card. In fact, the cover range will be extended by using a suitable external antenna however, this extension is ignored in this paper and will be investigated more details in next stage of the research to figure out suitable applications for the developed device.

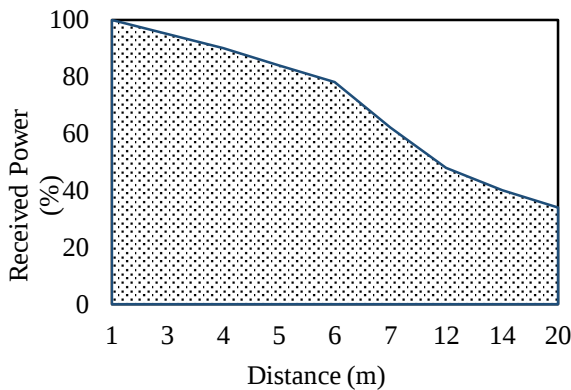


Figure 9. The percentage of received power versus distance.

Finally, we also measured the received power of the mobile station to explain the dependence of the performance on the connection distance. Figure 9 demonstrates the dependence of the received power on the connection distance. The power received at the mobile station is significantly reduced as the mobile station is moving far from the AP. This power decrease explains why the bandwidth of the Wi-Fi connection is lessened with the increase of the connection distance. Again, in order to cope with long range connections, it is required to equip the developed SDN-based Wi-Fi AP device with an appropriate external antenna.

IV. CONCLUSION

In this paper, we have successfully developed a cost-effective scalable SDN-based Wi-Fi AP prototype that

enables OpenFlow 1.5 for IoT communication. The proposed device, that is based on a Raspberry pi 3 and therefore, it is very cost-effective, uses OpenWrt firmware and Open vSwitch software. By using SDN technology, our developed device can be deployed as a network device with various functions such as hub, switch, firewall, ... which depend on Python-based controlling program installed in the SDN controller. The SDN-based Wi-Fi AP performance has been verified by testing experiments. The obtained numerical results proved the effective and scalable performance of the developed SDN-based Wi-Fi AP prototype. With a built-in Wi-Fi card and without external antenna, the SDN-Wi-Fi access point prototype can cover the range of about 20 meters. This device can be a promising approach for creating flexible and effective IoT network devices.

REFERENCES

- [1] Bizanis, Nikos, and Fernando A. Kuipers, "SDN and virtualization solutions for the Internet of Things: A survey", IEEE Access, vol. 4, pp. 5591-5606, 2016.
- [2] Samaresh Bera, Sudip Misra, and Athanasios V. Vasilakos, "Software-Defined Networking for Internet of Things: A Survey", IEEE Internet of Things Journal, vol. 4, no. 6, pp. 1994-2008, 2017.
- [3] Qin Zhijiang, Grit Denker, Carlo Giannelli, Paolo Bellavista, and Nalini Venkatasubramanian, "A software defined networking architecture for the internet-of-things", 2014 IEEE network operations and management symposium (NOMS), pp. 1-9, 2014.
- [4] Bizanis Nikos, and Fernando A. Kuipers, "SDN and virtualization solutions for the Internet of Things: A survey", IEEE Access, vol. 4, pp. 5591-5606, 2016.
- [5] Bakhshi, Taimur, "State of the art and recent research advances in software defined networking", Wireless Communications and Mobile Computing 2017, 2017.
- [6] Diego Kreutz, Fernando M.V. Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig, "Software-defined networking: A comprehensive survey", Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, 2015.
- [7] Hu Fei, Qi Hao, and Ke Bao, "A survey on software-defined network and openflow: From concept to implementation", IEEE Communications Surveys & Tutorials, vol. 16, no. 4, pp. 2181-2206, 2014.
- [8] Haque Israat Tanzeena, and Nael Abu-Ghazaleh, "Wireless software defined networking: A survey and taxonomy", IEEE Communications Surveys & Tutorials, vol. 18, no. 4, pp. 2713-2737, 2016.
- [9] Jararweh Yaser, Mahmoud Al-Ayyoub, Elhadj Benkhelifa, Mladen Vouk, and Andy Rindos, "SDIoT: a software defined based internet of things framework", Journal of Ambient Intelligence and Humanized Computing, vol. 6, no. 4, pp. 453-461, 2015.
- [10] Vipin Gupta, Karamjeet Kaur and Sukhveer Kaur, "Developing Small Size Low-Cost Software-Defined Networking Switch Using Raspberry Pi," Next-Generation Networks, pp. 147-152. Springer, Singapore, 2018.
- [11] Kim, Hyunmin, Jaebeom Kim, and Young-Bae Ko, "Developing a cost-effective OpenFlow testbed for small-scale Software Defined Networking," IEEE 16th International Conference on Advanced Communication Technology (ICACT), pp. 758-761, 2014.
- [12] Behnam Dezfouli, Vahid Esmaelzadeh, Jaykumar Sheth, and Marjan Radi, "A review of software-defined WLANs: Architectures and central control mechanisms," IEEE Communications Surveys & Tutorials 21, no. 1, pp. 431-463, 2018.
- [13] Hisham Elzain, and Yang Wu, "Software Defined Wireless Mesh Network Flat Distribution Control Plane," Future Internet, vol. 11, no. 8, pp. 166- , 2019.
- [14] Won Jin Lee, Jung Wan Shin, Hwi Young Lee, and Min Young Chung, "Testbed implementation for routing WLAN traffic in software defined wireless mesh network," 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 1052-1055, 2016.

- [15] Kristián Košťál, Rastislav Bencel, Michal Ries, Peter Trúchly, and Ivan Kotuliak, "High Performance SDN WLAN Architecture," *Sensors*, vol. 19, no. 8, pp. 1880- , 2019.
- [16] Ahmed Abdelaziz, Ang Tan Fong, Abdullah Gani, Suleman Khan, Faiz Alotaibi, Muhammad Khurram Khan, "On Software-Defined Wireless Network (SDWN) Network Virtualization: Challenges and Open Issues", *The Computer Journal*, Volume 60, Issue 10, pp. 1510–1519, October 2017.
- [17] Hsu, Huai-Wen, Kuei-Li Huang, Yi-Chih Kao, Shi-Chun Tsai, and Yi-Bing Lin, "Deploying WLAN service with OpenFlow technology", *International Journal of Network Management* 27, no. 3, pp. e1970, 2017.
- [18] Alshnta, Abdalkrim M., Mohd Faizal Abdollah, and Ahmed Al-Haiqi, "SDN in the home: A survey of home network solutions using Software Defined Networking", *Cogent Engineering* 5, no. 1, pp. 1469949, 2018.
- [19] Miano Sebastiano, and Fulvio Risso, "Transforming a traditional home gateway into a hardware-accelerated SDN switch", *International Journal of Electrical and Computer Engineering* 10, no. 3, pp. 2668, 2020.
- [20] *OpenWrt official homepage*, <https://openwrt.org/>
- [21] John C. Shovic, "Raspberry pi IoT projects," Apress, 2016.
- [22] "Raspberry Pi Foundation.", <https://www.raspberrypi.org/>
- [23] <https://openflow.stanford.edu/display/ONL/POX+Wiki.html>
- [24] *OpenvSwitch Official Homepage*, <https://openvswitch.org>
- [25] https://standards.ieee.org/standard/802_11-2012.html
- [26] 802.11-2012 - IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Tech. Rep. IEEE Std 802.11TM-2012, IEEE-Inst.
- [27] *Wi-Fi Alliance*, <http://www.wi-fi.org>

đo kiểm. Các kết quả đạt được đã thể hiện sự hiệu quả và khả năng mở rộng linh hoạt của thiết bị được phát triển.

Từ khoá: Mạng định nghĩa bằng phần mềm, điểm truy nhập Wi-Fi, OVS, OpenFlow.



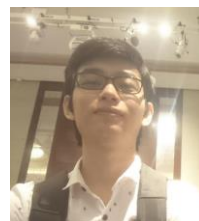
Hai-Chau Le received the B.E. degree in Electronics and Telecommunications Engineering from Posts and Telecommunications Institute of Technology (PTIT) of Vietnam in 2003, and the M.Eng. and D.Eng. degrees in Electrical Engineering and Computer Science from Nagoya University of Japan in 2009 and 2012, respectively. From 2012 to 2015, he was a researcher in Nagoya University of Japan and in University of California, Davis, USA. He is currently a lecturer in Telecommunications Faculty at PTIT. His research interests include optical technologies, network design and optimization and future network technologies. He is an IEEE member.

NGHIÊN CỨU PHÁT TRIỂN HỆ THỐNG TRUY NHẬP WI-FI ĐỊNH NGHĨA BẰNG PHẦN MỀM SỬ DỤNG RASPBERRY PI 3 VÀ OPENWRT

Lê Hải Châu và Nguyễn Khắc Tuấn

Học viện Công nghệ Bưu chính Viễn thông

* Email liên hệ: chaulh@ptit.edu.vn



Khắc-Tuan Nguyen is a 5th year bachelor student in the field of Electronics and Telecommunications Engineering at Posts and Telecommunications Institute of Technology of Vietnam, and is currently an active member of Information network research group of Telecommunications Faculty I in PTIT. His research interests include SDN, NFV, and future network technologies.

Tóm tắt: Trong bài báo này, chúng tôi đã nghiên cứu và phát triển thành công một mẫu thiết bị truy nhập Wi-Fi định nghĩa bằng phần mềm cho truyền thông IoT. Thiết bị được phát triển dựa trên nền tảng Raspberry pi 3, sử dụng firmware OpenWrt dựa vào nhân Linux và phần mềm chuyển mạch SDN mã nguồn mở Open vSwitch. Hệ thống này có khả năng tương thích với giao thức OpenFlow 1.5 và hỗ trợ giao diện WAN qua một cổng Ethernet dung sẵn 100 Mbps. Nhờ việc khai thác các ưu điểm của công nghệ SDN, công nghệ mã nguồn mở và nền tảng máy tính cỡ nhỏ giá thành rẻ, mẫu thiết bị truy nhập Wi-Fi định nghĩa bằng phần mềm có giá thành chấp nhận được trong khi vẫn rất linh hoạt, có khả năng mở rộng tốt và cho phép hỗ trợ đầy đủ các tính năng mạng tiên tiến. Thiết bị này cũng có thể được triển khai thành các thiết bị mạng với các tính năng khác nhau như hub, switch, router hay firewall, ... bằng cách sử dụng phần mềm điều khiển được lập trình và cài đặt trên bộ điều SDN. Cấu hình và hiệu năng của mẫu thiết bị này được kiểm thử và xác minh thông qua các thí nghiệm