ICIC Express Letters
Part B: Applications
Volume **13**, Number **2**, February **2022**

ICIC International ©2022 ISSN 2185-2766
pp. 145–153

EVENT-DRIVEN REMOTE CONFIGURATION FUNCTION IN COST-EFFECTIVE ROUTER/SWITCH CONTROL SYSTEMS BASED ON SOFTWARE-DEFINED NETWORKING USING IOT DEVICES

Keigo Mino[1], Akihiro Imae[1], Osanori Koyama[1], Ippei Tomo[2]
Minoru Yamaguchi[1], Kento Oyama[1], Kanami Ikeda[1] and Makoto Yamada[1]

[1]Graduate School of Engineering
[2]College of Engineering
Osaka Prefecture University
Gakuen-cho 1-1, Naka-ku, Sakai, Osaka 599-8531, Japan
{ szb01158; sab01019; dd104005; md104012 }@edu.osakafu-u.ac.jp
{ koyama; nf101345; kanami; myamada }@eis.osakafu-u.ac.jp

Received August 2021; accepted October 2021

Abstract. *Software-defined networking (SDN) is one of the network virtualization technologies that has gained attention recently, and it has now been implemented in networks. SDN can reduce the network management workload by flexibly changing many network device parameters and centralizing control methods specific to the network devices. However, SDN-enabled network devices are currently expensive, posing a barrier to SDN adoption in enterprise and data center networks. In this study, an event-driven remote configuration function based on monitoring input packets in a router/switch control system using inexpensive Internet of Things devices is proposed to reduce the cost of SDN deployment and network management workload. We developed the proposed function and evaluated its performance on an experimental network. The experimental results are discussed, and the effectiveness of the proposed function is verified.*
Keywords: Software-defined networking, OpenFlow, Raspberry Pi, RYU, Event-driven remote configuration, IoT devices

1. Introduction.

Virtualization technology has recently made it easier to virtualize various network resources, such as router/switch for packet exchange and server/storage for data sharing service. However, control and management of network resources have been complex because the resources have been dynamically integrated and distributed [1-4]. Furthermore, the number of network facilities, called data centers, where huge network resources are located, has been increasing because of new social network services. In particular, it must change several parameters of router/switch more flexibly and dynamically than the times when social network services were not extensively used [5,6]. However, in conventional networks, the parameters of the router/switch are required to be set using vendor-specific configuration methods and flexibly and dynamically changing parameters of router/switch are tangled. Software-defined networking (SDN) technology has emerged and been researched for flexibly, dynamically, and efficiently operating complex networks [7-11].

In the non-SDN environment, network devices such as routers and switches have integrated the data plane that exchanges packets and the control plane that controls the data plane. When new network devices are added or the network configuration is changed, the network administrator should set the parameters of each device individually using the vendor-specific control plane, which depends on the manufacturer and physical tasks

DOI: 10.24507/icicelb.13.02.145

such as plugging and unplugging cables. Therefore, the more complex and larger the network is, the more time and human resources are required and the heavier the workload becomes.

By contrast, the control and the data planes in the network devices are separated in the SDN environment. Additionally, a vendor-independent control plane is composed of a single software called an SDN controller. Furthermore, the SDN environment provides a common interface to administrators, enabling flexible and dynamic parameter settings for network devices of various vendors. OpenFlow (OF) is adopted as a protocol for SDN [12-15]. The OF protocol can centralize the management of SDN-enabled network devices using the OF controller and can be expected to significantly reduce the workload of network administrators by liberating them from vendor-specific configuration methods.

Although SDN provides considerable benefits in terms of network flexibility, operability, and manageability, replacing all legacy network devices with SDN-enabled devices requires high cost because the existing SDN-enabled network devices are expensive. Therefore, cost is a significant barrier for the construction of the SDN environment [16,17]; therefore, a gradual replacement of the legacy network devices with the SDN-enabled ones within an acceptable cost is considered instead of replacing all of them at once [18-22]. However, to realize the full benefits of SDN, all the legacy network devices should be replaced with the SDN-enabled ones; to achieve this, low-cost solutions are required for SDN deployment. Some studies have reported the low-cost solutions using Internet of Things (IoT) devices as Ethernet switches or OF switches [23-25]; however, the packet forwarding capability depends on the IoT devices' processing performance. Therefore, these solutions are limited to low-speed networks such as home networks, and unsuitable for high-speed networks such as the Giga class. Another study has been reported that uses the devices called field-programmable gate arrays (FPGA) to update the legacy network devices to the SDN-enabled ones [26]. Although the cost of the FPGA device is low, this solution requires a significant amount of time and effort in addition to the high development costs for updating the legacy device to an SDN-enabled one. This solution also implements the packet forwarding function in FPGA, making it difficult to be applied to high-speed networks. A study has been reported for updating the legacy network devices to the SDN-enabled ones to link general-purpose servers with legacy network devices instead of IoT devices [27]. However, this solution requires a high-performance general-purpose server for its application to the high-speed network because the packet forwarding function is performed by software switches in the general-purpose server, resulting in high cost for the SDN implementation.

For this background, we have researched a method to insert and link an inexpensive IoT device between the SDN controller and a legacy network device to realize a common interface, which is one of the SDN functions, and remote network control [28-31]. In our proposed method, only IoT devices have been added to construct an SDN environment. Therefore, the cost can be significantly reduced, compared with the method of replacing legacy network devices with SDN-enabled devices, and it is practically possible to make all devices in the network SDN-enabled at once. In this study, we propose an event-driven remote configuration function using the input-packet monitoring of the IoT device to improve the technology for SDN deployment by linking the inexpensive IoT devices with the legacy network devices. This study, compared with state-of-the-art studies, has effectiveness in terms of realizing the function of the packet-in message and the translation of control command which have not yet been realized in our past studies. Result has shown that an event-driven remote configuration function performs properly and the function of the packet-in message has been realized in our proposed method. In Section 2 of this paper, we present the functional module design and operation flow of the event-driven remote configuration function. In Section 3, we present the experimental setup and results of the demonstration on the experimental network. In Section 4, we present the conclusions.

2. **Event-Driven Remote Configuration Function.** Figure 1 shows that the development of data plane in the legacy network device was not the focus of our research, and the function of translating OF-based control commands into vendor-specific ones was implemented in low-cost IoT devices. Furthermore, we proposed the function of the common interface (IF) to link the data planes of the legacy network devices.

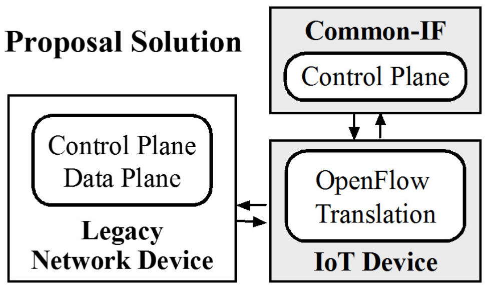

Figure 1. Outline of the proposed solution

Figure 2 shows the functional module structure and operation flow of the common IF. The OF controller shown in Figure 2 is implemented in a personal computer (PC) for centrally controlling network devices, and the OF switch is implemented in the IoT device. The legacy network devices, IoT devices, and control PC are physically connected via a general-purpose Ethernet (unshielded twisted pair) cable. As shown in the flow (1) in Figure 2, after the IoT device boots up, it sends the OF switch identification (ID) to the OF controller through the OF IF. The OF controller registers the ID of the OF switch in its database and then establishes an OF connection. Control commands based on OF are input by the network administrator through the user IF, and the control commands are sent and received between the OF controller and OF switch using the OF connection, shown in the flow (2), respectively. Then, the adapter in the IoT device translates the OF control commands into vendor-specific commands, as shown in the flow (3), and sends the vendor-specific commands to the legacy network device through the telnet IF (the flow (4)). The common IF function in the IoT devices enables the administrator to directly control the OF controller and remotely configure the parameters to the legacy network devices. In this study, the IoT device is a device that can provide an IF for communicating. Figure 2 also shows how the IoT device provides an IF between the legacy network device and the SDN controller.

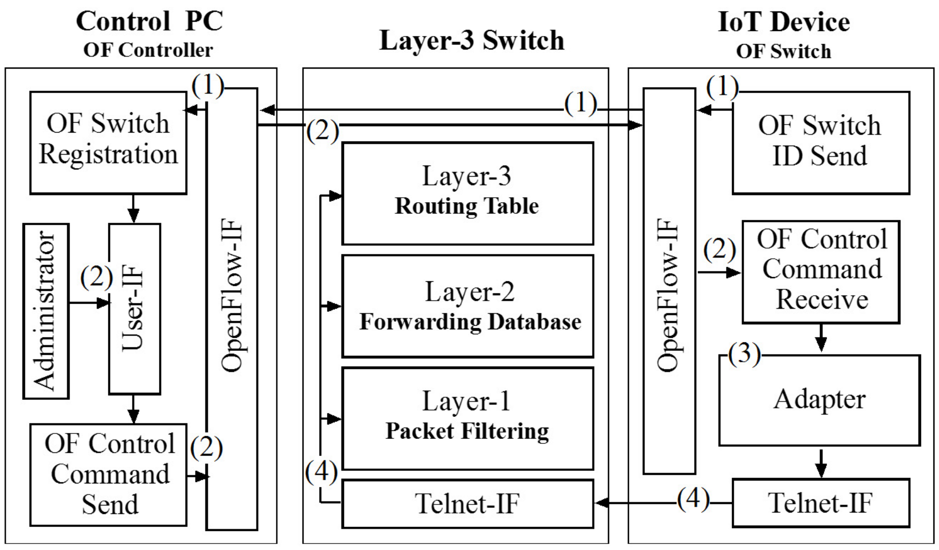

Figure 2. Function of remote configuration by OpenFlow-based common interface

In an SDN environment, packets entering a network device are routed based on the OF forwarding policy. Packet-in message is used to send incoming packets in the OF switch to the OF controller if they do not follow the forwarding policy [32]. By extracting information that was forwarded by the packet-in message to OF controller, the network information and current status can be shared with various external services. The function of the packet-in message has not yet been realized in our proposed function of the common IF using the IoT devices. Therefore, in this study, we implement the function of receiving and analyzing the packet-in message in the IoT devices using an SDN-enabled software switch called Open vSwitch (OVS) [33] to obtain the packet-in message generated with network events on the OF controller side. We propose that the function of event-driven remote configuration is linked with the conventional common IF function. In this section, we first provide an overview of the operation flow of the network information acquisition using the packet-in message and then describe the function of the event-driven remote configuration.

2.1. **Network information acquisition using packet-in message.** Figure 3 shows the configuration of the functional components and the operation flow for acquiring network information using packet-in message implemented in the IoT device. The container of the functional components consists of network devices, namely legacy layer-3 switch, IoT device, and control PC. As shown in Figure 3, after the IoT device boots up, it sends the OF switch ID to the control PC through the OF IF, and the control PC registers it in its database and establishes an OF connection, which is shown as a bidirectional link in the flow (1). The forwarding policy to the OF controller is set in the OVS in the OF switch shown in the flow (2). The layer-3 switch is configured for port mirroring, and all packets entering the layer-3 switch are forwarded to the OF switch, where the OVS analyzes the packet headers, as shown in the flow (3). Based on the forwarding policy, packets are checked continuously, and if the packet matches the forwarding rule, a packet-in message is created and sent to the control PC, as shown in the flow (4). The network administrator can check the receiving of the packet-in message through the user IF, as shown in the flow (5). With this function, the network administrator can obtain information from packets to be input to the network device and grasp the events that occurred in the network.

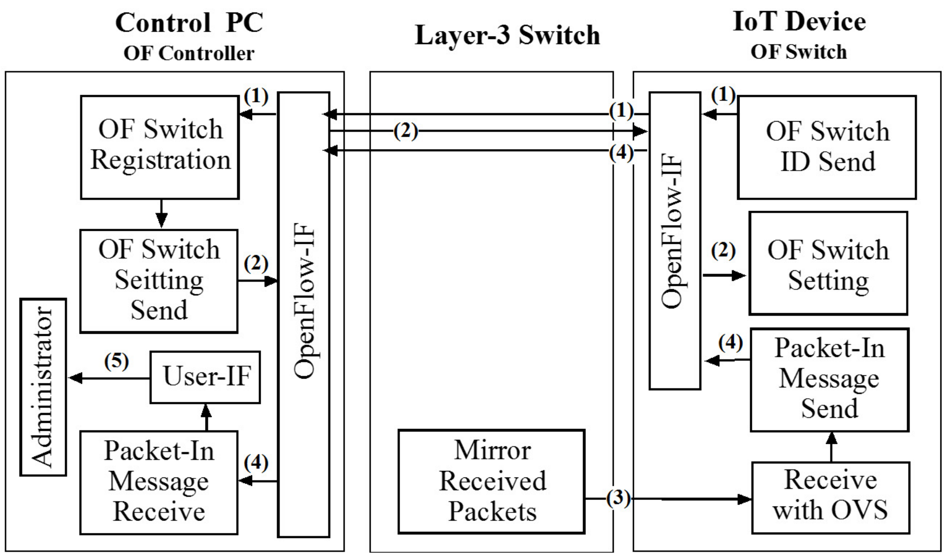

FIGURE 3. Information acquisition function with packet-in message

2.2. **Event-driven remote configuration.** Figure 4 shows the OF controller and OF switch components and the operation flow of the function of the event-driven remote configuration. The function is implemented by linking the function of the common IF and the function of the network information acquisition. As shown in Figure 4, the hardware consists of a network device, an IoT device, and a control PC with the functions of the common IF and network information acquisition. The operation flow from (1) to (4) in Figure 4 is equal to the operation flow from (1) to (4) in Figure 3. Therefore, the network administrator obtains packet information by receiving packet-in message in control PC and grasps the events that occurred in the network. Then, the control PC sends OF-based control commands based on the administrator's policy according to each event from the control PC to the IoT device, as shown in the flow (5). The OF-based control commands are then translated into vendor-specific ones in the adapter, as shown in the flow (6). Finally, the OF switch sends vendor-specific control commands to the layer-3 switch through the telnet IF and the layer-3 switch is reflected in various parameters of configuration belonging from layer-1 (physical layer) to layer-3 (network layer), as shown in the flow (7). The above mentioned operation flow provides a common IF and realizes the function of the network information acquisition by the function of the packet-in message.

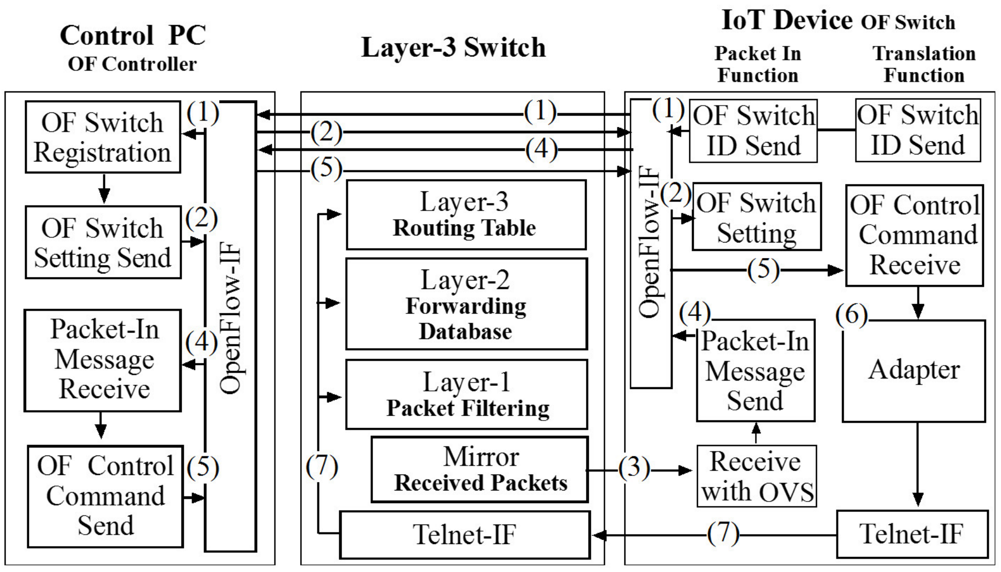

FIGURE 4. Event-driven remote configuration function

3. **Experiment and Results.** We have designed and developed the function of the event-driven remote configuration and demonstrated the function on an experimental network. This section describes the experimental setup and procedure, and we then present the results.

3.1. **Experiment of packet blocking using an enabled filter.** Figure 5 shows the experimental setup. In Figure 5, an IoT device, PC-1, and a layer-2 switch are connected to a layer-3 switch and a packet monitoring PC, a control PC, and PC-2 are connected to the layer-2 switch. Furthermore, the layer-3 switch had two types of links for IoT devices: a normal packet forwarding link using port P-a and a mirroring link to receive packets from PC-1 using port P-b. An OF controller was installed in the control PC, as shown in Figure 4. In this experiment, we adopted RYU [34], an open source, free software, as the OF controller and the Raspberry Pi 3 Model B [35] as the IoT device. The Raspberry Pi 3 Model B was equipped with only one Ethernet local area network (LAN) port as a network interface; however, as we used two ports for IoT device in this experiment, we converted one of the universal serial bus ports of the Raspberry Pi 3 Model B into an

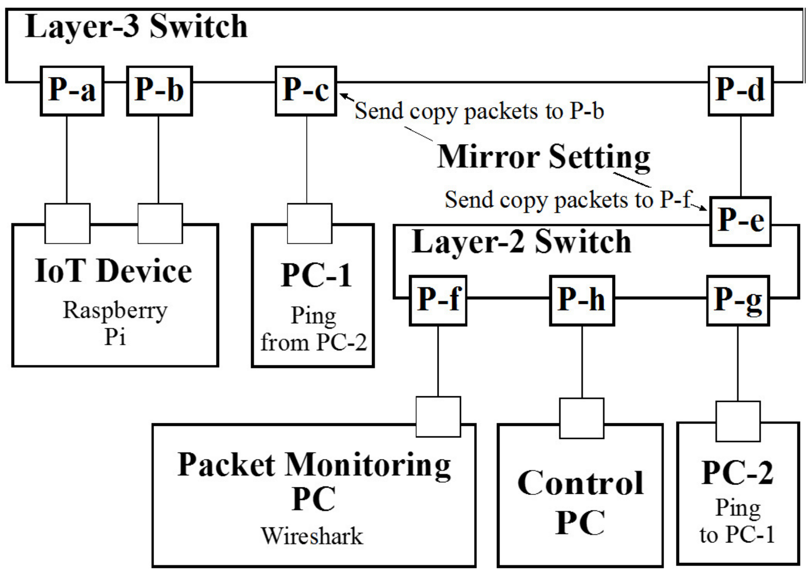

FIGURE 5. Experimental setup

Ethernet LAN port using a connector. We also installed Wireshark [36], a packet capture software, as the LAN analyzer in the packet monitoring PC.

We configured the mirroring port P-b of the layer-3 switch as the forwarding destination of the input packets in port P-c. This configuration allows the IoT device to acquire the input packets from PC-1 to the layer-3 switch and vice versa. Moreover, we configured the mirroring port P-f of the layer-2 switch as the forwarding destination of the input packets in the port P-e. This configuration allows the packet monitoring PC to acquire the input packets from the layer-3 switch to the layer-2 switch and vice versa. In this paper, we implemented the function of the event-driven remote configuration as follows. When the event of "echo reply packets based on Internet control message protocol (ICMP) [37] are input 20 times from PC-1 to PC-2" occurs, the configuration of "the filter in the layer-3 switch is enabled to block the ICMP echo request packets from PC-2 to PC-1" is conducted. The mirrored ICMP echo reply packet triggers the creation of a packet-in message, which is sent to OF controller, and the OF controller processes the packet-in message based on the policy. In this experiment, the policy is to apply a filter to the layer-3 switch to block the ICMP echo request packets. We confirmed that the function of the event-driven configuration worked properly by executing the Ping on PC-2 and sending the packets 20 times which were the ICMP echo request packets from PC-2 to PC-1 and checking the packets between PC-1 and PC-2 and between the IoT device and the control PC in the packet monitoring PC. The packets between PC-1 and PC-2 and between the IoT device and the control PC were checked by the packet monitor PC for validation.

3.2. **Results and disccusions.**

Figure 6 shows the results of the experiment. The horizontal axis represents the elapsed time, and the start time of the transmission of the ICMP echo request packets from PC-2 to PC-1 is set to 0. The vertical axis represents the type of observed packets: the packets of the ICMP echo request, ICMP echo reply, packet-in message, and OF-based control command. As shown in Figure 6, when a packet-in message was observed 20 times, the policy was confirmed by OF-based control command. Therefore, the ICMP echo reply packet was observed until the OF-based control command was sent. However, after the OF-based control command was applied, only the ICMP echo request packets were observed and no ICMP echo reply packet was observed. This result indicates that the filter has been enabled to block the ICMP echo

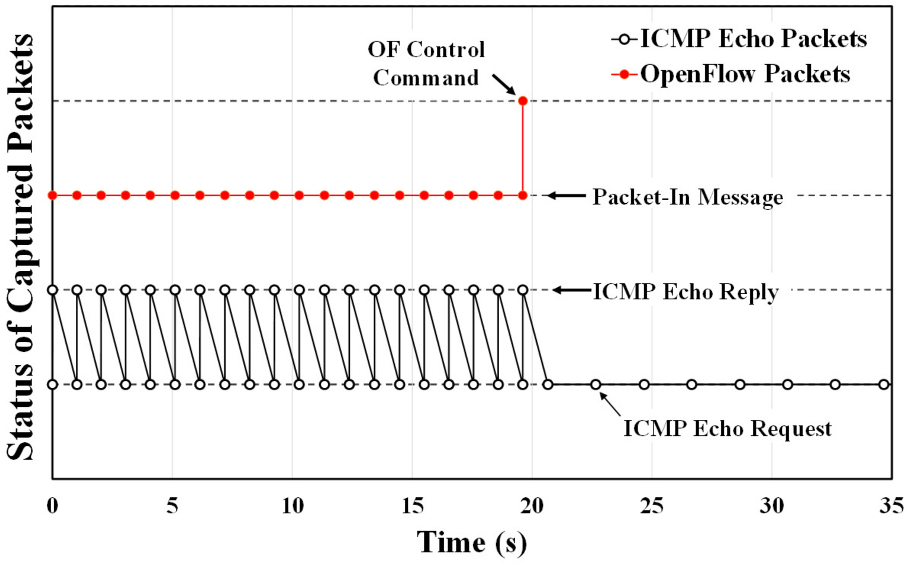

FIGURE 6. Result of packet blocking with an enabled filter

request packet from PC-2 to PC-1 in the layer-3 switch. In conclusion, we confirmed that the event-driven configuration function is working properly.

4. Conclusions.

We proposed a function of event-driven remote configuration using inexpensive IoT devices as a low-cost solution for implementing an SDN environment. In this letter, a function of the event-driven configuration triggered by input packets is developed and demonstrated on the experimental network. As a result, by monitoring the packets of the ICMP echo request and reply packets, whether the function was working properly can be verified. In the future, we will improve the method of the low-cost SDN deployment using IoT devices and expand the IoT devices to optical components in SDN used in optical IP networks and sensing devices such as electrical sensors and optical fiber sensors used in sensor networks.

Acknowledgment. This work was supported by JSPS KAKENHI Grant Number 16K06 306.

REFERENCES

[1] A. Laghrissi and T. Taleb, A survey on the placement of virtual resources and virtual network functions, *IEEE Communications Surveys & Tutorials*, vol.21, no.2, pp.1409-1434, 2019.
[2] J. G. Herrera and J. F. Botero, Resource allocation in NFV: A comprehensive survey, *IEEE Trans. Network and Service Management*, vol.13, no.3, pp.518-532, 2016.
[3] L. Wang, Z. Lu, X. Wen, R. Knopp and R. Gupta, Joint optimization of service function chaining and resource allocation in network function virtualization, *IEEE Access*, vol.4, pp.8084-8094, 2016.
[4] P. S. Khodashenas et al., Service mapping and orchestration over multi-tenant cloud-enabled RAN, *IEEE Trans. Network and Service Management*, vol.14, no.4, pp.904-919, 2017.
[5] M. Noormohammadpour and C. S. Raghavendra, Datacenter traffic control: Understanding techniques and tradeoffs, *IEEE Communications Surveys & Tutorials*, vol.20, no.2, pp.1492-1525, 2018.
[6] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, A. Kanagala, J. Provost, J. Simmons, E. Tanda, J. Wanderer, U. Hölzle, S. Stuart and A. Vahdat, Jupiter rising: A decade of clos topologies and centralized control in Google's datacenter network, *ACM SIGCOMM Computer Communication Review*, vol.45, no.4, pp.183-197, 2015.
[7] D. B. Rawat and S. R. Reddy, Software defined networking architecture, security and energy efficiency: A survey, *IEEE Communications Surveys & Tutorials*, vol.19, no.1, pp.325-346, 2017.
[8] B. A. A. Nunes, M. Mendonca, X. Nguyen, K. Obraczka and T. Turletti, A survey of software-defined networking: Past, present, and future of programmable networks, *IEEE Communications Surveys & Tutorials*, vol.16, no.3, pp.1617-1634, 2014.

[9] H. Kim and N. Feamster, Improving network management with software defined networking, *IEEE Communications Magazine*, vol.51, no.2, pp.114-119, 2013.

[10] F. Bannour, S. Souihi and A. Mellouk, Distributed SDN control: Survey, taxonomy, and challenges, *IEEE Communications Surveys & Tutorials*, vol.20, no.1, pp.333-354, 2018.

[11] L. Shi, Q. Li, Z. Cheng, Z. Xue and X. Lv, End information hopping for active cyber-defense based on SDN, *ICIC Express Letters*, vol.11, no.1, pp.135-141, 2017.

[12] W. Braun and M. Menth, Software-defined networking using OpenFlow: Protocols applications and architectural design choices, *Future Internet*, vol.6, no.2, pp.302-336, 2014.

[13] A. Lara, A. Kolasani and B. Ramamurthy, Network innovation using OpenFlow: A survey, *IEEE Communications Surveys & Tutorials*, vol.16, no.1, pp.493-512, 2014.

[14] M. Alsaeedi, M. M. Mohamad and A. A. Al-Roubaiey, Toward adaptive and scalable OpenFlow-SDN flow control: A survey, *IEEE Access*, vol.7, pp.107346-107379, 2019.

[15] *OpenFlow*, https://www.opennetworking.org/, Accessed in August 2021.

[16] T. Das, M. Caria, A. Jukan and M. Hoffmann, A techno-economic analysis of network migration to software-defined networking, *arXiv.org*, arXiv: 1310.0216, 2013.

[17] B. R. Dawadi, D. B. Rawat, S. R. Joshi and M. M. Keitsch, Joint cost estimation approach for service provider legacy network migration to unified software defined IPv6 network, *Proc. of 2018 IEEE 4th International Conference on Collaboration and Internet Computing*, pp.372-379, 2018.

[18] J. N. Binlun, T. S. Chin, L. C. Kwang, Z. Yusoff and R. Kaspin, Challenges and direction of hybrid SDN migration in ISP networks, *Proc. of 2018 IEEE International Conference on Electronics and Communication Engineering*, pp.60-64, 2018.

[19] R. Amin, M. Reisslein and N. Shah, Hybrid SDN networks: A survey of existing approaches, *IEEE Communications Surveys & Tutorials*, vol.20, no.4, pp.3259-3306, 2018.

[20] E. Rojas, R. Amin, C. Guerrero, M. Savi and A. Rastegarnia, Challenges and solutions for hybrid SDN, *Computer Networks*, vol.195, DOI: 10.1016/j.comnet.2021.108198, 2021.

[21] M. Ibrar, L. Wang, G.-M. Muntean, J. Chen, N. Shah and A. Akbar, IHSF: An intelligent solution for improved performance of reliable and time-sensitive flows in hybrid SDN-based FC IoT systems, *IEEE Internet of Things Journal*, vol.8, no.5, pp.3130-3142, 2021.

[22] D. Ferreira T. Pontes, M. F. Caetano, G. P. R. Filho, L. Z. Granville and M. A. Marotta, On the transition of legacy networks to SDN – An analysis on the impact of deployment time number and location of controllers, *Proc. of 2021 IFIP/IEEE International Symposium on Integrated Network Management*, pp.367-375, 2021.

[23] S. Ferdoush and X. Li, Wireless sensor network system design using Raspberry Pi and Arduino for environmental monitoring applications, *Procedia Computer Science*, vol.34, pp.103-110, 2014.

[24] A. Collaguazo, R. Alcivar, J. Pesantez and R. Ponguillo, Cost effective test-bed for comparison of SDN network and traditional network, *Proc. of 2018 IEEE 37th International Performance Computing and Communications Conference*, Orlando, DOI: 10.1109/PCCC.2018.8711223, 2018.

[25] D. González, C. Mellado, K. Waltam and A. Lara, Low-cost SDN switch comparison: Zodiac FX and Raspberry Pi, *Proc. of 2019 IV Jornadas Costarricenses de Investigación en Computación e Informática*, pp.1-5, 2019.

[26] D. J. Casey and B. E. Mullins, SDN shim: Controlling legacy devices, *Proc. of 2015 IEEE 40th Conference on Local Computer Networks*, Clearwater Beach, FL, USA, pp.169-172, 2015.

[27] L. Csikor, M. Szalay, G. Rétvári, G. Pongrácz, D. P. Pezaros and L. Toka, Transition to SDN is HARMLESS: Hybrid architecture for migrating legacy Ethernet switches to SDN, *IEEE/ACM Trans. Networking*, vol.28, no.1, pp.275-288, 2020.

[28] S. Aso, Y. Tomioka, O. Koyama, T. Niihara, Y. Ogura and M. Yamada, Web-based remote management system for optical switch in AWG-STAR with loopback function, *Proc. of 2018 Opto-Electronics and Communications Conference*, Jeju Island, Korea, DOI: 10.1109/OECC.2018.8729871, 2018.

[29] K. Minou, S. Aso, O. Koyama, M. Yamaguchi, Y. Tomioka, Y. Ogura, K. Ikeda and M. Yamada, OpenFlow-based remote control of optical switch employing IoT device in AWG-STAR with loopback function, *Proc. of 2019 Opto-Electronics and Communications Conference*, Fukuoka, Japan, DOI: 10.23919/PS.2019.8817804, 2019.

[30] A. Imae, K. Mino, O. Koyama, K. Oyama, M. Yamaguchi, K. Ikeda and M. Yamada, Router control function using IoT device supported OpenFlow switch in IP over AWG-STAR network, *Proc. of 2020 Opto-Electronics and Communications Conference*, Taipei, Taiwan, DOI: 10.1109/OECC48412.2020.9273564, 2020.

[31] A. Imae, O. Koyama, K. Mino, I. Tomo, M. Yamaguchi, K. Oyama, K. Ikeda and M. Yamada, Cost-effective router/switch control system based on software-defined networking over world wide web, *International Journal of Innovative Computing, Information and Control*, vol.17, no.5, pp.1617-1627, 2021.

[32] *OpenFlow Switch Specification Ver 1.5.1*, https://opennetworking.org/software-defined-standards/specifications/, Accessed in August 2021.
[33] *Open vSwitch*, https://www.openvswitch.org/, Accessed in August 2021.
[34] *RYU SDN, Framework*, https://ryu-sdn.org/, Accessed in August 2021.
[35] *Raspberry Pi*, https://www.raspberrypi.org/, Accessed in August 2021.
[36] *Wireshark*, https://www.wireshark.org/, Accessed in August 2021.
[37] *Internet Control Message Protocol*, https://datatracker.ietf.org/doc/html/rfc792, Accessed in August 2021.