International Journal of Innovative
Computing, Information and Control
Volume **17**, Number **5**, October **2021**

ICIC International ©2021 ISSN 1349-4198
pp. **1617–1627**

COST-EFFECTIVE ROUTER/SWITCH CONTROL SYSTEM BASED ON SOFTWARE-DEFINED NETWORKING OVER WORLD WIDE WEB

Akihiro Imae[1], Osanori Koyama[1], Keigo Mino[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
1-1 Gakuen-cho, Naka-ku, Sakai, Osaka 599-8531, Japan
{ sab01019; koyama; szb01158; dd104005; md104012; kanami; myamada }@eis.osakafu-u.ac.jp
szb01103@edu.osakafu-u.ac.jp

Received April 2021; revised July 2021

Abstract. *Software-defined networking (SDN), which enables networks to be controlled centrally and can change parameters flexibly such as their topology, settings for packet-exchanging, and quality of service is one of the network virtualization technologies that have attracted significant attention recently. Besides, it has been widely used because it reduces network administrator's workloads of network management. However, SDN-enabled network devices are generally expensive, and the network construction cost is a significant factor when SDN is deployed in a network of enterprises and data centers. In this paper, from the viewpoint of the cost and network management workload, we propose a router/switch control system for Internet protocol over an optical network with database function over World Wide Web using cost-effective Internet of Things devices. We designed and constructed the proposed system. Besides, we conducted experiments using the remote control function in the proposed system, which can change the port status of network devices. The experiment results proved that the proposed system works correctly. Therefore, by using the proposed system, SDN-enabled network construction cost can be significantly reduced compared to replacing legacy network devices with SDN-enabled network devices. To be more precise, the cost of a single SDN-enabled network device can be reduced to the cost of a single Raspberry Pi as IoT device, and the total cost can be reduced in proportion to the scale of the network. Furthermore, we discuss future studies to be considered for further cost reduction.*

Keywords: Software-defined networking, Remote control, IoT device, OpenFlow, Raspberry Pi, Ryu

1. **Introduction.** With the development of the Internet and the increase in cloud services, communication networks in enterprises and data centers have become multivendor environments consisting of many network devices from various vendors. This situation is one of the causes of an increase in management workload on network administrators. To reduce the management workload on network administrators, an autonomous distributed network has been investigated [1-5]. Recently, network virtualization technology called software-defined networking (SDN) has emerged because the demand for programmable network construction has been increasing to reduce the management workload on network administrators [6-11]. In SDN, a software called SDN controller can centrally manage network devices such as SDN-enabled Ethernet switches and routers. The SDN technology virtually unifies all vendor-specific management methods in a multivendor environment.

DOI: 10.24507/ijicic.17.05.1617

It reduces the network management workload because network administrators can execute all configuration changes for SDN-enabled Ethernet switches and routers via a common SDN controller. Besides, there is the additional merit of significantly reducing the management cost of networks in enterprises [12,13]. In addition to the reduction of management workload, research on networks that incorporate SDN architecture, which allows flexible network construction, into the Internet of Things (IoT) network architecture has been presented [14]. Besides, some studies use SDN and apply the advantages of SDN to other fields [15-17].

It is desirable to replace all the legacy network devices in the network with SDN-enabled devices to maximize the benefits of SDN, but replacing all the legacy network devices extremely increases the cost [18]. Thus, the gradual replacement of legacy Ethernet switches and routers with SDN-enabled ones in the network is being considered [19,20], and hybrid-SDN, where legacy network devices and SDN-enabled ones are mixed, has been investigated [16-18]. Even in the case of hybrid-SDN, the total cost remains the same since all legacy network devices finally replace with SDN-enabled network devices, and this does not solve the cause of the problem. Therefore, a study that uses inexpensive IoT devices as SDN-enabled Ethernet switches has been presented [21,22]. It was low cost, but it was used in the low-speed network due to the low processing performance of the hardware. Thus, it is unsuitable for usage in high-speed networks such as a gigabit-class network. Besides, some studies have been presented on how to update legacy Ethernet switches and routers to be SDN-enabled ones without replacing them with SDN-enabled ones. For example, an inexpensive device called a field-programmable gate array (FPGA) can be used to update devices to support SDN [23]. Although the device cost is low, the development cost to make them SDN-enabled is high and the process is time-consuming. Besides, it implements packet-exchanging functions in the FPGA, thus making it difficult to apply to high-speed networks. Some studies also have been conducted on how to update SDN-enabled network devices by combining general-purpose servers with legacy network devices [24]. However, general-purpose servers require high processing performance because the packet-exchanging function is conducted by software switches in the general-purpose servers. Thus, the construction cost of the system is relatively high in introducing SDN. In short, hybrid-SDN does not solve the cost problem of SDN deployment. The cases of using IoT devices as SDN-enabled Ethernet switches and using FPGA as the device to update legacy network devices with SDN-enabled network devices are low cost, but they are limited to the low-speed network since the processing performance of packet-exchanging function is low. Besides, even in the case of using a general-purpose server to update legacy network devices with SDN-enabled network devices, the processing performance of packet-exchanging function depends on the performance of the server. Therefore, this case in high-speed network requires the use of a high processing performance server and it is difficult to achieve at low cost. From the background, to achieve cost-effective SDN deployment in high-speed networks, we have investigated how to update the legacy network devices to SDN-enabled ones by adding inexpensive IoT devices between SDN controller and legacy network devices and combining IoT devices with legacy network devices [25-27]. To reduce management workload, we investigated web-based remote control of optical switches using IoT devices in an inexpensive way in our research [25], and investigated the way to update legacy optical switch and Ethernet switch with SDN-enabled optical switch and Ethernet switch at low cost [26,27]. The advantage of our solution is that it can be adopted in high-speed network with low-cost, because it can use both a powerful packet-exchanging performance of hardware in the legacy network devices and many SDN functions installed in the inexpensive IoT devices.

To achieve cost-effective SDN deployment and reduce network management workload, we propose a router and switch control system using cost-effective IoT devices. We developed and designed functional components in the proposed system and demonstrated the system with legacy network devices. Besides, we implemented a function to translate SDN-based control commands into vendor-specific control commands in IoT devices. A control server was introduced in the system. A web server, SDN controller, and database were implemented in the control server. From the cost viewpoint for the system construction, the use of IoT devices, which are even more inexpensive than general-purpose servers, has significantly reduced the cost for SDN deployment. Then, it is possible to update legacy network devices to SDN-enabled network devices cost-effectively. From the reducing management workload viewpoint, web-based data processing system [28] was introduced, making it possible to manage and control the system via WWW. It provided the advantage of easy integration with other systems and functions, making it flexible management. Besides, it is independent of where the SDN controller is installed. The database-oriented management system has also been investigated based on reducing the network management workload on network administrators [29]. In the proposed system, a database has been implemented to accumulate information on the network status. It made it highly functional so that network administrators can refer to the current network status via WWW.

The remainder of the paper is organized as follows: In Section 2, we propose a router and switch control system using IoT devices, which have translation function from SDN-based control commands to vendor-specific control commands. In Section 3, we describe and review the demonstration of the proposed system on an experimental network. In Section 4, we describe the issues that must be addressed in the future. Finally, Section 5 presents the conclusion.

2. Router/Switch Control System.

2.1. System overview.

Figure 1 shows the architecture overview of the proposed system. The proposed system is divided into three layers: user, control, data planes. The user plane consists of the user interface (UI) implemented on the manager personal computer (PC) for the network administrator. Using the UI, the network administrator can issue control commands to legacy network devices and refer to the current network status. In the control plane, the web server, database, and SDN controller, are located and cooperate with each other. The SDN controller can also cooperate with various applications (App) using server-side function libraries. Besides, the SDN controller maintains SDN connections between each IoT device and between SDN-enabled network devices to send and receive control commands. The data plane is a mix of SDN-enabled network devices and legacy Ethernet switches and routers. In data plane, not only general packets but also packets of control commands flow in the network. In this paper, the IoT device means a device that can provide an interface between each thing and IP network for communicating each other. In Figure 1, the IoT device provides a communication interface between each legacy network device and the SDN controller over IP network.

2.2. System configuration.

Figure 2 shows the configuration of the proposed system. Node-1 in Figure 2 is assumed to be the network administration node, where the manager PC and the control server are located for network management. All nodes are equipped with many IoT devices to update to the SDN-enabled network devices. Each IoT device is connected to a legacy Ethernet switch or a legacy router on a one-to-one basis. It can translate SDN-based control commands into vendor-specific control commands. As shown in Figure 1, in Node-1, the web server, database, and SDN controller, are installed

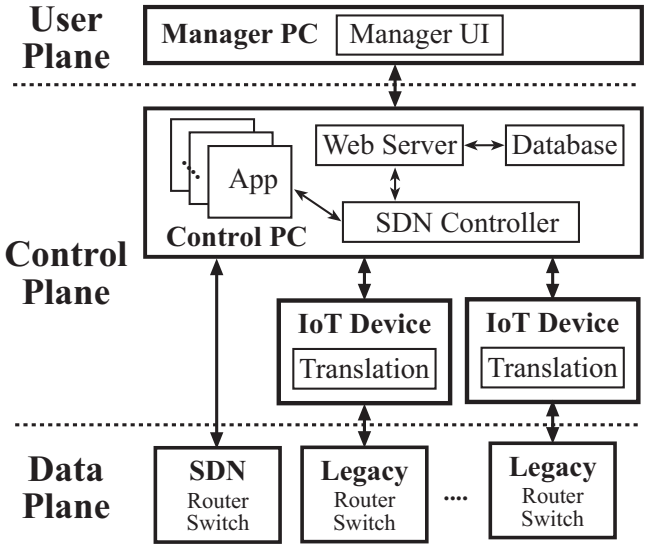


FIGURE 1. Architecture overview of the proposed system

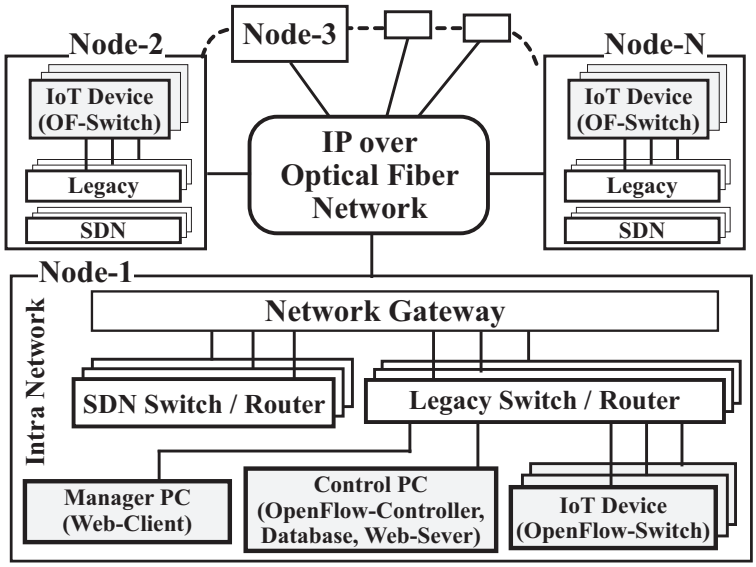


FIGURE 2. Configuration of the proposed system

in the control server. The web server constantly accepts the requests from the network administrator, and appropriately processes the accepted requests. The SDN controller establishes connections with each OpenFlow (OF) switch which is installed in each IoT device and each SDN-enabled network device. In Figure 2, Node-2 has legacy Ethernet switch or router and an SDN-enabled switch or router and IoT device. Besides, SDN controller establishes connection with each IoT device and SDN-enabled switch or router as in Node-1. From Node-3 to Node-N, the configuration is the same as Node-2. In Node-1, a web browser is installed in the manager PC as a web client, and as for the manager PC, it can exist in any node, not just Node-1. The network administrator can control and manage the network by the browser.

2.3. **System advantages.** The advantages of the proposed system are presented. From Figures 1 and 2, the function installed in the IoT devices is used to translate the SDN-based control commands, which have been sent from the SDN controller into vendor-specific control commands. The function makes it possible to centrally control and manage legacy Ethernet switches and routers using SDN technology in a multivendor environment without replacing them with expensive SDN-enabled network devices. The use of inexpensive IoT devices can significantly reduce the cost of deploying the SDN management environment. The translation function of the control commands installed in IoT devices has security advantages. For example, legacy Ethernet switches and routers may only support older protocols, such as Telnet, which does not encrypt packets, as the protocol for sending and receiving control commands. As shown in Figure 2, if the translation function of the control commands is located in the control server, unencrypted Telnet packets may flow over the public IP network, which is a high-security risk. When the control commands are translated at the IoT devices, the control commands can be sent based on a secure protocol through SDN connections up to the node where the target legacy Ethernet switches and routers are located. Eventually, some unencrypted control commands may flow within the node. However, they will not flow over the public IP network, so the risk is low. In Figure 2, the manager PC is located in Node-1. Besides, it can be located in any other node on the network. It means that from any node, the network administrator can operate the manager PC and then control and manage the Ethernet switches and routers through the control server. This function updates the network management system that depends on the location of the SDN controller to the flexible network management system that is independent of the location. The web server and database have cooperated. Thus, it is possible for the network administrator to refer to the information on the current network status stored in the database from the web client. These functions can significantly reduce the network administrator management workload. Besides, even if the connection between the IoT device and the SDN controller is broken and remote control is no longer possible, the settings of legacy network devices can be changed by using the vendor-specific control commands.

2.4. **Functional components and processing flow.** Figure 3 mainly shows the configuration of the functional components and the processing flow in the proposed system. First, network administrator operates the web browser in the manager PC and selects the configuration item for the legacy network devices to be controlled. The selected items are the functions that legacy Ethernet switches and routers originally have from layer-1 (physical), layer-2 (data-link), layer-3 (network). For example, a function included in layer-3 is rewriting the routing table of the legacy network device. Other functions in layer-2 are forwarding database rewriting and port trunking. Layer-1 functions are packet filtering and enable/disable switch for each port. Next, the SDN-based control commands are sent to the control server according to the selected configuration items. The web server in the control server receives it. Then, the database is rewritten according to the SDN-based control commands. The SDN-based control commands are passed to the OpenFlow controller. The OpenFlow controller sends the received SDN-based control commands to the IoT device according to the SDN protocol. In this study, we adopted the OpenFlow as the protocol for SDN [8,30-32]. The OpenFlow switch in the IoT device receives the OpenFlow control commands that are SDN-based control commands and translate them into vendor-specific control commands through the adapter. Finally, the desired configuration is completed by sending vendor-specific control commands from the IoT device to the legacy network device to be controlled through Telnet-interface (IF). Next, as shown in Figure 3, the database reference function is presented. First, network administrator

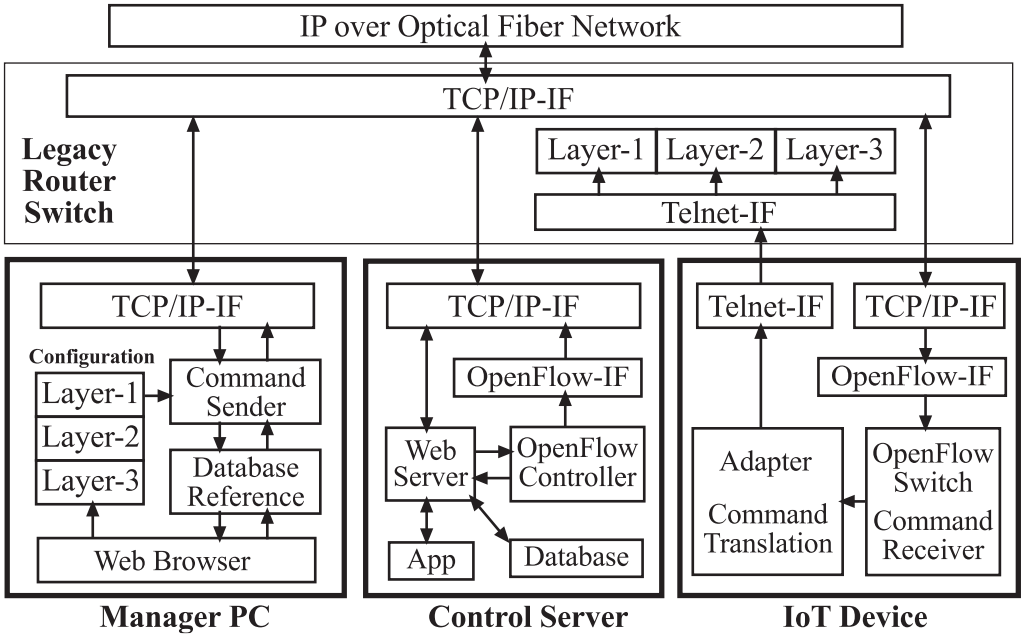


FIGURE 3. Configuration and processing flow of functional components

operates the web browser in the manager PC and selects database reference function. This database reference function is a function which displays the setting information of the managed legacy Ethernet switch or router on the manager PC. Next, a control command is sent to the control server to refer to the database based on the selected database reference function and then, the control command is received by the web server and the desired setting information of the legacy Ethernet switch or router is extracted from the database based on the control command and sent back to the manager PC. Finally, the desired setting information of the legacy Ethernet switch or router is displayed on the manager PC. Besides, the App in the control server shown in Figure 3 is intended to be a web application like a database reference function, and network administrator can use the same way as the database reference function.

3. **Experimental Demonstration.** We developed and implemented the functions of the proposed system and demonstrated whether the system works properly using two experiments. In this section, we describe the experimental setup used for the demonstration and procedure of each experiment and show the experiment results.

3.1. **Experimental setup.** Figure 4 shows the experimental setup. Two legacy SDN-non-supported layer-3 switches (L3SW) were prepared. A management PC, a control server, and an IoT device implementing the functional configuration (Figure 3) were connected to the left layer-3 switch in Figure 4. A general-purpose PC was used as the management PC. A general-purpose server was also used as the control server. As an IoT device, we adopted Raspberry Pi 3 [33] and Wireshark [34] for the local area network (LAN) analyzer installed on the general-purpose PC.

3.2. **Experimental procedure.**

(I) **System Verification with IP-Throughputs Changes.** As shown in Figure 4, the left and right layer-3 switches indicate IP traffic sending and receiving, respectively. We monitored the sending and receiving IP throughputs. Besides, we confirmed that the proposed system performs properly from the IP-throughput changes. First, a 1 Gbps communication link was established between the sending and receiving layer-3 switches.

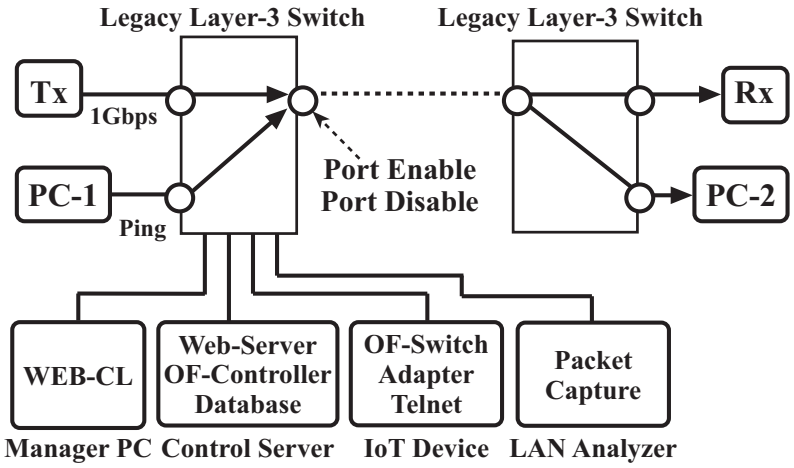
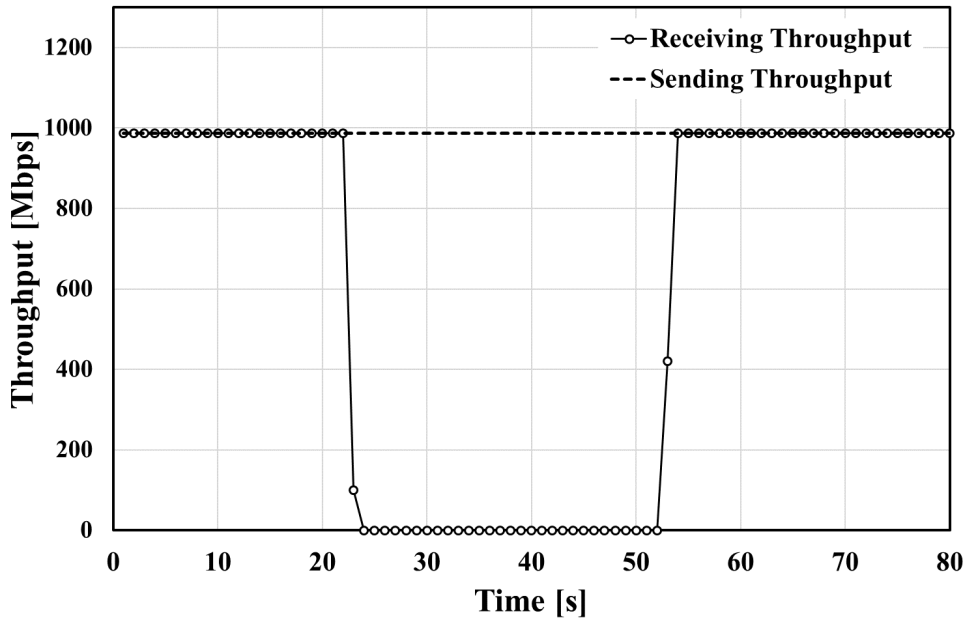


FIGURE 4. Experimental setup

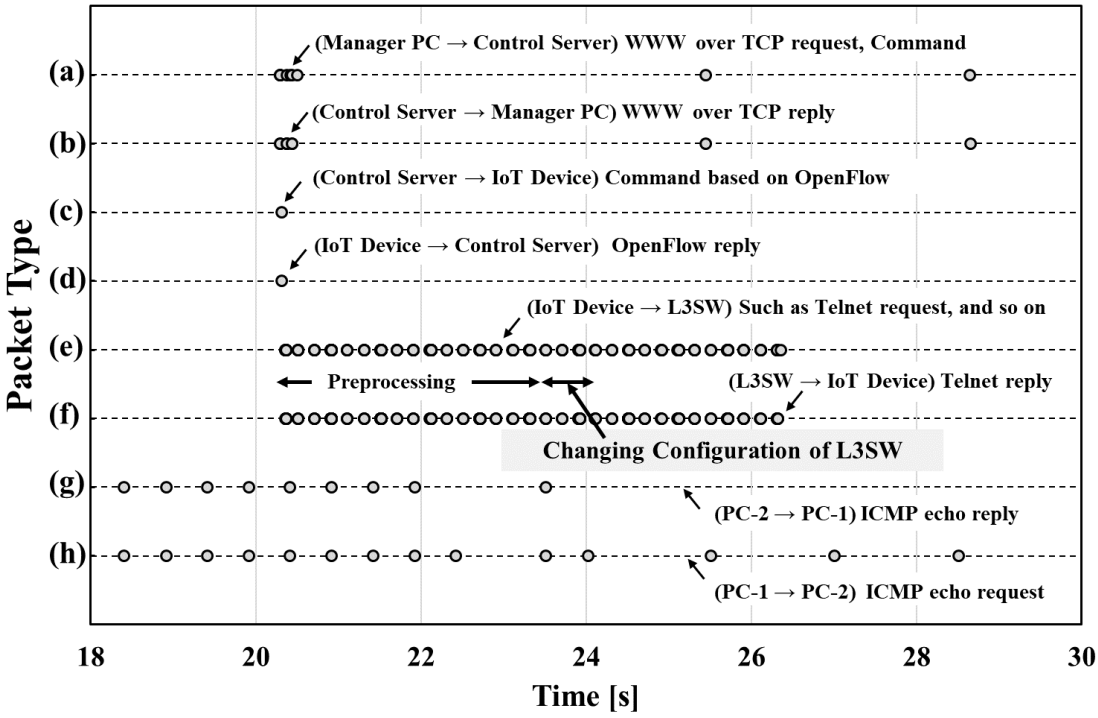
Then, constant IP traffic was kept flowing from the sending side to the receiving side using a router tester (Tx and Rx in Figure 4). We issued control commands to make the port enable/disable based on the processing flow shown in Figure 3 and monitored the IP throughput change. The control commands were issued in about 20 and 50 s after the traffic started flowing.

(II) **System Verification with Packet Analysis.** We used the LAN analyzer shown in Figure 4 to capture IP packets and verified that the proposed system performs properly. First, we executed the Ping command PC-1 and continuously sent Internet control message protocol (ICMP) echo request packets to PC-2. We issued control commands to make the port enable/disable as shown in Experiment I. All IP packets were collected and analyzed before and after the control commands were issued using Wireshark, which is one of the packet capture tools. The collected IP packets were four types: ICMP echo request and reply packets between PC-1 and PC-2, Web-based packets between the manager PC and the control server related to the processing flow shown in Figure 3, OpenFlow-based packets between the control PC and IoT device, and Telnet-based packets between the IoT device and left layer-3 switch.

3.3. **Results of experimental demonstration.** Figures 5(a) and 5(b) show the results of Experiments (I) and (II), respectively. Figure 5(a) shows the change in sending and receiving IP throughputs. First, both throughputs were about 987 Mbps. The control commands were issued to change the port in the left layer-3 switch in Figure 4 from the enable to disable, causing the receiving IP throughput to change from 987 Mbps to 0 bps after about 23 s. Since there is no change in the sending throughput, it indicates that the trouble is not in the sending function of the router tester. The results showed that the communication link between layer-3 is turned off and the packet forwarding function is disabled. Finally, the receiving throughput returned to about 987 Gbps after about 53 s by issuing a control command to turn the communication link again. This result indicates that the communication link has been turned on and the packet forwarding function has been enabled. Figure 5(b) shows the ICMP echo packets captured by the LAN analyzer between PC-1 and PC-2 and packets based on the processing flow shown in Figure 3 in chronological order. Types of packets (g) and (h) show the sending and receiving ICMP echo packets between PC-1 and PC-2. No ICMP echo reply packet was observed after changing the configuration of the layer-3 switch. Types of packets (a) and (b) show the communication packets between the manager PC and control server. Types of packets (c) and (d) show the communication packets between the control server and IoT devices.



(a) IP-Throughput change



(b) Packet analysis

FIGURE 5. Results of experiments

Types of packets (e) and (f) show the communication packets between the IoT device and layer-3 switch. The result of the packet analysis confirmed that the control commands were properly delivered to the IoT device and sent to the layer-3 switch using Telnet. The configuration was properly changed. The results of these two experiments showed that the proposed system performs properly. And then, the result also shows that our proposed

system can reduce the work load and construct an SDN-enabled environment with low-cost, because it can provide the centralized remote router/switch control functions based on SDN by using inexpensive IoT devices and a web server.

The cost of the hardware and operating system (OS) used in this research are shown in Table 1 in order to compare with another research in view point of cost. The cost of an SDN-enabled white-box switch is about $2,900 to $3,900, depending on the number of ports and switch performance [24]. Besides, the cost of the Raspberry Pi 3 model B+ is about $48 [35], and its OS, Raspbian, is open free. We also used HP Elite Desk 705 G3 SFF which was purchased for about $590 as a general-purpose server and used CentOS 7.9 as OS which is open free. The OpenFlow controller in the general-purpose server was developed using Ryu [36], which is open free source and the adapter function in the Raspberry Pi was developed using Python, which is also open free. Table 1 shows that it is much cost-effective in SDN deployment to combine newly added Raspberry Pi with legacy network devices already presented in the network instead of adding new SDN-enabled network devices. For the reasons, the proposed system can achieve the reducing management workload of network administrator and SDN deployment of low cost.

Table 1. Comparison about cost and hardware/operating system

	[24]		The proposed system	
Hardware/OS	SDN-enabled white-box switch		Raspberry Pi 3 model B+	General-purpose server (HP Elite Desk 705 G3 SFF)
	24 ports/ 10 Gbps	40 ports/ 10 Gbps	Raspbian 10 (buster)	CentOS 7.9
Cost	$2,900	$3,900	$48	$590

4. **Discussion.** We consider that the proposed system can be improved in cost. Figure 6 shows an overview of the advanced proposed system in cost reduction. The difference from Figure 1 is the number of IoT devices. In Figure 1, there is a one-to-one correspondence between IoT devices and legacy Ethernet switches and routers. However, in Figure 6, there is a one-to-many correspondence. It means that multiple legacy Ethernet switches

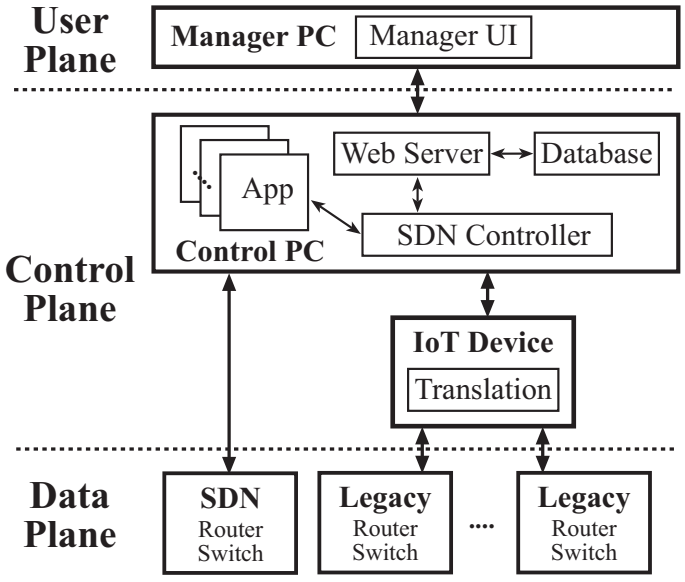


Figure 6. Overview of an advanced network management system

and routers can be supported by a single IoT device. This advanced proposed system will achieve SDN-based network management systems in multivendor environments that are more cost-effective. To achieve a more cost-effective system, we must determine how many legacy Ethernet switches and routers can be supported by a single IoT device in the future.

5. **Conclusions.** We proposed a cost-effective IP over the optical network management system over WWW with database function using inexpensive IoT devices to solve the increasing cost problem in SDN deployment and the dependency of SDN controller on network management. The proposed system has been developed and implemented. We conducted an experimental demonstration to verify the proposed system using IP throughput monitoring and packet analysis. The experimental results showed that the proposed system performs properly. The improvement of the proposed system is discussed from the point of cost. The outline of the improved system and the direction of future research are described.

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

REFERENCES

[1] M. Kahani and H. W. P. Beadle, Decentralised approaches for network management, *ACM SIGCOMM Computer Communication Review*, vol.27, no.3, pp.36-47, 1997.

[2] F. L. Koch and C. B. Westphall, Decentralized network management using distributed artificial intelligence, *Journal of Network and Systems Management*, vol.9, pp.375-388, 2001.

[3] R. D. C. Andrade, H. T. Macedo, G. L. Ramalho and C. A. Ferraz, Distributed mobile autonomous agents in network management, *Proc. of International Conference on Parallel and Distributed Processing Techniques and Applications*, 2001.

[4] S. Abar, S. Konno and T. Kinoshita, Autonomous network monitoring system based on agent-mediated network information, *International Journal of Computer Science and Network Security*, vol.8, no.2, pp.326-333, 2008.

[5] K. Sasai, J. Sveholm, G. Kitagata and T. Kinoshita, A practical design and implementation of active information resource based network management system, *International Journal of Energy, Information and Communications*, vol.2, no.4, pp.67-86, 2011.

[6] 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.

[7] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky and S. Uhlig, Software-defined networking: A comprehensive survey, *Proc. of the IEEE*, vol.103, no.1, pp.14-76, 2014.

[8] F. Hu, Q. Hao and K. 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.

[9] W. Xia, Y. Wen, C. H. Foh, D. Niyato and H. Xie, A survey on software-defined networking, *IEEE Communications Surveys & Tutorials*, vol.17, no.1, pp.27-51, 2015.

[10] Y. Li and M. Chen, Software-defined network function virtualization: A survey, *IEEE Access*, vol.3, pp.2542-2553, 2015.

[11] A. C. Baktir, A. Ozgovde and C. Ersoy, How can edge computing benefit from software-defined networking: A survey, use cases, and future directions, *IEEE Communications Surveys & Tutorials*, vol.19, no.4, pp.2359-2391, 2017.

[12] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller and N. Rao, Are we ready for SDN? Implementation challenges for software-defined networks, *IEEE Communications Magazine*, vol.51, no.7, pp.36-43, 2013.

[13] R. Alvizu, G. Maier, S. Troia, V. M. Nguyen and A. Pattavina, SDN-based network orchestration for new dynamic enterprise networking services, *Proc. of the 19th International Conference on Transparent Optical Networks*, Girona, Catalonia, Spain, pp.1-4, 2017.

[14] J. Liu, Y. Li, M. Chen, W. Dong and D. Jin, Software-defined Internet of things for smart urban sensing, *IEEE Communications Magazine*, vol.53, no.9, pp.55-63, 2015.
[15] W. Yu, B. Zhao and Z. Yan, Software defined quantum key distribution network, *Proc. of the 3rd IEEE International Conference on Computer and Communications*, Chengdu, China, pp.1293-1297, 2017.
[16] N. Huin, M. Rifai, F. Giroire, D. L. Pacheco, G. Urvoy-Keller and J. Moulierac, Bringing energy aware routing closer to reality with SDN hybrid networks, *IEEE Trans. Green Communications and Networking*, vol.2, no.4, pp.1128-1139, 2018.
[17] 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.
[18] 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.
[19] D. H. Chen, W. M. Wang, I. H. Chung and C. F. Chou, Incremental hybrid SDN deployment for enterprise networks, *Proc. of the 2017 IEEE 15th International Conference on Dependable, Autonomic and Secure Computing*, Orlando, FL, USA, pp.1143-1149, 2017.
[20] T. Feng, J. Bi, P. Xiao and X. Zheng, Hybrid SDN architecture to integrate with legacy control and management plane: An experiences-based study, *Proc. of 2015 IFIP/IEEE International Symposium on Integrated Network Management*, Ottawa, Canada, pp.1143-1149, 2015.
[21] 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.
[22] A. N. Toosi, J. Son and R. Buyya, CLOUDS-Pi: A low-cost Raspberry-Pi based micro data center for software-defined cloud computing, *IEEE Cloud Computing*, vol.5, no.5, pp.81-91, DOI: 10.1109/MCC.2018.053711669, 2018.
[23] D. J. Casey and B. E. Mullins, SDN shim: Controlling legacy devices, *Proc. of the 2015 IEEE 40th Conference on Local Computer Networks*, Clearwater Beach, FL, USA, pp.169-172, 2015.
[24] 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.
[25] 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.
[26] 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.
[27] 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.
[28] D. Lowe, Web system requirements: An overview, *Requirements Engineering*, vol.8, pp.102-113, 2003.
[29] Y. Kawai, Y. Sato, S. Ata, D. Huang, D. Medhi and I. Oka, A database oriented management for asynchronous and consistent reconfiguration in Software-Defined Networks, *Proc. of 2014 IEEE Network Operations and Management Symposium*, Krakow, Poland, DOI: 10.1109/NOMS.2014.6838333, 2014.
[30] W. Braun and M. Menth, Software-defined networking using Open-Flow: Protocols applications and architectural design choices, *Future Internet*, vol.6, no.2, pp.302-336, 2014.
[31] 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.
[32] *OpenFlow*, https://www.opennetworking.org/, Accessed in July, 2021.
[33] *Raspberry Pi*, https://www.raspberrypi.org/, Accessed in July, 2021.
[34] *Wireshark*, https://www.wireshark.org/, Accessed in July, 2021.
[35] *The Pi Hut*, https://thepihut.com/, Accessed in July, 2021.
[36] *RYU SDN Framework*, https://ryu-sdn.org/, Accessed in July, 2021.