IEICE TRANS. COMMUN., VOL.E96–B, NO.1 JANUARY 2013

PAPER

RISE: A Wide-Area Hybrid OpenFlow Network Testbed*

Yoshihiko KANAUMI†a), Shu-ichi SAITO†,††, *Nonmembers*, Eiji KAWAI††, *Member*, Shuji ISHII††, *Nonmember*, Kazumasa KOBAYASHI††,†††, *and* Shinji SHIMOJO††,††††, *Members*

SUMMARY The deployment of hybrid wide-area OpenFlow networks is essential for the gradual integration of OpenFlow technology into existing wide-area networks. Integration is necessary because it is impractical to replace such wide-area networks with OpenFlow-enabled ones at once. On the other hand, the design, deployment, and operation of such hybrid OpenFlow networks are often conducted intuitively without in-depth technical considerations. In this paper, we systematically discuss the technical aspects of the hybrid architecture for OpenFlow networks based on our experience so far in developing wide-area hybrid OpenFlow networks on JGN2plus and JGN-X, which are nation-wide testbed networks in Japan. We also describe the design and operation of RISE (*R*esearch *I*nfrastructure for large-*S*cale network *E*xperiments) on JGN-X, whose objective is to support a variety of OpenFlow network experiments.

key words: *OpenFlow, NWGN, testbed networks, operation*

1. Introduction

The Internet succeeded in acquiring the position of the most important infrastructure for cultivating an information society. However, its fundamental architecture must overcome several technical issues to enable this society to grow sustainably. Many researchers around the world have been attacking these issues and publishing technologies that renovate the current Internet architecture. These are called Future Internet (FI) or New Generation Network (NWGN) Technologies (we use the latter term in this paper).

To promote the R&D (research and development) of NWGN technologies worldwide, NICT (National Institute of Information and Communications Technology) is operating JGN-X [2], a large-scale distributed network testbed. In this study, we develop an OpenFlow [3] testbed called RISE (*R*esearch *I*nfrastructure for large-*S*cale network *E*xperiments) on top of JGN-X.

OpenFlow is one of the most promising technologies for allowing flexible programmability in its networks. It is based on Ethernet technology and exports the packet forwarding control interfaces to an OpenFlow controller (OFC) outside an OpenFlow switch (OFS), i.e., it implements the separation of the data plane and the control plane in the networks. In OpenFlow networks (OFNs), we can therefore control packet forwarding freely on each switch, violating the Ethernet forwarding principles. This feature attracts much attention from researchers and developers in the area of NWGN technologies because they can implement their own NWGN technology in OFNs. In addition, OpenFlow is also recognized as a technology that provides isolated logical network slices, which can accommodate NWGN experimental networks as well as production networks.

In this paper, we propose the RISE architecture, a wide-area OpenFlow testbed, and compile our experiences of developing and deploying this technology. Our empirical study contributes to the following three areas.

First, we are focusing on establishing wide-area OFNs. At the time when OpenFlow was initially invented, campus networks were the major target for its deployment [4], [5], which allowed several advantages. For example, campus networks provide us with more realistic experimental environments because they are much larger than the in-laboratory experimental networks usually set up for academic research. In addition, new technologies deployed in OpenFlow-based campus networks have a better opportunity to transfer production network traffic generated by many advanced real users such as students, faculties, and administrative staff. The organization of campus networks is usually geographically-concentrated, and therefore it is relatively easy to install, configure, and operate new (and unstable) technologies. However, from the viewpoint of NWGN technology deployment, we need to expand this limited scale of OpenFlow experimental environments to that of wide-area networks.

Therefore, we have been developing a wide-area OpenFlow network testbed since 2009 [6]. We took the approach of a hybrid OFN architecture, i.e., our OFNs are built on top of JGN2plus [7] and JGN-X** [2] networks. One reason for this is that the price of wide-area broadband networks is tremendously high and we cannot purchase one to dedicate to our OpenFlow testbed. Another, more important, reason is that we need to develop an OpenFlow transition method, i.e., an incremental deployment method for OFNs into the

Manuscript received March 2, 2012.
Manuscript revised August 7, 2012.

†The authors are with NEC Corporation, Tokyo, 108-8001 Japan.
††The authors are with National Institute of Information and Communications Technology, Tokyo, 100-0004 Japan.
†††The author is with Kurashiki University of Science and the Arts, Kurashiki-shi, 712-8505 Japan.
††††The author is with Osaka University, Ibaraki-shi, 567-0047 Japan.

*This article is the extended version of our previous literature [1]. We added a new section (Sect. 6) and revised the content thoroughly to include more detailed discussion.

a) E-mail: y-kanaumi@jgn-x.jp

DOI: 10.1587/transcom.E96.B.108

**JGN2plus is the former version of JGN-X.

existing network infrastructure. To achieve this, we have to consider the requirements of both the existing networks and the OFNs.

The second contribution is to share our knowledge of wide-area OFNs with other R&E (Research and Education) network organizations [8], [9]. To make our wide-area and hybrid OFNs work correctly, we need to leverage some implementation-dependent functions of communication devices, as well as those described in the OpenFlow specifications. Accordingly, we focus on the practical techniques required to establish real-world OFNs and their technology transfer to the world.

Finally, we operate our wide-area OFNs as a RISE OpenFlow testbed. To achieve this, we develop a service model, i.e., a framework to accept experimental OFCs. Although the basic mechanism to share OFNs among multiple users is available in the famous FlowVisor [10], we must sometimes provide completely independent OFNs simultaneously to multiple users. As we mentioned before, our OFNs are built on existing networks, which incurs some limitations in functionality and performance. By considering these limitations, as well as the scenarios in wide-area OpenFlow network experiments, we have developed the basic architecture of the RISE testbed.

The rest of this paper is organized as follows. We discuss the necessity of wide-area OFNs in Sect. 2, before describing the technical requirements of both OFNs and existing wide-area networks in Sect. 3. The RISE testbed architecture, which satisfies these requirements, is presented in Sect. 5. We summarize our experiences of deploying and operating RISE in Sect. 6, and conclude this paper in Sect. 7.

Details of the OpenFlow technology are not described in this paper. Please refer to the OpenFlow specifications[†] [11] for more detailed information. We have also omitted a detailed description of the experiments conducted during deployment of RISE. Please refer to our previous literature [6], [12].

2. Motivation for RISE

In this section, we describe the background of OpenFlow as an infrastructure technology for NWGN and outline our motivation for developing RISE, a wide-area OpenFlow testbed.

2.1 Issues in Experimental Environments for NWGN Technologies

To overcome the limitations of the current Internet architecture, a wide-variety of NWGN technologies have been developed. One of the challenges in the development of NWGNs is finding wide-area experimental environments for newly developed technologies. Because the Internet is now playing the role of the most fundamental infrastructure for the global information society, it is highly difficult to accept major changes in its architecture, even if the aim of the changes is to verify novel technologies for the renovation of the Internet. More specifically, we need to consider the following three issues.

The first issue is security. For all organizations and persons, security is crucial, and an inappropriate response to a security incident can erode their public confidence in a technology as well as cause economic loss. Therefore, even in campus networks where many R&D activities are conducted, there are many restrictions on using new technologies, and it is hard to prepare a large-scale experimental environment.

The second issue is service continuity. Quite naturally, newly developed technologies cannot accumulate much knowledge of their large-scale operation, and therefore may sometimes cause a wide-area functional failure. Because the Internet is now operating as an essential infrastructure for the information society, discontinuity in its service is unacceptable for many users.

Finally, we need to consider the economics of R&D. Recent network equipment deployed in wide-area networks, such as switches and routers, is often implemented with dedicated, proprietary hardware. This means that academic researchers must conduct development jointly with the companies that develop and release such equipment in order to deploy their original technologies in real wide-area networks. On the other hand, the recent development of network equipment is highly competitive and its cost is skyrocketing. Therefore, the integration of new technologies into existing systems is often an unacceptable risk for many network equipment companies.

2.2 Advantages of OpenFlow

OpenFlow has attracted much attention as a platform that can solve the issues of NWGN experimental environments mentioned in the previous subsection. One of the major features of OpenFlow is its separation of the data plane and the control plane (Fig. 1).

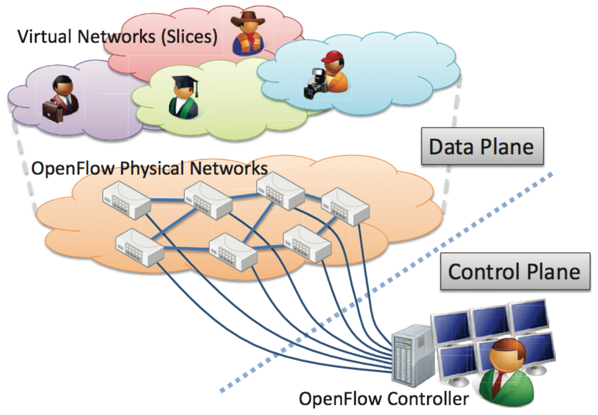


Fig. 1 Separation of the data plane and the control plane in OpenFlow.

[†]In this paper, we refer to version 1.0 of the OpenFlow specifications because its implementation is widely available. At the time of writing this paper, there is a newer OpenFlow specification (version 1.1), but only a small number of OpenFlow devices support it.

In the data plane, OpenFlow adopts the architecture of current L3 switches without major modification. This allows network equipment companies to utilize existing sophisticated hardware technology, and therefore they can afford to develop OFSs. In addition, this Ethernet-friendly architecture of OpenFlow makes it easy to introduce OFNs into existing networks.

In the OpenFlow control plane, a centralized architecture is adopted. Users can freely control packet forwarding on each switch via the OFC connected to all OFSs. Utilizing this controllability, users can develop their own mechanisms for NWGN such as routing, traffic engineering, multi-casting, and mobile communications, and easily deploy them to the networks by installing them in the controller. In addition, from an administrative viewpoint, the centralized architecture is favorable for security management as well as trouble shooting.

We can also develop network virtualization mechanisms in OFNs by leveraging the flexible packet forwarding controllability. FlowVisor provides in-flowspace separation mechanisms between networks, which allow us to configure experimental networks that are logically separated from the production networks shared by all users.

2.3 Motivation for Wide-Area OpenFlow Networks

Although OpenFlow started with campus networks as its initial deployment target, we cannot ignore its wide-area deployment, especially from the perspective of NWGN R&D. Clearly, NWGN technologies should cover global networks, and OFNs should therefore be deployed worldwide and support global NWGN experiments.

Accordingly, we have been developing and operating wide-area (nationwide) OFNs on top of JGN2plus and JGN-X since 2009. To the best of our knowledge, there were no such wide-area OFNs when we started this project. We devised the architecture of our wide-area OFNs based on the Ethernet Q-in-Q mechanism, which we will mention later. Our empirical knowledge was shared with other R&E network developers and operators through documents [6], [12] and presentations [8], [9].

3. Consideration of Wide-Area Deployment of OpenFlow

In this section, we describe what we considered in for deploying wide-area OFNs and operating them as an R&D testbed.

3.1 OpenFlow Functions

Before designing wide-area OFNs, it is important to consider what mechanisms they should incorporate. OpenFlow provides extremely flexible control of packet forwarding, and allows users to implement a wide variety of networking mechanisms. Ultimately, users can replace the current Internet architecture with one of their own. For example, OFN users can develop original mechanisms for network virtualization.

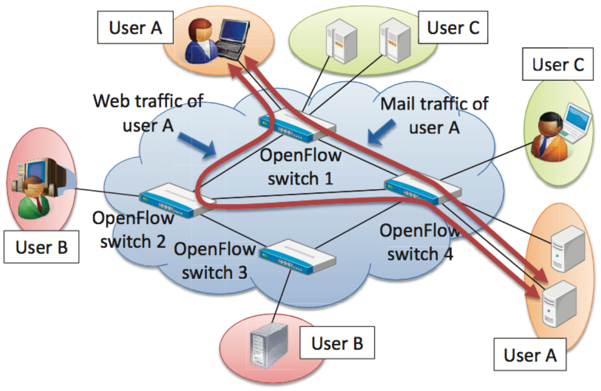


Fig. 2 Traffic engineering with OpenFlow.

It must be remembered, however, that OpenFlow is a developing technology and involves various implementation-dependent issues. Therefore, we should first define the function and performance requirements in OFNs, and then elaborate their composition in order to meet these requirements. In this study, we set the goal of advanced traffic engineering for each flow in our wide-area OFNs (Fig. 2). That is, end hosts connected to the OFNs generate traditional TCP/IP packets and their traffic engineering is operated with OpenFlow.

The details of the traffic engineering in our OFNs can be described as follows. In each OFSs, one or more dedicated physical ports are allocated to each user. The traffic flows that are controlled in our OFNs are defined with the OpenFlow tuple, except for 802.1Q VLAN tags [13]. Users can define a target flow for control by physical ports, MAC addresses, IP addresses, TCP/UDP port numbers, and so on. For example, the packets in a single TCP connection can be defined as a flow, and OpenFlow can define their control rules.

In our current OFN design, the 802.1Q VLAN tags are used for an administrative purpose — we identify the user of each packet by the VLAN ID attached to it. In other words, VLAN IDs are employed as user identifiers. With this mechanism, we can configure unique slices to allow each user free usage of all the other tuple space in their slice. In other words, a user need not consider which MAC addresses, IP addresses or TCP/UDP port numbers other users are using in our OFNs. In the example shown in Fig. 2, the control of web traffic for user A does not affect any kind of traffic control for user B and C, including web traffic control.

The administrative design whereby we identify users by the physical ports they use has another advantage from the viewpoint of network operation. When a specific user's traffic causes network trouble due to an unrecoverable error, we can stop this traffic by shutting down their physical ports. We adopt this model for accommodating users in the RISE testbed (we describe RISE in detail in Sect. 5).

On the other hand, users in our OFNs cannot utilize

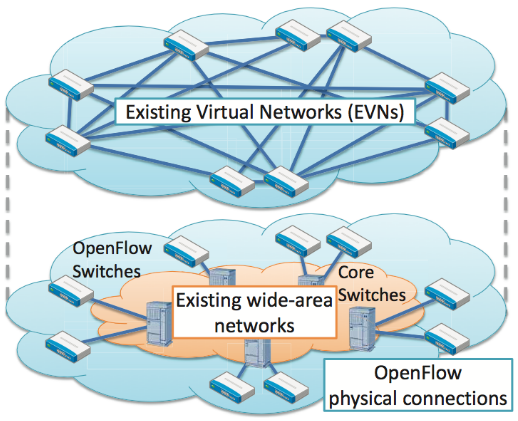

Existing Virtual Networks (EVNs)
OpenFlow Switches
Core Switches
Existing wide-area networks
OpenFlow physical connections

Fig. 3 OFNs in EVNs.

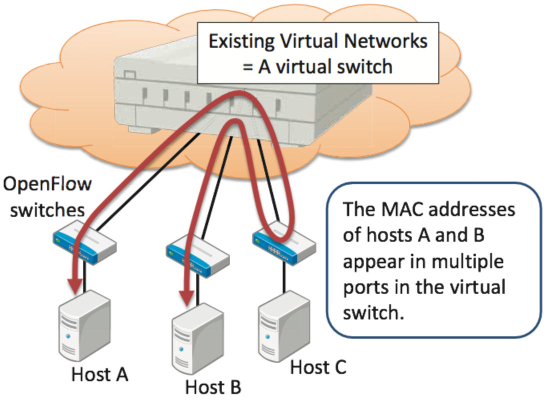

Existing Virtual Networks = A virtual switch
OpenFlow switches
The MAC addresses of hosts A and B appear in multiple ports in the virtual switch.
Host A
Host B
Host C

Fig. 4 The problem of MAC address learning in EVNs.

the VLAN tags for their own traffic control. This imposes a limitation on connecting user networks in which VLAN tags are utilized to our OFNs. To overcome this limitation, we can divide the 12-bit VLAN tag into two parts and provide one to the users, if the number of users is small enough. However, this requires per-packet processing of the VLAN tag and causes an additional overhead.

3.2 Deployment Techniques for OpenFlow Networks

We took the approach of deploying OFNs virtually in existing wide-area networks. This is because the enormous cost of dedicated wide-area communication lines between the OFSs was impractical for exploiting the possibility of wide-area deployment of OpenFlow when this project commenced. In addition, OpenFlow data packets follow the traditional Ethernet packet format, and it is theoretically possible to transfer OpenFlow data packets in traditional Ethernet networks. Therefore, we have investigated how to configure virtual networks in existing wide-area networks and accommodate OFNs within them (Fig. 3).

In this paper, the virtual networks that accommodate the OFNs are called as *Existing Virtual Networks* (EVNs). We consider the following issues of EVNs:

- Utilization of a wide-area Ethernet technology
- Avoidance of MAC address learning

3.2.1 Wide-Area Ethernet Technologies for EVNs

We need to choose virtual network technologies for the accommodation of OFNs. Basically, OpenFlow data packets are stored in the Ethernet format, and the EVNs must transfer the Ethernet packets from one OFS to another OFS without modification. There are a variety of wide-area Ethernet technologies in each network layer that provide this kind of virtual Ethernet connectivity (Ethernet over Ethernet, MPLS, IP, UDP, and so on).

At this time, we believe Ethernet- and MPLS-based technologies are the best for accommodating OFNs. An advantage of OpenFlow is its explicit and fine-grained control over packet forwarding. Therefore, it is preferable that we can statically configure the EVNs, which is allowed by Ethernet- and MPLS-based virtual network technologies. When we utilize virtual Ethernet links based on an upper layer technology such as EtherIP [14], it is relatively difficult to control the actual (physical) paths, especially in wide-area networks.

We can also consider using virtual network technologies other than wide-area Ethernet to virtually extend the OpenFlow data plane. However, such technologies generally terminate Ethernet connections on the border between the OFNs and EVNs, i.e., they modify the MAC addresses or other Ethernet header fields in OpenFlow data packets. Therefore, when we use these technologies for EVNs, we cannot use the Ethernet header fields in the OpenFlow tuple space.

In addition, current OpenFlow implementations depend on some Ethernet layer technologies, although they are not formally documented in the OpenFlow specifications. Thus, when we utilize L3 or upper-layer (non-wide-area Ethernet) virtual networks for EVNs, we must solve any relevant implementation issues. For example, many OFSs use LLDP (Link Layer Discovery Protocol, IEEE 802.1AB [15]) to obtain link topology information.

Accordingly, it is currently the best practice to utilize wide-area Ethernet technologies for EVNs.

3.2.2 MAC Address Learning in EVNs

The next issue for consideration is MAC address learning in EVNs. As we have mentioned, OpenFlow uses Ethernet packets. OpenFlow also allows us to control each packet beyond the Ethernet packet forwarding principles. Therefore, if EVNs work as a virtual Ethernet switch with a MAC address learning mechanism, they may not be able to transfer the OpenFlow data packets correctly depending on the forwarding control applied to them (Fig. 4).

A solution to this problem is to separate the MAC address learning domain by introducing tunnels (point-to-point connections in wide-area Ethernet networks) between OFSs (Fig. 5). With this approach, EVNs can decide where to deliver each OpenFlow data packet by binding the packet to a tunnel, rather than by learning MAC addresses. In other words, we must notify the EVNs which tunnel is to be used for each packet by some other means than MAC addresses.

The simplest approach to achieve this is to utilize separate physical links between an OFS and a network switch of EVNs, as shown in Fig. 6(a). In this case, the number of physical links is the same as that of the tunnels in the EVNs. Herein, we call an EVN switch an *accommodation switch* and a physical link between an OFS and an accommodation switch a *boundary link*. To transmit an OpenFlow data packet from an OFS to an accommodation switch, the OFS selects the tunnel to the next hop OFS, i.e., the physical port of the boundary link for the tunnel, and then the accommodation switch just adds some tunnel information to the packet, thus avoiding any confusion from MAC address learning in the other physical ports. For the reverse communication, when the accommodation switch receives a packet from a tunnel, it removes the tunnel information

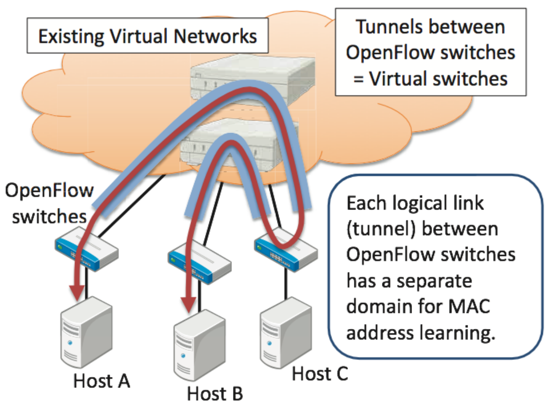

Existing Virtual Networks
Tunnels between OpenFlow switches = Virtual switches
OpenFlow switches
Each logical link (tunnel) between OpenFlow switches has a separate domain for MAC address learning.
Host A
Host B
Host C

Fig. 5 Separation of MAC address learning domains by tunnels.

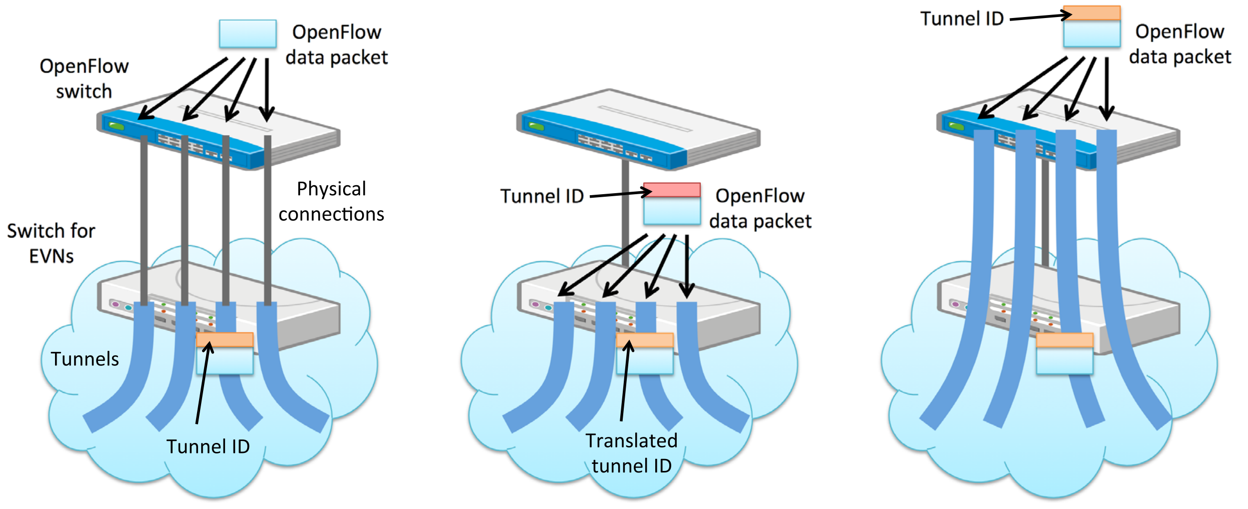
from the packet and forwards the packet to a boundary link corresponding to the tunnel. The receiving OFS then identifies the tunnel with the physical port of the boundary link.

When we cannot prepare as many physical boundary links as there are tunnels, we must extend the tunnels virtually to the OFS. For this, the OFS can add some information to a packet, and the accommodation switch uses this to identify which tunnel to forward the packet through, as shown in Fig. 6(b). In this case, we suppose that the virtualization (tunnel) techniques utilized in the OFS and the accommodation switch are different. For example, the OFS adds an 802.1Q VLAN tag to an OpenFlow data packet to notify the tunnel to which the accommodation switch should forward the packet. Then, the accommodation switch reads the VLAN tag in the packet and adds an MPLS header to the packet according to the switch configuration. Thus, the accommodation switch translates the tunnel IDs added by the OFS into those actually used in the EVNs.

If the virtualization techniques are the same, it means the tunnels are extended directly to the OFS, as shown in Fig. 6(c). Similar to the case in which different techniques are used, the OFS needs to add some information to the packet to identify the desired tunnel. We will discuss these virtualization techniques in the next subsection.

Another solution to the MAC address learning issue is to replace the MAC address with a safe one that does not confuse the EVNs. This replacement should be done in the OFS, and the OpenFlow specifications define this kind of MAC address modification capability in the OFS. However, the management of the MAC address translation will be highly complicated (theoretically, we need to define a unique MAC address for each logical link between the OFSs) and the per-packet processing cost of MAC address modification is not negligible.

Accordingly, we conclude that providing logical links



(a) Using as many boundary links as tunnels.

(b) Using a shared boundary link with tunnels terminated at the accommodation switch.

(c) Using a shared boundary link with tunnels extended to the OFS.

Fig. 6 Boundary links between an OFS and an accommodation switch.

between the OFSs by tunnels in the EVNs is the current best practice to deploy wide-area OFNs.

4. Wide-Area Ethernet

As discussed in the previous subsection, we adopted an approach to deploy our OFNs over wide-area Ethernet networks. We now briefly survey wide-area Ethernet technologies[†].

4.1 Wide-Area Ethernet with Ethernet

IEEE 802.1Q (Virtual LAN) [13], IEEE 802.1ad (Provider Bridges) [16], and IEEE 802.1ah (Provider Backbone Bridges) [17] are categorized as wide-area Ethernet with Ethernet technologies.

802.1Q Virtual LAN (also called tagged VLAN) is the most widely used technology for configuring a tunnel in the Ethernet layer (L2). In an 802.1Q VLAN, a 12-bit VLAN tag is added to the Ethernet header of a packet and a switch treats a series of packets with the same tag as those in a logically separated network (VLAN).

OpenFlow supports the manipulation of the 802.1Q VLAN tag in its specifications. We can use VLAN tags to control packet forwarding in OFNs, as well as for configuring tunnels in EVNs. Accordingly, when we utilize VLAN tags in EVNs, we must avoid using them in the OFNs or split the 12-bit space into separate spaces for OFNs and EVNs. In addition, we cannot use VLAN tags to create tunnels in EVNs in the configuration shown in Fig. 6(b), because the end-point of a VLAN should usually be configured as a separate physical port (the untagged port).

802.1Q is currently the only technology directly supported in the OpenFlow specifications. Therefore, it is the only technology we can use to extend the tunnels in EVNs to an OFS, as shown in Fig. 6(c). Actually, the latest (at the moment of writing this paper) OpenFlow specification version 1.3 supports other technologies such as Q-in-Q, MAC-in-MAC, and MPLS (we will discuss those later). However, those were not supported by the hardware-based switches available when we designed and deployed the OpenFlow networks.

IEEE 802.1ad Provider Bridging (also called Q-in-Q) adds an extra VLAN tag to an already tagged VLAN packet. When we use Q-in-Q in the EVNs, a Q-in-Q packet is forwarded from each switch according to the most recently added VLAN tag and the destination MAC address. Q-in-Q is now supported by many low-end Ethernet switches, and therefore we can configure EVNs that accommodate OFNs at low cost.

However, when we use Q-in-Q in EVNs with the configuration shown in Fig. 6(b), a VLAN tag needs to have been added to the packet in the OFNs before it enters the EVNs, as the tag is used to identify the desired tunnel in the EVNs. In other words, the VLAN tag space should be reserved for tunnel selection in EVNs and cannot be freely utilized by the OpenFlow users (applications). This is basically because L2 switches that support Q-in-Q usually add an extra VLAN tag to a packet, according to their configuration, based on physical ports or VLAN tags.

Another issue is MAC address learning in EVNs. Although the MAC address learning domains are separated between the tunnels, an accommodation switch maintains a MAC address table for each tunnel. Therefore, when a huge number of devices are connected to the OFNs and communicate with each other through a specific single tunnel, a shortage of MAC address table entries may occur.

IEEE 802.1ah Provider Backbone Bridging (also called MAC-in-MAC) adds extra source and destination MAC addresses and a VLAN tag to an Ethernet packet, and uses them for packet forwarding in virtual networks. In other words, MAC-in-MAC can completely hide the original Ethernet header information in EVNs. Compared with Q-in-Q, MAC-in-MAC is superior as it is more tolerant to the issue of MAC address learning.

On the other hand, when we use MAC-in-MAC in the configuration shown in Fig. 6(b), it suffers from the same limitation as Q-in-Q does — packets from OFNs must be VLAN-tagged beforehand. In addition, low-end Ethernet switches rarely support MAC-in-MAC.

4.2 Wide-Area Ethernet with MPLS

The MPLS technology adds information (an MPLS label) between the L2 and L3 headers, and each packet is forwarded according to this information. Although MPLS is as mature as other L2 technologies, such as Q-in-Q and MAC-in-MAC, utilizing MPLS is an expensive approach because only the network switches/routers for core networks support it.

To configure Ethernet tunnels with MPLS, we can use Ethernet over MPLS (EoMPLS) [18] or Virtual Private LAN Service (VPLS) [19]. Because VPLS is basically for connecting more than two user networks and uses MAC address learning mechanisms, we focus on EoMPLS in this paper.

EoMPLS maintains the mapping between Ethernet VLAN tag information and MPLS label information on the boundary of the tunnels. Similar to MAC-in-MAC, EoMPLS do not refer to the MAC addresses in EVNs and it does not suffer from the MAC address learning issue. However, EoMPLS has the same VLAN tag issue in OFNs as the other technologies, i.e., it requires packets in OFNs to be VLAN-tagged.

4.3 Summary of Wide-Area Ethernet Technologies for EVNs

Table 1 summarizes the wide-area Ethernet technologies described in this paper. Basically, because we should allow users to employ the VLAN tag, it is best to configure as many physical boundary links as there are tunnels in EVNs,

[†]We describe only the minimum necessary information. For detailed information, please refer to the protocol specifications.

Table 1 Wide-area Ethernet technologies and their configurations.

Technology	Fig. 6(a)	Fig. 6(b)	Fig. 6(c)
Tagged VLAN	OK[*1][*2]	–	OK[*1][*2]
Q-in-Q	OK[*2]	OK[*1][*2]	OK[*2][*3]
MAC-in-MAC	OK	OK[*1]	OK[*3]
EoMPLS	OK	OK[*1]	OK[*3]

[*1] User cannot use the VLAN tag.
[*2] EVNs learn MAC addresses.
[*3] Not supported by the OpenFlow specification version 1.0.

and use Q-in-Q, MAC-in-MAC, or EoMPLS to configure these tunnels. When we use Q-in-Q, the issue of MAC address learning in EVNs must be considered; we need to manage the number of MAC addresses used in an experiment. Note that we will be able to manage the tunnels directly in the OpenFlow switches as shown in Fig. 6(c), because newer versions of OpenFlow specification support Q-in-Q, MAC-in-MAC, and MPLS and many venders are developing OpenFlow switches that conform those specifications.

5. Design and Deployment of RISE

In this section, we describe the design and deployment of the RISE testbed infrastructure. This is based on the considerations in Sect. 3 as well as the technical limitations of constructing real EVNs on JGN2plus and JGN-X.

5.1 Design of RISE

5.1.1 Network Technologies for EVNs

We adopted the Ethernet Q-in-Q technology when constructing EVNs for RISE for two reasons. First, we decided to use 802.1Q VLAN tags as user identifiers as we explained in Sect. 3.1, and therefore we cannot use them to create tunnels in EVNs. Second, there are some switches in JGN2plus that support neither MAC-in-MAC nor MPLS-based tunneling technologies.

Accordingly, we were naturally led to the basic architecture depicted in Fig. 6(a), i.e., we configured as many physical boundary links as the tunnels (logical links) between OFSs. On the other hand, this architecture with Q-in-Q has an issue of MAC address learning in EVNs as we summarized in Table 1. Therefore, we need to be careful about the number of MAC addresses used by the users.

5.1.2 Identifiers

We now describe the identifiers defined for managing entities in RISE, such as end-users, OFNs, and EVNs. Figure 7 shows the relationships between these entities.

First, each end-host connected to RISE is not managed by its independent ID. Instead, we allocate some dedicated physical ports to each user for their experiments, and identify all communication through those ports as belonging to that user. To achieve this, we define an in-RISE unique

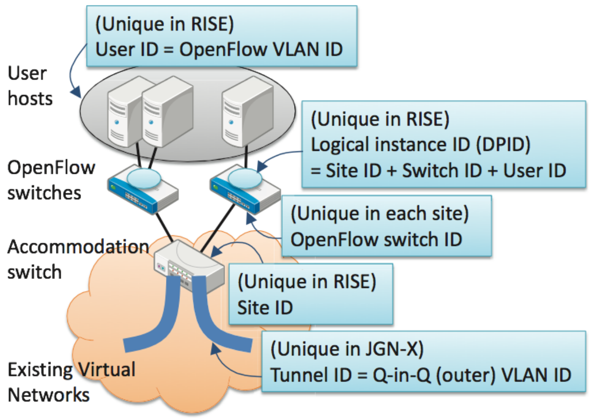


Fig. 7 Entities in RISE and their identifiers.

802.1Q VLAN ID to each user, and the OFSs attach this ID to every packet coming from the user's ports.

The physical OFSs of RISE are NEC IP8800 switches, in which we can configure multiple logical instances. Using this switch virtualization mechanism, we allocate a dedicated logical OFS instance to each user and bind the user's VLAN ID to the logical instance. To identify these logical OFS instances uniquely in RISE, we generate their DPID (OFS ID used by the OFC) by concatenating the in-RISE unique site ID, the in-site unique OFS ID, and the VLAN ID (user ID).

From the viewpoint of EVNs (JGN-X), we need Q-in-Q tunnels between the OFSs. Before configuring OFNs for RISE, we asked the JGN-X network operation group to set up these tunnels and obtained the outer VLAN IDs and port configuration information of the accommodation switches.

5.2 Basic Architecture of RISE

Next, we describe the RISE architecture. A basic set of two OFSs and one accommodation switch are installed in each site (OpenFlow access point) in JGN-X (Fig. 8). One of the two OFSs is called an edge OpenFlow switch (E-OFS) and the other is a distribution OpenFlow switch (D-OFS).

The E-OFS directly accommodates end-user connections. Therefore, all the user physical ports are allocated in those E-OFSs. An E-OFS adds a VLAN tag to packets coming from a user port, and removes a VLAN tag from packets going to a user port.

The D-OFS provides physical connections between the OFNs and the EVNs (tunnels). For this purpose, we configure OpenFlow-enabled ports and OpenFlow-disabled ports in a D-OFS and connect them with physical links, which we call *hard loops* in this paper. By selecting an output port from the OpenFlow-enabled ports, we can control the forwarding paths between the OFSs from the OFNs viewpoint. From the EVNs perspective, the boundary is the OpenFlow-disabled ports in the D-OFSs, for which we configure Q-in-Q. Naturally, the number of hard loops in a D-OFS should be at least the number of Q-in-Q tunnels between the D-OFS and the D-OFSs in other sites.

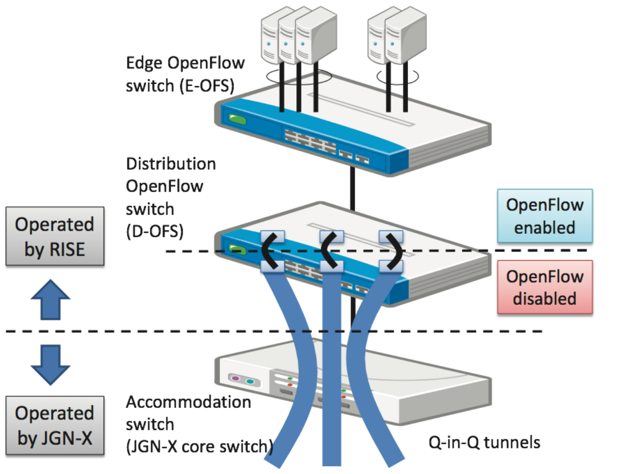

Edge OpenFlow switch (E-OFS)
Distribution OpenFlow switch (D-OFS)
Operated by RISE
Operated by JGN-X
OpenFlow enabled
OpenFlow disabled
Accommodation switch (JGN-X core switch)
Q-in-Q tunnels

Fig. 8 A basic set of two OFSs and one accommodation switch in RISE.

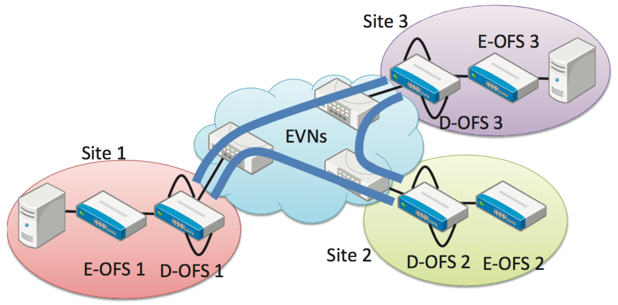


Fig. 9 Example of simple OFNs.

By using an E-OFS and a D-OFS, we can separate the design of the logical connections between the OFSs from the design of the physical links between the OFSs and EVNs. The best scenario is that we prepare many (depending on the number of physical ports the D-OFS implements) physical hard loops in the D-OFS and many (possibly full-mesh) logical tunnels in EVNs beforehand, and then flexibly modify the mapping between the hard loops and the tunnels according to user requests.

We strongly believe our architecture has the following three advantages. Firstly, the minimum number of physical connections between the accommodation switch and the D-OFS is 1[†], and therefore we can minimize the usage of physical ports in the accommodation switches, which are shared by other experimental environments deployed in JGN-X. Secondly, we can easily add OpenFlow access points to our infrastructure using the spare hard loops configured in D-OFS beforehand. Thirdly, our architecture provides flexibility in its logical configuration, meaning that we can reduce the physical modifications necessary, which naturally reduces the operating expense (OPEX) of our infrastructure.

On the other hand, our architecture has a drawback that we need two OFSs at each site. This means that the OPEX as well as the capital expenditure (CAPEX) worsens. However, OPEX of a network infrastructure generally grows underlinearly against the number of switches. As for CAPEX of the RISE infrastructure, the cost of OpenFlow switches are much lower than that of the communication lines. Therefore, the drawback of our architecture is acceptable at least for us.

5.3 Path Control Mechanisms

We now explain the mechanisms of traffic engineering using the simple network example[††] depicted in Fig. 9. In this example network, a normal TCP/IP packet from the host on the left side is transferred to the host on the right side via site 1, site 2, and site 3. More precisely, the packet traverses E-OFS 1, D-OFS 1, site 1, site 2, D-OFS 2, site 2, site 3, D-OFS 3, and E-OFS 3.

First, the left host generates a normal Ethernet packet and sends it to E-OFS 1. After E-OFS 1 receives the packet, it determines the VLAN ID, which is equal to the user ID, depending on the receiving physical port number and adds the ID to the packet. E-OFS 1 searches its flow tables using the packet header information and finds that it has to forward the packet to the hard loop connected to the tunnel between site 1 and site 2. After the packet is forwarded to the hard loop, an extra Q-in-Q tag is added to the packet header and then the packet is transferred from site 1 to site 2, where it reaches D-OFS 2 and has its Q-in-Q tag removed. D-OFS 2 then searches its flow tables using the packet header information and forwards the packet to the hard loop connected to the tunnel between site 2 and site 3. Similar to the previous in-tunnel transfer, an extra Q-in-Q tag is added to the packet, it travels through the Q-in-Q tunnel, and the Q-in-Q tag is removed at D-OFS 3. After the packet reaches an OpenFlow-enabled port of D-OFS 3, the packet is forwarded to E-OFS 3 according to its flow table information. E-OFS 3 receives the packet, and determines that it has to forward the packet to the host on the right-hand side of Fig. 9 from its flow table information. E-OFS 3 then removes the VLAN ID (user ID) from the packet and delivers it to the right host.

5.4 RISE Topology

Figure 10 shows the topology of RISE networks. The EVNs and their switches are shown in the lower part of the figure, and the OFNs and OFSs are shown in the upper part. All of the OFNs are logical (virtual) networks.

Our RISE networks can benefit from the high level of redundancy in the JGN-X physical network design (not shown in the figure). In the actual design of the logical connections between OFSs, we can reflect the physical redundancy in the logical redundancy. When we require more logical redundancy in the OFNs, we can configure some more

[†]In the current RISE infrastructure, we prepared multiple physical connections between an accommodation switch and a D-OFS because the bandwidth of each connection is just 1 Gbps which is not enough to accommodate multiple simultaneous experiments. For the same reason, we prepared multiple physical connections between a D-OFS and an E-OFS.

[††]A simple naming rule is used for this explanation, which is slightly different from that of RISE.

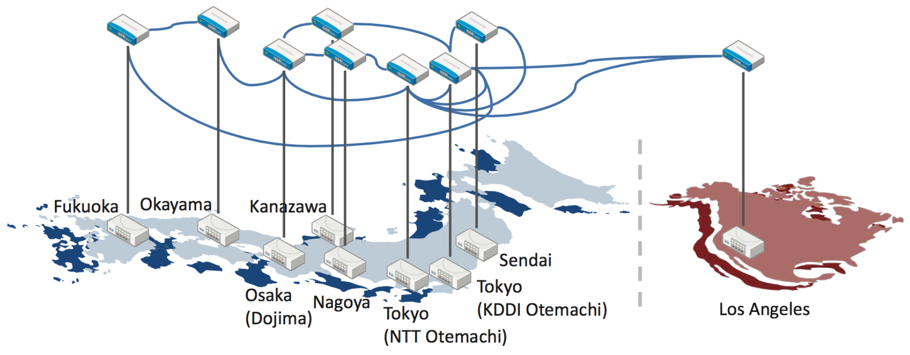


Fig. 10 RISE network topology.

redundant links using the spare hard loops.

6. Lessons Learnt from the Deployment of RISE

In this section, we describe the lessons learnt from the deployment and operation of the RISE testbed networks. From a technical viewpoint, OpenFlow is highly dependent on traditional Ethernet switch technology. However, the deployment and operation of OFNs are partially different from those of Ethernet-based networks. In addition, OpenFlow allows users to control their traffic at a low level. Therefore, we should be conservative in the operation of OFNs to prevent network troubles.

6.1 Incremental Deployment of OFNs

To achieve advanced traffic engineering in OFNs, it is highly important to prepare redundant paths in them, which allows us many options for traffic control. On the other hand, redundant paths mean that, if we make an incorrect configuration, loop packet forwarding can occur, which causes a severe traffic load in the loop paths and their switches. To avoid this kind of incorrect configuration, we recommend configuring small-scale tree-topology OpenFlow networks and validating the configuration and behavior of the networks at the outset. It is then possible to take steps towards expansion, inclusion of redundant paths, and traffic controls using these paths.

6.2 Information Sharing with Users about Their Experiments

As described above, incorrect configuration of OFC/OFSs can make the OFSs forward user packets to unexpected ports. Unfortunately, it is often difficult for OFN operators to know whether such traffic control is intended by the user or not. Sometimes, the only option for OFNs operators is a compulsory shutdown of the physical ports that have sent more packets than the pre-defined threshold. Therefore, the OFN operators should share sufficient information about the traffic engineering conducted in OFNs by the users, define the traffic threshold for port shutdown, and achieve an agreement with the users about this threshold and the possibility of compulsory port shutdown.

6.3 Separation of the Data Plane, Control Plane, and Management Plane

We should prepare separate physical ports for user-data transfer, secure channels, and remote access. As mentioned above, we sometimes need to shutdown physical ports for user-data transfer. If we share a physical port on an OFS for these purposes, its shutdown renders the OFS uncontrollable from the OFC or the remote terminal.

It is also important to protect communication networks for remote access by preparing separate VLANs for secure channels and remote access. This is because a secure channel can easily be overloaded by packets from unregistered flows, and in that case we can stop such traffic by shutting down the VLAN for the secure channel. As for remote access ports, we should not modify their configuration except in case of emergency.

6.4 Preparation for Uncontrollable OFSs

When loop traffic occurs, the OFSs on the paths become overloaded and user packet transfer, control packet transfer, and remote access on them can be almost impossible. In this case, we may try to ask the EVN operators to shutdown the tunnels between the OFSs. However, such an operation can also be difficult because the switches that create the tunnels can be overloaded. Accordingly, we should introduce a mechanism that enables us to power-off the OFSs remotely, even in the worst case.

6.5 Physical Location of the OFC

In OFNs, the OFC can be the single point of failure. A technical problem on the OFC can cause the malfunction of whole OFNs. Therefore, we should carefully choose the physical location of the OFC to be reachable by an operator in a short time, as well as preparing a spare OFC.

6.6 Operational Cooperation between OFNs and EVNs

When OFNs are deployed virtually over wide-area EVNs and are operated by different groups, their cooperation is indispensable. For example, even when an unexpected packet behavior is observed in an OFN, the OFN operators may not be able to solve it without the prompt assistance of the EVN operators. Therefore, it is important to establish a cooperative consensus for the operational procedures between OFNs and EVNs.

When OFNs are constructed over multiple wide-area EVNs connected with each other, or when multiple OFNs are connected with each other, we need more complex coordination between different infrastructures, users, operation groups, organizations, and so on. The first step to achieving this should be to clarify what traffic control mechanisms to deploy in OFNs, as we discussed in Sect. 3.1. Without these, it is highly difficult to establish appropriate coordination among the operation groups because operational procedures for trouble shooting become too wide-ranging.

7. Conclusions

In this paper, we described the motivation, design, technologies, and deployment of RISE, a wide-area OFN testbed. Actually, several demonstrations were conducted on RISE [6], [12], which showed the high potential of the wide-area deployment of OFNs.

Our future work will take the following three directions. The first is the introduction of MPLS-based tunneling technology (EoMPLS) in EVNs, because the current JGN-X infrastructure deploys new MPLS-enabled switches. We expect a lower OPEX with EoMPLS than with the current Q-in-Q as we need not worry about the MAC address learning issue.

Second, we will improve the management mechanism of user slices. The current design of RISE enables the establishment of multiple user slices with their management based on physical ports. Thus, an independent OFC can be connected to a user slice bound to separate (dedicated) physical ports. On the other hand, wide-area EVNs are physically shared among users, and we need to develop more sophisticated coordination mechanisms for logical and physical resource sharing among the users. To achieve this, we also need to establish closer cooperation between RISE and JGN-X (EVNs).

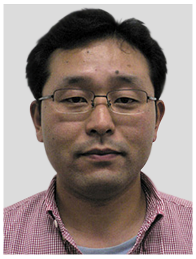
Finally, we will move forward to OpenFlow testbed interconnections. Recently, several research projects have developed wide-area OpenFlow infrastructure, such as the OS^{3}E project [20] in the United States and the OFELIA project [21] in the EU. We expect to connect RISE to these OpenFlow testbeds and develop a federation mechanism for both the data plane and the control plane among different organizations.

Acknowledgments

We would like to express our special thanks to the members of the JGN-X/JGN2plus network operation group for their support.

References

[1] Y. Kanaumi, S. Saito, E. Kawai, S. Ishii, K. Kobayashi, and S. Shimojo, “Deployment and operation of wide-area hybrid OpenFlow networks,” Proc. Fourth IEEE/IFIP International Workshop on Management of the Future Internet (ManFI 2012), Maui, Hawaii, USA, April 2012.
[2] National Institute of Information and Communications Technology, “JGN-X.” http://www.jgn.nict.go.jp/english/index.html
[3] Open Networking Foundation, “OpenFlow.” https://www.opennetworking.org/
[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: Enabling innovation in campus networks,” ACM SIGCOMM Computer Communication Review, vol.38, no.2, pp.69–74, April 2008.
[5] K.K. Yap, M. Kobayashi, D. Underhill, S. Seetharaman, P. Kazemian, and N. McKeown, “The stanford OpenRoads deployment,” Proc. Fourth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH), Beijing, China, Sept. 2009.
[6] Y. Kanaumi, S. Saito, and E. Kawai, “Deployment of a programmable network for a nation wide R&D network,” Proc. 2nd IEEE/IFIP International Workshop on Management of the Future Internet (ManFI 2010), Osaka, Japan, April 2010.
[7] National Institute of Information and Communications Technology, “JGN2plus.” http://www.jgn.nict.go.jp/jgn2plus_archive/english/index.html
[8] Y. Kanaumi, “Openflow switch demonstration at GENI conference 3rd on JGN2plus/APAN,” 27th APAN Meeting, March 2009.
[9] Y. Kanaumi, “Large-scale OpenFlow testbed in Japan,” The 31st APAN Meeting, Feb. 2011.
[10] R. Sherwood, G. Gibb, K.K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, “Can the production network be the test-bed?,” Proc. 9th USENIX Symposium on Operating Systems Design and Implementation, Vancouver, BC, Canada, Oct. 2010.
[11] “OpenFlow switch specification, version 1.0.0,” Dec. 2009. http://www.openflow.org/documents/openflow-spec-v1.0.0.pdf
[12] Y. Kanaumi, S. Saito, and E. Kawai, “Toward large-scale programmable networks: Lessons learned through the operation and management of a wide-area openflow-based network,” Proc. 6th International Conference on Network and Services Management (CNSM 2010), Niagara Falls, Canada, Japan, Oct. 2010.
[13] “IEEE 802.1Q – Virtual LANs.”
[14] R. Housley and S. Hollenbeck, “EtherIP: Tunneling Ethernet frames in IP datagrams,” RFC 3378 (Informational), Sept. 2002.
[15] “IEEE 802.1AB — Station and Media Access Control Connectivity Discovery.”
[16] “IEEE 802.1ad – Provider Bridges.”
[17] “IEEE 802.1ah – Provider Backbone Bridges.”
[18] L. Martini, E. Rosen, N. El-Aawar, and G. Heron, “Encapsulation methods for transport of Ethernet over MPLS networks,” RFC 4448 (Proposed Standard), April 2006. Updated by RFC 5462.
[19] M. Lasserre and V. Kompella, “Virtual private LAN service (VPLS) using label distribution protocol (LDP) signaling,” FC 4762 (Proposed Standard), Jan. 2007.
[20] “Open Science, Scholarship and Service Exchange (OS^{3}E).”
[21] “OpenFlow in Europe — Linking Infrastructure and Applications.”



Yoshihiko Kanaumi received an M.E. degree from the Graduate School of Engineering Science, Osaka Prefecture University, Osaka, Japan, in 1998 and is currently a Ph.D. candidate in the Graduate School of Engineering, University of Tokyo, Japan. He joined NEC Corporation in 1998 and now works in IP network division. He has been engaged in the development of hardware architectures for routers and of SDH/SONET, ATM, and IP networks. He has also been involved in research at the National Institution of Information and Communication Technology (NICT) on the operation and management of the Future Internet, including the OpenFlow controller and SDN infrastructure. He has been a member of WIDE Project, Cyber Kansai Project, and also Network Operation Center of NICT JGN-X. He was an NOC (Network Operation Center) member of the Interop Tokyo 2003 to 2010.

Shu-ichi Saito received an M.E. degree from the Graduate School of Engineering Science, Iwate University, Iwate, Japan, in 2003. He joined NEC Corporation in 2003 and now works in IP Network Division. He has been engaged in the development of hardware architectures for routers and OpenFlow Swiches. He has also been involved in research at the National Institution of Information and Communication Technology on the operation and management of the Future Internet, including the SDN/OpenFlow Network. He has been a member of Network Operation Center of NICT JGN-X. He was an NOC (Network Operation Center) team member of the Interop Tokyo from 2010.

Eiji Kawai received Ph.D. in information systems from Nara Institute of Science and Technology (NAIST) in 2001. From 2000 to 2003, he was an awarded researcher with Japan Science and Technology Corporation (JST). From 2003 to 2009, he worked for graduate school of information science, NAIST as assistant professor and associate professor. In 2009, He joined National Institute of Information and Communications Technology (NICT), and now he is director of the network testbed research and development laboratory.

Shuji Ishii received an M.E. degree from the Graduate School of Computer Science, University of Electro-Communications, Tokyo, Japan, in 1992. He joined NEC Corporation in 1995 and now works in Cloud System Research Laboratories. He has been engaged in the development of the IPv6 (IPSec) protocol stack for routers and hosts at NEC as well as research on software architectures for the OpenFlow controller.

Kazumasa Kobayashi received his B.S. in Mathematics from the Okayama University of Science, Okayama, Japan. From 1988 to 1993, He worked for Digital Equipment Corporation Japan, Educational division. He received his M.E. and D.E. degrees in computer science from Nara Institute of Science and Technologies, Nara, Japan, in 1995 and 2000, respectively. From 1999 to 2003 he was an Assistant Professor in Kurashiki University of Science and the Arts, Okayama, Japan. He has been a member of WIDE Project, Cyber Kansai Project, Okayama Information Highway project, and also a director of Network Operation Center for NICT JGN-X network. His research interests include technologies for multimedia communication over high speed network, network inter-operability, network management, network security for the Internet and Next Generation Internet for wide area distirbuted computing environment. He was an NOC (Network Operation Center) member of the Interop Tokyo 1995 to 2012.

Shinji Shimojo received the M.E. and Ph.D. degrees from Osaka University in 1983 and 1986, respectively. He was an Assistant Professor with the Department of Information and Computer Sciences, Faculty of Engineering Science at Osaka University from 1986, and an Associate Professor with Computation Center from 1991 to 1998. During this period, he also worked for a year as a Visiting Researcher at the University of California, Irvine. He has been a Professor with the Cybermedia Center (then the Computation Center) at Osaka University since 1998, and from 2005 to 2008 had been the director of the Center. He is an executive researcher at National Institute of Information and Communications Technology and a director of Network Testbed Research and Development Promotion Center Network Testbed Research and Development Promotion Center. His current research work is focusing on a wide variety of multimedia applications, peer-to-peer communication networks, ubiquitous network systems, and Grid technologies. He was awarded the Osaka Science Prize in 2005. He is a member of IEEE and IPSJ fellow.