Proceedings of APCC2015 copyright © 2015 IEICE 14 SB 0087

VNode Infrastucture Enhancement – Deeply Programmable Network Virtualization

Kazuhisa Yamada
NTT Network Innovation Laboratories
Yamada.kazuhisa@lab.ntt.co.jp

Akihiro Nakao
University of Tokyo
nakao@nakao-lab.org

Yasusi Kanada
Hitachi Corporation
Yasusi.Kanada.yq@hitachi.com

Yoshinori Saida
NEC Corporation
y-saida@cp.jp.nec.com

Koichiro Amemiya
FUJITSU LIMITED
amemiya.kouichi@jp.fujitsu.com

***Abstract*— We introduce the latest extended functions for the VNode infrastructure. We present new extended VNode infrastructure functions that achieve high performance and provide convenient deep programmability to network developers. In addition, we extend network virtualization from the core network to edge networks and terminals. We deploy an enhanced VNode infrastructure on the JGN-X testbed in evaluation experiments. We also succeeded to create international federation slice between GENI and Fed4FIRE.**

Keywords*— *Network Virtualization, Deep Programmability, Resource Abstraction, Resource Isolation, Testbed, International Federation

I. Introduction

The current communications infrastructure represented by the Internet has flourished as a foundation based on the Internet protocol (IP). However, there are limits to the Internet and to address all the various problems facing the current communications infrastructure, an information and communications base that has an innovative design must be constructed [1]. Network virtualization has recently attracted attention and is one technology that can help achieve this innovative communications infrastructure. We are promoting advanced network virtualization technology with the aim of constructing an information and communications base that incorporates innovative design thought. Considering a wide viewpoint, a communication infrastructure consists of “links” that provide network resources for transmitting data and “nodes” that provide computing resources and storage resources to execute programs for interpreting protocols and processing data. Advanced network virtualization technology virtualizes whole networks based on this wide viewpoint and contributes to network users. Specifically, advanced network virtualization technology meets five requirements: resource abstraction, resource isolation, resource elasticity, deep programmability, and authentication. This technology enables the creation and design of new generation networks from a clean slate to inspire free-innovative thinking. In addition, advanced network virtualization technology provides a new generation information and communications infrastructure that can contain multiple networks at the same time. To achieve advanced network virtualization technology, we designed and developed a network virtualization infrastructure called the VNode infrastructure [2]-[4].

In this paper, we introduce the latest extended functions for the VNode infrastructure. We present new functions that achieve high performance and provide convenient deep programmability to network developers. In addition, we extend network virtualization from the core network to edge networks and terminals. We deploy an enhanced VNode infrastructure on the JGN-X [5] testbed in evaluation experiments. We also successfully created international federation slice between GENI [6] in the US and Fed4FIRE [7] in the EU.

II. VNode Infrastructure

In this section, we briefly describe the conventional VNode infrastructure. Fig. 1 shows the basic construction of the VNode infrastructure. The VNode infrastructure provides programmability on both the control/management-plane and data-plane by establishing slices. We can configure our own topology and deploy our own software functions on a slice. The VNode infrastructure consists of multiple VNodes, access gateways (AGWs), and a network virtualization management system (NVMS). We describe the VNode, AGW, NVMS, and a slice in the following sections.

A. VNode

A VNode is the main component of the VNode infrastructure. A VNode is an integrated node unit comprising of the Programmer and Redirector. VNodes provide virtualized resources such as a slice according to the slice design specifications.

1) Programmer: We construct a VNode in two parts, Programmer and Redirector, because the technologies for providing virtualized computing resources and link resources may advance at different speeds. The Programmer provides various processing components such as general-purpose servers, network processors, and OpenFlow switches to yield various combinations of virtualized computing resources. The Programmer provides virtual machines (VMs) as slice nodes and they work as processing components. We can deploy various software functions on a VM to enhance service functionalities.

2) Redirector: The Redirector provides virtualized link resources such as bandwidth and buffer size as virtual links to connect slice nodes. We achieve the desired Quality of Service (QoS) by instructing the Redirector to send traffic data with particular characteristics such as a streaming service.

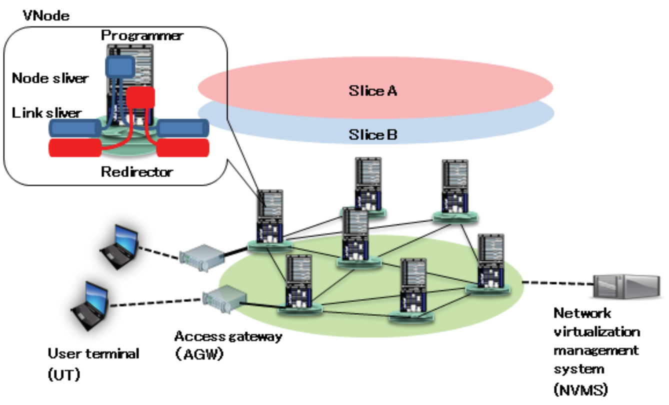

Fig. 1. VNode infrastructure

B. *AGW*

An AGW is a border node between the end-user access side and the VNode infrastructure domain. An AGW authenticates user terminals (UTs) and accommodates UTs into the slice that they are authorized to access.

C. *NVMS*

The NVMS manages and controls whole VNode resources. The NVMS provides a portal site to slice developers/owners as well as authentication, authorization, and accounting (AAA) functions. Slice developers/owners can reserve resources via the portal with operational security. Slice developers can create slices via a GUI on a web browser.

D. *Slice*

A slice is a set of connected computation resources and link resources. These resources are virtualized and isolated. Therefore we can configure our own network topology and deploy our own software functions into network nodes, i.e., the VNode infrastructure provides programmability.

1) Node Sliver: A node sliver is a set of virtualized node resources such as CPU time, memory, and storage. We can assign these resources to nodes of a slice and install an operating system such as Linux on the nodes by writing a slice design specification that respects the limitations of the physical resources. Node slivers work as various processing components and are mapped into the Programmer of the VNode.

2) Link Sliver: A link sliver is a set of link resources such as the bandwidth, burst size for QoS control, and queuing delay. The slice design specification determines how resources are assigned to the links of a slice. Link slivers are mapped into the Redirector of the VNode.

III. EXTENDED VNODE INFRASTRUCTURE

In this section, we present new VNode infrastructure functions that achieve a high performance and convenient deep programmability to network developers. In addition, we extend network virtualization from the core network to edge networks, terminals, and other network virtualization infrastructures. The specific functions are described below.

A. *Edge Network Viruralization*

The virtualization technology that includes edge networks to terminals as well as a core network is necessary to achieve the concept of a slice in the whole infrastructure. Generally, the requirements for an edge network are different from the core network. In network virtualization, sufficient resources and a highly precise band guarantee are required so that many slices can share the resources in the core network. On the other hand, various types of terminals must be contained in a scalable slice and virtualization technology must be easily introduced in an edge network. Therefore, we decided to constitute the infrastructure using the core network and edge networks.

We developed FLARE, a light-weight and low-power consumption network virtualization node for edge networks. Fig. 2 shows FLARE node architecture. The hardware of the FLARE node connects many core processors that execute packet processing on the data-plane and x86 architecture Intel processors that control packet processing on the control-plane in a PCIe interface. Using a VM constructed using the lightweight Linux container (LXC) technique, virtualization is executed on both types of processors, and an outside FLARE node management server creates a slice in the FLARE node. Since the slice is created on a many core processor, isolating the processing between the slices is secured by assigning a core to each slice. In addition, a slicer-slice, which is a special slice for allocating packets in a slice based on tag information that is added to the I/O port or a packet, is established in the FLARE node. Using the slicer, individual network processing is enabled in each slice. Fig. 2 shows that an Ethernet switch is established on slice 1, a switch for a packet in which the MAC address length is extended is established on slice 2, and OpenFlow switch is established on programmable slice n.

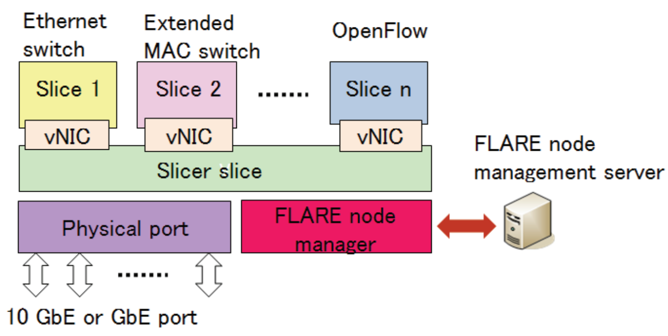

Fig. 2. FLARE node

B. *Extended Resource Abstraction and Elasticity*

The VNode infrastructure achieves flexible and integrated resource management by abstraction and virtualization of physical resources. We extend resource abstraction to the transport network and achieve integrated control and management of slices and the transport network.

The network infrastructure that provides general network services consists of a "service network" that consists of service nodes to provide network service and a "transport network" that connects and transmits a packet between service nodes. The transport network transfers a packet independent of the service contents and is used in common by multiple services. The VNode infrastructure also consists of the service network which comprises VNodes and the transport network which comprises existing switches and routers. Therefore, integrated control and management of the service and transport networks is necessary in the VNode infrastructure. Fig. 3 shows an integrated control and management system for the VNode infrastructure. We designed a service network controller (SNC) that controls and manages the service network and a transport network controller (TNC) that controls and manages the transport network as the NVMS. The TNC automatically allocates physical resources for a transport path that is necessary to establish a link slivers at slice creation in cooperation with the SNC. Transport network resources are managed as a transport path in the SNC, and SNC notifies TNC of the required number of resource such as the bandwidth to create a slice. The TNC allocates the required transport path with the prepared physical resources and provides it to SNC.

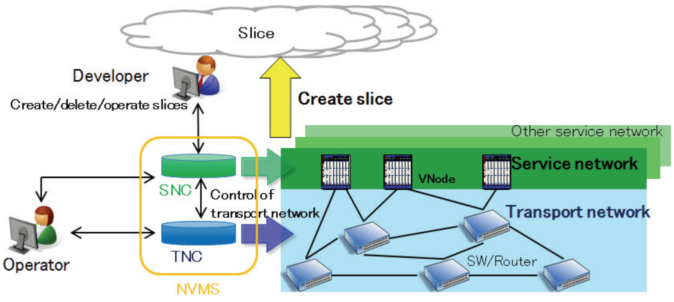

Fig. 3. Service network and transport network control

Due to resource elasticity, we can dynamically change the resource allocation to a slice in the VNode infrastructure and can maintain service quality and resource optimization. To achieve resource optimization, the performance of each slice in the VNode infrastructure must be measured. So we developed the Network Monitoring Manager (NMM) for the VNode infrastructure. Fig. 4 shows slice measurement based on cooperation between the NVMS and NMM. The Developer who is an owner of the slice appoints the slice ID for the measurement, the measurement point in a slice, and the date and time for reservation of the measurement from the NVMS Portal screen. The NVMS offers measurement link information and slice identification information to the NMM. Then, the NMM configures optical switches to change the measurement point and sets the filter in the existing network monitoring equipment, PRESTA 10G. The developer is notified of the measurement results through the NVMS from the NMM.

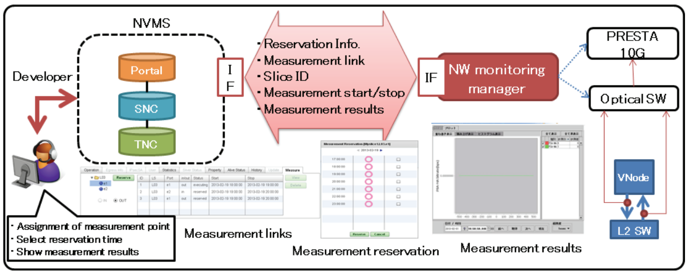

Fig. 4. Slice measurement based on cooperation between NVMS and NMM

C. *Precise Resource Isolation*

Preventing resource interference between slices is important in network virtualization. For example, if a certain slice monopolizes bandwidth, it must not produce an unfair delay in other slices. A function to avoid such resource interference is resource isolation. A basic resource isolation function was achieved in the Redirector in an early version of the VNode infrastructure. A summary of the basic function of resource isolation in the Redirector is given hereafter. The Redirector incorporates an L3 switch with a VLAN function, IP routing function, and QoS functions, which are Weighted Fair Queuing (WFQ) and policing. WFQ could achieve high-performance bandwidth control for each slice, but it was available to only a limited number of slices conventionally because it was relatively expensive to use high-speed memory.

Two problems of the basic function mentioned above were solved in the latest version of the Redirector. The first problem is to provide resource isolation based on WFQ and to achieve more precise resource isolation control for each link sliver. We added new hardware called hierarchical shaping on a built-in Redirector switch to address this problem and provided a precision resource isolation function that is scalable from 100 kbps to over a gigabit per second using this function for each sliver. The second problem is to achieve resource isolation throughout the VNode infrastructure including the Programmer, AGW, and transport network. We achieved cooperation in the resource management between the Programmer, Redirector, and AGW to address this problem. Specifically, the SNC supplements specifications for the isolation of the link slivers listed in a slice design and orders resource isolation for the Programmer and AGW. The Programmer and AGW achieve the appointed resource isolation, so resource isolation is achieved throughout the VNode infrastructure. Fig. 5 shows resource isolation between the Programmer and Redirector. The Slice developer only defines the link sliver (LS01) bandwidth for the resource isolation through the slice design file. The SNC automatically supplements the bandwidth of vport (vp1) on node sliver (NS00). The Programmer sets the bandwidth of vp1 in accordance with the supplemented slice design file. Then resource isolation between the Programmer and Redirector is achieved.

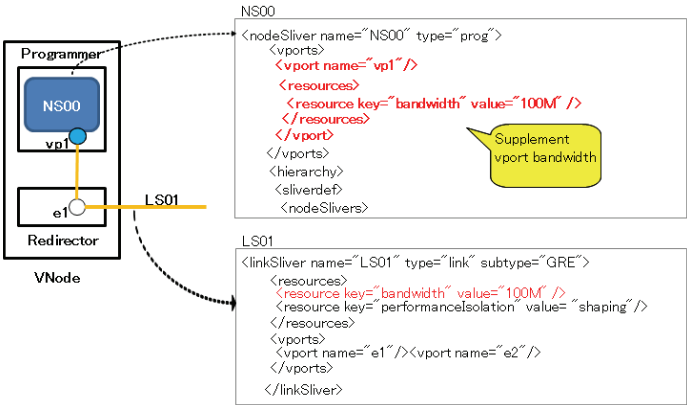

Fig. 5. Resource isolation between Programmer and Redirector

D. Coexistence Performance and Programmability

The Programmer, which is one of the components composing the VNode, enables programmability in the network processing and performance for high speed packet transfer to coexist. To achieve this, we actualized various network functions in the programs by preparing two kinds of mechanisms: one is a slow path to provide a flexible programing environment using a VM on general-purpose servers and the other is a fast path to provide a programing environment for a network processor to transfer packets at high speed. Furthermore, in regard to the slow path, we address a problem in achieving high computing performance to be caused by the gap in the network I/O performance.

Fig. 6 shows the Programmer architecture. We used an OpenFlow switch to connect the computer resources in the device. We can build a network node that combines various computer resources freely. The packet converter converts the packet format (MAC-in MAC to VLAN format) between that for the Redirector and the Programmer. To achieve coexistence performance and programmability, we applied a hardware off-load technique for the slow path. In addition, we achieved a network I/O performance gain by applying hardware of 10 GbE. Fig. 7 shows the performance evaluation results for the slow path. Offload represents results after employing the mechanisms and onload represents results before the mechanisms are employed. The SUM of the CPU Load is the total of the CPU utilization of the guest OS, and HOST is only for the host OS. The figure shows that the throughput performance improves and that the CPU utilization of the host OS decreases.

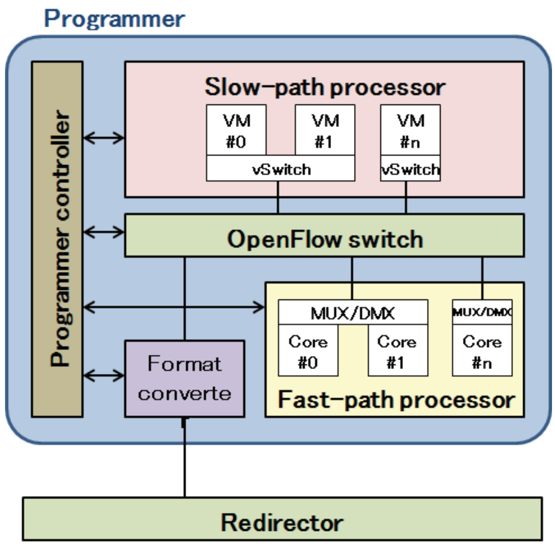

Fig. 6. Programmer architecture

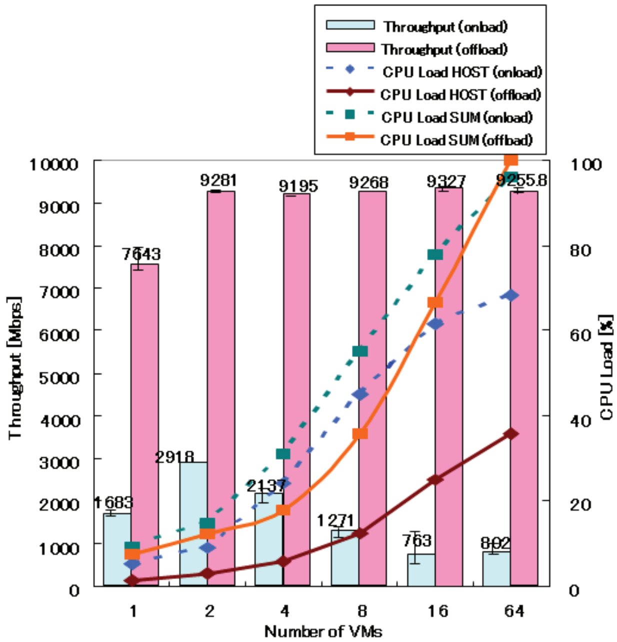

Fig. 7. Performance evaluation results for slow path

E. Enhancement of AGW

The AGW is gateway equipment for slices in the VNode infrastructure. The gateway equipment is deployed at the edges of the VNode infrastructure and provides connectivity between slices and physical devices or networks using various protocols including proprietary ones. The enhanced functions of the gateway are described hereafter.

The first enhanced function is connectivity. The gateway identifies the users and connects user devices or networks with slices using various protocols. We provide VLAN accommodation in addition to the conventional IPSEC accommodation.

The second enhanced function is security. The gateway authenticates the users and their corresponding packets using the IKE/IPSec protocol when users connect their terminals to the slices through the gateway.

The third enhanced function is programmability. The latest version of the AGW provides programmable virtual nodes (Node Sliver VM) on programmable virtual node blocks and enables execution of network/data processing applications at the edge of the slices.

The fourth enhanced function is a customizable protocol stack on the gateway block. With this functionality we can configure the gateway interface in accordance with the protocols that are used in the user terminal or network.

Additionally, the latest AGW provides higher performance for frame transfer even on the commodity Intel architecture (IA) severs (Table I). Higher performance is achieved through the Packet Processing Middleware (PPM) functionalities, Zero Copy I/O, and parallel processing framework utilizing multiple/multi-core processors.

Fig. 8 shows the internal architecture and the communication path in the gateway equipment. The gateway equipment is required to have a small footprint and to be economical rather than high performance compared to large footprint equipment. This is because more gateways than VNodes are deployed in the VNode infrastructure and the gateway is supposed to be widely distributed at the edge of the VNode infrastructure including near or at the user locations.

In order to satisfy these requirements, we designed the gateway with the following policies. First, all functionalities of the gateway, the programmable virtual node, and management functionalities, should be provided on a single commodity IA server. Second, the number of programmable virtual nodes that is provided by this gateway should be scalable as needed by adding IA servers on which the functionality of the programmable virtual node provider is deployed.

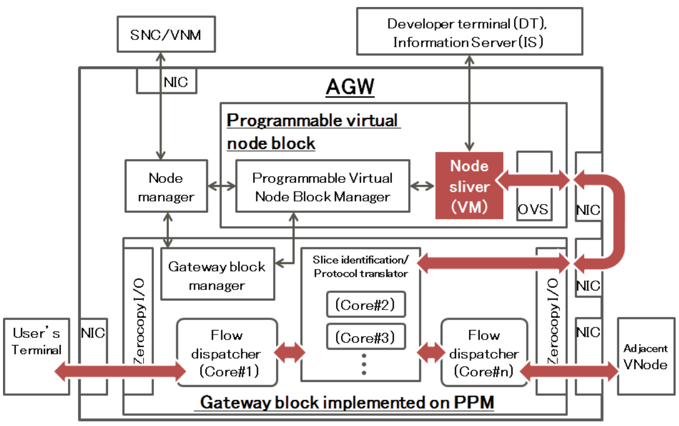

Fig.8. Gateway architecture

TABLE I. GATEWAY PERFORMANCE

Throughput with IPSec	Throughput with VLAN	Slow Path Node Sliver Throughput
1.3 Gbps*1	4.7 Gbps*1	1.5 Gbps*2

*1. 1372 bytes frame, Intel Xeon X5690 (3.46 GHz/6 cores) x 2

*2. Using Intel Xeon X5690 (3.46 GHz/6 cores) and allocate vCPUx2 and 2 GB mem. for VM, vCPUx4 for vhostnet

IV. DEPLOYING VNODE ON TESTBED

We deployed the latest version of VNode infrastructure on the JGN-X testbed. There are 7 VNodes and 6 AGWs on the testbed, and approximately 40 slices are used in the evaluation experiments as shown in Fig. 9.

In addition, we installed one small type VNode at the University of Utah and conducted an international federation with ProtoGENI, which is a network virtualization testbed of the GENI project in the US. We also conducted with Fed4FIRE, which is a European network virtualization testbed and demonstrated slice construction in a global multi-domain environment. Fig. 10 shows the construction of the international federation experiment. Each network virtualization testbed is federated through a Slice Exchange Point (SEP) which is established within a small type VNode at the University of Utah. We successfully created a large scale international federation slice for the evaluation experiments.

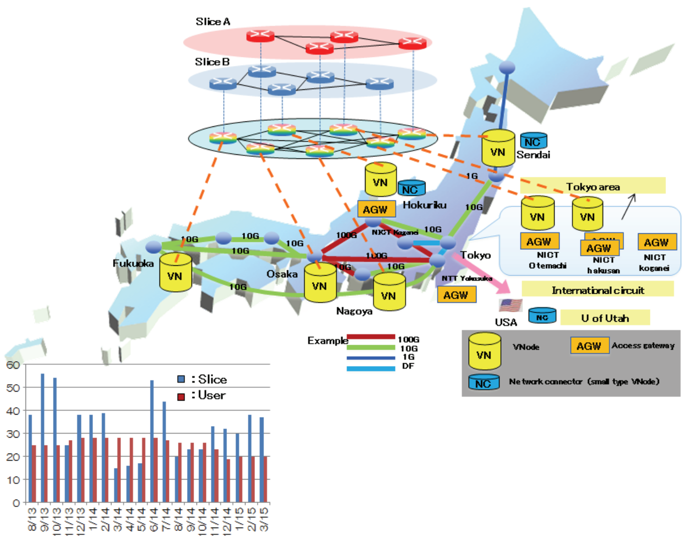

Fig.9. Deployment of VNode on JGN-X testbed

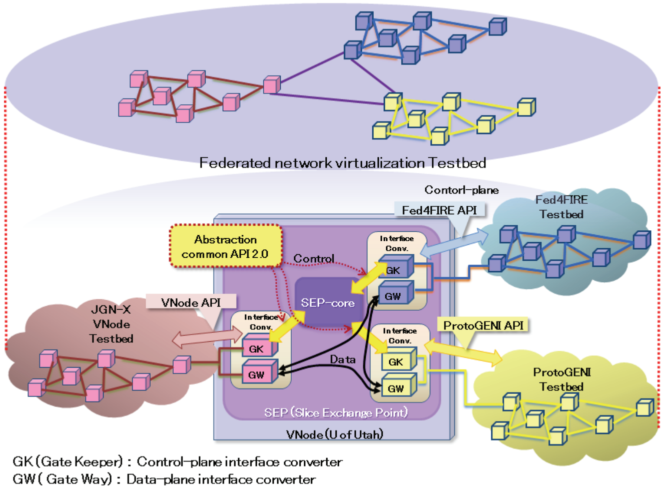

Fig.10. International federation experiment among Japan, US, and EU

V. Conclusion

In this paper, we introduced the latest extended functions of the VNode infrastructure. We provided new functions to achieve high performance and convenient deep programmability to network developers. We extended resource abstraction to the transport network. We achieved precise resource isolation for the VNode infrastructure. We also achieved coexistence performance and programmability and enhanced AGW functions. In addition, we extended network virtualization from the core network to edge networks and terminals. We deployed an enhanced VNode infrastructure on the JGN-X testbed in evaluation experiments. We also successfully created an international federation slice between GENI and Fed4FIRE.

Acknowledgment

This research was commissioned by the National Institute of Information and Communications Technology (NICT). The authors thank the VNode project members.

References

[1] "AKARI" Architecture Design project. [Online] Available: http://www.nict.go.jp/en/photonic_nw/archi/akari/akari-top_e.html

[2] A. Nakao, “VNode: A deeply programmable network testbed through network virtualization,” *3rd IEICE Technical Committee on Network Virtualization*, 2012.

[3] Y. Kanada, K. Shiraishi, and A. Nakao, “Network-resource isolation for virtualization nodes,” *IEICE Transactions on Communications*, vol. 96, no. 1, pp. 20–30, 2013.

[4] Y. Katayama, et al, “Design and implementation of network virtualization management system,” *IEICE Transactions on Communications*, vol. 97, no. 11, pp. 2286–2301, 2014.

[5] New Generation Network Testbed JGN-X. [Online]. Available: http://www.jgn.nict.go.jp/english/index.html.

[6] Global Environment for Network Innovations (GENI) Project. [Online]. Available: http://www.geni.net.

[7] Federation for Future Internet Research and Experimentation (Fed4FIRE). [Online]. Available: http://www.fed4fire.eu.