High-performance Network Accommodation and Intra-slice Switching Using a Type of Virtualization Node

Yasusi Kanada
Central Research Laboratory
Hitachi, Ltd.
Yokohama, Japan
Yasusi.Kanada.yq@hitachi.com

Kei Shiraishi
Telecommunications & Network
Systems Division, Hitachi, Ltd.
Kawasaki, Japan
shiraishi_kei@itg.hitachi.co.jp

Akihiro Nakao
Graduate School of Interdisciplinary
Information Studies, The University of Tokyo
Tokyo, Japan
nakao@iii.u-tokyo.ac.jp

***Abstract*—The architecture for programmable network-virtualization platforms, i.e., the VNode architecture, has been developed in a project called the Virtualization Node Project. This paper introduces a type of physical node called Network ACcommodation Equipment (NACE) to the VNode architecture. NACE has dual roles in this architecture. The first role is as a *network-slice gateway* between an external network (Ethernet/VLAN) and a slice (virtual network). NACE can accommodate a data center or another testbed in a slice with high-performance (up to 10 Gbps) data-format conversion. The second role is as a special type of virtualization node that implements *intra-slice virtual switch* by using Ethernet hardware, which can replace software-based switching using a VM or a network processor. These roles are modeled as a node sliver (virtual node) with a gateway function and a node sliver with a switching function (i.e., a switch node-sliver), and these node slivers are specified by using XML. These functions were evaluated by using two testbeds, and the evaluation results confirm that both functions work correctly and perform well in terms of delay and packet loss.**

Keywords*—network virtualization; virtual network; network accommodation; network-slice gateway; virtual switch; intra-slice switching

I. INTRODUCTION

In Japan, several projects targeting new-generation networks (NwGN) have been conducted [Aoy 09] [AKA 10]. These projects aim to develop new network protocols and architectures (i.e., the "clean slate" approach [Fel 07]) and to develop various applications that are difficult to run on IPs but work well on NwGNs. In one of these projects, "Virtualization Node Project" (VNP), Nakao, et al. [Nak 10] has developed a virtualization-platform architecture [ITU 12] called VNode architecture.

There have been two issues concerning the original implementation of VNode architecture [Nak 12a][Nak 12b]. The first issue is that the original implementation does not have a high-performance gateway between a slice and an external network. This architecture has a type of gateway called an access gateway (AGW) to accommodate end users' computers in slices. However, in an AGW, because the data rate between a user and an AGW is assumed to be 1 Gbps or less, and the security of this link should be guaranteed by IPSec, the performance of the AGW is not sufficient for accommodating a data center or high-performance testbed in a slice. In addition, because an AGW is optimized to accommodate user terminals, it is not the best means for accommodating a network with many hosts.

The second issue is that this implementation does not fully utilize the performance of the hardware component of a VNode, especially the high-performance Ethernet switching function. A VNode implements node slivers (virtual nodes) using Linux virtual machines (VMs) or network processors (NPs). The node slivers can be freely programmed, so any protocol (either IP/Ethernet based or non-IP/non-Ethernet based) can be implemented. However, the performance of the node slivers is not optimum. Although a VNode contains a high-performance L3 switch with 10-Gbps Ethernet interfaces and 190-Gbps (or more) switching capacity, it has been hard to achieve multi-gigabit intra-slice switching speed by using VMs and/or NPs.

To address these issues, a type of physical node called Network ACcommodation Equipment (NACE or NC) has been developed. NACE has dual roles. The first role is as a high-performance gateway between a slice and an external network. By means of this gateway function, NACE can accommodate a data center or another testbed in a slice with high-performance (up to 10 Gbps) data-format conversion. By using NACE, external networks can also provide services to slices, and slices can provide services to external networks. The second role is as a special type of virtualization node that implements an intra-slice switching function using Ethernet hardware. This function makes it possible to switch not only Ethernet packets but also packets of arbitrary format with arbitrary type and size of addresses that can be mapped to Ethernet MAC addresses.

The rest of this paper is organized as follows. Section II summarizes the virtualization platform and the slice model developed in VNP. Section III outlines NACE. Sections IV and V describe two roles of NACE, i.e., network-slice gateway and intra-slice switch, respectively. Section VI describes potential usage of NACE, including a data-center or testbed gateway, an intra-slice Ethernet switch, and an intra-domain slice interaction. Section VII evaluates NACE by using two slices on test beds, and Section VIII concludes this paper.

II. VIRTUALIZATION PLATFORM AND SLICE MODEL

This section explains network virtualization, the structure of virtualization platform (i.e., physical network), and the structure of virtual network.

A. Network Virtualization

When many users and systems share a limited amount of resources on computers or networks, virtualization technology creates an illusion that each user or system owns resources of their own. Virtualization technology was initially developed as virtualization of computer memory

 ISBN: 978-1-61208-226-4

and multiplexed (time-sharing) use of computational resources such as CPU time. However, recently, a whole computer can be virtualized as a VM.

Concerning networks, wide-area networks (WANs) are virtualized by using virtual private networks (VPNs). When VPNs are used, a physical network can be shared by multiple organizations, and these organizations can securely and conveniently use VPNs in the same way as virtual leased lines. Nowadays, networks in data centers are virtualized by using VLANs, while servers are virtualized by using VMs.

Many programmable virtualization-network research projects have been carried out, and many models, including PlanetLab [Tur 07], Virtual Network Infrastructure (VINI) [Bav 06], Global Environment for Network Innovations (GENI) [GEN 09], and Genesis [Kou 01], have been proposed. Slices are created by network virtualization using a *virtualization platform* that operates the slices.

In VNP, Nakao et al. [Nak 10][Nak 12b] developed network-virtualization technology that makes it possible to build programmable virtual-network environments in which slices are isolated logically, securely, and in terms of performance (QoS) from one another [Kan 12b]. In these environments, new-generation network protocols can be developed without disrupting other slices.

B. Structure of Virtualization Platform

A virtualization-platform domain is managed by a domain controller (DC) and has two types of nodes (**Figure 1**).

- *VNode* (virtualization node) is a physical network node that forwards packets on the platform. Each packet in the platform contains a virtual packet in a slice (as the payload).
- *Gateway*, of which access gateway (AGW) is one type, is a network node that forwards packets from the platform to user terminals (PCs) or another network, or vice versa.

A domain may contain conventional routers or switches that do not have virtualization functions. VNodes are connected by tunnels using a protocol such as Generic Routing Encapsulation (GRE) [Far 00]. A virtual network with free topology, which is not constrained by the topology of the physical network and does not depend on the specific functions of the nodes in between, can therefore be created. A VNode can operate as a router or a switch for platform packets, so it can be deployed in conventional networks.

Each VNode consists of the following three components.

- *Programmer* processes packets on the slices. Slice developers can inject programs into programmers.

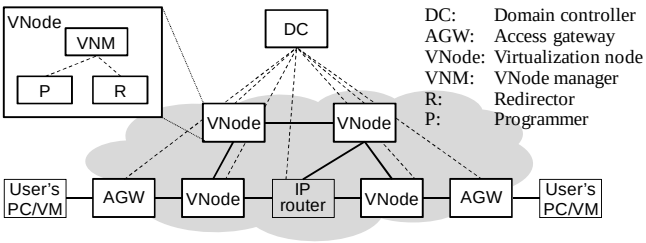


Figure 1. Physical structure of virtualization platform

- *Redirector* forwards (redirects) packets from another VNode to a programmer or from a programmer to another VNode.
- *VNode manager* (VNM) is a software component that manages the VNode according to instructions from the DC.

C. Structure of Virtual Network

In the virtual-network model developed by VNP, a virtual network (or a collection of resources in a virtual network) is called a *slice*, which consists of the following two types of components (**Figure 2**) [Nak 10][Nak 12b].

- *Node sliver* (a virtual-node resource) represents computational resources that exist in a VNode (in a programmer). It is used for node control or protocol processing of arbitrary-format packets, and it is generated by slicing physical computational resources.
- *Link sliver* (a virtual-link resource) represents resources of a (layer-2) virtual link that connects two node slivers. In the VNode architecture, any IP or non-IP protocols can be used on link slivers. A link sliver is mapped on a physical link between two VNodes or a VNode and a gateway, and it is generated by slicing physical network resources such as bandwidth.

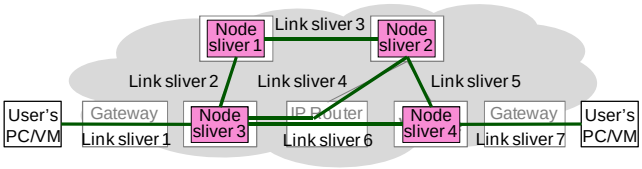


Figure 2. Example of slice design

The DC of a domain receives a slice design given by an XML-based *slice definition*. It then distributes the slice definition to each VNM, which sends necessary sliver definitions to the programmer and the redirector: the programmer receives information required for node-sliver configuration, and the redirector receives the information required for configuring link slivers.

III. OUTLINE OF NACE

This section outlines NACE; namely, describes roles, requirements, and the structure of NACE.

A. Roles of NACE

NACE has two roles. The first role is as a gateway between a slice and an external network. NACE can accommodate an Ethernet-based network or a bundle of VLANs in a slice. Thanks to network-slice gateway function, NACE can accommodate a data center or another testbed in a slice with high-performance (up to 10 Gbps) data-format conversion. This function has been implemented as an extended function of a node sliver.

The second role is as a special type of VNode that implements an intra-slice virtual switching function for packets of arbitrary format by using Ethernet hardware. This function is intended to make it possible to switch packets of arbitrary format. However, in the current version of NACE, only the Ethernet protocol can be used in slices. This function has been implemented as a special type of node

 ISBN: 978-1-61208-226-4

sliver called a switch node-sliver.

These two roles (described in detail in Sections IV and V) have been given to the same equipment because both require similar hardware and software components, including data converter between VLANs and GRE/IPs (GRE link sliver) and management software.

B. Requirements

As described above, the two roles require gateway and switching functions, which share two common requirements. One requirement is that methods for modeling and specifying these functions must be developed. As parts of the VNode architecture, these functions must be modeled as a combination of node slivers and link slivers in the slice definition and specified by using XML. NACE must implement these functions using VLAN functions of an Ethernet switch. However, this implementation is out of scope of this paper.

The other requirement is that high-performance data-conversion functions, which enable 10-Gbps wire-rate accommodation and switching, must be developed. The data conversion is required because both network-slice gateways and intra-slice virtual switches handle two different data formats. The former must convert packets from the virtualization-platform format (GRE/IP) to the external format (VLAN), and vice versa. The latter must convert packets that can be switched by the hardware from the platform format to a VLAN format, and vice versa. In addition, because both functions are modeled using the same XML-based language and managed in the same way by the management system, the required software functions are also similar. Therefore, although the two roles are very different, the same components can be used for both roles.

C. Structure of NACE

The above requirements are satisfied by NACE. The current version of NACE is a remodeled version of the VNode (**Figure 3**). Similar to a normal VNode described in Section II, a NACE consists of three components. Two of them are almost the same as those in a normal VNode, i.e., a *redirector* and a *VNM*. However, the other component, a programmer, is replaced by a *pseudo programmer manager* (PPM). The redirector in the NACE consists of the three components: Ethernet switch, node manager, and service module cards.

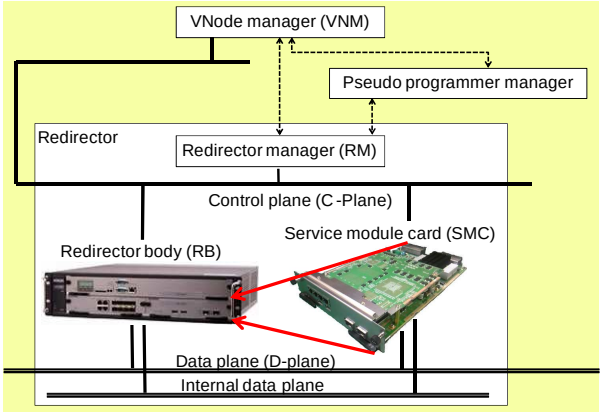


Figure 3. Structure of NACE

- *Redirector body* (RB) is a layer-3 (L3) switch, i.e., an IP/Ethernet node; it has both VLAN and IP-routing functions. A carrier-class high-end switch is used to build a VNode. To make the NACE physically smaller and portable, a 2.5U-size L3 switch is used for the RB, which is much smaller than the switch used in a normal VNode.
- *Redirector manager* (RM) is a software component in a Linux-based server. RM has link-sliver management functions; that is, it creates, modifies, and deletes link slivers. It also has a range of equipment-management functions for the redirector. The RM has an XML-RPC [XML 04]-based control API, and the VNode manages the redirector using this API.
- *Service module card* (SMC) is an add-on card installed in the RB. An SMC is programmable because it contains a 10-Gbps-class network processor. It is used for the bidirectional conversion. Similarly to a normal VNode [Kan 12c], NACE requires packet-format conversion between slice-internal formats and external formats. The internal format is GRE-based, and the external format is VLAN-based.

The redirector is connected to the following networks. *Data plane* (D-plane) is a 10-Gbps network for sending and receiving data packets between VNodes. IP/Ethernet is used in the current implementation as described above. *Control plane* (C-plane) is a network (VLAN) between a VNM and a redirector in a VNode and between VNMs. The XML-RPC-based API between VNMs (and between VNodes) and a redirector uses the C-plane. *Internal data plane* is a closed network (VLAN) in a VNode.

Finally, the PPM is briefly explained. NACE should be programmable, but the programmability of NACE should be different from that of a normal VNode because NACE works as a network gateway or a special type of VNode. To reduce the size of NACE, programmer hardware, which is not required for the gateway and switching functions, was not added to this NACE implementation. Instead, a pseudo (software only) programmer for communicating with the VNM and RM was implemented in the redirector.

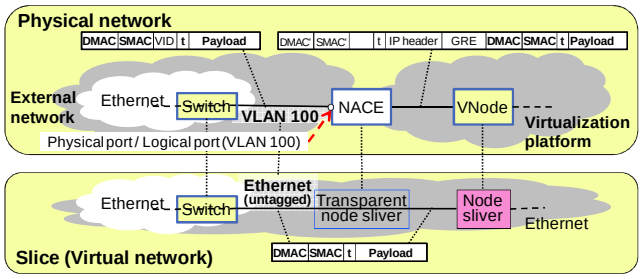
IV. NACE AS GATEWAY

NACE can function as a gateway that connects slices with external networks such as the Internet, an Ethernet-based network, or any other type of network. Slices on a virtualization network can be used through the Internet and access gateways. If NACE does not exist, slices are basically closed; that is, they cannot exchange packets with other networks such as the Internet. However, NACE enables external networks to provide services to slices and enables slices to provide services to external networks.

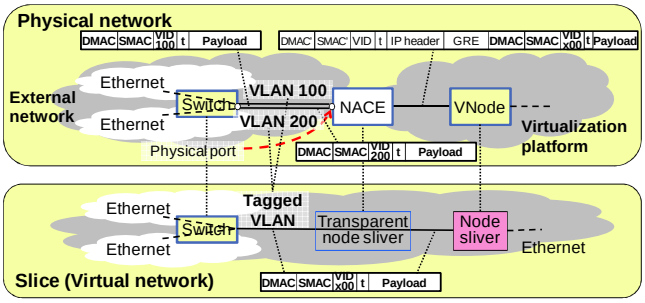
In the future, NACE will be able to accommodate slices and external networks with non-IP/non-Ethernet protocols. Although the VNode architecture supports non-IP/non-Ethernet protocols, the external networks are almost always IP and/or Ethernet networks. Therefore, network accommodation with high-speed data-format conversion between these protocols should be implemented. However, currently, NACE only supports Ethernet (and IP/Ethernet) networks.

The gateway function is modeled in the following way. A network-slice gateway can accommodate multiple external

 ISBN: 978-1-61208-226-4



(a) One-to-one accommodation



(b) Many-to-one accommodation

Figure 4. NACE accommodation types and packet formats

networks (i.e., only VLANs in the current version). A gateway should provide two types of network accommodation (**Figure 4**).

- *One-to-one accommodation* connects an external network (i.e., a VLAN or an Ethernet network) to a slice (Figure 4(a)). Node slivers on the slice cannot access non-virtualized information (i.e., a VLAN tag) on a packet that arrives at the external network, but they read an untagged packet. Subtypes of one-to-one accommodation are explained below.
- *Many-to-one accommodation* connects multiple networks (i.e., VLANs with different VLAN IDs) to a slice (Figure 4(b)). Node slivers on the slice can read and handle the network identifier (i.e., the VLAN IDs) on a packet as is. The physical and virtual packet formats used in the current NACE implementation are also shown.

One-to-one accommodation can be split into two subtypes.

- *Physical accommodation* connects an external network through a specified physical port of a network interface. The port number is specified in the slice definition (as part of a special-purpose node-sliver for network accommodation). In the current NACE version, the packet format of the external network may be tagged VLAN or untagged Ethernet. If a tagged format is used, the VLAN ID must be specified in the slice definition (in the node sliver). If no VLAN ID is specified, the untagged format is used.
- *Logical accommodation* connects an external network through a specified logical network name (i.e., a VLAN ID). In accordance with the configuration of the platform, the network can be connected to any physical port that is available, and packets may be either tagged or untagged. The VLAN ID is specified in the slice definition.

To accommodate an external network in a slice, the slice developer must include a node sliver in the XML-based slice definition. A node sliver usually contains a link to a programmable component such as a VM image. However, in this version of NACE, no node sliver can contain a programmable component because the NACE just forwards packets from the slice to the external network, or vice versa, as is, unless packet-header conversion is required for virtualization.

A node sliver with a gateway function is specified using an interface specification in the node-sliver definition. An example of interface definition is shown below.

```
<interfaces>
   <interface name="ExtIF" type="VLAN">
      <params><param key="VLANID" value="100" />
         <param key="port" value="1/2" />
      </params></interface></interfaces>
```

In the interface tag in this definition, the name property **"ExtIF"** specifies the identifier of the accommodation, and the type property **"VLAN"** specifies the type of external network. The above interface definition specifies a one-to-one accommodation, and **"1/2"** is used for the port number, and **"100"** is used for the VLAN ID. That is, only packets with VLAN ID 100 that arrive at port 1/2 are forwarded to the slice, VLAN ID 100 is added to packets that arrive at the node sliver, and the packets are forwarded to the port 1/2. The accommodation type is one-to-one because a specific port number is specified, but it is many-to-one if the port number is specified as **"all"**. The accommodation subtype is physical because a port number is specified.

These parameters are validated by the redirector and can be validated by the DC. However, some malicious access or erroneous accommodation is still possible. This problem should be solved in the future.

A node sliver that contains accommodation parameters is virtually managed by the PPM in NACE; the DC sends the node-sliver definition to the PPM through the XML-RPC interface. Because the accommodation function is actually implemented by the redirector, the parameters are passed to the RM, which configures the physical switch, i.e., the RB.

V. NACE AS VNODE FOR INTRA-SLICE SWITCHING

NACE can function as a special type of virtualization node that implements an intra-slice virtual switch by using Ethernet hardware. This function is intended to enable switching of not only Ethernet packets but also packets of arbitrary format with arbitrary type and size of addresses that can be mapped to Ethernet MAC addresses.

A slice may implement the Ethernet protocol or any other protocol suited for switching. If a packet contains the source and destination addresses of any format (with 48-bit (or less) effective address space), it can be switched by the same switching method as Ethernet switches. In addition, if the source and destination addresses can be computed from the packet content, content-based switching can be implemented by using the same hardware. The address converter implemented by using a network processor can be used for a wide range of conversions. However, in the current NACE version, only the Ethernet protocol can be used in slices.

The switching function is modeled as a special type of

 ISBN: 978-1-61208-226-4

node sliver called a *switch node-sliver* (**Figure 5**). In contrast to normal node slivers, this node sliver is not programmable, but several switch parameters can potentially be specified. The physical-packet formats used in these slivers in this version of NACE are also shown in this figure. The virtual-packet format always follows the Ethernet protocol.

In Ethernet networks, two types of forwarding functions are available: broadcasting (implemented by repeaters) and switching. We believe that a link sliver should be used for modeling an intra-slice broadcast function because broadcasting is non-intelligent. However, in contrast, a node sliver should be used for modeling an intra-slice switching function because switching is intelligent. Therefore, a node sliver specialized for the intra-slice switching function, namely, a switch node-sliver, is introduced into the VNode architecture.

A switch node-sliver is connected to other node slivers in normal VNodes by link slivers. It is natural to use VLAN-based link slivers to connect an Ethernet-based virtual switch to other virtual node functions; however, the virtualization platform currently only has GRE link slivers, which are thus used for connecting a switch node-sliver to other node slivers. These link slivers, which are specified as normal GRE link slivers, are implemented by using a specialized conversion program in SMCs. Therefore, although these link slivers are specified by using a normal XML definition, the switch node-slivers and these GRE link slivers are handled as a *macro* called a *switch sliver*, which is a combination of node and link slivers and functions as an Ethernet network (Figure 5).

A switch node-sliver can be specified using a switch instance definition in the node-sliver definition. An example of switch node-sliver definition is shown below.

```
<nodeSliver name="InSliceSW">
   <vports><vport name="p1" /><vport name="p2" />
      <vport name="p3" />
   </vports>
   <instance type="switch" /></nodeSliver>
```

It has three ports, p1, p2, and p3, for connecting to link slivers. Because no data conversion is used (currently not available), no other parameters are specified.

A switch node-sliver is virtually managed by the PPM in NACE; that is, the DC sends the node-sliver definition to the PPM through the XML-RPC interface. Because the switching function is actually implemented by the redirector, the parameters in the definition are passed to the RM through a control interface between the RM and the PPM, and the RM configures the physical interface of the RB.

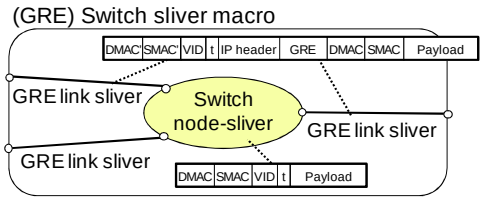


Figure 5. Switch node-sliver, switch sliver macro, and packet formats in the platform

VI. Applications of NACE

This section describes three applications of NACE; namely, gateway for data center, network with intra-slice switches, and slice interaction.

A. Gateway for Data Center

NACE can be used as a gateway for a data center (**Figure 6**). In typical cases, a VLAN used in a data center is connected to a NACE by using one-to-one accommodation. However, two or more VLANs can be accommodated in a slice by using many-to-one accommodation. The NACE can be placed either in the data center via a 10-Gbit Ethernet link or near another VNode of the platform. The NACE can be connected to a virtualization platform through an IP network, and the data center can be accommodated by using a GRE link sliver. The NACE is managed by the DC. Users of the slice can also use the servers in the data center. The maximum bandwidth between the slice and the data center is 10 Gbps. The platform does not manage the addresses of servers and PCs. If IP is used, the addresses are distributed in the usual manner by Address Resolution Protocol (ARP).

Instead of a data center, an external testbed can be connected to the platform and accommodated in a slice by using the same method. Any protocol can be used between a slice and the external testbed, and hardware-based OpenFlow [McK 08] can be used between them. The latter type of deployment is called “OpenFlow In A Slice” (OFIAS) [Du 12][Nak 11].

If protocol converters are inserted at the NACE and the AGW in Figure 6, respectively, a non-IP and/or non-Ethernet protocol can be used in the slice. For example, IP-Ether-Chimera (IPEC) [Kan 12a] can be used here. Each VNode in the platform contains an IPEC software switch as a node sliver. Usual IP/Ethernet is used both in a data center and a user terminal. If both NACE and AGW have a protocol-conversion function from IP/Ethernet to IPEC or vice versa, the user can use servers in the data center through the terminal. However, because NACE and AGW currently do not support this protocol conversion, VNodes connected to them must have the protocol-conversion function instead.

B. Network with Intra-slice Switches

Slices with one or more intra-slice Ethernet switches can be implemented. Two or more switch node-slivers are connected by using link slivers to form a spanning tree. Each slice may contain a different tree structure because a virtual network topology is not restricted by the physical topology. Although the platform packet-format used in

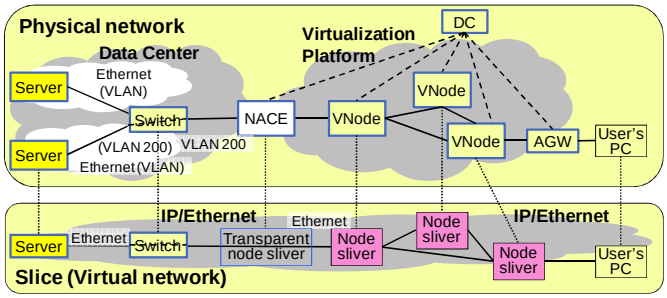


Figure 6. Network with a VLAN-slice gateway

NACE is different from that used in an Ethernet switch, it only implements the Ethernet switching function, so it is not very useful unless it is combined with other node slivers that convert the address format in VNodes. However, a future version of NACE will implement non-Ethernet switching functions customized to each slice.

C. Slice Interaction

A well-controlled interaction between slices, called *slice interaction* or *slice exchange*, may be useful in situations such as an extranet that connect multiple enterprise intranets. Two or more slices can be connected by using NACE. Instead of connecting external networks to the slices, two untagged Ethernet ports can be connected by using a wire for slice interaction. Although the current version of NACE cannot implement filters to control communication between the slices, filters will be implemented by using node slivers in a future version of NACE.

VII. EVALUATION

NACE was evaluated by using two slices on testbeds.

A. IPEC-Ethernet Protocol-conversion Gateway

A slice and PC server environments were set up as shown in **Figure 7** as a testbed for measuring the performance of NACE. Unlike Ethernet, IPEC allows redundant paths (loops) in the network [Kan 12a]. The user can use features of IPEC while using Ethernet packets on this network. This slice contains bidirectional IPEC-Ethernet protocol conversion function because, as described above, this version of NACE does not have protocol conversion function. This network consists of three node slivers on three VNodes, and three PCs through two NACEs and a gateway.

Performance between the PC servers and the PC client was measured by using **`iperf`** command. For these measurements, 2-Mbps UDP traffic was used, and the measurement results show that the packet loss rate is less than 0.1%. The performance is better than that in the LAN environment. Round-trip time, 2.8 ms on average, was measured by using a **`ping`** command.

B. VNode with Intra-slice Switch

A slice with a distributed key-value storage application was set up (**Figure 8**). In this slice, each node sliver has a database server. When a new key-value pair is stored in a server, it sends a packet that contains a key to the slice by

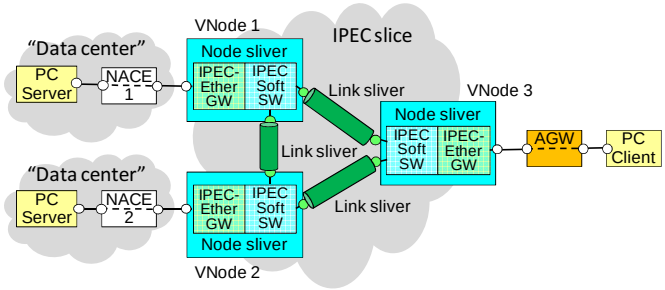


Figure 7. Experimental network with IPEC-Ethernet protocol conversion

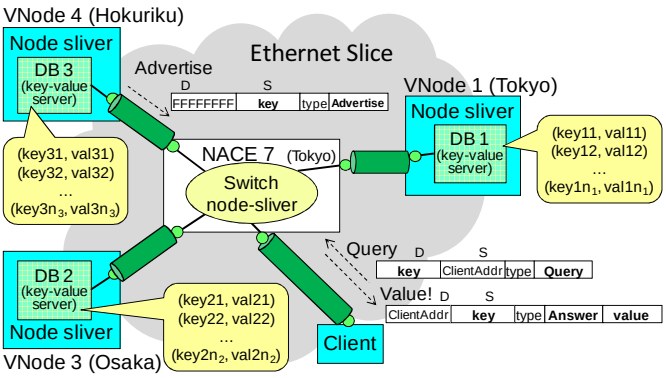


Figure 8. Experimental network for distributed key-value storage using intra-slice switch

using an advertise message. The intra-slice switch learns the key. When a client sends a query message that contains the key as the destination MAC address, the server that contains the key-value pair receives the message and sends an answer message that contains the value. A switch can usually learn 10k to 100k keys (addresses). If many more key-value pairs are stored, broadcast storms may occur. This application must also work on a non-virtual Ethernet-based network, but it may cause trouble in a network in real use because it is an unexpected usage of Ethernet. A virtualization platform with NACE is a good environment to test such applications.

In the evaluation, a client that randomly generates queries and three servers (each server for one third of the keys) were used. The virtual switch successfully learned and retrieved 16,384 keys, but it failed to learn 32,768 keys. The servers wait for 8 ms to learn a key, so it takes more than two minutes to send 16,384 advertise messages.

VIII. CONCLUSION

This paper has introduced a type of physical node, called NACE, which has two roles, i.e., a *network-slice gateway* and a VNode with an *intra-slice virtual switch*, to the VNode architecture. These roles are modeled as a node sliver with a gateway function and a node sliver with a switching function (i.e., a switch node-sliver), and these node slivers are specified by using XML. NACE contains an SMC with a network processor and enables 10-Gbps wire-rate accommodation. These functions were evaluated on two testbeds, and the evaluation results confirm that both functions work correctly and perform well in terms of delay and packet loss.

The most-important focus of future work is to add a high-performance protocol-conversion function to NACE. This function will enable accommodation of external network in a non-IP/non-Ethernet slice and intra-slice switching of packets of any format. Future work also includes automatic configuration of accommodation parameters, such as VLAN ID or physical-port number, and federation of virtualization networks (including those with our virtualization platform), which are connected by NACE. The federation should enable creation and management of slices across domains.

 ISBN: 978-1-61208-226-4

ACKNOWLEDGMENTS

We thank Atsushi Takahara and Noriyuki Takahashi from NTT for discussing the design of NACE and for implementing the management function. We also thank Akihiro Motoki from NEC, Ken'ichi Abiru from Fujitsu Laboratories, Makoto Kitani from ALAXALA Networks, Yasushi Kasugai from Hitachi, and other members of VNP for discussing the design of NACE.

Part of the research results described in this paper is an outcome of the Advanced Network Virtualization Platform (Project A), funded by the National Institute of Information and Communications Technology (NICT), and experiments using a testbed called JGN-X deployed by NICT.

REFERENCES

[AKA 10] AKARI Architecture Design Project, "New Generation Network Architecture — AKARI Conceptual Design (ver2.0)", http://akari-project.nict.go.jp/eng/concept-design/-AKARI_fulltext_e_preliminary_ver2.pdf, May 2010.

[Aoy 09] Aoyama, T., "A New Generation Network: Beyond the Internet and NGN", *IEEE Communications Magazine*, Vol. 47, Vol. 5, pp. 82–87, May 2009.

[Bav 06] Bavier, A., Feamster, N., Huang, M., Peterson, L., and Rexford, J., "In VINI Veritas: Realistic and Controlled Network Experimentation", *2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (*SIGCOMM'06*), pp. 3–14, 2006.

[Du 12] Ping Du, Maoke Chen, and Nakao, A., "OFIAS: A Platform for Exploring In-Network Processing", *TridentCom 2011, LNICST 90*, pp. 142–151, Springer, 2012.

[Far 00] Farinacci, D., Li, T., Hanks, S., Meyer, D., and Traina, P., "Generic Routing Encapsulation (GRE)", RFC 2784, IETF, March 2000.

[Fel 07] Feldmann, A., "Internet Clean-Slate Design: What and Why?", *ACM SIGCOMM Computer Communication Review*, Vol. 37, No. 3, pp. 59–74, July 2007.

[GEN 09] The GENI Project, "Lifecycle of a GENI Experiment", GENI-SE-SY-TS-UC-LC-01.2, April 2009, http://groups.geni.-net/geni/attachment/wiki/ExperimentLifecycleDocument/-ExperimentLifeCycle-v01.2.pdf?format=raw .

[ITU 12] ITU-T, "Framework of Network Virtualization for Future Networks", Recommendation, Y.3011, January 2012.

[Kan 12a] Kanada, Y. and Nakao, A., "Development of A Scalable Non-IP/Non-Ethernet Protocol With Learning-based Forwarding Method", *World Telecommunication Congress 2012* (*WTC 2012*), March 2012.

[Kan 12b] Kanada, Y., Shiraishi, K., and Nakao, A., "Network-resource Isolation for Virtualization Nodes", *17th IEEE Symposium on Computers and Communications* (*ISCC 2012*), July 2012.

[Kan 12c] Kanada, Y., Shiraishi, K., and Nakao, A., "Network-Virtualization Nodes that Support Mutually Independent Development and Evolution of Components", *13th IEEE International Conference on Communication System* (*ICCS 2012*), October 2012.

[Kou 01] Kounavis, M., Campbell, A., Chou, S., Modoux, F., Vicente, J., and Zhuang, H., "The Genesis Kernel: A Programming System for Spawning Network Architectures", *IEEE J. on Selected Areas in Commun.*, Vol. 19, No. 3, pp. 511–526, 2001.

[McK 08] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J., "OpenFlow: Enabling Innovation in Campus Networks", *ACM SIGCOMM Computer Communication Review*, pp. 69–74, Vol. 38, No. 2, April 2008.

[Nak 10] Nakao, A., "Virtual Node Project — Virtualization Technology for Building New-Generation Networks", *NICT News*, No. 393, pp. 1–6, Jun 2010.

[Nak 11] Nakao, A., Ping Du, and Maoke Chen, "OpenFlow In A Slice (OFIAS)", http://www.corelab.jp/images/poster2.pdf

[Nak 12a] Nakao, A., et al., "Advanced Network Virtualization: Definition, Benefits, Applications, and Technical Challenges, January 2012", NVSG White Paper v.1.0, https://nvlab.nakao-lab.org/nv-study-group-white-paper.v1.0.pdf

[Nak 12b] Nakao, A., "VNode: A Deeply Programmable Network Testbed Through Network Virtualization", *3rd IEICE Technical Committee on Network Virtualization*, March 2012, http://www.ieice.org/~nv/05-nv20120302-nakao.pdf

[Tur 07] Turner, J., Crowley, P., Dehart, J., Freestone, A., Heller, B., Kuhms, F., Kumar, S., Lockwood, J., Lu, J., Wilson, M., Wiseman, C., and Zar, D., "Supercharging PlanetLab — High Performance, Multi-Application, Overlay Network Platform", *ACM SIGCOMM Computer Communication Review*, Vol. 37, No. 4, pp. 85–96, October 2007.

[XML 04] XML-RPC Home Page, http://www.xmlrpc.com/.

 ISBN: 978-1-61208-226-4