# Implementation and Evaluation of VXLAN Gateway-based Data Center Network Virtualization

**Junji KINOSHITA[1]*, Kazuhiro MAEDA[1], Hitoshi YABUSAKI[1], Ken AKUNE[1], Motohide NOUMI[2], Norihisa KOMODA[3]**

[1] Center for Technology Innovation, Information and Telecommunications, Hitachi Ltd., 292 Yoshida-cho, Totsuka-ku, Yokohama, Kanagawa, 244-0817, Japan, {*junji.kinoshita.he|kazuhiro.maeda.yj|hitoshi.yabusaki.vw|ken.akune.dw}@hitachi.com

[2] Alaxala Networks Corp., 1-1-2, Kashimada, Kawasaki, Kanagawa, 212-0058, Japan, motohide.noumi@alaxala.com

[3] Codesolution, K.K., 1-2-11-9F, Edobori, Nishi-ku, Osaka, 550-0002, Japan, komoda@codesol.jp

**Abstract:** To achieve platform-independent network virtualization among multiple service infrastructures in service providers' data centers, we propose hardware gateway-based data center network virtualization architecture where we place a gateway on each service infrastructure and inter-connect them using overlay network virtualization. By considering a type of VXLAN gateway for this architecture, integration method with server virtualization environment and management mechanism for virtual network among service infrastructures, we propose the implementation method of our proposed architecture. We evaluated and confirmed feasibility of the proposed architecture based on our prototype. We also confirmed that the proposed architecture does not have any performance issue on our network throughput measurement evaluation.

**Keywords:** cloud, data center, network virtualization, VXLAN, gateway.

## 1. Introduction

As more and more enterprise companies and organizations have been using IT services like cloud computing, service providers have been facing challenges to achieve higher resource utilization and scalability in their data centers to become competitive in the market.

However, their data centers are becoming "siloed" environment where resources are physically divided into service infrastructures (silos) rather than a single flat resource pool. This is because scalability limitations of components used in those silos, like specification maximums of server virtualization software, network and storage system. As a result, service providers are suffering from problems like resource shortage in a silo, migration among silos and integration among silos.

To solve these problems, Layer 2 network extension technologies have been tried in the last several years so that service users' network can be expanded among different silos. But those solutions come with vendor lock-in or require replacement of whole inter-silo network.

We proposed a new architecture[1] that connects silos using VXLAN[2] (Virtual Extensible LAN) gateways to solve problems and also avoid vendor lock-in. We discuss implementation and evaluate feasibility of our approach. We also confirm there is no performance issue caused by this approach.

## 2. Overview and Challenges of Current Data Center

### 2.1 Overview of data center

In the IaaS (Infrastructure as a Service) service infrastructure, server virtualization software has been widely used where the server virtualization manager software manages hypervisor software on physical servers (hypervisor host), deploys Virtual Machines (VM) on hypervisor hosts and moves VMs among hypervisor hosts (live migration). There are commercial and open source server virtualization and IaaS software like VMware vSphere[3], Microsoft Hyper-V[4], OpenStack[5] and so forth. The logical network separation technology like VLAN[6] (Virtual LAN) or VXLAN are used to isolate VM network traffic for multi-tenancy, and

shared storage system is used to store VM images so that VMs can be moved among hypervisor hosts.

In such an environment, service infrastructure cannot scale well and has to be divided into silos because there are scalability limitations in server virtualization software, network and storage system.

**Limitations in server virtualization**: Server virtualization software has configuration maximums in its specification like the maximum number of hypervisor hosts, maximum number of VMs and so on. Some of them are explicit and documented, but some are not and recognized only in real practice. Even different versions of the same server virtualization software sometimes cannot work together and a service provider has to manage different versions respectively. When a service provider has a multi-vendor policy and uses different types of server virtualization software to avoid vendor lock-in, it is likely that they cause inter-connectivity problem as well. Even if a service provider tries to make homogeneous infrastructure, technology and software evolve and change day by day and result in heterogeneous infrastructure.

**Limitations in network**: Network also has scalability limitation like the maximum number of VLANs. Even though there have been software-based overlay network virtualization technologies[7] like VXLAN, NVGRE[8] (Network Virtualization using Generic Routing Encapsulation) and STT[9] (Stateless Transport Tunneling) practiced in the last several years to overcome VLAN limitations, its scalability is limited by server virtualization because it is implemented in software switch on hypervisor host and tightly integrated with server virtualization. Software-based overlay network virtualization has inter-connectivity problem among different server virtualization software as well. Even worse, it's not just technical problem, it's up to vendors' business strategies that service providers cannot control.

**Limitations in storage**: Shared storage system also has scalability limitations like the maximum IOPS, the maximum number of host computers and the maximum number of paths and so on.

When a service provider tries to provide a wide variety of services, not only IaaS, but bare metal service, managed service, PaaS (Platform as a Service) and SaaS (Software as a Service), they often have to separate each service infrastructure because of difference among service requirements.

As a result, service provider data center in a real environment is not a flat resource pool. It is divided into many silos as shown in Figure 1. In this example, a service provider data center consists of 3 silos where two of them provide IaaS but use different versions of server virtualization software, and one of them provides bare metal service and does not use server virtualization. Sometimes a silo is called a zone, an island and so forth. In a large data center, there might be even hierarchy among silos. A silo can be a rack or multiple racks.

In siloed environment, service providers need to take care of resource shortage in silos, silo transition and inter-service connection as shown in Figure 2.

**Resource shortage**: As the number of service users increases, service providers have to take care of resource shortage in a silo. If a user in a silo requests additional resources that exceed capacity of the silo, a service provider needs to expand the user system to a different silo.

**Silo transition**: As services grow, some silos get old and outdated. And service providers need to think about shutting down old silos. In case users still exist in old silos, service providers need to temporarily expand users' systems to new silos so that they can migrate users.

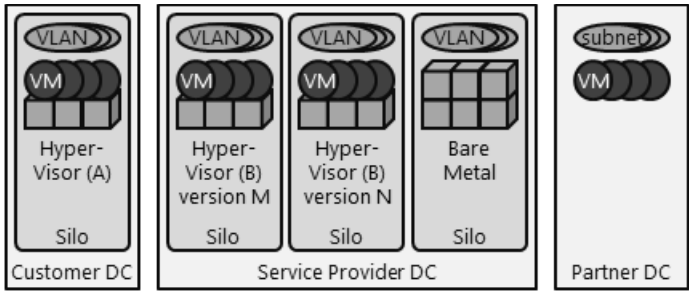


**Figure 1.** Overview of Data Center

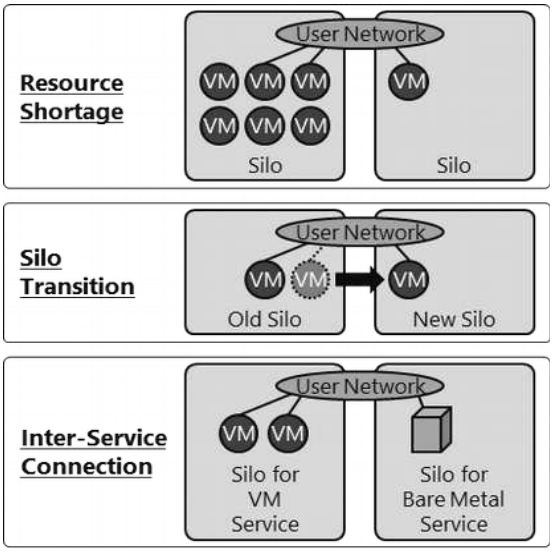


**Figure 2.** L2 extension

**Inter-service connection**: When a service provider has several services and a user of one service requests another service hosted in a different silo, the service provider might need to expand a user system across different service silos.

Because silos are problems in service infrastructure behind the scene, service providers cannot enforce their users to change network configuration like IP addresses when they have to expand users' systems among silos. So, extending user network across silos in Layer2 (L2) has been required so that users don't have to change network configuration (L2 extension). The L2 extension among silos also has to be silo agnostic so that service providers can use it as inter-silo connection regardless of service and platform inside silos.

## 2.2 Current L2 extension solution

**VPN**: In this approach, a L2 network is encapsulated and transferred to another silo using VPN (Virtual Private Network). There are many internet VPN technologies like IPsec[10], SSL VPN[11], and carrier VPN technologies like PBB[12] (Provider Backbone Bridge), Q-in-Q[13], MPLS[14] (Multiprotocol Label Switching). Because internet VPN technologies are intended for a point-to-point connection, it is difficult to use for multiple silos. Regarding carrier VPN technologies, they are not always supported in commodity data center network switches. In case of PBB and Q-in-Q, L2 over L2 requires a large L2 network among silos and could cause lack of topology flexibility and a reliability issue.

**Network Fabric**: Network Fabric is a technology to tightly couple network switches as shown in Figure 3. It is originally intended to increase bisection bandwidth and scalability of data center network. It is implemented as a group of switches and they are connected in a specific topology like leaf-spine, fat-tree and so forth. Since some implementations use either a standard or a proprietary encapsulation mechanism among switches, it can be used for inter-silo connection[15,16]. But network fabric comes with tight interaction among network switches and requires the same vendor's network switches. This could cause limitation in flexibility of inter-silo network, inter-operability issue, and thus, replacement of whole inter-silo network.

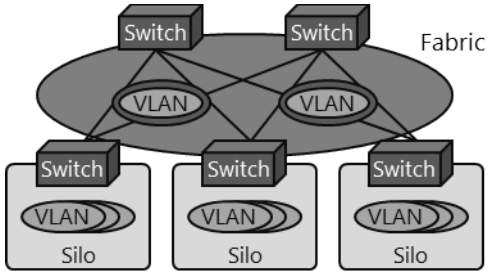


**Figure 3.** Network Fabric

**Hop-by-Hop traffic control**: In this approach, network traffic is controlled from a centralized controller using a protocol like OpenFlow [17] and forwarded to an another silo as shown in Figure 4. This approach could bring flexibility

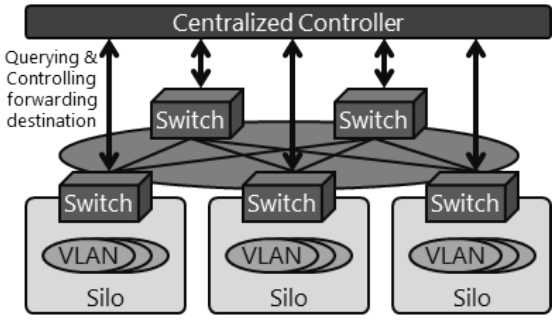


**Figure 4.** Hop-by-Hop traffic control

but a controller could become a single point of failure and scalability bottleneck. And it requires replacement of whole inter-silo network switches.

# 3. Network Virtualization using VXLAN Gateway

## 3.1 Proposed architecture

To overcome challenges in inter-silo L2 extension using current technologies, we propose VXLAN gateway-based network virtualization as shown in Figure 5. In our approach, we place a VXLAN gateway on the edge of each silo, virtualize network (VLAN) inside a silo, and connect multiple silos using overlay network virtualization (VXLAN) over existing inter-silo network. By removing VLAN tag at each gateway, this approach can inter-connect different VLANs in different silos using the same VXLAN. Because this approach doesn't rely on software used inside silos, it can create platform-independent virtual network among any type of service infrastructure like IaaS infrastructure using commercial server virtualization software, IaaS infrastructure using another version or another vendor of software virtualization software, IaaS infrastructure using open source software, and even another service infrastructure like bare metal service, managed hosting, PaaS and SaaS. And because it expects ordinary VLAN as silo network and uses L2 over L3, it can be applied to inter-connection among existing silos. Regarding deployment of VXLAN gateway, there are several options like per-rack deployment, per-racks (row, floor, etc.) deployment and per-DC (data center) deployment as shown in Figure 6.

To achieve this architecture in a real data center, we need to consider a type of VXLAN gateway, integration method with server virtualization environment, management mechanism for virtual network among silos.

## 3.2 Types of VXLAN gateway

There are two types of VXLAN gateway implementations: software and hardware.

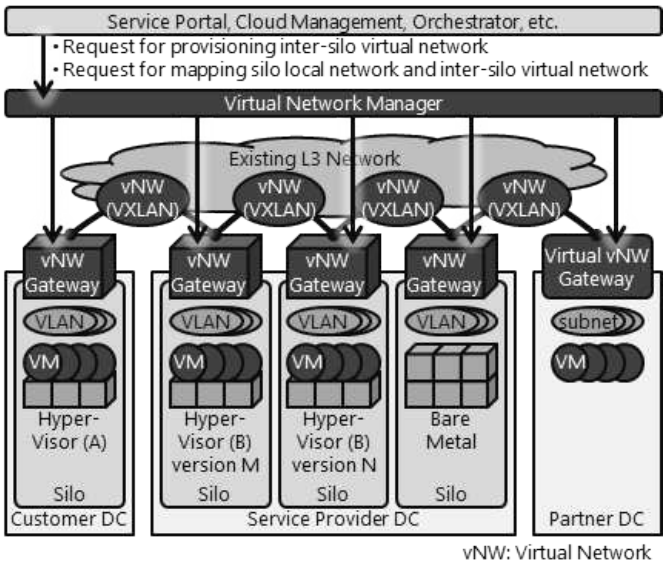


**Figure 5.** Gateway-based Network Virtualization

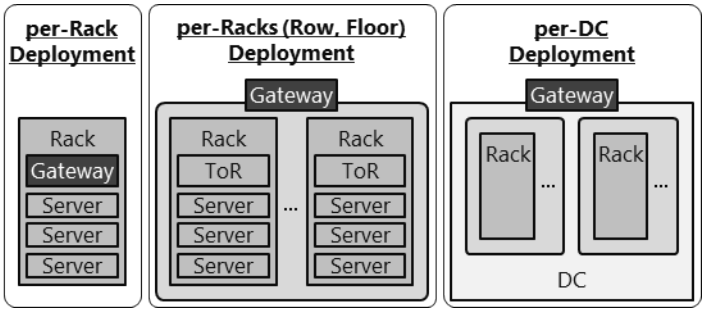


**Figure 6.** Deployment of VXLAN Gateway

**Software**: Software-based VXLAN gateway is implemented as VTEP (Virtual Tunnel End Point) function in software[18] running on either a virtual machine on a server, or a bare server. Because a software-based VXLAN gateway runs on a physical server anyway, its reliability is affected by a physical server. In a data center, it is already known that server-based approach can become causes of failures while network devices are highly reliable[19,20]. So, using a server as a network device like a gateway is not always a good idea. And a typical rack-mount 1U-sized physical server can have only 2 or 4 network interfaces in general. It is not cost effective compared to a network device like a switch because a typical 1U-sized switch can have 48 network interfaces. Especially considering per-rack VXLAN gateway deployment in a standard 42U-sized rack, software-based approach is costly.

**Hardware**: A hardware VXLAN gateway is a VXLAN capable switch and originally intended to complement software-based VXLAN as shown in Figure 7. Because a software-based overlay network virtualization environment needs to connect legacy non-virtualized environment, either software or hardware gateway is necessary. Since the de-facto ASIC (Application-Specific Integrated Circuit) has supported overlay virtualization protocols, some vendors are releasing hardware gateways in the market as VXLAN gateway products. Those VXLAN gateway products are not necessarily Network Fabric products. And it is used to connect a silo that uses software-based VXLAN as inside-silo network to legacy VLAN network. But in our approach, we use it to connect inside-silo VLAN to inter-silo VXLAN.

### 3.3 Integration with server virtualization

Because we expect ordinary VLAN as silo network, we need to set VLAN configuration for gateway network interfaces as we do for ordinary network switches. But we cannot expect VLAN IDs for gateway network interfaces in advance in a server virtualization environment, because server virtualization manager software automatically live-migrates VMs among hypervisor hosts to equalize load of hypervisor hosts.

The software-based overlay network virtualization solves this issue using tight integration with server virtualization software. It is possible to follow the same way. But integration interface isn't always open and the feasibility depends on server virtualization software vendors' business strategies. Even if the integration interfaces are open, they can differ among different versions and different software and requires service provider to continuously develop integration that would result in a lot of development cost.

So we set all (4k) the VLANs on all the gateway interfaces so that we don't have to automatically configure VLANs in accordance with VM deploy and live-migration. In that case, a VTEP feature of a VXLAN gateway has to hold a number of mapping information between VLAN and VXLAN. For example, if a VXLAN gateway has 48 network interfaces and we set 4k VLANs for all the interfaces, a VTEP of the VXLAN gateway needs to hold 48 x 4k mapping information. But ASIC used in VXLAN gateway doesn't have enough memory to hold all the mappings between VLANs and VXLANs on its VETP feature for all the interfaces.

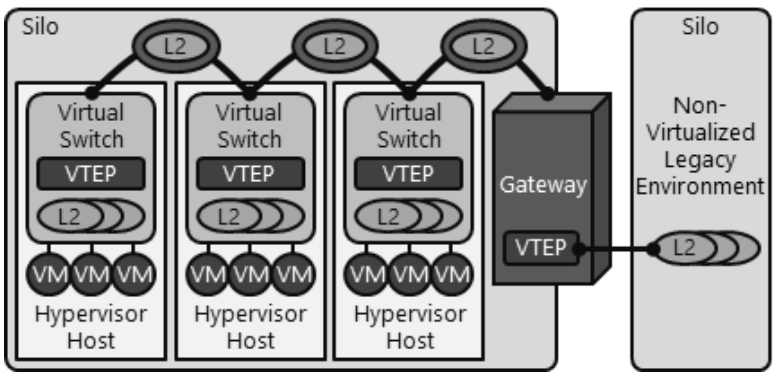


**Figure 7.** Hardware VXLAN gateway

To solve this problem, we use "switch-back" on each VXLAN gateway to aggregate traffic from servers and reduce the number of mappings between VLANs and VXLANs as shown in Figure 8. "switch-back" can be either physical using cables and logical using internal links. Physical "switch-back" requires physical interfaces, decreases the total amount of switching traffic and causes cabling nightmare. But physical "switch-back" can be applied for existing products that don't have internal "switch-back" feature.

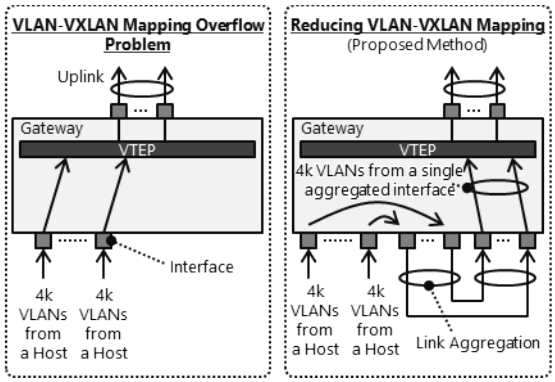


**Figure 8.** Reducing VLAN-VXLAN Mapping

### 3.4 Virtual Network Management

Inter-silo virtual networks become necessary when a user system in one silo needs to be expanded to another silo. Only service management layer (service portal, cloud management, system orchestration and so forth) can recognize which silos should be inter-connected with virtual networks. On the other hand, relationship between a gateway, a silo and a VLAN domain (a single 4k VLAN ID spaces) differs from each environment. So we use a virtual network manager that receives a requirement to inter-connect silo networks from the service management layer through API (Application Program Interface) and deploys mapping between an inter-silo virtual network and silo networks on gateways based on relationships between gateways, silos and VLAN domains.

Because there are several options for VXLAN gateway deployment (per-rack, per-racks, per-DC), one silo can contain multiple VLAN domains especially when the silo is large. On the other hand, one VLAN domain can expand among multiple VXLAN gateway. When the virtual network manager manages mappings between VLANs and VXLANs, it needs to assume that there can be multiple VLAN domains under one VXLAN gateway or there can be one VLAN domain among multiple VXLAN gateways.

To solve this management problem, we introduce "interface group" to manage VLANs in a silo as shown in Figure 9. The interface group is a group of VXLAN gateway interfaces where each interface can be on a different VXLAN gateway. Each interface group hosts one VLAN domain and can be expanded to several gateways. By using the interface group instead of binding VLANs with either VXLAN gateways or silos, we can manage mappings between VLANs and VXLANs for a variety of VXLAN gateway deployment cases.

## 4. Evaluation

### 4.1 Feasibility evaluation

To evaluate feasibility of our approach, we created an experimental setup of this architecture using Alaxala AX4630S as a VXLAN gateway and a virtual network manager software running on Linux (Figure 10). We created 4 experimental silos using

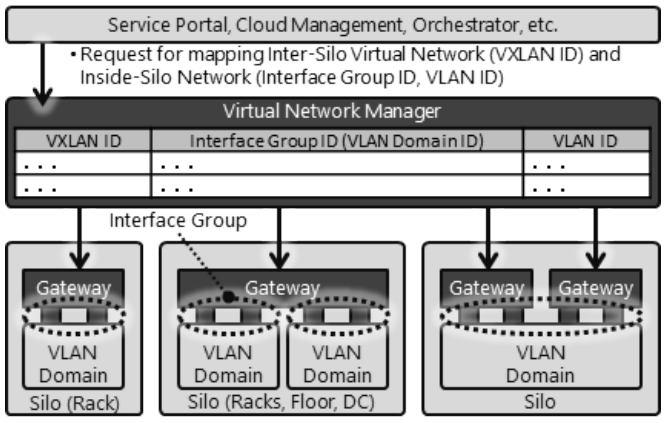


**Figure 9.** Managing VLAN-VXLAN Mapping

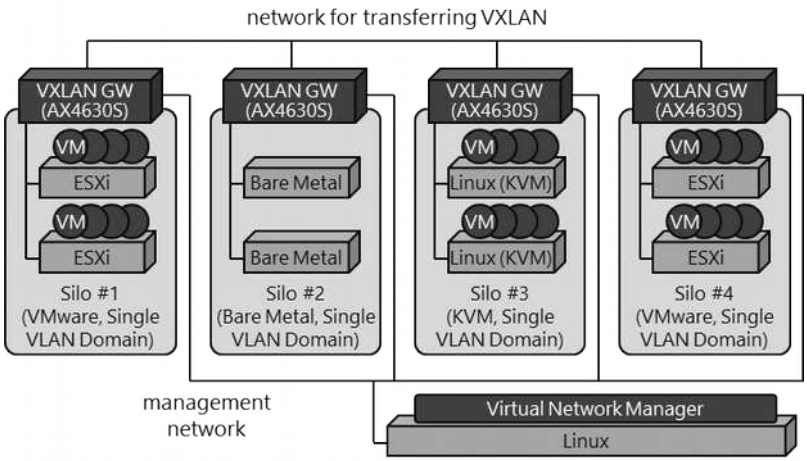


**Figure 10.** Prototype

different platform: two different VMware vSphere, bare metal server and KVM. We assumed that each silo hosted different service and had a single VLAN domain. We placed a gateway on an each silo and connected gateways. We also connected gateways to a management network so that our virtual network manager software, that is running on a Linux server connected to a management network, can control gateways to change mappings between silo network and inter-silo VXLAN. While we could have many options of a control protocol, we controlled gateways using SSH (Secure Shell Protocol) in this evaluation. When we connected VLANs among silos using inter-silo VXLAN, our virtual network manager automatically perfomed login to gateways and changed VLAN-VXLAN mapping configuration. By connecting a VLAN in a silo and VLANs in different silos via the same inter-silo VXLAN, we confirmed that this architecture can inter-connect multiple silos with different platform.

## 4.2 Performance evaluation

### 4.2.1 Evaluation environment

We also evaluated performance to make sure there is no performance issue on our approach. We created an experimental environment as shown in Figure 11 and Table 1 where two physical servers are connected through two VXLAN gateways. Each server represents a silo and hosts virtual machines. We measured network traffic throughput between virtual machines on two physical servers using two different network virtualization methods: software-based network virtualization and VXLAN gateway-based network virtualization (our approach). We compared throughput to check if our approach has performance problem compared to current software-based solution.

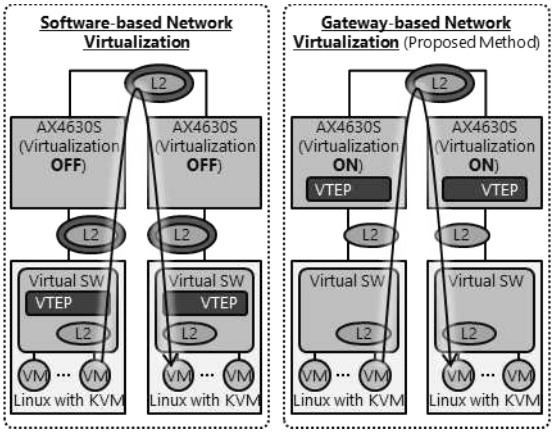


**Figure 11.** Experimental Setup

**Table 1.** Specifications of Hardware and Software

| |
|---|
| Hypervisor Host for VMs that send traffic<br>CPU: Intel(R) Xeon(R) E5-2470, 16 cores, 2.30 GHz<br>Memory: 86GB |
| Hypervisor Host for VMs that receive traffic<br>CPU: Intel(R) Xeon(R) E5-2470, 16 cores, 2.30 GHz<br>Memory: 94GB |
| VM that sends and receives traffic<br>CPU: QEMU Virtual CPU version 2.0.0, 1 core<br>Memory: 1GB |
| VXLAN gateway<br>AX4630S-4M, Firmware version 11.13 |
| L3 switch (between VXLAN gateways)<br>Arista DCS-7050S-52-F, Firmware version 4.11.7 |
| Software<br>Hypervisor Host OS: Ubuntu 14.04.2 LTS, Linux kernel 3.13.0-54-generic<br>Hypervisor: qemu-kvm 2.0.0<br>Virtual Switch: Open vSwitch 2.0.2<br>VM OS: Ubuntu 14.04.3 LTS, Linux kernel 3.13.0-71-generic<br>Software-based network virtualization: VXLAN feature of Linux kernel<br>Throughput measurement: iperf 2.0.5 |

### 4.2.2 Evaluation method

We put a pair of virtual machines on two hypervisor hosts where each virtual machine is on each hypervisor host respectively. And we measured network throughput using a network performance measurement software "iperf". We measured throughput for 60 seconds at an interval of one second, and calculated an average of 60 seconds.

In the measurement, we used different packet sizes (88, 100, 250, 500, 1000, 1500, 2000, 3000, 4000, 5000, 6000, 7000, 8000 byte) and measured 60 seconds average throughput for each packet size. We set MTU (Maximum Transmission Unit) a value more than 8000 byte on all the network interface and VXLAN gateways so that network between virtual machines doesn't affect the result of the measurement.

And then, we increased the pair of virtual machines from 1 to 10 to make our evaluation more realistic where each pair performs iperf in parallel. We measured 60 seconds average throughput in each packet size for each pair, and calculated average throughput of one pair of virtual machines.

We performed this entire measurement for software-based network virtualization and VXLAN gateway-based network virtualization respectively.

### 4.2.3 Evaluation result

The results of the throughput measurement are shown in Figure 12 (software-based network virtualization) and Figure 13 (VXLAN gateway-based network virtualization). Both reached maximum bandwidth (10Gbps) especially in case of larger packet size or multiple pairs of virtual machines, and we confirmed that our approach didn't have any performance problem. Our approach showed even better throughput in case of smaller packet sizes. In software-based network virtualization environment, the throughput reached the maximum bandwidth when the packet size was over 4000 (two or more pairs of virtual machines) or 6000 (a single pair of virtual machines). On the other hand, in

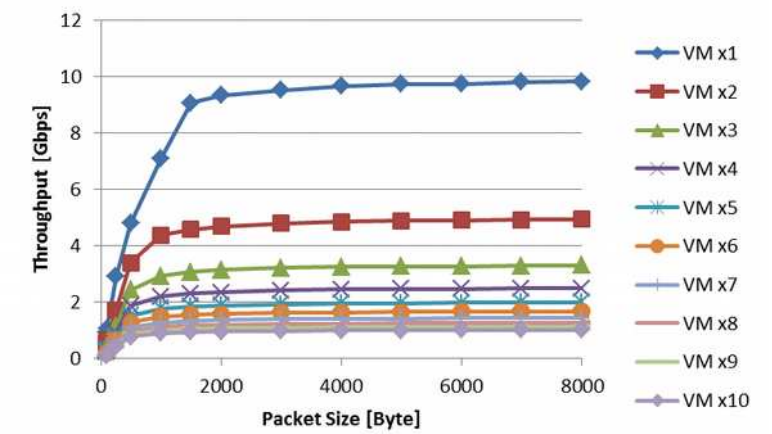


**Figure 12.** Throughput (Software-based Network Virtualization)

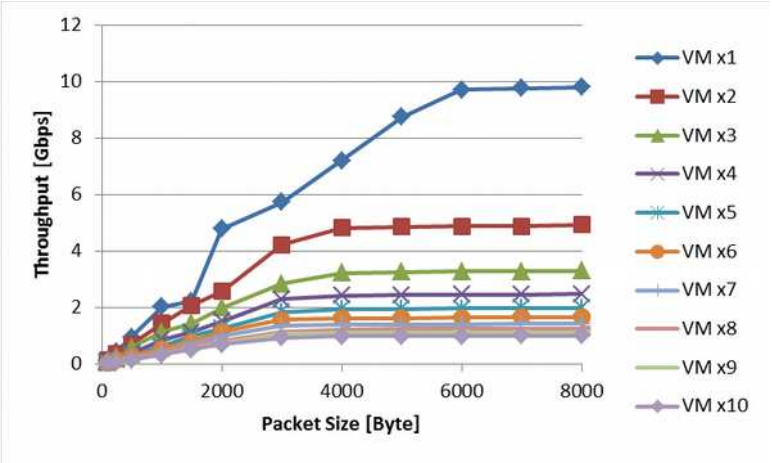


**Figure 13.** Throughput (Gateway-based Network Virtualization)

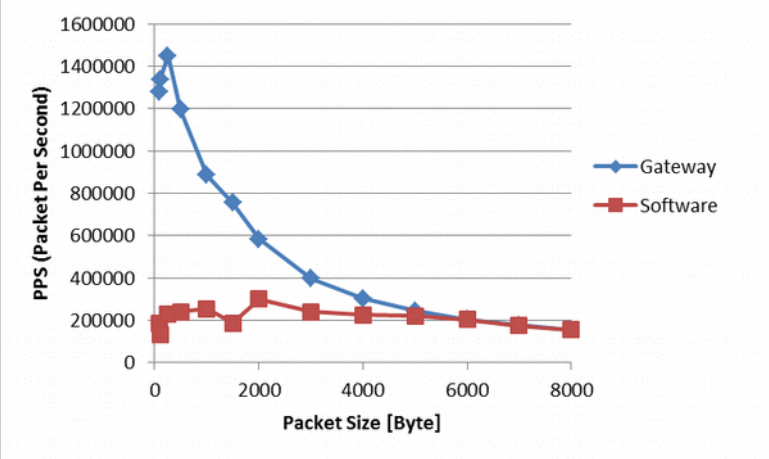


**Figure 14**. Packet per Second.

VXLAN gateway-based network virtualization environment, throughput almost reached the maximum bandwidth even in Ethernet (Ethernet is a registered trademark of Xerox corporation) standard packet size (1500 byte). This might be because software-based network virtualization had overhead of packet encapsulation and resulted in the limitation of the number of packets in a unit of time. Figure 14 shows packets-per-second (pps) of software-based and VXLAN gateway-based network virtualization calculated from the result of measurement shown in Figure 12 and Figure 13. We can see pps of software-based network virtualization was almost constant. This might be a problem of specific software-based network virtualization implementation that we used in our evaluation and other implementations could have better results. But we can conclude that our approach is not affected by those implementations and can achieve platform independent network virtualization.

## 4. Conclusions

We proposed a new architecture that connects silos using VXLAN (Virtual Extensible LAN) gateways to solve problems like resource shortage in a silo, migration among silos and integration among silos, and also avoid vendor lock-in. We considered a type of VXLAN gateway, integration method with server virtualization environment and management mechanism for virtual network among silos. We implemented a prototype and evaluated feasibility of our approach. We also confirmed that there was no performance issue caused by this approach.

## Acknowledgements

In the prototyping of this gateway-based network virtualization architecture, Alaxala Networks let us use a hardware gateway AX4630S. We appreciate their support.

## REFERENCES

1. KINOSHITA, J., K. MAEDA, H. YABUSAKI, K. AKUNE, N. KOMODA, **Realization of VXLAN Gateway-based Data Center Network Virtualization**, IIAI AAI EAIS, 2016.
2. **Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks**, https://tools.ietf.org/html/rfc7348, IETF, RFC7348, 2014.
3. **vSphere**, https://www.vmware.com/products/vsphere, last visited 2016-01-13.
4. **Hyper-V**, http://www.microsoft.com/en-us/server-cloud/solutions/virtualization.aspx, last visited 2016-01-13.
5. **OpenStack**, https://www.openstack.org/, last visited 2016-01-13.
6. **802.1Q - Virtual LANs**, http://www.ieee802.org/1/pages/802.1Q.html, IEEE, last visited 2016-01-13.
7. PFAFF, B., J. PETTIT, T. KOPENEN, K. AMIDON, M. CASADO, S. SHENKER, **Extending Networking into the Virtualization Layer**, ACM SIGCOMM Workshop on Hot Topics in Networking (HotNets), 2009.

8. **NVGRE: Network Virtualization Using Generic Routing Encapsulation**, https://tools.ietf.org/html/rfc7637, IETF, RFC7637, 2015.
9. **A Stateless Transport Tunneling Protocol for Network Virtualization (STT)**, https://tools.ietf.org/html/draft-davie-stt-06, IETF, 2014 last visited 2016-01-13
10. **Security Architecture for the Internet Protocol**, https://tools.ietf.org/html/rfc4301, IETF, RFC4301, 2005.
11. **The Secure Sockets Layer (SSL) Protocol Version 3.0**, https://tools.ietf.org/html/rfc6101, IETF, RFC6101, 2011.
12. **802.1ah - Provider Backbone Bridges**, http://www.ieee802.org/1/pages/802.1ah.html, IEEE, last visited 2016-01-13
13. **802.1ad - Provider Bridges**, http://www.ieee802.org/1/pages/802.1ad.html, IEEE, last visited 2016-01-13
14. **Multiprotocol Label Switching Architecture**," https://tools.ietf.org/html/rfc3031, IETF, RFC3031, 2001.
15. Multitenancy Options in Brocade VCS Fabrics, https://www.brocade.com/content/dam/common/documents/content-types/whitepaper/multitenancy-options-wp.pdf, Brocade, last visited 2016-01-13.
16. **Cisco Application Centric Infrastructure Fundamentals**, http://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/1-x/aci-fundamentals/b_ACI-Fundamentals/b_ACI_Fundamentals_BigBook_chapter_0100.html, last visited 2016-08-13.
17. McKEOWN, N. T. ANDERSON, H. BALAKRISHNAN, G. PARULKAR, L. PETERSON, J. REXFORD, S. SHENKER, J. TURNER, **OpenFlow: Enabling Innovation in Campus Networks**, in ACM SIGCOMM Computer Communication Review, vol. 38(2), 2008, pp. 69-74.
18. LIM, C.-G., S.-M. PAHK, T.-I. KIM, J.-H. LEE, **Design and Implementation of Hardware Accelerated VTEP in Datacentre Networks**, Advanced Communication Tech. (ICACT), 2015.
19. GILL, P., N. JAIN, N. NAGAPPAN, **Understanding Network Failures in Data Centers: Measurement, Analysis, and Implications**, ACM SIGCOMM 11 Proceedings of the ACM SIGCOMM 2011 conference, 2011, pp. 350-361.
20. SINGH, A., J. ONG, A. AGARWAL, G. ANDERSON, A. ARMISTEAD, R. BANNON, S. BOVING, G. DESAI, B. FELDERMAN, P. GERMANO, A. KANAGALA, J. PROVOST, J. SIMMONS, E. TANDA, J. WANDERER, U. HÖLZLE, S. STUART, A. VAHDAT, **Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network**, ACM SIGCOMM 15 Proc. of the 2015 ACM Conference on Special Interest Group on Data Communication, 2015, pp. 183-197.