Abstraction Layer Based Distributed Architecture for Virtualized Data Centers

Ali Kashif Bashir, Yuichi Ohsita, and Masayuki Murata
Graduate School of Information Science and Technology
Osaka University
Osaka, Japan.
e-mail: {ali-b, y-ohsita, murata}@ist.osaka-u.ac.jp

***Abstract*— Network virtualization was envisioned to enhance the capabilities of data centers. However, existing virtual data center network architectures are unable to exploit the features of network virtualization. In this paper, we propose a distributed virtual architecture that groups virtual machines into clusters of different service types. This architecture introduces a concept named *abstraction layer* consisting of virtual switches that are logically grouped together to perform the role of cluster heads. The abstraction layer provides a better control and management of clusters. This architecture enables several features of network virtualization such as scalability, flexibility, high bandwidth, etc. However, in this work, we evaluated the failure of servers and virtual machines to prove the efficiency and scalability of the architecture.**

Keyword-virtualization; infrastructure for clouds; data center network architecture; future internet; scalable data center architecture.

I. INTRODUCTION

Data Center Networks (DCNs) are experiencing a rapid growth in both scale and complexity as they can host large-scale applications such as cloud-hosting. Such growth imposes huge challenges to upgrade the current infrastructure of data centers. However, the current infrastructure is owned by a large number of Internet Service Providers (ISPs) and it is difficult to adopt new architectures without the agreement of all stakeholders.

Virtualization is a technique where the functionalities of server are copied to Virtual Machines (VMs). With server virtualization, we can create multiple logical VMs on top of a single server that can support various applications e.g. VMware [1] and Xen [2]. These VMs can take away the computation from servers. However, a Virtual Network (VN) is a virtual topology that connects devices of the VN or physical network [3]. One of the properties of VN is that links can be added and deleted easily in it.

Network virtualization [4]-[6] can be defined as a technique where multiple VNs are created on top of a physical DCN. It was envisioned to provide several features to the data centers to support several cloud applications. Some of these features are: scalability to network expansion, adaptability to demands of users, and improve network performance in terms of bandwidth and energy, etc. However, existing virtual architectures of DCNs [6]-[10] provide only one or two features at a time and utilize network resources poorly. Therefore, they are unable to exploit most of the features of network virtualization.

Literature work in virtualizing data centers can be divided into centralized and decentralized approaches. The main centralized architectures are SecondNet [7], which provides bandwidth guarantees and CloudNaas [8], which provides support for deploying and managing applications Centralized architectures suffer when network expands. In decentralized approaches, PolyVine [9] and adaptive VN [10] are two worth discussing approaches. Polyvine embeds end to end VNs in decentralized manners. Instead of technical, it resolves the legal issues among infrastructure providers. In adaptive VNs [10], every server is supposed to have an agent. Each server agent communicates with another to make local decisions. This approach is expensive and needs additional hardware. In general, decentralized architectures have obvious advantages over centralized ones. They have no single point of failure, can run multiple applications concurrently, and are scalable and flexible to network changes.

To exploit the features of network virtualization, in this paper, we propose a distributed virtual architecture named A*bstraction Layer Based Virtual Clusters* (AL-VC) for data centers where VMs are grouped into clusters according to their service types. *Abstraction Layer* (AL), used first time in network virtualization architectures, is a key concept of this paper. An AL is created by logically combining a subset of VN switches with an identifier. One AL is assigned to each group of VMs and they jointly form a cluster where AL will perform the jobs of a cluster head.

Introducing AL helps in managing clusters and brings several features to the virtual architecture, such as making AL-VC scalable, adaptable, and flexible. We will discuss these features in the next section. Though AL-VC offers several features, however, in this work, we evaluated its scalability and efficiency in replacing failed VMs or servers.

The rest of the paper is organized as follows: in Section 2, we present the overview of the proposed architecture and discussed its topology, and addressing during routing. Section 3 includes the AL construction algorithm and discusses the features it offers. In Section 4, we present the evaluation of this work and Section 5 concludes the paper.

II. SYSTEM OVERVIEW

Virtual data centers are those where some or all of the hardware of the data center is virtualized. A virtual data center is a collection of virtual resources connected via virtual links. This section discusses the overview of AL-VC. It is important to mention that this work does not provide any VM mapping algorithm. There are several VM mapping algorithm proposed such as [12] that can be used for VM mapping. Therefore, in this work, we assume that servers are already hosting VMs.

A. Architectual Overview

Virtual Clusters (VCs) are more desirable than physical data center because the resource allocation in VC can be rapidly adjusted to meet changing needs of the users. DCN have high correlations. In data centers, every server provides a set of services and their data usually have a high correlation [12]. VMs hosted by these servers also provide similar servers; therefore, they need to interact with each other frequently to provide services to the users. To take advantage of this, in our approach we group VMs into clusters according to their service types, as shown in Figure 1, where VMs offering Social Networking Services (SNS) form one cluster, VMs offering web services form another one, and so on. Forming clusters according to service types will save search and allocation time of queries [13].

B. Topology

Ideally, VN topology should be constructed in a way that it achieves minimum energy consumption, larger bandwidth without much delay. Minimum energy consumption can be achieved by minimizing the active number of ports and constructing energy efficient routes. Larger bandwidth can be achieved by adding virtual links in the VN and by managing traffic efficiently. Delay can be improved by using efficient routes and by processing data faster at switches. Our architecture is capable to meet these challenges.

The topology of AL-VC is presented in Figure 2, where all the servers in the server racks are connected to one Top-of-the-Rack (ToR) switch. Each server is hosting multiple VMs. To construct VN, we use virtual switches name as Optical Packet Switches (OPSs) as they provide large bandwidth and small energy consumption [14]. They are capable to store, buffer, and can inter- convert electronic and optical packets. Note that TOR switches produce electronic packets and, in order to route those packets over VN, they first need to be converted into optical packets. OPSs send optical packet and they need to be converted before forwarding to TOR switches. An OPS has the tendency to do this inter-conversion. In AL-VC, we restricted the communication among VMs only via OPSs. Every VM is connected to multiple OPSs. OPS that joins a particular AL can have four possible types of connections, namely: 1- with TOR switches, 2- with VMs of local cluster, 3- with OPSs of local AL, and 4- with OPSs of VN that are not part of its local AL. In Figure 2, we show the block diagram of this connectivity and as well the logical construction of AL-VC.

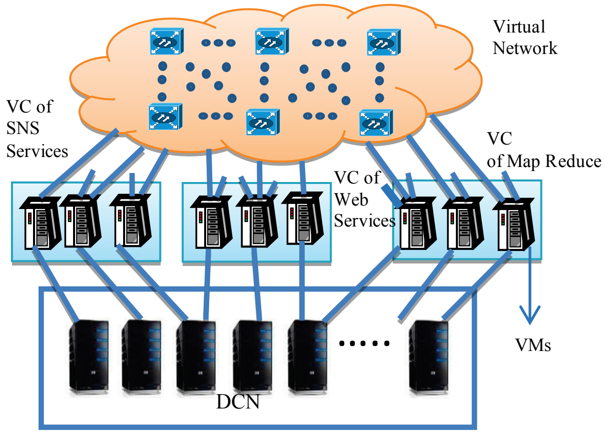


Figure 1. Overview of AL-VC

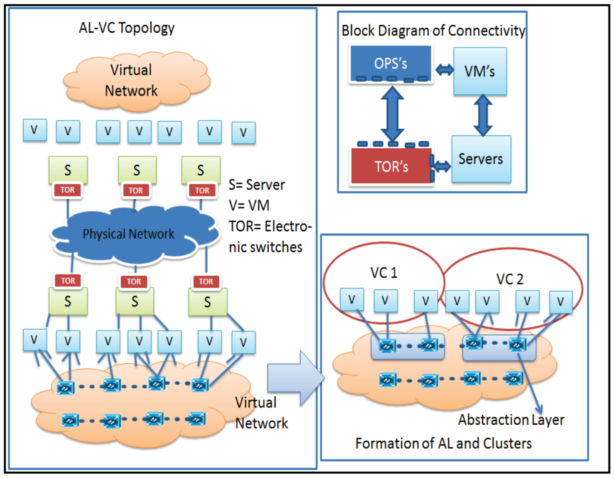


Figure 2. AL-VC Topology

C. Addressing

AL-VC is monitored by a central entity called *network manager*. It monitors and controls all resources such as servers, VMs, links, etc. Network manager is responsible for VC formation and deletion. It decides the number of clusters according to service types, sizes of the clusters, and how they are mapped to the servers. It also assigns each VC with a unique VC_{ID} and IP address. However, controlling and managing the cluster after creation is the job of its AL. For address isolation, every VC has its own IP address space. VMs within a cluster communicate with each other via AL.

III. ABSTRACTION LAYER

In this section, we first present our AL construction algorithm and then we discuss the advantages that an AL offers to distributed architectures.

A. Construction of an AL

The basic idea behind the construction of an AL is logically allocating a subset of VN switches to a particular group of VMs. Each switch in an AL knows the topology of its cluster such as VMs locations and their connections.

To construct an AL, VMs of every cluster selects the minimum subset of OPSs that connect all the VMs. This approach selects the switches with highest connections and then switches with second highest connections and so on until all the VMs are connected. The subset of switches that covers all the VMs of a cluster will be declared as its AL. They can be distinguished from other switches of VN with the respective cluster ID. Information of these switches such as switch ID and IP addresses is forwarded to all the VMs. All devices will update their routing tables to identify other switches of the AL. This procedure is repeated for every cluster until all the clusters have an AL. The detailed mechanism is as follows:

Step 1: After VMs are grouped into clusters, they connect themselves to the switches of VN. These connections can be established randomly or based on a specific criterion. In this work, we use random approach shown in Figure 3. The selection probability of the switches of AL is based on the distance, in which we have

$$P_i = \frac{R_i}{\sum_j d_j} \tag{1}$$

where
P_i = probability of selecting switches v_s^i
d_j = distance of switches from VM

Step 2: Each VM sends a list of the candidate switches they connect to the network manager. Figure 4 (a) shows the list of switches each VM will send to network manager.

Step 3: Network manager selects the minimum set of switches that cover all VMs. To explain this, let's assume a graph $G = (V, E)$ with links $l_i \geq 0$, where the objective is to find a minimum subset of switches that covers all VMs. For this, we apply the following condition to VMs

$$S_i = \begin{cases} 0 & \textit{if VM } v_s^i \textit{ is not covered} \\ 1 & \textit{if VM } v_s^i \textit{ is covered} \end{cases}$$

Objective function: minimize $\Sigma l\,S$ for all v_s

Figure 4 (b) is the final minimum set of switches required to form an AL for a cluster. These switches will be announced as an AL for a cluster such as S_1, S_2, and S_3 in Figure 4(b) and are discussed as OPS in this paper. These OPSs will be assigned with VC_{ID}. In routing the traffic, OPSs in the intra-cluster phase can be addressed with (S_{ID}, and IP address) and in inter-cluster phase as (S_{ID}, VC_{ID}, and IP address). Selecting a switch with maximum connections will reduce the number of switches in an AL and it will also help in aggregating the traffic. On the other hand, it may increase load on particular switches. Thus, there is a trade-off that needs to be considered for efficient architectures, however, this objective is not considered in this work.

Step 4: After selecting an AL, the remaining candidate switches will be discarded and they again start acting as ordinary switches of VN.

This procedure is repeated until every cluster has an AL.

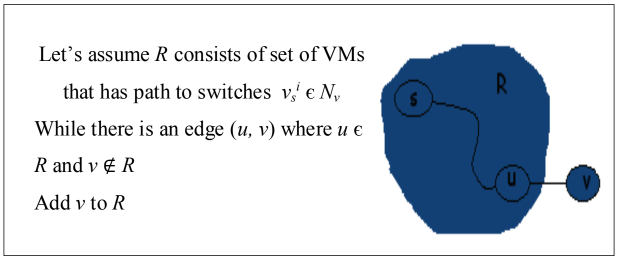
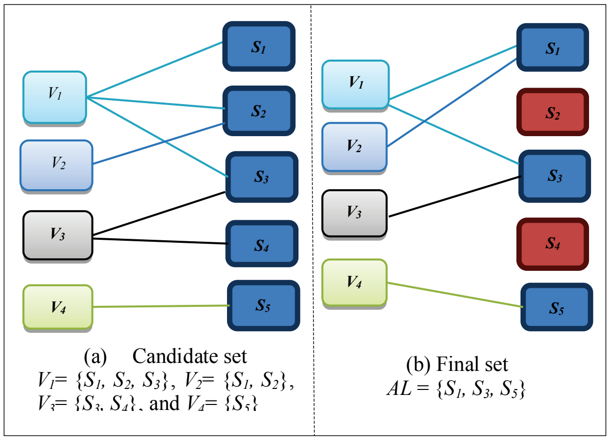

Let's assume R consists of set of VMs
that has path to switches $v_s^i \in N_v$
While there is an edge (u, v) where $u \in R$ and $v \notin R$
Add v to R

Figure 3. Switch selection criteria



(a) Candidate set
$V_1 = \{S_1, S_2, S_3\}$, $V_2 = \{S_1, S_2\}$, $V_3 = \{S_3, S_4\}$, and $V_4 = \{S_5\}$

(b) Final set
$AL = \{S_1, S_3, S_5\}$

Figure 4. Selection of an AL

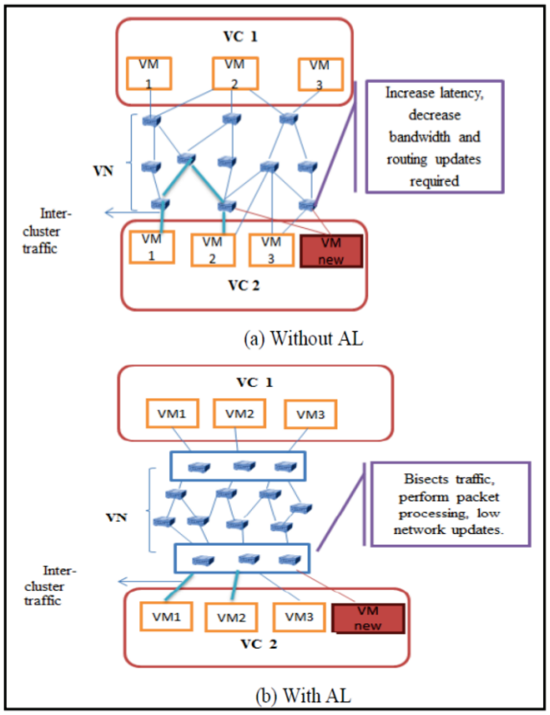
B. Features ALs Offer

It is depicted from the literature study [5], [12] that virtualization architecture should be capable to meet the required bandwidth, should be scalable to network changes, should manage traffic efficiently to preserve resources, and should use available resources efficiently to meet the future demands of the users. We claim that AL-VC is a potential architecture to meet these challenges.

In our architecture, an AL provides an abstraction to the clusters. Suppose we group VMs without an AL, then the traffic generated by the VMs is directly routed to the switches of VN. VN switches have to first convert electronic packets coming from TOR switches into optical packets and then route to the destination as shown in Figure 5. Switches near the destination VM have to convert optical packets again into electronics before sending to TOR switches. However, in our architecture, switches of an AL converts TOR packets into optical and then route towards the VN switches. It takes away the computation burden from VN switches and leaves them only for routing data. This allocates more bandwidth for the data. Moreover, due to an AL, we can bisect the traffic into intra-cluster and inter-cluster. When data arrives at an AL, it checks the destination device ID. If the destination machine belongs to its own cluster, AL sends data directly. If destination machine belongs to another cluster, AL will route the data towards the VN. This bisection of traffic provides shorter routes to intra-cluster traffic and let inter-cluster traffic use all the bandwidth of VN switches, which results in lower latency, and higher bandwidth as shown in Figure 5. Below we discuss more features our architecture offers:

Local management and control: Due to ALs, VMs in a cluster can be easily managed and modified without interrupting the operation of the rest of the network. For that, local decisions can be made without the involvement of an external entity.

Scalability: Introducing AL makes clusters quite flexible to network changes. The number of OPS in an AL can vary depending upon the resources of the network resources or the demands of the users. A cluster that has high


VC 1
VM 1
VM 2
VM 3
Increase latency, decrease bandwidth and routing updates required
VN
Inter-cluster traffic
VM 1
VM 2
VM 3
VM new
VC 2
VC 1
VM1
VM2
VM3
Bisects traffic, perform packet processing, low network updates.
VN
Inter-cluster traffic
VM1
VM2
VM3
VM new
VC 2

(a) Without AL

(b) With AL

Figure 5. Benefits of an AL

bandwidth demands might need more switches in its AL for faster processing.

Flexibility: AL based VCs are scalable to network expansion and flexible to network failures. In AL-VC, new machines can be added or deleted easily. In case of deletion or failure of machines, AL-VC can replace them with the new ones by local discovery mechanism.

Security: One of the assumptions of this architecture is that one OPS cannot be part of two ALs. OPSs of two different ALs will communicate via intermediate switches of VN. However, within an AL, they communicate directly to process the cluster data jointly. Avoiding direct communication of VMs helps in improving security. VMs can be attacked by intruders when connected to the Internet. Restricting their communication only via OPSs will hide their physical location, hence, will result in a better security.

Implement-ability: Unlike other proposed virtual architectures, AL-VC is implementable on any underlying physical topology of data centers such as on VL 2, B Cube, etc. It basically collects all the virtual resources in a pool and forms VCs according to the requirements of the ISPs, as shown in the Figure 6.

Meeting Application Criteria: VCs should be flexible enough to meet the changing demands of the users. Due to above features, we think, AL-VC is a potential architecture for this purpose. For example, the number of clusters, the number of VMs in a cluster, and the number of switches in an AL can be adjusted to provide the required bandwidth and latency.

All these features make AL-VC a standalone virtualization architecture that tends to exploit most of the features of network virtualization.

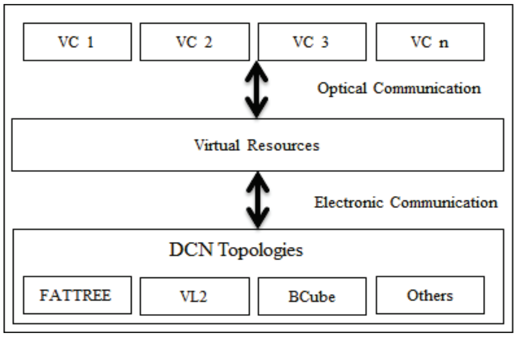


Figure 6. Implementability of AL-VC

IV. PERFORMANCE EVALUATION OF AL-VC

Though AL-VC offers several advantages, it is not possible to evaluate all of them in this work. In this work, we evaluated the scalability and efficiency in recovering from network failures such as VM and server failure. We use centralized architecture as the base scheme. For implementation, we select FATTree [15] as the underlying physical topology.

A. Failure of VMs

When a server detects the failure of a VM or when a VM is not replying to the control messages of its AL, VM is considered as failed. AL will inform all the VMs of its cluster about this failure. After this detection, AL will request the server that was hosting this failed VM to launch a new VM.

If sever does not have enough resources to host a new VM, it will send attributes of the failed VM to the AL. AL will request other servers that have the resources to host a new VM. Servers will send the attributes of candidate VM to AL. AL will select the VM of the server that has the closest attributes to the failed VM. Finally, the failed VM will be replaced with a new VM. The attributes of the requested VM can be represented as:

$$att_{N_v} = ((att_1,n_v{}^1),\ (att_2,n_v{}^2),.....,(att_n,\ n_v{}^n)) \qquad (2)$$

Non-Functional (NF) attributes of the two VMs can be calculated by the following dissimilarity metrics:

$$dism(i,j) = \frac{\sum_{r=1}^{l} \delta_{ij}^{r}\, dis_{ij}^{r}}{\sum_{r=1}^{l} \delta_{ij}^{r}} \qquad (3)$$

where:

l is the number of NF attributes

dis_{ij}^{r} denotes the dissimilarity of VM i and j related to att_l.

$\boldsymbol{\delta}_{ij}^{r}$ expresses the coefficients of the NF attributes of VMs i and j.

In Figures 7, 8, and 9, we evaluated the performance of AL-VC in detecting and replacing the failed VM/VMs. Centralized approach uses central fault detection and recovery. First, each server detects the failure and informs the central entity. For that, the central entity exchanges

TABLE I. SIMULATION ENVIRONMENT

Number of Servers	96
Number of VMs	360
Max VM a server can host	10
Number of switches in AL	10 % of VM in the cluster
Number of clusters	2, 4, 6, 8, and 10
DCN topology	FATREE
Parameters	Average time and Communication Cost

messages with all the participating servers to discover a new host. However, in our approach this procedure happens at local AL where AL takes the decision involving local machines. Therefore, AL requires less number of messages and less time to find new a VM to replace the failed VM, as can be seen in Figures 7 and 8.

The average time is the time required to detect a failed VM and replace it with a new VM. The communication cost is a measure of the number of messages required to replace these VMs. From Figures 7 and 8, we can see that an increased number of clusters decreases the average time and communication cost. This is because the number of participating entities in finding a new VM decreases. Increasing number of clusters helps in improving the performance of our approach. However, too many clusters in the network may result in overhead.

In Figure 9, we measure the time to replace multiple failed VMs. We can clearly see that when the number of failed VMs increases, the performance of the centralized approach deteriorates as the central entity has a lot of the workload and failure of multiple VMs can result in queuing delay at central entity. In case of AL-VC, each AL can run the VM discovery procedure locally to find the new VMs with less overhead.

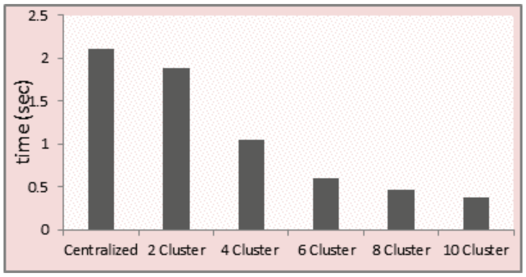


Figure 7. Average time required to replace failed VM.

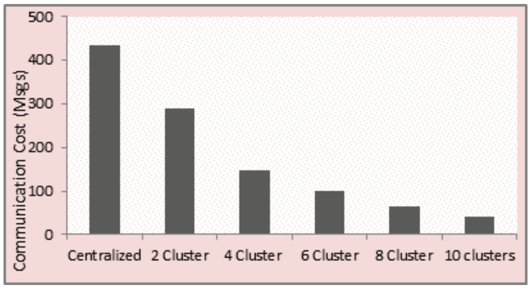


Figure 8. Communication cost required in replacing failed VM

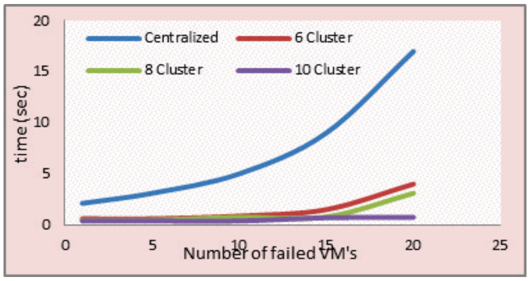


Figure 9. Average time required to replace failed VMs

B. *Failure of Servers*

When a server fails, all the VMs hosted by this server will also go down. When a VM does not respond to keep-alive messages, AL considers it as failed and contacts the hosting this VM. If server also does not respond, AL assumes that the servers has failed or has been removed from the cluster. AL informs to the network manager and asks for the attributes of the failed server and its VMs. After receiving NF attributes, it runs a local VM discovery algorithm to find new hosts for the VMs as explained before.

Note that failure of servers or VMs belonging to one cluster will not affect the operation of other clusters. In this evaluation, we assume that the failing server has three VMs that need to be relocated or replaced. From Figures 10 and 11, we can clearly see that AL-VC takes less time and less number of messages to replace these VMs. If the resources of the cluster are tight, network manager can search for a new server; if new server is available, it can replace the failed server with the new one by matching their nonfunctional attributes. Attributes of the requested server can be represented as following:

$$att_{Ns} = ((att_1, n_s^{1}), (att_2, n_s^{2}), \ldots.., (att_n, n_s^{n})) \qquad (4)$$

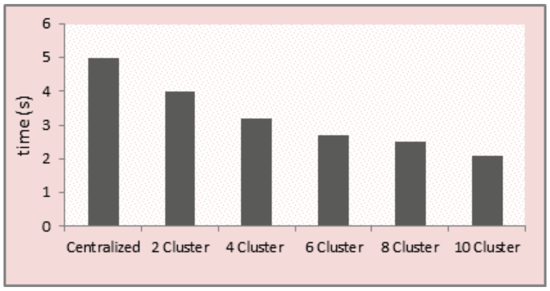


Figure 10. Average time to recover from a server failure

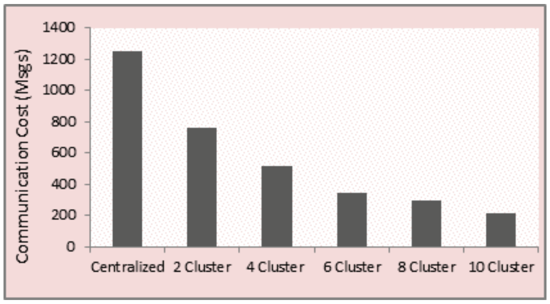


Figure 11. Communication cost required in recovering from a server failure

ISBN: 978-1-61208-388-9

These results prove that AL-VC is efficient in terms of time and cost. We conducted this evaluation in comparison with centralized approach; however, in extension of this work, we plan to evaluate our architecture with other distributed schemes as well.

V. CONCLUSIONS

Network virtualization is essential for the future Internet. It provides several features to the data centers. Existing virtual data center architectures are not capable to provide all these features. Therefore, in this paper, we proposed AL-VC that groups VMs into cluster based on service types. AL is the main feature of our architecture consisting of virtual switches of the VN. The introduction of AL helps in meeting most of the challenges that network virtualization envisioned such as scalability, flexibility, better control and management, and so on. In this work, we evaluated only its efficiency and scalability in the presence of failures. However, in the future, we plan to evaluate other parameters like bandwidth, latency, etc, as well.

ACKNOWLEDGEMENT

This work was supported by the National Institute of Information and Communications Technology (NICT), Japan.

REFERENCES

[1] WMware, “Virtualization Overview,” White paper, pp.1-11, 2006.

[2] T. Abels, P. Dhawan, and B. Chandrasekaran, “An Overview of Xen Virtualization,” Virtual. Tech., pp. 109-111, Aug. 2005.

[3] D. Stezenbach, M. Hartman, and K. Tutschku, “Parameters and Challenges for Virtual Network Embedding in the Future Internet,” Network Operations and Management Symposium (NOMS 2012) IEEE, Apr. 2012, pp. 1272-1278, ISSN: 1542-1201, ISBN: 978-1-4673-0267-8

[4] N.M.M.K Chowdhury, and R. Boutaba, “Network Virtualization: State of the art and Research Challenges,” Communications Magazine, IEEE, vol. 47, pp. 20-26, Jul, 2009, doi: 10.1109/MCOM.2009.5183468

[5] K. Tutschku, T. Zinner, A. Nakao and P. Tran-Gia, “Network Virtualization: Implementation Steps towards the Future Internet,” EASST [Online]. Available from: http://journal.ub.tu-berlin.de/eceasst/article/view/216/218

[6] M. F. Bari, R. Boutaba, R. Esteves, L. Z. Granville, M. Podlesny, M. G. Rabbani, Q. Zhang, and M.F. Zhani, “Data Center Network Virtualization: a Survey,” Communications Survey and Tutorioals, IEEE, vol. 15, pp. 909-928, Sep. 2013,doi: 10.1109/SURV.2012.090512.00043

[7] C. Guo, G. Lu, J. H. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y.Zhang, “SecondNet: a Data Center Network Virtualization Architecture with Bandwidth Guarantees,” The Sixth International Conference on Emerging Networking Experiments and Technologies (Co-Next 2010) ACM, Dec. 2010, pp. 1-14, doi: 10.1145/1921168.1921188

[8] T. Benson, A.A.A. Shaikh, and S. Sahu, “CloudNaaS: a Cloud Networking Platform for Enterprise Applications,” Second Symposium on Cloud Computing (SOCC’2011). ACM, Oct. 2011, doi: 10.1145/2038916.2038924

[9] M. Chowdhury, F.Samuel, and R. Boutaba, “PolyViNE: Policy-based Virtual Network Embedding across multiple Domains,” The Second Sigcomm workshop on Virtualized Infrastructure Systems and Architectures (VISA 2010) ACM, Sept.2010, pp. 49-56, doi: 10.1145/1851399.1851408

[10] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, “Adaptive Virtual Network Provisioning,” The Second Sigcomm workshop on Virtualized Infrastructure Systems and Architectures (VISA 2010) ACM, Sept.2010, pp. 41-48, doi: 10.1145/1851399.1851407

[11] I. Houidi, W. Louati, and D. Zeghlache, “A Distributed Virtual Network Mapping Algorithm,” The Eigth International Conference on Communications (ICC 2008) IEEE, May. 2008, pp. 5634–5640, doi: 10.1109/ICC.2008.1056

[12] Y. Zhang, A.J. Su, and G. Jiang, “Evaluating the Impact of Data Center Network Architectures on Application Performance in Virtualized Environments,” The Eigteenth International Workshop on Qualiting of Service (IWQoS 2010)IEEE, Jun. 2010, pp. 1-5, doi: 10.1109/IWQoS.2010.5542728

[13] A. Fischer, J.F. Botero, M.T. Beck, H.D. Meer, and X. Hesselbach, “Virtual Network Embedding: a Survey,” Communication Surveys and Tutorials, IEEE, vol. 15, pp. 1888-1906, Nov. 2013, doi: 10.1109/SURV.2013.013013.00155

[14] Y. Ohsita, and M. Murata, “Data Center Network Topologies using Optical Packet Switches,” The Thirty Second Distributed Computing System Workshop (ICDCSW 2012) IEEE, Jun. 2012, pp. 57-64, doi: 10.1109/ICDCSW.2012.53

[15] M. A. Fares, A. Loukissas, and A. Vahdat, “A Scalable, Commodity Data Center Network Architecture,” Conference on Data Communication (SIGCOMM 2008) ACM, Oct. 2008, pp. 63-74, doi: 10.1145/1402958.1402967

[16] N. Farrington and A. Andreyev, “Facebook’s Data Center Network Architecture,” Facebook, Inc [Online]. Last access: Mar, 2015. Available from: http://nathanfarrington.com/papers/facebook-oic13.pdf .

[17] K. Chen, A. Singh, and K. Ramachandran, “OSA: an Optical Switching Architecture for Data Center Networks with Unprecedented Flexibility,” Networking Transactions, IEEE/ACM, vol. 22, pp. 498-511, Apr.2014, doi: 10.1109/TNET.2013.2253120