Software Defined Network based Adaptive Routing for Data Replication in Data Centers

[1]Renuga Kanagavelu, [2]Bu Sung Lee
[1]A*STAR(Agency for Science and Technology),
Data Storage Institute,Singapore,
Renuga_k@dsi.a-star.edu.sg

[1]Rodel Felipe Miguel, [1]Le Nguyen The Dat, and
[1]Luke Ng Mingjie
[2]School of Computer Engineering
Nanyang Technological University, Singapore
ebslee@ntu.edu.sg

Abstract— **Data availability is a major challenge faced by today's Data Centers where a large number of high performance servers are organized into racks interconnected by a switching network. Data replication is an effective approach for data protection as redundancy ensures at least one copy of data is available in the event of failures. To achieve low latency and high throughput for data replication operations, the traffic load must be spread out evenly across various paths in a Data Center network to minimize congestion. Recently, software defined networking (SDN) has become an attractive solution which enables us to control traffic forwarding in a desired way to achieve our goal. We develop an SDN-based framework for effective data replication operations. We develop an adaptive routing scheme which routes the flows based on the current network state; specifically it chooses routes based on the current load on the paths. Our scheme monitors link- and path- loads and assigns traffic flows to appropriate paths in an efficient way so as to achieve high throughput. We develop a prototype using OpenFlow-enabled switches and carry out experiments to demonstrate the effectiveness of our proposed scheme.**

Keywords; Software defined networking, Data Center network, data replication, openflow

I. INTRODUCTION

In recent years, Data volume has been increasing at rapid speed which drives the creation of large Data Centers hosting a broad range of services such as Web search, e-commerce, storage backup, large scientific applications, video streaming, data analytics and social networking. In such large Data Centers there are tens of thousands of servers spread over hundreds of racks with tens of peta bytes of storage interconnected by high capacity network infrastructure. Failures have severe impact in a large center with huge number of server- and network- elements and massive data storage. Therefore, it is critically important to develop solutions to ensure high availability of resources.

Data availability is a major challenge faced by today's Data Centers. As data availability becomes a critically important requirement, many businesses use increased amount of resources to ensure continuous operations. Maintaining uninterruptible access to all Data Center applications is highly desirable. As a result, Data Centers need a range of business continuance solutions [1], from simple tape backup and remote replication to synchronous mirroring and mirrored distributed Data Centers. Proactive replication is a key strategy coupled with mirroring for data protection. Data replication is an effective approach [2, 3, 4, 5, 6] for achieving high data availability and durability in Data Centers. Data replication is a technique designed for replicating data at two or more storage nodes attached to different racks in a Data Center. Such redundancy ensures at least one copy of data is available for continuous operation in the event of a rack switch failure or rack power failure. However, the design choice of data replication is complicated by keeping the copies as closely synchronized as possible and using as little network bandwidth as possible. Synchronous update of all copies provides high resilience to data loss but has poor write performance and results in high network cost.

Network bandwidth and latency are the two limiting factors for data replication. If enough network bandwidth is not available, the messages are queued up in the network buffers and consequently, read and write operations to remote physical storage volume takes longer time, leading to longer response time. We note that long response time might be acceptable for batch applications but not for critical applications like online transactions. The latency increases if the number of switches/routers/links on the communication path of the messages increases and also when a switch is congested. We note that remote mirroring demands ultra-low latency and high throughput.

Current Data Center networks are built hierarchically [7], with servers directly connected to a rack-level switch (RS). Rack-level switches are in turn connected to a layer of Aggregate switches (AS) through a number of network links. Conventional spanning tree architecture is not used in Data Centers as it limits any pair of end points to use only a single communication path, leading to congestion along the single path and underutilizing the resources in other paths. As a result, a number of recent efforts were targeted toward deploying more efficient Data Center networks using multi-routed tree architectures [8, 9, 10, 11 and 12]. In such Data Centers, due to the use of large number of commodity switches, multiple paths exist between a given pair of hosts leading to multi-rooted tree architectures [9]. In order to guarantee short latency for the most demanding applications such as disk mirroring and social networks which are replicated on multiple servers for greater

capacity and reliability, the traffic load must be spread out evenly across multiple paths to minimize network congestion.

Recently, software-defined networking (SDN) architectures in which the control plane is decoupled from data plane are becoming popular as users can intelligently control routing and resource usage mechanism. An essential element of SDN is that it explicitly links network control to each application's functional requirements. An SDN-enabled switch (e.g., OpenFlow [13]) forwards traffic in the data plane based on the control plane rules which is running on a separate controller [14]. SDN enables us to manage the traffic flows dynamically which facilitates high bandwidth, low latency storage data replication among data centers.

Data replication has been widely adopted by key players like Google, Yahoo, and Netflix. For example, Google's e-mail service Gmail synchronously replicates across five data centers to sustain two data center outages: planned and unplanned. An online social network (OSN), for example, Facebook employs full replication strategy wherein each geo-distributed Data Center maintains one copy of all the data. Replication across data centers, however, is expensive. Inter-data center network delays are in the order of hundreds of milliseconds and vary significantly.

In our work, we consider data replication within a Data Center to protect against server/switch/rack failures. Even in the case of replication across Data Centers, keeping one of the copies in the same Data Center and the remaining copies across Data Centers has the advantage of lower latency. Thus, our focus on intra-Data center replication becomes significant. A key challenge in replicating data across racks in a Data Center is network congestion. With the use of SDN we can effectively control the routes of the flows to ensure reduced congestion and low latency.

We develop an SDN-based framework for effective data replication operations. We develop an adaptive routing scheme which routes the flows through the least loaded paths based on the current network state. We develop a prototype using OpenFlow-enabled switches and carry out experiments to demonstrate the effectiveness of our proposed scheme.

The rest of the paper is organized as follows. Section II presents the background and related work on data replication and software defined networks. Section III discusses the proposed SDN-based data replication framework, the adaptive routing algorithm, the various functional blocks, and the experimental testbed. Section IV studies the performance of the proposed mechanism and discusses the OpenFlow testbed experimental results. We conclude the paper in Section V.

II. BACKGROUND AND RELATED WORK

In this section, we will first present the available Data replication strategies. We will then explain the concept of Software-defined Networking (SDN) and OpenFlow protocol.

A. Data Replication

Data Replication is widely used in Data Centers to ensure that in the event of a rack failure due to power or cooling problems and others, another copy of data is readily available to ensure continued operation. Companies such as IBM, EMC, Veritas, etc., all provide their own proprietary solutions for remote mirroring [15, 16, 17, 19]. EMC's symmetrix Remote data facility (EMC_SRDF)[15] provides inter-array remote mirroring. A good summary of various remote mirroring approaches can be found in [15] including a new asynchronous remote mirroring protocol called Seneca. The iSCSI based mirroring has been presented in [18]. All these solutions require low latency IP connections to do mirroring. SnapMirror[5] deals with execution performance as network latency and traffic becomes an issue under high load replication environment. The above works primarily focusses on replication across data centers. On the other hand, as we explained earlier, we focus on data replication within data centers and develop an SDN-based solution.

Recently, there has been some research work done in the literature related to the load balancing problem in Data Center networks [11, 22]. Hedera's [22] flow scheduling algorithm is based on the assumption that it only needs to schedule long-lived flows because they contribute most of the bytes. Other flows are considered as background noise. It schedules a flow onto a link that can accommodate the flow. This method is not suitable for end-host limited flows. For example, network bandwidth can exceed disk performance for many workloads. End Host-limited flows can be long lived and transfer a great amount of data, but never exceed the scheduling threshold. The Hedera scheduler ignores these flows and they can collide with scheduled flows. As a result, the overall throughput will drop.

B. Software Defined networking and Openflow

Recently, software-defined networking (SDN) architectures in which the control plane is decoupled from data plane have become popular as users can intelligently control routing and resource usage mechanism. This separation opens up the control flow to be easily managed and remotely accessed. A separate controller is used for the purpose of routing control, which contains the knowledge of switches, routing information and network status. With the centralized controller, SDN manages routing algorithms separately while keeping data flows running on original network paths. Hence, SDN provides simple and flexible routing control which can adapt to the changing networking state.

OpenFlow protocol is the most popular enabler for SDN. Every OpenFlow switch maintains its own table of flow entries (flow table), in which each entry contains a set of packet fields to match, and the corresponding action to perform, (e.g. forward, drop, modify header). In the event when a switch is unable to find a match in the flow table, the packet is forwarded to the controller to make the routing decisions. After deciding how to route the new flow, the controller then installs a new flow entry at the required switches, so that the desired actions can be performed on the

new flow. Installing an OpenFlow firmware gives engineers access to flow tables and set rules for routing traffic. With OpenFlow, it is possible to use various routing algorithms within the controller to generate the forwarding tables that govern adaptive routing of flows in the data plane.

III. PROPOSED SDN-BASED DATA REPLICATION FRAMEWORK

We consider a multi-routed Data Center network as shown in Figure 1. The network is comprised of switches interconnected in three layers- Top of Rack (ToR), aggregate, and core- as shown in the figure. Also shown in the figure are the pods, replication host (which initiates the replication operation), the primary site where the primary copy of data is stored in a server and backup site where the backup copy of the data is stored in a server. A replication operation generates two flows- one flow between the host and primary server and the other flow between the host and the backup server.

Adaptive Routing Method

We develop an adaptive routing method which is a key component in the data replication framework. We select appropriate routes for the flows based on the current network state. We monitor the load statistics at the switches and based on the loads on the paths we choose the minimum cost paths. In our method, we choose the least-loaded path which is the path with minimum load among all candidate paths. We use the shortest-hop paths as candidate paths. We note that there exist many shortest-hop paths in the Data Center network. The cost of a path is calculated as the sum of the utilization of the links along the path. We choose the path that has the minimum cost calculated as above. If there is more than one path with the same minimum cost, the path with maximum residual capacity is preferred. By taking the load into consideration, we try to balance the load and reduce network congestion which leads to high throughput.

Functional Modules

We implemented our proposed data replication framework on an SDN-based Data replication Testbed topology as shown in Figure 2. To deploy SDN based remote replication, we set up the networks, namely, control network (CN) and network under test (NUT). The CN will be used as the OpenFlow channel, i.e. the network used for management of the OpenFlow switches. The NUT will be used as the OpenFlow data path where the storage network will be deployed and tested. We have used NOX controller [14] as the base controller for the HP series OpenFlow switches which runs our adaptive load-balancing algorithm. The details of the system components and functional modules in the proposed data replication framework and the testbed are shown below.

In order to test the performance of data replication using our adaptive routing algorithm, we use two storage systems namely 'Primary server 'and 'Back-up server' from different network paths of the NUT to the Data replication host. To achieve this, we configured the data replication host as an iSCSI initiator and the two storage servers connected to different network paths as iSCSI targets. Each remote site exports storage as a backing store for the iSCSI target. We use IOmeter [21] as the work load generator.

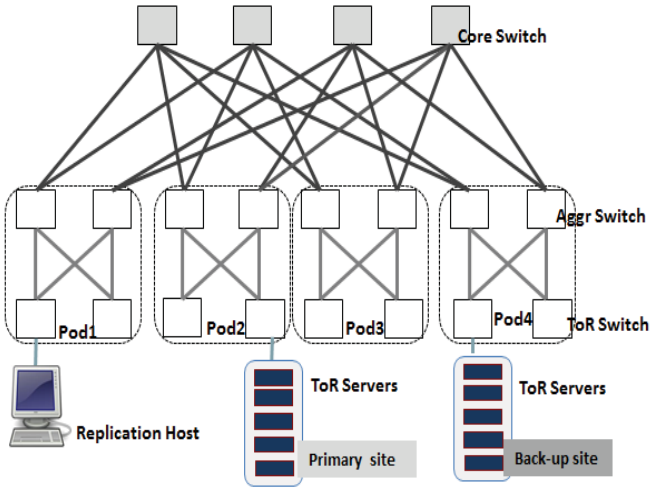

Figure 1. SDN based Data Replication Architecture

A. Control Network

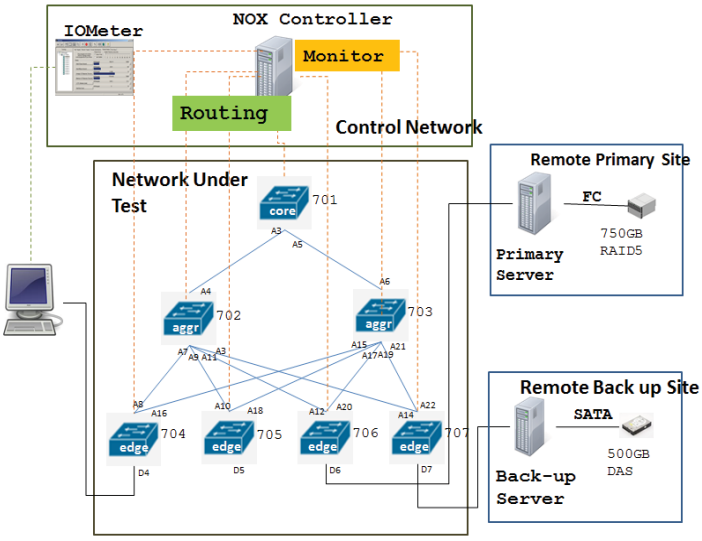

Figure 2. SDN based Data Replication Testbed

The Centralized NOX controller has the key functional modules namely, (i) Monitor and (ii) Routing functional modules. The key idea behind the algorithm is to route packets along the least loaded path. The packet counts and flow count information from the Monitor module together with capacity of

each link along the paths will be used to compute the least loaded path.

We briefly describe below the functions of each module in our adaptive routing framework.

1. Monitor Module

This component collects, stores, and queries statistics from all OpenFlow switches that will be used by the Routing Component to compute and compare the load on various links. These statistics are polled at fixed intervals and stored in NOX Controller as snapshot objects. Every snapshot will be assigned a sequence number, which increments by one after each interval. The two most recent snapshots will be maintained in memory at any instant of time. Other components can access these snapshot objects directly to obtain the required information.

2. Routing Module

This component is responsible for routing functions. Least loaded path with highest residual capacity for a flow is computed based on the most recent load statistics collected by the Monitor module.

When the first packet from a new flow arrives at an OpenFlow switch, the switch forwards the packet to the controller since there is no flow entry matched. The controller will extract the flow's source and destination host addresses for the Routing Module to compute the least loaded path.

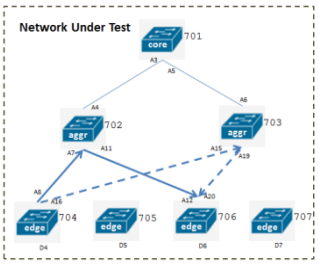

Figure 3. Shortest Routes from Host to Primary server

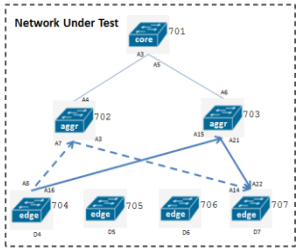

Figure 4. Shortest Routes from Host to Back-up server

We shall consider the case of a data replication operation from the Host to primary server and Backup server as shown in Figure 2 as an example to illustrate the routing mechanism. Let $R_{1,2}$ represents the set of candidate shortest paths from Host to Primary Server as shown in Figure 3: $\{r_1, r_2\}$ where r_1 is 704-702-706 and r_2 is 704-703-706. Let $R_{1,3}$ represents the set of candidate shortest paths from Host to Back-up server as shown in Figure 4: $\{r_3, r_4\}$ where r_3 is 704-702-707 and r_4 is 704-703-707. Based on the snapshots collected by the Monitor Module in the recent interval we calculate the load P measured in bits per second. If the capacity of a link is B bits per second, the utilization of a link is calculated as P/B. The cost of a route is calculated as the sum of the utilization of the links along the path. After computing *route cost*$_i$ $\forall$ r_i $\square$ R, the controller picks the route that has the minimum *route cost*$_i$. If there are more than one route with the minimum *route cost*$_i$, route with maximum residual capacity will be chosen.

B. Network Under Test

In order to measure the SDN-based data replication performance, we used a hardware switch [20] (HP ProCurve E5412-96G zl Switch with K.15.05.5001 firmware) for NUT. We chose this switch because it is a large switch with OpenFlow support and that all the ports are 1GigE which matches all the ports on the servers that we used. We have configured the switch to support the FAT tree Data Center architecture. To achieve this, we have assigned groups of 1GigE ports to 7 sets of VLANs controlled by a single instance of our forwarding algorithm running on top of the NOX OpenFlow controller. We used VLAN numbers 701 to 707 to emulate the switches of the FAT Tree topology.

As shown in Figure 2, VLAN 701 represents the core switch of the Data Center. VLAN 702 and 703 represent the aggregate switches. VLAN 704, 705, 706, and 707 represent the access switches where the data center storage servers are connected. We have used Ethernet cables to inter-connect the VLANs to form the FAT tree architecture.

We briefly describe the individual components of the network architecture:

1. Primary Sever

This is the primary storage site for replication. This host acts as iSCSI target 1. It is connected an external RAID5 storage system with 750GB capacity. We have created a logical volume of 250GB from this physical storage to be used as the backing store of this iSCSI target host.

2. Backup Server

This is the secondary storage site for replication. This host acts as iSCSI target 2. It has a direct attached storage as the backing store of this iSCSI target host. We have created a logical volume of 250GB from the direct attached storage to match the backing store that was set up in Host2.

3. Replication Host

This host is the data replication host with iSCSI initiator configuration. This host has two Ethernet ports which are connected to both the CN and the NUT. After starting the iSCSI initiator service, and discovering the exported storage by the remote sites, this host will be able to detect two new storage devices (e.g. /dev/sdb and /dev/sdc) available for use.

IV. PERFORMANCE STUDY

We run a benchmark called "IOmeter"[21], which is an industry standard I/O benchmark tool, to test the data

replication performance. IOmeter can generate workloads of various characteristics including request size, read/write ratio, queue size and maximum number of outstanding requests. The IOmeter will create one worker thread for each CPU core. For each block size we ran several read, write, and read/write tests. We set up a testbed using a HP OpenFlow switch, 3 hosts and an OpenFlow controller, with the topology as shown in Figure 2. The hardware configurations of our experiments are shown in Table 1 and Table 2.

TABLE 1: .SWITCH CONFIGURATION

Manufacturer	Hewlett Packard
Model	HP E5412-96G Zl
Firmware	K.15.05.5001
#of GE ports	96

TABLE 2: NOX, HOST, SERVER CONFIGURATION

S.No.	Host	Primary Server	Back-up server	NOX controller
Operating System	Fedora 16 (Linux 3.3.8-1.fc16.i686)	Fedora 16 (Linux 3.4.4-4.fc16.i686)	Fedora 16 (Linux 3.4.4-4.fc16.i686)	Ubuntu10.04 (Linux 2.6.32.33-Server)
Memory	1.5GiB	3.4GiB	2.2GiB	3GiB
Processor	AMD athlon 64 X2 Dual core $200+	Intel xeon CPU W3530@2.5GHz	Intel xeon CPU W5630@2.5GHz	Intel Core2 CPU 6400@2.13GHz
#of Processor	2 cores	8 cores	8 cores	2 cores

A. *Effectiveness of proposed SDN based adaptive routing*

In this section, we evaluate the effectiveness of our proposed SDN based adaptive routing mechanism by comparing it with the traditional iSCSI based approach for read and write operations. We use synchronous operations wherein the operations on primary and backup copies are performed simultaneously. Fig.5 shows the throughput achieved by our method and iSCSI method for various request sizes. We can observe that our method achieves higher throughput demonstrating the effectiveness of adaptive routing.

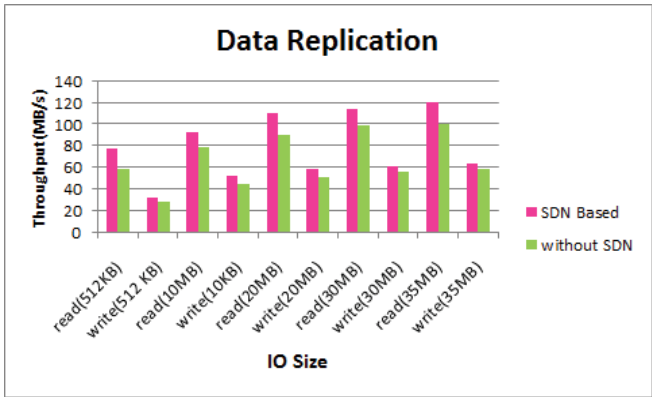

Figure 5. Performance of SDN vs. iSCSI based schemes

B. *Performance for different Queue size*

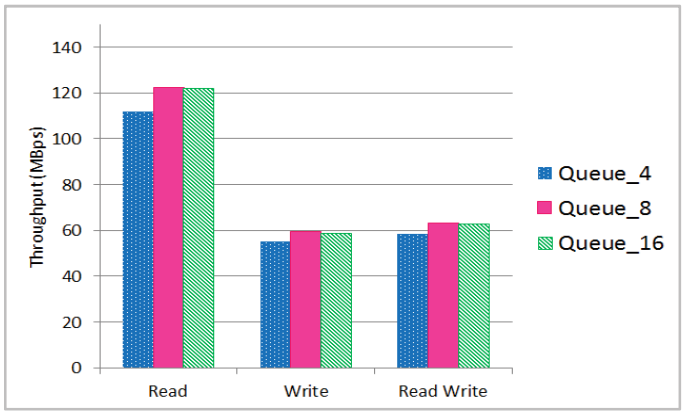

Figure 6. Performance for different queue Size

Figure 6 shows the impact of the queue length on synchronous replication between Primary server and Backup server. Since queue length is an important parameter that affects small I/O, we measure the performance with varying queue lengths. From the figure, we can see that throughput increases with the queue length until the queue length equals to 8. Further analyzing the captured data , we find that the maximum effective command queue length is 8 in our prototype and experiments.

C. *Maximum core switch utilization*

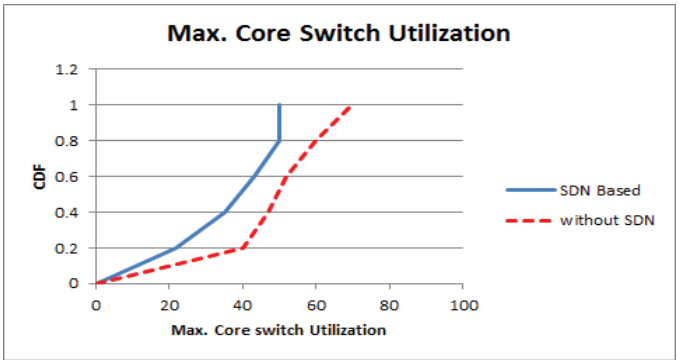

Figure 7. Maximum Core switch Utilization

Figure 7 shows the Maximum core switch utilization in the OpenFlow testbed. We consider maximum core utilization is the utilization of the most congested links connecting core switches, which is often a good estimate of the network bisection bandwidth. It can be observed that using our algorithm performs better. This shows the effectiveness of our algorithm in balancing the load.

D. Performance Analysis from Primary Server

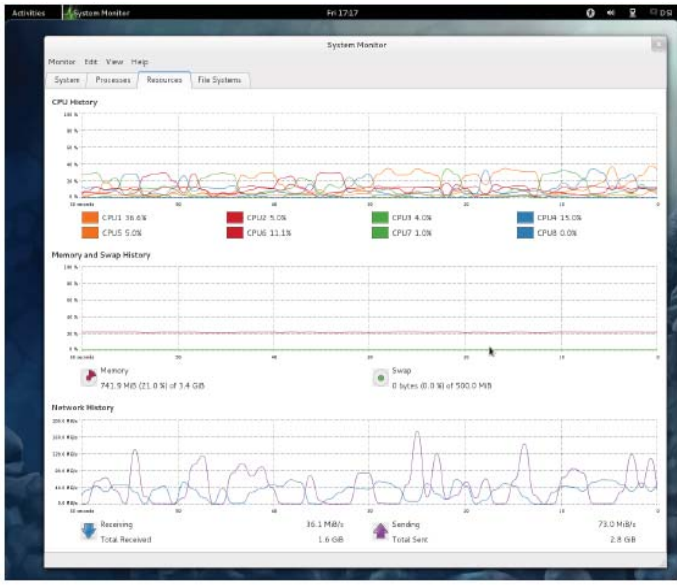

Figure 8. Screen shot from the Primary server during Replication

Figure 8 shows the snapshot taken at the Primary server. It shows the throughput measured in terms of million Bytes per second (MiB/s) for the data bytes sent and received. We observed that the throughput measured is reasonably high.

V. CONCLUSIONS

In this paper we addressed the problem of data replication in Data Center networks. We developed an SDN-based framework which assigns the routes for the flows (related to the replication operations) considering the network loads. It monitors the loads at switches and assigns traffic flows to appropriate paths in an efficient way so as to reduce congestion leading to high throughput. It also ensures even distribution of load across switches. We also developed an SDN-based prototype using the OpenFlow-enabled switches and NOX controller. We carried out experiments on the testbed and demonstrated the effectiveness of our proposed scheme by comparing it with the traditional iSCSI-based approach.

REFERENCES

[1] ORACLE whitepapaer, *Data Protection Strategies in today's Data Center*, 2012.

[2] Z. Zhao, T. Qin, F. Xu, R. Cao, and Xiaoguang Li GangWang, *CAWRM:A Remote Mirroring System Based on AoDI Volume*, In Proceedings of IEEE conference on Dependable Systems and Networks Workshops,2001.

[3] H. Weatherspoon, Lakshmi.G, T. Marian, Mahesh.B. , and K.Birman, *Smoke and Mirrors: Reflecting Files at a Geographically Remote Location without Loss of Performance*, In Proceedings of the 7th USENIX Conference on File and Storage Technologies, FAST 2009, USA.

[4] M. Ji, A. Veitch, J. Wilkes, *Seneca: remote mirroring done write.* In Proceedings of USENIX Annual Technical conference, General Track, USA,2003.

[5] H. Patterson, S. Manley, M. Federwisch, D. Hitz, S. Kleiman, S. Owara, SnapMirror: *File System Based Asynchronous Mirroring for Disaster Recovery*, In Proceedings of USENIX Conference on File and Storage Technologies, USA, 2002.

[6] Dot Hill Corporation, Tech. Rep, *Secure Data Protection With Dot Hills Batch Remote Replication*,2009.

[7] L.A. Barroso, U. Hlzle, *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*, Synthesis Lectures on Computer Architecture, 2009

[8] H. Abu-Libdeh, P. Costa, A. Rowstron, G. OShea, A. Donnelly, *Symbiotic Routing in Future Data Centers*, In Proceedings of SIGCOMM 2010.

[9] M. Al-Fares, A. Loukissas, and Alexander .L. *A Scalable, Commodity Data Center Network Architecture*, In Proceedings of SIGCOMM 2008.

[10] N.Farrington, G. Porter, Sivasankar R., H.H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat, *A hybrid electrical/optical switch architecture for modular data centers,* In Proceedings of SIGCOMM 2010.

[11] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S.Sengupta,*VL2: A Scalable and Flexible Data Center Network* , In Proceedings of SIGCOMM 2009.

[12] C Guo, G Lu, Dan Li, H Wu, Xuan Z, Y Shi, C Tian, Y Zhang, and Songwu Lu, BCube: *A High Performance, Server centric Network Architecture for Modular Data Centers*, In Proceedings of SIGCOMM 2009.

[13] N.McKeown, T.Anderson, Hari Balakrishnan, G. Parulkar, L. Peterson, Jennifer Rexford, Scott Shenker, and J. Turner, *OpenFlow: enabling innovation in campus networks*, In Proceedings of SIGCOMM 2008.

[14] A. Tavakoli, M. Casado, T. Koponen, *Applying NOX to the Datacenter*, In Proceedings of 8th ACM Workshop on Hot Topics in Networks,2009.

[15] EMC Corporation, Tech. Rep. EMC SRDF: *Zero Data Loss Solutions for Extended Distance Replication*, 2009.

[16] Symantec Corporation, *VERITAS Volume Replicator,* USA, Tech. Rep. 249505, 2002.

[17] IBM ,Tech Report, *Enhanced Remote Mirroring*,2009

[18] Ming Zhang, Yinan Liu, and Qing (Ken) Yang, *Cost-Effective Remote Mirroring Using the iSCSI Protocol*, In Proceeding of IEEE Mass Storage Systems, 2004

[19] Herman Mmutiso, *Multi-node mirrored NVRAM in a virtualized environment*,2011

[20] Http://www.openflow.org. Configuring HP Procurve.

[21] Http://www.iometer.org. IOMeter Users Guide. Version 2003.12.16

[22] Mohammad Al-Fares, Sivasankar Radhakrishnan, Barath Raghavan, Nelson Huang, and Amin Vahdat, "*Hedera: Dynamic Flow Scheduling for Data Center Networks*", in Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI '10), April 2010.