The SDN Controller Placement Problem for WAN

Peng Xiao[*, †], Wenyu Qu[*], Heng Qi [‡], Zhiyang Li[*], Yujie Xu[*]
[*]College of Information Science and Technology, Dalian Maritime University, Dalian 116026, China
Email: wenyu@dlmu.edu.cn
[†]School of Information Science and Engineering, Dalian Polytechnic University, Dalian 116034, China
[‡]School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

***Abstract*— Large networks are always partitioned into several small networks when deploying software defined networking (SDN). The network partitioning with SDN control planes open many unanswered questions such as latency, reliability, and load balancing. This paper opens the investigation by focusing on two specific questions: given a wide-area network topology, how to partition it into several small SDN domains, and where should the controller go in each SDN domain? To answer these questions, we propose a novel approach to efficiently and accurately evaluate SDN controller placement problem for WAN. This approach uses the Spectral Clustering placement algorithm to partition a large network into several small SDN domains. After presenting our metrics of the SDN controller placement problem, we evaluate our approach using the Internet2 topology and other available WAN topologies. For testing purposes we developed a new framework to test the SDN controller in WAN. The results show its effectiveness for the SDN controller placement problem.**

Keywords—SDN; Controller Placement; Spectral Clustering;

I. INTRODUCTION

When deploying SDN in real networks, large networks are always partitioned into several smaller ones due to numerous reasons: privacy, scalability, incremental deployment, security and so on [1, 4]. For SDN partitioning, a large network is likely to be divided into multiple SDN domains. Each SDN domain runs one controller such as Floodlight, NOX, and Beacon [9] etc. A SDN domain can be a sub-network in a data center (DC), an enterprise network or an Autonomous System (AS), but in this paper SDN domain is an AS which is partitioned from WAN. We consider only wide-area networks where the "best" controller placement minimizes propagation delays and improves reliability.

The most relevant work can be found in [2]. The authors motivated the controller placement problem and examined the impacts of placements on average-latency and worst-case latencies on real topologies. However, they treat WAN as a whole rather than multiple SDN domains and ignore the reliability of each controller. While propagation latency is certainly a significant design metric, we argue that reliability and load balancing design is also an essential part for operational SDN. They assume that nodes are always assigned to their nearest controller using latency as metric. In the average-latency placement, the number of nodes per controller is imbalanced and ranges from 3 to 13 when the number of controllers k=4 as Figure 1 shows. The more nodes a controller has to control, the heavier is the load on that controller. From Figure 1, we can see the imbalance between the controller 2 and controller 3.

With regarding to the controller failure tolerance, Hock et al. [10] optimize the placement of controllers called POCO. Their proposed method can be used to implement a scalable and reliable SDN control plane. However, they also treat WAN as a whole rather than multiple SDN domains and the controllers work together. On one hand, this method requires frequent queries to the individual devices in order to be accurate for a desired interval, which is bound to affect the inter-controller latency. With the expanding of WAN, the inter-controller broadcast storm grows sharply. On the other hand, a resilient controller placement often reassigns nodes to their new controller as required. The latencies of the reassigned nodes to their new controller can be significantly higher than the latencies to the primary controller.

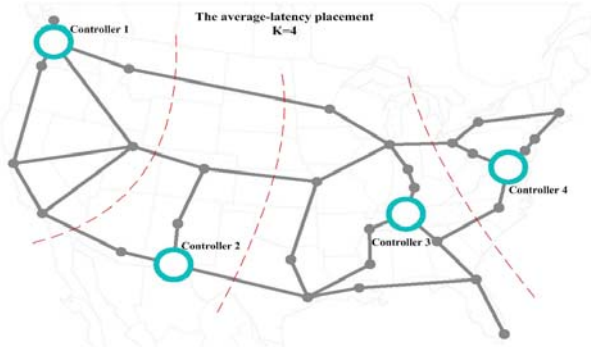

Fig. 1 Four partitions based on the average-latency placement

These works mentioned in [2, 10] assume that the mapping between a switch and a controller is configured dynamically, as ElastiCon [11] has been proposed an elastic distributed SDN controller. The dynamic allocation can improve the scalability and reliability of the SDN deployed in LAN, but it is not suitable for WAN. Usually, the propagation latency is longer than the queuing delay in network, and the dynamic mapping between a switch and a remote controller will significantly affect the response time of WAN. Moreover, the switch migrations are also complex works and need more overhead. The dynamic allocation strategy is suitable for the architecture of LAN but not WAN.

Motivated by these analyses, we propose a method to partition WAN into several SDN domains, each with their own controller, similar to the Domain Name System (DNS). In this paper, we assume that the communication between controllers have been solved perfectly. H. Yin et al. [3] and Pingping Lin et al. [4] had proposed two solutions to exchange control and application information across multiple SDN domains. Our work focuses on using the spectral clustering method to partition a WAN topology into several small SDN domains,

978-1-4799-4146-9/14/$31.00 ©2014 IEEE

and how to place the controller with low latency and high reliability in each SDN domain.

The major contributions of this paper are:

(1) We propose that using the spectral clustering to partition SDN domains when deploying SDN in WAN.

(2) We define the metrics of the SDN controller placement problem regarding not only latency, but also load balancing and reliability.

(3) The performance evaluations show that our approach can significantly improve the reliability of SDN control networks.

The rest of the paper is organized as follows. We discuss how to balance the load among controllers, how to account for inter-controller latencies, and define the metrics in Section Ⅱ. Section Ⅲ introduces our spectral clustering algorithm and describes the design of our testing methodology. Our work is implemented and evaluated in Section Ⅳ. Finally, we conclude in Section Ⅴ.

II. PROBLEM DESCRIPTION AND PLACEMENT METRICS

In this section we briefly introduce the definitions of SDN domains partition for WAN, and discuss the optimization placement metrics we are going to study, which are NP-hard problems with an input for k, the number of controllers to place.

A. Problem Description

WAN is a network that covers a broad area including many regions or countries. With the deployment of SDN in WAN, the SDN domains partition problem of WAN has been exposed. For a network graph G(V,E,W) where the node set V represents nodes in network topology which means OpenFlow switches, the edge set E represents propagation latencies and the weight matrix W represents the network bandwidths of the links. We assume that the bandwidths are same ($W_{ij}=1$) and partition G into k subgraphs, namely SDN_i. The SDN_i can be defined as $SDN_i(V_i,E_i,W_i)$, where $i=1,2,\ldots,k;\bigcup_{i=1}^{k}V_i=V;\bigcup_{i=1}^{k}E_i=E;\bigcup_{i=1}^{k}W_i=W$.

As a new deployment in WAN, the partition choice influences every aspect of the SDN network in WAN, from performance to security. In this paper, we want to find a partition of the topology such that the edges (which means defined metrics) between different groups have a very low weight (which means that OpenFlow switches in different clusters are dissimilar from each other) and the edges within a group have a high weight (which means that OpenFlow switches within the same cluster are similar to each other). Furthermore, controller placement must be placed in the clustering center to ensure maximum the performance.

B. Placement Metrics

In the paper, we only consider WAN where the "best" controller placement minimizes propagation delays, balances the load, and ensures the reliability of controller. In long-propagation-delay WAN, propagation latency is certainly a significant design metric. Considering the sheer scale of WAN, we decide to use the divide-and-rule philosophy to solve the performance problem of controller. For WAN, the best placement depends on propagation latency and the controller performance. By partitioning WAN into several smaller SDN domains, the service ability of controller with less nodes will be improved greatly, and the inter-controller broadcast storm will be reduced sharply.

What is a sound principle well established in WAN partition? We wish to get some balanced cuts which minimize similarity between sets and maximize similarity within a set. For example, if partition G into two subgraphs A and B，the similarity or association between two nodes x and y is their edge weight W_{xy}. Thus the similarity between A and B is the cutsize:

$$cut(A,B)=W(A,B) \tag{1}$$

where

$$W(A,B)=\sum_{x\in A,y\in B}W_{xy},W(A)\equiv W(A,A). \tag{2}$$

Similarity or association within subgraph A is the sum of all edge weights within A: W(A). Thus, the mini-max clustering principle requires we minimize cut(A,B) while maximizing W(A) and W(B) at the same time. All these requirements can be satisfied simultaneously by the following partition metric function,

$$Mcut=\frac{cut(A,B)}{W(A)}+\frac{cut(A,B)}{W(B)} \tag{3}$$

This function is called the min-max cut function or Mcut for short [5]. When G is partitioned into K SDN domains $SDN_1,SDN_2,\ldots,SDN_k$, Eq.(3) can be adjusted as follows:

$$Mcut_k=\frac{cut(SDN_1,\overline{SDN_1})}{W(SDN_1)}+\frac{cut(SDN_2,\overline{SDN_2})}{W(SDN_2)}+\ldots+\frac{cut(SDN_k,\overline{SDN_k})}{W(SDN_k)} \tag{4}$$

Inspired by the previous work of $Mcut_k$, we use this metric as to solve WAN partition problem. $Mcut_k$ can provide balanced cuts and average the load of each controller.

D is the degree matrix and x is a $n=|V|$ dimensional indicator vector. The objective function of $Mcut_k$ for SDN partition is:

$$Mcut(SDN_1,SDN_2,\ldots,SDN_k)=\frac{1}{2}\sum_{i=1}^{k}\frac{x_i^T(D-W)x_i}{x_i^TWx_i} \tag{5}$$

After solving the WAN partition problem, we now introduce the controller placement problem in each sub-network. Each SDN domain has only one controller. Where should the controller go in order to ensure the performance of single SDN domain? It is called a facility location problem and appears in many contexts [2]. In each sub-network, the reliability and efficiency of the controller can be improved by

reducing its sizes. So the propagation latency becomes the most significant design metric for the sub-network partitioned from WAN.

For each SDN domain $SDN_i(V_i, E_i, W_i)$ separated from topology G, where Dist(s,d) is the shortest path from node $s \in V_i$ and $d \in V_i$, the number of nodes $n = |V_i|$, the average propagation latency for the controller C' is:

$$L_{avg}(C') = \frac{1}{n}\sum_{s \in V_i} \min_{(d \in C')} Dist(s,d) \quad (6)$$

Eq.(6) represents the average latency of the sub-network. We can use the metric to find the placement C' from the set of all possible controller placements C, along with $L_{avg}(C')$ is minimum.

In the following sections, we introduce an approximation algorithm to optimize the NP-hard problem by using the spectral clustering. To test the two optimization targets mentioned above, we design our testing methodology for SDN partitions.

III. PLACEMENT ALGORITHM

In recent years, spectral clustering has become one of the most popular modern clustering algorithms, and has been applied in machine learning, text summarization, etc [5, 6]. The success of such algorithms depends heavily on the choice of the metric. From the analysis regarding balanced cuts of WAN topology, we tend to select $Mcut_k$ because $Mcut_k$ is essential in K-way partition.

In each SDN domain SDN_i, the data points $V_1, V_2, ..., V_n$ can be the nodes of WAN topology, and their similarities $s_{ij} = s(v_i, v_j)$, measured according to some similarity function which is symmetric and non-negative.

$$s_{ij} = 2\arcsin\sqrt{\sin^2(\frac{\alpha}{2}) + \cos(lat_i) \times \cos(lat_j) \times \sin^2(\frac{\beta}{2})} \times 6378.137 \quad (7)$$

where $v_i(lat_i, lon_i), v_j(lat_j, lon_j)$ represents the latitude and longitude of the points v_i, v_j, $\alpha = |lat_i - lat_j|$, $\beta = |lon_i - lon_j|$, and the radius of the earth is 6378.137 km. We denote the corresponding similarity matrix by $S = (s_{ij})_{i,j=1...n}$, which can be used to evaluate the propagation latencies between the nodes. Now we would like to state our spectral clustering for SDN controller placement problem in WAN. The whole algorithm 1 is outlined as follows:

Algorithm 1 Spectral Clustering Algorithm()

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters.
Output: Clusters $SDN_1, ..., SDN_k$ with $SDN_i = \{j \mid y_j \in C_i\}$, the location of the controllers.

1: Construct a similarity graph A with the Eq.(7). Let W be its weighted adjacency matrix.
2: Compute the degree matrix D with $D(i,i) = \sum_j A_{ij}$.
3: Compute the graph laplacian matrix L.
4: Compute the first k eigenvectors $v_1, ..., v_k$ of L.
5: Let $V \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $v_1, ..., v_k$ as columns.
6: **for** (i=1,…,n) **do** let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of V.
7: cluster the points $(y_i)_{i=1,...,n}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, ..., C_k$.
8: **return** $SDN_1, ..., SDN_k$.

Aiming to understand the benefits of the spectral clustering, we have evaluated our algorithms using the Internet2 OS3E topology [7]. The OS3E has 34 nodes and about 41 edges, shown in Figure 2.

We implemented a Matlab-based framework to compute the Spectral Clustering placement. Figure 2 shows a SDN domains partition plan based on the spectral clustering algorithm. As we can see, the OS3E topology is partitioned into four SDN domains equally when k=4. Among these SDN domains, the controllers will be placed in the nodes which are labeled as stars. As expected, our spectral clustering algorithm meets the requirements of the metrics mentioned in Section Ⅱ.

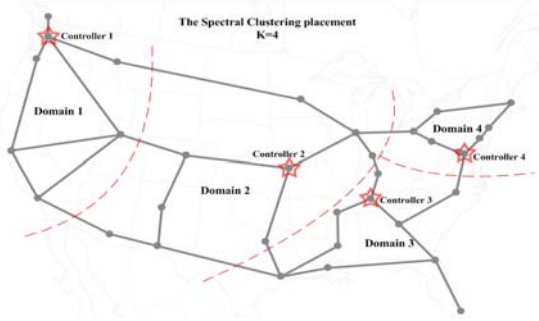

Fig. 2 Four SDN domains partition based on the Spectral Clustering algorithm

In the following sections, we compare the performance of our placement with other placements mentioned in [2], and create a set of advanced testing scenarios to verify it.

IV. EXPERIMENTS

In this section, we introduce our testing methodology and describe the experimental results of our placement compared with the others. All the algorithms mentioned in Section Ⅲ are evaluated with the Beacon controller and cbench [8]. All experiments have been performed on a cluster which consists of 36 machines running Ubuntu server 64 bit. Each node has 2 AMD Opteron 2212 2.00 GHz CPUs, 80 GB SCSI HDD, 8 GB of RAM, Intel 100 Mbps Ethernet Controller. In the meantime, we deploy a host as Beacon controller and others as cbench hosts. The test framework is shown in Figure 3.

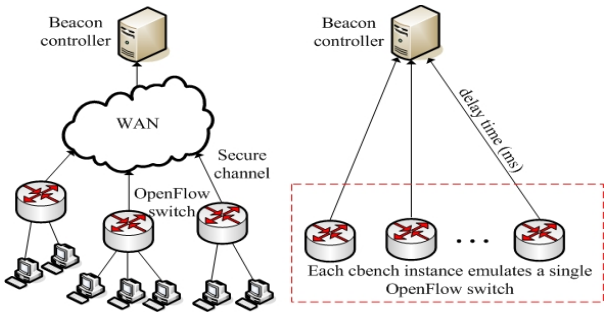

Fig. 3 Cbench emulation for WAN topology

To evaluate the controllers, we ran Beacon controller software as WAN SDN controller with the recommended settings, which is a multi-thread java-based controller. We relied on the latest available sources of Beacon version 1.04 dated April.2014. We chose Beacon because it shows the better performance than the other controllers [8, 9]. We ran cbench instances on multiple modes of the cluster to emulate the switches. As shown in Figure 3, each cbench instance in our experiments emulated a single OpenFlow switch and all of these instances sent OpenFlow packet-in events to a single controller. The cbench instances were connected to the controller with 100 Mbps interconnects.

To emulate a real network for WAN propagation latencies, we used most nodes of the cluster for running cbench instances. The number of cbench nodes was varied for different experiments, which depended on the metric being calculated. Each cbench emulated a single OpenFlow switch sending packet-ins to the controller at uniform rates with different delay time. The delay time are calculated by the propagation latencies between two nodes in WAN topology mentioned above.

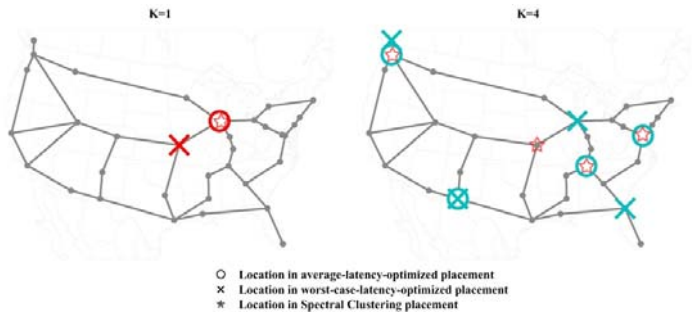

Fig. 4 Three placements for 1 and 4 controllers in the OS3E deployment

For comparing the performance of our placement with other placements, we also evaluated the average-latency-optimized placement and worst-case-latency-optimized placement mentioned in [2] as Fig. 4 shown. The Spectral Clustering placement is most similar to the average-latency placement and completely different from the worst-case-latency placement. For example, all the controllers of the Spectral Clustering and average-latency placement should go Chicago when k=1, which balances the high density of east coast cities with the lower density of cities in the west. The different ways produce the same results. However, to minimize worst-case latency for K=1, the controller should go in Kansas City instead, where is near the center of the topology. As we expected, the Spectral Clustering placement is most similar to the average-latency placement when k=4. By using the mini-max clustering principle, the Spectral Clustering placement can combine latency with performance. The worst-case-latency placement is defined as the maximum node-to-controller propagation delay, and is proved to be the least effective method. Thus, we will not consider it in subsequent sections.

In order to test the effectiveness of our solution, we present a comparative performance analysis of the Spectral Clustering and average-latency placement. We create a set of advanced testing scenarios and conduct experiments under many different settings and metrics, which allow us to get a deeper insight into the WAN controller performance issues. All experiments are performed with Beacon and cbench. We run each experiment five times and take the average number as the result.

A. Latency

An important ability of the OpenFlow controller is that it can process the incoming packet-ins as fast as possible, which we call latency. We keep a cbench instance emulating a single switch, and made many cbench instances send packet-ins to their controller with different amounts of connected hosts. Depending on the metric being calculated, the number of cbench instances was varied for different experiments.

For latency experiments each test consists of 500 loops each lasting 100 milliseconds. The first loop and last loop are considered as controller warm-up and cool-down, which results are discarded. Each test uses from 100 to 100,000 unique MAC addresses (representing emulated end hosts). We keep a constant number of 1 thread and progressively increased the host density.

Figure 5 shows the controller latencies of different placements with different number of hosts and one thread. In each placement the controllers are labeled from left to right. For example, Spectral Clustering placement has four controllers as Figure 2 shows. The controller deployed in Seattle is registered under the number 1, the controller deployed in Kansas City is registered under the number 2, and so on. From Figure 5, it can be seen that the most controllers of the Spectral Clustering placement have the faster responses than the average-latency placement. In the average-latency placement, the third controller shows the best performance because it services 13 nodes. But the second controller services only 3 nodes and has exactly the opposite effect. The controller latency is also affected by the propagation latency between controller and switch. The first controller deployed in Seattle has lower performance as Figure 5 shows because of the vast distances between the northwest cities.

The average latency reflects the average response from each cbench to the controller, which demonstrates significant correlation with the number of connected nodes and one thread. Due to the balanced cuts mentioned above, Figure 6 indicates that the Spectral Clustering placement performs significantly well in terms of average latency. In the average-latency placement, the second controller deployed in El Paso but not Kansas City, which leads to the imbalance between the second and third controller.

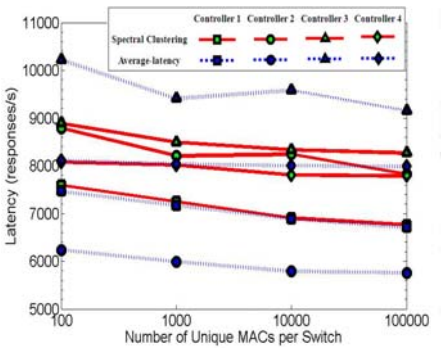

Fig. 5 Latency comparison

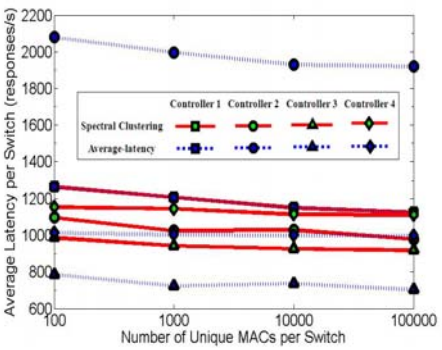

Fig. 6 Average latency comparison

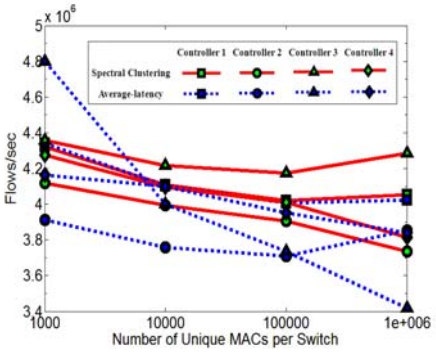

Fig. 7 Throughput with different hosts

B. *Throughput*

In this experiment, we evaluate the controllers of the two placements on the throughput performance. All cbench instances are kept in throughput mode under which it continuously sends Packet-In messages to Beacon over a period of time. Our focus in this subsection is to study the average throughput of each controller with different number of connected hosts.

For throughput experiments, each test consists of 5 loops each lasting 1,000,000 milliseconds. The first and last loop results are discarded. The number of hosts is from 1000 to 1,000,000 in each test. We keep a constant number of 8 worker threads and progressively increased the host density. Figure 7 shows the correlation with the number of connected hosts. The number of hosts in the SDN domain has immaterial influence on the performance of most of the controllers under test. The controller 1 of the Spectral Clustering decreases its throughput from 4.3 to 4 million flows per second with 106 hosts. But in the average-latency placement, the performance of controller 3 goes down significantly when more hosts are connected. Also, its controller 2 has the lowest throughput in all controllers. This is caused by specific details of the average-latency placement, namely, the imbalance of its SDN domains partitioning. From Figure 7, it can be seen that the Spectral Clustering placement shows the better performance than the average-latency placement because of its SDN domains partitioning. We also see an unstable trend in throughput with increasing number of hosts for the average-latency placement.

V. CONCLUSION

In this paper, we showed the answer to where and how many controllers to deploy on wide-area networks. Our approach is based on partitioning a large network into several small SDN domains by using the Spectral Clustering placement algorithm. To maximize the reliability of controller and minimize the latency of WAN, we presented the metrics of the Spectral Clustering placement. After presenting a test framework with Beacon and cbench, we conducted experiments under many different settings and metrics. Finally the experiment results show the Spectral Clustering placement is good at solving the SDN controller placement problem compared with others. In the future work, we expect to find a method to determine the number of clusters automatically for the Spectral Clustering placement.

ACKNOWLEDGMENT

This work is supported by the National Science Foundation for Distinguished Young Scholars of China under grant No. of 61225010, National Nature Science Foundation of China (Nos. 61173162, 61173165, 61370199, 61300187, 61300189, 61370198 and 61432002).

REFERENCES

[1] Xie, H., Tsou, T., Lopez, D., Yin, H., Gurbani, V.: Use cases for alto with software defined networks. IETF Internet-Draft (2012)

[2] B. Heller, R. Sherwood, N. McKeown, The controller placement problem, in: Proceedings of the first workshop on Hot topics in software defined networks, HotSDN '12, 2012.

[3] H. Yin, H. Xie, T. Tsou, D. Lopez, P. Aranda, R. Sidi, Sdni: A message exchange protocol for software defined networks (sdns) across multiple domains, Tech. Rep. draft-yin-sdn-sdni-00.txt

[4] Lin, Pingping, Jun Bi, and Yangyang Wang. "East-West Bridge for SDN Network Peering." Frontiers in Internet Technologies. Springer Berlin Heidelberg, 2013. 170-181.

[5] Ding, Chris HQ, et al. "A min-max cut algorithm for graph partitioning and data clustering." *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*. IEEE, 2001.

[6] Liu Na, Lu Ying, Tang Xiao-jun, Wang Hai-wen, and Li Ming-xia. Study on Automatically Determining the Optimal Number of Clusters Present in Spectral Co-clustering Documents and Words. Journal of Chinese Computer Systems, 35(3):610-614, 2014.

[7] Internet2 open science, scholarship and services exchange. https://www.internet2.edu/network/ose/.

[8] Shah, Syed Abdullah, et al. "An architectural evaluation of SDN controllers." Communications (ICC), 2013 IEEE International Conference on. IEEE, 2013.

[9] Shalimov, Alexander, et al. "Advanced study of SDN/OpenFlow controllers." Proceedings of the 9th Central & Eastern European Software Engineering Conference in Russia. ACM, 2013.

[10] Hock, David, et al. "Pareto-optimal resilient controller placement in SDN-based core networks." Teletraffic Congress (ITC), 2013 25th International. IEEE, 2013.

[11] Dixit, Advait, et al. "Towards an elastic distributed sdn controller." Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, 2013.