Physical-Virtual Topological Visualization of OF@TEIN SDN-enabled Multi-site Cloud

Muhammad Usman, Aris Cahyadi Risdianto, Jungsu Han, JongWon Kim*
School of Electrical Engineering & Computer Science
Gwangju Institute of Science & Technology
Republic of Korea
{usman, aris, jshan, jongwon}@nm.gist.ac.kr

Nguyen Van Huynh
School of Electronics & Telecommunications
Hanoi University of Science & Technology
Vietnam
nguyenvanhuynhhust@gmail.com

Abstract **— Infrastructure visualization based on monitored resource status is essential for the effective operation of modern SDN (Software-Defined Networking)-enabled cloud. More specifically, collecting and visualizing visibility information about both physical & virtual (p+v) types of resources is one of key requirement for multi-site cloud operators. However, most existing visualization tools fall short in effectively providing the p+v topological visualization of multi-site cloud infrastructure. Thus, in this paper, by taking an example of OF@TEIN (OpenFlow @ Trans-Eurasian Information Network) multi-site experimental cloud, we attempt to realize visualization software to assist both developers and operators in capturing the p+v topological visualization of infrastructure resources.**

Keywords — Multi-site cloud, multi-layer visibility, topological visualization, physical and virtualized resources.

I. INTRODUCTION

Unlike traditional network-focused testbeds, Future Internet testbeds should provide experimental networking facility without limiting the number of concurrent users with different resource requirements, the number or type of supported services, and most importantly the topological shape of deployed networks. Thanks to the emerging SDN (Software-Defined Networking) paradigm, open and programmable types of experimental networks are beginning to be deployed in the construction and operation of multi-site testbeds. Aligned with Future Internet testbed projects like GENI [1] and FIRE [2], we launched OF@TEIN project [3] over TEIN (Trans-Eurasia Information Network) in 2012, which is now expanded as an SDN-enabled multi-site cloud testbed. As shown in Fig. 1, OF@TEIN multi-site cloud (denoted as OF@TEIN Playground) connects 10 international sites spread across 9 countries (i.e. Korea, Malaysia, Thailand, Indonesia, India, Vietnam, Pakistan, Taiwan, and Philippines).

OF@TEIN Playground consists of two types of resources: physical and virtual types. Generally, physical servers and switches (i.e., referred as boxes in this paper) and physical interconnects are referred as physical resources. In comparison, virtual machines (via the support of hypervisors), virtual switches, and virtual interconnects (e.g., path, links, and virtual ports) are referred as virtual resources. In order to effectively operate OF@TEIN Playground, it is really important to understand where physical and virtual resources are located and how they are coupled (i.e., inter-related) together. The physical & virtual (p+v) topological visualization, as a helper solution to the above challenge, presents as unified visualization all the essential information (e.g., p+v resource couplings and resource status including associated performance values). To effectively realize the p+v topological visualization, significant types and amount of visibility data is required to be accumulated, organized, and interpreted. However, to the best of our knowledge, existing open-source solutions does not provide the required p+v topological visualization capability to directly reflect and match the multi-site distributed resources of OF@TEIN Playground.

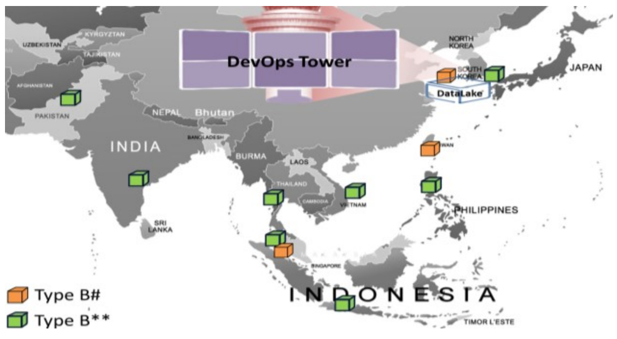

Figure 1: OF@TEIN Playground as multi-site cloud.

Thus, in this paper, we first leverage the resource-level visibility solution from our earlier work [4], which can enable the collection of multiple resource-level visibility data. Next, the collected visibility data is carefully integrated (i.e., organized and interpreted) to enable the required p+v topological visualization. Like this, in this paper, we focus on utilizing resource-layer visibility data to realize the p+v topological visualization for OF@TEIN Playground. Furthermore, additional support for large-scale visualization is provided through NetWall networked tiled display [3]. In summary, the main contributions of this paper are the design, implementation, and verification of the physical & virtual topological visualization tool for OF@TEIN SDN-enabled multi-site cloud.

The remainder of this paper is organized as follows. In Section II, we discuss the limitations of existing tools and the requirements for p+v topological visualization. In Section III, we cover the overall design for the p+v topological visualization. After discussing visibility data preparation for the p+v topological visualization in Section IV, we explain the implementation details in Section V. Finally, section VI concludes the paper.

II. Background and Related Work

A. *Limitations of Existing Tools*

There are several commercial and open-source network monitoring and visualization tools available in the market; among those we studied and tested Cacti, Nagios, and Zenoss [6, 7]. Cacti is a complete network graphing solution. However, the limit on the storage data and outdated visualizations makes it less promising. Nagios is also well-known for server monitoring and alerting, which is suffering from scalability limitation. Finally, Zenoss is a python-based visualization tool with interactive GUI. However, our deployment trial with OF@TEIN Playground revealed that Zenoss is resource extensive and data is spread across multiple data stores, which makes it difficult to flexibly utilize visibility data.

B. *Physical+Virtual Topological Visualization: Requirements*

The p+v topological visualization tool should be innovative to offer a unified topological view for SDN-enabled cloud segments by covering both p+v resource components of OF@TEIN Playground. Also, for faster troubleshooting of OF@TEIN Playground, this p+v topological visualization must integrate the visualization of visibility data collected from multiple sources. Furthermore, enabling large-scale visualization support through NetWall is another requirement for the p+v topological visualization.

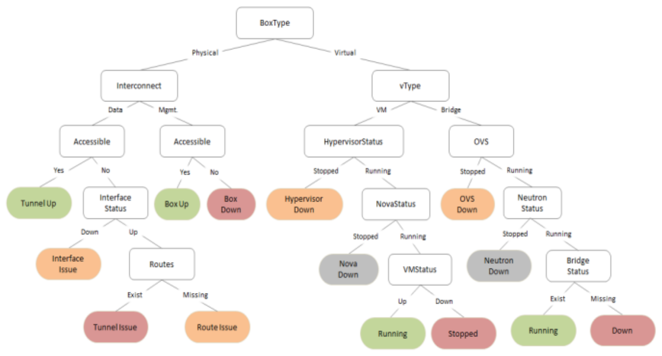

Figure 2: Design of p+v topological visualization.

III. OF@TEIN Playground and Physical & Virtual Topological Visualization

A. *OF@TEIN Playground Visibility*

Initially, the operation of visibility collection requires the identification of key performance metrics that assert major influence on the operational states of p+v resources [4]. In this paper, for visibility collection, we extensively utilize Intel Snap monitoring framework [8]. Intel Snap is an open-source telemetry framework designed to simplify the collection, processing, and publishing of DC (data center) equipment visibility via unified APIs. We linked the publishing part of Snap framework with Apache Kafka[1] and collected visibility data from remote resources to the centralized Visibility Center. Note that Apache Kafka messaging is chosen to provide data delivery functionality due to its fault-tolerant capabilities and easy-to-use APIs.

B. *Design for p+v Topological Visualization*

The design for p+v topological visualization follows tree structure where each node represents attributes, branches represents the values of attributes, and leafs represents classification results. Different colors at the leaf nodes represent special status messages associated with the resources, which is color-coded to facilitate the operators with visual information for faster troubleshooting of OF@TEIN Playground. As an example, by using the tree structure shown in Fig 2, let us consider tunnel verification process. In a tunnel verification example, the value of 'attribute boxtype' = "Physical" and the value of 'attribute interconnect' = "Data". Next, based on the value of 'attribute accessible', further decisions are made. That is, if 'attribute accessible' = "yes", tunnel is up-and-running and no further processing is required. However, in case of 'attribute accessible' = "no", 'attribute interface' is checked and then next decision is made. If the result of 'attribute interface' = "up" and followed by Routes = "Exist", then the final conclusion will be "tunnel issue".

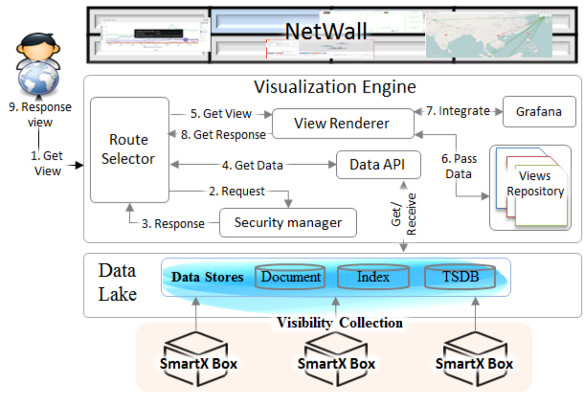

Figure 3: Workflow for p+v topological visualization.

The p+v topological visualization needs several functionality modules for easier processing and flexible integration. The workflow for p+v topological visualization is shown in Fig. 3. It starts with the view request from developers/operators and then 'Route Selector' module processes view requests and facilitates inter-communications between other modules. Next, 'Security Manger' module performs user authentication & authorization, which is followed by 'Data API' module that provides data acquisition functionality from Data Lake[2] for visualization rendering. After

[1] Apache Kafka is a distributed streaming platform for building real-time data pipelines and streaming apps [4].

[2] Data Lake refers to an open-style storage repository that holds a huge amount of raw data in its native format until it is needed (e.g., to be used for visualization in this paper).

that, 'View Renderer' module parses data object and generates requested views by utilizing graph-based visualization library Vis.js [11].

IV. Visibility Data Preparation for p+v Topological Visualization

A. Data Lake for p+v Resource-layer Visibility

The collected operation visibility data is stored into so-called SmartX Operation Data Lake [4]. Currently, we use InfluxDB to store near-real-time metric data while MongoDB is used for the archival of visibility data. Note that MongoDB is a NoSQL, document-oriented database tool while InfluxDB is an open-source time series database.

B. Managing Configuration/Status Data for Visualization

To enable p+v topological visualization, we rely on separate database that stores updated configuration/status data for the resources of OF@TEIN Playground. This configuration/status database is managed by using MongoDB collections. Also in order to facilitate data integration across different collections, specific keys (e.g., physical box identifier) are defined. The p+v resource states are updated using Java-based plugins that are specifically developed for collecting and updating data in MongoDB collections on regular intervals.

V. Visualization for OF@TEIN Playground Visibility

We considered both Kibana and Grafana [9] to visualize performance metrics. However, Kibana can only support ElasticSearch as a backend data store, which makes it a less futuristic option for OF@TEIN Playground visualizations. In comparison, Grafana supports a large number of backend data stores. Thus, we integrated Grafana to create customized dashboards.

Since the p+v topological visualization tool is to be deployed over the web, we use a Node.js web server. Node.js is known for its event-driven, non-blocking I/O model, which makes it a lightweight and efficient solution for visualization tasks.

To enable the p+v topological visualization by using graph-based approach, we can consider Three.js [10] and Vis.js [11] libraries. Three.js offers dynamic and 3D drawings capability but it introduces a lot more complexity. On the other hand, Vis.js offers a simple and easy-to-use graph-based library for interactive network graphs. Therefore, we select Vis.js over Three.js.

The p+v topological visualization of OF@TEIN Playground is visualized on a large-scale, SAGE-enabled [5] NeTD (networked tiled display), named as NetWall. The key benefit of using NeTD is that it allows both developers and operators to quickly visualize the p+v topological visualization over multi-screens with flexible controls provided by the SAGE framework. Fig. 4 shows the snapshot of p+v topological visualization over NetWall.

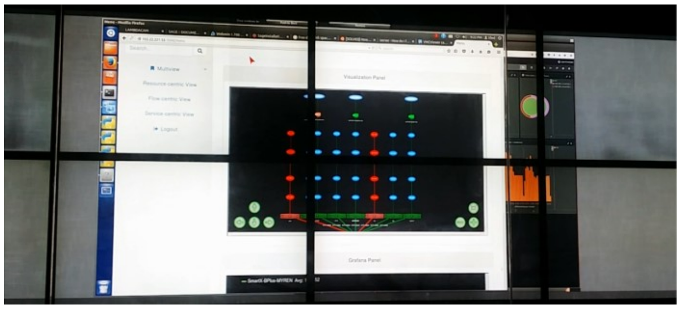

Figure 4: Visualization of OF@TEIN Resources over 4x2 NetWall.

VI. Conclusion and Future Work

In this paper, we presented a unique attempt to enable the p+v topological visualization tied with large-scale NetWall display to fully illustrate the resource-layer visibility of OF@TEIN Playground as an example for multi-site SDN-enable cloud. However, OF@TEIN Playground still requires a lot of further improvements to reach the ultimate goal of visualizing so-called multi-layer/multi-level visibility. As a second step toward multi-layer/multi-level visibility, we are currently working hard to expand our visualization support for flow-layer and workload-layer visibility.

Acknowledgment

This work was supported in part by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.R7117-16-0218, Development of Automated SaaS Compatibility Techniques over Hybrid/Multisite Clouds). This work was also supported in part by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. B190-15-2012, Global SDN/NFV Open-Source Software Core Module/Function Development).

References

[1] M. Berman et al.: GENI: A federated testbed for innovative network experimetns. Computer Networks, vol. 61, pp. 5-23. (2014)

[2] FIRE: Future Internet Research and Experimentation, https://www.ict-fire.eu/

[3] A. C. Risdianto and J. Kim: Prototyping media distribution experiments over OF@TEIN SDN-enabled testbed. In: APAN-NRW, pp. 12-18. Nantou (2014)

[4] M. Usman, A.C. Risdianto and J. Kim: "Resource Monitoring and Visualization for OF@TEIN SDN-enabled Multi-site Cloud". In: ICOIN, pp. 427-429. Kota Kinablu (2016)

[5] M. Usman, A.C. Risdianto, T. C. Ling and J. Kim, "Visualizing Life Cycle Experiments in SmartX Operation Tower". In: JCCI. Buyeo (2015)

[6] The Open-Source Monitoring Landscape, http://www.slideshare.net/vo_mike/the-opensource-monitoring-landscape

[7] Open source monitoring systems, http://www.slideshare.net/forthscale/open-source-monitirng-systems-1

[8] Snap, http://snap-telemetry.io/

[9] Grafana, http://grafana.org/

[10] Threejs, https://threejs.org/

[11] Visjs, http://visjs.org/