ENHANCED ONOS SDN CONTROLLERS DEPLOYMENT FOR FEDERATED MULTI-DOMAIN SDN-CLOUD WITH SD-ROUTING-EXCHANGE

Aris Cahyadi Risdianto[1], Pang-Wei Tsai[2], Teck Chaw Ling[3], Chu-Sing Yang[4], and JongWon Kim[5]

[1, 5] School of Electrical Engineering and Computer Science
Gwangju Institute of Science and Technology (GIST)
61005 South Korea

[3] Faculty of Computer Science and Information Technology
University of Malaya (UM)
50603 Malaysia

[4] Institute of Computer and Communication Engineering, Department of Electrical Engineering
National Cheng Kung University (NCKU)
701 Taiwan

Email: aris@nm.gist.ac.kr[1], pwtsai@ee.ncku.edu.tw[2], tchaw@um.edu.my[3], csyang@ee.ncku.edu.tw[4], jongwon@nm.gist.ac.kr[5]

ABSTRACT

The requirement of auto provisioning, administration, management and governing of networking resources in distributed environment, which spreads over multiple administrative domains (i.e., multi-domain) is a challenging task. This paper demonstrates an effort to inter-connect distributed SDN-Cloud resources which spread over three administrative domains in three different countries on OF@TEIN Playground. The proposed SD-Routing-Exchange utilizes BGP routing protocol as control-plane for federated multi-domain SDN-Cloud data-plane. This method has improved the flexibility of networking control, and provides reasonable inter-connections redundancy and performance enhancement.

Keywords: Software-defined networking, distributed SDN-Cloud, BGP routing, and software-defined routing exchange.

1.0 INTRODUCTION

Motivated by world-wide Future Internet testbed deployments, GIST launched OF@TEIN Playground (i.e., testbed) collaboration to deploy OpenFlow SDN-enabled SmartX Boxes over TEIN (Trans-Eurasia Information Network) infrastructure [1]. Since its initial deployment in 2012, OF@TEIN Playground has gone through several upgrades to improve the experience of operators and developers. Recently, in spring 2015, we began to deploy OpenStack [2] cloud-enabled resources so that we can unify user (i.e., tenant) accounts and GUIs with OpenStack Keystone and Horizon, and improve virtualized tenant networking with OpenStack Neutron. Thus, OF@TEIN Playground can now address the key requirements of distributed SDN-Cloud testbed by combining the open-source software support of OpenFlow-enabled SDN and OpenStack-leveraged cloud.

Several ideas have been proposed to utilize programmable networking for virtual tenant networks over distributed cloud data centers. Several enhancements are required to provide fully-programmable virtual tenant networks, where network slices are created based on FlowSpace information of current OpenFlow-enabled SDN infrastructure. Multi-tenant virtual networks can be adopted to provide cloud networking which is aligned with network slicing requirement in Future Internet testbed [3]. Also, [4] propose multi-cloud provider approach to build multi-regional cloud deployment. Generally, they need to build private WAN infrastructure for connecting their distributed cloud data centers. In reality, it is well-known that individual WAN links are very expensive for connecting multiple high-end and specialized premium equipment to support high availability. This is because WAN links used for business applications, where packet loss typically thought unacceptable. In addition, it is an expensive deployment to provide high-capacity bandwidth over long distances and allow bandwidth over-

provisioning. Apart from that, usually the links are in a poor efficiency situation (i.e., the average utilization is less than 40-60% in busy hours), no entity has a global view and ingress routers greedily select the paths with current mode (e.g., MPLS-TE). As a result, the network can support locally optimal but globally sub-optimal routing. Especially in inter-DC WANs with limited support for flexible resource allocation, typically it will obtain throughput proportional to their sending rate, and it will lead into a poor sharing situation between servers in the data centers [5] [6]. Another approach tries to utilize SDN paradigm for improving current internet routing system which also may be used to inter-connect between distributed public cloud data centers. The current limitations of BGP-based routing systems are routing-based on IP prefix, which is influenced by direct neighbor, and indirect expression of policy. The results are unreliable, inflexible, and difficult to manage routing information exchange.

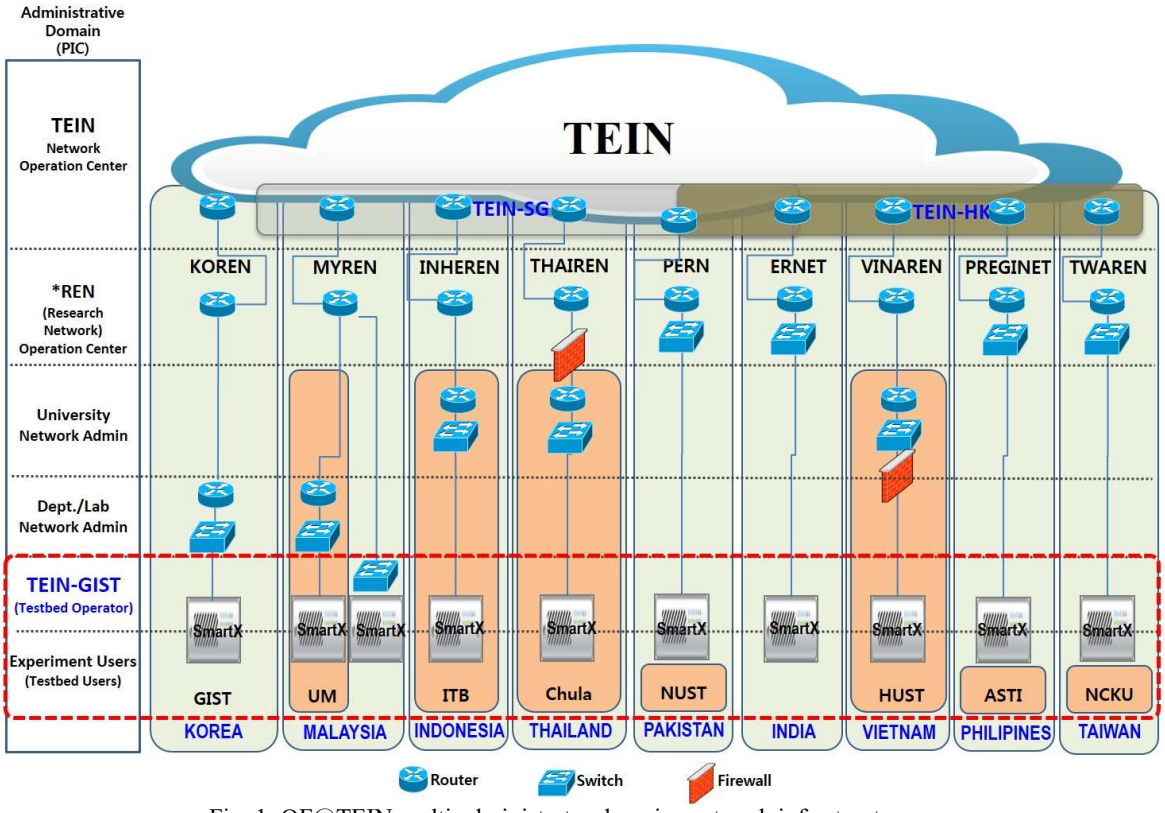

Fig. 1: OF@TEIN multi administrator domains network infrastructure.

Similarly, OF@TEIN resources, spread over several R&E (research and education) network sites with different administrative domain, are attempted to be inter-connected through the TEIN network, as shown in Fig. 1. The deployed hyper-convergent SmartX Boxes provide several types of physical and virtualized resources in building OF@TEIN SDN-Cloud infrastructure. The virtualized resources can support specific (e.g., computing, networking, and monitoring) functions and configurations for either SDN or cloud entity. The inter-connections between SmartX Boxes are utilized an OpenFlow-based overlay networking over TEIN, emulating the WAN infrastructure between distributed cloud data centers in OF@TEIN Playground. Similarly, in OF@TEIN overlay networking, if no entity has a global view and privilege over underlying infrastructure, inter-DC inter-connection also experience a low-bandwidth throughput and connection instability caused by problematic sub-optimal routing of the underlay network. Moreover, there is no shared information about underlying infrastructure (i.e., routing table, forwarding state, etc.) which is required to improve that inter-connections.

Aligned with the importance of worldwide federation of SDN-based future internet testbeds, several ideas and standards have been proposed. Those ideas include control/data-plane federation, controller-to-controller communication, and distributed multi-domain controllers. While the ideas still debatable for some time to come, the experience of deploying federated SDN testbed in real environment is really important. OF@TEIN tries to

select one option of federated SDN, deploy in the multi-domain network environment, and evaluate the flexibility and performance.

2.0 BACKGROUND AND RELATED WORK

The OpenFlow-enabled SDN concept that decouples software and hardware evolution, will allow control-plane software to become simpler. Also, it evolves more quickly while data-plane hardware evolves around programmability and performance. By leveraging this phenomenon, many researchers are extracting the possibilities to utilize OpenFlow-enabled SDN for building centralized and programmable WAN architectures, or enabling expressive policies in interdomain routing. The following section will discuss several works from different parties such as Google [5], Microsoft [6], Noise Lab [7], and ON.Lab (Open Networking Laboratory) [8] to address example efforts in solving current architecture limitations.

2.1 Google B4 (Globally Deployed SD-WAN)

Google B4 was driven by the observation that they could not achieve the level of scale, fault tolerance, cost efficiency, and control required for their network, by using traditional WAN architectures. These are due to their requirements in networking such as i) elastic bandwidth demands, ii) moderate number of sites, iii) end application controls, and iv) cost sensitivity. They attempt to address these requirements by adopting the SDN concept that gives a dedicated, software-based control plane running on commodity servers to perform the simplified coordination for both planned and unplanned network changes. It also allows them to leverage the raw speed of commodity servers. OpenFlow enables them an early investment for the SDN ecosystem that can leverage a variety of switch/data plane elements. As mentioned in the literature, the goals are: i) rapid iteration on novel protocols, ii) simplified testing environments, iii) improved capacity planning, and iv) simplified management through a fabric-centric rather than router-centric WAN view [5]. After three years of implementation, they finally came out with the SD-WAN solution for their internal distributed data centers network called as “G-Scale”. The design for each B4 site contains the switch hardware layer to primarily forward traffic without complex control software. The site controller layer consists of NCS (network control server) that hosts both OpenFlow controllers and network control applications. The global layer also consists of logically-centralized applications (e.g., an SDN Gateway and a central TE server) that enable the centralized control of the entire network via the site-level network control applications [5].

2.2 Microsoft SWAN (Software-driven WAN)

In order to solve the similar problem with previous work, Microsoft decided to design a new WAN architecture that attempts to carry more traffic and support flexible network-wide sharing. This new network could be explained as below [6]:

1. All services, except interactive ones, inform the SD-WAN controller about their demand between the pairs of DCs.
2. The controller, which has an up-to-date, global view of the network topology and traffic demands, computes amount and paths for the traffic.
3. Based on the SDN paradigm, the controller directly updates the forwarding states of OpenFlow switches.

Their final design is deploying Service hosts and brokers collectively to estimate the demand and limit of services, by taking into account of the rate allocated by the SDN controller. Network agents track topology changes and traffic patterns with the assist of switches. Controller uses the obtained information on service demands and network topology (for path calculation) for every Tc interval [6].

2.3 SDX (Software-Defined Exchange Point)

The aim of this work is to change wide-area traffic delivery by designing, prototyping, and deploying a software defined exchange (SDX) which addressing four challenges such as i) compelling applications, ii) programming abstractions, iii) scalable operation, and iv) realistic deployment. Virtual SDX switch abstraction allows each AS to run SDN applications that specify flexible policies for dropping, modifying, and forwarding the traffic, and then the SDX must combine the policies of multiple ASes into a single coherent policy for the physical

switch (es). The SDX route server allows each participant to forward traffic to all feasible routes for a prefix (even if it learns only one) because it has unique characteristic such as overriding default BGP routes, forwarding only along BGP-advertised paths, grouping traffic based on BGP attributes, originating BGP routes from the SDX, and integrating SDX with existing infrastructure. SDX runtime system efficiently compiles the policies of all participants into low-level forwarding rules by minimizing the number of rules in the switches (data-plane efficiency) and minimizing the computation time under realistic workloads (control-plane efficiency). The SDX controller implementation has two main pipelines: a policy compiler, which is based on Pyretic; and a route server, which is based on ExaBGP.

2.4 ONOS (Open Networking Operating Systems)

To answer the challenge from the service provider in providing the carrier-grade SDN controller based on OpenFlow, ON.Lab from Stanford University is leading the development of open-source NOS (network operating system). By managing the network resources and providing high-level abstractions and APIs, ONOS (Open Networking Operating Systems) provides an open platform that simplifies the creation of innovative and beneficial network applications and services that work across a wide range of hardware [8]. By the end of 2014, they release their first ONOS version that is primarily focused on use cases around retrofitted data centers of service providers. ONOS adopts a distributed architecture for high availability and scale-out capabilities. It provides a global networking view to the applications, which is logically centralized even though it is physically distributed across multiple servers. Several ONOS features are summarized as follows [8]:

- *Global Network View*: manage and share network state across ONOS servers in a cluster.
- *Scale-out*: runs on multiple servers, each of which acts as the exclusive OpenFlow master controller for a subset of switches.
- *Fault tolerance*: a switch connects to multiple ONOS instances, but only one master instance for discovering switch information and programming the switch.
- *RAMCloud Data Store*: low-latency, distributed key-value store, which offers remote read/write latency in the range of 15-30 ms.
- *Optimized Data Model*: a network object table (e.g. switch, link, flow entry, etc.) may reduce unnecessary contentions between independent updates and no longer maintains the data integrity at the data store level.
- *Topology Cache*: in-memory topology view implements a set of indices on top of the data to allow faster lookups.
- *Event Notifications*: polling is set to build several publish-subscribe event notifications among all ONOS instances based on the notification type.
- *Network View API*: replace the generic blueprint graph API with an API designed specifically for network applications (topology, events, and a path installation).

2.5 Related Work

Several efforts have been done to figure out the proper SDN-based federation. Future Internet and distributed cloud (FIDC) testbeds [9] still progress for defining control-plane and data-plane federation including policy enforcement and ID authentication. While current FIDC frequently used VLAN-based data-plane federation and tunneling for less desirable solution, SDN-based federation is one of promising solution by providing isolated FlowSpace for each resource or presenting virtual *meet-me* point with some pre-defined policies. Another efforts directly defined SDN controller-to-controller communications to provide network-wide views. Distributed Multi-domain SDN Controllers (DISCO) [10] proposed agent/messaging-based communications for providing overlay WAN network. Despite of the ability to run distributed controllers, relying on “specialized” agent impose difficulties to integrate with current network environment.

In addition, one standard from IETF [11] for federating SDN controller by utilizing MP-BGP through the Internet or L2/L3 VPNs has been proposed, as BGP is proven mechanism to achieve scale and secure isolation for multi-tenant, multi-domain and multi-provider environment. ONOS SDN-IP application is developed to show possibility communication between BGP and SDN controller.

3.0 OF@TEIN SDN-CLOUD PLAYGROUND

3.1 Hyper-convergent SmartX Boxes

In OF@TEIN Playground, multiple SmartX Boxes are deployed in 11 sites across 9 countries since 2012. The last deployment attempts to convert the existing OpenFlow SDN-enabled playground with hyper-convergent SmartX Boxes into a SDN-Cloud-enabled playground [12]. The main idea is to provide cloud-based services on the top of playground infrastructure for executing the experiments, while still maintaining the OpenFlow-SDN paradigm to inter-connect distributed cloud data centers.

The overall design of hyper-convergent SmartX Boxes needs to consider both cloud and SDN aspects. For cloud aspect, open-source OpenStack cloud management software provides VM (virtual machine) instances and basic networking options for diverse tenants (i.e., testbed users/developers). For SDN aspect, several types of virtual switches (derived from open-source Open vSwitch [13]) are installed as shown in Fig. 2 while allowing developers to share them simultaneously. Each SmartX Box requires several dedicated interfaces, dedicated for P/M/C/D (power, management, control, and data) connections.

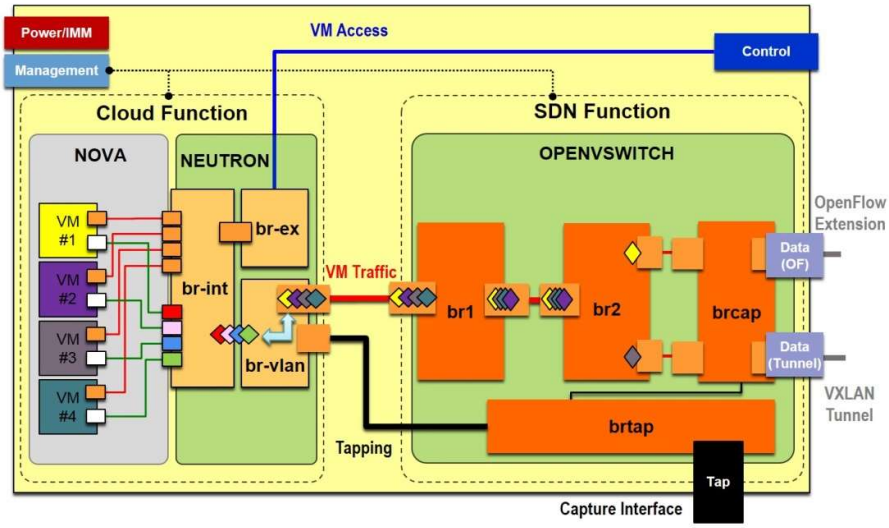

Fig. 2: SmartX Box for SDN-Cloud support.

Besides the flexible dedicated connections, there are no specific hardware requirements for hyper-convergent SmartX Boxes, any commodity hardware with reasonable computing, storage, and networking resources can be utilized. The resources specification only effect the capacity (e.g., total number of VM instances per flavor types) in the specific boxes. In addition, hardware acceleration support for virtualization and networking need to be considered.

By merging all the functionalities into a single box, it is easier to realize the scale-out capability of SDN-Cloud playground by simply adding more boxes to increase the resource capacity in order to form clustered (i.e., bigger) box playground.

3.2 Multi-regional OpenStack cloud with virtual tenants

OF@TEIN playground relies on diverse sets of physical underlay networks across multiple administrative domains. Thus, the multi-regional OpenStack cloud deployment is currently investigated as the deployment option as shown in Fig. 3, because it gives simple and common configurations for all regions (i.e., SmartX box sites) and depends less on layer-2 tunneled overlay networking among the sites. Despite of multi-regional cloud deployment, OF@TEIN playground is accessible by an integrated cloud management interface, enabled by web-

based OpenStack Horizon dashboard and by OpenStack Keystone account/token authentication.

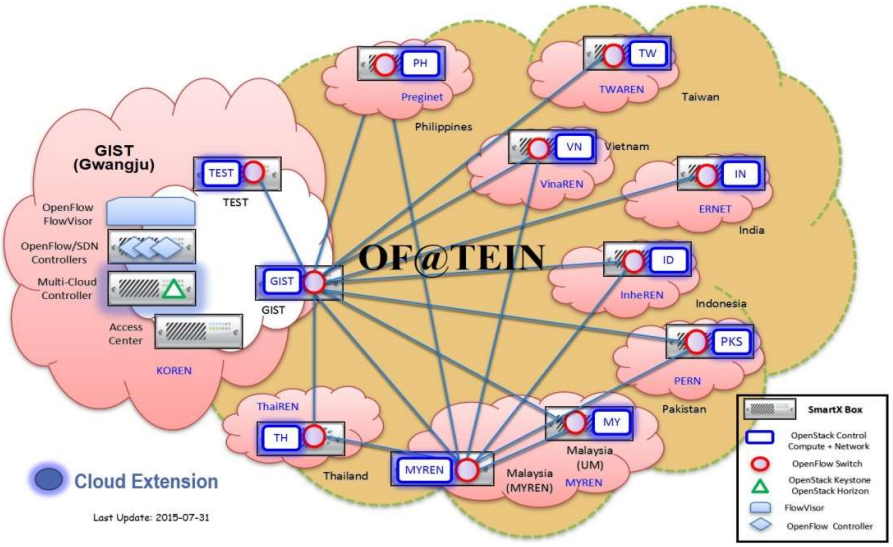

Fig. 3: OF@TEIN multi-regional OpenStack-enabled cloud.

3.3 OpenFlow SDN with virtual network slicing

SDN-based virtual functions consist of several virtual switches with different functionalities such as creating tenant networking topology, OpenFlow-based overlay networking, and tapping aggregation for visibility. The main challenge is how to accommodate cloud-based multi-tenant virtual networks (e.g., flat, VLAN, or tunnel network) in OpenFlow-SDN FlowSpace-based network slicing (e.g., IP subnet, VLAN ID, or TCP/UDP ports). Eventually VLAN-based multi-tenant traffic control (e.g., tagging, steering, and mapping) is chosen to integrate tenant-based cloud networking and slicing-based OpenFlow-SDN virtual network. However, integrating several virtual switches (e.g., br-int, br-ex, br-vlan, br-cap, and br-tap) is quite complex and challenging.

3.4 Example experiments with SDN-Cloud Playground

By manipulating both OpenStack cloud management and OpenFlow-enabled SDN control, the experiment, depicted in Fig. 4, is designed to verify the SDN-Cloud integration for deploying VLAN-based multi-tenant traffic control. First, we place VM instances in two different cloud regions and prepare basic connectivity for these VMs. These VMs are tagged by OpenStack Nova with specific tag ID. Second, OpenStack Neutron automatically maps and matches VLAN ID with SDN slice parameters. This allows VM inter-connection flows to be steered by the developer's SDN controller under the FlowVisor [14] supervision. The steering action inserts flow entries based on particular incoming and outgoing ports in developer's SDN switches, where several ports are mapped to other cloud regions. Finally, based on the destination, it maps to a specific pre-configured tunnel interface that is pre-configured by operator's SDN controller. The end-to-end connection testing between VMs is required to verify the consistency of flow tagging, steering, and mapping for specified testing flows.

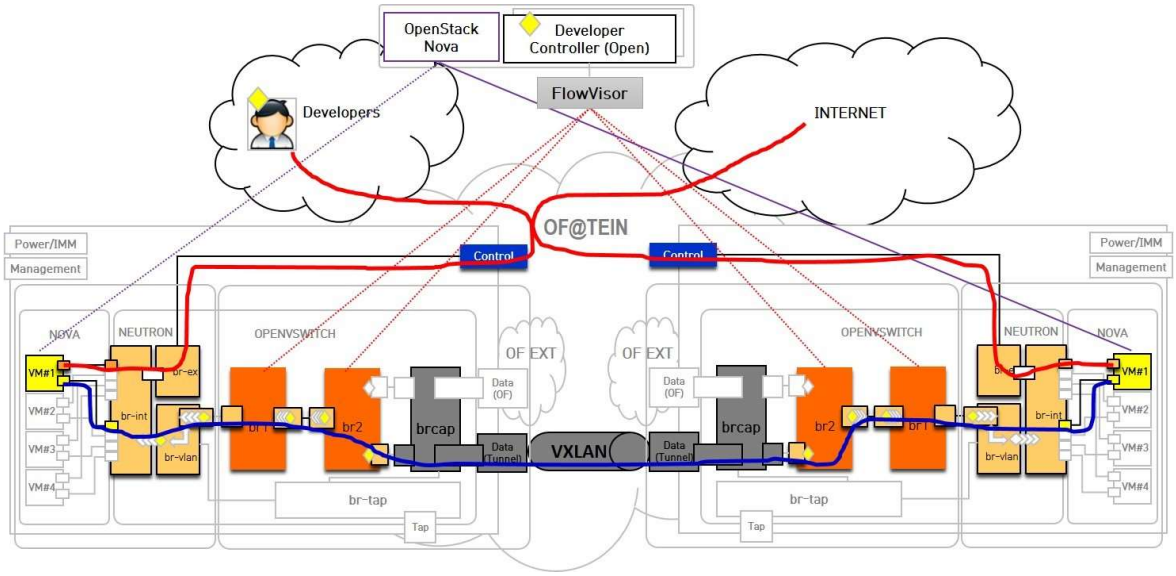

Fig. 4: OF@TEIN SDN-Cloud experiment verification.

4.0 EXPANDING OF@TEIN BY FEDERATING MULTI-DOMAIN SDN-CLOUD WITH SD-ROUTING-EXCHANGE

4.1 SD-WAN and SDX Requirements

As described in previous chapter, Google SD-WAN and Microsoft SWAN are deployed in order to address its special requirements on WAN configuration, carry more traffic, and support flexible network-wide sharing [5] [6]. Some of the challenges and critical decisions are summarized as follows:

- Low-cost router deployment for simple scaling while sacrificing hardware fault tolerance and requiring deep buffering plus large routing tables.
- Nearly 100% link utilization to withstand substantial capacity loss during link/switch failures.
- Centralized traffic engineering with scalable algorithm, where no existing protocols for site-to-site visibility for global resources allocation.
- Separation of hardware from software to break fate sharing between them.
- Atomic reconfiguration of a distributed system to update network forwarding state for adapting to traffic demand or network topology.
- Forwarding rules limitation in the switch hardware, which makes it hard to fully utilize network capacity.

From another perspective, SDX is motivated to solve several required applications that difficult to provide today by current internetworking systems. Those applications are compelling applications for wide-area traffic-delivery, such as:

- *Application-specific peering* for allowing traffic exchange between two AS only for specific applications
- *Inbound traffic engineering* for avoiding obscure technique to influence neighboring AS
- *Wide-area server load-balancing* for announcing anycast prefixes and selecting any hosting locations
- *Redirection through middlebox* for redirecting targeted subset of traffic

4.2 Carrier-friendly ONOS SDN controller

As mentioned above, ONOS has focused primarily on use cases for service provider networks that should meet demanding requirements for scalability, performance and availability such as up to 1M requests/second

throughput, 10~100 ms event processing latency, and 99.99% service availability. Initial ONOS demonstration shows the basic features such as controlling hundreds of virtual switches, programming end-to-end flows, dynamically adding switches and ONOS instances to the cluster, failover if ONOS instance shutdown, and rerouting for link failure. Unfortunately, several design choices in real deployment resulted in poor performance and usability. In the second development phase, ONOS focuses on the performance improvement in adding new switches by using optimized network I/O (e.g., kernel bypass) and 10 Gb/s *Infiniband* hardware on a RAMCloud cluster, where the combined optimization took 0.099 ms. Then, for re-routing 1,000 flows, experiments show the total latency 71.2 ms (median) and 116 ms (99 percentile). ONOS also implements a limited set of applications that focus on service provider solutions: i) simple proactive route maintenance, and BGP interfacing (SDN-IP); ii) traffic engineering and scheduling of packet optical core networks; iii) SDN control of next-generation service provider central offices and PoPs; and iv) remote network management, including virtualization of customer networks [7].

4.3 Hardware expansion: physical SDN switches and Router-Box (with Virtual Router Function)

In order to deploy/extend large-scale SDN-Cloud-enabled playground with carrier grade performance, physical hardware (e.g., OpenFlow switches and Router-Box) are tied with hyper-convergent SmartX Boxes. The switches extend the OpenFlow network coverage within SmartX Box (terminated on VXLAN tunnel interface inside the Box) toward outside of SmartX Box (i.e., Router-Box). The idea is to aggregate multiple individual hyper-convergent SmartX Boxes and others SDN-Cloud playgrounds in different sites into one big federated playground, and to extensively share the playground among all federation members. The on-going OF@TEIN hardware expansion for federated multi-domain SDN-Cloud playground is depicted in Fig. 5.

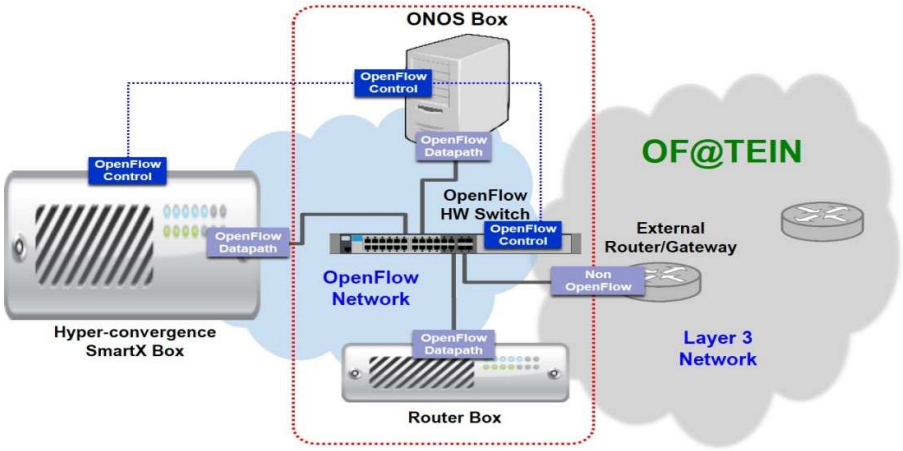

Fig. 5: OF@TEIN physical hardware expansion.

As shown in Fig.6, to support routing requirements for the expansion, low-cost virtual routers can be deployed inside SmartX Box, or outside SmartX Box with Router-Box which also deployed in a low-cost commodity hardware. This routing functionality is required to support distributed cloud data centers inter-connections in both previous works either SD-WAN or SDX. The detail deployment of the hardware switches and Router-Box will be discussed in Chapter 4.

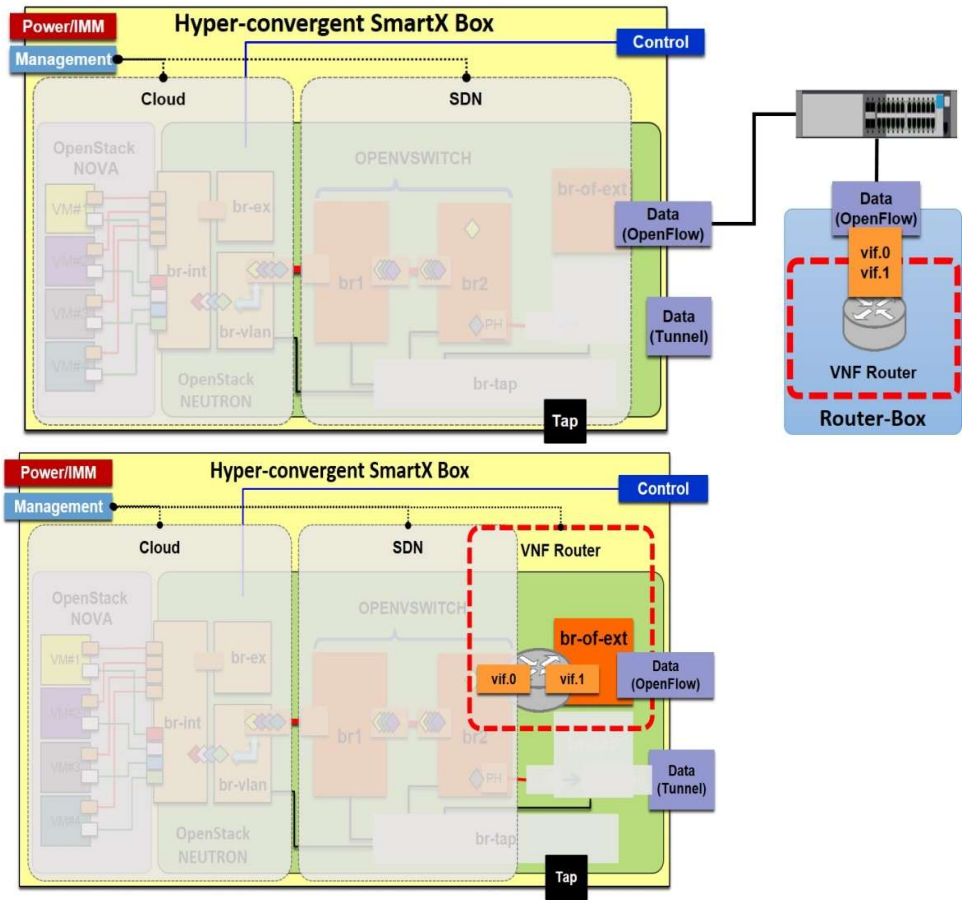

Fig. 6: Virtual Router Function addition outside SmartX Box with Router Box (above) or inside SmartX Box

4.4 Distributed and independent ONOS SDN controllers with shared BGP Speaker as SDN Control-plane

With carrier-focused ONOS, there are lots of use cases for the service providers. By utilizing the easy development and deployment of ONOS, federation of SDN control-plane should be easy to deploy in geographically distributed networking environment with different administrative domain. When the OpenFlow switches are linked to an ONOS controller, they can be flexible managed and utilized for many different applications such as segment routing, packet optical, NFV, SDN-IP, or IP RAN. Specifically for SDN-IP applications is very important for this work because it provides standard BGP routing exchange capabilities between BGP routers inside and outside SDN domain (with and without OpenFlow supports). The BGP router (either inside or outside SmartX Box with Router-Box) used to collect network information and translate it into BGP routes update. Those BGP routes advertised to the BGP speaker which has peering with all others BGP speakers in others domain. BGP speaker also peer with BGP listener of SDN-IP application inside the ONOS Box, so ONOS can learn all the routes in the network. With help of specific SDN-IP configuration, ONOS controller knows the inter-connection between routers and advertised network, and then inserts respectively flow entries in the SDN switches with help of the intent networking. As a result, BGP-based routing exchange can be used to build flow information in underlying hardware infrastructure. In other words, we can use BGP as control-plane for building OpenFlow SDN data-plane.

5.0 OF@TEIN SDN-CLOUD PLAYGROUND WITH SD-ROUTING-EXCHANGE

5.1 Preliminary experiment for OF@TEIN SD-WAN-enabled Playground

This preliminary SD-WAN experiment tries to provide inter-connections between two sites of SDN-Cloud playground under programmable WAN approach, which utilizes tunnel-based layer-2 inter-connections between the sites. This preliminary experiment needs specific installation and configuration of ONOS controller, SDN-IP applications, BGP speakers, BGP routers, and additional OpenFlow switches. We deploy single ONOS box (on physical commodity hardware) in one site without any ONOS clustering option for controller high availability, because the stability of multi-site ONOS cluster deployment still not verified. High availability addressed by providing one BGP speaker in each site (also run on physical commodity hardware) which communicate with single ONOS controller. In order to extend the SDN-Cloud playground with routing function, light-weight VMs (i.e., containers) are deployed inside hyper-convergent SmartX Boxes to provide virtual routers. It uses *Linux container* (LXC) to create the containers and install *Quagga* [15] open-source routing suites inside the containers. Each router in both sites configured as BGP router with specific BGP configuration for that site such as AS (autonomous system) number, and advertised network prefix. BGP speaker make peering to all BGP routers, collect BGP routes and send it to BGP router listener inside ONOS SDN-IP applications. BGP routers and speaker are logically connected to OpenFlow switches, controlled by the ONOS controller.

The implementation of SD-WAN prototype is still limited and requires further testing/measurement as it still unaware of underlying infrastructure. The preliminary verification is tested for end-to-end communication among VMs in two cloud sites. Similar experiment as depicted in Fig.4 is required to verify matching and consistency of virtual networking configuration between clouds, SDN infrastructure, and VNF functionality for building inter-DC WAN through SD-WAN infrastructure. The basic measurements for this SD-WAN prototype infrastructure are end-to-end delay (i.e., latency) and WAN availability for end-to-end inter-connection.

In line with continuous playground deployment and more affiliated sites in this experiment, the current approach is not suitable for existing underlying network infrastructure where managed by different network administrator. The hardware/software deployment will be different and also configuration/privilege is more diverse. In order to overcome this situation, it required to look forward another approach for inter-connecting distributed multi-domain SDN-Cloud playground including flexible routing through inter-domain network (i.e., SDX – Software Defined eXchange Point) [7]. Unfortunately, this work aims to address problem in the service provider level by providing flexible routing policies in the internet exchange point, is also either not suitable for OF@TEIN environment. Therefore, sharpening the requirements and scope of the work is required to define appropriate approach for addressing those issues.

5.2 OF@TEIN Federated SDN-Cloud Playground Requirements

As described above, SD-WAN and SDX have different purposes of implementation as well as different specific requirements criteria related with service availability and traffic load balancing. To fulfill those purpose and criteria, centralized view and control of the network infrastructure in SD-WAN, or combining multiple networking policies for single device in SDX is required. OF@TEIN also have a different purpose but partially of the requirements are similar with SD-WAN and/or SDX. Our purpose is to federate distributed programmable SDN-Cloud resources under different administrative domain, using existing interdomain routing system with flexible policies for all the members. As the resources do not belong to single domain (e.g., company, organization) who can control WAN links end-to-end, and also do not owned by IXP (Internet Exchange Point) who can control routing coming different ASes, so our requirements modified as follows:

- Low-cost deployment with scaling capability and supporting of open networking solution (hardware/software)
- Providing configuration flexibility in selecting peers and forwarding path between sites amongst all OF@TEIN SDN-Cloud federated member
- Providing reasonable inter-connection redundancy and load-balancing between sites based on advertised network prefix in traditional layer-3 routing information from OF@TEIN underlying infrastructure
- Avoiding layer-2 tunnel inter-connection across different administrative domain between sites because it is not underlay-aware and has scalability issues

Those all requirements are adopted either SD-WAN or SDX, as the deployment is considering both approaches for building OF@TEIN federated multi-domain SDN-Cloud Playground. In order to distinguish with both previous works, this approach is called as *Software-Defined Routing Exchange* (SD-Routing-Exchange), where it utilizes routing exchange for building federated SDN-Cloud infrastructure. The detail SD-Routing-Exchange approach as illustrated in Fig. 7.

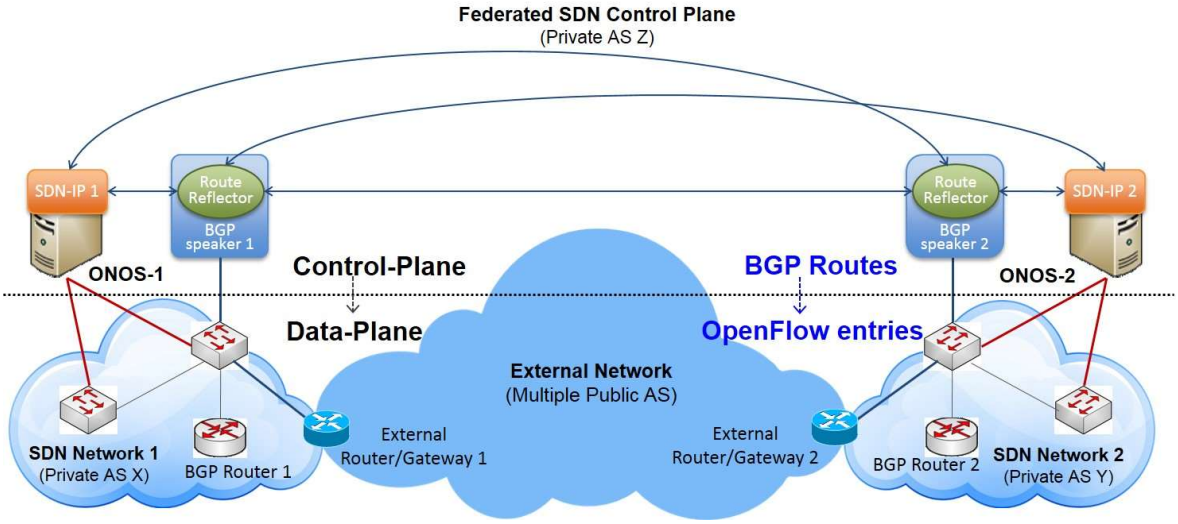

Fig. 7: Software-defined Routing Exchange: Concept.

5.3 OF@TEIN SD-Routing-Exchange Deployment

This section will explain about how to deploy SD-Routing-Exchange in OF@TEIN federated multi-domain SDN-Cloud playground. As a prototype, its deployed in three different sites (i.e., GIST Korea, UM Malaysia, and NCKU Taiwan) which are connected to TEIN network through three different RENs (i.e., KOREN, MYREN, and TWAREN). Those three member sites have their own deployment consideration (hardware and software), but all of them will have similar boxes and functions for supporting multi-domain federation deployment. The detail discussion based on the boxes/functions rather than per sites basis.

5.3.1 Deployment Topology

For designing and developing a distributed SDN-based testing environment, some Future Internet testbeds may use network technologies to cut a logical network for experiments, such as VLAN and Tunneling. However, for exploring the idea of applying BGP-based layer-3 SD-Routing-Exchange, the developers need to have fully control ability on routing configuration between sites. Therefore, the principle of designing SD-Routing-Exchange is to provide a gateway to underlying interdomain network at the edge of SDN-based virtual network for each site, and then use the existing interdomain network as the inter-connections to deliver developers testing traffic (i.e., SDN data-plane). In this way, the SD-Routing-Exchange deployment is able to act as hybrid (i.e., OpenFlow and non-OpenFlow support) overlay architecture between the sites without interfering underlying network architecture and configuration. Fig. 8 shows logical topology for current prototype deployment of three pilot sites with three different R&E networks. These sites play the role as Point-of-Presence (PoP) of SD-Routing-Exchange federated infrastructure for SDN-Cloud playground.

5.3.2 Site Deployment

There are four basic components in each sites: ONOS Controller, OpenFlow Switch, Speaker and BGP Router.

ONOS Controller. It is core component to control the network infrastructure (i.e., OpenFlow switches) of each site. Special application called SDN-IP [16] application is activated for helping controller to configure packet forwarding in data-plane (i.e., intents/flows), based on BGP routing information.

OpenFlow switch. To physically connect devices (e.g., Speaker and BGP Router) of each site and fully-controlled by the ONOS controller using OpenFlow protocol. ONOS can handle many different hardware.

Speaker and BGP router. Quagga-based router is used to collect network information and translate it into routing updates. BGP router each site peers only with BGP speaker in respective site, while all the speakers are peering with each other over existing layer-3 network to exchange routing information. It also peers with BGP listeners inside SDN-IP application for helping ONOS controller translating into intents or flows in the OpenFlow switch(es).

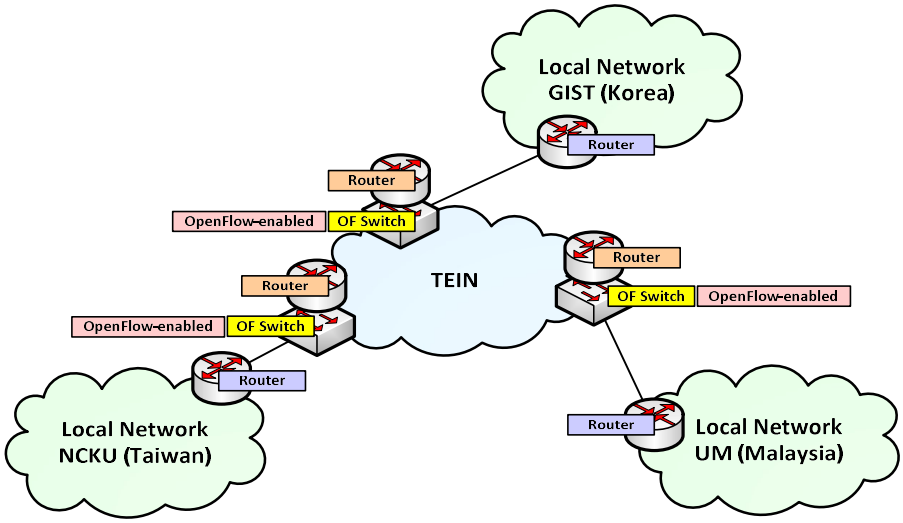

Fig. 8: Logical topology of SD-Routing-Exchange in OF@TEIN.

5.3.3 Configuration

In SD-Routing-Exchange configuration, it has two group of IPs for control-plane and data-plane. Control-plane IP group is used to communicate between ONOS controller, OpenFlow switch, and speaker. The control messages (e.g., OpenFlow control message and BGP packets) are transmitted with these IP addresses. Data-plane IP group is utilized by end-hosts, BGP Router, and gateway for transmitting the user traffic. It is resulting communications separation for control-plane and data-plane, and able to be logically isolated without interfering. The most important configuration for control-plane is BGP peering configuration between speaker, router, and listener. Speaker-router peers using standard eBGP protocol, because router needs to advertise site local IP prefix with specific AS (autonomous systems) number. Thus, speaker-speaker/speaker-listener using standard iBGP protocol, because it acts single federated network with single AS number and intensively shared the route information. In addition, ONOS controller and OpenFlow switch control channel configuration required for selecting control interface and supported protocol version. For the data-plane configuration, specific configuration of SDN-IP application define the peering IP address, physical network address, and interface between speaker and router. It is required for correct translation from network prefix into flows including source/destination's interfaces and destination's MAC address.

Especially for SD-Routing-Exchange deployment, some specific configuration has been made differ from common ONOS SDN-IP configuration. The deployment needs to accommodate "peering" configuration between BGP-enabled router and non-BGP-enabled gateway (i.e., router with only default/static route), and to manipulate indirect/unreachable next-hop address for routes learned from other sites.

5.3.4 Verifications

The main verifications of this deployment are the translation of advertised/learned routing information into flow entries by ONOS controller, and site-to-site user traffic delivery through hybrid datapath over underlying interdomain network. This procedure relies on the ONOS SDN-IP application, since it receives BGP routes advertisement from speaker, and then check the reachability of the next-hop address for specific advertised prefixes. These routes are used to generate intents (a policy-based specification for describing object or rule) in ONOS controller. Finally, the installed intents are translated into flow entries for corresponding OpenFlow switches to forward the user traffic.

```
onos> bgp-routes
   Network            Next Hop        Origin LocalPref       MED BGP-ID
   140.116.154.144/29 140.116.154.150    IGP       100         0 140.116.154.148
                      AsPath 65031
   61.252.52.0/24     61.252.52.54       IGP       100         0 61.252.52.53
                      AsPath 65001
   203.80.21.48/29    203.80.21.54       IGP       100         0 203.80.21.53
                      AsPath 65002
Total BGP IPv4 routes = 3
```

Fig. 9: Advertised routes learned on ONOS Controller.

Application ID	Key ▾	Type	Priority	State
35 : org.onosproject.sdnip	61.252.52.128/25	MultiPointToSinglePointIntent	225	Installed
(No resources for this intent)				
Details: Selector: [IPV4_DST{ip=61.252.52.128/25}, ETH_TYPE{ethType=ipv4}]Treatment: [ETH_DST{mac=90:E2:BA:1E:34:30}]Constraints: [org.onosproject.net.intent.constraint.PartialFailureConstraint@99d714c] Ingress=of:0065a0b3ccf32380/16 , Egress=of:0065a0b3ccf32380/15				

Fig. 10: Route is translated into intent by ONOS SDN IP.

10133100202078151	36	0	0	225	0	true	Added
Criteria: IN_PORT{port=16}, IPV4_DST{ip=61.252.52.128/25}, ETH_TYPE{ethType=ipv4}							
Treatment Instructions: ETH_DST{mac=90:E2:BA:1E:34:30}, OUTPUT{port=15}							

Fig. 11: Flow is installed in the switch based on installed intents.

Fig. 9 shows an instance of learned routes on ONOS controller during the deployment verification. Fig. 10 & Fig.11 depicts intent and flow-entry are generated by learned route by ONOS controllers. The route for remote site is forwarded to specific MAC address and port, which are the local gateway information to reach remote gateway through underlying interdomain network. Since the packet delivery amongst sites is forwarded by underlay interdomain networks, currently only single site-to-site routing path is selected. However, if multiple R&E networks are able to stitch in each site, the SD-Routing-Exchange also possible to do multipath routing selection with multiple gateways configuration.

In next section, several experimental measurements for initial performance evaluation and analysis are presented. The purposes of these measurements are designed to observe and evaluate SD-Routing-Exchange stability and scalability. The results also show the real operation aspect of SD-Routing-Exchange in OF@TEIN playground.

6.0 MEASUREMENT AND ANALYSIS

In order to verify and analyze the deployment, several measurement scenarios are executed depending on parameters that required to be analyzed. There are two measurements categories which are operational (continuous) measurement and benchmarking scenario. The operational measurement is required to verify the stability of the deployment over long period of time, and benchmarking is required to verify the response of the deployment for a specific conditions or failure.

6.1 Operational (Continuously) Measurement

The stability of SD-Routing-Exchange deployment can be measured from two different aspects: control-plane and data-plane. Control-plane measurement will continuously monitor the operational status of BGP routing exchange information between BGP routers in different sites. The data-plane measurement is related with end-to-end inter-connection between one user in one federation site and another user in another federation site.

6.1.1 BGP Control Plane Performance

Latency and Availability. In order to monitor the status of the overlay network, measuring the quality/status of underlay networks is necessary. The availability is an important index in evaluating network quality. Smokeping [17] is a suitable solution to collect the availability of connections among nodes, because it is able to setup a node list required to be checked and it measures continuously by assigned methods (e.g., SNMP, Ping). Currently, Smokeping monitoring node located at one physical box in NCKU site, but it plan to move into "independent" location (e.g., public/private cloud infrastructure) for avoiding site dependencies. The ping with small packets (56 bytes) is used to evaluate the system, and measured results are shown in Figs. 12a, 12b, and 12c. The observation of control-plane stability from long-term measurement results shows the latency between monitoring node and BGP speaker at GIST is about 88 ms, at UM is about 249 ms. These values reflect the network latency (i.e., RTT – *round-trip time*) from Taiwan to Korea and Malaysia. Due to the intermediate network of SD-Routing-Exchange is based on interdomain networks, the performance (especially congestions in short periods) between TEIN and national R&E networks influence the measurement results. Furthermore, during developments and experiments, hardware and software involved in the deployment would also shut down or reset as necessary. As a result, some jitter values are shown in the MRTG charts. Overall, the results still show that the underlay network is stable and open for peering and exchanging routing information. On the other side, the summary of the latency between the sites is also another important concern in measurement. Table 1 concludes the measured ping latency between BGP speaker in each sites and monitoring node for about 50 days. It provides considerations in routing path selection for achieving routing optimization for each site. Due to the current circumstance, the available paths among sites are not able to satisfy large-scale and routing redundancy configurations, while this issue is planned to be studied in future development.

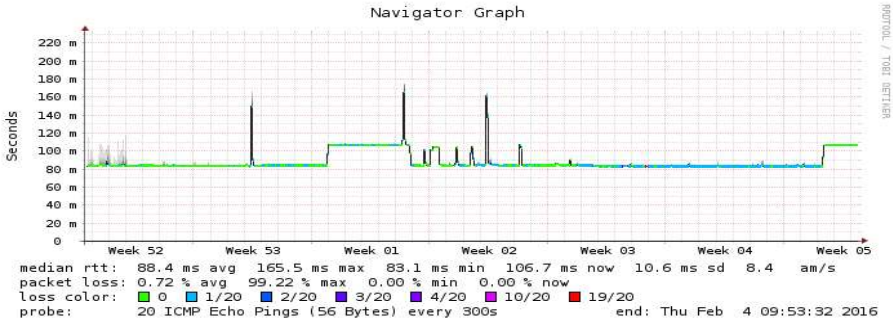

Fig. 12a: Measurement result of GIST site.

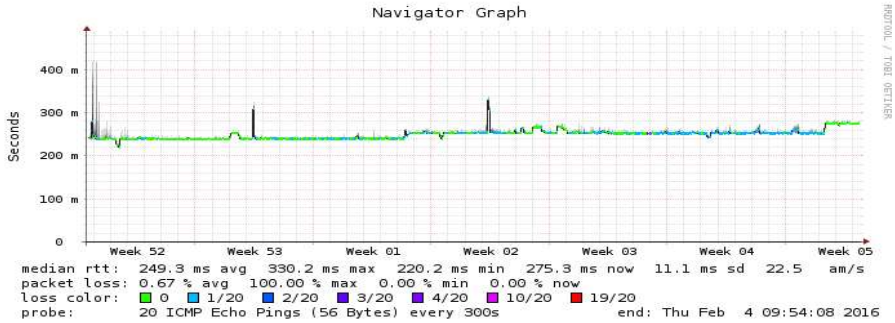

Fig. 12b: Measurement result of UM site.

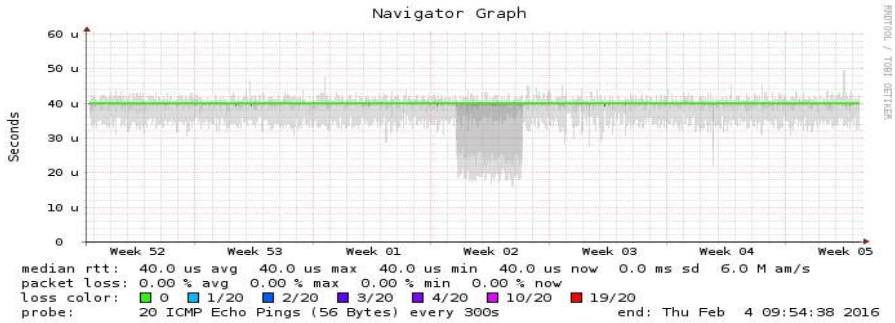

Fig. 12c: Measurement result of NCKU site.

Table 1: Ping latency between BGP speakers and monitoring nodes

ICMP RTT (ms)				
Node	**Target Site**	**Avg.**	**Max**	**Min**
BGP Speaker	GIST	88.40	165.50	83.10
BGP Speaker	UM	249.30	330.20	275.30
BGP Speaker	NCKU	0.04	0.04	0.04
Packet Loss Rate (%)				
Node	**Target Site**	**Average**		
BGP Speaker	GIST	0.72		
BGP Speaker	UM	0.67		
BGP Speaker	NCKU	0.00		

BGP sessions. For establishing a graphical monitoring mechanism, the Cacti tools [18] is used to measure the performance of control-plane functions (e.g., BGP speaker and SDN-IP BGP listener) in SD-Routing-Exchange. The functions configured inside the ONOS Box in each site are SNMP-enabled to allow Cacti collect its operational statistics. Especially for monitoring BGP session of the routers, the Cacti community [19] had proposed a Linux-script-based plug-in to collect BGP information from router CLI and translated into readable data for MRTG chart. Therefore, the BGP sessions for all functions are recorded completely, and routing information are presented in MRTG charts. Figs. 13a, 13b, and 13c show the monitoring charts of BGP sessions (approx. in 5 weeks) including "down" status when the hardware or software are offline for making adjustment and testing. These monitoring results are helpful for OF@TEIN operators to monitor the stability of SD-Routing-Exchange operational continuously.

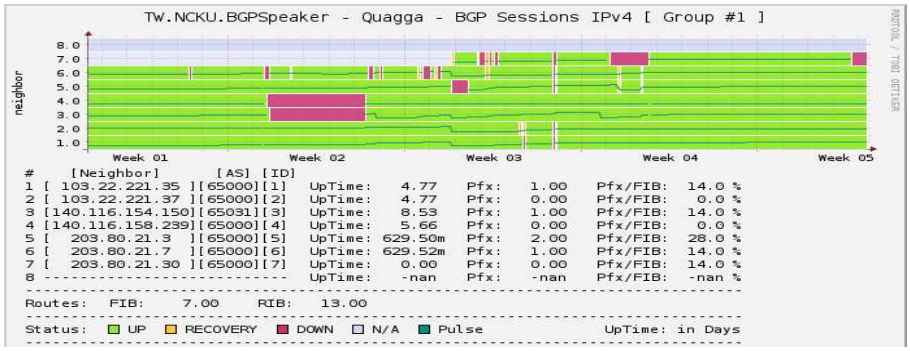

Fig. 13a: BGP sessions in SD-Routing-Exchange.

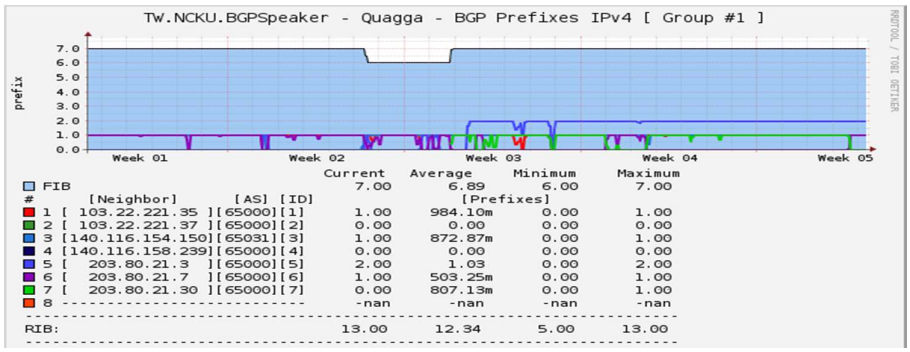

Fig.13b: BGP prefixes in SD-Routing-Exchange.

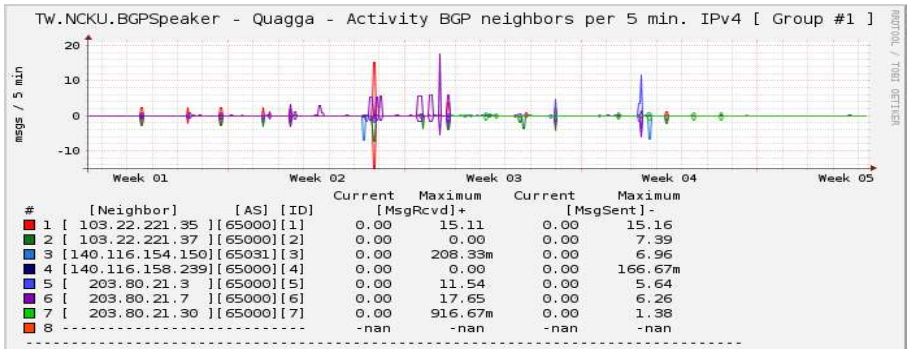

Fig. 15c: BGP neighbors in SD-Routing-Exchange.

6.1.2 End-to-end Data-Plane Performance

Latency. In order to monitor the end-to-end data-plane performance, one VM instance or host in each site is prepared for continuously measurement. For the end-to-end latency between VM instances or hosts, simple ping application is used with additional time recorder and result summary. Table 2 shows the detailed result of latency between the sites.

Table 2: End-to-end ping latency between VM instances/hosts

ICMP RTT (ms)				
Source	**Destination**	**Min.**	**Avg.**	**Max.**
GIST	UM	80.898	81.105	81.105
GIST	NCKU	127.274	127.640	214.435
UM	NCKU	213.486	215.516	681.778
Packet Loss Rate (%)				
Source	**Destination**	**Average**		
GIST	UM	~ 0 %		
GIST	NCKU	~ 0 %		
UM	NCKU	~ 1 %		

Throughput. It is similar setup with latency measurement, but the tool utilized is different. Iperf-based or scheduled file transfer is used to send the traffic into maximum capacity of this end-to-end data-plane inter-connection continuously. Table 3 shown the detail result of throughput between the sites.

Table 3: End-to-end throughput between VM instances/hosts

Iperf throughput TCP (60 seconds)		
Client	**Server**	**Throughput (Mbps)**
GIST	UM	0.783
NCKU	GIST	176
NCKU	UM	0.811
Iperf throughput UDP (60 seconds)		
Client	**Server**	**Throughput (Mbps)**
GIST	UM	1.050
NCKU	GIST	244
NCKU	UM	0.941

The results show the latencies in ICMP ping and the throughputs in Iperf measurement are close to the latencies and throughputs of underlying infrastructure. By analyzing the performance comparison results and discussing configuration details with NREN operators, leveraging BGP-based SDN control-plane for controlling hybrid SDN data-plane is required minimal configuration and minimal performance impact to the underlying infrastructure.

6.2 Benchmarking

The important parameters to be observed from SD-Routing-Exchange are the convergence time of BGP protocol as the control-plane, and the convergence time for hybrid data-plane that used for the users to deliver the traffic.

6.2.1 Redundancy

Before discussing about the convergence time, the first important verification is the impact of the BGP peering failures for both control and data planes. So far the inspection only for single peering failure as the work still needs to be expanded. The result of experimental verification is shown in Table 4.

Table 4: Verification: Single peering failures

(Single) Peering Failures	Control Plane	Data Plane	Notes
ONOS SDN IP - Speaker	Failed	Failed	***ONOS SDN IP peering only one speaker in the same site***
Speaker - Speaker	OK	OK	***Speaker can learn from two speakers from different sites***
Speaker-Router	Failed	Failed	***Router only peer with one speaker in the same site***

6.2.2 Convergence Time

Control-plane. Similar with common BGP router and protocol implementation, the first criteria for the control-plane performance is convergence time amongst all participated routers in the network for different number of routes. In this measurement scenario, it measured the control-plane performance in two different sites (i.e., local and remote sites) and two different BGP entities (i.e., speaker and router) with number of routes vary between 100 routes till 100000 routes. For this purpose, "bgp simple" tool provided by [20] is used to inject "real" internet routes (e.g., RIPE dump routes) into our deployment. The starting time is when the BGP tool started to advertise prefix to the local speaker through eBGP peering, and the finish time is the last prefix received and installed in the each router routing table. It will mixed between iBGP and eBGP convergence performance, but the same timer values are used and no filtering policy applied for both mechanism.

Table 5: Route convergence time versus number of routes

Router Function	Convergence Time for # of Routes (seconds)			
	100	1000	10000	100000
Local Router (BGP Tool)	10.1080	10.4034	13.3905	43.2225
Local Speaker (Quagga)	12.9033	13.6955	16.1268	46.3295
Remote Speaker (Quagga)	18.1679	18.8889	19.9218	50.3101
Remote Router (Quagga)	18.1598	18.9038	20.0089	50.3562

Table 5 shows the convergence time for different router function in SD-Routing-Exchange. As expected, more route prefixes required more time to converge and the furthest router is the last router to converge. However, the maximum convergence time is ~50 seconds which still under one minute. Also, the different convergence time between the first router (BGP tool) and the last router (remote router) are almost the same in between 7 ~ 8 seconds.

Table 6: End-to-end data-plane convergence time

Source	Destination	Convergence Time (seconds)
GIST	UM	6.289
GIST	NCKU	1.887
UM	NCKU	4.126

Data plane. For this measurement, the observation of detailed time requirement for establishing end-to-end inter-connection to deliver the traffic, is required. The starting measurement time is recorded time in the router

when it starts to advertise the prefix. The end-time is recorded time in user's VM instance/host when the first packet is going through the inter-connection from one site to other site. The detail results are shown in Table 6.

From the end-to-end data-plane convergence time measurements, the interesting results are shown that sites with better latency not always better in convergence time. GIST and UM has lowest latency ~80 ms but it has bigger convergence time ~6 seconds, while GIST and NCKU has best convergence time ~2 seconds even though the latency is ~127 ms. The preliminary analysis is GIST and NCKU has better TCP performance (i.e., response time) than GIST and UM as shown in Table 3 for Iperf TCP throughput performance. It is well-known that BGP peering and updates rely on TCP-based communication between the routers, while latency relies on the ICMP messages.

7.0 CONCLUSION AND FUTURE WORKS

OF@TEIN distributed multi-domain SDN-Cloud infrastructure can be inter-connected by using proposed Software-defined Routing Exchange for executing end-to-end experiments. The proposed approach is the first transition step from SD-WAN towards SDX (software-defined exchange). The deployment in three pilot sites shows the capability of ONOS (open network operating systems) controllers on utilizing BGP as SDN control-plane for hybrid OpenFlow SDN data-plane with flexible networking control and reasonable redundancy. As a result, end-to-end underlay-aware overlay inter-connections with minimal configuration interference and performance impact on underlay infrastructure.

ACKNOWLEDGEMENT

General thanks for OF@TEIN collaborators and respective network administrator involved in the collaboration work. This work was supported in part by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. B0190-15-2012, Global SDN/NFV Open-Source Software Core Module/Function Development). This work is also supported in part by 'Software Convergence Technology Development Program', through the Korea Ministry of Science, ICT and Future Planning (S1070-15-1071). The research is also partially supported by National Applied Research Laboratories (NARLabs) (No. 03103A7300) and TWAREN SDN testbed from National Center for High-Performance Computing (NCHC) for which authors are grateful.

REFERENCES

[1] J. Kim, B. Cha, J. Kim, N. L. Kim, G. Noh, Y. Jang, H. G. An, H. Park, J. Hong, D. Jang et al., "OF@TEIN: An openflow-enabled SDN testbed over international smartx rack sites," *Proceedings of the Asia-Pacific Advanced Network*, vol. 36, pp. 17-22, 2013.

[2] O. Sefraoui, M. Aissaoui, and M. Eleuldj, "OpenStack: toward an opensource solution for cloud computing," *International Journal of Computer Applications*, vol. 55, no. 3, 2012.

[3] M. Banikazemi, D. Olshefski, A. Shaikh, J. Tracey, and G. Wang, "Meridian: an SDN platform for cloud network services," *Communications Magazine, IEEE,* vol. 51, no. 2, pp. 120-127, 2013.

[4] M. Mechtri, I. Houidi, W. Louati, and D. Zeghlache, "SDN for inter cloud networking," in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for IEEE*, 2013, pp. 1-7.

[5] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu et al., "B4: Experience with a globally-deployed software-defined WAN," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 3-14.

[6] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven WAN," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 15-26.

[7] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett, "SDX: A software defined internet exchange," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4, pp. 551-562, 2015.

[8] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O. Connor, P. Radoslavov, W. Snow et al., "ONOS: towards an open, distributed SDN OS," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 1-6.

[9] M. Berman and M. Brinn, "Progress and challenges in worldwide federation of future internet and distributed cloud testbeds," in *Science and Technology Conference (Modern Networking Technologies) (MoNeTeC), 2014 International. IEEE*, 2014, pp. 1-6.

[10] K. Phemius, M. Bouet, and J. Leguay, "DISCO: Distributed multi-domain SDN controllers," in *Network Operations and Management Symposium (NOMS), 2014 IEEE. IEEE*, 2014, pp. 1-4.

[11] F. Balus, N. Bitar, K. Ogaki, and D. Stiliadis, *"Federated SDN-based controllers for NVO3," IETF*, 2013.

[12] A. C. Risdianto, J. Shin, and J. Kim, "Building and operating distributed SDN-Cloud testbed with hyper-convergent SmartX boxes," In International Conference on Cloud Computing (pp. 224-233). Springer International Publishing.

[13] B. Pfaff, J. Pettit, K. Amidon, M. Casado, T. Koponen, and S. Shenker, "Extending networking into the virtualization layer." in *Hotnets*, 2009.

[14] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, *"FlowVisor: A network virtualization layer,"* OpenFlow Switch Consortium, Tech. Rep, pp. 1-13, 2009.

[15] P. Jakma and D. Lamparter, "Introduction to the quagga routing suite," *Network, IEEE*, vol. 28, no. 2, pp. 42-48, 2014.

[16] "*ONOS wiki, SDN-IP use cases*," 2016. [Online]. Available: https://wiki.onosproject.org/display/ONOS/SDN-IP

[17] T. Oetiker and N. Tyni, "*The smokeping website*," 2005. [Online]. Available: http://oss.oetiker.ch/smokeping/index.en.html

[18] "*Cacti: the complete rrdtool-based graphing solution*," 2010. [Online]. Available: http://www.cacti.net/

[19] "*Monitoring of BGP session via quagga daemon in linux*." [Online]. Available: http://forums.cacti.net/viewtopic.php?f=12&t=51271

[20] "*Networks and mobile system projects, bgp tools*." [Online]. Available: http://nms.lcs.mit.edu/software/bgp/bgptools/