SDX Architectures: A Qualitative Analysis

Joaquin Chung, Henry Owen
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: joaquin.chung@gatech.edu, henry.owen@ece.gatech.edu

Russell Clark
School of Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: russ.clark@gatech.edu

***Abstract*—A novel internetworking paradigm, software-defined exchange (SDX), allows multiple independent administrative domains to share computing, storage, and networking resources. Although SDX is a recent concept, the concept has already been used by many disciplines. For example, while cloud computing researchers use it to add network virtualization resources to their inter-clouds, networking researchers use it to insert SDN technologies into the networking exchange infrastructure. Despite the various uses, the efforts of the various disciplines that share networking resources converge to a single point enabled by virtualization and separation of control and data planes. This paper presents a survey of the most relevant SDX studies from various research areas, focusing on their architectures. The paper defines a taxonomy for the SDX, provides generalized architecture models, and concludes by presenting a qualitative analysis of the architectures that focus on the scalability, resilience, peering technologies, and deployment of SDX.**

I. Introduction

A novel internetworking paradigm that allows sharing among multiple independent administrative domains is software-defined exchange (SDX). Approaches to the use of SDX, which allows stakeholders to share computing, storage, and networking resources, depends on the background of an observer. Although SDX is a new concept, it has already been used in various disciplines. For example, while cloud-computing researchers use it to add network virtualization resources to their inter-clouds, networking researchers use it to insert software-defined networking (SDN) technologies into the networking exchange infrastructure [1][2]. Despite its various uses, the efforts of separate disciplines that share networking resources converge to a single point enabled by virtualization and SDN.

Researchers on the SDX panel during GEC 23 conference presented a range of ideas and use cases for SDX [3]. Taking into account the resources exchanged, we can classify SDX solutions as follows: (1) Layer-3 SDXs, which exchange BGP updates in Internet exchange points; (2) Layer-2 SDXs, which exchange multi-domain Ethernet circuits in research and education networks; and (3) SDN SDXs, which interconnect SDN islands. Likewise, a cloud-service SDX enables the orchestration of computing and storage resources for inter-clouds. In the same manner, the applications of SDX will depend on previous classifications. For instance, a Layer-3 SDX should effectively augment or enrich BGP policies that can be configured. A Layer-2 SDX would be used to automate the provisioning of Ethernet circuits. An SDN SDX will allow the deployment of end-to-end SDN policies between two or more SDN islands.

978-1-5090-2246-5/16/$31.00

From an architectural standpoint, an SDX could be classified as centralized, which either controls a slice of the network or uses a hierarchy of SDN controllers, or peer-to-peer, which takes advantage of communication protocols that implement a west-east interface between SDN controllers. The best choice of an SDX architecture will depend on its size, and operational requirements of the exchange. Although size and operational requirements of a network exchange are important metrics that define the feasibility of an SDX deployment, we must also consider scalability, resilience, and peering technologies in our decision process.

The contributions of this paper are as follows:

- A survey of existing SDX projects
- A taxonomy of SDX architectures
- A generalized architecture and comparison of SDX types
- A qualitative analysis of the presented architectures.

This paper is organized as follows. Section II presents a background of the network infrastructures that inspired most of the SDX work, and Sections III, IV and V discuss Layer-2, Layer-3, and SDN SDXs, respectively. Section VII-A discusses a service model approach, Section VII presents a qualitative analysis and deployment considerations, and Section VIII concludes.

II. Background

The infrastructures that gave birth to some of the many faces of SDX are discussed in this section. Starting with Internet exchange points (IXP), it reviews studies that added SDN-enabled switching fabrics to IXP infrastructures and then Inter-cloud studies that investigated adding network virtualization technologies to their federated or hybrid clouds. Then it discusses research and education networks, which started connecting federated SDN testbeds, an SDX taxonomy, and a list of use cases for SDX.

A. Internet Exchange Points (IXP)

An Internet exchange point (IXP) allows Internet service providers (ISPs) and content providers (CPs) to exchange Internet traffic by peering on a common facility or a physical infrastructure using BGP. This common facility can be as

simple as a single switch connecting the participants border routers and as complex as a globally distributed architecture based on Ethernet switching fabric[4]. Chatzis et al. [4] identified two types of IXPs: the European non-profit business model and the North-American for-profit business model. The openness of the European model has attracted the attention of networking researchers, who identified opportunities in the following areas: a richer peering fabric, acquisition of a detailed understanding of traffic flow, measurement-driven networking research, content distribution, cloud providers, IXP-specific traffic engineering, switching architectures for IXPs, and networking operations [4].

Interestingly, the largest IXP can handle daily traffic volumes as large as those reported by the largest Tier-1 ISPs. As a result, several studies have tried to map IXP peering connections [5][6]. In [5], Augustin et al. attempted to map all IXPs by using a complete input dataset composed of publicly available information (e.g., IXP databases, websites, the Internet Routing Registry). The authors looked for IXPs in available datasets (e.g., CAIDA, DIMES, and PlanetLab) and proposed new methods of generating datasets that use traceroute traces from strategically located Looking Glass routers, source routing, and BGP tables. Their process detected 223 IXPs out of 278 and showed that most of the remaining undetected IXPs are inactive or not visible through a traceroute. Similarly, Ager et al. [6] attempted to map peer-to-peer connections and the traffic matrix of a large European IXP. In this work, the authors used a nine-month sFlow trace, IP-to-MAC address mapping, traceroutes, and external database information to analyze the IXP. Their results showed that 67% (or more than 50,000) of all possible peer-to-peer connections between the 396 IXP members were actually established, compared to the 40,000 peer-peer AS links reported in the Internet by the time of their publication.

Modern IXPs are more than just an Ethernet fabric. Most include services such as a route server (RS), which allows multilateral BGP connections. The authors of [7] used an RS to collect data and to map a large IXP. The main lesson of this study is that the majority of peering links in an IXP consist of multi-lateral peering through an RS. However, the bulk of traffic is carried out by bilateral peering links. The authors defined directions for innovation in inter-domain routing. For example, todays RSs do not support BGP traffic engineering (e.g., AS prepend, MED, and scoped advertisement); monitoring capabilities of the RS are in their infancy; and an RS adds an additional middleman that everyone has to trust. Finally, the authors suggested that an RS could be a place for the future deployment of security mechanisms for BGP (e.g., sBGP and RPKI).

B. Inter-Cloud

Inter-cloud is defined as the interconnection of various cloud provider infrastructures that share their computing and storage resources. Grozev et al. [8] classified inter-cloud architectures as a volunteer federation, in which participants willingly share resources, or independent/multi-cloud, in which charges may apply for the use of resources. The authors also identified centralized and peer-to-peer architectures under the volunteer federation category. Most research projects in inter-clouds are developed as centralized federations such as InterCloud [9] and Contrail [10]. Although the centralized federation is the most common approach, RESERVOIR[11] and Open Cirrus[12] are also examples of peer-to-peer federated inter-clouds.

C. National Research and Education Networks

A specialized ISP that supports the needs of research and education communities within a country is referred to as a national research and education network (NREN). It is characterized by a high-speed backbone network used to provide dedicated connections for individual research projects. To incentivize collaboration and increase the scale of some research projects, NRENs from several countries have agreed to interconnect their networks. For instance, Internet2 in the United States is connected to GEANT in Europe. NRENs can be used as nationwide testbeds for networking research. However, if a testbed spans more than one NREN, it is usually called a federated testbed. Currently, the addition of SDN capabilities to federated testbeds open the door to SDX research [13],[14].

D. SDX Taxonomy

Considering the infrastructures previously defined as our baseline, the spectrum of definitions for an SDX ranges from networking exchanges to cloud-service exchanges. Moreover, the network SDX definition, as mentioned above, includes Layer-3 SDXs for IXPs, Layer-2 SDXs for multi-domain Ethernet circuits in NRENs, and SDN SDXs for interconnecting SDN islands. Likewise, the cloud-service SDX enables the orchestration of computing and storage resources for InterClouds. In Figure 1, we show a taxonomy for an SDX based on the architecture infrastructure.

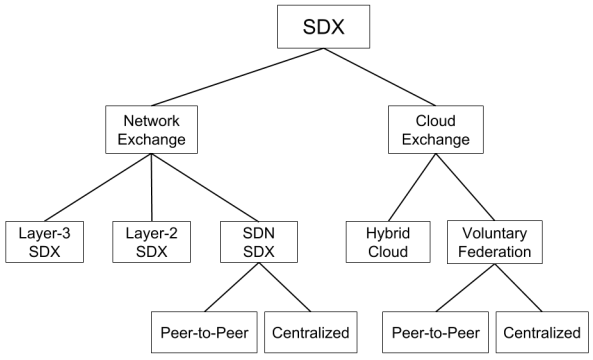

Figure 1. SDX Taxonomy

E. SDX Use Cases

Our SDX taxonomy classifies SDX architecture infrastructures. The type of architecture infrastructure influences the type of application or use case for an SDX. For instance, on

an IXP, we would like to define richer policies than those allowed by BGP. To augment the capabilities of BGP policies in an IXP, Gupta et al. [1] proposed four SDX applications: application-specific peering, inbound traffic engineering, wide-area load balancing, and redirection through middle boxes. In the context of federated testbeds, the FELIX project[15] defines six applications classified into two main groups: the data domain and the infrastructure domain. Under the data domain, the authors defined three applications: data-on-demand, data preprocessing, and high-quality media transmission over long-distance networks. The applications for the infrastructure domain are data mobility for inter-cloud use, follow-the-sun (or -moon) principles, and disaster recovery by IaaS migration. Table I summarizes SDX applications that we found in the literature.

III. Layer 2

In the United States, Layer-2 network exchanges are mainly used in research and education networks such as Internet2 and ESnet. In this scenario, NRENs interconnect using Layer-2 technologies such as Ethernet VLANs or optical circuits. For instance, the Advanced Layer 2 Service (AL2S) of Internet2 [21] allows network operators to create their own Layer 2 circuits in the Internet2 AL2S backbone. Similarly, the On-demand Secure Circuits and Advance Reservation System (OSCARS) [22] accomplishes the same goal in the high-performance science network of the Department of Energy (ESnet). Another example is the Open Exchange Software Suite (OESS) of the Global Research NOC, which allows network operators to configure and control dynamic Layer-2 virtual circuit (VLAN) networks on OpenFlow-enabled switches. To achieve a Layer-2 SDX, a viable alternative would be to deploy a single SDX controller that manages the entire exchange switch fabric.

IV. Layer 3

A layer-3 SDX provides SDN capabilities to the switching fabric of an IXP. The main characteristics of this kind of SDX is that to handle the exchange of BGP routes between IXP participants, the layer-3 SDX requires a BGP process. The minimum additional requirements are an SDN-enabled fabric, an SDN controller that installs flows between the participants and a BGP process that listens to the BGP announcements of participants. To enrich policies that can be defined with BGP, a policy manager is recommended. A general architecture for layer 3 SDXs is shown in Figure 2. Some examples of layer-3 SDXs are SDN-IP [19], Cardigan [2] and SDX [1], which are described in more detail in the next subsection.

A. SDN-IP

Lin et al. [19] proposed a solution that enables BGP peering between SDN and non-SDN autonomous systems in which the centralized SDN control plane integrates a BGP process that turns the entire SDN AS into a single BGP router from the point of view of its peers. The authors developed the solution for a Layer 3 SDX as an application in the ONOS

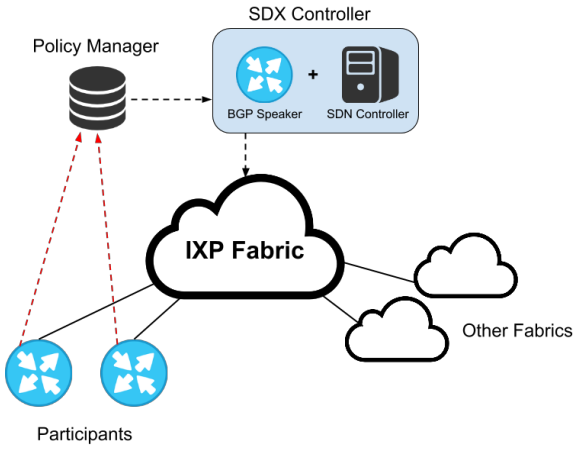

Figure 2. General architecture for a Layer 3 SDX

controller[20], and tested it using an emulated Mininet topology. Their SDN-IP architecture is composed of OpenFlow switches, a ONOS controller, the SDN-IP application, and a BGP speaker. Compared to the general architecture, this SDN-IP architecture lacks a policy manager. Their experiments tested how the number of routing information base (RIB) entries affects the incremental costs of the memory. The authors concluded that SDN-IP can scale up to 10,000 RIB entries and process 100 RIB updates per second.

B. Cardigan

Cardigan[2] is a distributed router based on RouteFlow and a mesh of OpenFlow switches represented as a single logical switch. The datapath of Cardigan works in a MPLS full mesh, and its control plane includes a forwarding path manager (FPM) and a Quagga routing engine that allows more flexible route selection. Cardigan, which connects the Research and Education Advanced Network of New Zealand (REANNZ) to the Wellington Internet exchange (WIX), handles 1,134 flows with a TCP performance of approximately 800Mbps.

C. SDX

Gupta et al. [1] designed, implemented, and evaluated an SDN framework to improve the network management capabilities of BGP participants in an IXP called SDX. The main idea behind SDX is to provide a virtual SDX switch to each BGP participant so that they can realize high-level tasks such as application-specific peering, inbound traffic engineering, wide area load balancing, and redirection through middle boxes all while ensuring isolation between each participants policies. The SDX architecture includes an OpenFlow-enabled switch that acts as the IXP fabric and an SDX-enabled route server, or SDX controller, which includes a BGP speaker and an SDN controller that accepts policies from participants and installs flow tables in an OpenFlow switch. For this solution, each participant sends its policies to the SDX controller; then the SDX engine compiles the individual policies and installs a single set of policies on the SDX switch.

Table I
SDX USES CASES

Use Case	Description	Current Solution
Application-specific peering[1]	Two neighboring AS exchange traffic only for certain applications. SDX could instead install custom rules for groups of flows corresponding to specific parts of flow space.	ISP could configure packet classifiers, Virtual Routing and Forwarding (VRF) and policy based routing
Inbound traffic engineering[1]	By installing forwarding rules in SDN-enabled switches at an exchange point, an AS can directly control inbound traffic according to source IP addresses or port numbers.	Destination based routing. Need to use AS prepending, communities and selective advertisement
Wide-area server load balancing[1]	A participant could announce anycast prefixes and the SDX controller would rewrite the destination IP address to match the chosen hosting location based on any fields in the packet header.	DNS global load balancing, IPv6 anycast
Redirection through middleboxes[1]	An SDN-enabled exchange point can redirect targeted subsets of traffic through one or more middleboxes.	Manipulate the routing protocols to "steer" traffic through a fixed set of middleboxes.
Data-on-demand[15]	An SDX controller could automatically establish links between different end points (e.g. data storage and applications or users), guaranteeing the reliability of the end-to-end communication (e.g. minimum delay and jitter).	Content Centric Networks / Information Centric Networks
Data preprocessing for minimizing network latency effect for live data[15]	In these situations, a dedicated platform would be placed near the receiver station and perform a suitable preprocessing of the data, reducing the size of the data to be transferred.	WAN compression / WAN acceleration
High-quality media transmission over long-distance networks[15]	A higher quality of the media playback, imposes higher bandwidth and lower delay constraints on the network. This will require inter-domain QoS policies to achieve satisfactory QoE.	Compression and buffering techniques at the client and server ends
DDoS Mitigation[16]	Add DDoS detection and mitigation capabilities in the IXP	BGP FlowSpec
Prefix hijacking prevention[17]	Deploy RPKI in an Software Defined IXP	RPKI
Inter-Cloud use case[15]	Data mobility service by SDN technologies. A SDX controller would be able to transfer user data (such as credentials, applications and services) to a cloud system closer to his/her visiting place.	Mobility Management
Follow the sun (or moon) principles[15]	An SDX controller could shift the load of one federated cloud to another one depending on the availability of resources and time of the day.	Cloud orchestration / scheduling
Disaster recovery by IaaS migrating[15]	An SDX controller could coordinate the configuration of the hypervisor resources with the network bandwidth constraints to allow a fast and efficient migration of the IaaS instance from one site to the other.	Replication to contingency sites
Source-Address Based Multi-Path Routing[18]	Supports multipath inter-domain routing to the same destination based on different source IP addresses.	Traffic Engineering in private WAN
Inter-Domain Path Computation (IDPC)[18]	The purpose of IDPC is to achieve end-to-end QoS routing. After WE-Bridge modules in all the domains exchange local virtual network views including the bandwidth information, each domain can construct a relative global network view.	None

V. SDN SDX

The design objective of the SDN SDX is to interconnect SDN islands managed by independent administrative domains. Two ways to build an SDN SDX are utilizing either a centralized or peer-to-peer architecture (see Figure 1). A centralized SDN SDX architecture could be implemented in one of three ways, shown in Figure 3: (1) implementing a single logically centralized controller (option 1 in Figure 3); (2) using an intermediate slice manager such as FlowVisor[23] or FlowSpace Firewall[24], which allow an external controller (i.e., the SDX controller) to manage a portion of each participants network, as shown in the middle of Figure 3 (option 2); or (3) creating a hierarchy of controllers with a local controller at each exchange being managed by a separate higher level controller, shown on the right side of in Figure 3 (option 3). The second way to build an SDN SDX, the peer-to-peer architecture, uses a West-East protocol between controllers in the SDX [18], shown in Figure 4.

A. Centralized SDN SDX

Carrozo et. al [15] proposed FELIX (Federated Testbed for Large-scale Infrastructure Experiments) as an architecture that integrates SDN testbeds dispersed across continents. FELIX, which uses a hierarchical model for inter-domain dependency management and resource orchestration, can be mapped to a

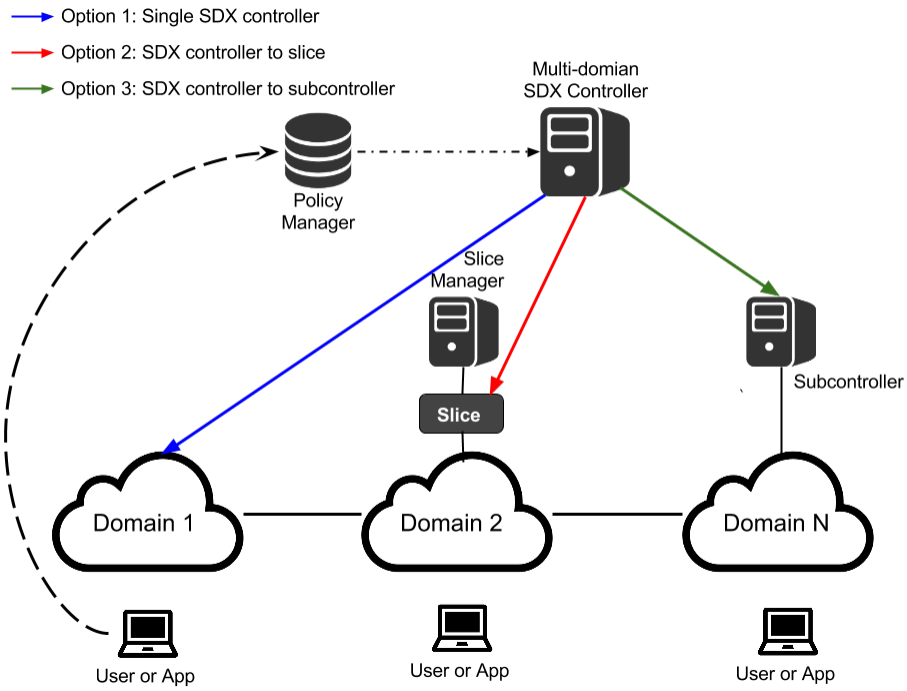

Figure 3. Centralized SDX controller approaches architecture for SDN SDX

centralized SDX architecture. In the FELIX architecture, the resource manager (RO), which establishes end-to-end network services and resource reservation, functions as the SDX controller of their generalized architecture. The architecture uses the network-service interface (NSI) protocol to exchange inter-domain topology information. Similarly, the SDX proposed by Mambretti et al [25],[26], uses a centralized path controller and an NSI to manage the resources of federated controllers that interconnect federated SDN islands.

Eddy et al. [16] described an architecture that leverages SDN to mitigate DDoS attacks by allowing customers to push flow definition rules upstream to their ISP that block undesired inbound traffic. The architecture is composed of a filter management application (FMA), customer switches, a customer controller, ISP switches, an ISP controller, and an ISP sub-controller. The ISP sub-controller is the most important element enabling the inter-domain SDN in this architecture. OpenFlow establishes communication between the various elements of the architecture and a limited subset of OpenFlow messages (Restricted OpenFlow) between the FMA and the sub-controller. The authors recognize that this architecture could be implemented using FlowVisor, as this architecture is a subset of use cases for FlowVisor. However, FlowVisor suffers from scaling and isolation problems that prohibit its use in ISP scenarios.

B. Peer-to-Peer SDN SDX

WE-Bridge [18] is a mechanism that enables independent SDN administrative domains to peer and cooperate. WE-Bridge itself is not an inter-domain routing protocol. Its main goal is to improve inter-domain routing by announcing domain-views containing rich/fine-granularity information/policies that enable various inter-domain innovations based on network information. This solution includes network view virtualization and a virtual network format and distribution using JSON. Peer relationships are established through a peer-to-peer control plane and a modified version of the Link Layer Discovery Protocol (LLDP) that connects domain border switches.

VI. Service-Level Model

Service-level model network architectures, developed in disciplines such as cloud computing and service provider networks, resemble SDX. For instance, Meridian [27] is an SDN architecture for cloud networking with service-level model orientation. This architecture includes three main logical layers (see Figure 5): a network model and API, a network orchestrator, and interfaces toward underlying network devices. The abstract API layer provides a network model and associated APIs that expose only information needed to interact with the network. The network orchestration layer performs a logical-to-physical translation of commands. The network driver layer consists of “plug-ins” or “drivers” that interact with the network. A major challenge in Meridian is to manage dynamic updates to a virtual network topology. To manage this challenge, Meridian maintains a control block for each virtual network.

Similar to Meridian, Kempf et al. [28] proposed a service provider SDN (SP-SDN), an approach to rapid and flexible cross-domain service creation that complements the SDN and

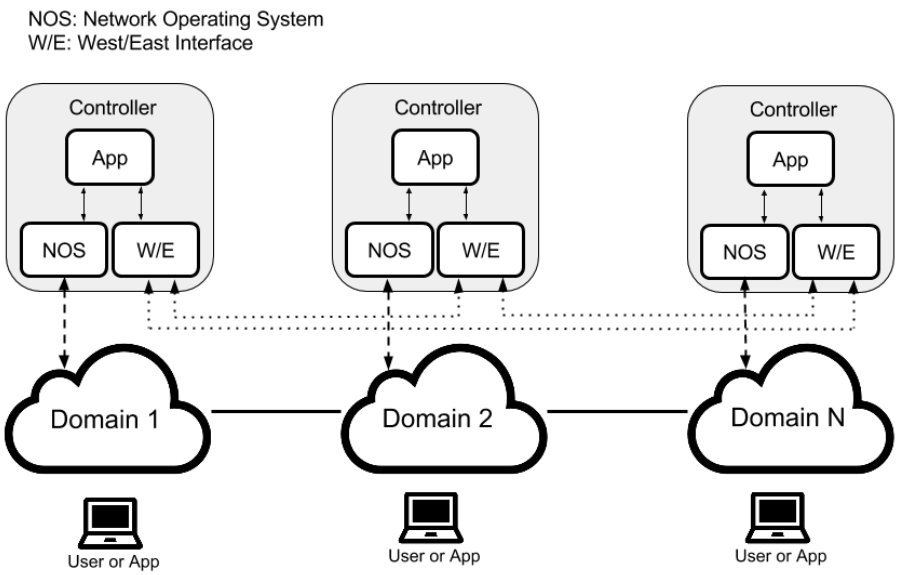

Figure 4. Peer-to-peer architectural SDX controller for SDN SDX

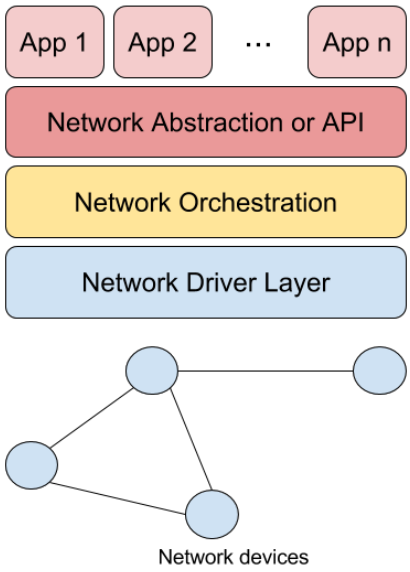

Figure 5. Merdian Architecture

NFV. SP-SDN relies on a cross-domain orchestrator to expose network functionality at the service layer. The interface of the orchestrator is abstract, simple, and real time. The authors presented two examples of applications: an elastic WAN application and a Network slices-on-demand application.

VII. Architecture Comparison

This section presents a comparison of the SDX architectures described in previous sections (see Table II).

A. Qualitative Analysis

This section compares several SDX architectures according to metrics such as scalability, resiliency, and peering boundaries.

1) Scalability: From the literature presented in this paper, only SDX [1], Cardigan [2] and SDN-IP [19] present a performance evaluation of their solutions while FELIX [15], WEBridge [18], and SP-SDN[28]) do not provide any quantitative evaluation of their work. The main scalability metric for SDX, Cardigan, and SDN-IP is the number of RIB entries or flow rules, since they are examples of Layer-3 SDXs. Table III shows the performance evaluation results presented in each paper. For SDX, we considered the maximum values of 1,000 prefix groups with 300 participants in the IXP [1]. For SDN-IP, we considered the most recent value shown in ONOS SDN-IP Wiki[1] as opposed to the value presented in the paper.

2) Resilience: Resilience is the ability to maintain acceptable levels of service in the face of faults in the network infrastructure. In this regard, since the SDX controller becomes a single point of failure in the centralized approach, the peer-to-peer SDX architecture is inherently more resilient. However, resilience measures can be built into the centralized SDX architecture by the addition of distributed-computing techniques. For instance, SDN-IP is an application that runs on top of ONOS[20], an SDN-distributed network operating system.

3) Peering Technologies: To enable inter-domain networking, the boundaries between domains need to be set. BGP is used for Layer-3 routing while VLAN mapping and Q-in-Q tunneling are preferred in Layer-2 deployments. More recent architectures such as peer-to-peer SDN SDX require modifi-

[1]https://wiki.onosproject.org/display/ONOS/SDN-IP+Architecture

Table II
SDX ARCHITECTURE COMPARISON

	Layer-3 SDX (Section IV)	**Layer-2 SDX (Section III)**	**Centralized SDN SDX (Section V-A)**	**Peer-to-Peer SDN SDX (Section V-B)**
Goal	To allow BGP peering between SDN and non-SDN networks, enabling richer policy descriptions	To interconnect independent networks using L2 technologies (VLAN) and OpenFlow	To interconnect SDN islands and to extend SDNs across multiple domains using a centralized/hierarchical controller	To interconnect SDN islands and to extend SDNs across multiple domains using a West-East interface/protocol
Architecture components	SDX controller, SDX fabric, Route Server, BGP Speaker, Policy Server	SDX controller, SDN fabric, Policy Server	SDX controller, SDN switches or slices, inter-domain link, Policy Server	SDX controller, West-East Interface/Protocol, West-East Link
SDX Type	Centralized	Centralized	Centralized	Peer-to-Peer
Topology Design	Centralized Single Fabric (Edge and Core), Single centralized controller	Centralized Single Fabric, Centralized controller	Inter-domain L2 links between switches/slices, Centralized controller	Inter-domain L2 links between switches/slices, Peer-to-Peer control plane
Peering Protocol	BGP	Static VLAN, Q-in-Q	OpenFlow, NSI	WE-Bridge + Modified LLDP
Isolation	Virtual switch per participant	VLAN	Slices	Virtual network view

Table III
LAYER-3 SDX SCALABILITY COMPARISON

Name	**Max. Flow Rules**	**Compilation or Convergence Time**
SDX	30000	700 sec.
Cardigan	1135	1 sec.
SDN-IP	15000 RIB entries	100 RIB updates/sec

cations to the LLDP protocol that delineate the boundaries between two SDN domains [18] in an automated fashion.

B. Deployment Considerations

The deployment of an SDX requires several considerations with regard to downtime, network topology changes, and disruption of services. Although Cardigan was deployed in a small IXP in New Zealand, the study does not discuss deployment issues or experiences [2]. By contrast, the remainder of the proposed solutions have been tested in emulation environments (i.e., Mininet) or in research and education networks. To deploy a large-scale Layer-3 SDX in an IXP, network operators might take advantage of hybrid SDN approaches in which SDN and conventional switches coexist in the same infrastructure. For instance, it is preferable that an IXP maintain its high-speed core fabric while adding SDX switches in the edge, which allow richer BGP policies.

Regarding centralized SDN SDX architectures, a sliced approach will increase the cost of deployment. For instance, if a certain SDN domain does not participate in an exchange using a slice manager, the deployment of the slice manager should be carefully planned so that it does not interrupt service in a production network. Since the resources allocated to the SDX are isolated and restricted, the sliced approach provides the most security. In the case of the subcontroller architecture, one should consider the communication protocol between controllers in the hierarchy (i.e., the customer controller, the ISP controller, and the ISP sub-controller) and have a rule conflict resolution mechanism for security.

Finally, control messaging should be deployed over secure protocols. In the case of a centralized SDX controller over slices, the use of TLS should be mandatory, and mutual authentication between the controller and slices is recommended. Likewise, peer-to-peer approaches should include security mechanisms in their protocols such as encryption of the payload and authentication of peer SDX controllers.

VIII. CONCLUSIONS

In this paper, we presented an architecture-focused survey on the most relevant SDX studies from various research areas. We defined a taxonomy for SDXs and provided generalized architecture models. We compared key characteristics of the general SDN architectures presented in this paper. Finally, we analyzed these architectures qualitatively, focusing on scalability, resilience, peering technologies, and deployment considerations. In future work, we will continue evaluating SDX architectures with a quantitative focus by building large-scale testbeds in the GENI platform and measuring important network parameters such as latency, uptime, and network throughput as well as controller performance metrics (e.g., rule computation time, scalability, and resiliency).

ACKNOWLEDGMENTS

The authors would like to thank the National Science Foundation and the GENI Project Office for their generous support of this work and the GENI community for its continued collaboration with the authors on defining and creating a software-defined exchange. The authors will also like to thank the reviewers for their valuable comments.

REFERENCES

[1] 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," in *Proceedings of the 2014 ACM conference on SIGCOMM*. ACM, 2014, pp. 551–562.

[2] J. Stringer, D. Pemberton, Q. Fu, C. Lorier, R. Nelson, J. Bailey, C. N. Correa, C. Esteve Rothenberg *et al.*, "Cardigan: Sdn distributed routing fabric going live at an internet exchange," in *Computers and Communication (ISCC), 2014 IEEE Symposium on*. IEEE, 2014, pp. 1–7.

[3] GEC23, "Sdx panel," http://groups.geni.net/geni/wiki/GEC23Agenda/SDX.

[4] N. Chatzis, G. Smaragdakis, A. Feldmann, and W. Willinger, "There is more to ixps than meets the eye," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 5, pp. 19–28, 2013.

[5] B. Augustin, B. Krishnamurthy, and W. Willinger, "Ixps: mapped?" in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. ACM, 2009, pp. 336–349.

[6] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a large european ixp," in *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*. ACM, 2012, pp. 163–174.

[7] P. Richter, G. Smaragdakis, A. Feldmann, N. Chatzis, J. Boettger, and W. Willinger, "Peering at peerings: On the role of ixp route servers," in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014, pp. 31–44.

[8] N. Grozev and R. Buyya, "Inter-cloud architectures and application brokering: taxonomy and survey," *Software: Practice and Experience*, vol. 44, no. 3, pp. 369–390, 2014.

[9] R. Buyya, R. Ranjan, and R. N. Calheiros, "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services," in *Algorithms and architectures for parallel processing*. Springer, 2010, pp. 13–31.

[10] E. Carlini, M. Coppola, P. Dazzi, L. Ricci, and G. Righetti, "Cloud federations in contrail," in *Euro-Par 2011: Parallel Processing Workshops*. Springer, 2012, pp. 159–168.

[11] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres *et al.*, "The reservoir model and architecture for open federated cloud computing," *IBM Journal of Research and Development*, vol. 53, no. 4, pp. 4–1, 2009.

[12] A. I. Avetisyan, R. Campbell, K. Lai, M. Lyons, D. S. Milojicic, H. Y. Lee, Y. C. Soh, N. K. Ming, J.-Y. Luke, H. Namgoong *et al.*, "Open cirrus: A global cloud computing testbed," *Computer*, no. 4, pp. 35–43, 2010.

[13] T. Magedanz, F. Schreiner, and S. Wahle, "Service-oriented testbed infrastructures and cross-domain federation for future internet research," in *Integrated Network Management-Workshops, 2009. IM'09. IFIP/IEEE International Symposium on*. IEEE, 2009, pp. 101–106.

[14] J. Ibarra, J. Bezerra, H. Morgan, L. Fernandez Lopez, M. Stanton, I. Machado, E. Grizendi, and D. Cox, "Benefits brought by the use of openflow/sdn on the amlight intercontinental research and education network," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, May 2015, pp. 942–947.

[15] G. Carrozzo, R. Monno, B. Belter, R. Krzywania, K. Pentikousis, M. Broadbent, T. Kudoh, A. Takefusa, A. Vieo-Oton, C. Fernandez *et al.*, "Large-scale sdn experiments in federated environments," in *Smart Communications in Network Technologies (SaCoNeT), 2014 International Conference on*. IEEE, 2014, pp. 1–6.

[16] W. Eddy, G. Clark, and J. Dailey, "Customer-controlled filtering using sdn," https://datatracker.ietf.org/doc/draft-eddy-sdnrg-customer-filters/.

[17] J. Bailey, D. Pemberton, A. Linton, C. Pelsser, and R. Bush, "Enforcing rpki-based routing policy on the data plane at an internet exchange," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 211–212.

[18] P. Lin, J. Bi, S. Wolff, Y. Wang, A. Xu, Z. Chen, H. Hu, and Y. Lin, "A west-east bridge based sdn inter-domain testbed," *Communications Magazine, IEEE*, vol. 53, no. 2, pp. 190–197, 2015.

[19] P. Lin, J. Hart, U. Krishnaswamy, T. Murakami, M. Kobayashi, A. Al-Shabibi, K.-C. Wang, and J. Bi, "Seamless interworking of sdn and ip," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 475–476. [Online]. Available: http://doi.acm.org/10.1145/2486001.2491703

[20] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, and G. Parulkar, "Onos: Towards an open, distributed sdn os," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '14. New York, NY, USA: ACM, 2014, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/2620728.2620744

[21] Internet2, "Advanced layer 2 system," http://www.internet2.edu/products-services/advanced-networking/layer-2-services/.

[22] ESnet, "On-demand secure circuits and advance reservation system," http://www.es.net/engineering-services/oscars/.

[23] 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*, 2009.

[24] GlobalNOC, "Flowspace firewall," http://globalnoc.iu.edu/sdn/fsfw.html.

[25] J. Mambretti, J. Chen, and F. Yeh, "Software-defined network exchanges (sdxs): Architecture, services, capabilities, and foundation technologies," in *Teletraffic Congress (ITC), 2014 26th International*. IEEE, 2014, pp. 1–6.

[26] J. Mambretti, J.-J. Chen, and F. Yeh, "Software-defined network exchanges (sdxs) and infrastructure (sdi): Emerging innovations in sdn and sdi interdomain multi-layer services and capabilities," in *Science and Technology Conference (Modern Networking Technologies)(MoNeTeC), 2014 First International*. IEEE, 2014, pp. 1–6.

[27] 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.

[28] J. Kempf, M. Korling, S. Baucke, S. Touati, V. McClelland, I. Mas, and O. Backman, "Fostering rapid, cross-domain service innovation in operator networks through service provider sdn," in *Communications (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3064–3069.