Standards

Editor: Barry Leiba barryleiba@computer.org

The Software-Defined-Networking Research Group

David Meyer

Brocade Communications/University of Oregon **//au: please pick a primary affiliation to run on the first page of the article; we can list multiple affiliations in the bio at the end//**

//au: please provide a brief abstract of roughly 50 words to run here and in the Computer Society digital library, along with at least three keywords//

The “Moore’s law” effect has seen network hardware improve over the past several decades along almost all hardware-oriented dimensions, including port densities, speeds and feeds, and price **//per?//** bit. At the same time, the network control plane has evolved far more slowly **//au: ok?//**. In fact, configuring and managing networks continues to be about network element operations, rather than network services. This mix of ever-growing networks and ever-advancing configuration requirements makes network agility, service velocity, operation, and debugging increasingly difficult and expensive. We can see this effect across all network types, including enterprise, data centers, and service provider networks.

*Software-defined networks* (SDNs) are widely seen as a promising solution for resolving these challenges. In particular, SDNs promise to provide a multilayer platform that encompasses programmability not only at the forwarding and control planes, but also at the transport layers below and orchestration and services layers above the data and control planes.

Early SDN models focused primarily on moving the control plane out of the network elements into “controllers” on the theory that the switching elements could remain simple, general-purpose, and cost-effective while letting the control plane rapidly evolve. Several recent SDN models, on the other hand, include approaches in which control- and data-plane programmability works in concert with existing and future distributed control planes.

# SDNRG Goals and Objectives

SDN aims to benefit all network types, including wireless, cellular, home, enterprise, data centers, and wide-area networks. The Software-Defined Networking Research Group (SDNRG) is part of the Internet Research Task Force. IRTF is the research arm of the IETF; it’s a parallel organization that focuses on research, whereas the IETF works on engineering solutions. The SDNRG investigates SDN from various perspectives with the goal of identifying both approaches that can be defined, deployed, and used in the near term, and future research challenges. Key areas of interest include solution scalability, abstractions, and programming languages and paradigms particularly useful in an SDN context. Moreover, the SDNRG explicitly aims to provide a forum for researchers to investigate key and interesting problems in the SDN field.

Finally, the SDNRG provides objective definitions, metrics, and background research that it can provide as input to protocol, network, and service design for standards developing organizations (SDOs) and other standards-producing organizations **//au: you list several SDOs later in the article, when talking specifically about SDN and standards, so I’ve removed the examples here to avoid redundancy//**.

# Current Events

Although a wide variety of what people call SDN is familiar, it’s still a nascent idea. Many of SDN’s underlying ideas — such as separating the control and data planes — have turned up in telecommunications in prior technologies, but what’s interesting is the design space it represents. For example, SDN models range from OpenFlow/SDN (OF/SDN),1 which is characterized by complete separation of control and data planes and open access to the forwarding plane, to overlay models that are less concerned with existing control and data planes. Equally diverse are ideas about what kinds of APIs should exist and which abstractions they do or should implement. Many ideas **//have arisen?//** about distributed controller design and implementation, several revolving around how **//to manage?//** state. Finally, the number of SDN use cases is growing daily. Given this backdrop, the SDNRG currently operates as more of a workshop to allow the diverse SDN community to explore all these factors and their interactions.

Notwithstanding the vast diversity of thought in the SDN community, we agree on several topics that could use some directed consideration, including

classification of SDN models;

* definitions and taxonomies;
* SDN model scalability and applicability;
* multilayer programmability and feedback control systems;
* system complexity;
* network description languages, abstractions, interfaces, and compilers;
* methods and mechanisms for (online) verification of correct network/node operation /**/au: ok?//**; and

security.

For example, in the case of SDN models, at least three architectural approaches are under discussion. **Points** **//au: what do you mean by “points” in this context?//** in this space represent key architectural features and include centralized versus distributed control, various degrees of separation of control and data planes, and different programmability points (for instance, one model might make the control plane programmable and another might make the data plane programmable). We can view the different approaches as design points in a continuous, multidimensional design space, termed “the SDN Continuum” (see www.1-4-5.net/~dmm/talks/nanog58.pdf).

One point in the SDN Continuum is OF/SDN.1 Work on this model began at Stanford University as part of the Clean Slate project (http://cleanslate.stanford.edu) and now takes place in the Open Networking Foundation (ONF; www.opennetworking.org). OF/SDN is characterized by the complete separation of the control and data planes, open interfaces to the forwarding plane (that is, OpenFlow), and centralized control (note that “logically centralized” is the term currently used for this kind of centralization of control; logically centralized actually means “distributed”2).

A second interesting point in this space is Control Plane/SDN (CP/SDN); here, designers seek to make the existing, distributed control plane programmable. Interface to the Routing System (I2RS; http://datatracker.ietf.org/wg/i2rs/charter/**)** is a protocol designed to make the *routing information base* (RIB) programmable and enable new kinds of network provisioning and operation. The basic idea is that because the RIB is already an arbitration engine for various routing state sources, CP/SDN will provide mechanisms for using that arbitration engine to inject routing state into the system as well as read possibly abstracted state from it. Examples of protocols in this class include the Border Gateway Protocol-Link State **//au: correct?//** (BGP-LS) and Application-Layer Traffic Optimization (ALTO).

Finally, Overlay/SDN (OL/SDN) represents a design point that encompasses those models in which a virtual network is overlaid on the network (which might be physical, virtual, or both). In this case, the designers overlay a new control plane on top of existing control and data planes, and (in theory) have minimal interaction with the “underlay” network. (Note that the degree to which an OL/SDN control or data plane requires knowledge or control of network state in the underlay network is a topic of vigorous debate.) The IETF’s Network Virtualization Overlays (NVO3) working group is involved in standardizing OL/SDN models.

Of course, these are discrete points in the design space; we can mix and match the features they represent to yield interesting architectures. For example, in some use cases, it might make sense to direct traffic down traffic-engineered tunnels using **OpenFlow** **//or “OF/SDN,” as previously?//** at the network edges and the Path Computation Element (PCE; http://datatracker.ietf.org/wg/pce/charter/) to build the tunnels themselves. On the other hand, OL/SDN architectures are typically less concerned with the programmability of the “underlay” control or data planes; rather, scaling here is achieved by overlaying a new programmable control plane and, in some cases, a virtual data plane such as the Open Virtual Switch (OVS).

# SDN and Standards

Given the myriad ideas on what SDN architecture might look like and what its components might be, it’s unsurprising that almost every **SDO** and many quasi-**SDOs //au: correct?//** are involved in SDN standardization. These include the IETF, the European Telecommunications Standards Institute (ETSI), the Alliance for Telecommunications Industry Solutions (ATIS), ITU-T, IEEE, the Open Networking Foundation (ONF), the Metro Ethernet Forum (MEF), the Distributed Management Task Force (DMTF), and the Object Management Group (OMG) **//au: except for a few well-known to IC’s readership, we spell out the names of most standards organizations. Let me know if any of these are incorrect. Also, I combined this list with the one I moved from a previous section, so please strike any that should be removed here//**. Moreover, several open source consortia are producing software in the SDN space (www.opendaylight.org).

In the IETF’s case, we can consider Forwarding and Control Element Separation (ForCES; http://datatracker.ietf.org/wg/forces/charter/), Network Configuration (NetConf; http://datatracker.ietf.org/wg/netconf/, YANG (http://en.wikipedia.org/wiki/YANG), I2RS, PCE, the Locator/ID Separation Protocol (LISP; https://datatracker.ietf.org/wg/lisp/charter/), and many other technologies to be, at the very least, SDN-enabling. Even BGP3 is used in many SDN solutions (see www.juniper.net/us/en/dm/junos-v-contrail/).

Because it’s early in SDN’s evolution, the SDNRG has an important role to play in the standards process. In particular, research into many SDN technologies currently being standardized in the IETF, such as I2RS or PCE, is essential to not only guide the development of the standards themselves but also help inform deployment models. Hence, the SDNRG encourages research and provides a forum for discussion. It also seeks to understand the larger SDN context and where standardization might be needed.

Finally, the SDNRG will publish documents that outline active research, development, and pre-standardization areas in the SDN space. To this end, the SDNRG has active drafts dealing with terminology for SDN (a surprisingly contentious area), security, and formal methods for correctness.

# Open Problems

Several open problems exist in the SDN space, ranging from architectural questions that are fundamental to how networks scale and evolve4 to implementation issues such as how we might build distributed “logically centralized” control planes.2 For example, how much programmability should an SDN system provide, and where? **OpenFlow** **//OF/SDN?//** offers one solution (the forwarding plane should be directly programmable), whereas I2RS provides another (the RIB should be the locus of programmability). Other questions include how distributed state management systems (such as controllers) should implement state consistency — issues here generally revolve around the CAP theorem (www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed) — and which APIs and corresponding abstractions should be standardized, if any.

The future, however, **//will most likely use?//** a combination of these (and perhaps other) models, standards, and implementations. Models such as I2RS are likely to be used to provide network programmability in those heterogeneous environments in which scalability is the primary concern. The architectural models underlying I2RS envision layered, distributed control planes in which the control and data planes share fate. (Note that breaking fate-sharing between the control and data planes induces a lower bound on control plane convergence — namely, (*convergence*)  *RTT*(*controller*, *switch*) + *PPT*(*controller*) + *PPT*(*switch*), where *RTT* is “round-trip time” and *PPT* is “packet-processing time.”) On the other hand, in some environments (for example, data centers) overlay technologies can provide scale and adaptability.

What we do know from our 25 or so years of building networks is that systems that are scalable, resilient, and evolvable have certain, well-defined architectural features. One such feature is a distributed control plane. Nevertheless, the degree to which a control plane is centralized versus distributed remains one of the many active debates in the SDN community. Other features conferring scalability and evolvability include a high degree of layering, robust feedback control, and protocol-based architectures, among others.4 **//Disagreements? Debates?//** notwithstanding, it’s clear that one key to building scalable and evolvable SDN systems will be to understand which pieces of the distributed control plane can be efficiently “peeled off” and run in a (logically) centralized fashion. For example, traffic engineering can clearly benefit from a global, network-wide view. PCE is an example of a traffic engineering technology under consideration as a candidate for measured separation of control and data planes, with the goal of making the network (more) programmable and architecturally centralized. Note, however, that even in the PCE case, the underlying distributed control plane remains. In this way, PCE retains the resilience and scale of the underlying distributed control plane while still providing programmability and a global network view for use in traffic engineering.

**//au: conclusions for IC should avoid summarizing what the article has already discussed, so I’ve removed such text here//**

It is early in SDN’s evolution and an exciting time to be involved in its development. In particular, active technical debates are under way on almost every architectural point. Initial SDN models challenged (and continue to challenge!) much of what we might think of as the “central dogma” of the Internet architecture — namely, distributed control, layering, hop-by-hop forwarding, and separation of control and data planes. Other, more recent models attempt to provide programmability while still retaining the Internet architecture’s core features. Given the existing diversity of thought and approaches, the SDNRG is ideally situated as a forum for facilitating discussion and documenting these open and other questions.

References

1. *OpenFlow Switch Specification*, *Version 1.3.2 (Wire Protocol 0x04)*, Open Networking Foundation **//correct?//**, 25 Apr. 2013; www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.2.pdf.

2. D. Levin et al, “Logically Centralized? State Distribution Trade-Offs in Software Defined Networks,” *Proc. 1st Workshop on Hot Topics in Software-Defined Networks* (HotSDN 12), ACM, 2012, pp. 1–6; http://conferences.sigcomm.org/sigcomm/2012/paper/hotsdn/p1.pdf.

3. *A Border Gateway Protocol 4 (BGP-4)*, IETF RFC 4271, Jan. 2006; http://tools.ietf.org/html/rfc4271.

4. M.E. Csete and J.C. Doyle, “Reverse Engineering of Biological Complexity,” *Science*, vol. 295, no. 5560, 2002, pp. 1664–1669; doi: 10.1126/science.1069981.

David Meyer is a **//position?//** at **//affiliations?//**. His research interests include **//please list a few//**. Meyer has a **//highest degree?/** in **//subject?//** from **//institution?//**. Contact him at **//preferred email?//**.

Editor’s Introduction

A couple of issues ago, we ran a Standards department about theInterface to the Routing System. I2RS is a piece of a very broad set of solutions called *software-defined networking*. The Software Defined Networking Research Group is chartered to investigate approaches that can be deployed in the near term, and to look for future research challenges in the area. Here, David Meyer, a chair of the SDNRG, explains further. —Barry Leiba