**Introduction to the Software Defined Networking Research Group (SDNRG) in the IRTF**

David Meyer

Brocade/University of Oregon

dmm@1-4-5.net

**Introduction**

The “Moore’s Law” effect has seen network hardware improve along almost all hardware oriented dimensions, including port densities, speeds and feeds, and price/bit over the past several decades. At the same time network control plane has experienced dramatically slower evolution. In fact, configuring and managing networks continues to be about network element operations (as opposed to network services). This mix of ever growing networks and ever advancing configuration requirements makes network agility, service velocity, operation, debugging increasingly difficult and expensive. This effect is seen across all network types, including enterprise, data center, and service provider networks. Software-Defined Networks (SDNs) are widely seen as a promising set of solutions to resolve these challenges. In particular, SDN promises to provide a multi-layer platform, which 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 at the same time allowing the control plane to rapidly evolve. A number of 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 types of networks, 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), which is the research arm of the IETF, a parallel organization that focuses on research, where the IETF works on engineering solutions. The SDNRG investigates SDN from various perspectives with the goal of identifying the approaches that can be defined, deployed and used in the near term as well identifying future research challenges. In particular, key areas of interest include solution scalability, abstractions, and programming languages and paradigms particularly useful in the context of SDN. In addition, it is an explicit goal of the SDNRG to provide a forum for researchers to investigate key and interesting problems in the Software-Defined Networking field.

Finally, the SDNRG provides objective definitions, metrics and background research with the goal of providing this information as input to protocol, network, and service design to Standards Developing Organizations (SDOs) and other standards producing organizations including as the IETF, ETSI, ATIS, ITU-T, IEEE, ONF, MEF, and DMTF.

**Current Events**

While there is wide variety of what people call SDN that are familiar, SDN is still a nascent idea. And while many of the ideas underlying SDN, for example, separation of control and data planes, have turned up in telecommunications in prior technologies, what is interesting about SDN is the design space it represents. For example, SDN models range from OpenFlow/SDN [0], 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 Application Programming Interfaces (APIs) should exist and what kinds of abstractions they do or should implement. There are also many ideas about distributed controller design and implementation, many revolving around how state is managed. 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 of these factors and their interactions.

Notwithstanding the vast diversity of thought in the SDN community, there are several topics that we agree could use some directed consideration. These include

* Classification of SDN models
* Definitions and taxonomies
* SDN model scalability and applicability
* Multi-layer programmability and feedback control systems
* System complexity
* Network description languages, abstractions, interfaces and compilers
* Methods and mechanisms for (on-line) verification of correct operation of network/node function.
* Security

For example, in the case of SDN models, there are at least three architectural approaches being discussed. Points 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 example, one model might make the control plane programmable and another might make the data plane programmable). The approaches different approaches can be seen as design points in a continuous multi-dimensional design space, and has been termed “the SDN Continuum” [11].

One design point in SDN Continuum is the Openflow/SDN (OF/SDN) model [1]. Work on OF/SDN started at Stanford as part of the Clean Slate project [13] and now takes place in the Open Networking Foundation (ONF) [10]. OF/SDN is characterized by the complete separation of control and data planes, centralized control[[1]](#footnote-1), and open interfaces to the forwarding plane (i.e., Openflow).

A second interesting point in this space has been termed Control Plane/SDN (CP/SDN); here designers seek to make the existing, distributed control plane programmable. I2RS [4] is an example of a protocol designed to make the Routing Information Base, or RIB, programmable and enable new kinds of network provisioning and operation. The basic idea here is that since the RIB is already an arbitration engine for various sources of routing state, and CP/SDN seeks to provide mechanisms to use that arbitration engine (the RIB) to inject routing state into the system as well as read possibly abstracted state from the routing system. Examples of other protocols in this class include BGP-LS and ALTO.

Finally, Overly/SDN (OL/SDN) represents a design point that encompasses those models in which a virtual network is “overlaid” on the network (which may 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[[2]](#footnote-2). The NVO3 working group in the IETF is involved in standardizing OL/SDN models.

Of course, these are discrete points in the design space; the features represented by various points, including key features such as separation of control and data planes, forwarding plane programmability, and centralized control can be mixed and matched to yield interesting architectures. For example, in some use cases it might make sense to direct traffic down traffic engineered tunnels using Openflow at the edges of the network while the tunnels themselves might be built using PCE. On the other hand, OL/SDN architectures are typically less concerned with programmability of the “underlay” control or data planes; rather here scaling 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 multitude of ideas as to what SDN architecture might look like and what its components might be, it is not surprising that almost every SDN and many quasi-SDNs are involved in SDN standardization. These include (but are not limited to) the IETF, ONF, ETSI, DTMF, MEF, OMG and the ITU-T. There are also several open source consortia producing open source software in the SDN space [14].

In the case of the IETF, one could consider ForCES [1], NetConf [2], YANG [3], I2RS [4], PCE [5], LISP [6], and many other technologies IETF to be at the very least "SDN enabling". Even BGP [7] is being used in many SDN solutions [8].

Since it is very early in the evolution of SDN, the SDNRG has a very important role to play in the Standards Process. In particular, research into many of the SDN technologies being currently standardized in the IETF, for example, I2RS or PCE, is essential to guide the development of not only of the standards themselves but also can help to inform deployment models. Hence the SDNRG encourages research and provides a forum for discussion. The SDNRG also seeks to help understand the larger SDN context and where standardization might be needed.

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

**Open Problems**

There are several open problems in the SDN space. These range from architectural questions that are fundamental to how networks scale and evolve [12] to implementation issues such as how we might build distributed “logically centralized” control planes [9]. For example, how much programmability should an SDN system provide, and where? OpenFlow provides one solution (the forwarding plane should be directly programmable) while I2RS provides a different solution (the Routing Information Base, or RIB, should be the locus of programmability). Other questions include how distributed state management systems (e.g., controllers) should implement state consistency[[3]](#footnote-3), and which APIs and corresponding abstractions should be standardized, if any.

The future, however, is most likely a combination of these (and perhaps other) models, standards, and implementations. For example, models like I2RS, which retain a distributed control plane, are likely to be used to provide network programmability in those heterogeneous environments in which scalability is the primary concern. On the other hand, in some environments (for example, data centers) overlay technologies can be used to provide scale and adaptability.

What we do know, from our 25 or so years of building networks, however, is that systems which are scalable, resilient and evolvable have certain, well-defined architectural features. One of these features is a distributed control plane[[4]](#footnote-4). Nevertheless, the degree to which a control plane is “centralized versus distributed” remains one of the many interesting active debates in the SDN community. Notwithstanding these debates, it is clear that one of the keys 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, it is pretty clear that traffic engineering can benefit from a global, network-wide view. Path Computation Element (PCE) is an example of a traffic engineering technology that is being considered 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.

**Summary and Next Steps**

SDN is a nascent and active area of research. That said, several components of SDN systems are already the subject of standardization. These include protocols and interfaces such as Openflow, I2RS, PCE, BGP-LS and ALTO. These protocols represent very different architectural approaches to how one might build networks. Openflow/SDN envisions centralized control, complete separation of control and data planes, coupled with an open interface to the forwarding plane. On the other hand, the architectural models underlying I2RS, for example, envision layered, distributed control planes in which, in contrast to the Openflow/SDN model, the control and data planes share fate[[5]](#footnote-5).

In summary, it is early in the evolution of SDN and an exciting time to be involved in the development of SDN. In particular, there are active technical debates on almost every architectural point. For example, initial SDN models challenged (and continue to challenge!) much of what may be thought 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 core architectural features of the Internet architecture. Given the diversity of thought and approaches, the SDNRG is ideally situated as a forum that facilitates discussion and documentation of these open and other questions.

**Bibliography**

[0] “OpenFlow Switch Specification”, Version 1.3.2 (Wire Protocol 0x04), April 25, 2013, <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.2.pdf>

# [1] Forwarding and Control Element Separation (forces),<http://datatracker.ietf.org/wg/forces/charter/>

# [2] Network Configuration (netconf), <http://datatracker.ietf.org/wg/netconf/>

# [3] YANG, <http://en.wikipedia.org/wiki/YANG>

# [4] Interface to the Routing System (i2rs), <http://datatracker.ietf.org/wg/i2rs/charter/>

# [5] Path Computation Element (pce), <http://datatracker.ietf.org/wg/pce/charter/>

# [6] Locator/ID Separation Protocol (lisp), <https://datatracker.ietf.org/wg/lisp/charter/>

# [7] A Border Gateway Protocol 4 (BGP-4), <http://tools.ietf.org/html/rfc4271>

# [8] Contrail, <http://www.juniper.net/us/en/dm/junos-v-contrail/>

# [9] Levin, D. et. al, “Logically Centralized? State Distribution Trade-offs in Software Deﬁned Networks”, HotSDN 2012, <http://conferences.sigcomm.org/sigcomm/2012/paper/hotsdn/p1.pdf>

# [10] The Open Networking Foundation, <https://www.opennetworking.org/>

# [11] Meyer, D., “Macro Trends, Complexity, and Software Deﬁned Networking”, <http://www.1-4-5.net/~dmm/talks/nanog58.pdf>

[12] Marie E. Csete and John C. Doyle, “Reverse Engineering of Biological Complexity”, *Science* 295, 1664 (2002); DOI: 10.1126/science.1069981

# [13] Clean Slate Project, <http://cleanslate.stanford.edu/>

# [14] The OpenDaylight Project, <http://www.opendaylight.org/>

1. Note that the term “logically centralized” is the term currently being used for this kind of centralization of control. Logically centralized *actually* means “distributed” [9]. [↑](#footnote-ref-1)
2. Note that the degree to which an OL/SDN control and/or data plane requires knowledge and/or control of network state in the underlay network is a topic of vigorous debate. [↑](#footnote-ref-2)
3. Issues here generally revolve around the “CAP Theorem”. See e.g., <http://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed> [↑](#footnote-ref-3)
4. Other features conferring scalability and evolvability include a high degree of layering, robust feedback control, and protocol based architectures, among others [12]. [↑](#footnote-ref-4)
5. Note that breaking fate sharing between the control and data planes induces a lower-bound on control plane convergence, namely

   where *RTT* is “round trip time” and *PPT* is “packet processing time”. [↑](#footnote-ref-5)