The Interface to the Routing System

Elias Hazboun
Supervisor: Edwin Cordeiro
Innovative Internet Technologies and Mobile Communications WS2015
Chair for Network Architectures and Services
Department of Informatics, Technical University of Munich
Email: hazboun@in.tum.de

ABSTRACT

Management of today's ever growing communication networks is posing a challenge for network operators around the globe; networks are expected to react quickly to change, and automation is steadily becoming a necessity to cope with complexity. This paper presents one of the protocols proposed by the Internet Engineering Task Force (IETF) to cope with such a challenge, called the Interface to the Routing System (I2RS). The protocol focuses on routing operations in networks by offering operators standardized programmatic interfaces to the routing information stored in their devices. I2RS is based on NETCONF protocol while its data models are based on YANG modeling language. We argue that I2RS is a viable solution for the challenge at hand and has a good business case for operators since it leverages existing routing protocols and does not require a potential overhaul of network architectures.

Keywords

I2RS, Software defined networking, network management, NETCONF, YANG, routing system.

1. INTRODUCTION

Traffic carried by networks whether mobile, global or intra-data-center is rapidly increasing, one estimate by Cisco predicts that the annual global IP traffic will surpass 2.0 zettabytes (10^{21} bytes) by 2019 which translates to a three-fold increase from 2014 [1]. Moreover, businesses are moving fast and are expecting their underlying networks to be able to follow suit [2]. Therefore, network operators today are facing more and more pressure to be flexible to match business change and to be efficient to cope with the ever-growing traffic.

Software defined networking (SDN) is one possible approach to offer this flexibility. SDN development and large scale implementations [3] show that SDN is a viable solution for the current network challenges [4]. In traditional networks, the control plane and the data plane are distributed across complex devices in the network. On the other hand, SDN is an approach to networking in which control of the network or what is called the control plane is partially or fully centralized in a logical entity and decoupled from the actual forwarding devices which carry the traffic [5].

Nevertheless, both SDNs and traditional networks are facing a challenge to support complex automation and quick policy-based interaction with network operations. Today, these aspects are supported usually by proprietary and limited protocols such as Cisco ACI [6] which do not facilitate use in multi-vendor networks [7].

This gap between the needs of network operators and the standard solutions available motivated vendors and carriers to investigate a new protocol, which culminated in early 2013 with the Internet Engineering Task Force (IETF) creating a working group to find such a solution particularly aimed towards routing operations in networks, known as the Interface to the Routing System (I2RS) [8]. Figure 1 illustrates where the new protocol I2RS interacts with a network compared to the currently used SDN protocol Openflow [9] (Not shown in the figure is the possible interaction between I2RS and the data plane for retrieving data flow information).

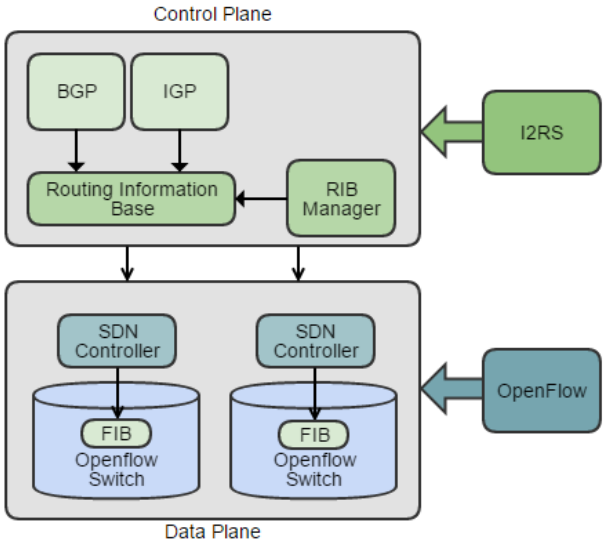

Figure 1: I2RS and Openflow interactions

In this paper, we present and shortly analyze the idea of I2RS and the protocol being developed at IETF. The rest of the paper is structured as follows: section 2 describes the driving force behind I2RS. Section 3 focuses on the architecture of I2RS and the different elements in it, while section 4 describes its data model. Section 5 mentions typical use-cases for the protocol as set out by the IETF and section 6 offers brief analysis of I2RS. Finally, we conclude the paper in section 7.

doi: 10.2313/NET-2016-07-1_04

2. THE NEED FOR A NEW PROTOCOL

We must first understand the current state of affairs of accessing information on routing devices to recognize the key requirements expected from I2RS and its goals.

Today, a typical network operator manages a large number of routing devices purchased from multiple vendors and running a variety of routing protocols. These devices maintain information that is integral to their function. For example, a Routing Information Base (RIB) contains routes to network destinations learned from routing protocols such as BGP or OSPF. Additional information might include counters and statistics in addition to packet forwarding rules pertaining to the forwarding plane of these devices.

Access to the previous information is an essential part for successful network management. To this end, operators usually use a combination of the following three methods.

- Command line interface (CLI): vendor specific commands are entered by a network engineer in a Unix like shell to edit or learn device states.
- Simple Network Management Protocol (SNMP): a historic and popular protocol that is most often used to retrieve (and to lesser extent modify) state information about devices. It is based on simple scalar data types to represent network configuration data [10].
- Network Configuration Protocol (NETCONF): a modern protocol that uses remote procedure calls (RPC) to configure devices; it is focused on being simple and extensible therefore it uses XML for data encoding [11].

Vendors have realized the shortcomings of using legacy techniques in network-oriented applications and automation, for example CLI scripting is not easy to use when it comes to data retrieval and filtering, while SNMP users are faced with lack of both configuration semantics and expressing power of models. Consequently, proprietary protocols and interfaces to access information have emerged through the years to solve specific use-cases. These protocols however are limited and hard to integrate into multi-vendor networks, nevertheless they are used today. Their deployment demonstrates the dire need for a standardized method of accessing and manipulating routing information [7].

This is where I2RS steps in. It is in a sense, not a replacement of the three methods of management mentioned above, but a new method that focuses on creating a standardized data-model driven interface for the secure and dynamic access of information in routing devices. Thus, I2RS can be defined as "a programmatic asynchronous interface for transferring state into and out of the internet routing system" [12].

I2RS is still a work-in-progress and some of its aspects have not been agreed upon completely yet. Nevertheless, from the previous discussion, we can already derive four key aspects that drive I2RS development as proposed by the IETF. First, it has to offer a fast, programmatic, asynchronous access for atomic operations. Second, it has to offer access to information not easily accessible by existing configuration protocols. Third, it has to offer the ability to retrieve data as well as to subscribe to event notifications from devices. Fourth, it has to be data-model driven to facilitate extensibility and standardization [12].

3. ARCHITECTURE

The architecture of I2RS is designed in a way that facilitates control and enables network applications to be built on top of networks. In this section we explain I2RS architecture, its properties and interactions.

3.1 Architectural Properties

I2RS protocol has to possess some defining properties to achieve its goals. The IETF has laid these out in [12] and can be summarized as such:

Perhaps the most logical property is *simplicity*, since most network operators agree that complex protocols are often error prone and are difficult to operate and implement correctly. However, maintaining simplicity of a protocol that accesses a wide variety of data types stored on different types of devices can be a challenge of its own.

Additionally, for a protocol that is reliant on modeling current and future data, I2RS must have *easy extensibility* or otherwise its limitations will quickly catch up to its potential and hinder its adoption. That is why the IETF is being careful with designing models that are extensible in a straightforward fashion.

Moreover, *model-driven programmatic interfaces* are required since current routing data models and the mechanisms to access them on devices are not standardized and are governed by vendor specific rules. This hinders interoperability and the ease of application implementation. Hence, I2RS must utilize a standard model-driven protocol which facilitates data access through automated applications.

Finally, *performance and scalability* are expected of I2RS because routing systems are anticipated to have a high number of operations and changes per second while requiring low latency execution to ensure smooth management. One method for achieving scalability is through filterable access to data, while another is through multi-channel communication between I2RS clients and agents as discussed in subsection 3.2.

3.2 Major Architectural Components

In terms of the architecture of I2RS, we identify five major components [12] and discuss their interactions below.

Network application: a network oriented piece of software with the goal of accessing or manipulating network states. It achieves its goal by communicating with I2RS clients.

I2RS client: an entity that implements the I2RS protocol and communicates with I2RS agents to access their services, in order to gain access to network information or to modify it. A client could be either an external I2RS library or simply the piece of code that is I2RS aware inside an application.

I2RS service: a set of functions for information access and

modification coupled with their usage policy. They are defined by a given data-model such as MPLS or BGP services which provide access to MPLS and BGP related states respectively.

I2RS agent: an entity that actually interacts with the routing element sub-systems to obtain and modify their states; it provides this functionality as an I2RS service for requesting clients. How agents access this information is out of the scope of I2RS. However, how the data is presented, i.e. its models, is an integral part of I2RS.

Routing element: in the scope of I2RS, a routing element is any device that implements some functionality pertaining to routing; it could be a traditional router implementing BGP or the logical control plane of an SDN controller.

No matter what the implementation of a specific routing element is, an I2RS agent's behavior to clients must not be affected. For example, in the case of a physically distributed routing element, an agent should still support the access of data from the whole element. Additionally, multiple agents can reside on a single routing element; in that case, they must be responsible for the service of separate sets of information to ensure simplicity.

3.3 Roles, Identities, and Priorities

Access to such delicate information on routing elements must have some kind of access control and tracking; to this end I2RS defines roles that can be assigned to clients, where a role of an I2RS client defines its read/write and subscription rights (called scopes). Furthermore, I2RS assigns identities to clients which agents use for authentication [13]. Finally, a client may have a priority attribute that can aid in case of state access conflicts which we will discuss at the end of subsection 3.4.

3.4 I2RS Interactions

Figure 2 illustrates the components mentioned in subsection 3.2 and their associated connections. An application can communicate with multiple local and remote I2RS clients and conversely an I2RS client can respond to multiple applications. Clients on the other hand are served by agents running on routing elements, where if necessary, a single client can request from multiple agents on the same or different routing elements. Finally, an agent is able to serve more than one client at a time.

I2RS is mainly focused on the interactions between agents and clients, where agents provide - through their advertised services - the ability for clients to access and modify data on the routing elements in addition to the ability to subscribe to events affecting these elements.

Agents can access the data of three components on the routing elements: The *routing subsystem* which includes the RIB manager and routing protocols like BGP, the *dynamic system state* which includes the various counters and data flow information, and finally the *static system state* which includes data pertaining to the system itself such as interface information. One point to note is that I2RS agents are not directly accessing the Forwarding Information Base

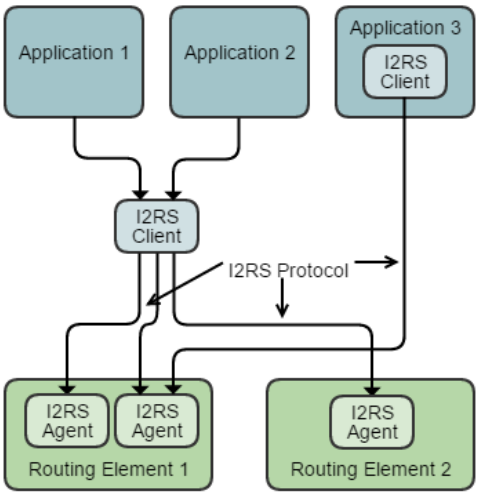

Figure 2: Interactions between I2RS clients and agents

(FIB), but rely on devices themselves to translate I2RS RIB changes into corresponding FIB entries on their own.

Routing elements and their agents keep track of I2RS state changes in the I2RS data store which comprises records of changes and their requesting clients as well as the active subscriptions by clients. A data store is only stored in memory and will be lost upon reboot, hence it is called an ephemeral state [14]. Therefore, any implementation must be careful to specifically assign I2RS changes as ephemeral, so that even when the running configuration is copied to a persistent memory for example, these changes will not. Using the data-store, an agent must have the ability to roll back changes it has applied since the client interaction when necessary; this roll back usually reverts to the state specified by other means of device configuration.

As agents modify these states on routing elements, conflicts may arise between requested modifications and the configuration provided by means other than I2RS such as SNMP or CLI. In such cases, a clear operator policy must be in place to enforce a pre-determined behavior. Whatever the policy may be, the agent must notify the requesting client if their request was blocked.

Additional conflicts can arise from two clients trying to modify the same state on an I2RS agent; however, this case should be avoided as I2RS considers it an error. Nevertheless, if such a case does occur, an agent must try to resolve this issue by first considering the priorities of clients or by a first come first served basis. No matter the conflict scenario that might occur in agent implementation, the design property of simplicity dictates that the behavior must be predictable and the error reportable to affected clients.

3.5 Notable Protocol Considerations

The IETF working group opted for a model of I2RS where existing protocols are utilized and leveraged as much as possible. To that end, it was agreed that the I2RS shall be based on NETCONF [11] and its close sibling with a REST

interface RESTCONF [15]. Nevertheless, the I2RS working group has also recognized that some modifications must be made to these protocols before being used in I2RS, especially in regards to security aspects. As for the transport layer protocol, the working group has left it as an operator chosen aspect, as long as features of integrity, authentication and ease of deployment are fulfilled.

In regards to the atomicity of operations performed by agents, no guarantees beyond a single client message shall be made. An agent will not try to have roll back mechanisms for multiple client messages. This limitation is in a sense a feature that pledges simplicity and predictable behavior which are part of the goals of I2RS. In the case of error handling within a single message, a client can signal to the agent one of three kinds of behavior; perform all operations or none at all in case of error, perform all operations up to the point of error or attempt to perform all operations regardless of errors. No matter the behavior set by the client, an agent must reply with explicit success or failure messages to requesting clients.

3.6 Security Considerations

I2RS exposes sensitive interfaces to the routing system, the access of which requires security guarantees in order to be adopted by operators. In this section, we present major I2RS security aspects.

First of all, I2RS assumes that a routing element can trust an I2RS agent residing on it, since it is a part of it, whether as a part of the operating system image or as a signed add-on to it. However, such trust cannot be established between a client and an agent, therefore some kind of mutual authentication must take place before operations can be permitted. A client must be able to verify the identity of the agent it is trying to communicate with and its attached routing element. Additionally, an agent must be able to authenticate a client based on its supplied identity in the communication channel [13].

Using this identity, an agent can link a client to a role that has a set of specific scopes (read/write and subscription rights); these in turn will be used for authorization purposes before performing any operations on behalf of the client.

Moreover, data confidentiality is vital for sending sensitive configuration and statistics over the network; operators are reluctant to transport network information in plain text. However, I2RS acknowledges that there should be support for cases where confidentiality is not explicitly needed by an operator. As a result, communication channels may support unsecure transport layer protocols. As for data integrity, the proposed I2RS protocol should be able to protect against data modification in transit as well as replay attacks where messages are merely repeated by attackers [13].

4. INFORMATIONAL MODEL

At the heart of I2RS are the standard data models and their semantics, which serve as interfaces for information in routing elements. These models should be extensible by design and try to - if possible - use preexisting data models. The I2RS working group has chosen YANG [16] to be the language used for modeling. YANG was also developed at the IETF and is an acronym for "Yet Another Next Generation", it is a modern data modeling language that uses trees to model configuration and state data. Although not based on XML, it has an associated language called YIN which maps its data models into XML. Yang features a small set of prebuilt standard models but supports extensibility through derivation for vendor created data types [17].

I2RS aims primarily to use YANG to model the states and elements in the RIB, which is where a standardized programmatic interface is currently critically missing. However, I2RS agents shall support services for other states in a routing element ranging from Border Gateway Protocol (BGP) and Inter-gateway Protocol (IGP) to Quality of service (QoS) and policy mechanisms.

To be vendor agnostic, the I2RS information model must be compatible with all the various routing elements in the network and their different implementations. To that end, I2RS borrows from object oriented paradigm to define object classes, types and inheritance. For example, a parent class can have all the common attributes found in all routing devices, while its subclasses add vendor or use-case specific attributes. Moreover, an agent may not support all classes or attributes in a service and shall communicate to requesting clients through a capability model what it currently offers.

Finally, objects in routing elements seldom exist alone and are rarely unaffected by other objects, thus the I2RS information model must express these relationships as clear and robust as possible.

Figure 3 shows an example of partial modeling of a route object in the RIB, a route is matched based on one of five criteria and of course for a given route a next hop must be given.

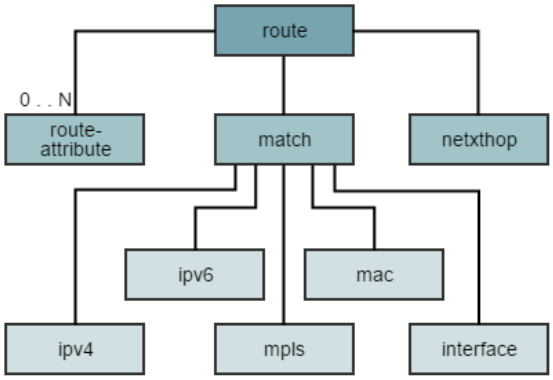

Figure 3: Interactions between I2RS Clients and agents

5. USE CASES

For a new protocol to succeed and be adopted by network operators, it has to meet their needs and expectations. It has to stem from their current and anticipated requirements. Therefore, I2RS must be designed with its envisioned use-cases in mind. Nevertheless, covering all possible use-cases can lead to the protocol bearing too many responsibilities and becoming too complex. That does not mean however,

that a protocol must simply ignore proposed use-cases, but perhaps leave their support for the future or vendor specific extensions.

So for a protocol that promises standardized access for routing information across a wide variety of devices, the task of adopting and supporting use cases becomes a balancing act that should be carefully analyzed. The following are two example use-cases accepted by I2RS working group [18].

Distributed Reaction to Network Based Attacks: I2RS can be used to quickly modify control planes in case of attacks to either filter or direct suspicious traffic to analyzers. I2RS here is essential for three key elements: quick reaction times, distributed control, and the injection of temporary states that do not affect long term policies installed. Today, administrators handle this use-case by either manually entering (and later deleting) commands to filter traffic, or by asking their network provider to do such a task from their end. Both of which are not as quick and are more error prone than an automated solution provided by I2RS.

Intra-Data Center Routing: Data centers today are rapidly increasing in size and network operators are resorting to applying large multi-tiered topologies with BGP and IS-IS as routing protocols. I2RS provides operators in data centers quick access to topology changes and data flow information, which in turn translates into faster adaption and insertion of new routing policies as needed.

One possible example to the first use case is an application that collects statistics in a network. The application could use a library that acts as an I2RS client to subscribe to relevant notifications offered by I2RS agents. In one scenario, the application receives enough notifications that could lead it to conclude that an attack is being mounted on a part of the network. The application could then - using the I2RS client - issue write operations to agents in the affected part to defend against the attack by changing the routing policy. Moreover, when the attack is sensed to have come to an end, the application could revert its policy change. Such automation and interaction using complex reasoning is a key element for the future of traditional networks and SDNs.

From the previous two use-cases in addition to others, some frequent interactions of I2RS can be deduced:

- Accessing routes currently installed in the RIB as well as receiving near real-time notifications in case of their removal or change.
- Installing source and destination based routes in the RIB with all their related information.
- Interacting with traffic flow and other network traffic measurement protocols to determine path performance and make path decisions.

6. ANALYSIS

I2RS as a protocol is still in development. Nevertheless, based on the working group drafts and its reliance on NETCONF and YANG, we can analyze it and discuss its possible future implementations.

6.1 Possible Implementations

I2RS still has no existing real world implementations yet. However, some implementation efforts are already underway; one of which is currently planned at our chair for network architectures and services; we present here two approaches for possible I2RS implementations. The first approach, currently spearheaded by the chair of I2RS working group (Susan Hares) [19], relies on integrating I2RS into OpenDaylight (ODL) which is an open source platform for programmable SDNs [20]. This approach benefits from the capability of ODL to access the Linux kernel and is also supported by industry vendors who already expressed interest in supporting ODL. The other approach which is based on Bird (an open source software routing project) [21], relies on programming I2RS as another protocol like BGP or OSPF, which will allow I2RS to directly access and manipulate routes as needed [22].

These implementation efforts are essential for the development of I2RS because they prove its feasibility and deployability and most importantly they expose unanticipated shortcomings, as stated by David Clark and added to IETF Tao: "We believe in rough consensus and running code" [23]. For example, at the 94th IETF hackathon, the team working on I2RS found out that I2RS lacked any information regarding secondary pathways for sending analytical data by other protocols such as IPFIX [24], which prompted that more work and specification must be done in that area [19].

6.2 SDN & I2RS

A comparison can be made between I2RS and OpenFlow which drives SDN at the moment. OpenFlow is focused on direct interaction with the forwarding plane and essentially treats devices as simple switches [9]. On the other hand, I2RS is focused on policy change and the RIB, and actually depends on the device itself to do the appropriate forwarding plane changes. Furthermore, I2RS relies on much more interaction with existing routing protocols and technologies to enable reuse and easy deployment.

For that reason, we believe that I2RS succeeds in proving its business case to network operators better than a complete SDN solution, since its adoption won't mean the complete change of the network architecture like most SDN solutions require. They can still use all the existing solutions for routing and control plane to FIB communication, but they now have the power to automate complex operations across all their various devices. Nevertheless, it can also be projected that I2RS will help the gradual introduction of SDN approaches to traditional networks and even facilitate the adoption of what is called hybrid SDNs [25].

6.3 Reliance on NETCONF and YANG

The possibility of using NETCONF and YANG for automation of network operations was of interest long before the I2RS working group was created. A 2011 paper [26] by Tail-f Systems, highlighted the benefits of using a rich language such as YANG compared to solutions based on SNMP, in addition to the importance of transaction-based management protocol that supports consistency checking such as NETCONF.

Based on these standards, we can already have some un-

derstanding of I2RS expected performance. For example, the use of XML in NETCONF grants three main benefits: human-readability which facilitates debugging, flexibility in structuring data, and extensibility of message format [27]. Moreover, multiple studies have analyzed resource usage and efficiency of XML based solutions compared to other management protocols and the consensus was that with proper processing and data compression, XML outperforms legacy management solutions [28] [29]. Additionally, several implementations of NETCONF/YANG have been done by both the industry and the academia with partial support of older protocols like SNMP. This means that although I2RS is a new protocol, its reliance on modern yet tested protocols gives it robustness and a starting point for early implementers.

7. CONCLUSION

In this paper, we presented the I2RS protocol which is currently in development by the IETF as a solution for the challenge of accessing information stored in the routing system of today's complex and ever growing networks. I2RS aims to offer programmatic standardized interfaces in which read/write operations and event notification subscriptions are offered to network oriented applications. Its base protocols of NETCONF and YANG have both been proved to be efficient and are currently supported by major vendors and operators. Some considerations must still be amended to both of them before being adopted, for example, in terms of security.

Finally, a protocol's specification and architectural soundness do not guarantee its adoption, it must also be able to present a case for its adoption to network operators. To this end, we have shown that I2RS offers a middle ground between traditional networks and complete SDN solutions, where operators are not required to change their entire infrastructure but can deploy I2RS to leverage existing routing protocols.

8. REFERENCES

[1] Cisco Inc. Cisco Visual Networking Index: Forecast and Methodology, 2014-2019. White paper, Cisco Inc., May 2015. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.pdf (Accessed 9 Dec. 2015).

[2] Susan Hares and Russ White. Software-Defined Networks and the Interface to the Routing System (I2RS). *IEEE Internet Computing*, 17(4):84–88, 2013.

[3] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. B4: Experience with a Globally-deployed Software Defined Wan. *SIGCOMM Comput. Commun. Rev.*, 43(4):3–14, August 2013.

[4] Hyojoon Kim and Feamster, N. Improving network management with software defined networking. *Communications Magazine, IEEE*, 51(2):114–119, February 2013.

[5] Open Networking Foundation. Software-Defined Networking: The New Norm for Networks. White paper, Open Networking Foundation, Palo Alto, CA, USA, April 2012. http://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf (Accessed 20 Dec. 2015).

[6] Cisco Inc. Cisco Application Centric Infrastructure Microsegmentation Solution. White paper, Cisco Inc., 2015. http://www.cisco.com/c/en/us/solutions/collateral/data-center-virtualization/application-centric-infrastructure/white-paper-c11-736420.pdf (Accessed 9 Dec. 2015).

[7] Alia Atlas, Tom Nadeau, and David Ward. Interface to the Routing System Problem Statement. Internet-Draft draft-ietf-i2rs-problem-statement-08, IETF Secretariat, December 2015. https://datatracker.ietf.org/doc/draft-ietf-i2rs-problem-statement/ (Accessed 20 Dec. 2015).

[8] I2RS Working Group. Interface to the Routing System Charter. Working Draft, October 2015. https://datatracker.ietf.org/doc/charter-ietf-i2rs (Accessed 16 Dec. 2015).

[9] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. OpenFlow: Enabling Innovation in Campus Networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, March 2008.

[10] Jeffrey D. Case, Mark Fedor, Martin Lee Schoffstall, and James R. Davin. Simple Network Management Protocol (SNMP). STD 15, RFC Editor, May 1990.

[11] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman. Network Configuration Protocol (NETCONF). RFC 6241, RFC Editor, June 2011.

[12] Alia Atlas, Joel Halpern, Susan Hares, David Ward, and Tom Nadeau. An Architecture for the Interface to the Routing System. Internet-Draft draft-ietf-i2rs-architecture-10, IETF Secretariat, November 2015. https://datatracker.ietf.org/doc/draft-ietf-i2rs-architecture/ (Accessed 19 Dec. 2015).

[13] Susan Hares, Scott Brim, Nancy Cam-Winget, Dacheng Zhang, Qin Wu, Ahmed Abro, Salman Asadullah, Joel Halpern, and Eric Yu. I2RS Security Considerations. Internet-Draft draft-hares-i2rs-security-03, IETF Secretariat, February 2015. https://datatracker.ietf.org/doc/draft-hares-i2rs-security/ (Accessed 11 Dec. 2015).

[14] Jeffrey Haas and Susan Hares. I2RS Ephemeral State Requirements. Internet-Draft draft-ietf-i2rs-problem-statement-02, IETF Secretariat, March 2016. https://datatracker.ietf.org/doc/draft-ietf-i2rs-ephemeral-state/ (Accessed 19 Dec. 2015).

[15] Andy Bierman, Martin Bjorklund, and Kent Watsen. RESTCONF Protocol. Internet-Draft draft-ietf-netconf-restconf-09, IETF Secretariat, December 2015. https://datatracker.ietf.org/doc/draft-ietf-netconf-restconf/ (Accessed 18 Dec. 2015).

[16] M. Bjorklund. YANG - A Data Modeling Language

for the Network Configuration Protocol (NETCONF). RFC 6020, RFC Editor, October 2010.

[17] J. Schönwälder, M. Björklund, and P. Shafer. Network Configuration Management Using NETCONF and YANG. *Communications Magazine, IEEE*, 48(9):166–173, Sept 2010.

[18] Russ White, Susan Hares, and Alvaro Retana. Protocol Independent Use Cases for an Interface to the Routing System. Internet-Draft draft-white-i2rs-use-case-06, IETF Secretariat, July 2014. `https://datatracker.ietf.org/doc/draft-white-i2rs-use-case/` (Accessed 18 Dec. 2015).

[19] Hares, Susan. "I2RS Implementation". I2RS Mailing List. IETF, 16 Dec. 2015. `https://mailarchive.ietf.org/arch/msg/i2rs/KDvuzS3B_cvyF6nA-zZkKE3Ibo4` (Accessed 20 Dec. 2015).

[20] J. Medved, A. Tkacik, R. Varga, and K. Gray. OpenDaylight: Towards a Model-Driven SDN Controller architecture. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*, pages 1–6, June 2014.

[21] The Bird Internet Routing Daemon Project. `http://bird.network.cz` (Accessed 15 Dec. 2015).

[22] Private correspondence with Bird Developers. "I2RS Implementation". 17 Dec. 2015.

[23] Paul Hoffman. The Tao of IETF: A Novice's Guide to the Internet Engineering Task Force, 11 2012. `https://www.ietf.org/tao.html` (Accessed 20 Dec. 2015).

[24] B. Claise, B. Trammell, and P. Aitken. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. STD 77, RFC Editor, September 2013. `http://www.rfc-editor.org/rfc/rfc7011.txt`.

[25] C. E. Rothenberg, R. Chua, J. Bailey, M. Winter, C. N. A. CorrÃła, S. C. de Lucena, M. R. Salvador, and T. D. Nadeau. When open source meets network control planes. *Computer*, 47(11):46–54, Nov 2014.

[26] Stefan Wallin and Claes Wikström. Automating Network and Service Configuration Using NETCONF and YANG. In *Proceedings of the 25th International Conference on Large Installation System Administration*, LISA'11, pages 22–22, Berkeley, CA, USA, 2011. USENIX Association.

[27] Gerhard Münz, Albert Antony, Falko Dressler, and Georg Carle. Using NETCONF for Configuring Monitoring Probes. In *IEEE/IFIP NETWORK OPERATIONS & MANAGEMENT SYMPOSIUM (IEEE/IFIP NOMS 2006), POSTER SESSION*, 2006.

[28] James Yu and Imad Al Ajarmeh. An Empirical Study of the NETCONF Protocol. *International conference on Networking and Services (ICNS'06)*, 0:253–258, 2010.

[29] T.F. Franco, W.Q. Lima, G. Silvestrin, R.C. Pereira, M.J.B. Almeida, L.M.R. Tarouco, L.Z. Granville, A. Beller, E. Jamhour, and M. Fonseca. Substituting COPS-PR: an evaluation of NETCONF and SOAP for policy provisioning. In *Policies for Distributed Systems and Networks, 2006. Policy 2006. Seventh IEEE International Workshop on*, pages 10 pp.–204, June 2006.