Carrier-grade Network Management Extensions to the SDN Framework

Alisa Devlic, Wolfgang John
Ericsson Research
Kista, Sweden
{alisa.devlic, wolfgang.john}@ericsson.com

Pontus Sköldström
Acreo AB
Kista, Sweden
pontus.skoldstrom@acreo.se

***Abstract*— The concept of software-defined networking (SDN) recently gained huge momentum in the industry, driven mainly by IT companies interested in datacenter applications. In this paper, however, we consider SDN applied in the carrier domain, which poses additional requirements on the network architecture, including network management functions. We derive concrete requirements for the use-case of a virtualized multi-provider access/aggregation network based on carrier-grade SDN. We then provide initial architectural considerations for integration of network management extensions to the SDN framework as defined by the Open Networking Foundation (ONF). Architectural considerations include definition of the required entities and their interactions. Finally, we apply the proposed architecture on the access/aggregation network use-case, outlining procedures of how the network management extensions can enable network wide and node specific management & configuration.**

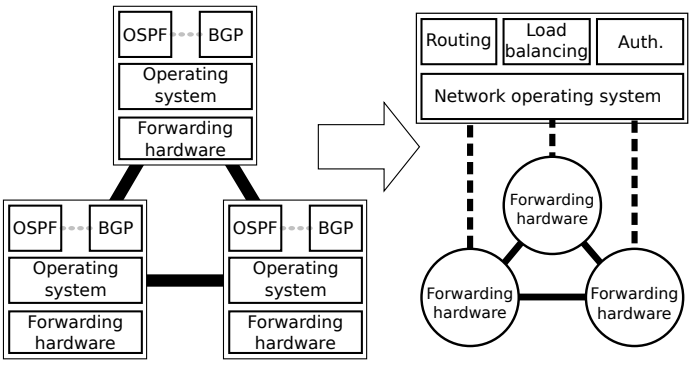

Fig. 1. From autonomous network elements to Software Defined Networking.

I. Introduction

Traditional network elements have been constructed as autonomous entities (see Figure 1 on the left) using a distributed control plane to communicate with the outside world. Various protocols allow to autonomously decide what actions to take. Typically this involves a number of processes running within a closed operating system (OS) calling a proprietary API which in turn causes the OS to program specialized forwarding hardware, again using a proprietary API. Adding new functionality to a network element usually involves standardizing a new protocol that reinvents mechanisms such as distribution and signaling, and waiting for the vendors to implement the new protocol. Comparing this process to the PC world reveals why network development is moving so slow compared to other areas. In the PC world, adding functionality can be done quickly by simply writing a piece of software and installing it on a machine. This is made possible by open APIs and reuse of existing functionality through software libraries.

Software Defined Networking (SDN) proposes a new model by creating open APIs between the hardware and the operating system, and between operating system and *network applications*. In the SDN model (seen to the right in Figure 1) a *Network Operating System (NOS)* is responsible for maintaining an up-to-date view[1] of the network and its current state. The NOS does not only maintain a view of the network but is also responsible for handling changes to the view and then transferring those changes to the network hardware. Changes to the view come from *network applications* running on top of the operating system. The network applications are software modules that are able to access the network view maintained by the NOS as well as to modify it. In this model, adding new functionality is greatly simplified: it only takes to write a software module utilizing the API provided by the NOS and the NOS is responsible for updating the network and distributing the new state. Functionality for distributing state and synchronizing autonomous systems does not have to be reimplemented since it is already handled by the NOS.

The most well known component in the SDN world is *OpenFlow* [1], an open protocol designed to expose the internals of a network element and provide an API to modify them. In the center of the protocol's model of a network element is the *FlowTable(s)*. FlowTables contain rules that can be used to match incoming packets (e.g. "destination IPv4 address 1.2.3.4" and "TCP destination port 80") and associated them to a number of actions (e.g. "modify the destination address" and "output the packet on interface 4"). If an incoming packet does not match an existing rule in the network element, the packet can be sent to the NOS where a network application can investigate the packet further and decide what to do, e.g. installing a new rule that takes care of all packets in this particular packet flow. The OpenFlow protocol is mainly intended for managing the FlowTable(s) mentioned above by installing permanent or transient rules with a relatively high churn-rate.

[1] A graph of the nodes and links in the network, with all their attributes.

A. Status of Network Management in SDN

The SDN framework enables centralized control of data path elements, independently of the network technology used to connect these devices that can originate from different vendors. The centralized control embeds all the intelligence and maintains the network-wide view of the data path elements and links that connect them. This centralized up-to-date view makes the controller suitable to perform network management functions, while allowing easy modifications of networking functions through the centralized control plane. The goal is to make the network management more intelligent and automated in order to decrease the operational expenditure (OPEX) for network operators. Due to its internal characteristics, SDN is also a good candidate technology to realize a multi-domain and multi-provider network management framework.

SDN research and development is currently centered around an official standardization body, the Open Networking Foundation[2], which was established in 2011. The ONF is currently standardizing two protocols, the OpenFlow protocol itself and the newly proposed **OF-Config**, a configuration and management protocol based on NETCONF [2]. NETCONF is a transactional protocol that uses remote procedure calls (RPCs) on top of a secure transport channel (such as SSH) to manage configurations on remote devices. It provides methods for installing, manipulating and deleting configuration not only on a single device but on multiple devices within a single transaction. If a new configuration causes problems in the network, all configured devices can be rolled back to the previous state. While NETCONF itself is XML based the data model - that describes what can be configured and how different classes of configuration relate to one another - is written in YANG [3]. OF-Config adopts the NETCONF protocol, extending it with specific YANG models.

B. Our contribution

SDN has gained momentum in the industry forum, being primarily driven by leading IT companies with a focus on Datacenter applications (e.g., Google recently adopted SDN in their operational network connecting data centers [4]). However, the focus of this paper is on SDN in modern carrier networks. We have studied carrier-grade use-cases, requirements and solutions within the framework of the EU FP7 SPARC project[3]. In order to be considered as a carrier grade network by network operators, an SDN solution needs to fulfill the following requirements defined by Metro Ethernet Forum (MEF): standardized services, scalability, reliability, quality of service (QoS), and service management [5].

In this paper, we focus on fulfilling the service management requirement, which is important to service providers to ensure high performance and availability of services that they provide to their customers. According to MEF, service management is defined as the ability to *monitor*, *diagnose*, and *centrally manage* the network, using standards-based vendor independent implementations. It also includes carrier-grade OAM (operations, administration and maintenance) and rapid service provisioning. In order to fulfill the service management requirement, we propose architecture extensions to the SDN framework to include these functions, while referring to them as *network management functions*. The proposed extensions are based on concrete requirements from the use-case of a virtualized multi-provider access/aggregation network. We apply the proposed architecture on the use-case and outline possible procedures of how the network management extensions allow network wide as well as node specific management and configuration.

[2]www.opennetworking.org

[3]www.fp7-sparc.eu

II. Use case

In this paper, we consider the primary SPARC use-case [6], depicted in Figure 2: A carrier-grade SDN controlled MPLS based access/aggregation network that provides services to clients. Directly on top of the physical network, a network virtualization system is running [7] [8] [9]. This enables multi-provider support by granting each service provider access to a slice of the physical network resources (note that the OpenFlow nodes in Figure 2 are actually virtual nodes connected by virtual links). The services that are provisioned on top of the virtual networks are either point-to-point connections to other clients (VPN services) or to the Internet, and point-to-multipoint services such as IPTV. Connectivity is provided by MPLS tunnels or pseudowires running on MPLS tunnels, which can be monitored using standard OAM tools such as Bidirectional Forwarding Detection (BFD). Connectivity services in the aggregation network can be provisioned in three ways: i) through the NOS by the different service providers; ii) through interaction with the IP/MPLS core through existing NNI protocols such as LDP or RSVP-TE; and iii) through interaction with the clients (e.g. IGMP).

In the SPARC project, this use-case has been implemented with basic network management functions integrated in an ad-hoc fashion. To the best of our knowledge, besides recent ONF activities, there are no comprehensive network management proposals for SDN. In the next section we gather requirements for network management from different standardization bodies and the experiences of the SPARC project, on which we can base our initial architectural considerations.

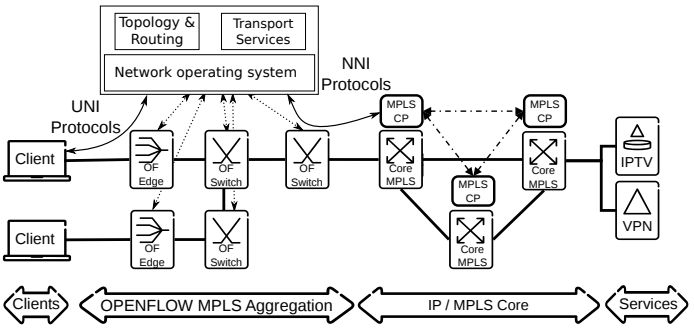

Fig. 2. The use case considered: A carrier-grade SDN based MPLS access/aggregation network providing clients with services provisioned through a legacy IP/MPLS core.

III. Network Management Requirements

A network management model describes a set of recommendations or a framework to design a network management system (NMS). Several network management models have been defined through the years by different standardization bodies, focusing on management of different network technologies. The International Organization for Standardization (ISO) defines five functional areas of network management from a network-centric perspective: Fault, Configuration, Accounting, Performance, and Security management - the so called OSI FCAPS model [10]. As an alternative view, a more business oriented model was later defined in the Enhanced Telecom Operations Map (eTOM) by the TeleManagement Forum (TMF) [11] - the FAB model (short for Fulfillment, Assurance, and Billing)

The ITU-T introduced the Telecommunications Management Network (TMN) framework [12] as a logical layered architecture consisting of five management layers, each providing the appropriate FCAPS functionality according to the layer definition and passing the collected information to the next layer: network element layer, element management layer, network management layer, service management layer, and business management layer. In this paper, we also follow the OSI FCAPS model in order to define the basic network management functions required, i.e. *fault, configuration, accounting, performance*, and *security management*.

A clear trend in network management is to strive for increased self-planning and self-configuration capabilities of network functions, following the spirit of Self-Organizing Networks (SON) introduced by 3GPP in 2008 for radio access networks [13]. SON-like features enable operators to easier plan, configure, manage, optimize and heal the network. A main motivation for SON features is to reduce the human effort introduced due to the ever increasing network complexity, which is in contrast to the operators desire to reduce operational costs (OPEX). Thus, a major goal of modern network management systems is to simplify network operations and to reduce the human involvement by automating the management functionality, while allowing the operator to remain in control. From these trends, we derive *auto-deployment* and *auto-configuration* of the network elements as the ultimate goal for our SDN network management framework.

Having traditional network management models and the goal of automated network management in mind, we derived a set of functional requirements. Our list of requirements is furthermore based on the recommendations of the ONF regarding configuration and management of SDN networks [14] as well as carrier-grade requirements identified by the FP7 SPARC project [6]. Following the ONF's recommendations, the primary task of the network management framework is basic configuration of (virtual) network elements, including basic *device management* and *bootstrapping* (plug and play) of network elements and the control network. However, the ONF recommendations also include *operational requirements* on configuration, which go beyond the scope of initial configuration, such as connectivity configuration and tunnel management. These operational requirements could benefit from the transactional communication model as offered by OF-config and NETCONF. Finally, as identified in SPARC, *carrier-grade* networks pose additional requirements on network management, such as management support for multi-provider operation.

The following list summarizes the resulting functional requirements for carrier-grade NM extension to SDN:

I Device management:
- Password and Certificate management
- Firmware management
- Network booting (e.g. PXE)

II Bootstrapping:
- Resource discovery
- Instantiation of logical switches
- Port and Queue configuration
- IP address management
- OF Controller discovery
- Control Network discovery
- Virtual link management

III Operational configuration:
- Capability discovery
- Topology discovery
- Tunnel management
- Connectivity configuration

IV Additional carrier-grade requirements:
- Multi-provider support
- Support for configuration of OAM tools
- Event triggers from network elements

IV. Architectural considerations

Figure 3 shows the current SDN architecture as defined by ONF [15]. In this model, an *OpenFlow capable switch*, which is a physical or virtual network element, is hosting one or more *OpenFlow logical switches*. The logical switches represent the actual OpenFlow network elements, which are controlled by one or more *OF Controllers* via the OpenFlow protocol. *Network Apps* on top of the OF Controller use the network via the OF Controller's northbound API (NB API)). Finally, an *OF Configuration Point* represents the service which communicates via the OF-Config protocol with an OpenFlow capable switch and partitions resources among OF logical switches (such as ports and queues).

The current ONF model fulfills the requirement groups I and II (see section III) which are purely focused on configuration management. However, during operations, the OF Controller is not supported with further operational configuration tasks, since OF Controller and OF Configuration Point are not interfacing each other.

In Figure 4 we extend the ONF model with additional functions, also considering requirement group III. We propose to unite the OF Controller and OF Configuration point in an NM-enabled NOS, allowing sharing of data and synchronization of

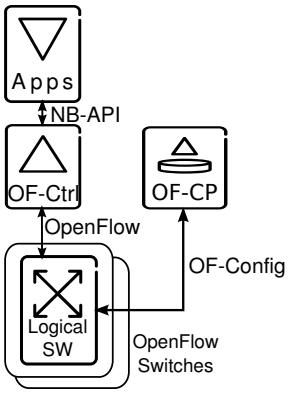

Fig. 3. The current ONF SDN model with an OF Configuration Point (OF-CP) separate from the OF Controller (OF-Ctrl).

their network views. The extended NOS includes additional NM functions providing alarm management, diagnosis of faults and performance degradations, and other NM functions according to FCAPS. Via the OF Controller's northbound API, NM applications can proxy connections from external NMS entities to the NM functions. External NMS entities can range from full-fletched NMS systems, customized NM applications enforcing local policies to web interfaces used to provision Openflow network monitoring, visualize monitoring state, and validate SLAs.

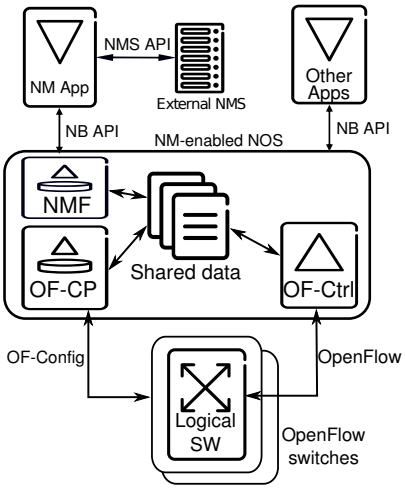

Fig. 4. Our extended ONF SDN model in which the OF Configuration Point and the OF Controller are tightly bound in the NOS. Furthermore, additional Network Management functions (NMF) and optional connections to an external NMS are included.

A further evolution of this architecture in depicted in Figure 5, in which a multi-provider carrier network is considered (requirement group IV). In order to configure and bootstrap the virtual networks based on slices of the physical network resources, a *Master OF Configuration Point* is used by the owner of the network infrastructure (the Network Owner [16]). The virtual networks can then be leased by individual service providers. Each service provider can manage and control its virtual network using its own NM-enabled NOS, similar to the one depicted in Figure 4. Additionally, each service provider can integrate further operational carrier-grade specific functions into the NOS or NM Apps, such as configuration of OAM tools or specific event triggers on the network elements.

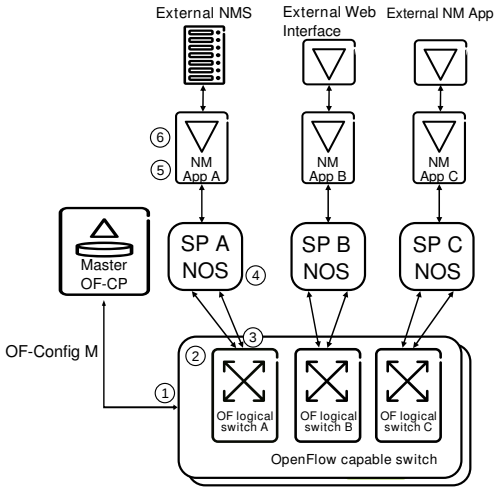

Fig. 5. Our SDN model for virtualized networks. Here a Master OF Configuration Point belonging to the Network Operator is responsible for configuring virtual networks. The virtualized logical OF switches are controlled and managed by a NM-enabled NOS (similar to the one depicted in Fig. 4) belonging to different service providers (SP).

V. Applying the Proposed Architecture

Going back to our use-case in Figure 2, the process of configuring the network with our multi-provider architecture can be described. Here we go through the steps of **connecting a new service provider and setting up an OAM monitored pseudowire**. This involves both network (and virtual network) wide management setup as well as network wide and node specific configuration. The steps performed to go through this process are enumerated in Figure 5 and described below:

1) **Device configuration and bootstrapping:** The Network Operator first needs to configure a minimal part of each network element manually, e.g., by connecting to a default IP address using SSH and setting a management IP address and a password. Next, the Network Operators Master OF Configuration Point can connect to the device and continue the configuration. The OF Configuration Point may at this stage upload a new firmware image, SSH keys, X.509 certificates, and retrieve the capabilities & available resources on each node. Other resources that may need configuration at this stage (or at the next stage) are ports and queues on the switches.
2) **Virtual network creation:** The Network Operator configures a virtual network. The virtual network *view* is injected by the Network Operator's NMS into the Master OF Configuration Point, which configures all involved switches using OF-Config. The deployed configuation parameters include port assignments to the virtual networks, size of the allocated bandwidth share, and assignment of other resources (e.g., address spaces). If the virtual

topology is different from the physical topology, it may be necessary to configure tunnels in order to create virtual links. Other parameters that may need configuration are the edge interfaces that are connected to customer equipment. These may need to be configured for example with a specific VLAN in order to map traffic from a service provider into the virtual network.

3) **Connectivity bootstrapping:** Once the virtualization specific parameters have been configured by the master OF Configuration Point and logical switches are instantiated, the master sets the IP addresses of the logical switches. Furthermore, the IP address(es) of the service provider's OF Controller(s) are configured in the logical switches as well.
4) **Topology and capability discovery:** After the service provider's OF Controller has connected and has been authenticated, the network is in an operation state. However, some management tasks are still left. For example, the OF Controller must retrieve the connected switches' capabilities and initialize topology discovery of the connected virtual topology.
5) **Pseudowire creation:** When the topology has been discovered by the service provider's OF Controller it can start to configure a pseudowire between two external ports. This can be triggered either by the service provider's NMS, by incoming traffic or control protocols on the network element level. The service provider's OF Controller calculates a path through the virtual network and establishes an MPLS tunnel on all involved nodes (using the OpenFlow protocol). It can then configure the pseudowire at the edges using either OpenFlow in case of an Ethernet-based pseudowire or OF-Config if for example a TDM interface needs to be configured.
6) **OAM configuration:** Once the pseudowire has been established, the service provider can install monitoring and other OAM tools. OAM tools may require configuration through both OF-Config (for creating and configuring monitoring endpoints) and OpenFlow (for associating OAM packets with the packet flows to be monitored).

VI. Summary and Conclusions

We have investigated the current status of network management in the ONF SDN framework with respect to carrier-grade multi-provider networks. As a first step towards a more complete network management solution for SDN networks, we have derived a set of requirements from traditional network management forums, from the ongoing ONF standardization efforts, and from our own work in the FP7 SPARC project. Based on the derived requirements we have extended the ONF SDN model to support multi-provider network management functions in carrier networks. This initial study is only the first step towards an automated network management framework for future SDN-based carrier networks. Based on this study, we plan to take the next step by designing interfaces between control, management, and datapath elements, including extensions to the existing protocols such as OpenFlow and OF-Config.

Acknowledgments

This work was funded by the European Commission under the 7th Framework ICT research programme projects SPARC. The authors would like to thank all SPARC partners for discussions and comments.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[2] R. Enns, "NETCONF Configuration Protocol," RFC 4741 (Proposed Standard), Internet Engineering Task Force, Dec. 2006, obsoleted by RFC 6241. [Online]. Available: http://www.ietf.org/rfc/rfc4741.txt

[3] M. Bjorklund, "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)," RFC 6020 (Proposed Standard), Internet Engineering Task Force, Oct. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc6020.txt

[4] A. Fewell. (2012, April) Google showcases openflow network. [Online]. Available: http://www.networkworld.com/community/blog/google-pwns-networking-part-1

[5] Metro Ethernet Forum. (2012, April) Carrier ethernet defined. [Online]. Available: http://metroethernetforum.org/page_loader.php?p_id=140

[6] The SPARC consortium, "SPARC Delivarable D2.1: Initial definition of use cases and carrier requirements," 2011. [Online]. Available: http://www.fp7-sparc.eu/projects/deliverables

[7] R. Sherwood, G. Gibb, K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Flowvisor: A network virtualization layer," *OpenFlow Switch Consortium, Tech. Rep*, 2009.

[8] P. Skoldstrom and K. Yedavalli, "Network virtualization and resource allocation in openflow-based wide area networks," in *Proceedings of SDN'12: Workshop on Software Defined Networks*. IEEE ICC, June 2012, to be published.

[9] N. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," *Communications Magazine, IEEE*, vol. 47, no. 7, pp. 20–26, 2009.

[10] ISO/IEC, "Iso/iec 10040: "information technology - open systems interconnection - systems management overview"," 1998.

[11] TMForum. (2012) Business process framework (etom) frameworx release 12. [Online]. Available: http://www.tmforum.org/FrameworxRelease12/13063/home.html

[12] ITU-T, "Itu-t recommendation m.3010: Principles for a telecommunications management network," 2001.

[13] 3GPP, "Tr 36.902 v1.0.1: Evolved universal terrestrial radio access network (e-utran); self configuring and self-optimzing network (son) use cases and solutions," 2008.

[14] Open Networking Foundation (ONF). (2012, April) Configuration & management working group. [Online]. Available: http://www.opennetworking.org/working-groups/config-a-mgmt

[15] ——, "Openflow configuration and management protocol of-config 1.0," 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/openflow/OF-Config1dot0-final.pdf

[16] M. Forzati, C. Larsen, and C. Mattsson, "Open access networks, the Swedish experience," in *Transparent Optical Networks (ICTON), 2010 12th International Conference on.* IEEE, 2010, pp. 1–4.