POSTER: Self-Healing Mechanisms for Software-Defined Networks

José Sánchez[12], Imen Grida Ben Yahia[1], Noël Crespi[2]
[1] Orange Labs, Paris, France
[2] Telecom SudParis, Evry, France

***Abstract*— Programmable networks brought by Software Defined Networks (SDN) are perceived by operators as cornerstone to reduce the time to deploy new services, to augment the flexibility and to adapt network resources to customer needs at runtime. However, despite the vulnerabilities identified due to the centralization of the intelligence on SDN, its research is more centered on forwarding traffic and reconfiguration issues, not considering to a great extent the fault management aspects of the control plane. The aim of this paper is to provide SDN with fault management capabilities by using autonomic principles like self-healing mechanisms. We propose a generic self-healing approach that relies on a Bayesian Networks for the diagnosis block and it is applied to a centralized SDN infrastructure to demonstrate its functioning in the presence of faults.**

Keywords*— autonomics; self-healing; SDN; fault management; Bayesian Networks;

I. MOTIVATION AND APPROACH

Today's networks are complex sets of heterogeneous and vendor-dependent equipment that use proprietary management applications. This complexity is characterized by the high rate of events occurred, the enormous amount of involved equipment, their short life-cycle and the continuously growing introduction of new technologies and services. Such diversity and complexity implies a huge cost and effort to deploy new services and manage all the equipment, what prevents Operators from improving network and service quality and reliability to satisfy the market needs and cope with the aggressive competitiveness of the industry. One symptom of this complexity is the overwhelming and unmanageable amount of alarms received by medium-sized operational teams, estimated in millions per day [1]. This urgently requires automation and intelligence to keep this management under control.

The arrival of new technologies such as Software Defined Networks (SDN), Network virtualization, Networks Function Virtualization (NFV)[2] and the Cloud, will suppose a transition from configurable to programmable networks, which is an enabler for the introduction of services and a reduction of costs, but, on the other hand, it emphasizes the need for autonomics.

SDN paves the way towards programmable networks, which would permit to change the network behavior by reprogramming the equipment in a flexible manner as response to dynamic changes. SDN has diverse definitions [2][3][15], but all of them propose a clear separation of the control plane from the data plane. SDN is characterized by:

-A programmable control of the network through open Application Programming Interfaces (APIs).

-The centralization of the intelligence in the control plane to provide fine-grained control of the network.

-The Service Abstraction Layer (SAL) hides the details of lower SDN layers.

However, this centralization of the intelligence on SDN jeopardizes the control plane, what has disastrous consequences on the data plane. Thus, the resiliency of the controller is paramount, because it orchestrates the data plane by installing the forwarding flows. Despite this importance, no so much research has been found that surveyed fault management for SDN. Nevertheless, there is some related work [5-12] to provide SDN with fault management capabilities, but counts on several limitations:

(i) this work is OpenFlow oriented, so the recovery is an OpenFlow dependent mechanism that reconfigures or provides alternative paths to the affected OpenFlow switches.

(ii) this work does not consider malfunctions in the control plane to a great extent.

(iii) there is a lack of an integrated fault management for legacy equipment and OpenFlow-based.

(iv) most of the recovery mechanisms are for in-band SDN architectures.

The introduction of autonomic principles on SDN guarantees an immediate detection and reparation of any malfunction. Autonomics [13] enables advanced automation and intelligence on current fault management approaches in order to perform them more efficiently and better. The autonomic properties are known as self-x, including self-healing. We define autonomic fault management based on self-healing systems. Self-healing systems recover the network from abnormal states and bring it back to a normal state.

A self-healing system is a closed-loop system composed of detection, diagnosis and recovery blocks. It is fed with alarms through its sensors to detect faults and runs this closed-loop to diagnose the root cause and applies the appropriate recovery strategy through the actuators. We consider that self-healing principles are mature enough and supported by many researches [14] that position them as the proper solution for enabling automation and intelligence on SDN architectures.

Our main research question is centered on bringing these autonomic principles to ensure the resiliency and robustness of centralized SDN out-of-band architectures, whose data plane not only is composed of OpenFlow switches but also of other legacy equipment (e.g. routers or APs).

II. ENVISAGED SELF-HEALING FRAMEWORK FOR SDN

We envision a generic self-healing module for SDN architectures (Fig. 1), capable of monitoring the services deployed over the control plane, the SDN controller, and its underlying network. Although the self-healing module can be placed at the points (1), (2), (3) or (4), we initially consider the (2), due to the fact that the self-healing module requires a continuous interaction with the controller to retrieve controller performance, network performance and the topology. The self-healing module also interacts with the plane (1) through the SM to be informed about the state of the services.

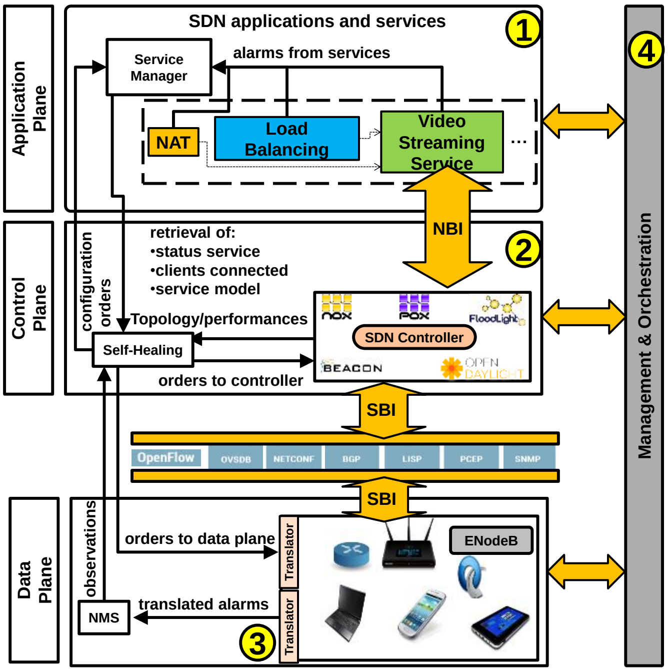

Fig. 1. Self-Healing proposed module for SDN

The detection block is ensured through the Network Management System (NMS) and the Service Manager (SM). The former takes the alarms from the network elements and the latter takes information from the services deployed. This information contains the status of the services, the list of clients connected to each service as well as the service model. We consider three different levels of alarms:

-Service-related alarms: concern malfunctions on the services of the application plane such as misconfigurations, errors or SLA violations.

-Transport-related alarms: concern malfunctions on the forwarding mechanisms of the control plane such as misconfiguration, errors or QoS violations like dropped packets.

-Physical-related alarms: concern physical failures on the equipment.

The diagnosis block is based on a Bayesian Networks algorithm that incorporates the observations coming from the translated alarms from the NMS and SM and adds them as evidences for diagnosing the root cause. This block uses the network topology information, given by the SDN controller, for building the Bayesian Network graph. Also, it receives the status of the services, given by the SM. Bayesian Networks is a model-based algorithm that models complex dependencies of the network under uncertainty. It is supported by many researches on network diagnosis [16] and counts on many free-licensed toolboxes [17][18][19].

The recovery block takes the diagnosis result coming from the diagnosis block and launches the appropriate recovery strategy that launches reconfiguration orders to the plane causing the malfunction (the controller, the SM, and the data plane) to re-establish the affected service.

As a first part of our work, we integrated our self-healing module in an in-band centralized SDN architecture, whose SDN controller was based on POX [20], under a video streaming service delivered through a fixed network topology emulated on Mininet [21]. The self-healing module can detect, diagnose and repair faults of different nature (e.g. physical failures, streaming services, OpenFlow crashes and traffic drops on any interface). If the streaming service is stopped abnormally, the self-healing detection block is notified, diagnoses the root cause and suggests the corresponding actions to re-establish the streaming service.

III. FUTURE WORK

Catering for resiliency capabilities to SDN is indeed a novel topic, but more concretely, this approach fills the gap between autonomics, self-healing in particular, and SDN. The inclusion of a Bayesian Network-based algorithm on SDN, empowers this architecture with the capability of detecting disruptions on the application plane, the control plane and the data plane at run-time.

We aim to extend our developed framework to monitor virtualized services like OpenIMSCore [22], OpenLTE [23] or OpenVoLTE [24] deployed over an SDN OpenDaylight [25] controller. The identified research lines to be followed in our current framework:

-Learning of topology: Extraction of the topology from the SDN controller and building of the dependency graph automatically from the topology.

-Accuracy on diagnosis: Reduction of the uncertainty of the diagnosis through a closed-loop that considers the effects of the proposed actions.

-Proactive framework: Inclusion of QoS measurements on both control and application planes to prevent future malfunctions by monitoring degradations.

-Recovery techniques: Inclusion of different techniques such as reconfiguration orders, swapping mechanisms for the controller, alternative forwarding for OpenFlow-enabled switches or load balancing on access points.

-Semantic translation for alarms: Implementation of translation mechanisms for the alarms emitted from the data plane that depend on different southbound protocols.

REFERENCES

[1] S. Wallin, “Rethinking Network Management: Models, Data-Mining and Self-Learning”, Ph.D dissertation, Dept. Comp. Sciencies, Luleå, Sweden, 2012.

[2] “White paper on ”Network Functions Virtualisation”. http://portal.etsi.org/NFV/NFV White Paper.pdf, 2012. [Online; accessed 18-Apr-2013].

[3] Open Networking Foundation white paper ‘Software-Defined Networking: The New Norm for Networks’. April 2012. Available at: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp sdn-newnorm.pdf (Accessed: January 2014).

[4] ITU-T Resolution 77, ‘Standardization work in the ITU Telecommunication Standardization Sector for software-defined networking’. November 2012. Available at: http://www.iut.int/pub/T-RES-T.77-2012 (Accessed: January 2014).

[5] Sejun Song; Sungmin Hong; Xinjie Guan; Baek-Young Choi; Changho Choi, "NEOD: Network Embedded On-line Disaster management framework for Software Defined Networking," Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on , vol., no., pp.492,498, 27-31 May 2013.

[6] Hyojoon Kim; Santos, J.R.; Turner, Y.; Schlansker, M.; Tourrilhes, J.; Feamster, N., "CORONET: Fault tolerance for Software Defined Networks", Network Protocols (ICNP), 2012 20th IEEE International.

[7] Fonseca, P.; Bennesby, R.; Mota, E.; Passito, A., "A replication component for resilient OpenFlow-based networking", Network Operations and Management Symposium (NOMS), 2012 IEEE, vol., no., pp.933,939, 16-20 April 2012. DOI: 10.1109/NOMS.2012.6212011.

[8] Hyojoon Kim; Feamster, N., "Improving network management with software defined networking," Communications Magazine, IEEE , vol.51, no.2, pp.114,119, February 2013

[9] Beheshti, N.; Ying Zhang, "Fast failover for control traffic in Software-defined Networks", Global Communications Conference (GLOBECOM), 2012 IEEE , vol., no., pp.2665,2670, 3-7 Dec. 2012.

[10] Kempf, J.; Bellagamba, E.; Kern, A.; Jocha, D.; Takacs, A.; Skoldstrom, P., "Scalable fault management for OpenFlow," Communications (ICC), 2012 IEEE International Conference on , vol., no., pp.6606,6610, 10-15 June 2012

[11] S. Sharma et al., Fast failure recovery for in-band OpenFlow networks, DRCN, 2013.

[12] M. Borokhovich and S. Schmid, How (Not) to Shoot in Your Foot with SDN Local Fast Failover, in Principles of Distributed Systems, R. Baldoni, N. Nisse, and M. van Steen, Eds. Springer International Publishing, 2013, pp. 6882.

[13] John O. Kephart and David M. Chess, "The Vision of Autonomic Computing," in IEEE Computer, Vol. 36, No. 1, pp. 41-50, 2003.

[14] H. Psaier, and S. Dustdar, “A survey on Self-Healing systems: approaches and systems”, Computing 91, 1 (January 2011), 43-73. DOI=10.1007/s00607-010-0107-y.

[15] ONF, “SDN Architecture Overview v1.0, Dec 2013. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf

[16] Steinder M, Sethi AS. A survey of fault localization techniques in computer networks. Science of Computer Programming 2004;53(2): 165–194. http://www.sciencedirect.com/science/article/pii/S0167642304000772 [27 August 2013].

[17] Kevin Murphy’s Bayesian Networks Toolbox, MIT AI lab,200 Technology Square,Cambridge. Available at: http://www.ai.mit.edu/˜ murphyk/Software/BNT/bnt.html

[18] GeNIe Toolbox, Decision Systems Laboratory, University of Pittsburgh, B212 SLIS Building, 135 North Bellefield Avenue, Pittsburgh, PA 15260, USA, http://www.sis.pitt.edu/˜ genie/.

[19] JavaBayes Software, Fabio Gagliardi Cozman, Escola Politcnica, University of So Paulo, http://www.pmr.poli.usp.br/ltd/Software/javabayes/ (recent versions).

[20] POX controller software. Available at: https://openflow.stanford.edu/

[21] B. Lantz, B. Heller and N. McKeown. A network in a laptop: Rapid prototyping for software-defined networks (at scale!). In Proc. HotNets, Oct. 2010.

[22] OpenIMSCore software. Available at: http://www.openimscore.org/

[23] OpenLTE software. Available at: http://sourceforge.net/projects/openlte/

[24] OpenVoLTE software. Available at: http://openvolte.com/solutions.php

[25] OpenDayLight project. Available at : http://www.opendaylight.org/ (Accessed January 2014).