Availability Modelling for SDN switch in Cloud based Infrastructure

May Thae Naing, Aye Myat Myat Paing
University of Information Technology, Yangon, Myanmar
maythae@uit.edu.mm, ayemyatmyatpaing@uit.edu.mm

Abstract

Attaining continuity and high availability of data transactions for cloud computing services are necessary for SDN architecture. The high-speed and complicated network of hosts and network devices often meet with a variety of failures due to links or system components. This failure affects the availability of the system. The proposed system uses a two-level availability model that is used to evaluate the availability of SDN concept in cloud based infrastructure. This paper offer availability solution for software defined network (SDN) in cloud computing Infrastructure and then describe the markov model for availability in SDN switches. Moreover, the impact of software and hardware failures on the overall availability of SDN switches is evaluated by SHARPE Tool.

Key Words- Cloud computing, Software Defined Networks, Availability

1. Introduction

Cloud computing has emerged as a widely accepted computing paradigm built around core concepts such as elimination of up-front investment, reduction of operational expenses, on-demand computing resources, elastic scaling, and establishing a pay-per-usage business model for information technology and computing services. There are different models of cloud computing that are offered today as services like Software as a Service (SaaS), Platform as a Service (PaaS), Network as a Service (NaaS) and Infrastructure as a Service (IaaS) [1]. In spite of all recent research and developments, cloud-computing technology is still evolving. Several remaining gaps and concerns are being addressed by alliances, industry, and standards bodies.

Software-Defined Networking (SDN) is an emerging networking paradigm that gives hope to change the limitations of current network infrastructures. First, it breaks the vertical integration by separating the network's control logic (the control plane) from the underlying routers and switches that forward the traffic (the data plane). Second, with the separation of the control and data planes, network switches become simple forwarding devices and the control logic is implemented in a logically centralized controller (or network operating system), simplifying policy enforcement and network (re)configuration and evolution [2], [3].

The key concept of the cloud computing is virtualization. Virtualization is the abstraction of the physical resources needed to complete a request and underlying hardware used to provide service. And also, the idea of the SDN is adopted from the concept of virtualization, where controls and managements of software subsystems are completely decoupled from hardware infrastructure. The decoupled components of the SDN are separated into three layers of the SDN architecture; (i) Data plane: SDN enabled network devices on a data plane reside at the bottom of the SDN architecture as the underlying physical layer, (ii) Control plane: network operating systems and hypervisors on the control plane resides at the middle layer to provide a bare virtualized environment; and (iii) Management plane: network applications running on the management plane resides at the upper-most layer. This virtualization approach brings three key attributes to the SDN: logically-centralized intelligence, programmability and high-level abstraction. Nevertheless, there are still many issues to use SDNs [4]. In fact, physically centralized network infrastructure still requires adequate levels of system availability and reliability.

High availability refers to the ability of a system to perform its function continuously (without interruption) for a significantly longer period of time than the reliabilities of its individual components would suggest. High availability is mostly often achieved through fault tolerance. Therefore, the effort in the proposed system will offer availability model by a comprehensive evaluation of the SDN in cloud infrastructure. To evaluate the model using SHARPE tool simulation is presented.

This paper organizes as follows: Section II describes the related work of the proposed system, Section III presents the two-level availability model, and Section IV describes the case study for the model. Finally, Section V concludes the paper.

2. Related Work

One of the main reasons of hesitating to adopt SDNs is the concern on availability. There are a few works on the availability of SDNs. In paper [5], the authors considered the impact of SDN application failures on the controller reliability. In paper [6], the authors proposed a stochastic model focusing only on the controller of a SDN rather than the whole SDN. In paper [7], the authors presented experimental results to improve the reliability and availability of core networks using SDN/Openflow. In paper [8], the authors proposed an approach to provide high-availability applications using a SDN. In this paper, hierarchical models will be presented for the availability of a SDN. In paper [9], the authors proposed a stochastic model focusing a stochastic availability model with the incorporation of hardware failures and software failures. They used RAID1 architecture for storage system.

In paper [10], the authors formalized a two-level availability model that is able to capture the global network connectivity without neglecting the essential details. It has highlighted the considerable impact of operational and management (O&M) failures on the overall availability of SDN. Moreover, its ‘results showed that the impact of software and hardware failures on the overall availability of SDN can be significantly reduced through proper over provisioning of the SDN controller(s). The paper [11] provided similar availability to the traditional IP backbone networks. It also used a two-level availability model which is able to capture the global network connectivity without neglecting the essential details and which includes a failure correlation assessment should be considered. It also presented the implementation on M¨obius of the Stochastic Activity Network (SAN) availability model of the network elements and the principal minimal-cut sets of a SDN backbone network and the corresponding traditional backbone network.

3. Two-Level Availability Model

A two-level hierarchical model [12] is introduced to evaluate the dependability of SDN in a global network. In this example, the dependability is measured in terms of steady state availability, in the following referred to as availability. The two-level hierarchical modelling approach consists of

- upper level: a structural model of the topology of network elements and controllers
- lower level: dynamic models (some) of network elements

The approach seeks to avoid the potential uncontrolled growth in model size, by compromising the need for modelling details and at the same time modelling a (very) large scale network. The detailed modelling is necessary to capture the dependencies that exist between network elements and to describe multiple failure modes that might be found in some of the network elements and in the controllers. The structural model disregards this and assumes independence between the components considered, where a component can be either a single network elements with one failure mode or a set of elements that are interdependent and/or experience several failure modes and an advanced recovery strategy. For the former we need to use dynamic models such as a Markov model or Stochastic Petrinet (e.g., Stochastic Reward Network [13]), and for the latter structural models such as reliability block diagram, fault trees, or structure functions based on minimal cut or path sets.

The objective of the modeling approach is to evaluate the availability of SDN.

4. Model Case Study

In this evaluation, this paper consider the network topology depicted in Figure 1. The service provider access the SDN controller through the northbound APIs. The service provider uses them for configuring the network resources and adding or removing tenants. The tenants use the northbound APIs to create subnets, to define policies. The controller controls the network switches and gateways through OpenFlow. We distinguish between the core network and the edge. The latter comprises the access switches to which end hosts are connected and the gateways that connect the service provider’s network to external networks.

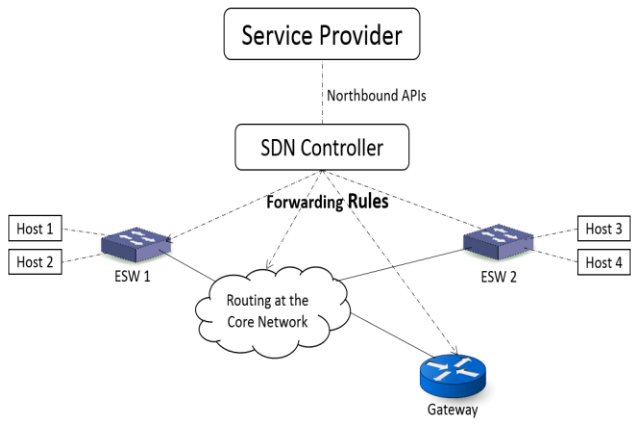

Figure 1. Topology for the model case study

5. Markov Model of the SDN switches

In the following, this paper present the markov model of the network element: SDN switches. Figure 2 shows the model of the switch in an SDN. The states related to the control hardware failures are not contained in this model, since all the control logic is located in the controller. In any case, we assume 1+1 redundancy of the SDN switch, which is a common best practice in any architecture. Multiple failures are not included in the model.

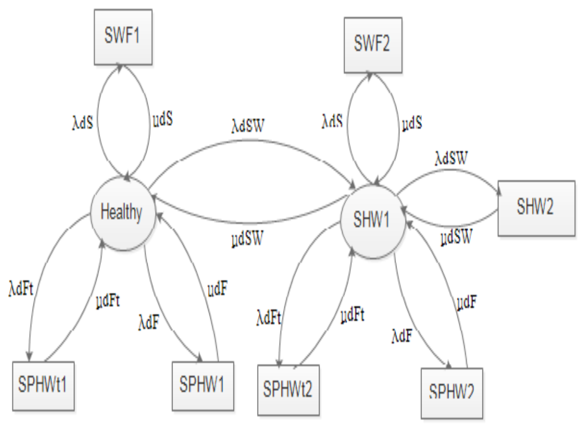

Figure 2. Markov model of SDN switches

Table 1. State Variables for SDN switch

State	Up/Down	Description
Healthy	up	System is fault free
SHW1	up	Hardware failure of one switch
SHW2	down	Hardware failure of both switches
FPHW1	up	Permanent hardware failure of one switch in forwarding plane
SPHW2	down	Permanent hardware failure of both switches in forwarding plane
FPHWt	down	Transient hardware failure in forwarding plane
SWF1	up	Software failure of one switch
SWF2	down	Software failure of both switches

Table 2. MODEL PARAMETERS USED IN THE CASE STUDIES

Intensity	Time	Description
1/λdF = 6	[months]	expected time to next permanent forwarding hardware failure
1/μdF = 12	[hours]	expected time to repair permanent forwarding hardware
1/λdFt = 1	[week]	expected time to next transient forwarding hardware failure
1/μdFt = 3	[minutes]	expected time to repair transient forwarding hardware
1/λdSW = 6	[months]	expected time to next control hardware failure
1/μdSW = 12	[hours]	expected time to repair control hardware
1/λdS = 1	[week]	expected time to next software failure
1/μdS = 3	[minutes]	expected time to software repair

Table 3. Model parameters for the SDN switch

Intensity	Description
λ F = λ dF	intensity of permanent hardware failures
μ F = μ dF	repair intensity of permanent hardware failures
λ Ft = λ dFt	intensity of transient hardware failures
μ Ft = μ dF	restoration intensity after transient hardware failures
λ sS = 0	intensity of software failure

All the model parameters are defined in Table II. In an SDN switch, the failure/repair intensities of (permanent/transient) hardware failures are the same because failures with the same cause and the same intensities. However, we assume that the software on an SDN switch will be much less complicated than on a traditional IP router because the control logic has been moved to the controllers.

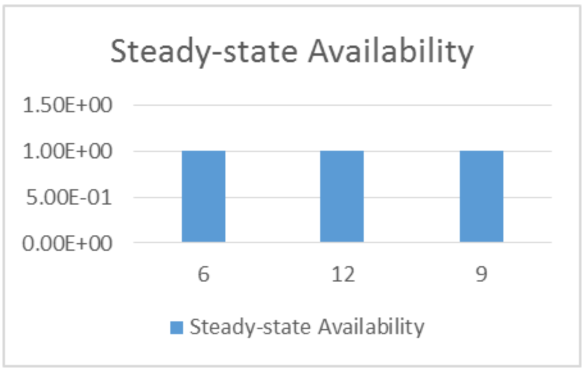

Figure 3. Evaluating result for the model case study

Figure 3 shows the evaluating result for the model case study: SDN switches for the proposed topology. The steady-state availability evaluates according to the expected time to repair permanent forwarding hardware. The evaluating results are the same availability although change the factor.

6. Conclusions

This paper offer availability solution for software defined network (SDN) in cloud computing Infrastructure. A two-level availability model that includes structural and dynamic models has been formalized and for the dynamic level Markov model of the single network elements have been proposed. The numerical analysis used to evaluate the availability model. In the future, this proposed system will evaluate through both analytical and simulation tool (SHARPE).

7. References

[1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," September 2011.

[2] N. Mckeown, "How SDN will Shape Networking," October 2011. [Online]. Available: http://www.youtube .com/ watch?v=c9-K5O qYgA.

[3] S. Schenker, "The Future of Networking, and the Past of Protocols," October 2011. [Online]. Available: http://www.youtube.com/watch?v= YHeyuD89n1Y.

[4] D. Kreutz, F. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," Proceedings of the IEEE, vol. 103, no. 1, pp. 14–76, Jan 2015.

[5] B. Chandrasekaran and T. Benson, "Tolerating SDN Application Failures with LegoSDN," in Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, ser. HotSDN '14. New York, NY, USA: ACM, 2014, pp. 235–236. [Online].

[6] F. Longo, S. Distefano, D. Bruneo, and M. Scarpa, "Dependability modeling of Software Defined Networking," Computer Networks, Apr. 2015.

[7] M. Tamura, T. Nakamura, T. Yamazaki, and Y. Moritani, "A study to Achieve High Reliability and Availability on Core Networks with Network Virtualization," Technical Report, vol. 15, no. 1, Jul. 2013.

[8] S. Dwarakanathan, L. Bass, and L. Zhu, "Application Level HA and QoS Using SDN," NICTA Technical Report, Tech. Rep., 2015.

[9] K. Han, T. A. Nguyen, D. Min, and E. M. Choi, "An Evaluation of Availability, Reliability and Power Consumption for a SDN Infrastructure Using Stochastic Reward Net," Advances in Computer Science and Ubiquitous Computing, vol. 421, 2016, pp 637-648.

[10] G. Nencioni, B. E. Helvik, A. J. Gonzalez, P. E. Heegaard, and A. Kami´nski, "Availability Modelling of Software-Defined Backbone Networks," submitted to the 2nd Workshop on Dependability Issues on SDN and NFV (DISN 2016).

[11] G. Nencioni, B. E. Helvik, P. E. Heegaard, "Implementing the Availability Model of a Software-Defined Backbone Network in M¨obius," submitted to the 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2017.

[12] P. E. Heegaard, B. E. Helvik, G. Nencioni, and J. W¨afler, "Managed dependability in interacting systems," in Principles of Performance and Reliability Modeling and Evaluation, L. Fiondella and A. Puliafito, Eds. Springer, 2016.

[13] G. Ciardo and K. S. Trivedi, "A decomposition approach for stochastic reward net models," Perf. Eval, vol. 18, pp. 37–59, 1993.