Analysis of Availability Model Based on Software Aging in SDN Controllers with Rejuvenation

Aye Myat Myat Paing
Faculty of Computer Science
University of Information Technology, Yangon
Yangon, Myanmar
ayemyatmyatpaing@uit.edu.mm

Abstract

Deficiency of flexibility and programmability of legacy network architecture has been the concern of many networking admirers for some years. Software defined networking (SDN) is a new emerging concept to logically centralize the network control plane in order to automate the configuration of individual network elements. However, the failures of SDN controller are every large impact on the network performance and availability. There are different failure modes in SDN controller outages such as hardware and software. Unplanned downtime is mostly caused by software failure due to software aging rather than hardware failure. The aging related faults have a huge effect on the availability for software components, SDN controllers. For that reason, the work presented in this paper offers the availability model for software aging of SDN controllers by applying software rejuvenation. A stochastic reward net (SRN) is proposed to evaluate the availability assessment of a cluster of SDN controllers. And then how software rejuvenation can improve the performance of SDN controllers is studied. To evaluate the availability of proposed model, mathematical analysis is performed.

Keywords: *Software Defined Networking, Availability, Software rejuvenation, SDN controller*

I. INTRODUCTION

Software defined networking brought a revolution in computer networks that is offered to manage the entire network to be more flexible and programmable [1]. The central approach for SDN is built on a single controller which can be managing of all the node in the infrastructure. However, it can be a single point of failure. Distribute SDN controllers face with all the above issue [2]. The availability of SDN controller is more important issue for the overall system availability and network performance. However, SDN controllers', software component replication mechanism is not a good solution to improve the availability, since the main issue of the failure is often shared among the replicas, for example a bug in a software code [3]. It can suffer unplanned downtime due to computer failure, network failure and software failure and so on. One of the causes of unplanned software outages are the software aging phenomenon due to the degradation of software. There are aging related faults, Mandelbugs, which imitate the gradual degradation of the system performance, because memory leaks, data corruption and accumulation of numeric errors, etc [4]. The most effective solution to handle software failure due to software aging is software rejuvenation. Since the preventative action can be done at optimal time interval, it reduces the cost of system downtime and gets higher availability compared to reactive recovery from failure.

Moreover, a cluster of SDN controllers' architecture is focused in order to improve high availability purposes for the network. ONOS, Open Network Operating System, is a newly released open-source SDN controller which is used to consider in this work.

And then availability model of SDN controller using stochastic reward net is illustrated in order to evaluate the availability of cluster SDN controllers. In order to solve the software failure, preventative maintenance (such as software rejuvenation) is applied for enhancing the performance of proposed model. To show the performance of the proposed method, analytic analysis is presented. To evaluate the models throughout both analytic analysis and then SHARPE tool [5] are considered.

The organization of this paper is as follows. Section 2 provides an overview of the related work. The proposed research approach for how to solve software aging of a cluster of SDN controllers is

presented in section 3. The proposed model for enhancing system availability follows in section 4. Finally, the conclusion is described in section 5.

II. Related Work

In this section, selected publications are reviewed which are related to this work.

The researchers [6] presented that SDN is developed to afford more effective configuration enhance performance and more flexibility for huge network designs

The researchers focused on the state-of-art ONOS controller, designed to scale to large networks, based on a cluster of self-coordinating controllers and concentrate on the inter-controller control traffic in [7]. Vizarreta et.al [4] presented Failure Dynamics in SDN controllers' model and evaluates the impact that different controller failure modes have on its availability. In case study, they showed how the proposed model can be used to estimate the controller steady state availability, quantify the impact of different failure modes on controller outages, as well as the effects of software aging, and impact of software reliability growth on the transient behavior.

The authors [1] presented Software Defined Networking (SDN) brought an unprecedented flexibility and programmability into computer networks. And then they evaluated the benefits by enhancing the ONOS SDN-IP application with an adaptive Robust Traffic Engineering Algorithm.

The authors [8] describe that many software failures are those due to software aging phenomena. So, they proposed a new framework for predicting in real time the time-until-crash of web applications which suffer from software aging using machine learning techniques.

Ros et. al [9] have shown that in order to achieve the availability of five nines, the forwarding devices are required to connect at least two controllers, for all wide area network include in their study. Studies [9] and [4] separate between permanent and transient hardware and software failures.

In the following section, an availability model for software aging of SDN controllers is proposed using stochastic reward nets model and analyze the availability for cluster of SDN controllers in case of software aging failure.

III. Proposed research approach for Software aging in SDN controllers

With the aim of making networks not only more programmable but also easier to manage, SDN based network architecture has been presented. This system consists of a cluster of ONOS controllers and then the exchange of routing traffic among the controllers as shown in Figure 1. In this SDN based network scenario, the configuration, ONOS 1.2 [10, 11] controllers are considered. It is based on all in one SDN hub VM. To imitate the network controlled OpenFlow protocols is used.

Each network device can connect to multiple ONOS controllers, but each domain has its primary controller with full control for forwarding tables and so on. Each controller interacts with all the other controllers and then they send and accept keep-alive messages among controllers for monitoring in a cluster member. If one controller fails, the other controller in a cluster can take the operation from the affected controller. Controller failure frustrates the network ability to serve new requests coming from the network application. Moreover, after node or link failure, it needs to be routing or rerouting for the physical network. There are different failure modes in controller occur with different occurrence. In this work, node failure cause of software aging failure is considered on SDN controllers. Therefore, in order to detect and solve the software aging in case of memory exhaustion in SDN controllers, management server which includes aging detector and rejuvenation manager is considered. Accordingly, in this model two kinds of preventive maintenance (software rejuvenation methodology) have been considered and evaluated.

Taking the rejuvenation action, there is a downtime. For this reason, the rejuvenation action can be done at optimal times in order to reduce the cost of system downtime. The main process of aging detector and how rejuvenation policy used to solve the aging problem are discussed in subsection A and B.

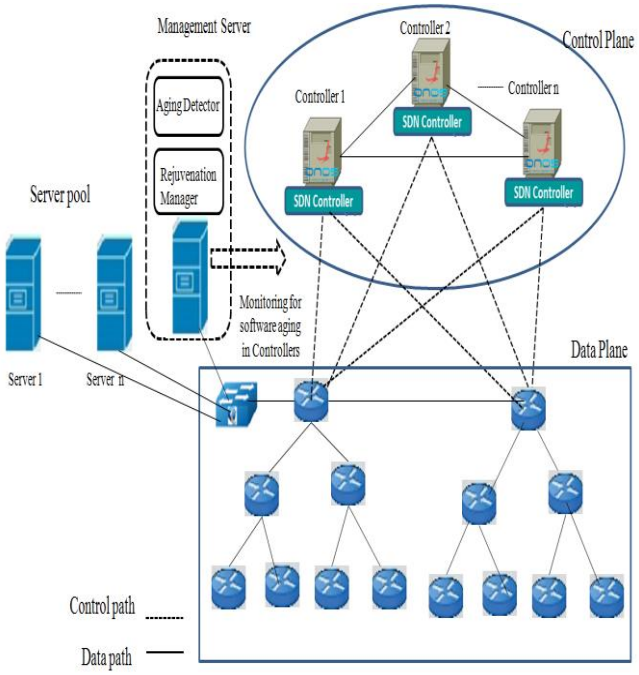

Figure 1. System Architecture

A. Aging Detector

In the management server, the Aging Detector module make the detection of software aging. The responsibility of the aging detector, it can detect the resource exhaustion for memory due to the amount of traffic exchange among a cluster of controllers and updating their states.

SDN controller's failures related to the software aging are detected from both configuration and network. In this work, Mandelbugs in SDN controller is considered. So, the Aging Detector is collecting the resource status Memory, and number of threads or number of connections among the SDN controllers. Because of these parameters can estimate the service time to crash due to software aging. When the aging detector detects resource exhaustion, the proposed rejuvenation policies are applied. Some potential failure happens in SDN controller, the rejuvenation manager that is software component in management server triggers the partial and full rejuvenation according to the failure and estimation of time to crash that detected form Aging Detector.

The work of aging detector and rejuvenation manager is shown in the sequence diagram as figure 2.

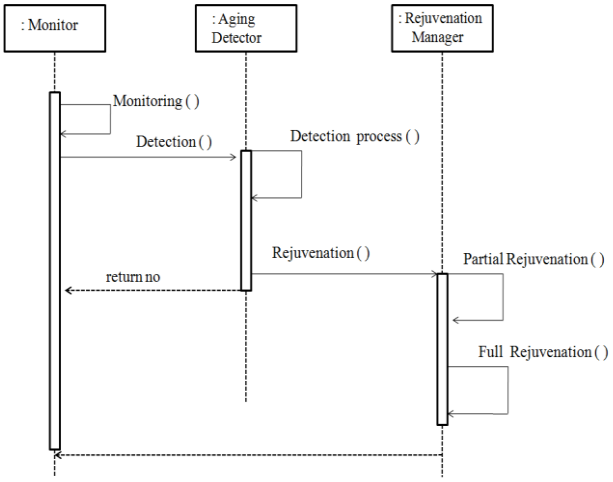

Figure 2. Sequence Diagram for Aging Detector and Rejuvenation Manager

B. Rejuvenation Policy

When Aging Detector detects software aging happens in each SDN controller, the rejuvenation manager will activate the rejuvenation action. The two kinds of rejuvenation policies will be performed on that system. It is based on the condition of the unstable state a minimal maintenance (a partial system clean and restart) or a major maintenance with rejuvenation (clean and restart) is applied.

According to the collecting the resource from Aging Detector, it can be detected some condition could not work well. At that time minimum maintenance is conducted. It is called partial restart and can restore its not working or out of order service such as configuration or Database connections and some other resources back to a healthy state. During the minimal maintenance, controller can provide continued services because this maintenance is a partial system cleanup.

Then, Aging Detector estimate the time to crash of controllers that have violated the threshold (Time Limit) defined by the system administrators per each service or per the whole framework, the rejuvenation manager triggers the major maintenance that is full rejuvenation. So, OS and all services on that controller must be stopped and the rejuvenation action restarts the OS to recover all its free memory. This rejuvenation is also called full restart. When one of the controllers in cluster needs to do major maintenance with rejuvenation, another SDN controller in cluster is to take the responsible for that rejuvenation triggered controller.

When software rejuvenation is performed the system stops serving resulting in a downtime that can be affected the operational costs. But due to its proactive feature, the downtime due to rejuvenation is less than downtime due to an unscheduled software failure. Furthermore, partial rejuvenation can be less downtime cost than full rejuvenation. Therefore, two kinds of rejuvenation (preventative maintenance) are considered in order to reduce system downtime in case of software aging.

IV. STICHASTIC REWARD NETS (SRN) MODEL FOR PROPOSED SYSTEM

A stochastic reward nets (SRN) model with two kinds of rejuvenation mechanism for a cluster of SDN controllers is described in Figure 3. It has n tokens which represented n SDN controllers in cluster. In initial condition, all controllers are working well state, indicated by a token in place P_{Up}. When transition T_I fires the controller enters the inspection state and a token moves from P_{Up} to P_I. After inspection is complete (firing the transition T_{up},) no action is taken if the system is found to be in working state. Transition $T_{U,I}$ models the unstable state of the controller. When this transition fires, (i.e., a token reaches place $P_{U,I}$) the controller is operational but in the unstable state. The transition $T_{U,I}$ models the unstable state and under inspection state of the SDN controller. At that time, some service in SDN controller is not working well but it is still running. So, it needs to a partial system clean and restart for SDN controller. After minimal maintenance is complete (firing the transition T_m), the controller enters the operational state.

As the time progress, each controller can eventually transit to software aging state in place P_{FP} detected from the aging detector as memory leaks which reproduces the gradual degradation of the system performance through the transition T_{Aging}. If the one of the SDN controller is about to be major maintenance with rejuvenated, the traffic control and the routing decision of the packet on affected controller is switched to another controller in cluster and then the effected controller will be started for the new requests and sessions before rejuvenation through the transition T_{sw}. It can return to the original SDN after the accomplishment of the rejuvenation such as system clean and restart.

Consequently, the rejuvenation interval is defined through clock with guard function ginterval. The tokens are in the place P_{clock} and P_{FP}. If there is a token in $P_{trigger}$ through guard gtrig, and there are controllers to be rejuvenated. In that state token is placed in P_{FP}, immediate transition t_{rej} is enable through the function grej as shown in Table 2. After the controller has been rejuvenated, it will be in the working state through transition T_{H_rej}. And then, immediate transition is enabled by using greset and a token is switched to P_{clock}.

If the software aging failure cannot be detected, the controller can fail because of software aging and the controller is not working in the place of P_{Aging}. At that time, there is no active controller in a cluster, the system may outage and then the system goes to Down state in the place of P_{Down} through the transition T_{Hfail}. From a whole system down, the system can be restoration with the transition T_{Repair}. If one SDN controller is used in system architecture, the downtime can be more because it does not switch to routing decision and some other services to another active controller. Otherwise, there is an active controller in a cluster; the controller can be replaced through the transition $T_{Replace}$. After repairing the fail controller through T_{Repair}, the controller can join in the cluster.

The Markov reward model which is mapped from proposed SRN constructs a marking dependency and enabling functions (or guards) facilitate the construction of model in order to solve the steady state availability for proposed model. Then, the SRN model has been evaluated through numerical derivation that can be shown how rejuvenation mechanism can reduce downtime and the architecture for a cluster of SDN controllers can improve the system availability with the mapping of reachability graph in sub section A

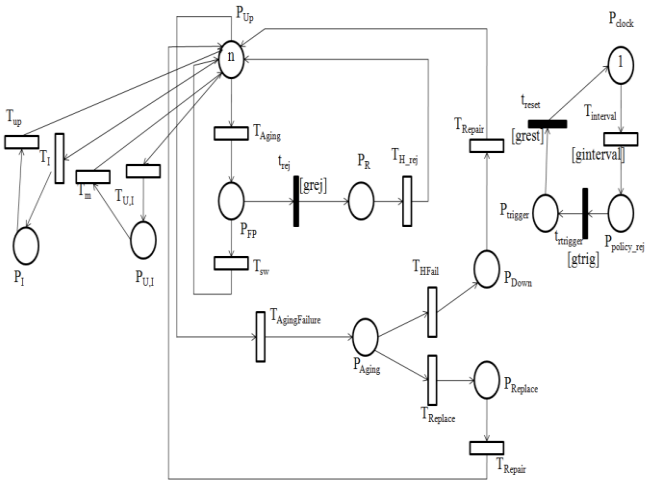

Figure 3. Stochastic Reward Nets Model for Software Aging in SDN controllers

TABLE I. DESCRIPTION FOR PLACES OF PROPOSED MODEL

Places	Description
P_{Up}	A cluster of SDN controllers Up state
P_I	Inspection state in SDN controllers
$P_{U,I}$	SDN Controllers unstable state under inspection
P_{FP}	Software Aging state in SDN controller
P_R	Proactive Software Rejuvenation state in SDN controller
P_{Aging}	SDN controller Software aging failure state
$P_{Replace}$	SDN controller replaced state
P_{Down}	Failure state of all SDN controller in cluster
P_{clock}	Rejuvenation interval state
P_{policy_rej}	Rejuvenation policy state
$P_{trigger}$	Rejuvenation state

A. Reachability Analysis

In this section, the reachability graph is constructed for the mapping of proposed model.

Let 7 tuples (π_{Up}, $\pi_{I\,i}$, $\pi_{U,Ii}$, $\pi_{R\,i}$, π_{Aging}, π_{Down}, and $\pi_{Replace\,i}$) denote the marking with π_x =1, if a token is presented in place $\pi_{x,}$ and zero otherwiseas shown in figure 4.

This figure illustrates with squares representing the markings and arcs representing possible transition between the markings. Let λ_I, $\lambda_{U,I}$, μ_{up}, μ_m, λ_R, μ_{rej}, $\lambda_{sw\text{-}age}$, $\mu_{replace}$, $\lambda_{fail}$and μ_{repair}be the transition rates associated with T_I, $T_{U,I}$,T_{up}, T_m, T_R, T_{H_rej}, $T_{AgingFailure}$, $T_{Replace}$, T_{Hfail}, and T_{repair}respectively. By mapping through actions of this graph with stochastic process, steady-state solution can be achieved.

TABLE II. GUARD FUNCTION FOR PROPOSED MODEL

Name	:	Guard Function
ginterval	:	(#(P_{FP}) = =1)
grej	:	(#($P_{trigger}$) = =1)
greset	:	(#(P_R) = =1)
gtrig	:	(#(P_{policy_rej}) = =1)

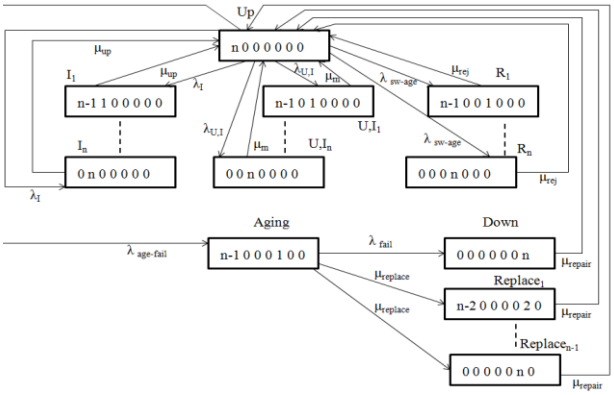

Figure 4. Reachability Graph for the proposed model

The summing the probabilities of all states in proposed system equation is described in Equation 1.

$$\pi_I + \pi_{U,I} + \pi_R + \pi_{Aging} + \pi_{Down} + \pi_{Replace} + \pi_{Up} = 1 \quad (1)$$

The combination of balance equation that obtained from figure 4 with the above the summing of probabilities, the closed form solution can be achieved.

$$\pi_I = \sum_{i=1}^{n} {}^{\lambda_I}/_{\mu_{up}}\, \pi_{Up} \quad (2)$$

$$\pi_{U,I} = \sum_{i=1}^{n} {}^{\lambda_{U,I}}/_{\mu_m}\, \pi_{Up} \quad (3)$$

$$\pi_R = \sum_{i=1}^{n} {}^{\lambda_{sw-age}}/_{\mu_{rej}}\, \pi_{Up} \quad (4)$$

$$\pi_{Aging} = A\pi_{Up} \quad (5)$$

$$\pi_{Down} = {}^{\lambda_{fail}}/_{\mu_{repair}}\, A\pi_{Up} \quad (6)$$

$$\pi_{Replace} = \sum_{i=1}^{n-1} {}^{\mu_{replace}}/_{\mu_{repair}}\, A\pi_{Up} \quad (7)$$

$$\pi_{Up} = \left\{ 1 + \left[\begin{array}{l} \sum_{i=1}^{n} \lambda_I / \mu_{up} + \\ \sum_{i=1}^{n} \lambda_{U,I} / \mu_m + \\ \sum_{i=1}^{n} \lambda_{sw-age} / \mu_{rej} + \\ A + \\ \lambda_{fail} / \mu_{repair} A + \\ \sum_{i=1}^{n-1} \mu_{replace} / \mu_{repair} A \end{array} \Big/ \begin{array}{l} \sum_{i=1}^{n} \lambda_I / \mu_{up} \\ + \sum_{i=1}^{n} \lambda_{U,I} / \mu_m \\ + \lambda_{sw-age} \\ + \lambda_{age\_fail} \end{array} \right] \right\}^{-1} \quad (8)$$

$$\text{Where } A = \lambda_{age\_fail} / (\lambda_{fail} + \mu_{replace})$$

TABLE III. THE MEANING OF THE PROBABILITY'S ARE AS FOLLOWS.

π_{Up}	:	The probability of SDN controllers are working state
π_{I_i}	:	The probability of SDN controller is under inspection state
π_{U,I_i}	:	The probability of SDN controller is not working well state
π_{R_i}	:	The probability of SDN controller is Rejuvenation state
π_{Aging}	:	The probability of SDN controller is software aging failure state
π_{Down}	:	The probability of SDN controllers are Down state
$\pi_{Replace_i}$	:	The probability of SDN controller is replace state

B. Analysis of Availability and Downtime

In this subsection, according to the numerical derivation, availability and downtime are evaluated. Availability is a probability of SDN concept-based network infrastructure which provides the services in each instant time through reachability graph. In this model, SDN controller services have been focused. Based on the proposed model, the whole system may be totally down when all controllers failed because of software aging failure and there is no hardware extra for replace SDN based network. The system is not available in Down state, there is a token in the place (P_{Down}). Downtime is the expected total downtime of a cluster of SDN controllers-based network in an interval of T (24*30*12 days). The system availability in the steady-state and Downtime are defined as follows:

$$Availability = 1\text{-}Unavailability \quad (9)$$

$$Availability = 1\text{-}\pi_{Down} \quad (10)$$

$$Downtime\ (T) = T^{*}\ \pi_{Down} \quad (11)$$

The applicability of the proposed model and solution methodology through numerical examples are illustrated using the transition firing rate. A good estimate value for a range of the model is assumed because the exact transition firing rates are not well known normally. So, experiments are performed using the following failure profile in literature [9, 4, and 12] mentioned in Table 4.

TABLE IV. PARAMETERS VALUES AND DESCRIPTION

Transition	Description	Values
T_I, $T_{U,I}$	Inspection of Aging Probably	1time/a day
T_{up}	After Inspection without maintenance Rate	0.3
T_m	After Inspection with maintenance Rate	0.6
T_{Aging}	Software Aging Rate	1 day
$T_{AgingFailure}$	Failure Rate that effect of aging	7 days
T_{sw}	Switch over rate	5 secs
T_{HFail}	Hardware/Physical host failure rate	6 months
T_{Repair}	SDN Controller Physically Repair Time	24 hrs
$T_{Replace}$	Replacement Time for Extra SDN Controller	2 hrs
$T_{interval}$	Rejuvenation interval	2times/a month
T_{H_rej}	Rejuvenation rate	2 mins
T	Unit Time Interval	24*30*12 days

The influence of number of SDN controller along with different failure rate caused of software aging on availability is shown in Figure 5. It can be seen the result that the need of at least two or three SDN controllers for achieving "five-nines" availability. So, the availability is dependent on the number of SDN controllers. According to the analysis,

the higher mean time to software aging failure of SDN controller, the higher availability can be obtained.

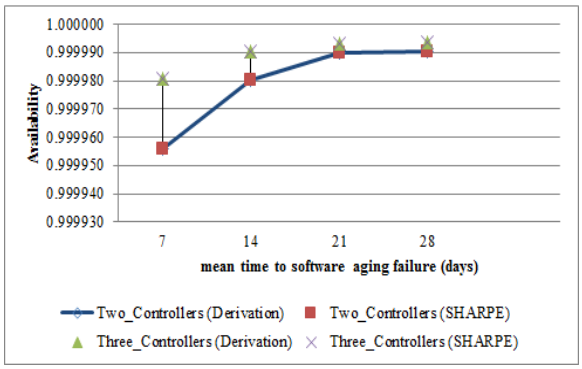

Figure 5. Availability vs different software aging failure time for multiple SDN controllers

The Figure 6 shows the differences in downtime with different number of SDN controller in cluster through different controller software aging failure time. The more controllers, the lower downtime can be obtained. As a result, the downtime decreases up to 5 minutes for three SDN controllers. According to the analysis, the consideration of a cluster of multiple controllers can be affected on system downtime. Therefore, cluster of SDN controller's architecture are applied in this work.

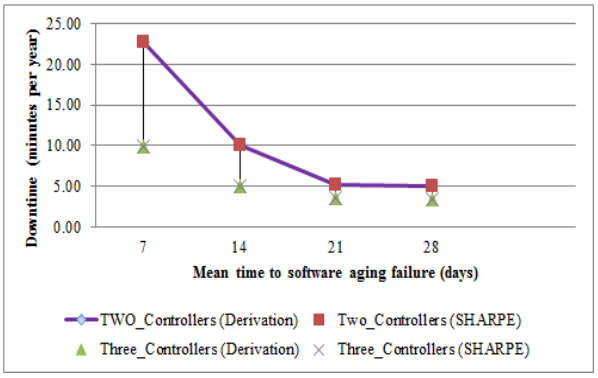

Figure 6. Downtime vs different software aging failure time for multiple SDN controllers

In our proposed research approach, two kinds of proactive rejuvenation are used for counteracting the software aging affected on SDN controller. With the purpose of it, the inspection for probably of software aging in SDN controller is very important because it can be noticed that the rejuvenation action applies or not. So, the influence of mean time between inspections (MTBI) along with different controller of software aging failure is evaluated through the proposed model. The analysis result of availability using above mentioned system-parameters is shown in Figure 7. There are several different values of time to carry out the inspection. The MTBI are assumed 6 hours and 12 hours. The lower MTBI, the higher availability can be reached.

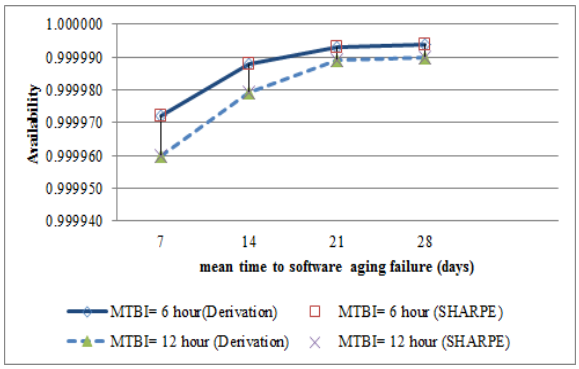

Figure 7. Availability vs different software aging failure time with different Inspection time for aging

The Figure 8 shows the differences in downtime with different software aging failure using various MTBI. From the result, the lower MTBI for SDN controller, the lower downtime can be obtained. For the best and most expensive of all SLA services, the downtime will be up to five minutes per year and application uptime would also be about 99.999% of the time. According to the downtime analysis, the inspection time for probably of software aging is important in SDN controller.

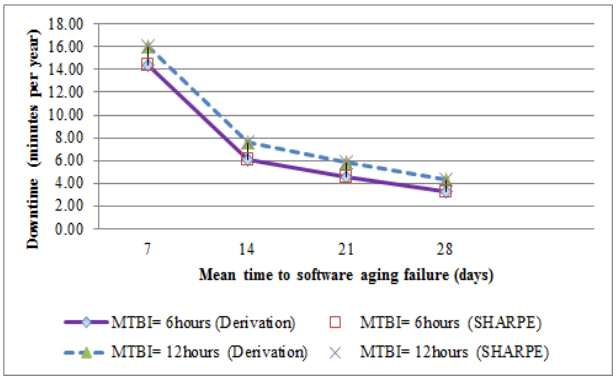

Figure 8. Downtime vs different software aging failure time with different Inspection time for aging

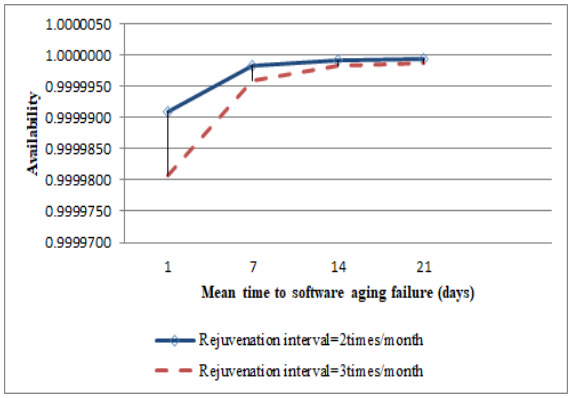

Figure 9. Availability vs different software aging failure time with different Rejuvenation interval

The Figure 9 shows the differences in availability with different software aging failure between once a week starting one day using various rejuvenation interval. From the result, if the rejuvenation did many times, the lower availability will be obtained. According to the analysis results, the rejuvenation interval should be optimal time interval and it is probably important for different software aging failure in SDN controller.

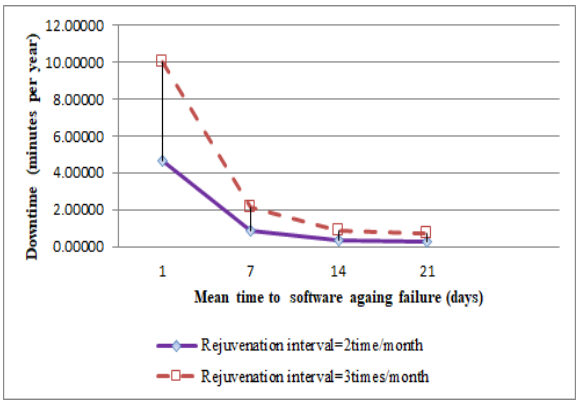

Figure 10. Downtime vs different software aging failure time with different Rejuvenation interval

The differences in downtime with different software aging failure using various rejuvenation interval is shown in figure 10. According to the analysis results, the rejuvenation interval should be optimum interval because the more rejuvenation, the lower downtime can be achieved especially software aging failure in a week.

According to the analysis results shown in figures, the failure rate caused of aging has also an impact on the numerical analysis of availability and downtime of the proposed model. However, aging failure rate is not accurate parameters, because it depends on the many factors such as controller utilization rate such as CPU, memory and so on. In our studies, the cause of software aging is based on memory exhaustion because SDN controller responsible that the control plan of forwarding networking devices is extracted and moved to the entity. Therefore, the varying aging failure rates are considered to evaluate the proposed model. The software aging caused by other factors like CPU utilization can be considered with different software rejuvenation interval in future.

According to the figures, it is shown that the derivation results and SHARPE [13] simulation results are the almost same. Therefore, it can be proved that the numerical derivation results for proposed model are acceptable.

V. CONCLUSION

This research work is intended for SDN based network infrastructure meant for improving the network performance. The objective is specially associated with how to solve software aging failure in SDN controllers which can be impact on network performance. Based on the numerical analysis using failure profile, a stochastic reward nets model for software aging of SDN controllers is evaluated in available and downtime. As the obtained results shown in figures, it can be said this approach get the high availability and lower downtime (uptime about 99.999% of the time) when the number of SDN controllers increased.

Two kinds of rejuvenations can also enhance the availability of SDN based network with optimal rejuvenation interval that can reduce the downtime of the system as shown in results.

Although the (n) numbers of controllers in a cluster are deployed in the proposed model, evaluation result show that multiple SDN controllers, at least two or three controllers offer the high availability of the services and in order to achieve minimizes downtime than traditional network. For future work, how software aging failures influences the impact on SDN based environment in other SDN controller instead of ONOS.

REFERENCES

[1] D. Sanvito, D. Moro, M. Gulli, I. Filippini, A. Capone, A. Campanella, ONOS Intent Monitor and Reroute Service: enabling plug & play routing logic, IEEE International Conference on Network Softwarization 2018, NetSoft 2018, Montreal, Canada.

[2] A. Panday, C. Scotty, A. Ghodsiy, T. Koponen, and S. Shenker, “CAP for networks,” in HotSDN. ACM, 2013, pp. 91–96.

[3] T. A. Nguyen, T. Ecom, S. An, J. S. Park, J. B., Hong and D.S. Kim, Availability modeling and analysis for software defined networks, Dependable Computing (PRDC) 2015 IEEE 21st Pacific Rim International Symposium on IEEE, 2015, pp. 159-168.

[4] P. Vizarreta, P. Heegaard, B. Helvik, W. Kellerer and C. M. Machuca, Characterization of Failure Dynamics in SDN Controllers, 9th International Workshop on Resilient Networks Design and Modeling (RNDM), 2017.

[5] K. S. Trivedi, SHARPE 2002: Symbolic Hierarchical Automated Reliability and Performance Evaluator. In Proc. Int. Conference on Dependable Systems and Networks, 2002, pp. 544.

[6] T. E. Ali, A. H. Morad, M. A. Abdala, Load Balance in Data Center SDN Networks, International Journal of Electrical and Computer Engineering (IJECE), Vol. 8, No.5, 2018, pp. 3086-3092.

[7] A. S. Muqaddas, A. Bianco, P.Giaccone, G. Maier, Inter- controller Traffic in ONOS clusters for SDN networks, 2016 IEEE International Conference on Communications (ICC).

[8] J. Alonso, J. Torres, J. Ll. Berral and R. Gavalda, Adaptive on-line software aging prediction based on Machine Learning, 2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN).

[9] F. J. Ros and P. M. Ruiz, Five nines of southbound reliability in software-defined networks, Proceedings of the third workshop on Hot topics in software defined networking. ACM, 2014, pp. 31-36.

[10] Distributed ONOS Tutorial ONOS Wiki, Created by Ali "The Bomb" AlShabibi, last modified by Jonathan Hart on Apr 03, 2015.

[11] ON.Lab, ONOS: Open Network Operating System, http://onosproject.orj/, 2017.

[12] Software Rejuvenation. Department of Electrical and Computer Engineering, Duke University, http://www.software-rejuvenation.com.

[13] J. Saisagar, K. D. Prashant, SDN Enabled Packet Based Load-Balancing (Plb) Technique in Data Center Networks‟ Department Of Computer Science Engineering, Srm University, Vol. 12, No. 16, August 2017.