Contents lists available at Mapta Publishing Group

Vol. 3, No. 1, January, 2021

Mapta Journal of Electrical and Computer Engineering (MJECE)

journal homepage: http://maptapublishing.com/index.php/mjece

ISSN: 2631-5718

Applying a Modified Triple Modular Redundancy Mechanism to Enhance the Reliability in Software-Defined Network

Baharak HassanVandi[a], Reza Kurdi[a, *], Mohammad Trik[b]

[a] *Department of Computer Engineering, Khorramabad Branch, Islamic Azad University, Khorramabad, Iran*
[b] *Department of Computer Engineering, North Tehran Branch, Islamic Azad University, Tehran, Iran*

Article Info

Article history:

Received Dec 20th, 2020
Revised Jan 28th, 2021
Accepted Feb 24th, 2021

Keyword:

Reliability
Performance Improvement
Triple Modular Redundancy (TMR) Mechanism
Fault Tolerance
Software-Defined Network (SDN)

Abstract

Software-Defined Networks are a new architecture of computer networks where network intelligence is logically focused on base software controllers, and network hardware becomes a simple device that is programmable via an interface. Indeed, software-defined networks bring new features such as routing, measurement, and monitoring through providing a global vision of the network. One of the major challenges in these software-defined networks is the discussion of fault tolerance and reliability because if the control unit fails in these networks for any reason, the entire network will impair. Therefore, to maintain reliability in this study, the redundancy feature, according to the Triple Modular Redundancy mechanism was used in the control unit. Accordingly, instead of a single control unit, several control units have been used as plugin to increase reliability in software-defined networks in addition to fault tolerance. Finally, through a series of experiments to evaluate the performance improvement at the error time, it was shown that multi-control topology, compared to single-control topology, could stop 18% of lost packets between hosts and also improve 4% of packets discarded by the switch on average.

1. Introduction

Software-Defined Networks (SDN) are a new architecture and approach for greater flexibility and controllability of networks and the capacity building to use a variety of applications, services, and software services on existing networks. These networks reduce the significant task and role of hardware and network equipment and add to the task and role of software layers and make management and monitoring of the network easier. In SDN, there is a centralized logic controller that the network switches are directly controlled by management functions. For this purpose, APIs such as Open Flow is used in Standard form [1]. The SDN controller should be able to intelligently perform the configuration and check the validity of the network topology to prevent manual errors and enhance the network availability. However, this intelligence may be due to the issue of dividing the data and control section causing the existence of a controller considered as a point of failure. In the architecture of the centralized controller and the absence of a stand-by controller, only a centralized controller is responsible for the entire network. If this controller fails, it may cause the disabling the network. To increase reliability, instead of a single control unit, we used several control unit samples as an extension (plugin) to increase the efficiency and lifetime of the network. Indeed, increasing the reliability of the operation of computer systems by implementing fault tolerance is possible. Fault tolerance in a digital system is obtained through redundancy in hardware, software, information, or calculations. Such redundancy can be implemented in static, dynamic, or hybrid configuration. Hardware redundancy is obtained by providing two or more physical samples of a hardware component. For example, a system can include additional processors, memory, or buses. In software-defined networks, there are different methods to achieve fault tolerance. The most common of these approaches is a certain amount of redundancy. Redundancy is the prediction of operational capabilities. Two types of space redundancy and time redundancy are possible [2]. Space redundancy provides components, functions, or additional data items that are not

* Corresponding author: rezakourdy@gmail.com

needed for a flawless performance. Space redundancy is itself classified as redundancy of hardware, software, and information that depends on the type of plug-in sources added to the system. In the time redundancy, calculations or data transfers are repeated and the result is compared with the result of the previously saved copy. The structure of this article is in a way that an overview of the work conducted in the architecture of software-defined networking is presented in Section 2. Then, in Section 3, we present the TMR mechanism to improve reliability in the network control unit. In Section 4, we propose a new architecture in the network control unit. Finally, simulation using MiniNet software and analysis of the results in Section 5 are presented.

2. Related activities

In [3], a model to modify the controller model is suggested. For example, single row request processing in the controller is divided into k rows, and each row corresponds to a single switch. Then, the controller prepares these rows in a scheduled order. The proposed model here limits the effect of DDoS attacks by applying a delay in demands. In [4], a distributed SDN controller is provided that is compatible with Open Flow. This article represents the critical applications of specific Open Flow mechanisms, especially Open Flow's unique selective and administrative functions. As architecture can be integrated with different controllers, it can change. Also, during the existence of available controllers, the architecture can change its size based on the requests available in the network. This, in conjunction with Kandoo and other distributed controllers, can help decrease certain classes of attacks, such as DoS attacks, against the control scheme. However, the purpose of this technology and other distributed controllers is to increase the fault tolerance and flexibility of the control scheme rather than using security issues. In a study carried out in [5, 6], another distributed controller compatible with Open Flow has been introduced. Authors have introduced Kandoo as a system for controllers that makes a responding hierarchical infrastructure only with first-level hierarchical controllers that are responsible for the entire network. The controllers at a lower level of the hierarchy are solely responsible for local decisions and delay any decision that is outside their sphere so that the data scheme sends all decisions to the control scheme. The system is also applicable to defend against certain attacks to the control scheme, but the level of attacks and their responses from those in [7, 8] are different. It should be noted that security was not the main objective of this research. In [7], by inspecting packets logging into the network, they suggested DAMASK architecture to reduce DDoS attacks in the cloud environment and SDN architecture. They tested SDN-based architecture in Amazon's EC2 cloud environment and provided the results related to performance in the form of a report. The simulation was performed using Mininet in the flexible Amazon cloud environment. The main issue concerning this scheme is the bottlenecks caused by checking any current against a list of attacks before sending that converts the network layer into an IPS.

3. The proposed architecture

Given that one of the major challenges in SDN is the failure of the central controller, so the architecture of distributed controllers is used. However, a Logically Centralized Controller point will cause a Single Point of Failure, and therefore necessary security measures need to be considered to increase accessibility and fault tolerance. The building of TMR is a fault-tolerant architecture based on three identical modules performing exactly the same function. Their inputs receive the same data that is quite close to each other, and their outputs feed a majority vote circuit. Thus, the TMR architecture reduces the probability of errors in the system's initial outputs. A defective module releases a false and incorrect value that can be hidden by two other error modules. In the simplest structure of the TMR, the voter is a drawback. If a problem arises in the voter, then the TMR structure may malfunction. To avoid this kind of problem, the voter can be identified by software or stronger design techniques. The most common form of inactive hardware redundancy is TMR. Its basic configuration is represented in Figure 1 [9].

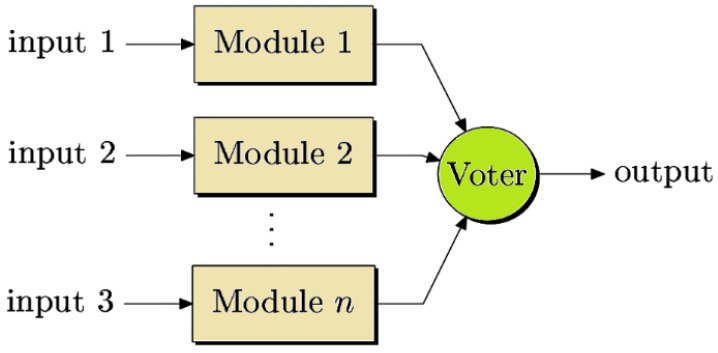

Figure 1. Triple Module Redundancy

The components are tripled to perform the same calculations in parallel. The voter is employed to determine the correct result. If one of the modules fails, the majority voter will hide the defect by the results of the two modules without another error. The TMR system can only hide one module defect. Failure in the remaining modules will cause the voter to produce the wrong result. A TMR system can function properly as long as the other two modules work correctly. Assume that the voter is complete and component failures are mutually independent, the reliability of a TMR system is calculated by Formula 1 [10].

$$R_{TMR} = R_1R_2R_3+(1-R_1)R_2R_3+R_1(1-R_2)R_3+R_1R_2(1-R_3) \quad (1)$$

The expression of R1R2R32 represents that all three modules may work properly, and the expression of R2R3 (1-R1) indicates that the first module may fail and the second and third modules may work properly, and the expression of R1 (1-R2) R3 suggests that the first and third modules may work properly and the second module may fail, and the expression of R1R2 (1-R3) indicates that the first and second modules may work properly and the third module may fail [9]. An accurate estimate for reliability of a TMR system, so that the reliability of a voter be considered, is as Formula 2 :

$$R_{TMR} = (3R^2-2R^3)\, R_V \quad (2)$$

The voter is in alignment with the plug-in modules because the entire system fails if the voter fails. If we wish reliability of a TMR system is much more than a simple one, the reliability of a voter must be too high. Fortunately, compared to plug-in components, voter is a straightforward means, and so its probability of failure is much lower. The existence of a single point of failure is still unacceptable in some systems. We call each component a single point of failure that its failure leads to system failure. In this case, more complex voting schemes are used. To focus the system on a voter, we expand it to three voters. Figure 2 illustrates this scheme. Decentralized voting avoids the single point of failure, but requires the consensus of three voters.

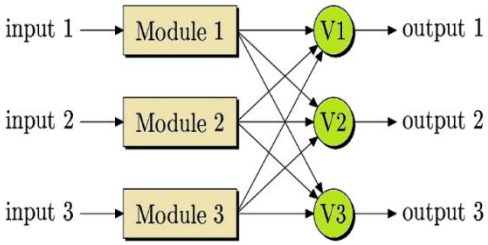

Figure 2. TMR system with three voters

The majority voting system with three inputs for digital data is represented in Figure 3.

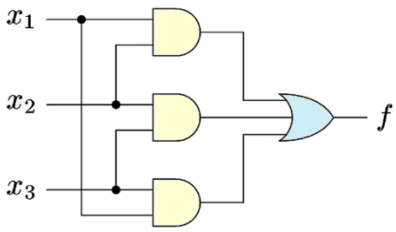

Figure 3. Logical diagram of a majority voter with three inputs

The output value F is determined by the majority of input values X3 and X2. Table 1 indicates the definition for this voting.

Given that the TMR mechanism in the control unit is the brain of the SDN network, the entire network will stop if it fails for some reason, our main purpose in this research is to increase the reliability of the controller unit in the SDN network by employing multi-controller units in the control unit. With this design, if the centralized control unit fails for any reason, the network will continue to operate without any failure through hiding defects. The topology considered is designed as a hierarchical architecture. To create redundancy, it consists of three identical controllers having the same responsibilities at one time, and with another controller lower than these three controllers, which has the duty for the correct output, the so-called voter. The controllers are placed vertically. They are divided between several levels, meaning that the control unit has several layers. This topology is designed in three levels. The voter controller, which is the output of the three controllers, can disrupt the whole network if it fails for any reason. To avoid this failure, in each

part, we connect to nested voters in other units via switch. By doing so, if a voter of a controller unit fails, we can communicate with voters of the other levels by switch. However, the voter fails very rarely. In Figure 4, the proposed architecture is shown.

Table 1. Definition table for majority voters

X1	X2	X3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

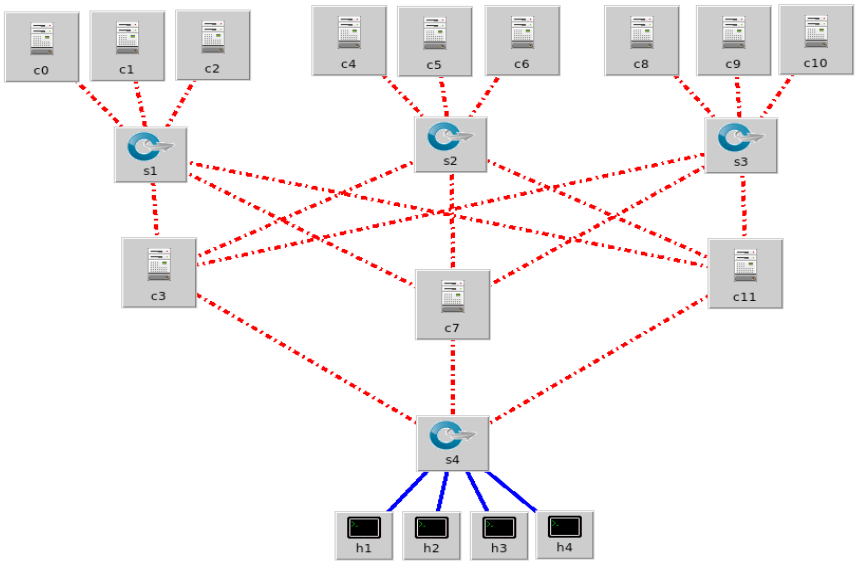

Figure 4. The proposed architecture using TMR mechanism

4. Evaluation

To evaluate the solution in this study, the open-source simulator Mininet is used [11]. The Mininet is designed to train and research in the field of software-defined networking (SDN) systems. With the help of it, we can easily create virtual software-defined networks. These networks include an open flow controller, an integrated Ethernet network consisting of open flow-based switches and multiple hosts connected to these switches. These networks have built-in functions that are supported by a variety of controllers and switches. Using the Mininet Python API, we can also produce more complex networks. Accordingly, to test the solution, the two single-control and multi-control topologies in fault-free and fault modes are compared in terms of improving efficiency to the number of packets that are exchanged between hosts and the packets sent to the switch, and the results obtained from using the Mininet simulator are investigated.

4.1. Investigating the number of packets lost between hosts in single-control and multi-control topologies

In normal network mode in single-control and multi-control topologies, the packets are sent and received between hosts without any packet loss. However, in the case of fault injection and the results obtained from the simulation at several different execution times, the rate of packets lost between the network hosts in the single-control topology is 62% on average. In the multi-control topology, it is equal to 44%, on average, at several different execution times. According to Table 2 and Figure 5, we will have:

Table 2. The percent of packet loss among different hosts in single-control and multi-control topologies at different execution times

result	Single control	Multi control
	75	50
#run1	75	50
(host to host)	75	50
	75	50
#run2	50	75
(host to host)	50	25
	50	25
#run3	75	50
(host to host)	25	50
	50	50
#run4	75	25
(host to host)	0	50
	75	0
#run5	100	50
(host to host)	75	25

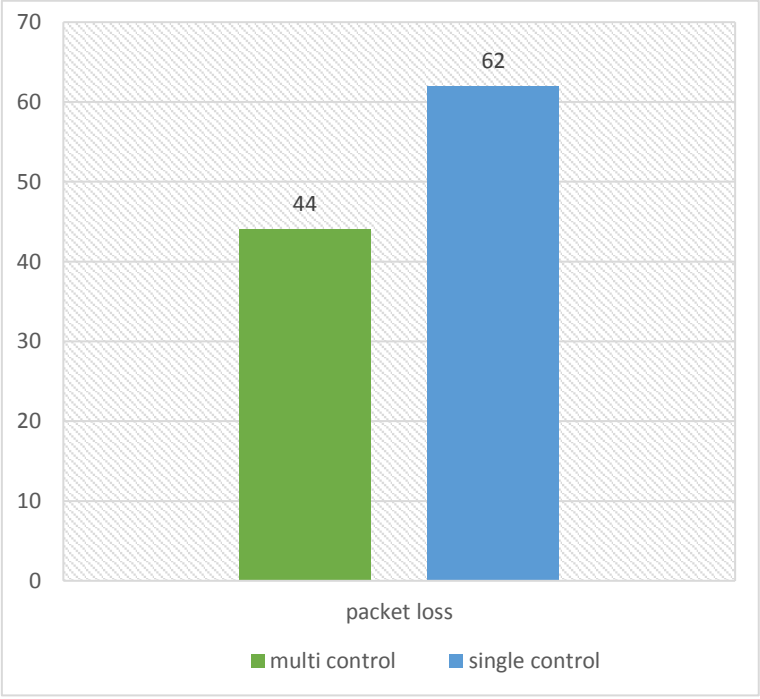

Figure 5. The average packet loss in single-control and multi-control topologies

4.2. Investigating the number of packets lost by the switch

In this study, in the normal mode of the network, all packets are sent to the switch without losing the packet switch. However, in fault mode, depending on the results gained from the simulator at several different execution times, the number of packets discarded in the single-control topology by the switch is 24% on average, and it is 20% on average in the multi-control topology. The results is shown in Figure 6 and Table 3.

Table 3. The rate of packets lost by switches in single-control and multiple control topologies at different execution times

Run#	Single control	Multi control
#run1	16	25
#run2	16	16
#run3	8	25
#run4	25	16
#run5	33	25
#run6	25	8
#run7	16	41
#run8	16	41
#run9	16	33
#run10	33	16

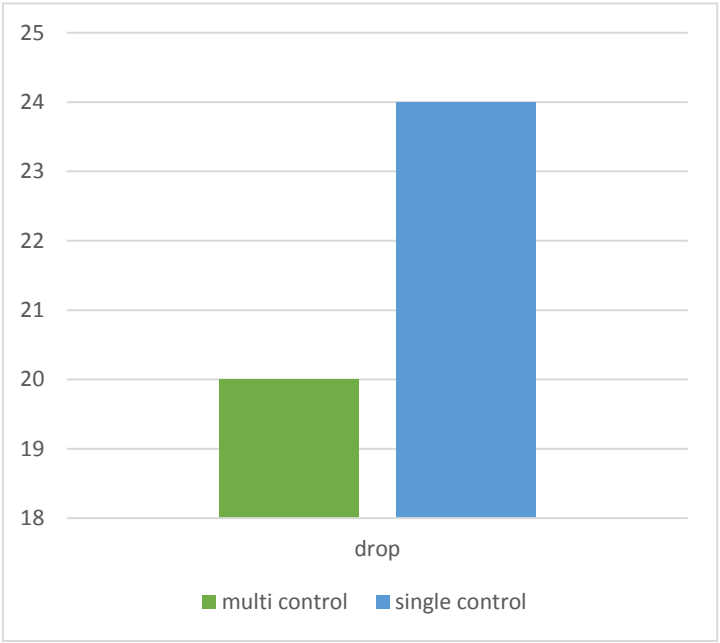

Figure 6. The average packets discarded by switch because of fault caused by single-control and multi-control topologies

5. Conclusions

Reliability is one of the major challenges in software-defined networks (SDN). To maintain and manage a network connection, network operations must be involved with a low level of configuration for implementing high-level and sophisticated network policies [12]. This may be due to the inflexibility of the network infrastructure. Hence, despite many approaches provided in the field of SDN reliability [13], many solutions are not employed today because of the problems in changing network infrastructure. Indeed, since these networks consist of a single control unit, the control unit is considered to be the brain of the network, and if it fails for any reason, the entire network will be inaccessible. Thus, in this study, with applying the Triple Modular Redundancy (TMR) mechanism, several control units have been used in SDN control unit that this increases the reliability of these networks. In fact, through this solution, if a controller in the control unit fails for any reason, there is another control to replace it. Therefore, the reliability and lifetime of the network rises. Then, to evaluate the proposed method, the two single-control and multi-control topologies are compared in terms of performance. The results demonstrated that the loss of packets between hosts in multi-control topology improved at a rate of 18% compared to single-control topology. Furthermore, the loss of packets discarded by the switch was optimized at a rate of 4%.

References

[1] A. Kannan, S. Vijayan, M. Narayanan, M. Reddiar. Adaptive routing mechanism in SDN to limit congestion. Information Systems Design and Intelligent Applications. Springer2019. pp. 245-53.

[2] I. Maity, A. Mondal, S. Misra, C. Mandal. Tensor-based rule-space management system in SDN. IEEE Systems Journal. 13 (2018) 3921-8.

[3] S. Lim, S. Yang, Y. Kim, S. Yang, H. Kim. Controller scheduling for continued SDN operation under DDoS attacks. Electronics Letters. 51 (2015) 1259-61.

[4] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, R. Kompella. Towards an elastic distributed SDN controller. Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking2013. pp. 7-12.

[5] S.H. Yeganeh, Y. Gangali. Kandoo. A Framework for Efficint and scalable Offloading of control Application. proceeding of the first workshop on hot topics in software defind networks HotSDN2016.

[6] M. Hamdan, E. Hassan, A. Abdelaziz, A. Elhigazi, B. Mohammed, S. Khan, et al. A comprehensive survey of load balancing techniques in software-defined network. Journal of Network and Computer Applications. (2020) 102856.

[7] M. Belyaev, S. Gaivoronski. Towards load balancing in SDN-networks during DDoS-attacks. 2014 International Science and Technology Conference (Modern Networking Technologies)(MoNeTeC). IEEE2014. pp. 1-6.

[8] C. Guo, D. Xie, Y. Han, J. Guo, Z. Wei. Survey of Software-Defined Network Security Issues. International Conference on Artificial Intelligence and Security. Springer2020. pp. 503-14.

[9] J. Vial, A. Bosio, P. Girard, C. Landrault, S. Pravossoudovitch, A. Virazel. Using TMR architectures for yield improvement. 2008 IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems. IEEE2008. pp. 7-15.

[10] E. Dubrova. Fault tolerant design: An introduction. Department of Microelectronics and Information Technology, Royal Institute of Technology, Stockholm, Sweden. (2008) 22-3.

[11] P. Chithaluru, R. Prakash. Simulation on SDN and NFV models through mininet. Innovations in Software-Defined Networking and Network Functions Virtualization. IGI Global2018. pp. 149-74.

[12] J. Seeger, A. Bröring, M.-O. Pahl, E. Sakic. Rule-based translation of application-level QoS constraints into SDN configurations for the IoT. 2019 European Conference on Networks and Communications (EuCNC). IEEE2019. pp. 432-7.

[13] A. Yan, Z. Xu, K. Yang, J. Cui, Z. Huang, P. Girard, et al. A Novel Low-Cost TMR-Without-Voter Based HIS-Insensitive and MNU-Tolerant Latch Design for Aerospace Applications. IEEE Transactions on Aerospace and Electronic Systems. 56 (2019) 2666-76.