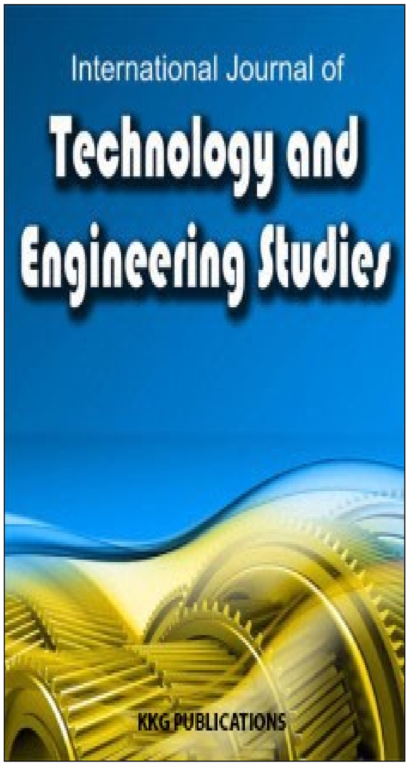

Key Knowledge Generation
Publication details, including instructions for author and subscription information:
http://kkgpublications.com/technology/

An Adaptable Failover Mechanism in Controller Cluster for Software-Defined Networking

YI-WEI MA [1], YAO-HONG TSAI [2], JIANN-LIANG CHEN [3], DA-WEI JIANG [4] JAI-YI JHOU [5]

[1] China Institute of FTZ Supply Chain, Shanghai Maritime University, China
[2] Department of Information Management, Hsuan Chuang University, Taiwan
[3, 4, 5] Department of Electrical Engineering, National Taiwan University of Science and Technology, Taiwan

Published online: 13 December 2016

To cite this article: Y. Ma, Y. Tsai, J. Chen, D. Jiang and J.Jhou "An adaptable failover mechanism in controller cluster for software-defined networking," *International Journal of Technology and Engineering Studies,* vol. 2, no. 6, pp. 185-188, 2016.
DOI: https://dx.doi.org/10.20469/ijtes.2.40004-6

To link to this article: http://kkgpublications.com/wp-content/uploads/2016/2/Volume2/IJTES-40004-6.pdf

KKG PUBLICATIONS

International Journal of Technology and Engineering Studies
vol. 2, no. 6, pp. 185-188, 2016

IJTES

AN ADAPTABLE FAILOVER MECHANISM IN CONTROLLER CLUSTER FOR SOFTWARE-DEFINED NETWORKING

YI-WEI MA [1], YAO-HONG TSAI [2], JIANN-LIANG CHEN [3], DA-WEI JIANG [4], JAI-YI JHOU [5*]

[1] China Institute of FTZ Supply Chain, Shanghai Maritime University, China
[2] Department of Information Management, Hsuan Chuang University, Taiwan
[3, 4, 5] Department of Electrical Engineering, National Taiwan University of Science and Technology, Taiwan

Keywords:
Software-Defined Networking
Failover
Multiple Controllers
Controller Cluster

Received: 02 August 2016
Accepted: 04 October 2016
Published: 13 December 2016

Abstract. This work proposes an adaptable failover mechanism for use in the control plane that combines various controllers to deal with controller failure to ensure that controller failure is tolerable. Each controller in the cluster of controllers has one of two roles: leader and member. The failover system is composed of four components, which are state monitor, failure detector, event actor and schedule handler. The failover mechanism reduces the overall network failover time, improves network stability and optimizes the processing performance in the data plane. This work analyzes the failover mechanism that is based on a controller cluster in software-defined networking. The results thus obtained reveal that the proposed mechanism effectively detects a failed controller, and the failover mechanism reduces the failover time by 22.46%.

INTRODUCTION

Everyday life now depends deeply on the Internet, and more services depend on it. The distributed architecture of a traditional network faces several challenges, such as virtualization, programmability, flexibility and others. A traditional network cannot meet the requirements of users today. Therefore, the architecture of a SDN must improve upon that of a traditional network. Overall, a SDN can be managed more automatically and flexibly. SDNs may therefore change the future of network environments [1].

The rest of the paper is organized as follows: Section 2 presents the background of this study. Section 3 introduces the proposed adaptable failover mechanism in controller cluster. Section 4 presents the performance analysis of this study. Finally, conclusions are drawn in Section 5.

Background Knowledge

The most popular way to establish a SDN involves OpenFlow. A hot technology, OpenFlow is used by many campuses and research institutions around the world. OpenFlow is based on centralized control and is composed of the OpenFlow protocol, the OpenFlow switch, and the OpenFlow controller. With respect to centralized control, all routing paths are established by the controller when the first packet of each data flow is forwarded to it. However, as the size of deployed SDN networks increases, depending on a single controller for an entire network may not be feasible. Additionally, if the controller fails, then another controller must replace it. If the main controller fails, then the hot standby controller will take over from it. As the size of networks continues to increase, the number of hot standby controllers in those networks will also increase costs [2], [3], [4], [5], [6], [7], [8].

SYSTEM OVERVIEW AND ARCHITECTURE

This work proposes an architecture of a cluster of multiple controllers to manage a network environment using SDN; to coordinate the controllers and to determine switch to appropriate controller when one controller fails. The cluster is composed of multiple controllers and a database. Each controller in the clusters has the role of either leader or member, and establishes a control channel to SDN switches in the data plane. In the control plane, a leader checks the timestamp of each member and one member checks the timestamp of the leader. If a controller fails, then it will be marked as a failure and failover will be executed by the failed controller, involving, for example, the scheduling of a reassignment one controller responsible for SDN switches and taking over the task of the failed controller. By the proposed mechanism, the stability of the network can be increased and the recovery time of a failed network can be reduced. Figure 1 shows proposed architecture of the system.

*Corresponding author: Jai-Yi Jhou
†Email: yiweimaa@gmail.com

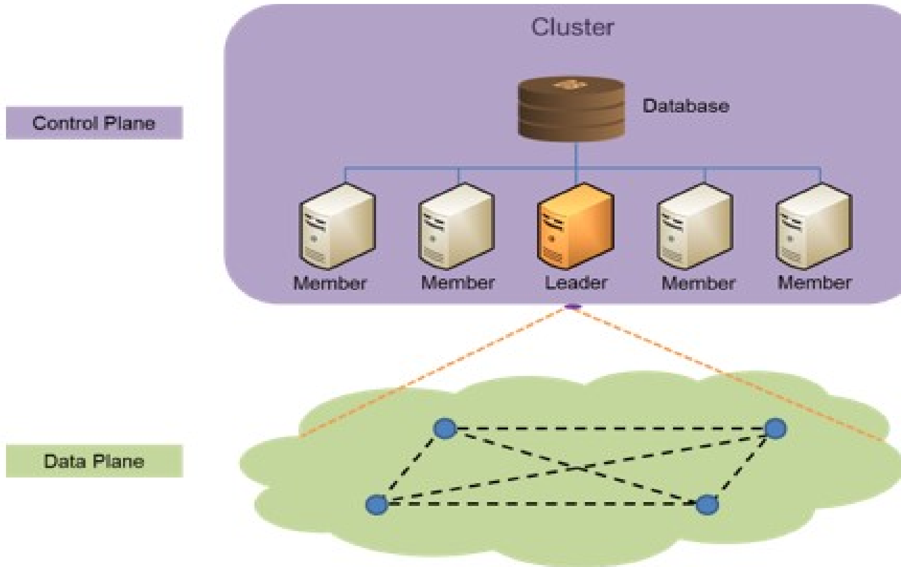

Fig. 1. System architecture overview

System Mechanism Design

This work developed a failover mechanism that is based on a controller cluster to solve SPOF (Single Points of Failure) problems. The failover system is composed of four modules, which are (1) state monitor, (2) failure detector, (3) event actor, and (4) schedule handler. The state monitor synchronizes the system clocks of all controller and collects network information, and the CPU utilization and system time of each controller. It then generates a timestamp based on the current system time and stores this information in the state database; the failure detector queries this state database for information in period of time and detects whether controllers have failed by calculating the latency between the timestamp and the current system time. If the latency of the controller in the cluster exceeded a threshold for timeout, then that controller will be marked as a failure and a HELLO message sent to determine its operating state. The failure of a controller triggers the notification of the event actor. When the event actor receives this notification, it will find an alternative controller and selects different from controller cluster based on the role of the controller. Then, the event actor sends a message to the schedule handler, which, upon receipt thereof, executes changes in the role of controller. Finally, by the alternative controller updates the reassigned information to switches in the flow table.

Operations of the System Mechanism

The initialization mechanism is described as follows:

- Step 1: The leader controller sends the clock-synchronized message to the member controller and receives the reply.
- Step 2: The leader controller checks whether the latency between the member controller and itself exceeds the systems limiting latency.
- Step 3: The leader controller chooses two nearby member controller close and activates the failure detector.

The failover mechanism operates as follows:

- State monitor: In every period of time, the state monitor collects the switch information of network under the member controller, the CPU utilization and the system time of the member controller. Then, it generates a timestamp and stores all of this information in a database.
- Failure detector: The failure detector queries the database for the required information and determines whether member controllers have failed by calculating the latency between the timestamp of each and the current system time. If a member controller has failed, its operating work state will be checked.
- Event actor: The event actor calculates the cost of finding an alternative member controller to take over the switches and sends the reassignment message to the alternative member controller after it has received the notification from the failure detector.
- Schedule handler: The schedule handler performs that they are selected for something upon receipt of the relevant message. The alternative controller updates the flow table with these reassigned switches for response.

The modules of the failover mechanism are described as follows:

- State monitor: In every period of time, the state monitor collects the switch information of network under the leader controller, the CPU utilization and the system time of the leader controller. Then, it generates a timestamp and stores all of this information in a database.

- Failure detector: The failure detector queries the database for the required information and determines whether leader controllers have failed by calculating the latency between the time stamp of each and the current system time. If a leader controller has failed, its operating work state will be checked.
- Event actor: Event actor sends the role-changing message and reassignment message to the appropriate controller.
- Schedule handler: The schedule handler changes the role of controller. The alternative controller then updates the assignments of the switches in the flow table in all instances.

PERFORMANCE ANALYSIS

This work establishes a test-bed with the OpenDaylight controller and Mininet as an experimental environment. In the following section, simulates failover times with and without failover mechanism are compared. Without this failover mechanism is composed of multiple controllers that use a heartbeat message to notify other controllers of their functioning. Without this failover mechanism, the multiple controllers cannot coordinate with each other, so failure detection takes longer. The multiple controllers do not execute an algorithm to calculate the minimum cost of switch allocation. Failure to consider latency between the switch and the controller increases the time required to take over switches. However, in the system with the failover mechanism, the controllers in the cluster have two roles: leader and member. The leader controller coordinates member controllers and calculates the minimum cost of switch allocation to identify the best alternative controller to take over switches. The mean failover time with the proposed failover mechanism is lower than that without it. Hence, the proposed failover mechanism reduces the failover time and solves the single point of failure problem.

CONCLUSION AND RECOMMENDATIONS

The distributed architecture of a traditional network is facing many problems, such as virtualization, programmability, flexibility and others. Hence, SDN architecture must improve upon traditional networks. SDN supports more automatic and flexible network management, and provides favorable programmability. However, as the size of deployed SDN networks increases, depending on a single controller becomes unfeasible. Moreover, if such a controller fails, then the network that is controlled by the failed controller will break down. The multiple SDN controller architecture is proposed to solve the problems of scalability and the SPOF. To improve the stability of a SDN network and prevent the single point of failure problem, this work exploits multiple controllers, decentralization, data sharing and grouping. A failover mechanism that is based on a controller cluster in SDN is proposed. A control plane that combines multiple controllers in a controller cluster is developed herein to improve network stability. A failure detection and failover mechanism is proposed, based on the use of a timestamp to identify the failed controller, assign its task to another, appropriate controller and execute action of corresponding.

Declaration of Conflicting Interests

This study possesses no conflicts of interest.

Acknowledgment

Authors thank for financial supports of the Ministry of Science and Technology, Taiwan under contract MOST 103-2221-E-011-069-MY2 and 103-2221-E-011-067-MY2.

REFERENCES

[1] S. A. Shah, J. Faiz, M. Farooq, A. Shafi and S. A. Mehdi, “An architectural evaluation of SDN controllers,” in *IEEE International Conference on Communications (ICC),* 2013, pp. 3504-3508.

[2] S. Betge-Brezetz, G. B. Kamga and M. Tazi, “Trust support for SDN controllers and virtualized network applications,” in *1st IEEE Conference on Network Softwarization (NetSoft)*, 2015, pp. 1-5.

[3] S. Baik, C. Hwang and Y. Lee, “SDN-based architecture for end-to-end path provisioning in the mixed circuit and packet network environment,” in *Network Operations and Management Symposium (APNOMS), 16th Asia-Pacific,* 2014, pp. 1-4.

[4] M. A. Santos, B. A. Nunes, K. Obraczka, T. Turletti, B. T. de Oliveira and C. B. Margi, “Decentralizing SDN's control plane,” in *39th Annual IEEE Conference on Local Computer Networks,* 2014, pp. 402-405.

[5] A. Dixit, F. Hao, S. Mukherjee, T. V. Lakshman and R. Kompella, “Towards an elastic distributed SDN controller,” in *ACM SIGCOMM Computer Communication Review,* vol. 43, no. 4, pp. 7-12, 2013.

[6] M. Karl, J. Gruen and T. Herfet, “Multimedia optimized routing in openflow networks,” in *19th IEEE International Conference on Networks (ICON),* 2013, pp. 1-6.

[7] S. Y. Wang, “Comparison of SDN open flow network simulator and emulators: EstiNet vs. Mininet,” in *IEEE Symposium on Computers and Communications (ISCC),* 2014, pp. 1-6.

[8] A. Bianco, V. Krishnamoorthi, L. Nanfang and L. Giraudo, “OpenFlow driven ethernet traffic analysis,” in *International Conference on Communications (ICC),* Sydney, Australia, 2014, pp. 3001-3006.

— This article does not have any appendix. —

KKG PUBLICATIONS