The Seybold Report

ISSN 1533-9211

DOI: 10.5281/zenodo.10025356

RELIABILITY AND SCALABILITY OF DIFFERENT CONTROLLERS IN SOFTWARE DEFINED NETWORK

Er. CHIMAN SAINI [1], Dr. PUNEET SAPRA [2], Er. POONAM KUKANA [3] and Er. SUKHJINDER KAUR [4]

[1, 2, 3, 4] Department of Computer Science and Engineering, University School of Engineering & Technology, Rayat Bahra University, Mohali, Punjab, India.
Email: [1] chimansaini1994@gmail.com, [2] puneetsapra91@gmail.com,[3] poonamkukana@gmail.com, [4] skaur29100@gmail.com

Abstract

Software defined network (SDN) is a computer-based solution that, as a smart network manager, creates a single control plane to monitor the entire network system that only transmits data, rather than handling each system alone that is wasting time and effort. The transformation from physical network to SDN poses significant challenges which need to be deeply considered. Scalability and reliability are considered as important factor affecting the SDN network. In general, the reliability and scalability requirements are as follows: the chosen data rate during the transmission of information, the elimination of a single cause of failure, which increases the controller's availability, and finally, there are several topologies that enable scalable network architecture. Moreover, these requirements must be fulfilled within the SDN framework. We analyze in depth the structure of several controllers (i.e., hierarchical, scattered, and centralized). We examined controller failure program that improves SDN network flexibility and allows the infrastructure more robust and scalable.

INTRODUCTION

Data volume grows dramatically, regardless of the network systems that manage such a massive data. Network maintenance cannot be evaded any longer, with the emergence of the Internet of Things era and computer clouding technologies dealing with big data, but is instead at the centre of business needs, efficiently operating network services in a costly, high-quality manner and enabling service provisioning as quickly as possible. There was a change in the manner company works. The physical departments and regional boundaries are unrestricted. Services must be always consistently available everywhere and all the times. Software Defined Networking (SDN) relies on cutting-edge methods that divide the data and control plane, which is refereed as a data transmitting white box network, without being able to make any decisions. Decision-making is limited only to plane controllers.

This strategy helps to simplify more than one network equipment, to develop various applications that relay on a broad view of the whole network status, and for simplicity in integrating the unique applications. Systems for traffic balancing and routing, for instance, can be put into place one after the other. The default structure of software defined networking serves as the design for the network's central control. This form of centralization is useful for the network's longer term. By adding features like virtualization, selective delegation, federation of responsibility, in addition to SDN hybrids and conventional networking, it needs to be improved and further developed. The SDN network has serious problems that need to be

ISSN 1533-9211

DOI: 10.5281/zenodo.10025356

addressed. We emphasize on reliability and scalability because these two related issues have an impact on how effectively the SDN network performs. If the control plane design can handle, deliver, and maintain the same level of quality for network services as the network gets more complicated, then an SDN network is scalable. We assume that the scalability and efficiency of the software defined network means specialized coverage at any time for any network size. In the default configuration, SDN shifted away the control plane from the information plane and uses only single controller to take only one choice. But as group measurement increases, the requirement cannot be met by the centralized architecture. The strategy accepts and rejects flow by protocols such as open flow in group modules in a conceptually centralized controller (such as NOX and Ryu). The SDN architecture served as a foundation for the development of network as a service and network function virtualization (NVF), Therefore, there is an only one failure point and scalability concerns, and the controller may be dispersed across various servers simultaneously maintaining a consistent network chart maintained in an excellent repository to organize the task across distributed control. In addition to the scaling problem, the device's whole reliability is compromised by the fact that a failure might occur at just one point in time. The entire system will stop functioning if a centralized controller malfunctions for some reason involving software, hardware, or networking. Only Open Flow protocol-based controllers are taken into consideration here, even though all types of these controllers are vulnerable to failure situations. Several SDN manage plane topologies are given and contrasted.

Clustered, hierarchical, and distributive SDN control architectures are the three types. Contrarily, we are discussing a conceptual framework for a fallback controller relies on a central controller duplicated with a backup controller running in a different site as a stand-by controller to also become a disastrous backup that provides a stable and scalable SDN device.

SDN Controller Architecture

The control layer exists between the application and data layers as shown in Figure 1.1.

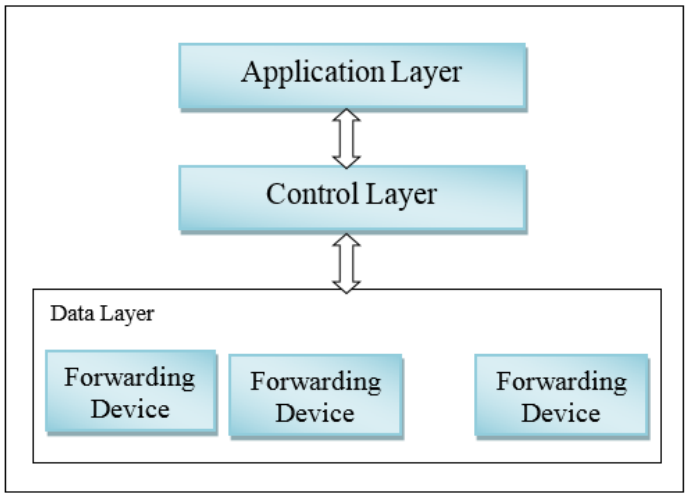

Figure 1.1: SDN Layers

ISSN 1533-9211

DOI: 10.5281/zenodo.10025356

The SDN controller meets the demands of the applications and executes actions on the forwarding devices while the giving network details for the various SDN applications up to the north bound interface drivers. It will contribute to developing flow rules and to gathering statistics.

It is easier to build a separate control network from the data network. It provides dependability, intrusion prevention, network planning simplicity, and a safe and stable atmosphere. We'll go over the controller configurations and illustrate the system's flexiability and dependability. Independent control theory blends in with the independent network management philosophy.

Centralized Controller

For an SDN the default architecture is a centralized controller. The controller initializes flow tables, and monitors them. The controller, too, is responsible for collecting data. Figure 1.2 illustrates a unified control plane (similar to the standardized design for Ryu and NOX).

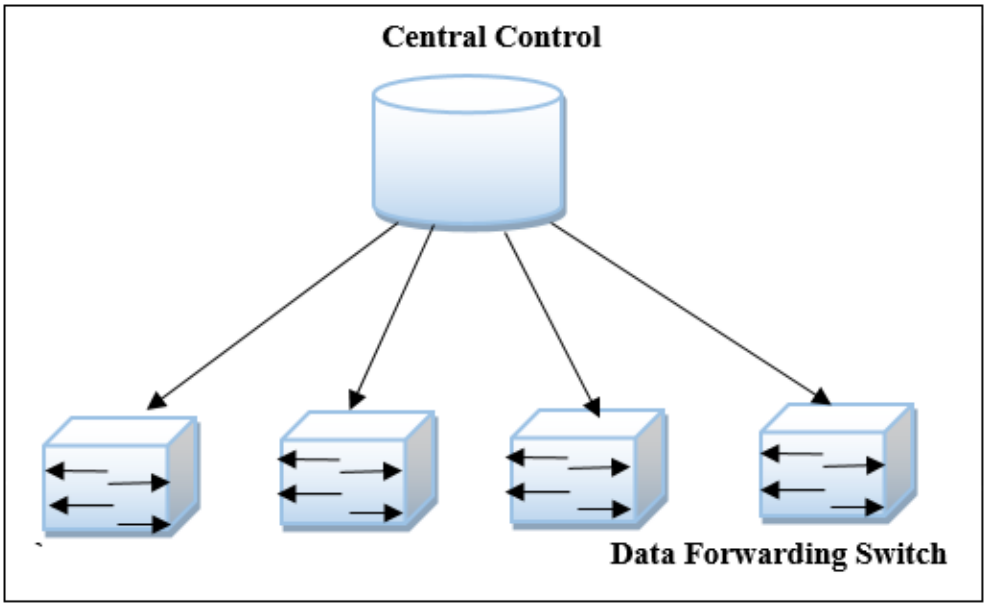

Figure 1.2 SDN Central Controllers

On a small or medium scale, the central controller is very reliable, yet, it suffers from massive overflow, which represents a high level of complexity. The intended quality of service goals does not match due to service or even a reactive pause that does not fit the current era of machine clouding and IOT. The central controller is unable to function owing to a wiring failure or a system corruption. Under this context, a single point of is defined as the entire network, or the entire open-flow, breaking free of constraints or being unable to transmit any flow. Free from constraint, or unable to transmit any wind. If we discover that the controller is a technical firewall and that it is deactivated due to a power outage, the data layer device will not redirect any traffic since it lacks the operator, so weakening the network. The primary controller has no alternative or alternate plan for taking over the primary controller. The central controller lacks

ISSN 1533-9211

DOI: 10.5281/zenodo.10025356

a standby controller installed at a distinct site to preserve the important configurations in the event of any environmental conditions, such as a significant fire affecting the central controller, placing the SDN network in serious danger.

The resulting considerations revel that the SDN's central controller is not versatile or stable, and thus does not fit the professional networks or SDN-WAN. We might want to expand the controller's design to create a more stable system, such as distributed and hierarchical architecture.

A. Hierarchical Controller

The traditional three - layer infrastructure is used in SDN in a simplified form in this model, with only 2 layers hierarchy control plane, similar to that used in (e.g., kandoo). The controllers coordinate themselves into a hierarchy as in figure 1.3. The first level contains one master and multiple slave controllers at second level. The distribution controller is physically separated from each other and controls one area of Open Flow switches. To eliminate network cycles, there is no direct link between the distribution controllers.

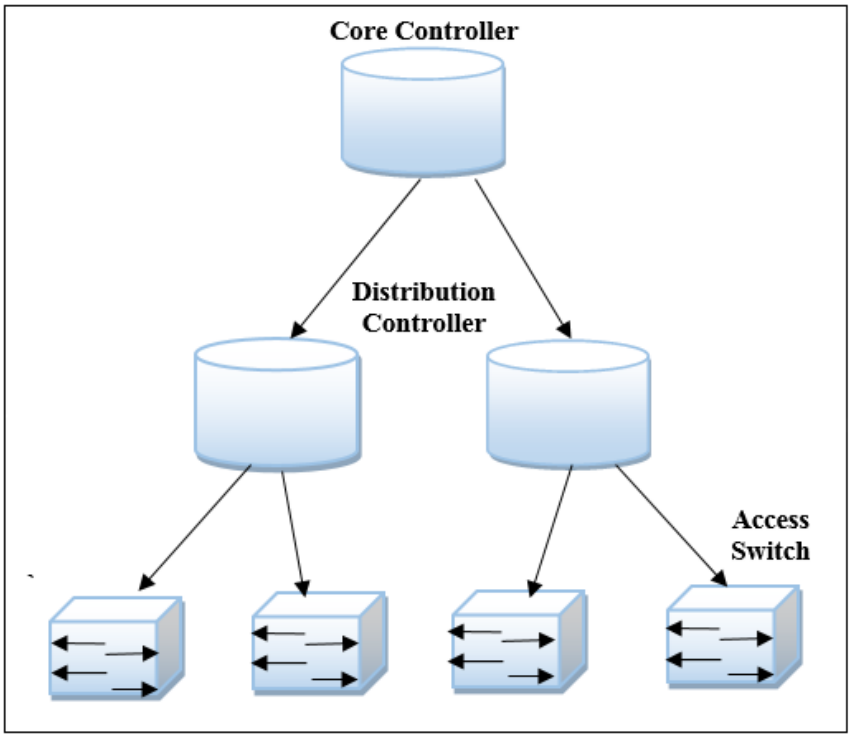

Figure 1.3 Hierarchical Controllers

As a Genius who controls the distribution controller relationship, each distribution controller is connected to the core controller. The location of the root controller in this situation is determined by the source and destination nodes. When a distribution controller's source and destination nodes are in the same region, for more precise demands, the distribution controller can then make a conclusion and bring in the Open Flow switches. Unless the destination and

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.10025356

source nodes are in distinct sections of the distribution controller, the transmission controllers will convey the query to the main controller, where a decision will be taken and extended to the distributed controller that it belongs to the sender.

The distributed controller resembles a layer 2 switch, permitting the routing of a specific virtual LAN and an unknown VLAN to the upper multilayer switch, which functions as a virtual router for the multiple VLANs. The central controller in this situation is known as a multilayer switch. The network state will become more complicated within the hierarchical control system. Each physical change, whether caused by a port failure or simply a replacement, must be conveyed to the distributed controller and the main controller. The hierarchical structure has the potential for greater scalability due to its capability to allocate some function from the main system distributed controllers, which allows for greater flexibility than the central controller but is still seen as a single point of failure by the root controller. Another master controller will support with this issue. Each root controller may manage specialized open flow switches while also supporting other open flow switches. This condition results in a fairly common network problem known as a "loop."

When there is more than one path connecting devices in a closed loop cabling, or in another setting, a network loop occurs. The conventional loop elimination approach is the spanning tree protocol, which blocks one port while allowing the other. The STP extends the configuration's complexities, as well as the connections between controllers and the periodic checks in the negations. On an SDN network, it is tough to configure.

To do the required work, for example, you must operate with HP switches that allow open flow policy and STP protocol, which restricts your options, as well as white box switches based on NetFPGA considering the complexity of execution, hierarchical design provides an approach that is efficient, scalable, and dependable.

B. Distributed Contoller

The architecture of a distributed network is shown in Figure 1.4. The controller has either a local sub-network view or a global view of the entire network, and each has absolute usability and can interact with other controllers. The controller must coordinate with other controllers in this so-called local-view peer-to-peer control plane to implement a query for flow and establish a universal flow as well. Among controllers the process of acquiring statistics always cooperates. Each controller collects statistics only for the sub-network they monitor.

The controllers act as integrated managers in the control plane P2Pdistributed with global perspective with similar responsibilities throughout the network and can therefore process entire flow initialization requests created by the switches within their control region. Despite the fact that this design eliminates the only failure point, it struggles from the great friction between the different controllers. It is necessary to repeat and update to the other controller in any controller which reduces the controllers 'efficiency as operator to the data paths architecture. The infrastructure control plane has the maximum overall performance of reliability and scalability and the P2p with local view achieves second performance in terms of size and diameter. In terms of scalability, dispersed and peer-to-peer with global view control

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.10025356

perform poorly. The previous designs have a notable benefit but they kill the network in varying volumes when spike in load happens and make a delay in flow configuration impossible. Within the next segment we suggest a failover strategy to eliminate the preceding problems.

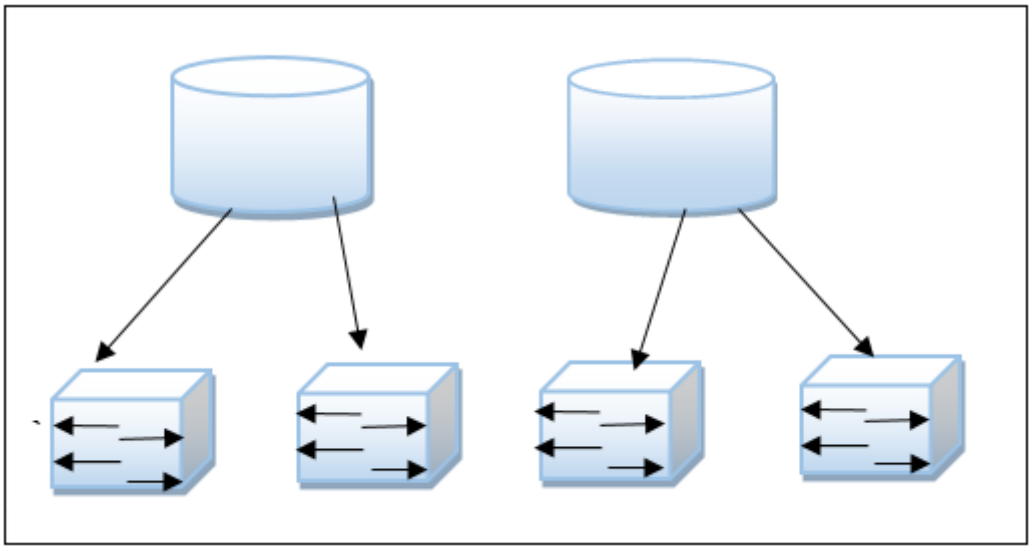

Figure 1.4: Distributed Controllers

Reduction in controller supply has a detrimental and direct effect on SDN reliability and scalability. We are developing a failover strategy prototype in order to build a strong SDN controller capable of providing uninterrupted network services. The key characteristics of a highly functional, failover system are durability, recoverability, and continuous operations.

C. Failover Strategies for Control Platform

An additional innovation is that each controller has its own raid system data store to provide hardware redundancy and coordinate through an atomic messaging network that keeps all servers synchronised, such as the Apache zookeeper. The failover approach is as dependent on a backup controller linked to the network devices as it is on the main controller. A public data server, such as NAS storage to duplicate the resources and states, or private cloud, will manage them. The Ryu controller can be used to carry out the final plan; we can immediately assign certain rules to a central controller and a slave controller and assess the master controller's availability using the switch and controller's standard communications. If the data route fails to link the master controller and the slave control requires that its position be tested on a regular basis, the switch informs the slave that the data route has been adjusted. The master becomes the slaves, and the slave controller is elevated to the position of master c ontroller.

The third alternative is the best, but because each controller cannot link directly in the normal way, switch load is raised.

A controller is either active in this architecture or is stand-by. The controller collects and processes Open Flow messages in an active situation while the controller duplicates the

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.10025356

disabled controller features in standby mode. In the case of an actual controller failure, the backup controller will substitute the real and complete the cycle. The standby controller immediately takes over monitoring of network connectivity and the handling of data flow. This technique is called a failure of controller. Controller failure is the opposite of failover procedure.

In each situation, there are three types of failover strategies: hot standby, warm standby, cold standby, and prober mode. Without impacting the consumer, the time required for failover can be reduced. The hot standby method is the highest powerful failover solution since it retains the whole network state prior to collapse.

D. Metrics for Reliability and Scalability based on Failover Strategy

1. Health positive rating for measuring the proportion of time spent in excellent health in a failover program.
2. Failover length describes the amount of time required for a system to overcome from failures. The failover cycle for the hot standby technique may be rather short.
3. Time during which the primary controller is managed is known as maintenance time.

Since milliseconds are the acceptable maximum service time delay, the SDN platform should at least have one stable controller with a controller dependability degree. Failure time will be as minimum as and near to zero.

E. Failover Controller Requirements

The control platform needs to fulfil the following criteria to enable multiple controllers, prevent single point of failure, and verify that the reserve controller has adequate processing capabilities for network structure and data transmission monitoring in the event of main controller failure:

1) As shown in figure 1.5, each of them must have minimum two controllers and an individual or shared data store.
2) Hardware and software types are similar.
3) To improve dependability, a separate fiber network combines the two controllers with huge bandwidth.
4) An alarm scheme having varying alert rates, ranging from critical to regular.
5) The controllers are linked to the data plane through a network.
6) High resources like large cash processor and huge memory to withstand the hard work.
7) To achieve the disaster backup strategy, the two controllers are placed at a separate location.
8) Every controller should have an uninterruptible power supply (UPS) on it.
9) Electricity from a station at various geographical regions.

ISSN 1533-9211

DOI: 10.5281/zenodo.10025356

10) Each device's network adapters should be comparable, with the similar procedure, capability, and frequency.

11) Every controller must be housed in a quality server center with secure access, a closed door, fire resistant, and a good chiller.

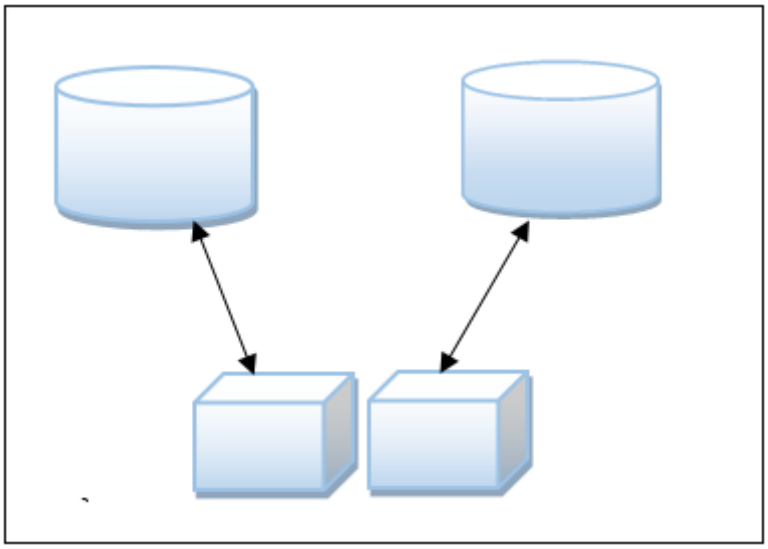

Figure 1.5 Data Storage Shared Between Two Controllers

F. Failover Controller Workflow

The key regions in the connection between the two controllers should be defined so as to build a stable and scalable system using standby techniques: a starting state that happens at first to coordinate the sequence of controller launch, a functional situation in which a devoted controller becomes the expert controller while the backup controller waits and observes by testing the relationship. Finally, failover denotes a slave controller attempting to substitute the main controller by pulling power from the former primary power supply.

During the start controller stage, the, main and backup controllers each run their respective configuration files. In any controller-connected system, related procedures include how to handle incoming traffic, how to handle a new data routing switch or new event, and how to manage failures. By constantly delivering Hello messages to switches, the master controller is kept alive via an echo mechanism. This approach is also used by the slave controller to guarantee that although the original master was down, he is still alive and ready to become the new master. Both have a mechanism for determining if their status is comparable, master, or slave. Lastly, the slave contains an extra mechanism that regulates its movement while the master is unavailable, enabling the controller to operate.

Based on the Open Flow protocol, the master controller arranges and analyzes the Open Flow switches in the functional specified direction, or the ordinary usual phase, and oversees the movement of data. Based on heartbeat signals, the backup controller detects malfunction in the

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.10025356

original controller failure mode. The standby controller takes over as primary controller after a timeout, determining the problem in accordance with certain legislation. The temporary master controller's primary responsibility is to restore network applications and services and to turn on the network interface, which allows network devices to be monitored. They suggest simplifying the SDN network even further by providing a data storage that exchanges data with the main and backup controllers via read only and repeating images inside a common network knowledge base.

The standby controller will understand this based on the network operating system's capabilities. Several notifications or data should be transmitted or shared between the two controllers in a gateway to enable for a quick recovery from the collapse of the master controller activities between them. Such officers include:

1) The master's failure warning if it abnormally shuts down to ensure real basis coordination between the two controllers.
2) Topological data, which comprises information such as the controller internet address, connectivity bandwidth, switch mac address, and host internet addresses, is one example of a network characteristic.
3) QOS details include the rate of loss of packets, transmitted packets, accepted packets, and delay.
4) Heart rate alert to keep an eye on controller's condition. It is communicated from one controller to another on a regular basis to ensure that it is not a deed.

G. Design of a Failover Controller

We recommend the control platform design presented in figure 1.6 to eliminate single point mistakes in the control platform.

Two controllers at two different locations include the master and slave controller with common data storage and a separate data network controller. We were using Mininet to digitally create a complicated network. Begin with the Python script for that controller, followed by the Mininet script for network creation.

Pinging the several devices under the master controller allows us to test the connection. Through simply canceling the query, you can disconnect the master controller through recognizing the evolving position that makes the master controller to slave and vice versa. If you can successfully ping two machines beneath the new master controller's oversight, your network is in good health

ISSN 1533-9211
DOI: 10.5281/zenodo.10025356

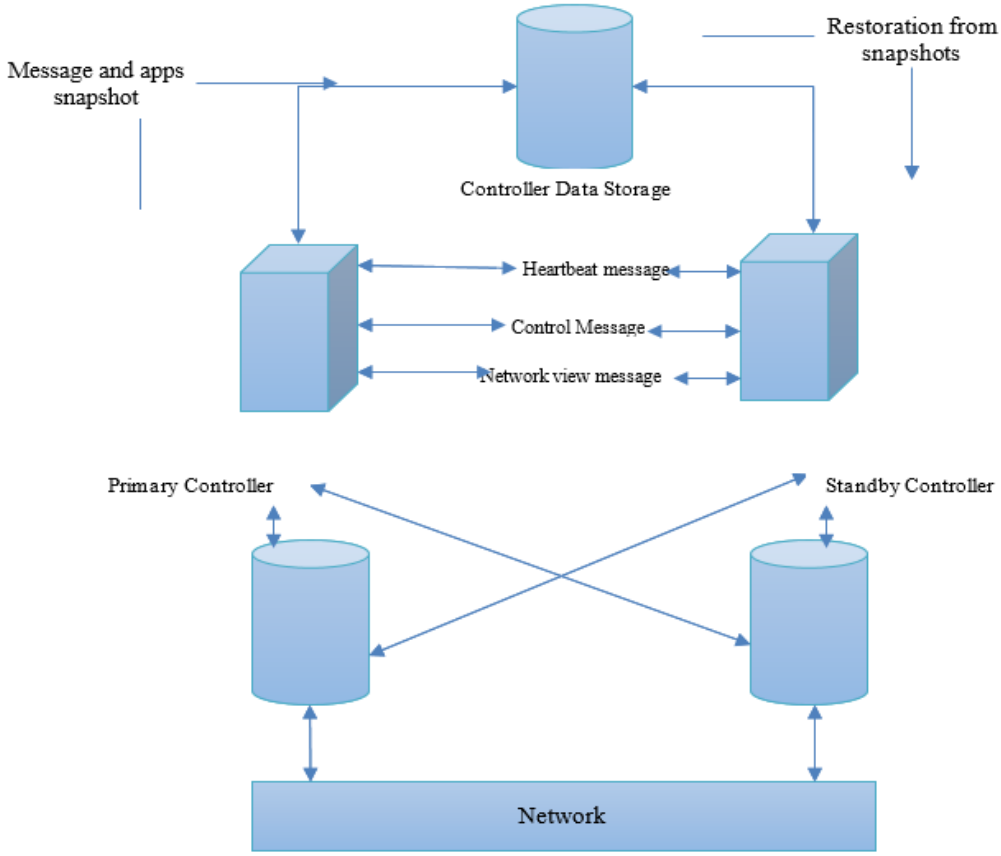

Figure 1.6: Failover SDN Controller

CONCLUSION

The study of several topologies ranging from hierarchical and central to distributed, as well as our recommendation to utilize a failover approach show how difficult it is to rely solely on single design to establish a scalable and reliable SDN network. While failure ensures high reliability, making the SDN more dependable and deployable in the shortest amount of time possible, it is not the optimal situation for large scalability because it is always dependent on single controller. Hierarchical structure is a worthy opponent, providing a highly scalable network at the sacrifice of dependability because the base controller is only point of failure, scalability suffers. We suggest an approach for future research relies on a hybrid from a conceptual structure with core layer failure strategy of becoming one master one basic controller controlling the configuration of data delivery switches and a slave comparison for the other data delivery switches in order to achieve both reliability and scalability. If one of them failed at any moment, the other would take over as the primary master for the whole master network of data delivery switches. Because the two major switches are physically coupled, s spanning tree approach is required to break the cycle.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.10025356

References

1) Hohlfeld, O., Kempf, J., Reisslein, M., Schmid, S., & Shah, N. (2019). Scalability Issues and Solutions for Software Defined Networks.

2) Oudin, R., Antichi, G., Rotsos, C., Moore, A. W., & Uhlig, S. (2018). OFLOPS-SUME and the art of switch characterization. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2612-2620.

3) Popovic, M., Khalili, R., & Le Boudec, J. Y. (2017, March). Performance comparison of node-redundant multicast distribution trees in SDN networks. In *2017 International Conference on Networked Systems (NetSys)* (pp. 1-8). Ieee.

4) Xiao, Y., & Krunz, M. (2018). Dynamic network slicing for scalable fog computing systems with energy harvesting. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2640-2654.

5) Yan, B., Xu, Y., & Chao, H. J. (2018). BigMaC: Reactive network-wide policy caching for SDN policy enforcement. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2675-2687.

6) Sakic, E., & Kellerer, W. (2018). Impact of adaptive consistency on distributed sdn applications: An empirical study. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2702-2715.

7) Lyu, X., Ren, C., Ni, W., Tian, H., Liu, R. P., & Guo, Y. J. (2018). Multi-timescale decentralized online orchestration of software-defined networks. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2716-2730.

8) Uddin, M., Mukherjee, S., Chang, H., & Lakshman, T. V. (2018). SDN-based multi-protocol edge switching for IoT service automation. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2775-2786.

9) Moradi, M., Zhang, Y., Mao, Z. M., & Manghirmalani, R. (2018). Dragon: Scalable, flexible, and efficient traffic engineering in software defined isp networks. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2744-2756.

10) Fu, Q., Rutter, B., Li, H., Zhang, P., Hu, C., Pan, T., ... & Hou, Y. (2018). Taming the wild: a scalable anycast-based cdn architecture (t-sac). *IEEE Journal on Selected Areas in Communications*, *36*(12), 2757-2774.

11) Zhang, X., & Zhu, Q. (2018). Scalable virtualization and offloading-based software-defined architecture for heterogeneous statistical QoS provisioning over 5G multimedia mobile wireless networks. *IEEE Journal on Selected Areas in Communications*, *36*(12), 2787-2804.

12) Aglan, M. A., Sobh, M. A., & Bahaa-Eldin, A. M. (2018, December). Reliability and Scalability in SDN Networks. In *2018 13th International Conference on Computer Engineering and Systems (ICCES)* (pp. 549-554). IEEE.

13) Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2014). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, *103*(1), 14-76.

14) Yeganeh, S. H., Tootoonchian, A., & Ganjali, Y. (2013). On scalability of software-defined networking. *IEEE Communications Magazine*, *51*(2), 136-141.

15) Bannour, F., Souihi, S., & Mellouk, A. (2018). Distributed SDN control: Survey, taxonomy, and challenges. *IEEE Communications Surveys & Tutorials*, *20*(1), 333-354.

16) Guan, X., Choi, B. Y., & Song, S. (2013, March). Reliability and scalability issues in software defined network frameworks. In *2013 Second GENI Research and Educational Experiment Workshop* (pp. 102-103). IEEE.