Addis Ababa University College of Natural and Computational Sciences

Distributed SDN Controller Architecture for LAN Management

Zenebe Kassie

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment of the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia

December 2023

Addis Ababa University

College of Natural and Computational Sciences

Zenebe Kassie

Advisor*: Minale Ashagrie* (PhD)

This is to certify that the thesis prepared by *Zenebe Kassie*, titled: Distributed SDN Controller Architecture for Network Management and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed By the Examining Committee:

Name	Signature	Date

Advisor: Dr. Minale Ashagrie_____

Examiner: Dr. Mesfin Kifle_____

Examiner: Dr. Solomon Gizaw_____

Abstract

The ever increasing demand to use information and communication technology services trails an increase in the size of networks and their heterogeneity. As network increases in size and heterogeneity, the complexity and chance to faults and errors in a network also increases for various reasons .In addition, failure of the underlying network may happen due to link and/or core element(s) failure including the worst of catastrophic incidents that devastate the network infrastructure within the administrative perimeter. Thus, network management is challenged and network administrators will have a burden. Hence, pro-active network management system is mandatory to alleviate the burden and/or address the challenge.

This system is primarily utilized by administrators and operators. Additionally, ensuring sustainability, scalability and quality of services needs a system that considers all factors attributed for it. Thus, we have chosen an approach called software defined networking which comes with two general implementation choices for controller architecture:-centralized and distributed. The distributed architecture has again three choices: flat, hierarchical and hybrid among which we have chosen the hybrid hierarchical.

The hybrid hierarchical works both by open flow and legacy non-open flow mode of switches as the controller may delegate some flow forwarding to the underlying devices. As a solution to this challenge, we have proposed our own architecture. We have also carefully labeled and assigned a role for each and every element in the anatomical framework of the architecture to optimize the physiological make up of the underlying infrastructure functionalities. Our architecture seems to have a loop due to redundant and mesh links we have used but it is tested and verified with various node size and topologies for its viability. This test is also further conducted for core performance indicators such as TCP-throughout and UDP-latency (jitter) variation as well. The result of the tests we have conducted is very good and promising. However, we have also a challenge regarding automatic recovery to system failure and avoidance of faulty controller failure alert messages that we have planned to do on it more in the future.

In general, our effort is on the use of redundant and mesh linked controller architecture in a LAN where network service is mandatory and its interruption is super crucial.

Key Words: Catastrophe, Network Management, Controller Architecture, Latency, Performance.

Dedication

To: - The memory and spirit of my father ***Ato Kassie Yigzaw*** and my mother ***W/ro Manalebish Dinkayehu***. ..... Towards strong wisdom and attitude they had to *the most important rule in the world* ***....." Do unto others as you would have others do unto you".***

Acknowledgement

Above all thanks to the almighty GOD for his willingness and permission to successfully accomplish that he has allowed me to begin.

Next, I would like to forward my deepest gratitude for everyone's help and support throughout the preparation of this thesis. In particular, I would like to thank my advisor Dr. Minale Ashagrie for his invaluable advices he has suggested and encouragement he has made at each and every stage of the thesis which was actually a concrete effort to arrive at this final deliverable state. Many thanks also to my instructors and the whole staff of Addis Ababa University (especially CNCS) Department of Computer Science for the constant help and support they have provided to me during my stay in the department.

Finally, my special thanks go to my families and the whole staff of Ethiopian Public Health Institute (EPHI) for the chance they have given me to join to the Addis Ababa University regular postgraduate program on behalf of EPHI. Especially to the staff of the following offices: General Director, Deputy General Director, PR Directorate, HR Administration Directorate, Legal office and ICT case teams. Thanks once again to all my colleagues at EPHI – ICT case team for their willingness and motion to support and approve me as a nominee to this graduate program on behalf of EPHI-ICT case team. Thanks and thanks again also to Ato Girma Zemedu-ICT case team leader and Ato Gashaw Gebrewold- a researcher at Vaccine Directorate for their willingness and motion at the second term.

Zenebe Kassie

December, 2023

Addis Ababa

Table of Contents

List of Figures

List of Tables

List of Algorithms

Acronyms and Abbreviations

AD	Attack Detection
AM	Attack Mitigation
APL	Application Plane
BckDoC	Backup Domain Controller
BckRC	Backup Root Controller
BW	Band Width
CA	Centralized Architecture
CAPEX	Capital Expenditure
CC	Centralized Controller
CL	Control Logic
CMOV	Communication Overhead
CP	Controller Placement
CPL	Control Plane
CPP	Controller Placement Problem
CS	Controller Scalability
CSC	Centralized Single Controller
CSDNC	Centralized SDN Controller
CSDNCA	Centralized SDN Controller Architecture
DA	Distributed Architecture
DC	Distributed Controller
DCA	Distributed Controller Architecture
DoC	Domain Controller
DPL	Data Plane
DSDNA	Distributed SDN Architecture

DSDNC	Distributed SDN Controller
DSDNCA	Distributed SDN Controller Architecture
EWBAPI	East West Bound API
EWBI	East West Bound Interface
FL	Flow Rules
HHSDNCA	Hybrid Hierarchical SDN Controller Architecture
HSDNCA	Hierarchical SDN Controller Architecture
LB	Load Balancing
MC	Multi Controller
MCA	Multi Controller Architecture
MPL	Management Plane
MPLS	Multiprotocol Label Switching
NA	Network Administration
NBAPI	North Bound API
NBI	North Bound Interface
NCC	Network Control Center
NCH	Network Control Host
NM	Network Management
NMAPP	Network Management Application
NMSAPP	Network Management System Application
NME	Network Management Entity
NMP	Network Management Protocol
NMPL	Network Management policy
NMS	Network Management System

NMSG	NMS Global
NMSL	NMS Local
NMT	Network Management Tools
OPEX	Operation Expenditure
PCEP	Path Computation Element Protocol
PLSWH	Programmable Switch
RC	Root Controller
SBAPI	South Bound API
SBI	South Bound Interface
SC	Single Controller
SCA	Single Controller Architecture
SDNA	SDN Architecture
SDNC	Single SDN Controller
SDNCA	SDN Controller Architecture
SDNCS	SDN Controller Scalability
SPF	Single Point of Failure
TMN	Tele Communication Network
VNF	Virtual Network Functions

Chapter 1: Introduction

1.1 Background

Network management is a broad concept addressing the administration, operation, provision, security and maintenance of services and elements in a network. Generally, it is all about the hardware, software and users in a network. Networks are managed and operated by administrators and operators of the network. In the course of doing these jobs there are various challenges faced by the administrators and operators of a network. The challenges they have faced also varies according to the level of functionality and scale the network may have. As the size and functionality of the network increases the complexity increases and hence the chance of problems may happen in a network will also increases in different ways.

The complexity and heterogeneousness of these days' networks has increased due to the fact that there is an ever increasing connectivity, devices, and associated service demands. Consequently Network Management (NM) is becoming a challenging task to the operators and administrators of networks in organizations and institutes. To alleviate this problem, a new software architecture called Software Defined Networking (SDN) has been introduced. Since then, remarkable achievements are gained in the ease of provisioning, configuration, resource optimization and programmability to change the characteristics of the entire network [1, 2]. This architecture improves the management of networks as its single logical component called controller controls the behavior of underlying network more flexibly than the traditional approach that disperse control information at individual switches [2, 3, 4].

The SDN architecture has three basic components: the Application Plane (APL), Control Plane (CPL) and Data Plane (DPL) hierarchically from top to bottom respectively. The top layer holds various Network Management Applications (NMAPPs) where as the middle and bottom layers hold the Controller Logic (CL) and SDN enabled programmable switches respectively. There are also interfaces and protocols between the boundaries of these planes. The NBI North Bound Interface (NBI) and protocol is between the network applications and the controller where as the south bound interface (SBI) is between the controller and the programmable switches at the DPL. The SDN is also vertically divided in to two as CPL and DPL with the CPL holding NMAPPs and the CL leaving the DPL as simple forwarding

switches. This can be called as an alternative SDN visualization in terms of software and hardware implementation of it [1, 2].

The use and benefit of SDN in contemporary networks is very great in number and large in size. In general, it is for efficient, secured and simplified NM. Whenever NMAPPs at the top layer want to communicate with the switches at the bottom layer their requests should pass through the NBI. The controller then accepts these requests and directs it to the intended Programmable Switch (PLSWH) at the bottom layer via the SBI according to the pre defined CL and flow rules [2, 5, 6].

On the contrary there are also weaknesses observed from using Single SDN Controller Architecture (SSDNCA) as it is a Single Point of Failure (SPF) when disasters such as attack happened on the controller that leads to the collapse of the whole network. Additionally, Single Controller (SC) is not sufficiently enough to support scalable network in terms of communication overhead and Load Balancing (LB). For those reasons various efforts have been made by various researchers to address the problem by developing risk detection and mitigation techniques [1,4,6]. But this solution is also found to have problems and limitations regarding the reliability of detection and mitigation techniques they employed. Furthermore, the scalability constraint of Single Controller Architecture (SCA) to sufficiently support scalable network in terms of Communication Overhead (CMOV) and LB is also another challenge [7, 8, 9].

The authors in [7] have used a technique that simply assess potential vulnerability for DDoS attack and propose the general method to detect and mitigate the attack as a function of time duration and time attack patterns respectively. This is unreliable technique as this kind of attack may not be exhibited by time related events all the time. And hence they should have considered other additional methods as a function of other metrics to detect and mitigate an attack reliably with less prone to errors. The same problem and limitation is exhibited on the techniques used by the authors in [9], except the scenario is for SDN-GUARD used to detect and mitigate SDN root kits which are capable of compromising the controller. The guard uses a technique of dual view comparison to detect malicious network programming attempts.

In addition, as risk mitigation to controller attack, another more approach that logically separate the SC of SDN architecture in to a Multiple Controller (MC) one was developed. Again this is also found to have some limitations regarding scalability and LB as well as SPF risks .It is due to the fact that on a higher scale a scalable network has LB constraint as size increases. In addition, controller may fail not only by an attack but also with failures, faults and other external factors including the worst of catastrophe [10, 11, 12]. In fact, the emergence of distributed SDN controller architecture (DSDNCA) is to address the scalability and fault tolerance constraint of SCA, However, this architecture is also found to have limitations to sufficiently support network failure due to catastrophe.

Thus, this research devoted to address the stated limitation in a LAN by proposing our own new DSDNCA optimized for LAN management. In addition, designing and developing an algorithm that best suits to the proposed architecture and testing the architecture for core performance indicators is also part of this research.

1.2 Motivation

The ever increasing demand to use information and communication technology trails an increase in the size of networks and their heterogeneity. As network increases in size and heterogeneity, the complexity and chance to faults and errors in a network also increases for various reasons .In addition, failure of the underlying network may happen due to link and/or core element failure including the worst of catastrophic incidents that devastate the network infrastructure within the administrative perimeter. Thus, network management is challenged and network administrators will have a burden. Hence, there should be a proactive network management system that moves ahead or abreast of the growing demand heterogeneously. As the management of network is encouraged, the productivity and efficiency of organizations and institutes increases. This again indicates that networks are one of the great role playing tools to assure goal achievement. In addition, the management of networks is more encouraged if there is a system to pro-actively protect or recover from faults, errors and incidents to failure in a comprehensive way. This includes the cases from simple link to the worst case failures due to catastrophe: primarily seismic waves and other environmental and human made factors too.

1.3 Statement of the Problem

Management and operation of networks is a tiresome task for dynamically growing and heterogeneous networks. However, with a use of SDN architecture these tasks become better. This is due to the fact that the centralized management with the CPL is flexible and cost efficient than the traditional approach in many ways [3, 4, 5]. Even though, the Centralized SDN Controller Architecture (CSDNCA) has emerged with very great achievements to the sector as compared to the traditional one, it is also again found to have problems and limitations. One reason is its SPF nature that arouses from its Centralized Architecture (CA). The other one is its limitation to sufficiently support scalable network which has high communication overhead and LB constraints by its nature [1, 13].

Even though there is no consensus about scalability definition in literatures, ensuring scalability along with other factors is the biggest challenge of most research works on Distributed SDN Controller (DSDNC). The general SDN Controller Scalability (SDNCS) issues can be viewed with two approaches, topology and mechanisms related approaches. The former deals with the relation between topology of architectures and scalability issues and the latter deals with the mechanisms used to optimize controllers and scalability issues. In addition, each of these Controller Scalability (CS) issues has again two sub categories. Centralized Single Controller (CSC) and distributed (multi controller) approaches are for the topology related approach. In the same manner, parallelism-based optimization and CPL routing scheme-based optimization are for the mechanisms related approach. Moreover, the distributed (multi controller) approach has also again three sub categories namely: distributed flat, hierarchical and hybrid controller design approaches, [1]. But, we have selected hybrid hierarchical one.

Various researches conducted to address Centralized SDN Controller (CSDNC) problems and limitations. However, each of them lacks completeness regarding scalability as it is addressing only SPF problem. Thus using an algorithm, another more approach have been used by other researchers that logically divide SCA in to multiple areas each of which again contains several logical interacting controllers that interact with the other area controllers via border gateway switches [7, 8, 9]. But this Distributed controller solution is again found to have problems and limitations regarding the physical placement of the controllers in a single

location. It is for the reasons that failure may happen by catastrophic incidents such as seismic waves and other environmental and human made factors .Additionally, the logical division of a single controller at a single location in to multiple one has computational over head and LB compliances for sufficiently scalable network size [12, 13, 14].

Yet another more fully distributed and multi-controller architecture approaches have been developed and utilized with many improvements and advancements to various aspects and limitations of CSDNCA. However, each of them lacks to address failure of network and fast recovery issues due to catastrophe satisfactorily [13, 14]. In other words, the approach they have followed to recover and/or resume the system from failure due to catastrophic incident have trade-off in between and/or among core performance indicators such as throughput, latency (jitter) variation and response time.

Considering the worst case failure of the underlying network and fastest recovery, this research is devoted to contribute a new distributed SDN controller architecture optimized for LAN management. This is for mission critical organizations where service interruption is critical or sustainable and quality service delivery is mandatory. Additionally, designing and developing an algorithm that best suits with the proposed architecture and testing the architecture for core performance indicators such as throughput and latency is part of this thesis work.

1.4 Objectives

General Objectives

The general objective of this research is designing Distributed SDN controller architecture for LAN Management.

Specific Objective

- To review related literature in the area of SDNCA (SDN Controller Architecture) for analysis, method selection and design.
- To identify the functions and problems of existing MC SDN architectures.
- To identify functions and problems of NMAPPSs.

- To identify and analyze components of existing SDNCA (SDN Controller Architectures) for better understanding and problem resolution.
- To review DSDNCA (Distributed SDN Controller Architecture) algorithms.
- To design DSDNCA for NM.
- To test the prototype with various topologies and node size.

1.5 Methods

The following Methods will be employed to conduct this research.

- **Literature Review**

  Different literatures in the area of SDN and NMSs will be intensively examined to understand what has been known or done and what has been observed as a gap on the DSDNCA and the algorithms that best suit for it.

- **Simulation**

  Mininet and NetSim Simulation tool will be used to create links and network elements, customize them accordingly for interaction among components and analysis of interactions.

- **Prototype Development Tools**

  Floodlight and HPE VAN SDN with python development tool will be used along with NMAPPSs for Distributed Controller (DC) installation, integration and algorithms development used to analyze various sizes of data significantly affecting the efficiency of algorithms and performance of the NMS.

- **Testing**

  Testing will be conducted using various sizes and composition data running at various topologies of the proposed system to evaluate and examine the performance and efficiency of the proposed architecture and NMS.

1.6 Scope and Limitation

Even though NM in compasses MAN and WAN, we are entitled here to design, test and implement DSDNCA for NM of LAN along with best suiting algorithm. It is with due

consideration of organizations and institutions network where sustainable quality services are crucial and its interruption is critical.

1.7 Application of Results

The results of this research work will not be limited to the efficient use of NMS in LAN management. However, it can also be used as a background for the upcoming researches in the same area: MAN and WAN. Additionally, it can be used as a very good background for future researches in the related fields of the discipline such as cyber security , especially in the area of cyber security quality management as it pays due attention to technical and managerial concerns to address cyber security problems.

1.8 Organization of the Rest of Thesis

The rest of this thesis is organized in to six major chapters and the detail overview of each is as follows:

Chapter Two deals with a review in the literature of this thesis theoretical framework organized in five major sub sections as follows: the first and second Sections deals with the NM and NMS categorized in to types, elements and functions etc. The third Section deals with the general SDNC overview. Summary is the last Section next to the fourth one that deals with DSDNCA in detail.

Chapter Three is all about the related work of others in the literature in lieu with this thesis theme. It is organized in six sections including summary. Architecture and performance, NMS and SDNCA, fault tolerance, DSDNCA in LAN and WAN are the subjects covered.

Chapter Four discusses the proposed architecture for LAN management. The general architectural discussion, function, component role and interaction are the subject matters with more emphasis.

Chapter Five is all about prototype development, testing and evaluation of proposed architecture. Analysis and discussion of test results which is conducted on various test result printouts and tables is also a subject matter herein.

Conclusion, recommendation and future work, is the last chapter that conclude, recommend and give a clue to the future research work direction of this thesis.

Chapter 2: Literature Review

This Chapter is all about the theoretical framework in the literature in lieu with this thesis theme. NM. NMAPPs, SDNA and DSDNC are the general areas in the literature that takes more emphasis and weight in the organization of this Section.

2.1 Network Management

NM in organizations and institutes network is one of the key activities that play great role for the success of organization. These days, most activities of organizations and institute are conducted through the use of IT or ICT infrastructure and services. Despite the fact that there is a challenge to sufficiently grow for various reasons, there is an ever increase demand to use network services .This again leads to an increase in the complexity of NMS activities that poses a burden to the administrators and operators of network which in turn have a significant impact on the organization's goal achievement. Thus, the appropriately development of the infrastructure and utilization of the services flexibly and sustainably is mandatory [6, 15, 16].

This again calls attention to a conclusion that, the efficient utilization of resources and services is mandatory. Resources and services are efficiently utilized if there is appropriate management. Hence, management of diversified and dynamic networks with the traditional approach that deploys, configures, troubleshoot and monitor individual network elements by administrators and operators is not efficient. For this reason, the existence of new paradigm called SDN come to existence and takes the attention of business, industries, academia and research [6, 16].

The emergence of SDN has contributed various features to NM. Primarily,on the management of the underlying network behavior with a programmable CL of the controller in SDN. As pointed out in [17], determination of the causes to the performance problems by enforcing a wide range of network policies in a high-level policy language is among several to state. In fact such better management of resources and services has also its own challenge with regard to enforcement of high-level policies, as the existing APIs to program SDN are too low level [18].

The very most thing to manage a network is the understanding of the current status and behavior of that network. It can be LAN, WAN or a combination of LAN and WAN but what is needed is a comprehensive set of NMS with a data collecting and controlling tool that is tightly integrated with the HW and SW of the underlying network [19].NM can also be further described in detail as follows: Synonymous

2.1.1 Definition

There are various definitions to the term NM in literatures in different ways but with the same sense of functions. Some define it based on managerial activities and functionalities as NM and others define it based on administrative activities and functionalities as Network Administration (NA). In most literature NM and NA are used synonymously in any measure of activities, functionalities or any other except regardless emergence time.

Referring to the chronological time table in the discipline, network administration work has emerged following the convergence of telecommunications and computer networks in the late 1980's. Network administration is a set of activities performed to keep the network running smoothly and efficiently. Alternatively it is an IT function that is a priority in the context of the operation of information systems in organizations and modern enterprises [20].

Raimo kangas and Esa Markus [21], visualize NMS in to a system consisting of element management system and actual NMS. Moreover, they stated that it also includes service management and customer care in its widest context. They explicitly stated the widest definition is typically called operations support system.

All those definitions shows a definition of similar or the same activities and functionalities in different ways which leads to call attention to the identification of common behaviors and intersections so that a generalized perception is inferred to define it. Generally, it is all about the management of HW, SW and users or human elements in a network to be managed or administered.

2.1.2 Objectives

The type, focus and objective of a NM in organizations and institutes may vary in accordance with the organizations goals, objectives and IT or ICT policies. The authors in

[20] also explained the central objective of a network administration as:" *continuous operation and optimal network infrastructure as well as HW and SW system it supports*. ". Whatever objectives and goals to be achieved organizations may have, abided by rules, regulations and policies to strictly follow: "The *goal of network management is to provide the end users with a service of the expected quality*" [22]. Even though, management of networks is performed with a series of different activities, methods, steps and tools used for the operation of a network.

2.1.3 Elements

NM in the widest and advanced context can have many things as part or element in a logical and physical perspective. This may mean written policies in hard and soft, configuration information and other applications software as logical and devices like computers, routers including the buildings where major devices are concentrating. It also includes the power and back up unit as well as air conditioning systems and many more other supporting equipments and devices. This calls attention to the fact that the detail understanding of infrastructure and working procedures is mandatory specially in a dynamic , heterogeneous and ever growing network size and functionality network for obvious reasons.

A NMS holds changing SW and HW additions within the network elements in the boundary of a particular network. The SW used for these activities is resided in the computers and the network elements. Each element of a network has assigned address and label. An element can be active or in active at a given period of time for various reasons nevertheless the active elements give a status information feed back to a controller. Each node in the network contains various SW that can be used for NMS activities and it is called Network Management Entity (NME) [19]. The look in to NMS architecture in Figure 2.1 and the interactions among various components discussed highlights that NMS design to an organization network needs careful planning with a consideration of several detail and comprehensive studies and surveys as an input.

2.1.4 Functions

The earliest network management functionalities definition was by the Tele Management forum and ITU-T. These functionalities of classical management system are formed from

the so-called FCAPS function which is abbreviated for fault, configuration, accounting, performance and security management [20]. The term classic management is used to signify the centralized management solutions which are well known for their higher complexity. The latest definition by ISO is also the same giving more emphasis by grouping five tasks as FACPS [20, 23, 24].

Slawomir KUKLINSKI and prosper CHERMOUIL, [23] summarized the general SDN management functionalities in to two and presented the detail as follows. The first part is all about FACPS functionalities in relation to classical operations. The second part is dealing with the specialized SDN specific management duties in relation to switch and controller of a particular architecture selected. Open flow configuration and management protocol by the development of ONF (open network foundation) is applied in the configuration of open flow switches and the controllers. However, SDN management operations can also be supported by legacy protocols of OSI model such as SNMP and NETCONF even if they have their own limitations. In addition they have explained too much on the analysis of network management capabilities of open flow. Specially they have tried to address anomalies of SDN management and suggested future research directions by elaborating functionalities of SDN in lieu with FACPS in five categories.

2.1.5 Network Management Tools (NMT)

NMTs are used to help the management of networks in various ways throughout the development and operation of networks [20]. It is applied staring from early installation and configuration of systems and devices to latter at operation in the provisioning of services and performance tuning of the whole network. The performance tuning, which is real time monitoring and controlling technique of fault or abnormal situation correction and/or avoidance is a subject of these tools that takes more attention here. It is in relation to prevention and control of degraded quality of services. Delvi Chadha, [19] explains this situation alternatively as *"any fault or underperformance of a network can cause a loss of large volume of data"*. Thus, it has very great impact on the profitability of the business or goal achievement of organizations and institutes with service delivery.

The monitoring and controlling is done at various stages with the aid of various features of this tools used for traffic analysis, packet inspection, incident alert and pattern recordings. In fully automated systems alerts are at “real-time” or “near real-time” and activities such as traffic analysis and packet inspection are done by the system. However, in partially automated systems and in non automated systems, there is an alert about fault to operators and administrators for appropriate manual packet inspection and traffic analysis accordingly.

Inspection and analysis are conducted with various metrics for variable or parameter of functions used to measure various performance aspects. Metrics are variables or parameters related to speed, accuracy, availability, error rate and/or volume of workload etc. and monitoring is the continuous control activity of a network by measuring the states of devices parameters. Performance analysis, performance metrics, monitoring methods and monitoring tools are the core concepts to be stated about network management tools among several of it [20].

The performance monitoring and measurement are done by IT or ICT personnel, primarily by system and network administrators. In this activity Metrics, methods and tools are selected in accordance with the network architecture, topology, configuration and other characteristics of the network and working procedures to be examined accordingly. This is just to assure the accuracy of the desired result measurement and analysis. The depiction is with various forms such as graphs, charts and tables. This depends on the respective personnel preference to output format and capability of tools, methods and metrics selected. A typical of this tools demonstrations are as follows:

Cacti: Cacti is an open architecture web based GUI monitoring tool developed with PHP and has a capability of studying the parameters of running services on the network. It is used to monitor status of elements, applications including BW and CPU utilization with the help of SNMP protocol as a tool of information retrieval and result depiction.

Nagios: Is another open source monitoring tool that allows monitoring and alerting of service and protocols for servers, switches and applications etc. such as: HTTP, FTP, SSH, PING, SMTP or POP3. Additionally, it has also a feature of alerting users twice i.e. early when problems happen and latter when it is resolved.

Ntop: Is also another GUI tool used for management and visualization of information that helps to analyze network status.

Danish Rafique and Luis Revasco, [25] have given a profound explanation to network management data details and generalizations as follows: most of network management data is primarily from two different sources. The first one is from configuration, topology and connectivity at various layers of the network and the second is from monitored network using SNMP. Based on type and forms these network data can also be further classified in to static, dynamic, text and multi-dimensional. Besides their type classification monitoring tools also vary in purpose: some are suitable for classic NMS, others for SDN based and yet another one for autonomic environment. In addition, some are good at monitoring HW, others SW and yet another ones suitable for a hybrid of them, etc. But, again all of them have similarities. Thus, it is also a very good idea to take a survey of the network infrastructure architecture elements, topology and services provided by the network before specifying a monitoring tool suitable for a particular network. Alternatively, it can be done by the guidance of the most knowledgeable IT or ICT personnel related to the network, system and services.

2.2 NMS

NMS is a collection of integrated monitoring and controlling tools highly optimized for two core features of managerial activities: the use of single operator interface with powerful user-friendly set of commands and minimal amount of independent HW and SW tool incorporated in to the existing user devices [24].Various paradigms have been proposed including the legacy classic NMS that significantly relies on the role of human operators. Each of them has its own feature that can be listed as pros and cons. For instance, if we take the automated NMS which is optimized for plug-and-play solutions and operations, it has better feature on availability and reduced OPEX which is a big deal in NMS. But, our interest here is on the SDN approach which has emerged with a feature of programmability and reduced CAPEX due to lower cost of switches.

2.2.1 Network Management System Architecture

The term architecture in relation to NMS is just about the components, their layout, organization, functions and interactions with administrative tools and other elements in the network which is called the administrative architecture. In lieu with these concepts network architecture can be broadly visualized and interpreted with three perspectives as network management, SNMP and TMN architectures [20].

The third category of architecture is about telecommunications networks with additional three sub architectures namely: physical, functional and informational sub-architectures and it is not in our scope to deal with here more. The second architecture is about simple network management protocol architecture which will be discussed separately as SNMP at Section 2.2.2. The first one which is called network management architecture is about LAN management architecture and it is our interest that we will deal here more in this section as it is the core concept with respect to our research theme. The network management architecture is a core concept in the network design premise of organizations and it is one of the key indicators to reliability, availability and security of a network. As pointed out by Devi Chadha, [19] *"A well designed NMS is very important to ensure smooth and reliable operation of a network. It also improves efficiency and reduces OPEX of the network"*.

A network management system is designed to have a holistic view of the network as a unified architecture. This is achieved as each: point, specific attributes of an element and links to the system in the architecture have assigned addresses and labels. Based on the suggested architecture of NMS at Figure 2.1 [24], each node of a network has a collection of software loaded on it and used for NMS activities. Each node is also referred in the diagram as a Network Management Entity (NME) and performs the tasks such as: collect statistics on communication and network-related activities, store this statistics locally, and respond to controller center and send message to NCC when there is a significant change locally [24].

In this architecture, at least one host in the network is designed as the network controller or manager. The manager, besides NME, has other collection of SW called NMAPP that is used to reply to user commands by displaying information and/or issuing commands to NMEs. Communication is also allowed using an application-level NMP (Network Management Protocol). For the purpose of network management NME is referred as an

agent. An agent can be user application supporting end systems and communication devices such as clusters, controllers, bridges and routers as well. The network controller host communicates and controls NMEs in other system. Redundant controllers are used to ensure high availability of the management functions. This is by the use of one active controller that controls centrally and the other simply idle collecting statistics info or can be used as a backup controller if the primary one fails.

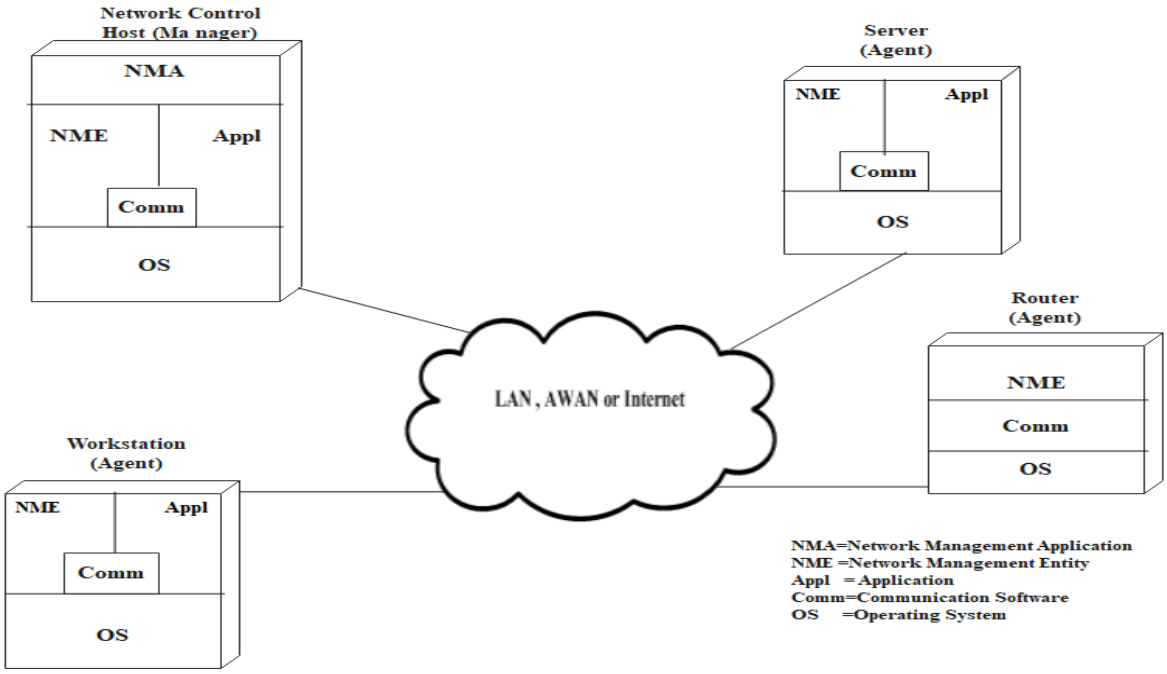

Figure 2.1 [24] NMS Architecture

2.2.2 Simple Network Management Protocol (SNMP)

Communication between network elements is conducted with the aid of NMP and services. Among several of them standardized and widely used SW for network management is SNMP, Communication Management Information Protocol (CMIP), Simple Gateway Management Protocol (SGMP) and Remote Network Monitoring (RMON) are also used seldom.

The SNMP architecture has four major parts: Management station or manger, agent, MIB (Management Information Base) and the network management protocol which is SNMP. The management station is a standalone device or it can be on a shared network system. At minimum, it can also have features such as management applications, user or operators

interface, translation of managerial requirements in to actual monitoring and remote control as well as a database of network management information extracted from all entities in the network. The agents in the platform such as hosts, bridges and routers have agent SW installed that allows them to be managed from the station. Additionally, they respond to requests of information from the station and responds also to actions from station. In the architectural platform, each resource is represented as an object and essentially it can be a data variable that represents a single aspect of the management agent. The collection of objects is referred to as Management Information Base (MIB) and also agents and station are linked by protocols [20, 24].

Currently the SNMP has three versions:- **SNMPv1, SNMPv2 and SNMPv3** chronologically from the oldest to the recent respectively with security and other major updates incrementally. It is used for the management of TCP/IP networks; however, the management of TCP/IP and OSI based networks is with an enhanced version of it called SNMPv2. Each of these versions of it has also key command capabilities such as Get Set and Notify.

2.2.3 Network Management System Applications (NMSAPPs)

In general, NMSAPP is a SW used to support the NM activities. Some are used to support managerial activities of SW system in a network, others used for HW and yet others for a hybrid of them.

A very good example of such applications is a network management platform deployed in network with heterogeneous elements. These platforms receive and processes events from various network elements such as servers, switches. routers, access points etc. They are also treated as a main panel for network operators to detect faults in the infrastructure as they have capability to quickly detect problems and graphically display the operational status of elements. A standardized platform of such kind have common functionalities such as: network discovery, topology mapping of network elements, event handler, data collector and graphically presenter and management browser [26].

In the SDN based approach, NMAPPs that are used in traffic engineering, network monitors and fire walls are at the application layer. The forwarding devices behavior at the data plane

is defined by the control logic that is implemented by network applications installed on the infrastructure plane [27].

2.3 The SDN Architecture

The early SDN architecture is centralized by its nature and has three planes clearly discussed in detail at Section1.1. This architecture helped the management of networks efficiently for limited size of networks. It is because the vendor-independent approaches that come with SDN vertically integrate management plane, single controller and devices at the data plane. It is also clear that management of SDN based networks is a collective some of managerial decision at the three layers. For those reasons, designing or choosing MC architecture of SDN based networks considering scalability, performance and catastrophe is a challenging task that needs careful attention.

The emergence of SDN has come with various flexible features in the area of NM and design, among several of them is the introduction of NOS (Network Operating System). There are several companies and organizations working on the adoption of SDN and formulation of standards. ONF (Open Network Foundation) is one of those organizations that standardize the first south bound interface called OpenFlow that allows communication between the control plane and devices at the data plane.

According to this non-profit organization, SDN is also defined as "*The physical separation of the network control plane from the forwarding plane; and where a control plane controls several devices*". Additionally, it is stated that the core idea of SDN is separation of control function from devices, with the control function centralized in a SW called SDN controller. In general, the idea of SDN is the result of efforts to advance computer networks programmability. Which is to say that, an independent operation of the network HW at the bottom layer, leaving the upper layer for the functionalities of the behavior of networks and control functions that are performed by SW. This helps to simplify network administration and management tasks as there is an option to reprogram the network infrastructure instead of trying to re-build it tediously [27].

2.3.1 Protocols

Standards are a common agreed framework upon which products such as HW are manufactured and SW is developed such that the use, adoption and communication among multi vendor products are smoothed. Protocols here have also the same role as OSI reference protocols at layer 2, 3 and 4 due to the fact that SDN is working with packet headers. OpenFlow, developed by ONF is one of the widely used protocols that provide a standard traffic management services with a description of how a controller communicates with the networking devices. Forwarding and Control Element Separation Standardized by IETF is also another south bound interface to be stated. OpenFlow has two basic logical components: a flow table defining packet processing and forwarding in a network and an OpenFlow interface in between the controller and the devices at the data plane called Application Program Interface (API).

The OpenFlow interface is also termed as the SBI and also NBI or API is the analogy for the interface in between the application plane at the top and the controller in the middle. Eastbound/Westbound API such as HyperFlow is also another type of interface that allows information exchange and communication among multiple controllers in relation to control information of the flow in the data plane. In addition, we will have a detail explanation of each of these APIs latter at the end of this section. The deployment of South Bound API communication can also be done considering two situations: One could be considering the traffic between the controller and network devices abide by the flow rule stated called *in-band communication (ibc)*. The other is without the enforcement of such rules but with the isolation of traffic flow from communications by VLAN implementation that depends on Openflow rules called *out-band communication (obc)* [28].

There are three modes of communication in OpenFlow protocol: controller-to-switch, asynchronous and symmetric communication. The controller-to-switch communication is used in the establishment of handshake and flow table configuration. In asynchronous communication, communication is established by OpenFlow compliant switch by sending packet-in messages to inform about port status and flow-removed messages where as in the symmetric one, either of them can initiate the communication and exchange of message. It has also to be noted that the very important three things of OpenFlow enabled switch in an

SDN network of OpenFlow protocol are: flow table, secure channel and OpenFlow protocol. Each OpenFlow enabled switch has also many tables and each of these tables is again equipped with a number of entries. An entry has also again three parts: a *rule* field used to define the flow entry in terms of packet headers such as source and destination addresses, an *action field* used to apply packet matching comparison on the values in the rule field and a *status* field used to update counters in the entries [1].

Northbound API (NBAPI): As an interface between the applications and the controller or NOS, this API allows the operator to utilize or program the devices at the bottom plane for tasks such as traffic engineering, QOS, topology discovery, LB, security policy enforcement, firewall and delay management etc.

East-Westbound API (EWBAPI): it is used as an interface to controllers in different network domains. But it can also be used as an interface to core network elements such as servers to controller or server to server which are also in different domains. Besides the key role of this interface to support large network size, it is also applicable for the global view of the network that needs mesh connectivity among heterogeneous NOS.

Southbound API (SBAPI): As an interface between the controller and devices at the data plane programmability using OpenFlow through this API is not easy. It is due to the fact that this protocol is closely tied to the hardware at the data plane and also the incorporation of policies is in bitwise fashion.

2.3.2 Benefits

The very core part of SDN is the controller which has controller logic and other network functionalities. The controller logic due to its programmability nature has come with features to modify network policies which are a less prone to error and supporting flexible network operation and management paradigms. Additionally, it is capable of control and global view of the underlying network infrastructure due to its centralized nature. Depending on the network design and architecture, the controller has also various forms such as centralized, distributed and hierarchical which in turn have various benefits that stem from the specific design and architecture selected.

The benefits of SDN in this topic are just the benefit of early and CSDNCA as compared to the classic or traditional one. These benefits are large in number and size; however, the most important and frequent of it that we found in literature are: programmability, centralized control and flexibility etc. In literature, it is also observed that there are various ways of explaining the benefits of SDN from various perspectives. Emilia Rosa Jimson et al, [27] indicated that this architecture has come with the benefits that are used to overcome the legacy approach. Specifically: centralized flow management, network programmability, simplification of network hardware task, simplification of network configuration, forwarding rule dynamic updating and flow abstraction. However; the authors in [29] indicated that, network programmability is not a new concept as there were efforts in the area of network programmability prior to SDN emergence but not as efficient as the SDN approach due to several reasons.

2.3.3 Operation Mode

The SDN controller has many features, among several of them the one with a key role is the mode at which the controller operates in the setup of flow rules and its relation to the protocols that we have dealt at section 2.3.1. Murat Karakus and Arjan Durresi,[1] classify the operation mode of a controller in to three modes as: Reactive, Proactive and Hybrid each of them having role and functionalities synonymous to their dictionary meaning.

2.3.4 Scalability

The general SDN challenges and limitations are indicated by many factors that will be pointed out at section 2.3.5 but we discuss here only the scalability constraint as it has key role and relation to the theme of this thesis in many ways. Scalability constraint in SDN is the result of scalability at all the three planes but primarily it lies at the second and third planes which is at controllers and switches respectively. The control plane scalability is diagrammatically represented with their sub categories at figure 2.2 [1].The controller scalability problem can also be the result of several factors. Jacob H.Cox et al, [30] suggested that it arouses from three major causes: latency due to controller interaction to multiple nodes, communication methods in between peer, supporting and slave controllers via EWBAPIs and lastly size and operation of the controller's back end database. The

authors also have hoped that the first and third constraints will get more attention by the research in the arena and suggested that physical distribution of controllers including network wide view as a solution to the second constraint.

Scalability is a broad concept claimed by many systems. Even thought there is no consensus so far on its definition in literature, it is a key property of a system that has positive constructs regarding system performance. In our case, the very core issue of controller plane scalability performance should be evaluated considering two metrics: *controller plane throughput* and *flow setup latency*. Where the first refers to the number of flow requests handled per second and the second to the delay to respond to flow requests. In addition, solution proposed for scalability problem may introduce trade-offs that have significant impact on the other behaviors of a network. For those reasons scalability constraint is not an independent problem that can be dealt exclusively but rather with a holistic view of other factors that may introduce trade-offs. Even though there is such controversy about this concept, there are also several efforts by researchers to propose a metric for scalability measure. Most of these metrics considers a homogenous system and focus on two types of scalability: *Isospeed* and *Isoefficiency* scalability [1].

Isospeed Scalability: It is a metric for scalability of an algorithm-machine combination in homogenous systems. In this scalability, an achieved average unit speed of an algorithm on a given machine can remain constant with increasing number of processors and problem size for an algorithm-machine combination. Mathematically it can be represented by a function as: $\Psi(p, p') = p'w / pw'$ (1)

Where, p and p' represent the initial and scaled number of processors of the system respectively, and w and w' are also again an initial and scaled problem size (work load) respectively.

Isoefficiency Scalability: With regard to this metric, scalability is defined as the ability of parallel machine to keep the parallel efficiency constant when the system and problem size increases. This parallel efficiency is formulated as speedup over the number of processors, $E=s/p$ (2)

Where, speedup is calculated as the ratio of problem size (w) and parallel execution time.

There are also similar efforts to define scalability for heterogeneous environment inheriting from homogenous and hybrid models. But, our interest here is on the control plane scalability. There are recent two efforts to quantify the metric for control plane scalability .The first is based on productivity of distributed systems adapted to the SDN case and the second by the ratio of work load and overhead. In general, the SDN scalability is the result of many factors at all the three planes; however, the control plane is a scalability bottleneck in SDN for three key reasons. The first is for its focal point behavior of core activities, the second is for the quantity of events/requests handled by a controller and lastly inter-controller latency related to the distance between devices and controller that introduces flow setup latency which is determined by packet processing time.

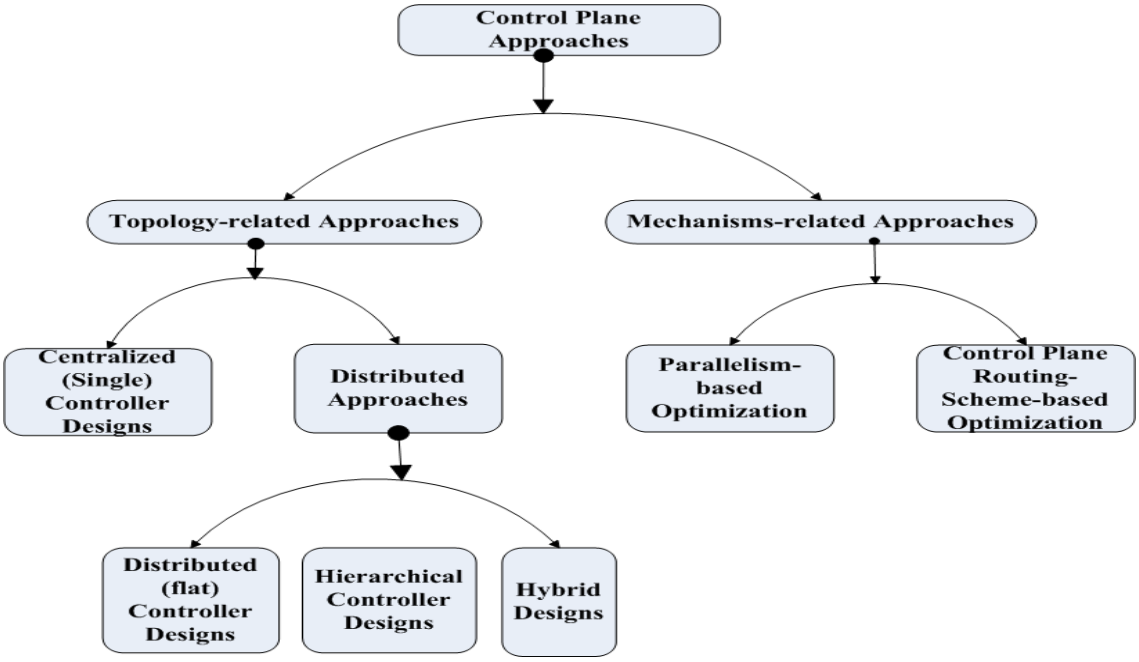

Figure 2.2 [1] Taxonomy of Control Plane Approaches in SDN

2.3.5 Challenges and Limitations

The limitations or challenges of SDN are diverse and very large in number. It is because the term SDN by itself is a broad concept addressing many issues in the context of network design, architecture, management and service provisioning etc. Identifying and listing these challenges could mean identifying factors that affect the performance and efficiency of networks directly or indirectly. In other words it is just the task of identifying parameters

and metrics that account for the general expected quality of network service. As pointed by [28], key indicators to these challenges are related to scalability, reliability, availability, elasticity, security, performance, resilience and dependability. We pay due attention for these factors to have a good knowledge of constraints and problems that leads to formulation, development and deployment of better solution design approach.

The emergence of SDN despite its numerous contributions has also several other challenges and limitations; one could be with APIs regarding the adaptation of more flexible features that support the management of networks. Especially a significant limitation is observed in NBAPIs as there is no clear programming language identified so far. Additionally, poor documentation, complexity unlike ease of use and unavailability of training to operators are the other anomalies about the use of this API. The EWBAPIs has also some anomalies and even it is not incorporated with some controller versions as well. In fact the SBAPIs have an OpenFlow as a de facto standard and others competing for it. But it is also incomplete at issues such as lack of mechanisms to manage devices, poor status info update and the absence of specific method of communication between controller and switches etc. [30].

The authors of [1], discuss the SDN challenges by taking due attention to the SDN control plane challenges along with their proposals. These challenges are summarized as follows: Controller(s) failure, state/policy distribution/consistency, flow-rule setup latency and controller placement. The authors in [28, 30] also expressed these challenges in a slightly different ways but in one way or the other related to the eight key SDN challenge indicators listed at [28]. Additionally, most of them agree in the idea that the general SDN challenge means almost the SDN controller challenge as the controller is the brain of SDN. This shows that the SDN challenge identification and solution development process is large and complex task that calls attention to many research themes visualizing this problem at different angles and directions.

2.4 DSDNC

The most important thing that comes with the DSDNCs is the physical distribution of controllers with logically centralized offerings. It is just for the purposes of scalability, availability and robustness with collective measures that ensures recovery and tolerance of

the controller failure. Latency due to controller's communication, spoofed messages, inconsistent update and problems due to routing changes are the common issues raised with it. The SDN controller consists of two main components: functional component and control logic. Multicasting of event information among controllers has an impact on the performance of controllers. Additionally, malicious controllers can affect the controller's view of the network which is defended by cryptographic signatures that ensures the agreement of the other controllers. But still this method is unreliable or vulnerable when information to the controller is delayed, prevented or some network elements are subjected to replay attacks. When new controllers join to the network, an XOR hash function is operated by controller to validate the history and state information received from the switches [30].

2.4.1 Types

The topology models and architectures for DCs (Distributed Controllers) in SDN are distributed (Flat), hierarchical and hybrid controller designs as shown at the bottom of Figure 2.2 hierarchy model. In the distributed design, each controller manages its own sub-network of the entire network but network view depends on either of the local or global view choices. On the contrary the hierarchical approach allows global view only to root with some limited local events leaving the local controllers to manage frequent local events. The hybrid on the other hand takes the advantage of both but with a distinguishing feature that allows DPL elements partially to get involved on decision process and network control.

2.4.2 Failure Recovery

Scalability, reliability, inter-operability and fault tolerance have remained the continuing challenge of SDN even though the positive aspect of this approach is its centralized view nature and flexibility via programming, The SDN allows multiple and DCs to be connected such that one controller acts as a backup in case a failure happened on the other. MCs are also used to share the burden of a single controller congested by flows. Additionally, these controllers can be used to reduce latency, improve availability and increase scalability as well as fault tolerance. The DC clustering is one of the techniques to tolerate a fault by the

use of MCs in reactive and pro active mode with an equally distributed role; however, this doesn't mean there is no primary and secondary controller scheme.

In literatures, the fault-tolerance is interpreted and categorized in different ways. The authors of [31], stated that it is an aspect that should be addressed at all the three planes and its architecture can be categorized in to two as proactive and reactive approaches based on its alternate path implementation. Additionally, they have classified the controller plane fault-tolerance architecture in to two as master-slave and slave-slave. Muhammad Reazul Haque et al [32], have addressed the fault- tolerance problem by using multiple smart back up controllers than multiple DCs which are costly relatively. It is done with the aid of proposed novel Integer Linear Programming (ILP) model that helps to decide the placement of single or multiple smart backup controllers.

2.4.3 Controller Placement (CP)

The CP is one of the key concepts in the design of network and architecture selection to ensure a service of expected quality. The placement of multiple and DCs have various benefits; however, the interaction among those controllers may have also its own draw backs that significantly affect performance. One problem could be difference and inconsistency among controllers. Thus the CP is used to keep consistency among different and distributed MCs. The location and number of controllers play a key role in the performance and reliability of the network which have also tradeoffs [33].

In literatures the classification and analysis of CP approaches in SDN are categorized based on various aspects of the network such as: environment and medium, controller capacity, traffic characteristics, reliability of network elements, solution approach and performance metrics. This CP strategy can also be uncapacitated or capacitated based on the controller capacity. A strategy is capacitated if and only if the assumption by the controller placement strategy to the controller's capacity is limited and uncapacitated otherwise. Static or dynamic is another categorization of a CPP based on network traffic characteristics. Another strategy called reliability aware CP considers failure of nodes, controllers or links in to account in the deployment. Additionally, the CPP in SDN can be further classified as:

latency aware, connectivity aware, cost aware, energy aware, QOS aware, and CPL placement overhead aware depending on the objective of the problem [34].

Various CP approaches have come and proposed to address the issues so far but found to have some limitations. A reliability-aware CP approach was proposed but it has limitations regarding latency incurred. Another approach was also proposed using a CC but it is also found to have limitations to sufficiently address scalability of dynamic networks. Yet another more approach was proposed for resilience by using a min-cut algorithm that partitions a network and this is also again exhibited by its capability to insufficiently support MC environment. Many more other approaches have been proposed based on some metrics and factors. Such as: optimization problem, heuristic, salience-based, game and graph theory etc. [33, 35].

Some CP approaches are also again mathematically formulated, quantified and evaluated, for instance the salience based approach, [35] shows its effectiveness in shortening the mean and worst path length, while keeping the load maximum as: The salience value $(S_{i,j}, 0 \leq S_{i,j<1})$ of the link connecting node i and j is defined as

$$S_{i,j} = 1/N \sum_{X} \mathbf{T}_{i,j}(X), \quad (3)$$

Where N denotes the number of nodes, $\mathbf{T}_{i,j}(X) (\in 0,1)$ represents if the link from i to j is included in the shortest path tree from x, or not.

The meta-heuristic technique, [36] is mathematically modeled and formulated. The basic idea of this technique is CP considering complexity of wide area SDN using an efficient generic algorithm that leads to efficient solution. At this approach, for controllers p in a network of n nodes where p << n, given n switches at a certain locations, there will be optimized placement choosing p controllers and their locations. By doing so, reduced propagation latency between controllers is achieved while reducing the total cost of the network. The DDoS-attack aware smart backup controller placement in SDN, [32] is also mathematically modeled by a function called objective function. This function is formulated from various notation sets of the model, constants, decision variables of the model under DDoS attack and cost functions. In this model the number of required smart backup

controller depends on the availability of network requirements and the probability of frequency of DDoS attack on the controller.

2.4.4 Algorithms

In an SDN based network, provision of expected quality service is the result of several factors that account for the overall performance of the network. One of such factors is the use of efficient algorithms at each and every processes and functions in the network. Practically algorithms are also applied at all the three planes at various functionalities. Some work in the control logic performance optimization at the controller plane or optimization of CPP at the same layer. The others may work for in intra and inter switch communication latency reduction and performance improvement at the data plane. Some others may work to support network management in the optimization of global view and performance analysis at the application plane. However, we mean the algorithms at the second layer when we talk about algorithms in the SDN based network due to the fact that programmability in SDN is allowed only at the controller. The general algorithm design and selection here is in relation to the approaches used to address the eight key performance indictor metrics in SDN that are listed at Section 2.3.5

The authors of [37], categorized and discussed CP problem in to six areas. Each approach is also again summarized in seven tables concisely in terms of method or algorithm type, implementation detail, evaluation metrics and network type. The six approaches of CPP are based on: brute-force and cluster, linear and quadratic programming, evolution, heuristic and greedy algorithm, simulation annealing algorithm and other CPP strategy approaches. The seven tables starting from table one are also summarized in the CPP strategies of: cluster based, linear and quadratic programming, bio-inspired, genetic algorithm based, heuristic and greedy algorithm based, simulated annealing based and other algorithm based strategies.

Kushboo Kanodia et al [38], also proposes an algorithm called meta-heuristic hybrid particle swarm optimization and simulation annealing, which is proposed for resilient controller placement that is used to minimize the average latency. This hybrid approach deals with a failure of a node or a link. Jie Lu et al [39], with a survey of CPP have also done an overview in the literature and have shown a profound mathematical formulation and

quantification along with the analysis of several possible proposed algorithms. Among several of algorithms is the survivor algorithm that works on connectivity, capacity and recovery to improve reliability and many more others such as bully, bargaining game, multi-start hybrid non-dominated storing genetic algorithm, multi ccriteria decision algorithms and adaptive bacterial foraging optimization etc. Rong Chai et al [40] , have approached CPP in different way by giving due attention to control plane delay primarily composed of transmission and processing delay which is between controllers and switches and inter controller delay. For this approach an algorithm called control plane delay minimization-based capacitated algorithm have been introduced.

Most algorithms we have discussed so far are in the area of CPP it is due to the fact that it greatly deals with is this thesis theme. But, we have also addressed other aspects, for instance the meta-heuristic algorithm [37, 38, 39], addresses cost optimization and minimization of propagation latency while it is used for CPP. The k-critical, K-center (brute-force), Integer Linear Programming (ILP), Linear Programming (LP), SSO with chaotic maps, genetic algorithm (AHP or BIP) and the greedy algorithm etc have also multiple other roles of such kind while addressing the CPP. The same is true for HC-BC algorithm [41] , on minimization of latency and optimization of LB while addressing CPP and the link-load balancing algorithm based on ant colony optimization (LLBACO) [42], for link LB , delay and shortest path. However, there are also various models with alternate algorithms to comply with specific problem context.

2.5 Summary

Theoretically CPP is mathematically modeled by graph theory, G= (V, E, U) topology where V is the set of n switches, E is the set of edges (physical links) among switches or controller and U is the set of K controllers. This is based on a probability of failure and has targeted two objectives: Minimization of cost and latency and maximization of resilience and reliability. In general, this formulation is an optimization problem that minimizes or maximizes some aspects of the network [37].

Physically centralized, physically distributed, logically centralized, logically distributed, flat architecture, hierarchical architecture, static architecture and dynamic architecture are

concepts with a paramount importance to the architectures of a MC SDN. They are all core concepts each of them specially adopted and designed for a particular or all features of the concepts in logical or physical distribution and global or local view of the network [43].

Chapter 3: Related work

3.1 Overview

In this research we have reviewed related work of many papers and have presented it in seven sections. The organization of this Section is as follows: the first part is about the related work in view of DSDNCA performance test addressing node size and topology used. The discussion and critical evaluation on algorithms, CPP and fault tolerance or recovery is the third part that follows the discussing on LB, algorithms and NMSs as a second part. SDN Controller Architectures in LAN and SDN controller architectures in WAN are the fourth and fifth parts respectively. Lastly, summary of this Section of the thesis is the last part next to the short summary of the related work in table form as a six part.

.We have also critically evaluated 18 papers at the first five sections in terms of strength, weakness and limitation separately. It is done with detail explanation of the methods and procedures followed findings, critical visualization and evaluation as well.

3.2 SDN Architectures and Performance Test

The authors of [8], have worked on performance test related to controller failure with a hybrid architecture and topology formed from physical SC and logical MC. They used a technique called lightweight and scalable MC SDN management. It is done by introducing an algorithm that divides a single controller of the previous architecture in to several logical areas. According to their design each of these areas again contains several logical communicating controllers which in turn communicate with controllers in another area via border gate way switches. Using this algorithm they tried to resolve high CMOV (Communication Overhead) and long convergence time across networks associated with scalability. The overall implementation and testing was done on a topology formed from 20 uniform SDN controllers distributed randomly in the network. The result of the experiment they have conducted shows an approximate 50% reduction of communication overhead and 50% faster convergence time as compared to the usual broadcast-based MC SDN operations. The result of their test was evaluated with four parameters: number of messages exchanged, convergence time, link failure recovery time and performance of partitioning on various topologies. The weakness of their work is that they used single physical but multiple logical

controllers at a single location for sufficiently scalable network size. For this reason their approach lacks to consider failure of controller due to catastrophic events. Additionally their fast convergence time result for message exchange among multiple areas is not clearly evaluated in comparison to physically DA convergence time.

The authors of [44] have worked on hierarchical Distributed SDN controller model that is aimed to reduce the burden of root controller. For this they have avoided intra-domain information exchange among subordinate controllers and allowed only inter-domain exchange to compare with the classic hierarchical models which uses the root to convey both messages. To implement this approach they have introduced a mechanism of global view sharing that eliminates unnecessary query to root. This mechanism defines inter-port and intra-port functionalities to edge switches which are at the border of domains and have observed and analyzed the behavior of LLDP and ARP packets received by the root. The analysis result showed that this mechanism helps the root not to handle unnecessary packets. In addition, they have also compared the performance of various controllers in table form. Even though, the paper has such good merits it lacks to address node size and heterogeneity which can be taken as a limitation.

The authors of [45] have worked on evaluation and comparison of the performance of utilizing multiple controllers in SDN based networks. Their work is on the comparison of four controllers (ONOS, Open Daylight, POX, and Ryu) with two successive tests. The first is conducted deploying linear topology with different number of switches or hosts that are connected to two controllers of each one of the four controllers. In the second test, different numbers of controllers are connected to a linear topology with fixed 64 switches. For emulation of the experimental set up they have selected virtualization tool to host ubuntu on windows 8 and conducted a test by measuring Throughput and Jitter. The result of the experiment have showed that repeated commands for ping, Iperf TCP connection, and Iperf UDP connection with different packer size and time interval, TCP window size, and UDP buffer size parameters degrades the performance of the controllers as there is a burden on it. In addition they have concluded the analysis of their test result as: POX controller shows better results in having constant low average RTT, high throughput, and low jitter values

with extra more durability and flexibility. Missing inclusiveness to conduct the test with other topologies different from linear can be the limitation of their work.

The authors of [32] have approached recovery to controller failure due to DDoS attack in different way. The approach they have used based on ILP model proposes a multiple smart back up controller placement strategy to ensure the SDN operation against DDoS attack. By doing so, they stated that the cost of multiple ordinary controllers is saved due to a shared approach used for various aspects of the network. The mathematical model formulated from various notations, constants and decision variables used to minimize cost of multiple SDN by replacing it using smart backup controllers is one of the outstanding achievements. Additionally, the probabilistic mathematical model for the placement of smart backup controllers, the mathematical programming language based testing and analytical analysis of results is the other strength to state. On the contrary, the paper deals in detail on the reduction of CAPEX but hasn't dealt with OPEX which is the most important thing in SDN based network. This can be taken as the limitation or weakness of the paper.

3.3 NMS and LB of SDN Controller

The work of the authors at [46] is on the network management framework of SDN to build the analogous FACPS (fault, configuration, accounting, performance and security) model for the purpose of a new model that they have proposed. This new model they have proposed is the management framework that serves SDN orchestration with a categorization of network functionalities as a guideline to implement SDN management. The method they have used is survey techniques used for network management of SDN based networks and emulation of proposed architecture for test. In addition, they have built map of management functions to build management scheme of SDN with a new architecture. This proposed architecture is emulated with MATLAB development tool, tested and analyzed by statistical approaches for performance. It is all done using the five FACPS elements as a parameter. The result of their experiment shows that the proposed model has a 27.5% increase in performance of network operation with variable application requests over time as compared to two other models they have used for the demonstration purpose. These two models are controller-only and manager-only models. The identification of non-critical and critical functions in the orchestration of the new proposed model along with its functionalities definition are the

strengths of this paper to be stated. However; the paper still lacks to have an expected creditability to the testing and emulation procedures and processes they have used to conduct the test.

The work of authors in [2] is on simplification of management tasks based on Route Flow, a framework that enables IP routing and virtualization in Open Flow Networks. To simplify management tasks such as maintenance they extend the frame work to allow live migration of virtual topologies to new physical infrastructure. They use Flow Map in their network to allow changes in the physical topology without affecting the logical topology. Due to the fact that the control and data plane are tightly coupled and vertically integrated in the same hardware, changes made to the data plane needs analogous modification to the CPL. In the CPL, an IP layer topology (the so-called virtual topology) with in a virtualized environment is used to control the forwarding behavior of the physical network. Their evaluation of proposed system in emulated environments shows that the proposed system is capable of performing live migration operation with a slight impact on network traffic.

The authors in [14] try to address scalability and failure recovery problems considering the worst case catastrophic controller failure due to seismic wave events and hurricane. Their work bases the recent researches in literature in the area of DC primarily on the controller load reduction using distributed placement and robustness due to information sharing among controllers. It is justified that coordinating functionality among controllers is the biggest challenge for optimized implementation of DC. A hierarchical architecture have been used. According to the proposal the nodes on the affected area should recover the CPL architecture in catastrophic events. It was also stated that nodes have predetermined priority rank and a node with the highest rank will take a substitute controller role when failure happens. Immediately after recovery two or more controllers happened in a network and conflicts that disrupt the network happened due to mutual control commands to control a switch. To resolve the problem a substitute controller negotiates to others and merges their planes in to one after discovering a recovery. The whole reconstruction of the CPL will be completed within two minutes.

Even though this approach is not complete to address failure of networks due to the worst case environmental factors, it shades lights to better and more alternate approaches better

than papers we have discovered so far in literatures in lieu with this challenge. It is also is actually indicated in the paper that a hybrid approach is better and is a planned a future work direction of the paper. This is actually the good merit of the works of this paper. On the other hand, computational complexity, convergence time and optimum algorithmic approaches are core concepts to this challenge that should have taken more credit in the organization and discussion of this paper which can be taken as incompleteness or weakness. it is also indicated that reconstruction convergence time is proportional to network size. This contradicts to the general scalability metrics and algorithms–machine measurement principles: Isospeed and Isoefficiency scalability.

Victoria Huang et al, [47] introduces a new approach to LB of DSDNCA unlike most approaches in literature that proposes dynamic migration of switches to underutilized controllers from the overloaded one. Most research works in literature suggested that static binding is one of the causes of uneven load distribution among controllers in DSDNCA. The authors also stated that they used their own approach it is because the dynamic migration is found to have an issue of latency to the network and complexity to the system. The approach they used is called BLACK, which is a binding less architecture for DC. This architecture optimizes uneven load distribution with the help of scheduling algorithm in the newly added scheduling layer to the existing SDN layers. The result of the test they have conducted on their prototype shows an improvement in terms of throughput and response time as compared to static migration approach and complexity of dynamic migration. The roles and pictorial representations of the binding less switch-controller association, improved randomized scheduling and BLACK system components are among the strengths of the paper to state. On the other hand, the authors lack to show the relation of their approach to CPP and heterogeneousness which are very crucial aspects of DCA that affect LB in one way or the other and this can be taken as a weakness.

The work of the authors in [48] is on the load distribution of software defined networking based on the controller performance. For this they have proposed a load balance method called Load Distribution mechanism and an algorithm based On controller Performance (LDOP). They have used the LDOP to reduce the burden of overloaded controller by migrating switches to other controllers. It is also used to minimize the impact associated

with overloaded controllers. In addition, they have indicated that there is a controller manager in the mechanism they have followed. This manager is fully engaged with a duty of switch migration decision among controllers with a capability of minimizing the number of messages the controller send to announce about their burden. The LDOP algorithms capacity, to run the system by the technique of reducing total number of exceeded loads from the threshold of all controllers when all controllers are overloaded is one of the great works of this paper that can be stated as a strength. However, the missing overload reduction technique to the case of different overload handling capacity controllers they have stated at introduction part can be stated as a weakness of this paper.

3.4 CPP and Fault Tolerance of SDN Controller

Bala Prakasa et al, [34] have done a comprehensive survey on controller placement problem. The method they have used was a comprehensive review of the existing literatures in the area of CPP paying due attention to six core aspects. These aspects are: target network environment, traffic characteristics, controller characteristics, solution approach, reliability of network elements and various optimization objectives. Their finding, which is well organized and summarized list of future research directions in the area of CPP is the result of analysis done on the literatures in terms of those six aspects. The overall organization of the paper, their key findings and the summary chart of taxonomy of CPP in SDN as Fig.2 at the paper are among the strength of their work to be stated. On the other hand the paper misses the expected creditability to deal with algorithms unlike too much dedication given to the theoretical mathematical modeling and formulation. This can be taken as a weakness.

Khushboo Kanodia et al, [38] have worked on a hybrid approach of resilient CPP in SDN. They stated that resilient CPP deals with issues of failure of a node or link. The method they have suggested was meta-heuristic hybrid particle swarm optimization and simulated annealing algorithm for resilient CPP to minimize the average latency. They have also tested their approach on different networks and found better results as compared to the approach that implement Particle Swarm Optimization (PSO) and Simulated Annealing (SA) algorithms. The presentation of customized algorithms, the demonstration on the use of PSO algorithm in two dimensional matrix to represent changing controller positions as well as latitudinal vs. longitudinal axis based graph representation of controller positions are the

strongest parts of the paper to state. On the other hand, the paper would have been more creditable if it had more emphasis to the explanation and demonstration of a hybrid approach in the context of other core aspects of the SDN based network.

The work of [49], is on the performance comparison of two met heuristic optimization algorithms: Non-dominated Sorting Genetic algorithm II (NSGA-II) and the Pareto Simulated Annealing (PSA) taking a consideration of latency and BW in the objective function. Both algorithms have showed good results for accuracy, processing time and lower utilization of computational resources; but, the NSGA-II has showed better convergence property and exploration of search engine space. It is because the PSA is good for optimal solution under small and medium combinatorial problems. For this reason the NSGA-II multi objective algorithm is selected by the authors as a heuristic for CPP. The very good experimental setup, demonstration and presentation of results are among the very good works of the paper that can be taken as strength; however, the paper lacks to show comparison of multiple algorithms for optimal solution in a comprehensive way. This can be taken as a limitation or weakness.

The authors of [33], have worked on DC clustering in SDN based networks. They have proposed novel clustered DCA with multiple popular controllers which are evaluated in real world topology emulated for SDN environment. The result of test has showed that the proposed method improves latency and packet loss compared to DCA without clustering. The experiment is conducted using HP Virtual Application Network (VAN) and ONOS controllers configured and installed on multiple Amazon cloud servers running Ubuntu server. Additionally, extra improvements are also shown by the proposed method in terms of CPU utilization and better tolerance to unexpected load fluctuation while operation or even to failures. An improvement in latency which is one of the core aspects of SDN and also an improvement on packet loss which is used to save data, the most important asset of a network system, are great achievements to be stated as strength. On the other hand, they have stated they used HP VAN and only Ubuntu server which can be taken as a weakness in terms of vendor neutrality and heterogeneousness concepts of SDN paradigm.

The authors of [50**],** have worked on CPP strategies for multiple link failures considering network properties to minimize the worst case latency and constraint-based solution for reassignment of switches to active controllers. The method they have used for this strategy is decision making under uncertainty with the objective of finding the optimum positions of controllers in the network with a reduced latency and maximum utilization of the controllers. They have also evaluated the proposed approaches on various topology zoos and have found that the decision making under uncertainty provides better CPP solution as compared to existing mathematical models for any SDN network topology. It is a very well organized and presented paper with no weaknesses we have observed.

The works of the authors of, [31], is on the study of fault tolerance performed for two architectures: single master with multiple slave controllers and multiple slave controllers. For this strategy, they have proposed a model called Generic Controller Adaptive Load Balancing (GCLAB) for SDNs. This model adapts the load among slave controllers based on GCLAB algorithm. They utilize Mininet 2.2 simulation tool with floodlight controller for experimentation. The experiment conducted using GCLAB for a master controller distributing switches among four and five controllers as a case study. Throughput and response time were used as a metrics to measure performance. In addition GCLAB is compared with two algorithms: Hyper Flow and Enhanced Controller Fault Tolerant (ECFT). They have found performance improvement for GCLAB 15% and 12% when compared to Hyper Flow and 13% and 15% when compared to ECTF in terms of response time and throughput. It is also a well organized and presented paper except regardless of the heterogeneousness issue (only ubnntu server) as well as the quantity and type of controllers used in the experiment.

3.5 SDN Controller Architecture in LAN

The authors of [51] have worked on improvement of end users utility in SDN WAN systems. Their motive to do in WAN is, most previous works for LAN environment which focuses on optimal switch to controller association targeting on minimization of response time do not consider the consequence of it in WAN. Consequently, they have considered that the importance of separate layer-2 and layer-3 SDN controllers in LAN and WAN environments of their proposed work respectively. The approach they have followed is load

prediction based alertness which reduces the burden of the controller. Even though the approach they have followed may create extra delay for initial packets of flow entry that leads to prediction error, it rather reduces the error by selecting an optimal timeout value of the flow. Additionally, they have also developed and used three separate algorithms for flow entry timeout computation, asynchronous load prediction and effective load migration. The simulated experiments have showed various test results for packet loss and jitter with an acceptable efficiency of the proposed scheme. The limitation of their work lies on the limitation of load prediction module which is explained as : whenever a flow entry expires a new request for the same traffic-flow arrives at L2 controller

The work of Abdul Hameed and Muhammad Wasim [52], is on the study of SDN for Emulating Virtual LANs. Even though the role of VLANs and SDN is on the traffic and packet isolation with orchestration of further network functionalities, VLANs are almost limited to layer 2 with the SDN committed for further extended functionalities. Their work is on the question and answer of whether to completely replace or not the existing VLANs by SDN or migrating it. This is accomplished with four VLANs function they have set it for study again which they have explained this change or migration as: throttled by the lack of performance results and the undefined benefits versus challenges tradeoff. The experimental setup topology they have used is formed from three switches and 30 hosts in a linear fashion. They have also stated that the in the emulated environment they have use wire shark and iperf for traffic capture and analysis with a hybrid of operating systems from windows 7 and Ubuntu for workstations. The result of their experiment and survey have showed that even if the implementation of VLANs wit SDN has some benefits it is difficult to see these benefits without SDN. This is due to the evolving nature of SDN as better performance and scalability of VLANs costs larger carrier networks and network infrastructure investment.

3.6 SDN Controller Architecture in WAN

The work of the Authors [53] is on the wide spread application and importance of SDN in research, academia and industries as well as in business, which is on resource utilization among different data centers of WAN. For this, they have used a method of traffic control scheduling based on theoretical formulation and have proposed two-layer traffic control structure with a multi domain distributed controller algorithm. This algorithm which works

with dual decomposition method is implemented by exchanging information between local domains and root controller. The simulated experiment they have conducted in case study with maximization of throughput optimization objective has compared and verified the proposed algorithm with the central algorithm.

The authors in [54] have worked on an SDN solution for performance improvement in WAN. For this they have proposed architecture with an SDN based overlay solution for legacy WAN aiming flexibility and control in case of failure. This is accomplished via a distributed SDN network overlay and edge devices. The architecture partition the bigger WAN in to metros in their's case is SANRoN in to metros which implement distributed controller to reduce latency as each metros may vary in size and density. In addition, they have implemented virtualization at the network edge and have conducted test with mininet emulator and evaluated the result with two categories: flexibility and reliability. The test bed emulated 16 switches connected to controller and MACs from thousands to millions sending packets in message and counted response message out. It is done 1000 unique MACs five times repeated in latency and throughput. The evolution is conducted using three different controllers (Ryu, floodlight and open Daylight) for variable MAC messages with various results. In general, the experiment result has showed a performance improvement when distributed and edge node placement is implemented.

Table 3.1 Short Summary of Related Work

Ref. ID	Specific Area	Methods	Findings/Result	Limitation
[8]	CS and reliability	algorithm development and Testing	reduction of COMV and convergence time	failure risk
[32]	DDoS attack-aware Smart Backup Controller Placement in SDN Design.	Integral Linear Programming model, emulation and testing.	Improving Performance with less CAPEX	Lesss emphasis to OPEX analysis

[14]	Re-construction of CPL of SDN against large scale disaster	mathematical formulation, Emulation and testing	C-plane logical connectivity against disaster with a probability rate of 90% for limited node size network (<=100)	Less creditability to computational complexity and conv time
[47]	Binding less architectu re for DSDNC	Architecture and prototype development, emulation and testing	An improvement on throughput and response time.	Less creditability to CPP and heterogeneity.
[48]	SDN LB based on controller performance	Algorithm development, emulation , and testing	Better controller load sharing algorithm.	Optimum overload handling anomaly
[34]	Survey of CPP in SDN	survey , literature review and analysis	Summarized future research directions in the area of CPP	
[38]	Hybrid approach in resilient CPP	meta-heuristic hybrid algorithms development , emulation and testing	Improved latency constraint	limitation to address core SDN aspects
[49]	Performance evaluation of metaheuristic algorithm for CPP	Algorithm development, emulation and testing	improved accuracy with minimum processing time and resources	Limitation to optimal solution
[33]	DC clustering	Emulation ,testing and evaluation	an improvement on latency and packet loss	Failure to address heterogeneity
[50]	Multi-link controller failure based CPP	mathematical model, testing and evaluation	improved performance	No weakness we have observed

[31]	Fault tolerance	algorithm development, emulation and testing	load balancing model with improved performance	Failure to test with enough controllers
[53]	Scalability and DSDNC	algorithm development and testing	traffic control and resource optimization in inter-datacenter WANs	
[55]	SDNfor performance improvement in WAN	Virtualization, emulation and experimentation	performance improvement	……..
[44]	hierarchical Distributed SDN controller model	Packet inspection and analysis	Performance improvement on root controller	Limited node size and missing heterogeneity
[51]	users utility in SDN WAN systems	prediction based alertness, algorithm development and experimentation	Improved packet loss and jitter with an acceptable efficiency	load prediction module anomaly
[45]	comparison of performance of controllers	Emulation and testing	Evaluating metrics to select controllers	In complete topology make up diversification
[52]	SDN for Emulating VLANs	Virtualization Emulation and Experimental setup	Selection and Decision criteria b/n VLAN and SDN	In complete topology make up diversification

3.7 Summary

The most important component of SDN based network is the controller, which is operating as the brain. In the DCA of any type the controller placement problem, controller failure and recovery are the challenges and bottleneck to optimize various aspects of the network such as LB, COMV, scalability, latency and cost etc. Despite this challenge, there are multiple different resources available in literatures with different approaches to the solution. However, these resources are again more creditable to WAN cases. As our thesis is on the design of DCA for NM in LAN these resources have thoughtful insights for the adoption we have to make in LAN. Of course, we have also discovered, re-viewed and present the direct related work to LAN at section 3.5

On the contrary, there is a scarcity of available resources in the areas of NM, NMS, NBI and NBAPI which are parts of the core research theme of this thesis. Even we can say there are no available resources to these areas as compared to other areas of SDN. Even if found some, most of them are obsolete in order of a decade or more. This is all about our discovery and exploration to the availability of resources and fact gathering. Table 3.1 is a quick and short summarization of the related work we have discovered throughout the thesis. It is arranged with for major headings as major criteria to summarize the related work. As much as possible we have tried to explain the evaluation of individual thesis with respect to this four headings character in a short phrase that is meaningful related to our thesis theme.

Chapter 4: Proposed Architecture

In this Chapter we present our proposed architecture, HHSDNCA for NM. The generalized system architecture, its sub-systems architecture and components are discussed in detail. In each essence, every component interaction along with what and how a component does something is thoroughly examined and discussed. Algorithms and pseudo codes that best suits to each component interaction are also part of the discussion. Additionally, we have also listed and briefed some major preliminary aspects we have considered for the design of architecture concisely.

4.1 Basics

This Section refers to the aspects we have considered in the design of proposed HHSDNCA for NM. The factors we have considered to design and propose our architecture are: CPP, Fault Tolerance, Resiliency, Latency, Response Time, LB, CS, CMOV, CAPEX/OPEX, and Mission. But, we have given more creditability to fault tolerance and resiliency of the services from controller and/or the link to controller failure. It is because controllers are key component of SDN acting as a brain. In addition, we have given less creditability to CAPEX-OPEX trade-off to signify or to give more weight to the organizational role or objective and mission where the network services have to be provided. It is with due attention of sustainable quality of services which are mandatory and crucial but its interruption is super critical.

In the section herein and subsequent sections we will follow the word region or the phrase regional network is just to explicitly represent or denote large geographic area spanning LAN with division among domain, department or campuses.

4.2 Proposed Architecture

In the worst case, catastrophic incidents may affect large geographic area with devastation which is demolition of the controllers along with the links to the controller in the administrative perimeter which is large devastation. A simple failure of controller(s) along with or without link(s) is automatically recovered by heartbeat tuning initiated action on backup controllers via feasible path from list of redundant links and controllers with a predefined setting of prioritization accordingly. In our approach, these two scenarios, the

worst case catastrophe and simple controller and/link failure, are extremities for failure of controllers and their links recovery approach in a LAN. On the other hand, in a secondary basis of our interest to such catastrophe, a failure of individual or group network elements such as nodes, links etc have also fast recovery from administrators and operators according to performance tuning alerts from NMS. This can be using remote fix tools via remote procedure call or onsite locally.

In mediatory scenario, which is the devastation equivalent in the range of the damage due to simple link and catastrophe, demolition of regional or domain network(s) may happen without affecting regional or domain network(s) but by damaging the link(s) to other domain or region(s) at a higher level. In this case also, regional or domain level connectivity and services will resume except regardless of interruption of data exchange from/to the other region(s) due to failure of communication link. This situation also has the worst-case local level resiliency via the trajectory built from a hybrid mode of switches in the topology of our architecture.

However, the most important challenge of this thesis and others in literatures is the crucial recovery time to failure from catastrophic incidents including alert about faults and errors in the controller. Thus, from technology point of view, it has a solution to such incident's recovery and tolerance approaches with the primary or secondary basis of consideration, which is another more redundant path among controllers via a wireless links, built of unguided medium such as satellites. But from the design point of view, it might not be feasible even again giving less creditability to CAPEX and OPEX but more weight for service provisioning to see this challenge as an optimization constraint among these three parameters. This is again due to optimization challenge of other constraints such as ease of implementation. In one way or the other, it can be a design choice to the solution design approach in super mission critical network design of organizations, institutes, industries and business as well.

Thus there is a need of non-heuristic solution approach to this challenge. The road map to the efficient theoretical system solution is careful assessment of those factors and addressing of all the factors attributed for it in a comprehensive way. The very best thing inferred from the attempts to address this challenge is a trade-offs in between or among CAPEX and/or

OPEX, quality of services and other core performance metrics. Thus, there is again a need of optimization among those concepts which are the most important aspects and design premise perspectives of HHSDNCA.

In the generalized architecture, the first two light blue horizontally extended arrows across the system architecture are used to reflect the demarcation lines among the major three SDN planes and the last light green one reflects the extension to the data plane, end users and workstations or computers.

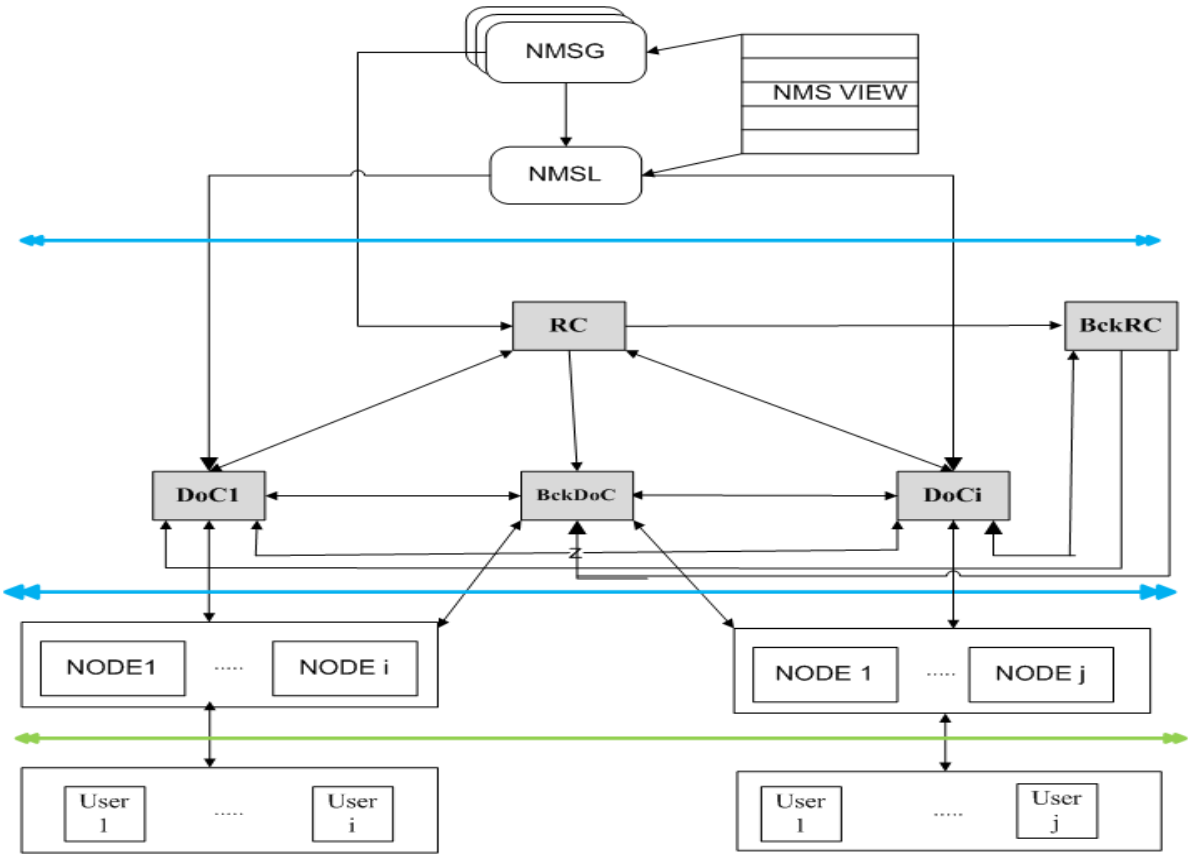

Figure 4.1 Proposed Architecture

4.2.1 Components Role and Interaction

The following are components from the proposed architecture with a detail explanation to each. The logic inside of each component working principles, their interaction to the other components as well as the best suiting algorithms is also discussed. Algorithmic

implementation of all controllers and backup controllers at this Section as well as those in the subsequent sections are in accordance with Algorithm 4.1.

We would like also to strongly express our special intention to some component's roles and interaction explanation in advance. The explanation to components such as nodes, workstations and users (administrators and operators) is as stated in literatures and books for traditional and modern networks in many contexts. We are re-explaining their interaction rules and roles here repeatedly not only for completeness but to include some special adopted features specific to our own costumed proposed architecture.

a. Network Management System Global (NMSG)

The global NMS is represented using multiple eccentric (not central) and overlapping ellipses to reflect that it has multiple and specific sub-domains at the local level. This NMS information is collected from root controller which has global view of SDN based networks. Additionally, it has interfaces to NMS view via applications both global and local levels. It also has backend database of management information system for real-time and offline management information analysis and actions as well as report generation. Real-time performance tuning is the core activity of the system including others such as SMS alert, remote fix, troubleshoot, configuration and maintenance support. It is all done with the aid of specific applications at APL which is customized and optimized for NMS.

b. Network Management System Local (NMSL)

The local level NMS is part of the global level NMS optimized and specified for each local level domain. The information of local level NMS is also collected from the respective domain controllers. Accordingly, it has sub-interfaces to each local level NMS view via applications. In general, it has customized and specified roles and features taken from the global NMS.

c. NMS VIEW

NMS view is a part of a NM task performed by NM applications and tools. It is represented with divided rectangle just to reflect various level users of NM according to

privilege and grants. In general, NMS view, NMSG and NMSL are parts of NMS that interact in an integrated manner for NM purposes.

d. Root Controller (RC)

All the controllers in the architecture are connected one to the other to form a redundant mesh topology capable of tolerating worst-case catastrophe. The RC **(**root controller) is connected to domain controllers with a double arrow just to reflect the mutual interaction and communication among domain controllers (DoCs) and the root. It is also connected to its own backup controller and backup controller of domain controllers to ensure the highest degree resiliency and fault tolerance. The root, besides control functions and forwarding flow rules, it also acts as a central hub to convey information among entities in different domains. The worst-case fault tolerance scheme is by automatic recovery of services and topology from the respective smart backup controller. This is accomplished with a prioritized fashion in accordance with Algorithm 4.1. In normal operation backup controllers are in active to operate or to forward flows except listening for failure notification alert via dedicated port and collection of backup information from the primaries. Accordingly, they automatically react to substitute the duty of active controller whenever a failure(s) happened to it or others in a secondary basis of pre-defined priority.

e. BckRC

The backup root controller (BckRc) is primarily a backup controller to the RC but it also can be used as a backup controller to domain controllers as well. This is again for the highest end resilient and fault tolerant mesh topology network design reasons. It is connected to RC with a single arrow to reflect that it is the immediate substitute to it whenever disaster happened. It is also again connected to DoCs with a double arrow to represent its role as a substitute of the RC to DoCs as it is designated accordingly in the same fashion.

f. DoC

The DoCs are connected to the RC, BckRc, backup domain controllers (BckDoC) and the underlying respective devices at the DPL. It is with double arrows just to reflect

mutual interaction and communication among DoCs, those controllers and to the devices at DPL. The DoCs are also connected each other by double arrows for the same reasons stated and also to ensure the highest end reliable, resilient and fault tolerant architecture design with mesh link among controllers. In addition, they are also major hubs to communication and exchange of data among devices at the same or different domains as well.

g. BckDoC

The BckDoC, as primary backup controller to DoCs, is connected to DoCs with double arrows for the same reasons stated at Section 4.2.1(f). It is also again connected to RC and BcKRC with a single arrow to reflect that each of them can also be alternatively used as a backup controller to it.

h. Node

Nodes are depicted with rectangle and represent multi-facet data forwarding behavior of the heterogeneous devices at the data plane and the vertexes just to reflect connectivity in various modes, medium and BW with multi-interface links. Additionally, this representation of node doesn't mean that the forwarding behaviors and interfaces are not limited to the sides and vertices of the polygon respectively. Primarily a node can be any open flow capable and enabled layer two or three switch; However, in a secondary basis it can be also any open flow capable and enabled devices such as routers, access point…etc. and non-open flow capable ones too as well.

Every Open Flow node has its own domain controller as assigned by the early-system configuration and also flow table entries are populated by flow rules for reactive flow forwarding. Each of these flow entries are again assigned by the RC and/or DoC accordingly to a specific entry or group of entries based on a certain rule and control logic. In terms of flow and forwarding, every node is controlled and managed by its own domain controller which is again controlled and managed by the root controller. Communication of nodes in the same domain is also by their IDs and addresses but for those in different domains it is via their domain controller according to pre-defined control logic for flow rules.

Input: $C_{n\text{-controller mesh cluster}}$ (c_0, c_1, c_2 … c_n), $C_{i\text{ -controller interface}}$ (c_1, c_2, c_3 … c_i), $L_{i\text{- link id}}$ (l_1, l_2, l_3 … l_i), $D_{i\text{- domain id}}$ (d_1, d_2, d_3 … d_i), $P_{k\text{- priority key}}$ (p_1, p_2, p_3 … p_k), Cs- controller status (active, inactive ,failed) ,Ls- link status(active ,inactive,failed),. HBMsg-heart beat message (C_s, l_s,C_n,C_i,l_i) / ($C_{nis}L_{is}$), SMS-alert for events and incidents, System Event Records (SER_t) where t represents time stamps for events and incidents at specific instant, Flr= failure records.

Output: system resiliency and/or recovery

0. Initialize default settings
1. Start, Flr=0
2. Resuming services and push for backup controller info update.
3. Listen for $_{HBMsg}$
4. If (Flag) //failures and catastrophic incidents
5. Record time Set
6. Flr++
7. End If
8. For i=1 to Flr
9. If controller(s) failure only or controllers and links
10. For i=1 to n of controllers
11. Identify $C_{nis\text{ info of HBMsg}}$($C_{nis}L_{is}$)
12. Look for immediate back up controller with top priority key
13. Attach it with top link priority key value
14. Resuming services
15. Update $C_{nis}L_{is\text{ inf of HBMsg about}}$ the substitute
16. Update for system event records
17. Alert with SMS for incidents and failures
18. If old controller recovered
19. Stop new link interface
20. Start old link interface
21. Record time stamps
22. Resume services with recovered controller
23. Update link and priority key
24. Update HBMsg status
25. Update system event records
26. End If
27. Else if link(s) only failure

28. For i=1 to L_i of link IDs
29. Update time stamps
30. Search for link with highest Pk
31. Activate selected link and interface
32. Resuming service with new link interface
33. Update HBMsg info
34. Update link and Pk order info
35. End for
36. End For
37. End If
38. End For

Algorithm 4.1:- Algorithm for Recovery and Resiliency.

4.2.2 Sub System Architectures

In this Sub-Section we will discuss two major sub-system architectures operations logic along with the detail explanation to each. Components other than the generalized architecture components but represented in the sub-system architecture are discussed exclusively at Sections 4.2.2 (a) and (b) accordingly. Components in the general system architecture and represented again in the sub-system architecture are omitted here as they are the direct descendants of their explanations at Section 4.2.1. This is except regardless of the user component which is depicted at both Figure 4.1 and 4.2 but abstracted with node at Figure 4.1 and accordingly explained at section 4.2.2 (b) of Figure 4.2. The workstation sub component is a non-general architecture component represented in both sub-system architectures 4.2.2 (a) and (b). Even though it is a common component of both, it is discussed repeatedly at sections (a) and (b) of the sub-system components. It is for the reason that it has a generalized similar role for both sub-components but with a little bit specialized representation contexts to each.

a. NMS

In this architecture, the experts (Administrators and Operators) and end users are at the top and bottom respectively to reflect their level of hierarchy in terms of privilege. There

are also levels of hierarchy among experts to access the system: primarily network and system administrators and others such as database administrators, front end technical personnel etc. The view and access to this system is using various tools via network management applications. The overall managerial rules and policies are also controlled by the RC that encodes and executes such codes accordingly. In general, Policies in SDN refers to traffic identifiers in two layers: high level identifiers (user and application name) and low level identifiers (IP address and port numbers). It is for this reason that the Network Management Policy (NMPL) is represented here as a separate entity inside APL watching, identifying and acknowledging high level traffic identifiers.

The depiction at figure 4.2 represents the general NMS architecture specifically identifying global and local view according to mandate. The major HW NM tool of NM at office is a computer. It is usually high-tech, largest screen size and resolution with enough of primary memory, processing capacity, storage and graphics memory well as compared to computers in the network. In advanced context, it can be a server of such complying specification accessed and utilized among IT staffs as per privilege and role based grants.

The selection of this tool is should be clearly identified for permanent use at office and data center and mobility outside of office .Thus for advanced NM, we recommend a super smart surface computing, multi-touch or any of wall mountable screen and/or processing tools. For mission critical organizations NM again can be a typical of such devices can be those that we see in broadcast media, weather forecast and astronomical stations, geophysical observatories or quantum computer of the recent times. On the other hand, outside offices these HW NM tools can be any of high-tech hand held devices for the flexible NM in terms of mobility inside a compound across the network administrative perimeter.

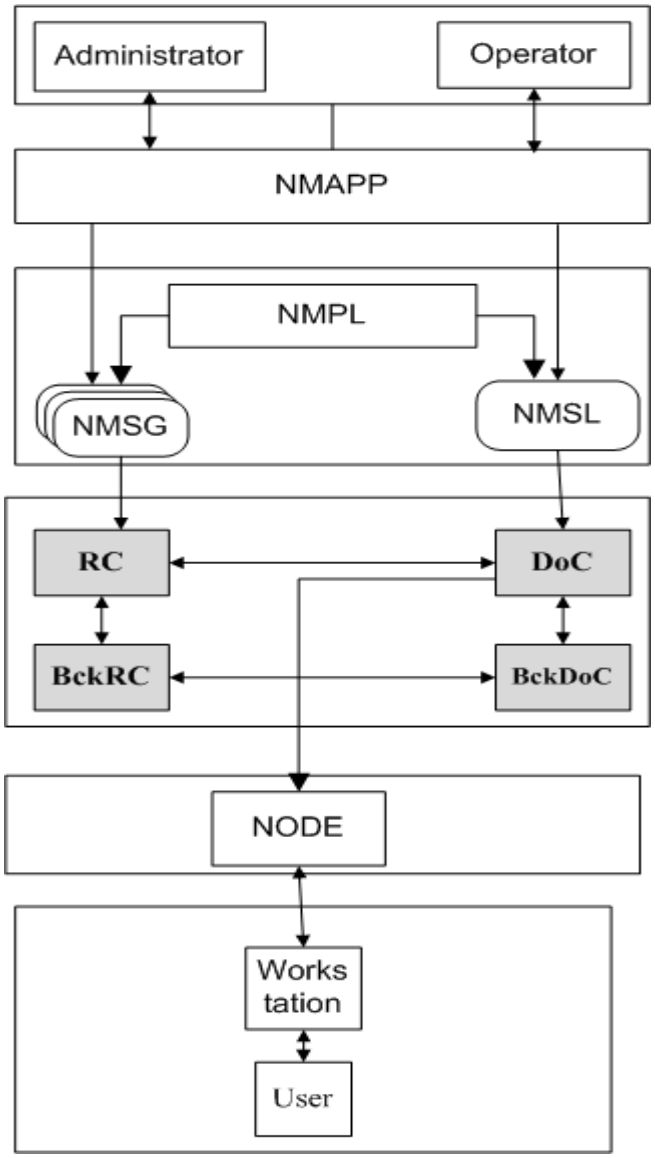

Figure 4.2 Proposed NMS Architecture

NMS Sub-System Components Interaction and Rules

Along with their detail explanation of each, the following are major components special to NMS.

i. NMAPP

NMAPPs are interfaces to communicate to the NMS. The top users are connected to two separate components of NMS with double arrow just to indicate privilege level or role based access to the NMS as well as popup messages and real-time notifications.

ii. Network Management Policy (NMPL)

NMPL is a component represented inside NMS and is connected to both sub-components of NMS with two single arrows separately. It is to show that the role or privilege based access to the system is based on high level policy identifiers requirements even though the rules and policies setup in HHSDNCA is centrally at RC.

iii. Administrators

Administrators are top level users to NMS in terms of privilege grant; however, there are also privilege difference and hierarchy among administrators based on the role they have to play in the system or the duties and responsibilities they are assigned in the organization,

iv. Operators

As compared to the administrators, they have less privilege and role with most of the duties outside of the central data center. Primarily they are engaged in monitoring, configuring, tuning and maintaining activities at the regional data centers and hubs.

V. Workstation

It represents any device attached to DPL devices port directly with wired or indirectly with wireless medium and has a definite address in the architecture. It can be any computer, mobile devices, gadget or any other machine directly interacting with humans by input devices with wired and/or wireless means or any other device or machine connected but self operating without direct human interaction.

VI. User

Users are also depicted with a square to represent diversity and heterogeneousness in terms of medium of connection, HW and OS platforms as well as privilege. But again, user representation doesn't mean it is limited to a square and its vertices. A user has access to the system and its resources by getting connected to the system in various modes and medium of transmission (wired or wireless) using various devices (pc, laptop, tablet, phone and other handhelds ...etc.). Access to the system and its resources, the type of device as well as medium of connection type he/she can use are also allowed

according to a pre-defined privilege grant of users in the enforced policy of access and usage.

b. HHSDNCA

Is a single domain component of Figure 4.1 with links to RC and other controllers. The explanation of hybrid hierarchical SDNCA is a direct descendent of the general proposed architecture except the explanation to the extended and redundant hybrid wiring topology built of switches. As depicted in Figure 4.3, the DoC is connected to nodes 1, 2 and 3 with three separate double arrows individually and again node 2 is connected to nodes 1 and 3 with two separate double arrows individually. As to the topology, whenever the link to DoC of primary forwarding devises, which is either of or both nodes 1 and 3 is failed, alternatively a connection to the underlying domain devices will resume via node 2 and either of or both nodes affected will act and operate as a simple legacy forwarding devices.

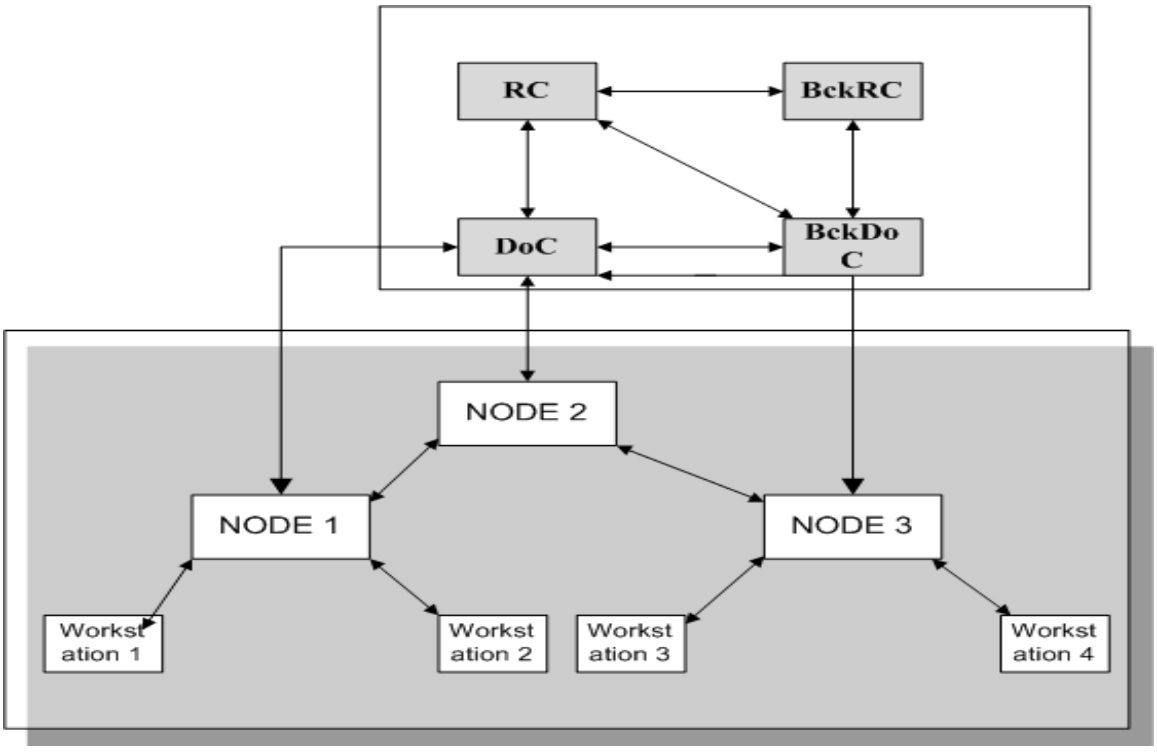

Figure 4.3 Proposed HHSDNCA Architecture

HHSDNCA Sub-System Components Interaction and Rules

Workstation

The workstations here have also the same representation as the representation at 4.2.2 (v) except regardless of its representation is limited to only to the devices connected at specific DPL port directly by cable or unguided medium. Apparently, users are human elements or machines that operate and/or utilize resources and programs of workstation.

4.3 Summary

The basic notion of our architecture is on holistic view of solution approach to resilient and fault tolerant system with minimal time. This needs a network architecture that can tolerate faults from simple link failure to worst case catastrophe. Catastrophe incident in our design consideration doesn't mean not only the damage due to seismic waves but also other equivalent severe damages to network infrastructure and services by environmental factors including human made factors on purpose or not. For this we have proposed our own architecture which is built of mesh links among redundant and distributed controllers.

The deep inside anatomical view of our architecture has showed controllers are linked with full mesh in the topology. This is to say that the theoretical total number of inks among controller are ten as calculated from [n (n-1) / 2] , n=No of controllers. It is done this way for an optimized physiology of controllers set functionalities. In general, our efforts lie too much to contribute for the solution of two challenges to this domain. The first is the use of distributed controllers in a LAN to avoid or minimize failure incidents recovery time that leads to ensure sustainable service delivery. The second is an introduction of alternate DSDNC deployment in a LAN as we have showed it separately for root and domain controllers. This a new paradigm as the previous rarely use of DSDNC in a LAN closely tied controllers each other in the same or different racks of a minimum proximity and maximum separation limit within a single data center in a room.

Chapter 5: Experiment and Result

5.1Preliminary

This Chapter is part of this thesis that verifies our proposed architecture. Accordingly, adoption of DSDNCA concepts of a WAN in to a LAN to cope up with incidents with severe consequences should take in to consideration too many parameters that will be used for test and verification. Among several of them, we have considered: bandwidth, throughput and latencies: variation (Jitter) including the icmp reply test for reach ability. Hence, we are going to exemplify here the deployment of analogous mininet custom topology to our proposed costumed architecture at figure 4.1. We have conducted the test on the depicted topology and found solutions to looping avoidance and automatic recovery response challenges that helps us to verify the existence and viability of our costumed architecture. This is conducted by default or with in-band communication implementation choice of SBAPI.

This test is also further conducted with VLAN implementation triply not only to comply with multiple vlan and domain topology complying factors but also to compare the results of out-bound communication implementation results with the in-band. Thus, this will lead us to re-enforce hybrid hierarchical approach of our research theme. Further test is also conducted with large size and diversified element network data set of DPL elements with both implementation choices. The metrics used are band width, throughput and round trip time and the whole process is reported including the explanation of the results obtained. This leads us again to the profound evidence of the existence and viability of our proposed system more reliably. In addition, this demonstration is re-revised by finding solutions to another more basic challenge. It is all about finding formula to the decision of how many size DCs we need to have and on what basics. Additionally, the measures taken to improve the performance in relation to this challenge will be thoroughly examined, verified and discussed in detail.

In the subsequent sections of this Chapter, we will elaborate in detail: integration, prototype development, testing, result analysis, evaluation, presentation and design demonstration. The discussion to the tools we have used is also the part included herein with a chapter summary at the end.

5.2 Tools

As part of testing stuff acquisitions the primary device we have used is a laptop with the following basic system rating. Additionally, we have used another high-tech desktop computer with a rating nearly to the order of server requirements for major computing resources. Accordingly, the specification for each is as follows, **Laptop**: Processor: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz, 2904 MHz , 2 Core(s), 4 Logical Processor(s) ,OS : Microsoft Windows 10 Pro , System Type:x64-based PC , Installed Physical Memory (RAM) : 8.00 GB. **Desktop**: Processor: Intel(R) Core(TM) i7-8700T CPU @ 2.40 GHz, 2400 MHz, 6 Core(s), 12 Logical Processor(s) and OS: Microsoft Windows 11 Pro, System Type: x64-based PC, Installed Physical Memory (RAM): 16.00 GB. In addition, the SW tools we have used here are for the purposes of coding and development, emulation, analysis and presentation. Each of these tools is also separately discussed in detail in the subsequent sections..

5.2.1 Development Tools

The following are major development tools we have used in relation to basic IDE features integration and demonstration. Primarily this is for coding, custom topology development and algorithm analysis.

A. Python: - Python 2.7.18 on Ubuntu-20.04.5 is the version we have used for custom topology creation, associated code generation and modifications. This is again in relation to the implementation of proposed architectural topology framework explicitly depicted at Fig 4.1, Fig 4.2 and fig 4.3..

B. Java: We have used openjdk version and build 1.8.0.342-8u342-b07-ubuntu01~20.04-b07 for code migration and customization related to floodlight controller and open switches features and behaviors.

5.2.2 Emulator Tools

For emulating and testing, realistic virtual network is created on a single or multiple virtual or native machines by different tools which have their own strengths and weaknesses. As our native host machine is windows based and the emulator and controllers are all Linux based, we have installed hypervisor that allows hosting virtual machines. In addition this

hypervisor allows hosting other software complying with emulator, controller families and versions. Accordingly, we have taken considerable time to learn, understand and select one from two popular virtualization tools we have practiced on demo in laboratory basis: Virtual Box and VMware Workstation Pro.

Finally, we have decided to use the proprietary VMware tool that supports only two OS platforms (Windows and Linux) with a special and extra bios feature than the open tool Virtual Box that supports multiple OS platform variants. Our selection decision is based on several criteria among which two decisive factors to select one are: Primarily, VMware outperform Virtual Box in many ways as it is less resource demanding in terms of processing and memory. Secondarily, VMware is more flexible and convenient than Virtual Box to our testing and demonstration in terms of ease of hosting a multitude of virtual machines and controllers with a simple interaction steps of configuration or by default. The selection criteria and details of controller(s) and emulator will also be discussed latter at the subsequent sections we will follow accordingly.

a. **VMware Workstation Pro 16**

VMware Workstation Pro.16.2 is the basic hypervisor we have deployed for the reasons explicitly stated at section 5.2.2. We have used it to host ubuntu-20.04.5-amd64-desktop that allows installing other testing tools and applications that will be utilized as its test staffs with an installation preference of CLI commands or GUI tools. But, we have used here CLI. Floodlight controller, mininet, python, open switches, wire shark, gnu plot and iperf are among the major applications and testing tools installed on ubuntu.

b. Mininet

The main emulator we have used to establish topology and to communicate among nodes, switches and controllers is mininet version 2.3.1b1. Basically, it is a tool purely command based of its own terminal environment or other external terminal windows as it is Linux based; however, it has also a capability to import GUI features on demand. One of its GUI feature is called miniedit.py. It is used for custom topology formation, visualization and synchronization as well as custom topology code generation and modification. There are various ways of using CLI command to call the miniedit window and the one we have used is: *$sudo ~/mininet/examples/miniedit.py*. This feature

can be installed from mininet command line environment by appropriate procedural calls to the local installation python files or live with remote procedural calls mitigation.

Mininet is also used for the interaction and use of various applications such as wire shark, gnu plot and iperf which are deployed to test performance and edit results. In addition, the stated versions of mininet and ubuntu are selected to comply with the system requirement of testing and evaluation environment and resources as well. In general, this environment is built with a multitude of considerations. Primarily we have considered, the hosting machine and OS installed, proposed network architecture, open virtual switches and controller requirements as well as other SW and HW requirements.

C. Controller

The controller we have used is floodlight-ubuntu-20.04 default. It is selected for its versatile features to sufficiently support high tech SDN environment testing and emulation environment. Among several features to list some are, its friendly web based utility to simply view topology elements and integrate it with applications, view and analyze flow tables and custom performance metric values. Simple integration steps for NBI and SBI apps are another feature to be stated. Additionally, on demand source code access privilege for customization is another prominent feature even though it is partially not open for some special functions by nature.

After its complete command based installation, floodlight controller needs instantiation for the services it has to provide to be checked for its readiness. Xxxx:~/floodlight$sudo java –jar target/floodlight.jar is the command used to instantiate the controller service at the floodlight directory CLI window of Ubuntu terminal. In addition, http://localhost|[127.0.0.1]:8080/ui/index.html is the default url to view the controller, nodes, open switches, links and other topology staffs and features. This is again possible after complete installation of the controller and instantiated services of it.

d. Open Vswitches

The version and type of OpenFlow capable switch (OpenVswitch) we have used is in accordance with OpenFlow, SDN based and proposed architecture compliances. Typically, it is an OpenFlow protocol 13 capable and compatible to the controller

version stated at section 5.2.2 (c) and others hosted on the hypervisor as part of testing tools set or staffs. The installation of open vswitch is the embedded part of mininet installation at the terminal windows or separately by GUI wizard of windows.

e. Smart Backup Controller

We have used such controllers to designate the behavior of two backup controllers at Fig 4.1 of our proposed architecture. It is an emergent and recent controller type technology we have investigated and indicated in literature review, at section 2.4.2. We have adopted and accustomed to our architecture for an optimized performance reasons explicitly stated at section 4.2.1 (e) and 4.2.1 (g). Even though, we have adopted the notion of such controller features to our architecture, we can't find these controllers in the web as download separately or as component part of emulating and testing environment tools. For those reasons, we have used it by customizing the notion and behavior of floodlight SDN Controller properties accordingly to match to the behavior of the smart backup SDN controller features.

f. Sketching Tools

Microsoft Visio 2007 and Edraw Max 11.1.0 are the tools we have used to sketch and draw. In addition, we have also used adobe Photoshop cs4 to edit, merge and integrate screen shots of test results.

g. Screen Shot

Microsoft snip is the application we have used for desktop scanning and screen printing.

5.2.3 Analysis Tools

The following are the tools we have used to capture traffic and analyze packets early at row data analysis of experimentation and latter at information analysis and presentation of the result demonstration as well.

a. Wire shark

We have used it to capture and analyze traffic using various parameters at various strategic locations of the network. This is to say that the analysis is conducted at locations that significantly affect performance of devices, their components, links and the system as a whole. Primarily it can be applied on the controller, links and devices at the DPL. Those links are links between controllers and switches, controllers and

controllers and switches and switches in the same or different domain. It can also be applied at interfaces of open virtual switches and controllers for the same purpose. The version we have used here is 3.0.6 and the CLI command to run wire shark from mininet VM terminal is *Xxxx: ~/$ sudo wireshark &*.

b. iperf3

Iperf3 is another tool we have used to measure performance in terms of speed and BW between two end points of network(s) elements as a client and server. Specifically it is used to measure TCP-throughput and UDP-latency.

5.3 Prototype Development

We have developed two variants of prototypes to exemplify the results obtained from the test that will help us to verify the essence of our solution approaches. These two prototypes are the analogous topology to specific architectural depiction at Figure 4.1 and another one applied for the generalized LAN design and deployment formulation. The test is conducted on mininet emulator locally deployed with an appropriate custom topology of each prototype. Floodlight controllers interact with DPL elements accordingly to the specific topology element role they are designed to represent and interact.

5.3.1 Prototype of Proposed Architecture

The basic design premise of our approach focuses on the use of redundant controller and mesh link topology. By doing so, we have tried to adopt the WAN's CPP best practices in to LAN. Thus, the first thing to be considered and adopted to our architecture is CPP determination for how many of them and in what arrangement and location. According to the graph theory the theoretical static assumption of k-controllers and n-nodes graph with total number of placements is $\binom{n}{k}$ ways at different locations of the topology or administrative perimeter of the network. Where, $\binom{n}{k} = \frac{n!}{K!(n-k)!}$ (4)

But the question comes to this generalization on how to adopt this notion in to dynamic allocation to determine the value of k for an optimized performance, which is to say that how many of them in a LAN and in what basics. Lets' say initially our solution approach hypothesizes five controllers of Figure 4.1 to denser nodes of classic LAN in a relatively small geographic area span and administrative zone. As to our assumption the spatial

arrangement of such controllers is as follows in a relatively small administrative perimeter. The root at the main data center is mesh linked to two backup controllers (BckRC and BckDoC) in triangular or planar arrangement depictions at Figure 5.1 and Figure 5.2.

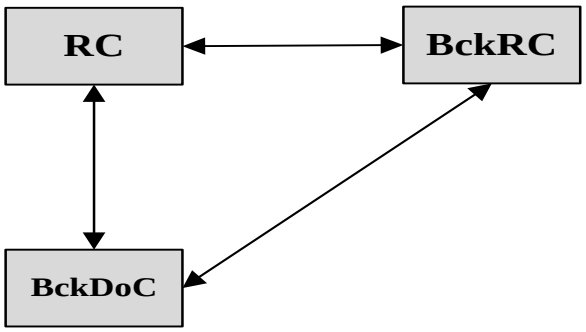

Figure 5.1 Triangular Arrangement

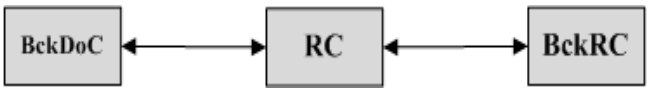

Figure 5.2 Linear Arrangement

Our approach with five controllers uses location arrangement or placement paying due attention to density of users and nodes, geographic span or administrative perimeter and distance between controllers and switches. Thus, the total number of controller determination criteria in large geographic span and administrative perimeter network is $3 + \sum_{i=1}^{n} DC(i)$ controllers. This indicates the decision of total number of controllers to be used is the function or result of how many domains to be used accordingly. It is because this decision can be mathematically modeled by a function or equation where the constant represents triangular arrangement of triple controllers and the variable represents the dynamically changing domain controllers to sufficiently support scalability constraints. By doing so we can calculate the total number of controllers for an optimized performance by differentiating this function and summing up it to three previous ones as $\frac{d}{dx}f(x) + 3$ (5)

, where f(x) is a function of : density, placement and other key performance indicators.

a. Topology

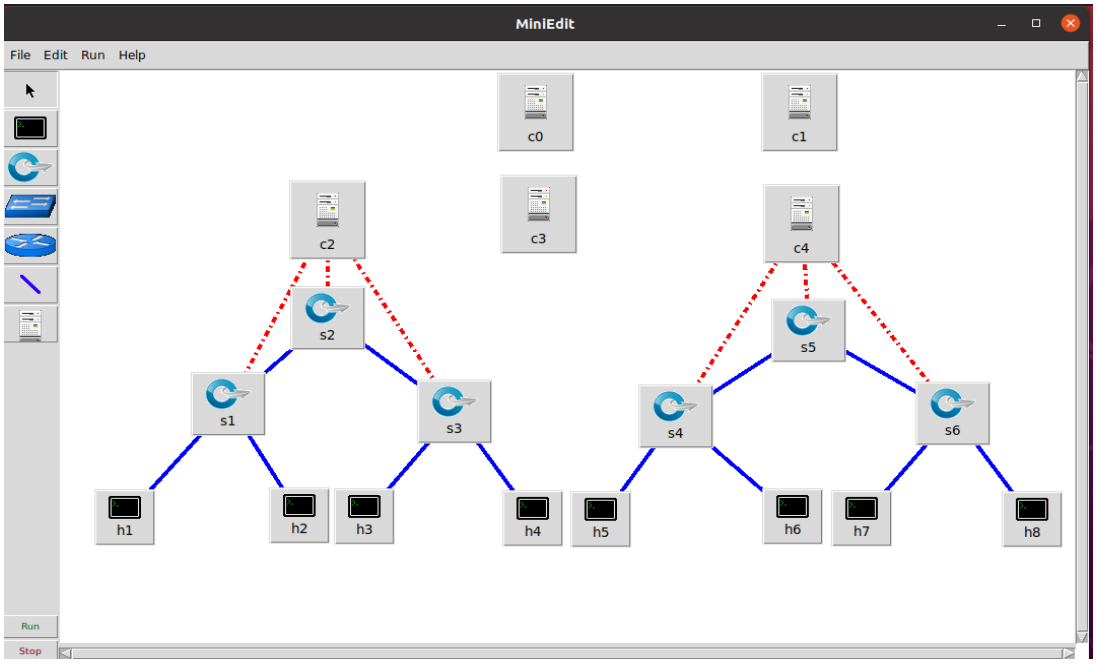

Figure 5.3 Proposed Architecture Topology

The basic topology we have used is the depiction at Figure 5.3, which is the mininet generated analogous topology of Figure 4.1. We have also re-modified this topology in the course of testing and evaluation in such a way that letting controllers' numbers and arrangement as it is but with a different DPL element combination and size to represent diversification and scalability. We have conducted the test variably in batch of such element sizes to observe the behavior of core controller performance metric values and the system as a whole.

b. Testing Approaches

The testing is conducted for each of these alternative deployment approaches with on purpose interruption of each or all of the controller services. This is primarily accomplished by mininet CLI commands and also by deleting or disabling controller and/or its links. This interruption is done this way to designate the real world analogy of controller(s) and link(s) failure and observe the behavior of instantiated network topology change responses. These responses are in terms of latency variation (Jitter), throughput including round trip time for reach ability test. The results obtained for each of these metrics is demonstrated and discussed in detail on Section 5.4.1. In addition,

the same interruption is exercised on switches and nodes in the same manner of what we have done to the controllers and their links.

Analysis applications like Wireshark and iperf are also used to capture traffic metric values and analyze bandwidth. In addition, automatic response time and related metrics such as round trip time and throughput values are also captured and analyzed at various interfaces, ports and locations of the testing environment elements. This will be used for the result evaluation of each approach deployment. In addition, it is used to compare it with evaluation of the other options of deployment in the same or a different approach. We have also conducted the test with two folds. The very likely reason is just trying to test and evaluate our proposed system incrementally starting from simple analogous topology depictions, configuration schemes to the more realistic and complex representation. This leads us to clearly outline and debug the problems for fast custom solution deployment accordingly or a re-revision of a new solution design approach if any regarding the uncertainty to looping constraint we may have from the use of mesh links among controllers.

5.3.2 Prototype for the Generalized LAN Design

It is just a consideration to use multiple distributed backup and domain controllers with a root controller isolated from and remotely linked to backup controllers and domain controllers. We have used two domain controllers and a single backup for the demonstration purpose of the proposed system, But, the basic notion to how many backup domain controllers should be used is again a consideration closely linked to the decision of how many domains to be used and the size of each. Primarily, the decision to use how many domains is again based on the decision of the type, number and size of switches to be used and number and size of nodes to be connected to switches. This is again affected by the general density of potential users irrespective of department and others commonly used for classification. In addition, it is also determined by diversified system use of potential users' geo locations and geographic span issues. The geo location and geographic span cases are considered for structured cabling and links. This is in relation to the medium used to connect the interfaces of devices at various stages

of the topology. This structuring is one of the very decisive factors of performance optimization besides labeling and indexing for identification and ease of debug.

a. Topology

The basic topology we have used is the direct descent of the depiction at Figure 5.3. But, we have customized it to find general formulation of how many domain controllers and back up domain controllers to be used. It is conducted as per the criteria we have explicitly stated at Sections 5.3.1 and 5.3.2.

b. Testing Approaches

We have also used here the same basic testing approach and evaluation as we have used on Section 5.3.1 (a) but with a use of a bit advanced applications and tools that helps to sufficiently capture traffic analysis metric values. This is done paying due attention to a relatively complex mesh topology behaviors with a large number of controllers used for backup and domain controller purposes.

5.4 Results

In both testing approaches, we have primarily tested for reach ability test among nodes and DPL elements in the same and/or different domains including iperf test for latency variation (Jitter) and throughput. The reach ability test is conducted by using the ping command via the mininet CLI window and the metric of evaluation is reply time with in definite TTL values (64,127 and 256). The experimental data set is miniet emulated topology of proposed for eight, sixteen and 24 node sizes. In addition, we have also tested automatic recovery from back up controllers and system resiliency with associated traffic analysis to the behavior of instantiated topology change responses. This topology change response is evaluated with respect to key performance indicator metrics values such as latency variation (jitter), throughput and round trip time. The results are all shown from Figure 5.9 up to Figure 5.17 and Table 5.1

5.4.1 Results from First Approach

By its default configuration scheme of mininet generated topology at Figure 5.3, hosts in the same domain communicate each other but not hosts in different domains .Thus we have added additional OpenVswitch to establish controllers mesh and a link between two domains. However, the addition of this extra switch alone can’t allow traffic flow

between two domains except physical link. For this reason, we have also modified our topology configuration scheme by changing the default ip base of miniedit from 10.0.0.0/8 to 127.0.0.0/8. This helps us to allow connectivity and traffic flow between hosts and devices in different domains by creating a single ip subnet of 127.0.0.0/8. Typically it is the ip address range of the extra switch, controllers mesh and the hosts at the DPL.

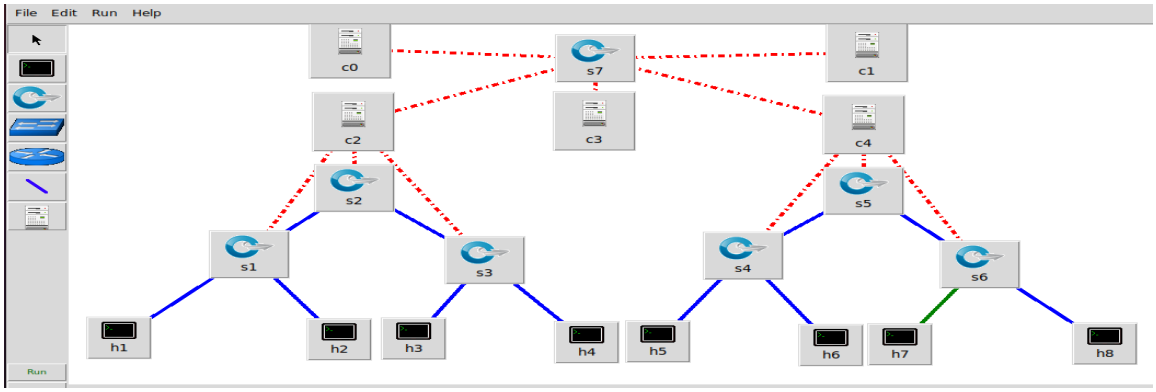

Figure 5.4 Customized Proposed Architecture Topology

The following are the major results we have got by this testing approach

A. Excellent icmp echo reply test among nodes in domain one and two. As shown in Figure 5.5 an average TTL value is in the order of 0.035 and 0.038 respectively.

Figure 5.5 Reach ability Test Among Nodes

B. BW and TCP- Throughput Test Result

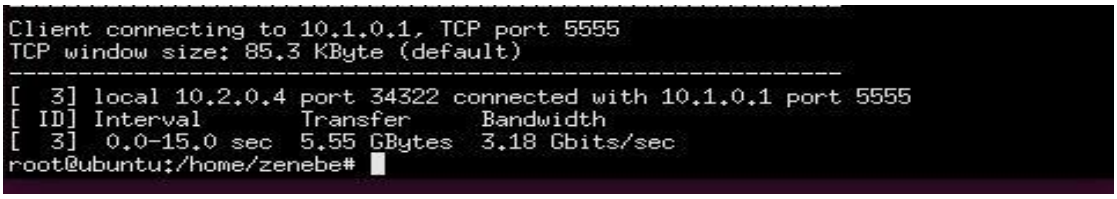

Figure 5.6 TCP Throughput Test

C. UDP – Latency (Jitter) Test Result

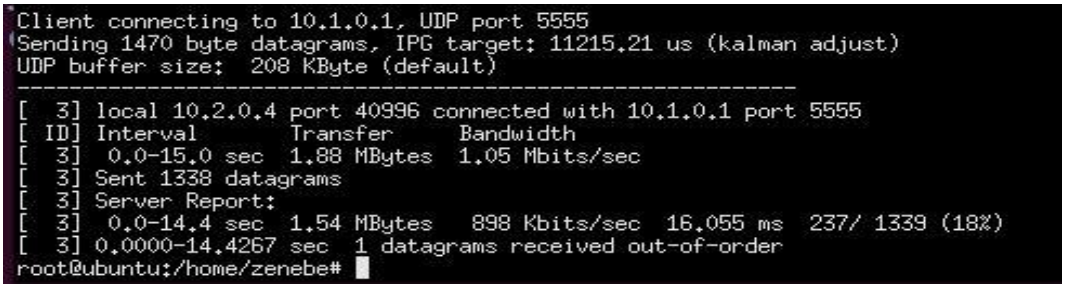

Figure 5.7 UDP Latencyt Test

However; this solution that allows traffic between two different domains is not complete as it lacks to support multiple domain and subnet network of real world analogy that we have explicitly stated as test and evaluation of its prototype. Thus, we have modified our topology depiction at Figure 5.4 to Figure 5.8 to represent and model a multiple subnet and domain network of real world analogy.

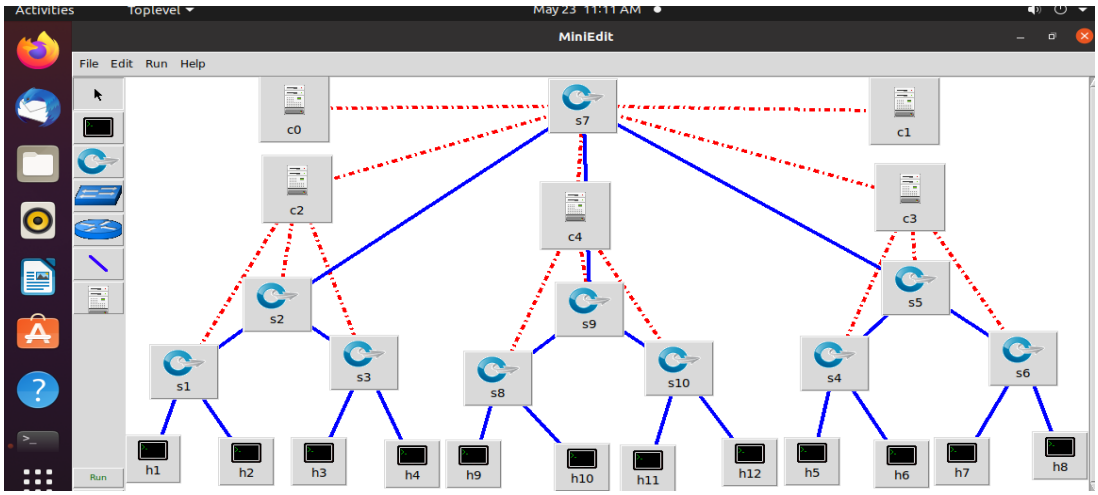

Figure 5.8 Multiple Subnet and Domain Topology

5.4.2 Results from Second Approach

The open flow implementation of SBAPI has two implementation choices as stated at Section 2.3.1, based on this Figure 5.6 is for the obc implementation as it supports VLANs. In addition, the same topology works for the ibc implementation except regardless of non-vlan tagged traffic with a single or multiple ip range pull and a specific in-band controller role unlike a remote-controller of the obc implementation. We have categorized the tests we have conducted for medium node size topology, local level resiliency and large node size topology.

A. Accordingly, we have sub-netted our proposed architecture with three VLANs each representing different subnets and domain as follows:- 10.1.0.1/8-10.1.0.4/8, 10.2.0.1/16-10.2.0.4/16 and 10.3.5.1/24–10.3.5.4/24 as shown in Figure 5.8 for obc implementation. In addition, we have added an extra link from the extra switch we have used for controllers mesh which is depicted on Figure 5.4. This link from this extra switch extends to three superior switches at the domains which aren‘t directly linked to the workstations. This allows us to establish a complete multi domain and multi VLAN network using these switches as a gateway to individual domain network. Hence, we are able to subnet each domain in different ip ranges flexibly by separating the previous 127.0.0.0/8 single VLAN configuration customization we have made for controllers mesh.

Accordingly, we have done reachability test using the usual ping command between and / or among the same domain elements to evaluate and verify **reachability** and latency constraints. This **reachability** test was successful with fast response time of TTL values in the order of 60’s seconds. In addition, we have also made testing and evaluation for core performance indicators such as latency-jitter and throughput. The experiment on obc and ibc implantation choices has been conducted with iperf3 for each of the choices in two timing intervals mode (t=15s and t=120s). The result is depicted as a gnu plot merged graphs frame at Figure 5.9 and the result of the plot curve have showed very good results compared to the test result plot of mininet built in topologies such as single, linear, and tree with the same node size of 12. We will deal more on the result and graph analysis at Section 5.4.3.In addition; we will test and verify a complete hybrid topology of large size network with a diversified element topology in the subsequent sections we will follow. The tcp throughput and udp latency graph plot printouts for each of ibc and obc is in two frames to represent the result of test for time t=15 and t=120 periods respectively.

Fig 5.9 Reachability Test with Obc

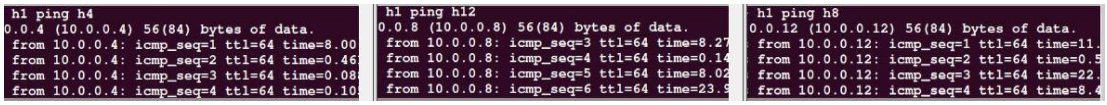

Fig 5.10 Reachability Test with Ibc.

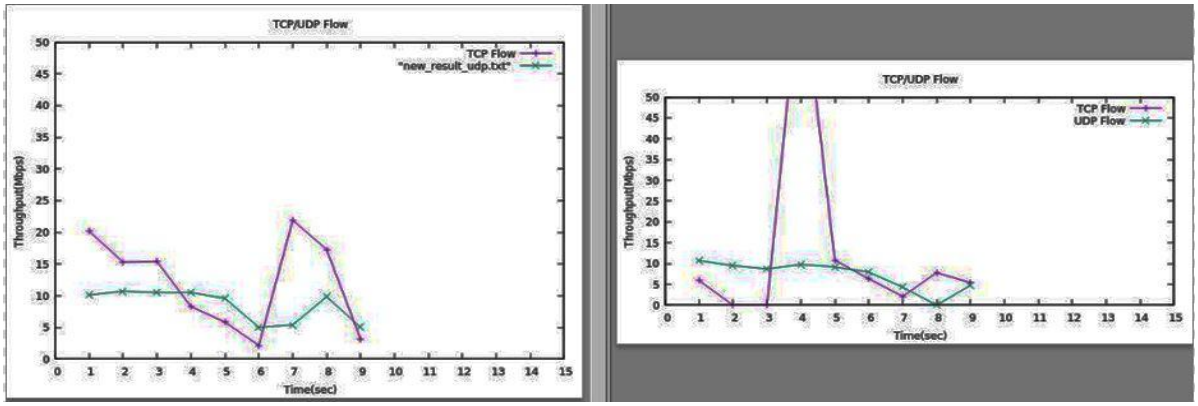

Fig 5.11 Throughput and Latency Test in Obc

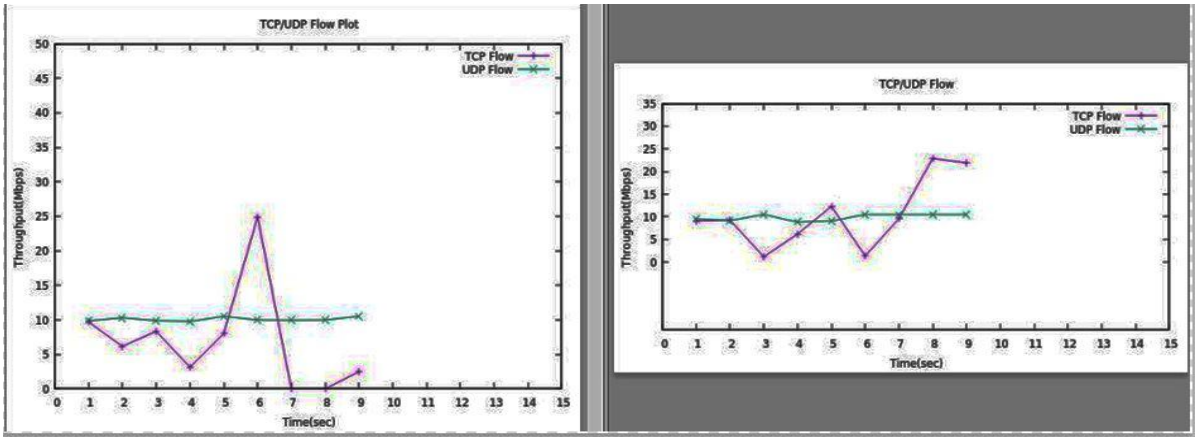

Fig 5.12 Throughput and Latency Test in Ibc

B. We have also verified the local level resiliency by testing the topology of Figure 5.8 . This is conducted by the ping command for successful reply among nodes of all domains in the topology and then after only on nodes of the first domain. The first test is conducted to reach ability test among all domain nodes and the second is conducted on the first domain nodes on purpose to compare this result with the previous one and verify local level resiliency and recovery built of the trajectory we have hypothesized. This is done specifically by disrupting the link from S2 to S7 of the first VLAN (10.1.0.1-4) which is used to establish Inter VLAN – routing. Additionally, we have

detached the rest of three links to the first domain network controller in the same manner. This is just to represent an incident due to worst case catastrophe that demolishes the rest of the network severely with a devastation extent including the links to the domain and domain controller. This on purpose disruption of the network topology by automatic detaching of the links was conducted by link s2 s7 down of the mininet CLI command and other equivalent CLI commands to the rest of three links stated. This is a test for our proposed solution clearly depicted at Figure 4.3 as a sub architecture to the main architectural solution we have stated at Section 4.0.

Fig 5.13 Local Level Resiliency Test of obc

The same resiliency test is conducted for ibc implementation topology with same average TTL results but we have put here the obc as part of completeness.

Figure 5.11 is a screen shot of the previous reach ability test result we have represented at figure 5.8 except regardless of unsuccessful reply or unreachable destination host reply message of ping echo from domain one to domain 2 and 3 for the on purpose disruption we have made to the links.

C. Test result for Large Size Network

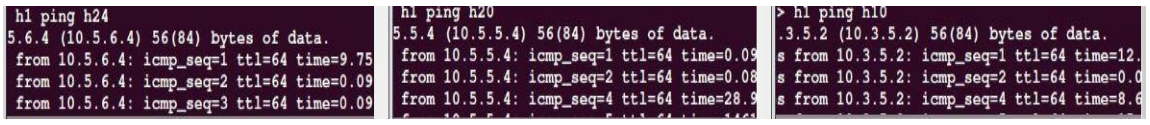

Fig 5.14 Reach Ability Test of Large Size with Obc

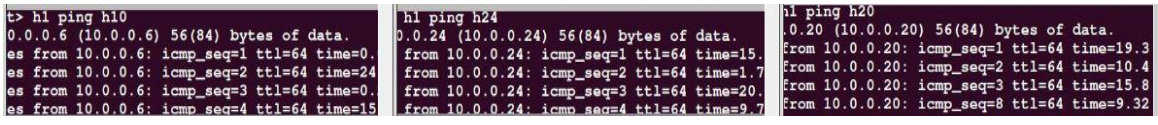

Fig 5.15 Reach Ability Test of Large size with Ibc

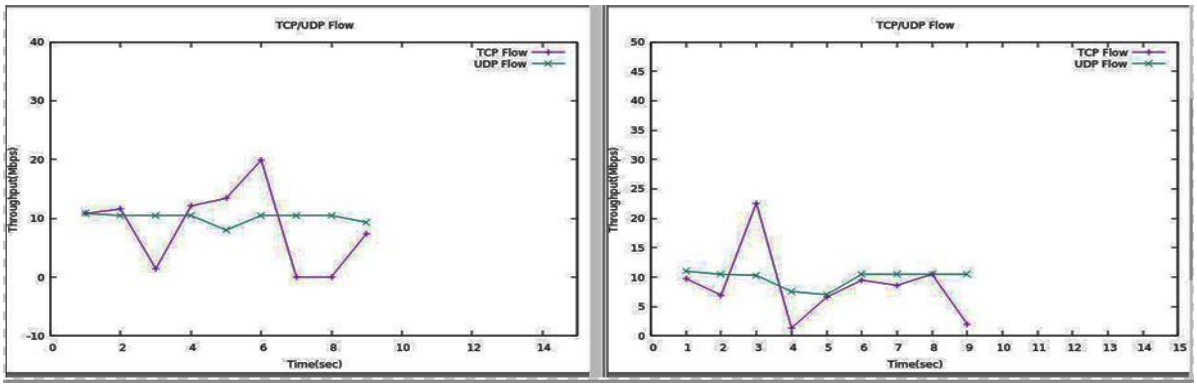

Fig 5.16 Throughput and Latency Test of Large Size with Obc

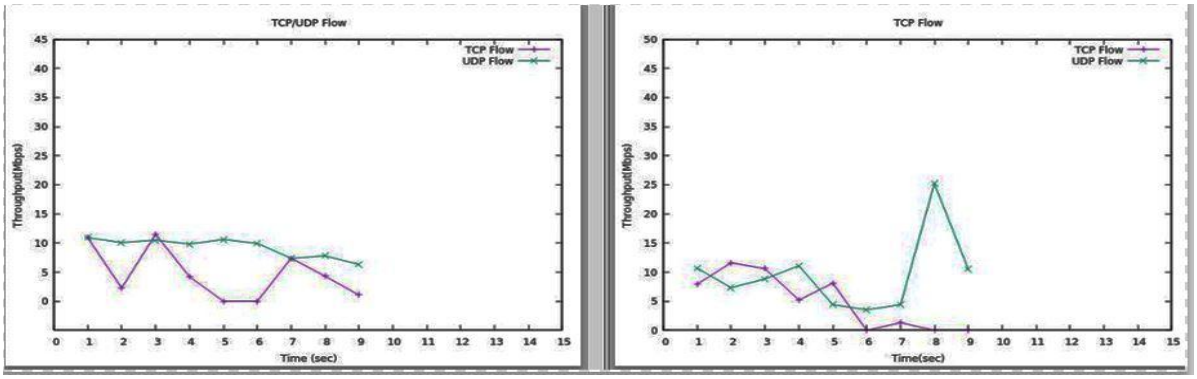

Fig 5.17 Throughput and Latency Test of Large Size with Ibc

The interpolation of performance test graphs for ibc (Figure 5.11 and 5.12) and obc (Figure 5.16 and 5.17) of individual throughput and latency plots have showed that the ibc UDP-latency test showed an interpolated nearly-linear curve with a slight decrease for latency values. On the contrary the TCP-throughput test interpolated nealy-linear curve for obc outperforms the ibc which favours our hybrid hierarchical design approach. In general, the individual performance test of ibc and obc throughput plots have showed a slight decrease or degraded performance as time increases.

5.4.3 Result Evaluation and Analysis

Table 5.1 Performance Measurement with Default Window Size and Interval

No	Topology	Interval (Seconds)	Transfer (GBytes)	Band Width	Window Size	Re-transmitted	Number of Hosts

				(Gbits/sec)	(Kbyte)	Packet	
1	Proposedobc	10	9.1,5.04	6.2,4.94	85.3	-	12,24
2	Proposedibc	10	5.96,4.78	4.96,3.48	85.3	-	-
3	Minimal	10	13.5,-	11.6,-	85.3	-	2,2
4	Single	10	8.77,7.73	7.53,6.73	85.3	-	12,24
5	Linear	10	4.84,4.95	4.65,3.69	85.3	-	12,24
6	Tree	10	7.05,7.29	6.07,6.27	85.3	-	16,25

The test results of column three and four at table 5.1 is an average value of result from three independent bidirectional test sessions among nodes in the same topology for the depicted metric values but with different client server tier for each test session. The comas at the cells of column three and four separate test result of medium and large node size of each row respectively. Additionally, the result values are iperf output printouts at client side as per the iperf recommendation. Even though the performance test using iperf tool can be conducted for trough put and band width metrics by tuning various parameters of buffering, timing and protocols such as tcp, udp and sctp we have take too much emphasis only to the TCP protocols.

The TCP and UDP plot graphs of mininet built in topologies with the same node size of our proposed except for minimal with node size of two is almost constant on the average as compared to the graph plot of our proposed at Figures :- 5.11, 5.12 , 5.16 and 5.17. In fact again the hierarchical of miniet-built violates this consistency (nearly the same as our proposed plot) when compared to its family of built-in topologies: minimal, single and linear. Even though, performance can be tested and evaluated with a holistic consideration of several parameters we have considered on three key indicators: TCP-throughput, icmp echo -reach ability latency and UDP-latency (jitter). Various icmp-echo reply tests have been conducted in 4.2 at various stages and the result of individual reply test shows an excellent average TTL value of ms. In addition, as compared to the TCP, the four UDP plots graph of our proposed at Figures : -5.11 , 5.12 , 5.16 and 5.17 have a constant graph in a relative manner. This result is also taken, with various test attempts at different days, times

and computers of various major computing resources and BW but we have depicted individual results selecting the average of the test plot graphs.

5.4.4 Discussion

Various researches have been conducted with different aspects to performance measurement of hierarchical distributed controller architectures. Some of them are totally applicable in WAN and yet some others are at both LAN and WAN. In addition, for some others applicable to LAN there might be in adequacy to comply with network management objectives. Thus, as our focus is on the deployment of hierarchical distributed SDN controller architectures in LAN for network management, it is difficult to directly evaluate the previous performance test results with our architecture in a compare and contrast manner. However, we can have a general comparison to our test result which considers three key performance indicators: -TCP-throughput, UDP-latency (jitter) and ICMP reply response time.

The authors of [44] have worked on the burden sharing of the root controller as clearly indicated at Section 3.1. They have similarities with our work in the general architecture families we have used; however, we have differences in the number of controllers, the number of nodes, test bed set up and methods followed. We have used a layer two hierarchical architecture of 5, 6 and 7 controller's topologies with 8,12 and 24 nodes respectively; however, they have used layer three and four hierarchy with 7 controller 12 nodes and 15 controllers 24 node respectively. In general, we have similarities regarding reduction of intra-domain propagation latency. They have evaluated these two topologies against the classic hierarchical approach which convey data among intra-domain and/or inter-domain controllers only via the root. They have got promising result regarding reduction of intra domain latency. In comparison, all the latency graph plots at Figures 5.11, 5.12, 5.16 and 5.17 interpolated between the beginning and ending have showed an average interpolated reduction to latency even though it hasn't direct measurement relation to theirs.

The authors of [55] have worked on performance analysis of hierarchical SDN controller arrangement. The experimental test bed setup they have employed is built of traditional

hardware-based networks using three controllers for two types of controller families ONOS and ODL. The three topologies they have deployed are emulated by mininet with a three controller each and addition of three, seven and seventy node sizes topologies respectively. The experimental test result is conducted for initial packet latency, average, minimum and maximum round-trip time RTT, TCP throughput and Latency. The result of the experiment on these three topologies is analyzed for hierarchical and non-hierarchical architecture of each controller families to observe the availability of their proposed architecture. In addition, they have also explained that assessment of SDN controller setup with different controller families is the main objective of the analysis. Even if it has similarities with the result, the latency depiction of theirs is not convenient to directly correlate and present it in a compare and contrast manner to ours due to the fact that we have depicted with Mbps Vs Time(s) and they have presented it ms Vs Number of Nodes. However, their depiction of RTT time with ms Vs Topology have general similarities to the result we have got even though we have presented on the average RTT time value in the order of 0.034ms for all three topology preferences via ping CLI command.

We have also done comparison of test results between the 12 node and 24 node size topology of the proposed architecture for throughput and response time. The response time is analyzed and compared among four individual response times of two topologies with ibc and obc implementation choices for each. Each of the four individual response times are taken with an average value of nine response times from three independent icmp echo reply messages of the first three successive response times for each. The result found is 6.647ms, 9.698ms, 6.637ms and 11.854 ms respectively as calculated from Figure 5.9, Figure 5.10, Figure 5.14 and Figure 5.15 depictions. The result of comparison has showed both the two ibc implementations are more latent than their counter part of the obc implementations and again the obc medium node size topology is more latent than the obc large node size topology.

We have also done comparison of the throughput test and the result from Table 5.1 has showed the proposed architecture of obc with large node size have showed a relative decrease in throughput with minor decrease for the rest of topologies except regardless

of linear and tree which have showed an increase in throughput when the node size increase doubly. However, this comparison is in a relative manner between 12 node size and 24 node size of each of the topologies and we can say there is no question of throughput issue at all from the result of the test we have found in the depicted window size and time interval.

5.5 Summary

The most important thing that comes with SDN environment proposed architecture test and evaluation is performance test. This can be conducted with a holistic approach by the due consideration of various attributed factors that significantly affects the desired result. In lieu with this fact, we have focused on the performance test related to BW, throughput and latencies: - inter and intra-controller. This is done by conducting test for **reach ability and icmp echo reply latency, UDP- latency (jitter), TCP-throughput and BW tests**. Even though, the limitation has to be dealt in the subsequent sections separately, we would like indicate our major challenge in advance. This is the challenge we have faced to flexibly customize and program floodlight controller for mesh link connectivity.

Latency in network is one of the core metrics of performance. It happened due to several attributed factors that accounts for it. In other words, latent network services are the indicator of poor quality network services. Thus, this leads to a conclusion that there is a question of expected quality demands for the network services. This again indicates that there is a demand to a system change or upgrade. Hence, ensuring expected quality for network services is a challenge that can be addressed with a holistic solution approach (es) that can be re-revised periodically according to circumstances for a network services quality demand in a certain period of time. This can be concluded as, it is an issue that is closely related to system dynamics in terms of services and resource usage with an optimum expected quality. It is also obvious that ensuring optimum performance or expected quality of network services is the result of several attributed factors, primarily cost.

Chapter 6: Conclusion, Recommendation and Future Work

This is the last section of this thesis organized with three major Sections: conclusion, recommendation and future work.

6.1 Conclusion

The theme of our thesis lies on the design of distributed controller network architecture for LAN management. This is with due attention of continuous service delivery and fast recovery time in organizations network where service interruption is critical. For this, we have proposed architecture built of mesh links among redundantly distributed controllers and have developed an algorithm for resiliency and recovery. We have tested and verified the prototype for viability with a mininet emulated topology. The test is conducted with a test bed of a relatively small, medium and large size DPL elements topology for two implementation choices of openflow SBAPI, in-band-communication and out-band-communication.

We have conducted tests on the prototype of our architecture among which icmp echo reply for **reach ability, TCP-throughput and UDP-latency (jitter*)***. The result of icmp echo reply test shows an average excellent reply time of multiple TTL values for all tests we have conducted. The same is true for TCP-throughput and UDP-latency test results we have conducted but with sound good results as compared to the mininet built in topology test results. The throughput and latency-jitter test are conducted by iperf CLI commands and the result is displayed in graph of through put versus time window by GNU plot CLI commands. Each graph is again compare to the GNU plot graph of mininet built in topologies minimal, single, liear and tree based on these two key performance indicators. In addition, the result of iperf BW and TCP-throughput test is conducted for various input sizes and timing window. The average numerical result of these two matrices is again tabulated in table form and compared with the results of mininet built in topology test results. The test result comparison analysis of both approaches showed very good results.

Our architecture seems to have a loop due to the use of mesh links among redundantly distributed controllers but tested and verified for it with excellent icmp echo reply message

TTL values. In addition, we have also tested the local level system resiliency against network failure due to catastrophe with mininet generated topology of the prototype.

6.1.1 Contribution of Our Work

The following are major achievements of our work.

- New controller architecture optimized for LAN management.
- Better flexibility and simplicity to enforce policies and guidelines in NM of LAN.
- Introduction of alternate controller architecture design paradigm in LAN.

6.2 Recommendation

Even though there are various solution approaches to recover from the worst case of catastrophe, each of them has its own limitations as pros and cons perspective or has a trade off in between any two or among several of performance metrics. There is no hundred percent efficient recovery solutions to the worst catastrophe. This can be from design point of view and ease of practicability. In lieu with our theme, it is due to the fact that the worst case catastrophe is not only infrastructures devastating but also environmental demolishing as well on a higher scale. The worst case recovery from catastrophe approach we have recommended at section 4.2, the non-terrestrial inter-controller communication via non-guided mediums such as satellite links is also on the notion of such premise.

In fact, it is possible to improve the performance of this approach by customizing and introducing advanced approaches such as telemetry and others in to this approach. Telemetry uses PCEP (Path Computation Element Protocol) for automatic recording and transmission of data from remote source to a receiving station for analysis. But still this doesn't mean it is an absolute solution approach for the worst of worst in terms of catastrophe. This is because all the three non-terrestrial medium approaches: Low Earth Orbit, Medium Earth Orbit and Geo Synchronous Earth Orbit are all dependent on ground stations antennas that are used as a gateway to communication and routing. In one way or the other we have recommend such solutions approach but still the implementation and deployment of this solution approach is within the constraint of various regional, national and/or international legal frame works and policies.

6.3 Future Work

The following are activities we have extended as a future work:

1. Conducting a complete test in better client server architecture with enough of major computing resource equipped machines to analogously model the development network. This can be conducted with different controllers from different vendors with different algorithms and comparing the results from each controller and algorithms to the others is also another one to be stated.
2. The other important issue we have to include here is the smart backup controller we have used in our proposed architecture. It is not easily available in the web as component for emulation or in some other way. Thus, we have represented and used it by customizing the behavior and functions of normal controllers' accordingly. Algorithmic efficiency improvement in the area of automatic heartbeat and reverse heartbeat message timings and backup teaming size and other compliances are also other aspects we planned for future.
3. Controller failure is also in one, two or all of the cases: HW, SW and/or network. In addition, failure sensing and notification algorithms are also found to be a victim of false positive messages for failure notification. The crucial recovery time is also again largely dependent of topology discovered. This is therefore to indicate that we have a deep in sight and planned future work on these two important issues: minimization of recovery time and avoidance of false positive failure notification.
4. 4. The derivation and verification of the formulation we have stated early at Section 5.2.1 is also another aspect we planned as a future work due to time constraint.

References

[1] Murat Karakus and Arjan Durresi,” A survey: Control plane scalability issues and approaches in Software-Defined Networking (SDN)“,Indiana University Purdue University Indianapolis , Vol.112, pages 279-293 , Indiana , USA , January 2017.

[2] Trung Truong , Qiang Fu and Christopher Lorier, ” FlowMap: Improving network management with SDN “,IEEE Symposium on Network Operations and Management , Istanbul, Turkey, 25-29 April 2016 .

[3] Hyojoon Kim and Nick Feamster,” Improving Network Management with Software Defined Networking “, IEEE Communications Magazine, Volume.51, Issue. 2, 14 February 2013.

[4] Jian Li∗, Jae-Hyoung Yoo†, James Won-Ki Hong , ” CPMan: Adaptive Control Plane Management for Software-Defined Networks “, IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), San Francisco, CA, USA , 18-21 Nov. 2015.

[5] Stanislav Lange; Lorenz Reinhart; Thomas Zinner; David Hock; Nicholas Gray; Phuoc Tran-Gia ,” Integrating Network Management Information into the SDN Control Plane “ , NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium , Taipei, Taiwan , 23-27 April 2018.

[6] Yudong Wang and Liong Liu ,”Centralized Network Management in SDN-based Communication Network of substation “ , 2019 IEEE International Conference on Industrial Internet (ICII) , Orlando, FL, USA, USA , 11-12 Nov. 2019

[7] I Gde Dharma N., M. Fiqri Muthohar, Alvin Prayuda J. D., Priagung K., Deokjai Choi , “Time-based DDoS Detection and Mitigation for SDN Controller “,International Conference on Transparent Optical Networks, Busan, South Korea , 19-21 Aug. 2015.

[8] Mohd Saalim Jamal, Anish Hirwe and Kotaro Kataoka,“ VIBHAJAN: A Lightweight and Scalable Control Plane Management for Multi-Controller SDN”,2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Verona, Italy, Italy , 27-29 Nov. 2018 .

[9] Dennis Tatang, Florian Quinkert, Joel Frank, Christian Ropke, and Thorsten Holz ,"SDN-Guard: Protecting SDN controllers against SDN rootkits " , IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN) , Berlin, Germany , 6-8 Nov. 2017.

[10] Wael Hosny and Fouad Aly," Controller Adaptive Load Balancing for SDN Networks ",2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN) , Egalia , Quait , Alexandria , Egypt , July 2019.

[11] Oluwatobi A Akanbi ,Amer Ajaedi,Xiaobo Zhou and Adel R Alehrbi,"Fast Fail-Over Technique for Distributed Controller Architecture in SDN", IEEE Access Vol.7, Page Numbers 160718-160727 . 14 November, 2019.

[12] Michael J. Watts and Kourosh Ahmadi," An Overview of Multi-Controller Architecture in Software-Defined Networking", CITRENZ 2019, Newzland , October 2019.

[13] Catherine Nayer Tadros , Mohamed Rizk and Basem Mukhtar ," Logically-Centralized Physically Distributed SDN Architecture", 2018 IEEE Global Conference on Internet of Things (GCIoT), Alexandria, Egypt, 5- 7 December 2018.

[14] Takahiro Hirayama; Takaya Miyazawa; Hideaki Furukawa; Hiroaki Hara," Reconstruction of control plane of distributed SDN against large-scale disruption and restoration", 2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) , Atlanta , Georgia. 1-4 May 2017.

[15] Yiran He and Zinchang Zhang," A Survey on Network Measurement for software defined Networks", 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 18-20 October 2019.

[16] Nand Mulchandani and John N.T. "Jack" Shanahan," Software–Defined Warfare", Center for Strategic and International Studies (CSIS), P.6-15, Sep. 1, 2022.

[17] Hyojoon Kim and Nick Feamster," Improving Network Management with Software Defined Networking ", IEEE Communications Magazine, Volume.51, Issue. 2, 14 February 2013.

[18] Robert Soule et al," Merlin;A Language for Managing Network Resources", IEEE/ACM Transactions on Networking, Vol.26, Issue.5, October 2018.

[19] Devi Chadha," OPTICAL WDM NETWORKS: from Static to Elastic Networks ", Network Management Control Chapter 7, John Willey and Sons Ltd publications, Hoboken, USA, 2019.

[20] Thadee Gatera,"Network Administration", African Virtual University, Applied Computer Science, Retrieved From http://oer.avu.org/handle/123456789/676 ,Last Accessed on April 19, 2021.

[21] Raimon Kangas and Esa Markus Mtetsälä , " Network Management ",LTE:Backhaul Planning and Optimization , John Willey and Sons Ltd publications, 2016 .

[22] Adrian Lara, Anisha Kolasani and Byrav Ryramurty,"Simplifying Network Management using Software Defined Networking and OpenFlow", IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Bangalore, India, 16-19 Dec, 2012.

[23] Slawomir KUKLINSKI and prosper CHERMOUIL," Network Management Challenges in Software-Defined Networks", IEICE TRANS.COMMUN, Vol.E97-B, No 1, January 2014.

[24] " Network Management ", Retrieved From https://documents.pub/document/chapter-20-558457bfe04ba.html, Last accesses on June 14, 2023.

[25] Danish Rafique and Luis Revasco," Machine Learning for Network Automation: Overview, Architecture and Applications", Optical Society of America, Vol.10, No.10, October 2018

[26] ”Network Management System Best Practices“, CISCO White Papers, Retrieved From https://www.cisco.com/c/en/us/support/docs/availability/high-availability/15114-NMS-bestpractice.html, Last Accessed on June 14, 2023.

[27] Emilia Rosa Jimson, Kashif Nisar and Mohd Hanafi Ahmad Hijazi, ” The state of the Art of SDN Issues in current Network Architecture and a Solution for Network Management Using SDN”, International Journal of Technology Diffusion, Architecture and Applications, Vol.10, Issue.3, September 2019.

[28] Kamal Benzekki et al, “Software-defined networking (SDN): a survey”, Security and Communication Networks, Wiley Online Library, Vol.9, Issue.18, February 2017.

[29] Slawomir KUKLINSKI and prosper CHERMOUIL,” Network Management Challenges in Software-Defined Networks”, IEICE TRANS.COMMUN, Vol.E97-B, No 1, January 2014.

[30] Jacob H. Cox et al,”Advanced Software-Defined Networks: A Survey”, IEEEAccess Journal, Vol.5.p.25487-25526, 12 October, 2017.

[31] Wael Honsy and Fouad Aly, “Generic Controller Adaptive Load Balancing (GCLAB) for SDN Networks”, Hindawi : Journal of Computer Networks and Communications , London,UK,Vol.2019, o1 December , 2019.

[32] Muhammad Reazul Haque, Saw Chin Tan et al, Zulfadzli Yusoff, Kashif Nisar , Ching Kwang Lee , Rizaludin Kaspin , Bhawani S. Chowdhry, Sameer Ali and Shuaib Memon” A Novel DDoS Attack-aware Smart Backup Controller Placement in SDN Design”, IAER: Annals of Emerging Technologies in Computing (ATEiC), London,UK,Vol.4, No.5 ,PP.75-92,20 December, 2020.

[33] Ahmed Abdelaziz , Ang Tan Fong , Abdullah Gani ,Usman Garba ,Suleman Khan ,Adnan Akhunzada ,Hamid Talebian and Kim-Kwang Raymond Choo ,” Distributed controller clustering in software defined networks”, PLoS ONE 12(4): e0174715, United States, 6 April, 2017.

[34] Bala Prakasa Rao Killi [a] and Seela Veerabhadreswara Rao [b],” Controller Placement in Software Defined Networks: A comprehensive survey”, Science Direct: Computer Networks 163 (20129) 106883 Bangalore, India, 13 August, 2019.

[35] Takahiro Hiramaya, Takiya Miyazawa, Abu Hena Al Mukutadir , Hiroaki Harai and Ved P. Kafle ,” Salience-based Distributed Controllers Placement in Software Defined Networks”, 2018 IEEE Global Communication Conference (GLOBECOM), Abu Dhabi, United Arab Emirates ,9-13 December, 2018.

[36] Sagarika Mohanty, Pateekshya Preyadarshini, Sampa Sahoo , Bibhudatta Sahoo and Srinivas Sethi ,” Metaheuristic Techniques for Controller Placement in Software Defined Networks”, TENCON 2019-2019 IEEE Region 10 Conference (TENCON), Kochi, India ,17-20 October, 2019.

[37] Bassey Isong and Nosipho Dladlu et al,” A Comprehensive Review of SDN Controller Placement Strategies”,IEEEAcess, Vol.8, pp 170070-170092, 14 September, 2020.

[38] Khushboo Kanodia, Sagarika Mohanty, Bibhudatta Sahoo and Kuldeep Kurroliya,” HPSOSA: A Hybrid Approach in Resilient Controller Placement in SDN”, 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE),Vellore, India, 24-25 February,2020.

[39] Jie Lu, Zhen Zhang, Tao Hu; Peng Yi and Julong Lan , “A survey of Controller Placement Problem in Software-Defined Networking”, IEEEAccess , Vol.7,P. 24290 – 24307, 07 March, 2019.

[40] Rong Chai et al, “Control plane delay minimization-based capacitated controller placement algorithm for SDN”, EURASIP Journal on Wireless Communications and Networking, 26 December, 2019.

[41] K Alhazmi, A Moubayed and A Shami ,“Distributed SDN Controller Placement Using Betweenness Centrality & Hierarchical Clustering”, DIVANet ’18: 8th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications, PP.15-20,Montreal, QC, Canada,25 October,2018.

[42] Chunzhi Wang, Gang Zhang, Hui Xu and Hongwei Chen , “An ACO-based Link Load-Balancing Algorithm in SDN”, IEEE 7th International Conference on Cloud Computing and Big Data, Macau, China,16-18 Nov.2016.

[43] Othmane Blial, Mouad Ben Mamoun,and Redouane Benaini ,” An Overview on SDN Architecture with Multiple Controllers ”, Hindawi: Journal of Computer Networks and Communications , London,UK,Vol.2016, 14 April , 2016.

[44] Esmaeil Amiri, Emad Alizadeh and Khalilollah Raeisi , “An Efficient Hierarchical Distributed SDN Controller Model”, 5th Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, Iran, 3 June, 2019.

[45] Mahmood Z. Abdullah, Nasir A. Al-awad and Fatima W. Hussein, “Evaluating and Comparing the Performance of Using Multiple Controllers in Software Defined Networks ”, I.J. Modern Education and Computer Science (MECS), P.27-34, Kuala Lumpur, Malaysia, 08 August, 2019

[46] Sarah Abdallah, Imad H. Elhajj, Ali Chehab and Ayman Kayss,” A Network Management Framework for SDN” 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS) , Paris, France,26-28 February, 2018.

[47] Victoria Huang, Qiang Fu, Gang Chen, Elliott Wen and Jonathan Har ,“BLACK: A Bindingless Architecture for Distributed SDN Controllers”,2017 IEEE 42nd Conference on Local Computers (LCN),Singapore, 9-12 October, 2017.

[48] Kanok Konglar and Yuthapong Somchit,” Load Distribution of Software-Defined Networking Based on Controller Performance”, 15th International Joint Conference on Computer Science and Software Engineering, Nakhonpathom, Thailand, 11-13 July, 2018.

[49] Ana Carolina O. Christofaro, Marcelo M. Carvalho and Daniel G. Silva , “Performance of Metaheuristic Algorithms for the Controller Placement Problem in SDN”, 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD),Pisa, Italy,14-16 September,2020.

[50] Shrinvas Petale and Jaisingh Thangaraj, “Failure Based Controller Placement in Software Defined Networks”, IEEE Transactions on Network and Service Management, Vol 7.Issue.1, 24 October, 2019.

[51] Kshira Sagar Sahoo, Pritish Mishra; Mayank Tiwary, Somula Ramasubbareddy and Balamuruga ,“Improving End-Users Utility in Software-Defined Wide Area Network System “, IEEE Transactions on Network and Service Management , Volume.17, Issue. 2, p.696 –707, 18 November, 2019.

[52] Abdul Hameed; Muhammad Wasim, “On the Study of SDN for Emulating Virtual LANs”, 2019 8th International Conference on Information and Communication Technologies (ICICT), Karachi, Pakistan, 16-17, November 2019.

[53] Qian Gao, Jiang Liu, Ningjie Gao and Tao Huang , “ A Dual Decomposition Method for Hierarchical Traffic Control in inter-DC WANs ” , 2019 IEEE International Conference on Communications in China (ICCC) , Changchun, China,11-13 August 2019.

[54] Themba Shozi, Sabelo Dlamini and Pragasen Mudali & Matthew O.Adigun , “An SDN Solution for Performance Improvement in Dedicated Wide-Area Networks”, 2019 Conference on Information Communications Technology and Society (ICTAS), Durban, South Africa, 06-08 March, 2019.

[55] Fernando W.P.T.N , Jayaweera M.N.R.S, Phatirana P.R.N, Dr.Maheshi B.Dissanayake and Anuradha Udunuwara, “Performance Analysis of Hierarchical Software Defined Networking (SDN) Controller Arrangements in SDNs” , Conference: IESL Young Members' Technical Conference, Colombo, Sri Lanka, October , 2021.

Declaration

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

Declared By:-

Name: Zenebe Kassie_____

Signature:_____

Date:_____

Confirmed by Advisor:

Name: Dr.Minale Ashagrie _____

Signature:_____

Date:_____