A Comparative Analysis between Centralized Routing and Distributed Routing in Multi-Hop Wireless Networks

by

Oluwasegun Abayomi Hassan

A thesis submitted to the

Faculty of Graduate and Postdoctoral Affairs

in partial fulfillment of the requirements for the degree of

Master of Applied Science in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario

December 2018

Abstract

A growing desire for granular network control, automation, virtualization and much more, has contributed to the emergence of Software Defined Networking (SDN) as a prominent research area. Though originally developed for wired networks, benefits of the centralized routing approach are now being leveraged for wireless network applications. These include SDN-based Multi-hop Wireless Networks (MWNs), as potential alternatives to traditional MWNs. This thesis presents a Software Defined Multi-hop Wireless Network (SDMWN) solution, with standard centralized routing characteristics and full mobility capabilities, evaluated against distributed routing in an equivalent traditional MWN architecture. Our emulation results, obtained with Mininet-WiFi, demonstrate a good degree of potential for SDMWN when operating under controlled (mobile) network conditions. By guaranteeing the availability of potential links between every node, SDMWN outperforms the traditional MWN, by about 15% and 65 ms, for Ping Success Rate and Round-Trip Time respectively. However, this comes at a relatively high cost of overhead.

To my dear parents, Mr. and Mrs. Hassan, and my best friend, Mercy Babalola;

you are my inspiration.

Acknowledgements

I remain most grateful to God for His grace and mercies so far in my academic pursuit, throughout my stay at Carleton University and most importantly for helping me choose the right career path to be successful.

In addition, I express my sincere and immense gratitude towards my supervisor, Prof. Thomas Kunz, for his quality guidance, constructive criticisms, and encouragements, over the course of my research. He always demanded the best results, and I am glad I could maximize my potential under his supervision.

I am equally grateful to the wonderful people I had the opportunity to interact with, throughout this entire period; including my fellow researcher – Afsane Zahmatkesh – for her technical assistance all through the early stages of working with the Mininet-WiFi emulator, and my lecturer – Prof. Chung-Horng Lung – for sharing his knowledge on the topic of Advanced Computer Communications, and the recommendation as well.

Table of Contents

List of Tables

List of Figures

List of Acronyms

ALM	Airtime Link Metric
AODV	Ad-hoc On-demand Distance Vector
AP	Access Point
CRP	Centralized Routing Protocol
HWMP	Hybrid Wireless Mesh Protocol
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
LLC	Logical Link Control
LLDP	Link Layer Discovery Protocol
LLDPDU	Link Layer Discovery Protocol Data Unit
MAC	Medium Access Protocol
MANET	Mobile Ad-hoc Network
MCC	Mobile Cloud Computing
MP	Mesh-Point
MWN	Multi-hop Wireless Network
NRO	Normalized Routing Overhead
OF	OpenFlow
OFDP	OpenFlow Discovery Protocol
OLSR	Optimized Link State Routing
PDR	Packet Delivery Ratio
PSR	Ping Success Rate

QoS	Quality of Service
RSU	Road Side Unit
RTT	Round-Trip Time
RWP	Random Waypoint
SDMWN	Software Defined Multi-hop Wireless Network
SDN	Software Defined Network/Networking
SNMP	Simple Network Management Protocol
SPOF	Single Point Of Failure
STA	Station
SSF	Strongest-Signal-First
TC	Topology Control
TCP	Transmission Control Protocol
TLV	Type Length Value
TTL	Time To Live
VANET	Vehicular Ad-hoc Network
WLAN	Wireless Local Area Network
WMR	Wireless Mesh Router
wmSDN	Wireless Mesh Software Defined Network
WSN	Wireless Sensor Network

Chapter 1

Introduction

1.1 Overview

Software Defined Networking (SDN) is generally regarded as a next-generation network paradigm, that has recently emerged to address various limitations of traditional networks, including manageability, flexibility, and extensibility [1, 2]. Traditional network, in this context, refers to networks that perform routing in a distributed manner, based on a vertical integration or coupling of a control plane in charge of making routing decisions and a data plane in charge of actual forwarding of messages [3]. This is why traditional networks are sometimes referred to as "inside the box" paradigms [4]. And while this distributed routing approach is most beneficial in terms of performance stability and network resilience, it also results in limited network control, increasing the cost and complexity of overall network management and policy implementation [3, 4].

Alternatively, to overcome the limitations of traditional networks, SDN employs a centralized routing approach, which is accomplished through a total (physical) separation of the control plane entities from the data or forwarding plane entities [2, 3, 4]. This "out of the box" approach allows for a more centralized and fine-grained network control, which

simplifies network management and policy implementation. Other SDN benefits include global network visibility and flexibility. A high-level illustration of the major difference in architecture between Software Defined Networks and traditional networks is shown in Figure 1.1 below:

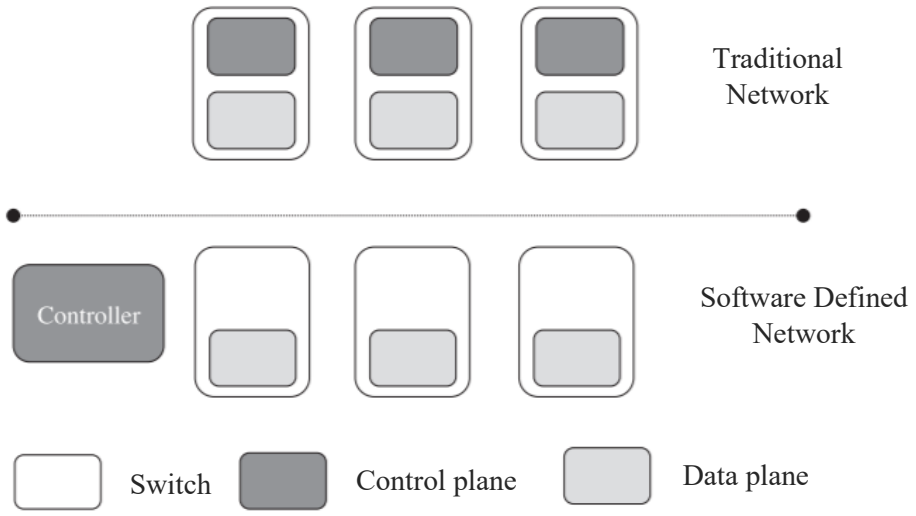


Figure 1.1: Architectural Difference in Network Paradigms [2]

From Figure 1.1, we can see the control plane is distributed and located within every switch in the traditional network architecture. The reverse is the case in the SDN architecture, where the control planes are completely taken out of the switches and replaced with a single controller or control plane, that is located away from the switches.

Furthermore, taking a closer look at the overall SDN architecture, it is made up of three separate major components, that are interconnected to communicate with each other via different APIs [2, 5]. These three components include the data plane, control plane, and applications, as shown in Figure 1.2 below:

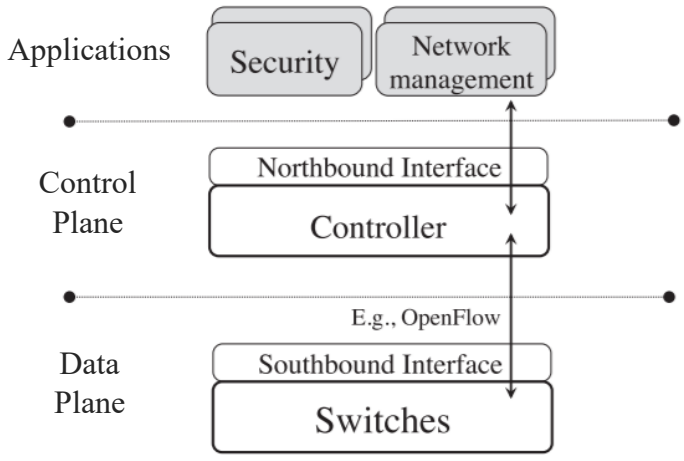


Figure 1.2: Components of SDN [2]

Figure 1.2 shows the data plane in the bottom layer, consisting of forwarding elements or devices like switches and routers, that perform the actual forwarding of data packets. In the middle layer, we have the control plane – which mainly consists of a logically centralized controller, equipped with global network visibility, for managing the entire network and controlling the devices in the data plane. The controller in the control plane communicates with the data plane via a southbound interface on the forwarding devices, shown in Figure 1.2. OpenFlow – an open protocol that supports programming of flow (packet forwarding) tables installed by the controller on forwarding devices – is by far the most widely used southbound interface [3, 5, 6]. Lastly, applications in the uppermost layer are end-user or business applications that run on the controller, to provide added network features such as security and network management, via the northbound interface [3, 5, 6].

It is worth stating that there are well-known limitations of SDN, which have largely been established in wired network architectures. These include a single point of failure (SPOF) issue associated with the (single) controller, affecting overall network resilience.

And there is a network overhead issue, resulting from the high number of control messages initiated by the controller and sent to appropriate parts of the network, for the controller to achieve global network visibility [4, 8, 9]. Nonetheless, aside from the few limitations that need to be addressed, the benefits and rewards of SDN seem to outweigh its limitations. This view is supported by the real-life adoptions of SDN in the enterprise networks of some communications technology industry leaders. For example, Google's B4 project, involving datacenter-to-datacenter WAN connection, highlights SDN's dynamic routing and traffic engineering capability to enhance WAN link utilization from about 30% up to 70% (with some links reaching 99% utilization) [3, 8, 9]. Other notable mentions include AT&T, Microsoft and Verizon, who have all adopted SDN in one way or another, to reduce costs, improve network utilization and service delivery to end users [3].

With the immense benefits and relative success of SDN, mostly attained in wired network environments, the technology is gradually gaining popularity in wireless networks as well. For example, SDN is currently being leveraged to address specific issues affecting Multi-hop Wireless Networks (MWNs). MWNs are wireless network types consisting of static and/or mobile nodes, where nodes can communicate directly with other nodes within a transmission range, and with distant nodes using one or more intermediate nodes [10]. MWNs may operate in ''ad hoc'' mode e.g. Mobile Ad hoc Networks (MANETs) – having self-configuring and self-organizing nodes, or in "infrastructure" mode e.g. Wireless Mesh Networks – having one or more entities, such as (fixed) Access-Points, that manage the network [7]. Issues generally affecting traditional MWNs include mobility management, optimal route computation, energy efficiency, Quality of Service (QoS) and so on [7, 10].

1.2 Research Problem and Motivation

Routing methods and protocols play a major role in the successful operation of MWNs. This is especially evident in the presence of external factors such as mobility, which brings about frequent network disconnections, link breakages, and topology updates. The main reason behind the selection of a particular routing method or protocol in MWN is to ensure the continuous provision of optimal path(s) for communication between wireless nodes, under different network conditions and with minimum network cost.

In traditional MWNs, the distributed routing approach is widely employed using protocols such as OLSR, which improve network resilience and fault tolerance to different conditions and factors that can affect network performance. However, such architecture is essentially less flexible and more difficult to manage in an administrative sense [1, 2]. This is mainly because network control or the control plane, for conducting routing/forwarding decisions, is shared across every node in the distributed MWN architecture. And therefore, network management, expansion, application update and routing policy implementation for the entire network becomes much more complicated and inefficient.

Furthermore, distributed routing can be inefficient in specific MWN applications such as VANETs – characterized by a high degree of mobility and high network density [10, 30]. An explanation for such inefficiency predominantly lies in the limited network visibility and routing information available to routing nodes. Consequently, in traditional MWNs, where routing is done in a distributed or collective manner among participating nodes, nodes often make routing decisions solely based on the limited routing information available, and this can negatively impact path selection and resource utilization.

The above-mentioned limitations of distributed routing, in the traditional MWN architecture, have promoted the search for a more efficient approach to routing in MWNs. Subsequently, this has led to various suggestions about SDN-based solutions, with a goal of leveraging the benefits of the centralized routing approach enjoyed with SDN, to either complement or completely replace distributed routing in traditional MWNs. Such benefits of the SDN-based solutions include network flexibility, scalability and manageability, due to the greater network control provided by the controller in the control plane, and global network visibility available to the controller [2, 3, 4]. The benefits listed above typically outweigh less-considered downsides of the centralized routing approach, such as the high network overhead and SPOF issues, associated with the SDN controller, currently beyond the scope of our research.

Furthermore, as exciting as the SDN-based MWN concept sounds, we believe there are still challenges that must be overcome, for centralized routing in SDN-based MWNs to be widely embraced as a reliable upgrade or alternative to distributed routing in traditional MWN. This cautiousness is underscored by the fact that centralized routing was originally designed for wired network architectures and its mode of operation is best suited to such environment, judging by its relative success so far. With the physical separation of the control plane from the data plane entities, for centralized routing, there is a major challenge of providing a stable and reliable communication technique between the two separated entities in the wireless environment. This is because any proposed technique or solution must also provide support for effective topology discovery and mobility management – both crucial to the successful operation of MWNs in general.

Although different SDN-based MWN architectures – addressing this plane-to-plane communication issue – have been put forward by various researchers, the practicality and real-life application of most of these solutions are being called into question. While some authors completely fail to factor in mobility in their solutions, by using wired connections for plane-to-plane communication, others only consider partial mobility events in which nodes connected directly to the controller are static. And for the remaining SDN-based MWN proposals with full network mobility, there is an unrealistic assumption that the controller in the control plane will continuously remain within transmission range, or is connected to, every wireless node in the data plane. With mobile interconnected devices fast becoming the norm today, many of these solutions appear less than ideal and do not always fortify the argument to fully adopt centralized routing in current and future MWNs

Finally, there is also a lingering concern about whether current concerns about the centralized routing approach, in existing SDN-based MWN architectures, truly allow for thorough and objective performance comparisons with the distributed routing approach in the traditional MWN architectures.

1.3 Research Objectives

The overall goal of this thesis is to investigate whether centralized routing should be widely employed as an alternative to distributed routing in MWNs, with an end goal of improving network control and overall performance. To achieve our objectives with a high degree of fairness and confidence, as much as possible, major steps to be followed are listed below:

- Identify current limitations of distributed routing in traditional MWNs, as it impacts network management and performance.
- Identify significant limitations of existing SDN-based MWN solutions, that impact network performance, evaluation, and the feasibility of real-life application.
- Design an SDN-Based MWN architecture, that primarily addresses the practicality and mobility issues in existing SDN-based MWN solutions.
- Implement the proposed SDN-Based MWN architecture in Mininet-WiFi, using a POX controller, OpenFlow-enabled entities and conventional network entities; all operating with standard network features and parameters.
- Implement a corresponding traditional MWN architecture in Mininet-WiFi, having OLSR daemons running independently on conventional network entities only; all operating with standard network features and parameters as well.
- Study and verify key network behaviours and activities of each MWN architecture, to provide a groundwork for our comparative analysis and ensure as much fairness and clarity as possible, during performance evaluations of both MWN architectures.
- Subject both MWN architectures to rigorous (mobile) network performance tests, under identical conditions.
- Compare and analyze performance results of both MWN architectures and identify the extent to which differences and/or similarities in results can be attributed to the different routing approaches or MWN architectures, and possible roles played by the protocol parameters and resulting network activities and behaviours.

1.4 Thesis Contributions

We are contributing to the state-of-the-art regarding SDN-based MWNs as follows:

- Proposing a truly practical SDN-Based MWN architecture, that can be incorporated in current or future MWNs, to promote a wide range of real-life applications, including the emerging areas of Mobile Cloud Computing (MCC) and Intelligent Transportation System (ITS). The practicality of our architecture is in the presence of fully mobile nodes with similar wireless properties, including the controller, in a completely (multi-hop) wireless network.
- Addressing existing plane-to-plane communication issues and resultant challenges of topology discovery, mobility management and flow entry installation in SDN-based MWN solutions, by introducing the IEEE 802.11s WLAN mesh standard as an underlying protocol to support de-facto OpenFlow Discovery Protocol (OFDP) and OpenFlow in general. The relative advantage of this combination arises from the fact that the IEEE 802.11s mesh technology operates entirely in the MAC layer and is transparent to upper layer entities. Consequently, this limits IEEE 802.11s control traffic and processes during topology discovery to the backbone network or MAC layer, without interfering with OpenFlow operations and other upper layer entities. Besides, the amount of flow entry installations performed by the controller is substantially reduced through IEEE 802.11s (backbone) mesh discovery, peering and forwarding operations, which cause the controller to view sets of intermediate or multi-hop links in the backbone network as single hops or links.
- Complete a fair comparative analysis between centralized and distributed routing

in corresponding MWN architectures, to investigate whether the centralized routing approach is truly more beneficial than the distributed routing approach, and to what degree and circumstance any such advantage holds.

1.5 Organisation of the Thesis

The structure of the remainder of this thesis is listed as follows:

- Chapter 2 provides an in-depth review of the state of the art, closely related to this research. This includes relevant backgrounds on OLSR and OpenFlow protocols, in the distributed and centralized MWN architectures respectively.
- Chapter 3 presents a practical SDN-Based MWN architecture, including its mode of operation, all aimed at addressing the limitations of existing SDN-based MWN solutions, and to be compared with our traditional MWN architecture.
- Chapter 4 provides detailed descriptions of our experiment designs and scenarios, emulation tool, implementations, measurements, and procedures for conducting network performance tests in this research.
- Chapter 5 presents Mininet-WiFi emulation results which provide the groundwork for comparative analysis in this research.
- Chapter 6 concludes the thesis, with final remarks on the network performance of the centralized and distributed MWN architectures that are compared, followed by recommendations and possible future work, for further advancements in this topic.

Chapter 2

Review of the State of the Art

2.1 Introduction

As established in the previous chapter, our main goal is to introduce an SDN-based or centralized routing solution in a practical MWN architecture, to be tested and evaluated against distributed routing in the corresponding traditional MWN architecture.

In this chapter, we generally provide a literature review of SDN-based solutions in MWNs. However, we initially begin in Section 2.2 by explaining the operation of OLSR – a distributed routing protocol, chosen for the traditional MWN architecture, to be compared with centralized routing in our SDN-Based MWN architecture. OLSR is the representative protocol for the distributed routing approach because it is one of the most important and effective proactive routing protocols, which makes it very popular among researchers [11]. Similarly, like the controller in the centralized routing approach, OLSR uses hop-count or shortest-path metric for route computation. The rest of this chapter is organized as follows.

In Section 2.3, we discuss OpenFlow operations for centralized routing. In Section 2.4, we conclude this chapter by reviewing SDN-based MWN proposals and closely related works about centralized routing versus distributed routing in MWNs. This highlights the

perspective of various researchers, who have previously evaluated and compared network performances of the centralized routing approach in SDN-based MWNs with the distributed routing approach in traditional MWNs.

2.2 Distributed Routing with OLSR

Optimized Link State Routing (OLSR) is an enhanced version of the classical link state algorithm, specifically developed for distributed routing in traditional MWN architectures like MANETs [12]. "OLSR is designed to work in a completely distributed manner and does not depend on any central entity" [12]. Therefore, unlike the centralized routing approach, each node in the distributed routing approach is responsible for building its own routing table, used for local routing, based on partial topology knowledge. An example of a distributed MWN architecture is shown in Figure 2.1 below:

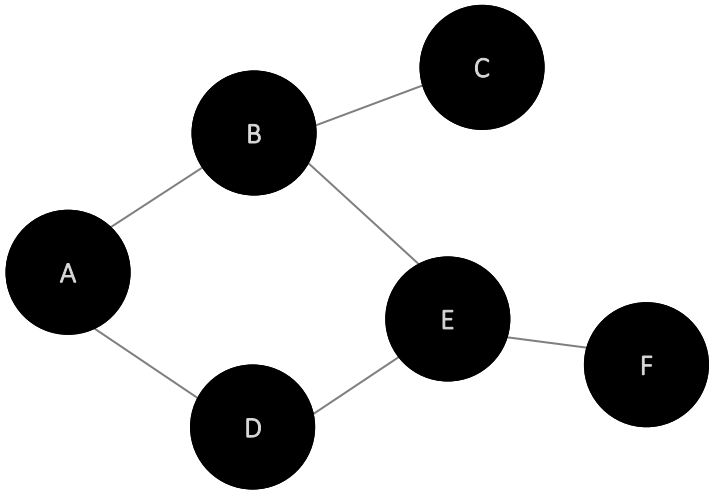


Figure 2.1: A Simple MANET

In Figure 2.1, nodes A to F are all independently running OLSR, allowing each node to have separate tables, which contain sufficient routing information such as destination node, next-hop, and number of hops; to reach (distant) nodes in the network.

OLSR is also a proactive routing protocol because link or neighbour information is periodically exchanged between adjacent nodes, enabling each node to maintain up-to-date information about its 1-hop and 2-hop neighbours. Moreover, up-to-date topology information is regularly dispersed across the entire network, for nodes to maintain their routing tables. This proactive state ensures that existing routes between nodes are always available when required. However, to avoid unnecessary flooding events, reduce redundant retransmissions and facilitate the efficient dissemination of topology information messages across the network, OLSR introduces the concept of Multipoint Relays (MPRs) [13]. MPRs are sets of nodes chosen by other nodes, within a 1-hop neighbourhood, to minimize the number of active relays or broadcasts needed to reach 2-hop neighbours [13, 14]. When a node selects another node as its MPR, it is called an MPR Selector. Similarly, more than one node can select the same node as MPR and an MPR can be an MPR selector as well.

From Figure 2.1 again, showing a wireless network consisting of 6 nodes (A to F), we can estimate that node B is an MPR for nodes A, C and E – MPR selectors. A major factor in this selection lies in the fact that node B is the only node that provides a path to node C, thereby making it an ideal candidate to be the MPR for nodes A, C and E, which are all within the 2-hop neighbourhood of each other. This is a perfect example of more than one node (A, C and E) selecting the same node (B) as their MPR. Similarly, node E is the MPR for nodes D and F (MPR selectors) because it is the only node that provides a

path to node F. This is also a perfect example of a node (E) acting as an MPR for other nodes (D and F) and an MPR selector, with another node (B) as its MPR. Consequently, with MPRs (B and E) serving different regions of the network, the dispersal of topology information is optimized, as the number of nodes relaying such broadcasts across the entire network is significantly reduced from 6 to 2.

OLSR uses two different types of control messages: Hello and Topology Control (TC) messages [11, 14, 15]. Hello messages are exchanged periodically among neighbour nodes to sense links to neighbours, determine the link status and signal MPR selection [12,15]. This message is locally exchanged by neighbour nodes only and is not forwarded to distant nodes [11]. On the other hand, TC messages are periodically broadcasted across the entire network. TC messages are used by MPRs to broadcast information about own advertised neighbours, which typically includes at least the MPR Selector list [12, 15]. This message helps signal link state information to all nodes in the network. And upon receiving this message, a node can construct a partial network topology, used for route computation to all other nodes in the network.

Moreover, only MPRs in the network are allowed to broadcast and forward TC messages while other nodes (presumably MPR selectors) basically use these TC messages to build and update their routing tables based on partial topology knowledge. Using Figure 2.1 as a reference, examples of node B's Hello message and node E's TC message are shown in Table 2.1 and Table 2.2 below:

Table 2.1: Example of Hello Message

Node	Neighbours
B	A, C, E

Table 2.2: Example of TC Message

Node	MPR Selectors
E	D, F

Table 2.1 and Table 2.2 only show a subset of node B's Hello message and node E's TC message respectively. These are subsets because ideally, other relevant message fields are included in (complete) Hello and TC messages. Table 2.1 shows all adjacent neighbours or directly connected nodes (A, C and E) of node B. Table 2.2 shows node E's TC message which contains its MPR selectors – D and F.

There are four major processes involved in successful OLSR operation: neighbour detection, MPR selection, topology discovery and route computation.

2.2.1 Neighbour Detection

Neighbour detection is the result of link sensing, which is achieved through the periodic exchange of Hello messages between adjacent nodes. In the case of multiple interfaces, separate Hello messages are generated for each interface [12]. This process is usually repeated every 2 seconds during which each node advertises its entire 1-hop neighbourhood to adjacent nodes that fall within the same 1-hop neighbourhood. When a node receives

Hello messages about the 1-hop neighbourhoods of its adjacent nodes, it is able to gather information not only about its 1-hop neighbours but 2-hop neighbours as well.

Furthermore, during this process, a node can detect neighbour changes and movements which are important in the case of mobility. Apart from a neighbour's interface address, other information obtained in the course of neighbour detection include: link type (symmetric, asymmetric or lost), neighbour type (symmetric, MPR or not a neighbour), how frequent the neighbour sends Hello messages, willingness of the neighbour to act as an MPR, and information about its own neighbour [12, 13, 14].

2.2.2 MPR Selection

The purpose of the MPR selection process is to minimize the overhead that comes with every node flooding messages across the entire network [12]. Therefore, by selecting a subset of nodes to broadcast such messages to specific regions in the network, flooding is mostly reduced. A node selects a set of nodes within its symmetric 1-hop neighbourhood, called MPR set, which can re-transmit its messages [12]. This set is selected in a way that ensures symmetric 2-hop nodes are covered at all times.

MPR selection is carried out using 2-hop neighbour information, which is obtained from Hello messages exchanged during neighbour sensing and this same process involving the periodic exchanged of Hello messages is used for signaling MPRs [15]. With this mechanism, each signaled or selected MPR also maintains an MPR selector set which contains information about the set of nodes which have selected it as their MPR [15]. Also, MPR set recalculation is performed when network topology changes are detected.

2.2.3 Topology Discovery

Each node gradually builds its routing tables, containing partial network topology mostly using TC messages broadcasted by MPRs across the entire network. These TC messages are periodically sent (about every 5 seconds) by nodes to advertise own links or topology information within the network and this must include the MPR selector set [14].

Also, Hello messages obtained from adjacent neighbours provide information about nodes within the 2-hop neighbourhood. With this combination, each node gathers sufficient topology information required for building an independent routing table. Sequence numbers are used as timestamps to identify new or updated messages and avoid looping of broadcasted (TC) messages [14]. And when changes occur in the topology, the routing table is reconstructed using new information from the Hello and TC messages.

2.2.4 Route Computation

After partial network topology discovery, using TC and Hello messages, each node is equipped with information about links to its neighbours and links between all MPRs and MPR selectors in the network. A combination of these sets of information is used for route calculation in the routing table. Topology information in the routing table includes: destination node/address, next-hop node/address and number of hops to the destination node [14]. Routes between source and destination nodes in the routing table are determined based on a shortest part algorithm; more so in large networks. Again, using Figure 2.1 as a reference, an example of the routing table for node A is illustrated in Table 2.3 below:

Table 2.3: Example of OLSR Routing Table

Destination	Next-Hop	Number of Hops
B	B	1
C	B	2
D	D	1
E	B	2
F	B	3

Table 2.3 shows how node B is predominantly used as the next-hop node to reach nodes C, E and F from node A. In node B's role as an MPR for nodes A, C and E (which is also MPR to nodes E and F), topology information about the links between node B and its MPR selectors (A, C and E) are included in TC messages available to node A. Such information, combined with Hello messages exchanged with nodes B and D, are used by node A to build a routing table, similar to Table 2.3.

2.3 Centralized Routing with OpenFlow

Centralized routing is the routing approach in SDN-based architectures, that is achieved by decoupling the control plane entities from the data plane entities. OpenFlow (OF) is the standardized protocol responsible for providing a means of communication between the separated planes [3, 5, 6, 16, 17]. Therefore, to facilitate plane-to-plane communication in such an architecture, OpenFlow-enabled entities (controller and forwarding devices) are

required in appropriate planes. The types of messages usually exchanged during the plane-to-plane communication include control messages for topology discovery, information on received/sent data packets, actions to be performed on specific flows, flow statistics etc [4]. These messages can be grouped into three main categories and they include:

- Controller-to-switch messages generated by the controller during processes such as topology discovery in the data plane and for other network management purposes [4, 5]. This type of message is usually encapsulated and sent out as a Packet-Out message by the controller.
- Asynchronous messages initiated by forwarding devices in the data plane to update or inform the controller about internal events such as packet arrivals or other external such as topology changes or mobility events [4, 5]. This type of message is usually encapsulated and sent as a Packet-In message to the controller.
- Symmetric messages often generated without solicitation either by the controller or forwarding devices. This includes echo messages exchanged between the controller and a switch to monitor the link-state of a controller-switch connection [4, 5].

In addition, OpenFlow is a flow-based protocol that requires OpenFlow-enabled devices in the data plane to hold one or more flow tables consisting of flow entries, each associated with an action or set of actions that determine the way packets or flows are handled [5]. Flow tables in the data plane or forwarding devices are mainly installed and manipulated by the controller, responsible for adding, updating, and deleting flow entries; based on its global view of the network. This process can occur reactively, such as when the controller receives a packet/packet information from a forwarding device, or proactively, based on

controller policy implementation or specification [2]. A flow table entry on a forwarding device consists of three key fields identified as follows:

- Match fields for matching an incoming packet based on determinants, such as the packet's header/field (e.g. source and/or destination MAC/IP address), ingress port, and optionally packet's metadata [2, 4, 5].
- An action set or instruction field to be enforced after a match. The set of actions that can be performed on a matched packet include modifying a packet field, forwarding the packet out of a port or number of ports for delivery within the data plane or to the controller in the control plane for assessment, and dropping the packet based on existing flow rules [2, 4, 5].
- Counters that collect flow statistics, such as: number of transmitted packets, number of bytes, duration of the flows and so on. [2, 4, 5]. This information can be used by the controller to make future decisions affecting the flow tables.

In addition, since forwarding devices can hold more than one table, linked in an increasing order, a received packet with no match in the first flow table may be passed on to the next (existing) flow table, to search for a possible match. Similarly, an unmatched packet/packet information in the data plane can be forwarded to the controller or dropped entirely [2, 5]. If the packet is forwarded to the controller, the controller can either drop the packet with no further action or install a new flow entry or flow entries on the forwarding device to determine how similar future packets should be handled [5].

An illustration of how a received packet is generally handled by a forwarding device, based on centralized routing with OpenFlow is shown in Figure 2.2 below:

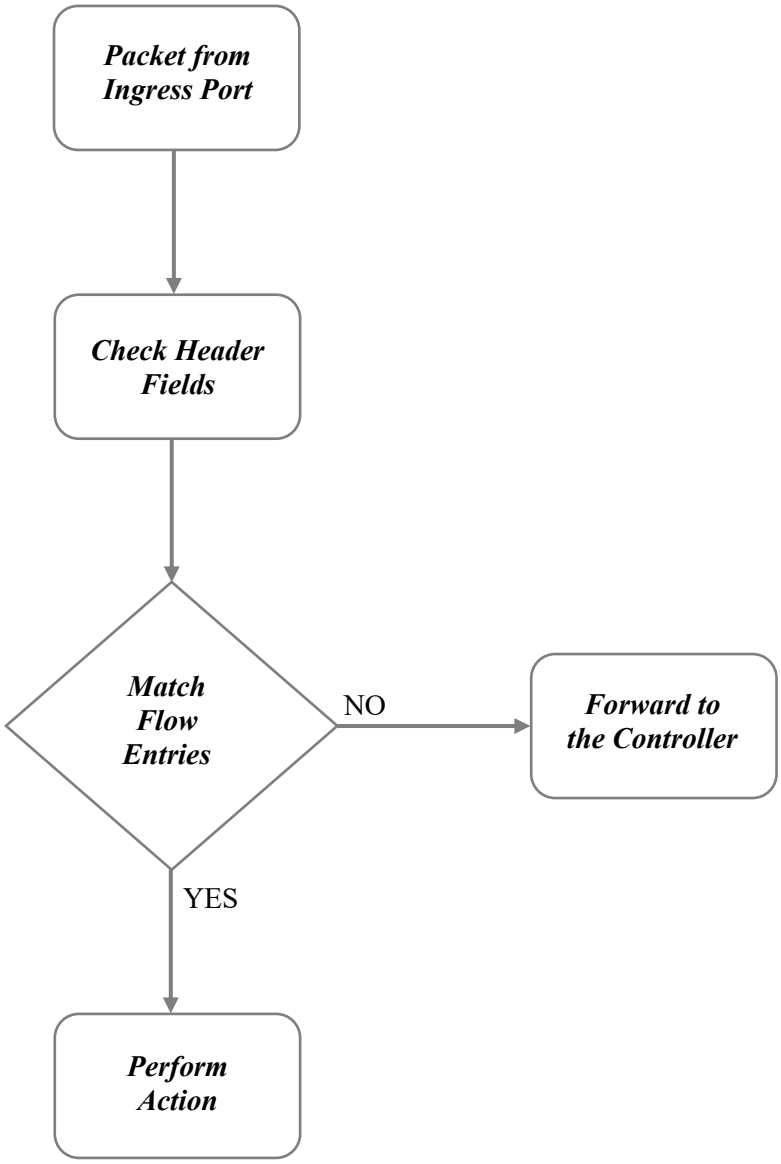


Figure 2.2: Simple Packet Forwarding with OpenFlow

Figure 2.2 is a flowchart showing how a packet is processed during centralized routing. The header field of the packet is initially extracted, then relevant fields are checked against flow table entries for a match. Finally, a specified action is performed if a match is found and if not, the packet is usually sent to the controller to determine the next step of action.

2.4 Related Work on SDN-Based MWN

Software Defined Wireless Networking is a research area that has recently become popular in both academia and industry. Likewise, there have been several studies and proposals, to demonstrate the application of SDN in MWNs and for addressing network issues, such as mobility management, peculiar to such wireless network architectures [10].

However, with SDN originally designed for wired networks, there are challenges involved in the integration of this network paradigm with existing MWN architectures. The major challenge facing most SDN-based MWN architectures relates to the provision of a secure and reliable plane-to-plane communication mechanism between the controller in the control plane and OpenFlow-enabled wireless entities in the data plane [10, 22]. Two main approaches have been proposed to address this issue and they include: an in-band control style and an out-of-band control style [10, 22]. The out-of-band control style is more widely adopted, due to its reliability, though, it suffers from scalability and flexibility issues.

Essentially, each style has a different architecture that defines how the control and data traffic is handled during network operations. Control traffic refers to messages flowing between the controller in the control plane and devices (wireless nodes) in the data plane. In most SDN-based MWNs, control messages are generated by an OpenFlow Discovery Protocol (OFDP), which leverages a Link Layer Discovery Protocol (LLDP) to perform topology discovery [18]. On the other hand, data traffic comprises actual data packets, exchanged between nodes, in the data plane only. For the out-of-band control style, two different channels are provided, each for conveying the control traffic and data traffic separately [10, 22]. Alternatively, the in-band control style involves using the same channel

to convey both sets of traffic [10, 22]. Examples of the two types of architectures are shown in Figure 2.3 and Figure 2.4 below:

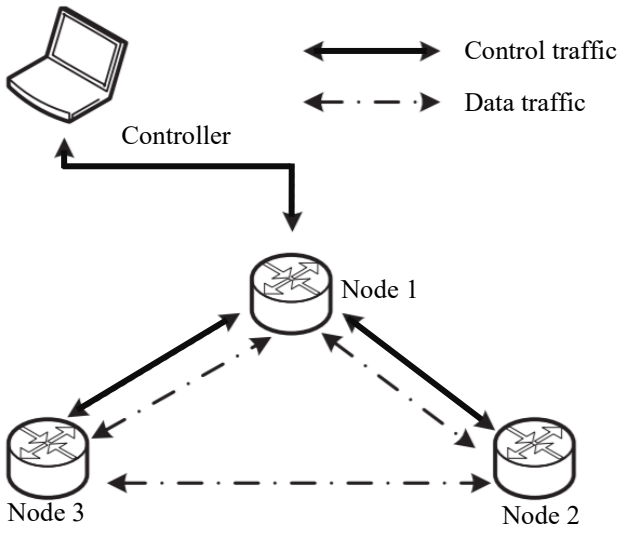


Figure 2.3: In-band Control Network Architecture [22]

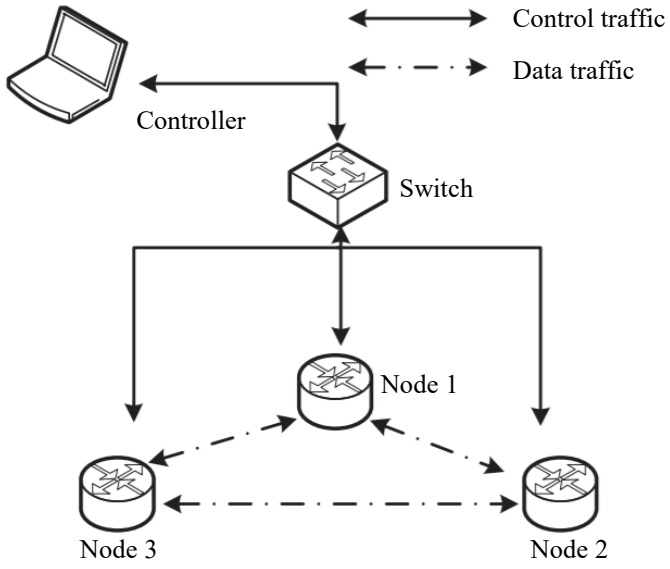


Figure 2.4: Out-of-Band Control Network Architecture [22]

Figure 2.3 shows the architecture of an OpenFlow in-band control network, where the control traffic/signal and data traffic share the same wireless channel. However, in Figure 2.4, a special provision is executed, using an OpenFlow-enabled switch to provide wired connections, which completely isolate control traffic from data traffic, in the OpenFlow out-of-band control architecture. While the (wired) out-of-band control style approach can provide reliable plane-to-plane communication, it is suited to static wireless networks and less appropriate for dynamic MWNs.

Therefore, the selection of a control style should be carefully considered when designing an SDN-based MWN architecture, especially when such an architecture is to be compared against a traditional MWN architecture. To investigate and in most cases lend support to the idea that centralized routing in SDN-based MWN architectures should be widely employed in modern wireless networks, either to complement or replace distributed routing in traditional MWN architectures, the following concepts, performance evaluations and comparative analyses have been put forward by various researchers:

In [19], a centralized routing solution is proposed to provide a shortest path and disjoint multipath routing between wireless nodes, while minimizing energy consumption and maximizing network lifetime. A controller, physically placed in the middle of this SDN-based MWN architecture, is directly connected to wireless nodes within transmission range – typically one or two hops away. During topology discovery, control messages from the controller are initially disseminated to wireless nodes across the entire network, via broadcasts. Subsequently, an appropriate broadcast path is used by each node as a reverse path to send node information back to the controller. The controller uses this information

to generate a global view of the network and maintain the residual-energy status of every node in the network. For data communication, shortest path computation is performed by the controller taking hop count and residual energy into consideration. Tests carried out by the authors of [19] show the proposed centralized routing approach consuming less energy per packet and experiencing lower end-to-end delay under static network conditions compared to distributed routing protocols (e.g. OLSR). However, a major limitation of this SDN-based MWN architecture is the flooding process used during topology discovery as this approach may not be efficient in large networks due to high network overhead.

In addition, the authors of [19] applied the exact same SDN-based MWN solution in [20] to Wireless Sensor Networks (WSN). With network lifetime being the most critical issue affecting sensor nodes [29], the proposed centralized routing solution is shown to significantly reduce average energy consumption per packet and subsequently improve network lifetime, compared to the distributed routing protocols including OLSR. As well, this SDN-based MWN shares similar limitations with its predecessor in [19].

Unlike [19] and [20], the authors of [21] offered a hybrid solution called Wireless Mesh Software Defined Network (wmSDN [21]) – an integration of Software Defined Networking (SDN) in a Wireless Mesh Network (WMN). This hybrid architecture tries to take advantage of existing SDN benefits like global network visibility while attempting to address limitations such as the SPOF issue, by employing OLSR as an underlying protocol to handle control traffic and topology discovery. The inclusion of a distributed routing protocol like OLSR plays an important role because the wmSDN architecture contains a controller that is connected to only one of the Wireless Mesh Routers (WMR). Therefore,

OLSR is required to work together with the controller, to facilitate communication between the controller and WMRs that comprise both OpenFlow and OLSR daemon technology in a single device. However, communication between the WMRs or data forwarding in the wmSDN architecture is solely directed by the controller except for periods when/if the controller fails and then OLSR steps in as a backup mechanism to route data traffic. This key process is achieved using separate IP subnets for control traffic and data traffic. The goal of wmSDN is extremely limited in scope and application to load balancing between internet gateways. It compares TCP goodput achieved while using centralized routing in the proposed hybrid MWN architecture and traditional OLSR, in a client/server-mesh network. Although results obtained in a static network show SDN performed considerably better than traditional OLSR routing protocol, this hybrid solution will most likely increase network overhead and complexity which OpenFlow already suffers from.

The authors of [22] argue that, while the primary objective of traditional routing protocols in distributed MWN architectures is to provide optimal communication paths between the source and destination nodes, there lies the possibility that data traffic will continually be pushed along a wrongly-perceived optimal path, even after such path becomes congested. Since network congestion can contribute to delay, packet loss, low throughput and so on, an SDN-based congestion-aware Routing algorithm (SDNR [22]) is proposed as a solution. By introducing a new metric called link saturation, the controller in the centralized routing architecture constantly monitors bandwidth, traffic size or congestion on every active link and determines whether to reroute new traffic to non-active or less congested paths. And to maintain up-to-date information on link saturation state in

the wireless network, relevant information regarding the congestion state of routes is periodically conveyed to the controller by nodes (forwarders [22]). For communication between the controller and the forwarders, the SDNR architecture employs the out-of-band control (wired) network which the authors consider being more reliable and efficient than the in-band control approach. With each wireless node directly connected to the controller via a wired link, topology discovery is easily achieved in this SDN-based solution. Based on end-to-end network performance tests and analysis done by the researchers, centralized routing with SDNR records higher throughput and packet delivery ratio values in a (static) wireless network unlike distributed routing with OLSR. However, not too surprising is the fact that OLSR demonstrated lower routing overhead statistics as traffic size increased, with the periodic updates concerning link saturation state in SDNR being a contributing factor. Also concerning is the existence of wired links in the SDN-based MWN architecture that hinder mobility.

In [23], a centralized routing approach is proposed for efficient data communication between wireless nodes in VANETs, where traditional routing protocols can be very susceptible to high packet loss and delay, due to rapid topology changes in such network [30]. The Centralized Routing Protocol (CRP [24]) promises effective topology discovery and shorter network convergence time, based on the controller's ability to collect vital network information to generate a global network view, used for optimal path computation. A new routing metric called minimum optimistic time (MOT [24]) is also introduced. MOT is based on dynamic network density and used to derive minimum packet transmission time between two vehicular nodes by predicting future movements of nodes in the VANET

environment. For their MWN architecture, the authors of [23] presented a very practical model, that not only supports mobility but also preserves the unique qualities of SDN. To achieve this, network traffic is grouped into vehicle-to-vehicle communication (via short-range WLAN technology) and vehicle-to-base station communication (via long-range WiMAX technology). During centralized routing, in this case, the highly mobile vehicular nodes communicate directly with one another or through appropriate (static) base stations which are connected via wired links to the controller. Performance tests show CRP outperformed distributed routing with protocols like OLSR in terms of packet delivery delay time and routing overhead. This SDN-based MWN architecture is best-suited for large networks, with high node mobility and node density. Although, there are concerns regarding the novel routing metric, MOT, which is based on mere assumptions and may not always guarantee the optimal path selection.

In [24], another group of authors presents an SDN-based solution for VANETs that is quite similar to that of [23]. The main goal here is leveraging the benefits of SDN to explore and eventually increase the application and efficiency of VANETs. The authors also suggest that incorporating centralized routing in VANET can tackle problems of interference while improving channel usage and other wireless resources. There are three major components that make up this MWN architecture: the controller, wireless (vehicular) nodes and Road Side Units (RSU) or access points. Similar to the SDN-based MWN architecture in [23], data traffic involving communication between vehicular nodes is conducted via WLAN for high bandwidth wireless connection while control traffic involving node-RSU-controller communication is conducted via WiMAX for long-range

wireless connection. Additionally, each vehicular node is equipped with a local SDN agent, which facilitates a backup distributed routing mechanism, in case the controller ever fails or becomes unreachable. In their most relevant analysis comparing centralized routing to distributed routing, the proposed centralized routing solution demonstrates a higher Packet Delivery Ratio (PDR) than distributed routing with OLSR where the controller and RSUs were absent.

A group of authors from [24] also explored the potential of an SDN-based Mobile Ad hoc Network (MANET) in [25]. The goal of the paper is to present a feasible and efficient approach to integrating centralized routing in MANETs, which could eventually form the basis of an SDN-based Mobile Cloud architectures. With the emergence of cloud computing technology, the integration of Mobile Cloud Computing (MCC) with wireless networks to interconnect mobile devices is viewed by the authors as the next step in the development process. A major difference in the SDN-based MWN architecture in [25] compared to [24] is that the RSUs are removed completely while vehicular nodes are being replaced with OpenFlow-enabled switches. The switches are wirelessly connected to the controller, with an assumption that they all fall within its transmission range. Nonetheless, WLAN and WiMAX technologies are still employed for data traffic and control traffic respectively while the backup routing mechanism is also preserved in [25]. PDR performance was superior in the centralized routing solution, compared to distributed routing with OLSR. However, a major limitation of this SDN-based MWN architecture is a lack of scalability and flexibility because it relies on a condition that all (mobile) wireless nodes will continuously fall within transmission range of the controller.

In [26], the idea is to incorporate a controller in a Wireless Mesh Network (WMN) architecture to assist or improve distributed routing operations. This hybrid architecture, called OLSR_SDN [24], combines centralized routing in SDN with distributed routing in OLSR, to address the drawbacks of distributed routing while taking advantage of SDN benefits, such as fast network convergence time and optimal path selection. The authors of [26] believe that the presence of the controller can assist or improve the performance of traditional protocols like OLSR without necessarily replacing it altogether. Also, this hybrid MWN architecture employs an out-of-band control approach by using wireless nodes equipped with two interfaces – operating in different channels to transmit data traffic and control traffic separately. This hybrid architecture consists of wireless routers, directly connected to the controller. Most notable is that the total WMN area is divided into clusters or levels, to minimize routing overhead in the network which is a similar routing method to Fisheye State Routing (FSR) [31]. In this method, wireless nodes within a cluster can communicate with each other via OLSR – using local routing information. Alternatively, communication between nodes in different clusters is handled by the controller based on its global network view. Performance results show that the proposed hybrid MWN architecture outperforms OLSR in terms of PDR, throughput and routing overhead in a static network. However, just like in [25], this hybrid MWN architecture is less dynamic and scalable as there is no guarantee that all wireless nodes will stay connected to the controller as the network area increases while the introduction of (hierarchical) clusters may add an extra layer of complexity to the network.

To address the subject of mobility, overlooked in [26], the same authors presented an extended SDN-based MWN solution in [27], where routing operations are completely centralized, and the idea of clusters is completely abandoned. Notwithstanding, some parts of the SDN-based MWN architecture in [26] are also preserved, such as the dual interface wireless node provision. It is also assumed that every (mobile) wireless node is within the transmission range of the controller. Performance results show centralized routing in the SDN-based MWN architecture outperforms distributed routing protocols, like OLSR, in terms of PDR, throughput and routing overhead. This is similar to the results obtained in [26], but this time in a mobile network scenario. This happens even as a similar dual interface feature is implemented for nodes in the traditional MWN architecture, to promote a fair comparison. Notably, scalability and flexibility issues still exist in this SDN-based MWN solution, based on the assumption that every node in the data plane will permanently fall within the transmission range of the controller.

Last but not least, a self-styled practical version of an SDN MANET is presented in [28], to minimize overhead and increase throughput. Similar to the SDN-based MWN architectures in some of the previously reviewed papers, switches here are equipped with dual wireless interfaces and the controller directly connects to each switch, via a wireless interface. However, a major distinction, in this case, is the addition of a third (wired) interface in each switch connecting to a host. End-to-end communications between the hosts are performed using intermediate switches, as directed by the controller. Performance test results show centralized routing outperforms OLSR, in terms of throughput. Again, as earlier highlighted in other proposals, this SDN-based MWN architecture lacks scalability

and flexibility, with the introduction of wired switch hosts and considering the assumption that every switch is permanently within the transmission range of the controller.

After reviewing the state of the art on the comparative analysis between centralized routing and distributed routing in respective MWNs, largely positive performance results from multiple authors support the argument to integrate centralized routing in modern MWN architectures. However, while some authors advocate the full replacement of the distributed routing approach with the centralized routing approach, others are cautiously proposing SDN-assisted or hybrid solutions instead.

A summary of the state of the art on SDN-based MWN architectures, discussed in this section, is provided in Table 2.4 below:

Table 2.4: SDN-based MWN Solutions

Proposal	Year	Contributions	Control Type	Topology Discovery	Mobile
[19]	2017	Minimize energy consumption and improve network lifetime in MWN	In-Band	Broadcasts & Reverse paths	No
[20]	2016	Minimize energy consumption and maximize network lifetime in WSN	In-Band	Broadcasts & Reverse paths	No
wmSDN [21]	2013	Hybrid protocol for traffic engineering/ load balancing with a fall-back routing mechanism	In-Band/ Out-of-Band (wired link)	OLSR	No
SDNR [22]	2016	Reduce congestion and improve throughput in MWN	Out-of-Band (wired link)	Neighbour discovery	No

CRP [23]	2015	Effective topology discovery and fast network convergence in VANET	Out-of-Band (wired link)	Neighbour discovery	Partial
[24]	2014	Hybrid protocol to reduce wireless interference and improve channel usage and other wireless resources	Out-of-Band (separate wireless standards or channels)	Neighbour discovery	Partial
[25]	2014	Hybrid architecture to facilitate Mobile Cloud Computing	Out-of-Band (separate wireless standards)	Neighbour discovery	Yes
OLSR_ SDN [26]	2016	Hybrid protocol to reduce control traffic, accelerate network convergence time, and guarantee optimal path selection	Out-of-Band (separate wireless interface and channels)	Neighbour discovery	No
[27]	2016	Improve network performance and scalability	Out-of-Band (separate wireless interface and channel)	Neighbour discovery	Yes
SDN MANET [28]	2017	Reduce network overhead and increase throughput in MANETs	Out-of-Band	Neighbour discovery	No

From Table 1, a key observation is the wide adoption of the out-of-band control style in 7 of the 10 reviewed SDN-based MWN architectures, most likely based on the simplicity and reliability of such an approach. Another key observation is a complete lack of mobility provisions in more than a half of the reviewed SDN-based MWN architectures, while just

a few have provisions for partial network mobility during which nodes connected directly to the controller are static. And two proposals with full mobility assume that the mobile nodes in the data plane will always fall within the transmission range of the controller, irrespective of the network size or area. Finally, there are 4 hybrid solutions that combine features of both centralized routing and distributed routing to encourage reliability.

2.5 Summary

In this chapter, we extensively reviewed the state of the art regarding the centralized routing approach in SDN-based MWN architectures against the distributed routing approach in traditional MWN architectures. We began by describing distributed routing operations with OLSR, including topology discovery and mobility management processes. During this discussion, we selected OLSR as the representative protocol for evaluating distributed routing in our traditional MWN architecture against centralized routing in our SDN-based MWN architecture – proposed in Chapter 3. Next, we discussed centralized routing with OpenFlow in general and briefly for SDN-based MWNs, as topology discovery and mobility management operations will be extensively covered in Chapter 3, as part of our proposed SDN-based MWN architecture. Finally, we reviewed relevant SDN-based MWN proposals that were compared against traditional MWNs and highlighted major limitations of such solutions, which we plan to address in the next chapter.

Chapter 3

Practical SDN-Based MWN

3.1 Introduction

An extensive review of the state of the art regarding SDN-based proposals for centralized routing in MWN was provided in Chapter 2. And though most of the reviewed centralized routing solutions were innovative and demonstrated plausible network performances, we found significant limitations in their corresponding architectures and modes of operations, which we plan to address in this chapter.

For instance, in solutions [19] and [20], the dissemination of broadcast messages across the entire network during topology discovery does not seem like an efficient approach or a long-term solution, particularly in large networks. Such a technique can only lead to additional overhead in SDN-based MWN architectures, where network overhead is an existing challenge [4, 7, 8, 9]. In SDNR [22] and CRP [23], wired links are employed to isolate control traffic in the out-of-band control network architectures. However, while the single-hop wired connection can eliminate interference and collisions between the control and data traffic and subsequently increase channel/resource utilization in the data plane, it is not suitable for dynamic wireless networks due to the lack of full mobility [10].

Likewise, the centralized routing solution under partial mobility in [24] has limited MWN applications. This is because fixed RSUs in the data plane – acting as gateways to the controller – via support single-hop wireless links, do not have any mobility provision. Moreover, to experience full mobility in an SDN-based MWN architecture, we expect at least every wireless node in the data plane to be mobile. Lastly, other centralized routing solutions like [25], OLSR_SDN [26], [27] and SDN MANET [28] suffer from practicality and scalability issues. This is due to the centralized routing architectures operating under the condition that every (mobile) node will always fall within the transmission range of the controller, irrespective of the network size.

Based on the practicality and mobility issues highlighted above, we believe that current centralized routing or SDN-based MWN proposals are far from ideal and do not adequately strengthen the argument to adopt the centralized routing approach in future MWNs. Likewise, these exiting issues in proposed SDN-based MWN architectures do not allow for fair comparisons with the distributed routing approach in traditional MWNs. Our conclusion, based on unfair comparisons, also takes hybrid solutions like wmSDN [21] into consideration. Such solution combines the two major routing approaches, under study, into a single MWN architecture.

Consequently, in an effort to address the limitations of existing centralized routing solutions, a practical version of the SDN-based MWN architecture is presented in Section 3.2. This new solution is identified as a Software Defined Multi-hop Wireless Network (SDMWN), and the practicality of its architecture is based on the existence of fully mobile nodes, in both control and data planes, in a completely wireless network.

In Section 3.3, we explain the centralized routing technique in SDMWN. Sections 3.4 and 3.5 provide detailed descriptions of topology discovery and mobility management in our SDMWN architecture respectively. For fair comparisons with distributed routing in our traditional MWN architecture, our centralized routing solution employs the proactive strategy for topology discovery and mobility management. Similarly, it uses the shortest path routing algorithm for path computation – just like OLSR.

3.2 SDMWN Architecture

With mobile interconnected devices fast becoming the norm today, it is crucial that the SDMWN architecture is equipped with fully mobile nodes to make provisions for unpredictable mobility scenarios, while also preserving significant features of SDN that make it impressive. To this effect, we are employing two different kinds of wireless nodes in an in-band control style network architecture. The nodes include OpenFlow-enabled Access-Points (APs) and Stations (STAs). A station is a wireless node with an IEEE 802.11 standard-compliant MAC and physical layer (PHY) interface while an AP is a special type of station with added functionalities for managing other stations that connect through associations [32, 33]. And in the case of the OF-enabled APs in the SDMWN architecture, these are simply OpenFlow switches equipped with full AP capabilities.

The SDMWN architecture also contains a controller component, operating from within a designated station in the network. This approach is based on the in-band control network architecture in Figure 2.3 and it is put in place to tackle the practicality and mobility limitations of other SDN-based MWN proposals. Moreover, this allows for a fair

comparative analysis between centralized routing in SDMWN and distributed routing in traditional MWN. A small-scale illustration of the described SDMWN architecture is shown in Figure 3.1 below:

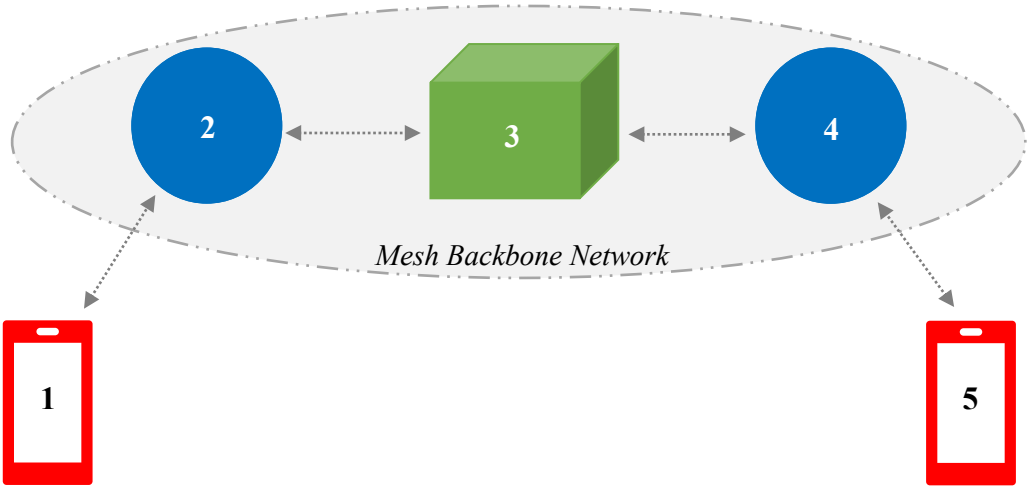


Figure 3.1: SDMWN Architecture

Figure 3.1 generally consists of 5 wireless nodes, including 2 APs and 3 stations, in the SDMWN architecture. Nodes 1 and 5 represent (end) stations (i.e. STA1 and STA5), nodes 2 and 4 represent (forwarding) APs (i.e. AP2 and AP4), and finally, node 3 represents the (centralized) controller-station component (i.e. STA3). Furthermore, AP2, STA3 and AP4 constitute a stand-alone wireless mesh backbone network, highlighted by enclosed area. This backbone network – based on IEEE 802.11s mesh technology – facilitates multi-hop communication and addresses the limitations of single-hop communication [32].

Consequently, APs in the SDMWN architecture communicate with one another, and with the controller, through the IEEE 802.11s mesh (backbone) network. These APs serve as intermediaries between associated stations and the controller in STA3. Even so, the controller, expected to have global network visibility, is primarily responsible for

managing communication between wireless nodes in the entire network, particularly amongst stations. To facilitate such end-to-end communication between stations, the OF-enabled APs provide OpenFlow integration and (controller) distribution services to associated stations. A high-level illustration of components in the SDMWN architecture and their connections is shown in Figure 3.2 below:

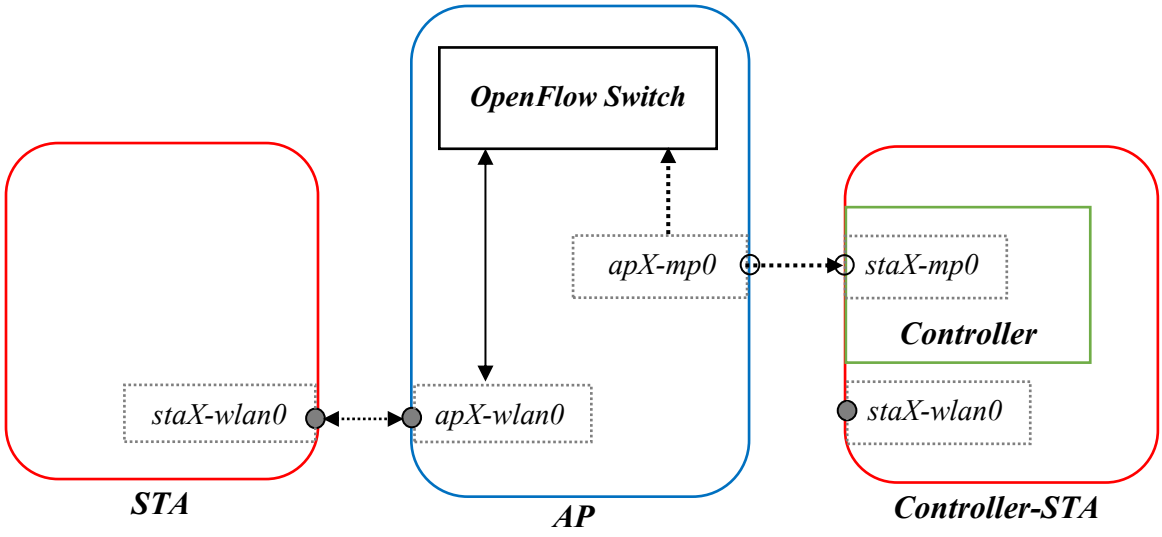


Figure 3.2: Components and Connections in SDMWN

Figure 3.2 shows three major components of the SDMWN architecture, which includes: a station, an access point and the controller-station component. It also shows the (in-band) controller located inside a designated station. The station typically has a single (WLAN) interface for associating with APs and communicating with other stations.

In contrast, the controller-station component and AP are each equipped with two interfaces – a mesh-point (MP) interface and a WLAN interface – for specific purposes. For the controller-station component, we have the staX-mp0 interface, connected to the

apX-mp0 interface of the adjacent AP. It is used by the controller to communicate with the AP, via the mesh backbone network. The letter "X" is any number (0, 1, 2, 3 …) that can be used to identify a station or AP. Then we have the staX-wlan0 interface that is used by the original station of the controller-station component to communicate with other stations, by associating with APs in the OpenFlow network. Similarly, for the AP in Figure 3.2, we have the apX-mp0 that connects to the staX-mp0 interface of the controller and is used by the OpenFlow switch of the AP to communicate with the controller and/or other APs (OpenFlow-switches), all in the backbone mesh network. The backbone network, based on IEEE 802.11s mesh technology, is necessary for transporting OpenFlow messages within areas of the multi-hop wireless network, to support OpenFlow operations. This mechanism facilitates reliable plane-to-plane communication between the controller and APs, required for topology discovery, mobility management, flow installations and so on. The other interface on the AP – apX-wlan0 – working in infrastructure mode, connects to the staX-wlan0 interface of the regular station (on the left-hand side) and is used by APs to connect with associated stations, and for multi-hop communication between stations. These interfaces (i.e. apX-wlan0 and staX-wlan0) do not operate in the mesh backbone network.

3.3 SDMWN Routing Approach

SDMWN mainly operates using the centralized routing method with OpenFlow, for data communication, via the controller and OF-enabled APs. However, the 802.11s mesh technology is also introduced through a backbone network to support the OpenFlow protocol, which is unaccustomed to the multi-hop wireless network environment.

Using Figure 2.2, Figure 3.1 and Figure 3.2 as references, if a source node, say STA1, has a packet destined for STA5, STA1 should forward the packet to AP2 – the connected AP. This is assuming that the destination node's IP and MAC addresses are both known to STA1 and have been included in the packet's header. However, if only STA5's IP address is known to STA1, STA1 will generate and send ARP broadcasts through the network, requesting for STA5's MAC address. The ARP request is initially received by AP2 which subsequently forwards it to the controller by default, based on existing flow rules, through the mesh backbone network. The controller analyses this message and broadcasts it to other parts of the network, including AP4 and STA5. This broadcast and forwarding process will continue until AP4 and subsequently, STA5 receives the ARP request. Finally, STA5 replies to STA1 with ARP replies containing its MAC address and then, STA1 can use this information to update its ARP table, for present and future use.

Once the MAC-to-IP association for STA5 is sorted out and all necessary information has been included in the packet's header, STA1 simply forwards the packet via its sta1-wlan0 interface to AP2. When AP2 receives the packet on its ap2-wlan0 interface, it extracts the packet's header fields and relevant fields are checked against flow table entries for a match. If a match is found in the flow table, a specified action or set of actions is performed. However, at an early stage and possibly being the first packet of its kind, it is highly unlikely that the packet will be matched. Based on existing flow rules concerning unmatched packets in the data plane, the default action is for the unmatched packet/packet information to be forwarded to the controller in the control plane. This plane-to-plane communication is achieved through (transparent) IEEE 802.11s operations in the

mesh backbone network. Therefore, AP2 will forward the packet (information) via its ap2-mp0 interface, encapsulated in a Packet-In message to the controller.

Next, when the controller – also part of the mesh network – receives the Packet-In message on the sta3-mp3 interface, it typically performs packet analysis, path computation using its global network topology information and (reactive) flow installation on AP2's flow table. To execute this process, the controller sends a Flow-Modification message back to the ap2-mp0 interface, via the mesh backbone network. Based on the shortest path algorithm, newly installed flow entries should contain flow rules instructing AP2 to forward the packet via its ap2-mp0 interface, towards AP4. This rule is based on the event that the controller views the multi-hop connection between AP2 and AP4 as a single hop. However, since AP2 and AP4 are not actually within transmission range, the packet forwarding from AP2 to AP4 is done through the multi-hop communication framework, provided by IEEE 802.11s. Hence, with STA3 serving as the intermediate node between AP2 and AP4, AP4 will receive the packet on its ap4-mp0 (mesh) interface, from STA3's sta3-mp0 interface. Corresponding flow entries for the unmatched packets are installed on AP4's flow table, with a new process, like AP2's. This is based on the reactive flow installation method as well. So, AP4 can now forward the packet via its ap4-wlan0 interface to the associated/destination node i.e. STA5. Ultimately, STA5 receives the packet on its sta5-wlan0 interface, to complete the packet forwarding process.

In most cases, more than one packet of the same type, destined for STA5, will be sent out by STA1. Therefore, the newly installed flow entries will be used by AP2 and AP4 to handle such subsequent packets, either till the flow entries expire after an idle period or

when external events that trigger topology changes occur (e.g. mobility). However, in a case where communicating stations are associated with the same AP, packet forwarding operations are handled locally by that particular AP only, based on IEEE 802.11 provisions.

The SDMWN routing process, described above, underscores the importance of IEEE 802.11s to our SDN-based MWN architecture. IEEE 802.11s acts as the underlying (MAC layer) protocol in the backbone network, to facilitate (upper layer) OF controller operations, in the unfamiliar MWN environment. This WLAN mesh technology has been introduced to replace (rigid) wired links/infrastructures, for flexible connectivity between APs and seamless integration with the OF distribution system [32, 33, 34]. As a multi-hop wireless network solution, IEEE 802.11s also delivers transparent network operations, a single broadcast domain, and supports full mobility in the SDMWN architecture [32].

During multi-hop communications between nodes the mesh backbone network, IEEE 802.11s performs packet forwarding – using a table in the MAC layer – transparent to the network layer and upper layer entities. Generally, IEEE 802.11s also employs a Hybrid Wireless Mesh Protocol (HWMP) as its default path selection protocol [32, 33, 34]. HWMP adopts key features of the Ad hoc On-Demand Distance Vector (AODV) protocol, for path discovery. This includes Path Request (PREQ), Path Reply (PREP) and Path Error (PERR) messages; equivalent to Route Request (RREQ), Route Reply (RREP) and Route Error (RERR) messages in AODV [33, 34]. PREQ is sent by a source node to discover a destination path, PRER is sent back by the destination node in response to the PREQ, and PERR is used to indicate that a path is no longer available. Lastly, IEEE 802.11s uses a default (composite) metric, called Airtime Link Metric (ALM), that combines hop count

with data rate, overhead, and frame error rate of a test frame of size 1 Kbyte [32, 33, 34]. By calculating the amount of time required to transmit a test frame, this metric reduces the number of lengthy transmissions in the network [34].

To conclude, IEEE 802.11s only plays an intermediate role in the total end-to-end communications between stations. Since stations are not members of the mesh backbone network, end links (connecting stations to APs) are solely determined by the OF controller, while paths between APs are determined through IEEE 802.11s operations. Besides packets forwarding within the mesh backbone network, IEEE 802.11s also facilitates topology discovery and mobility management in SDMWN. These key processes are critical to the successful operation of the SDMWN architecture.

3.4 Topology Discovery in SDMWN

For efficient operation of any SDN architecture, particularly in the case of dynamic multi-hop wireless networks like SDMWN, the centralized controller should always maintain a global and updated view of the network. This is realized through a topology discovery process. Topology discovery is a key process in SDMWN because it enables the controller to provide accurate routing information, used by (OF-enabled) APs in the data plane to facilitate communication between stations in the OpenFlow network. During centralized routing in the SDMWN architecture, topology discovery is primarily coordinated by the controller as the forwarding devices do not possess any special functionality for topology discovery. This is unlike distributed routing in the traditional MWN architecture, where topology discovery is independently performed by every device in the network.

In the beginning, the controller-station component and APs in the mesh backbone network can detect each other through mesh discovery and peering operations. Mesh discovery and peering occur between neighbour nodes, within transmission range. This is based on standard active or passive scanning mechanisms in the MAC layer [32, 33]. During the mesh discovery process, members of the mesh backbone network send out beacons while listening and responding to probe frames [32, 33]. Subsequently, nodes exchanging the mesh-specific beacons and probe frames can form a mesh profile – a set of parameters specifying attributes of the mesh network [32, 33]. These attributes include a Mesh ID (a name identifying the mesh network), a configuration element advertising mesh services and multiple parameters supported by the mesh nodes [32, 33].

Furthermore, members of the mesh backbone network are expected to use the same mesh profile with matching attributes. This facilitates the peering process as neighbour nodes in the mesh backbone network can easily identify and establish mesh peering with each other, based on the matching mesh profiles [32, 33]. Thus, mesh nodes with different mesh profiles cannot establish a mesh peering. However, once mesh peering is successfully established between nodes in the mesh backbone network, peer mesh nodes can then communicate directly with one another [33]. Similarly, a mesh node can establish mesh peering with multiple (existing or new) neighbor nodes in the mesh backbone network [33].

The next stage of topology discovery in the SDMWN architecture occurs in the original OpenFlow network/distribution system. Based on the state of the art and with no official standard for topology discovery in SDN [35, 36], at the time of writing this thesis, the popular OpenFlow Discovery Protocol (OFDP) is employed for topology discovery in

SDMWN. This topology discovery mechanism is widely implemented across multiple controller platforms and applications [35]. In SDMWN, topology discovery can be further divided into three related stages namely: AP discovery, link discovery and STA discovery. OFDP works by taking advantage of an existing Link Layer Discovery Protocol (LLDP) – typically employed by Ethernet switches in wired network environments, to advertise or exchange information about their identities and capabilities [36, 37]. Although OFDP and LLDP share a few similarities (e.g. advertisements-only using similar LLDP frames), their modes of operations are quite different.

To begin with, LLDP is not primarily responsible for topology discovery in Ethernet, instead, its main function is to perform neighbour discovery among Ethernet switches. And a network management system like SNMP uses the information obtained through LLDP messages to discover the Ethernet topology [36]. Another notable difference between the two protocols is that each LLDP advertisement in the Ethernet environment is only exchanged across a single hop using a bridge-filtered multicast MAC address [36]. This means that LLDP packets are only exchanged between directly connected switches in the wired network and therefore, switches do not forward LLDP messages received.

However, this is not the case for OFDP in our SDMWN architecture and SDN in general. In SDMWN, LLDP packets are generated and distributed by the controller to OF-enabled APs in the mesh backbone network, using a normal multicast MAC address. These LLDP modifications in OFDP makes it more suitable to the SDMWN architecture.

The structure of an LLDP frame generally contains a header (of EtherType 0x88cc) and payload (also called LLDP Data Unit). The EtherType field in the header of the LLDP

frame is set to a default value of 0x88cc for simple identification of topology discovery messages in the OpenFlow network [37]. LLDP Data Unit (LLDPDU) is a sequence of Type Length Values (TLV), consisting of optional and mandatory TLV structures. The structure of an LLDP frame is shown in Table 3.1 below:

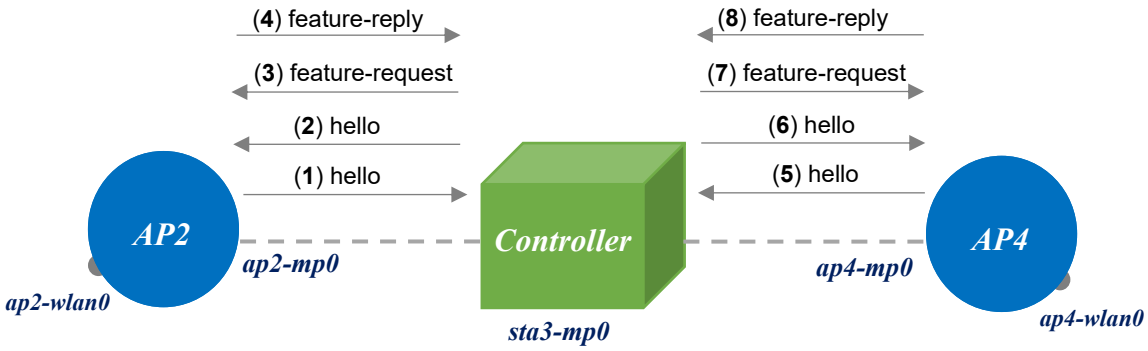
Table 3.1: LLDP Frame Structure

				LLDP DATA UNIT (LLDPDU)					
PREAMBLE	Dest. MAC	Source MAC	ETHERTYPE 0x88cc	CHASSIS ID	PORT ID	TTL	OPTIONAL TLVs	END OF LLDPU	FCS

From Table 3.1, we can see header fields in the LLDP frame, including destination and source MAC address fields. It also shows the payload or LLDPU section. LLDPDU starts with three mandatory TLVs, followed by several optional TLVs and is completed with a special mandatory TLV with type and length fields of zero [37]. These four mandatory TLVs that constitute to core OFDP operations include: Chassis ID (Type 1) – the identifier of a node sending the LLDP packet, Port ID (Type 2) – the identifier of a port through which the LLDP packet is sent, TTL (Type 3) – the value of time in seconds for which the information in the LLDP packet is valid, and End of LLDPDU (Type 4) – indicating the end of the payload in the LLDP frame. Optional TLVs include a basic set of TLVs and organizationally-unique TLVs, that can be used to introduce new topology discovery features using LLDP [37]. The important role played by LLDP during each step of the OFDP topology discovery process is reflected in the AP, mesh and STA discovery stages.

3.4.1 Access Point Discovery

For AP discovery, it is essential that APs in the SDMWN architecture are preconfigured with key sets of network information. The first set of such information includes an IP address and TCP port number used by the controller. Secondly, the OF-enabled APs are equipped with flow rules specifying that LLDP packets (of EtherType 0x88cc) received from any port other than the controller's port should be forwarded to the controller. These pre-installed flow rules are employed mostly in the link discovery stage. AP discovery is made possible through existing IEEE 802.11 mesh links. Using Figure 3.1 and Figure 3.2 as references, each step of the AP discovery process is illustrated in Figure 3.3 below:



ap2-mp0:
IP: 10.0.0.2
MAC: 00-02-00-00-00-01
Chassis ID = 1, Port ID = 1

ap2-wlan0:
MAC: 00-02-00-00-00-02
Chassis ID = 1, Port ID = 2

Controller
IP: 10.0.0.0.3
Port: 6633

ap4-mp0:
IP: 10.0.0.4
MAC: 00-04-00-00-00-01
Chassis ID = 2, Port ID = 1

ap4-wlan0:
MAC: 00-04-00-00-00-02
Chassis ID = 2, Port ID = 2

Figure 3.3: Access Point Discovery

Figure 3.3 shows the controller and APs in the mesh backbone network, participating in the AP discovery process. It also shows node parameters such as the controller's IP address and port number, and the IP address, MAC address, Chassis ID and Port ID of each AP interface. Such parameters are initially unknown to the controller before AP discovery is performed. Furthermore, we can see AP discovery messages (1 to 8) exchanged between the controller and APs, through previously established mesh links, with messages 1 to 4 involving AP2 and messages 5 to 8 involving AP4. This numbering format simply means that the controller carries out Access-Point discovery on one AP at a time.

At the beginning of AP discovery in SDMWN, every AP will try to contact the controller using the pre-configured IP address and TCP port number. This is to establish active sessions – used for exchanging configuration messages, installing flow table entries etc. APs kickstart the process by sending hello messages to the controller. On the other hand, the controller starts out by listening for such messages on its TCP port and then responds with corresponding hello messages to hello messages received from APs. Following successful session establishment, and as part of the initial handshake procedure, the controller sends out Feature-Request messages – inquiring about relevant configuration parameters and features from APs. Such information includes the Chassis ID, Port ID and corresponding MAC addresses of active interfaces. Similarly, APs respond to the controller with Feature-Reply messages, containing all the information originally requested by the controller. However, while both APs and their respective features are now known to the controller, the controller still has no knowledge about the connectivity between APs [36, 37]. The completion of the AP discovery process lays the groundwork for link discovery.

3.4.2 Link Discovery

Though all existing links in the mesh backbone network are typically discovered during IEEE 802.11s mesh discovery and peering operations, most of this information is currently unknown to the controller. Therefore, the controller has limited knowledge about AP2 and AP4 – obtained during the initial AP discovery process in Section 3.4.1. Moreover, the limited topology information does not yet include the existing multi-hop link between AP2 and AP4. Nonetheless, the controller is aware that both APs are each directly connected to it via a single-hop (wireless) link, based on AP discovery.

Since the main reason for incorporating IEEE 802.11s in SDMWN is to facilitate and not totally replace the original SDN operations, the controller is set up to independently discover the connectivity between APs, by performing the additional task of link discovery. This also fulfills an objective of our comparative analysis, which concerns the preservation of standard and original features of both MWN architectures. So, using the information contained in the Feature-Reply messages obtained during AP discovery, the controller periodically generates LLDP packets per active interface, for each AP, via separate Packet-Out messages. This process typically occurs at every 5-second interval and is executed using a normal multicast MAC address, aimed at each AP [36, 37]. Also, each LLDP packet sent out by the controller contains two key parameters i.e. Chassis ID (for identifying a destination AP, which we have labelled “1” for AP2 and “2” for AP4) and Port ID (for identifying a forwarding interface, which we have labelled “1” for the mp0 interface and “2” for the wlan0 interface, on that particular AP) that the packet is destined for. The Packet-Out messages will also contain instructions, indicating how received LLDP packets

should be handled/distributed using specific ports [38]. Using Figure 3.3 as a reference, the link discovery process is illustrated in Figure 3.4 below:

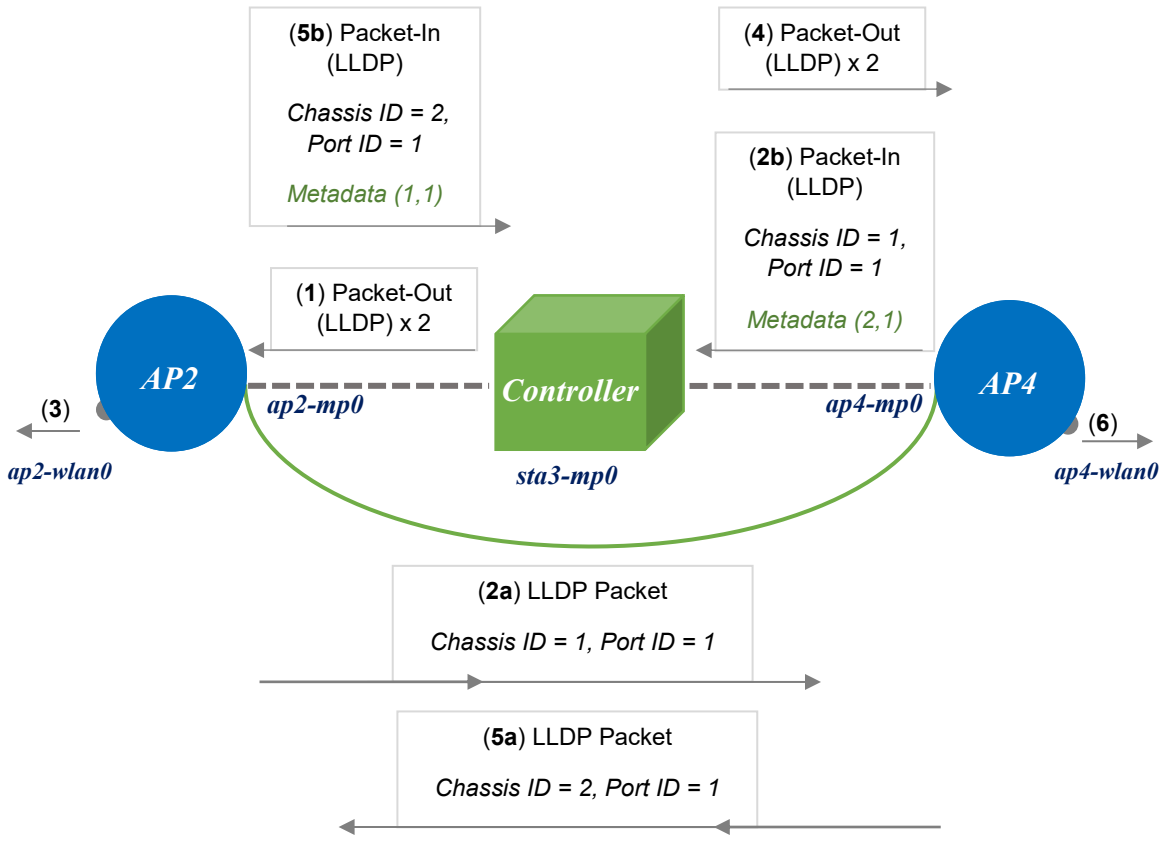


Figure 3.4: Link Discovery

Figure 3.4 shows messages (1 to 6) exchanged between the controller and APs, during the link discovery stage. It also introduces a new (virtual) link – highlighted by a green or solid curved line, illustrating the topology view of the controller. With AP2, messages (1, 2a and 2b) allow the controller to discover the wireless connectivity between AP2 and AP4.

At first, the controller sends out LLDP packets, encapsulated in the Packet-Out message (1), for each interface on AP2. AP2 then forwards the LLDP packets through its

appropriate interfaces, based on forwarding instructions specified by the controller in the original Packet-Out message. From the ap2-mp0 (mesh) interface, the LLDP packet (message 2a) is received by every other AP, which includes AP4, in the mesh backbone network. This process is achieved through the multi-hop connectivity and single (mesh) broadcast domain, provided by IEEE 802.11s operations in the mesh backbone network. In contrast, message (3) is irrelevant and discarded in the process, as there is no adjacent AP to receive the LLDP packet, on that side of the network.

When AP4 finally receives the LLDP packet, it inserts additional metadata and encapsulates the LLDP packet in a Packet-In message (2b), before forwarding it back to the controller. Such action is based on pre-installed flow rules on the OF-enabled APs, specifying that LLDP packets (of EtherType 0x88cc) that are not received directly from the controller should be sent back to the controller [35, 36, 37]. The metadata (y, z) inserted into the LLDP packet includes vital information about the Chassis ID (y) of the AP and ingress Port ID (z) on which the packet was received [35, 36, 37]. In this case, the metadata (2,1) inserted by AP4 into Packet-In message (2b) represents AP4's Chassis ID (i.e. 2) and its ingress Port ID (i.e. 1) – where the LLDP packet was received. Afterwards, information in the metadata together with the original information (i.e. Chassis ID = 1 and Port ID = 1) about AP2, in the payload of the LLDP packet, are is used by the controller to determine the existence of a (unidirectional) link from AP2 to AP4 [35, 36, 37].

For the controller to completely discover all possible links (and link types) in the network, the entire link discovery process – just concluded with AP2 – is replicated with AP4 as well. As shown in Figure 3.4, the controller uses messages (4, 5a and 5b) to detect

an equivalent link from AP4 to AP2. In that case, AP4 forwards LLDP packets, received from the controller in message 4, through its appropriate interfaces. So, after AP2 receives the LLDP packet in message 5a, it inserts metadata (1,1) and encapsulates the LLDP packet in a Packet-In message (5b), before forwarding it back to the controller. In the end, the overall link discovery process is leveraged by the controller determine that a bi-directional link exists between AP2 and AP4, in the OpenFlow network. Message (6) is irrelevant since there is currently no adjacent AP on that (right) side as well.

Taking another look at Figure 3.4, there is a visible difference in the topology of the mesh backbone network and the virtual network topology view of the controller. This is because the controller interprets the connection between AP2 and AP4 as a single-hop link or edge, represented by the (green) curved line in the OpenFlow network. However, in the physical network, what we truly have is a multi-hop/mesh link between AP2 and AP4. Likewise, two separate edges of the multi-hop wireless link can clearly be seen from Figure 3.1 in Section 3.2 – where STA3, the controller-station component holding the controller, serves as the intermediate node for forwarding traffic between AP2 and AP4.

An explanation for the link discovery phenomenon experienced by the controller is based on (underlying) IEEE 802.11s operations that enable LLDP packets, sent out of one (AP) mesh interface, to be received on the mesh interfaces of every other AP in the mesh backbone. It then allows each (recipient) AP to respond separately to the controller, via Packet-In messages (containing the original LLDP packet and additional metadata), based on existing flow rules regarding LLDP packets. This series of events creates an impression on the controller – where sets of (physical) multi-hop links between APs are represented

by (virtual) single-hop links. In the end, it appears to the controller that all APs are directly connected to one another. Equally, during the AP discovery stage, IEEE 802.11s operations also create the impression that each AP is directly connected to the controller.

To further understand this complex process, we will assume a new scenario where (end) stations, from Figure 3.1 in Section 3.2, are replaced with (OF-enabled) APs. This scenario is expressed in Figure 3.5 below:

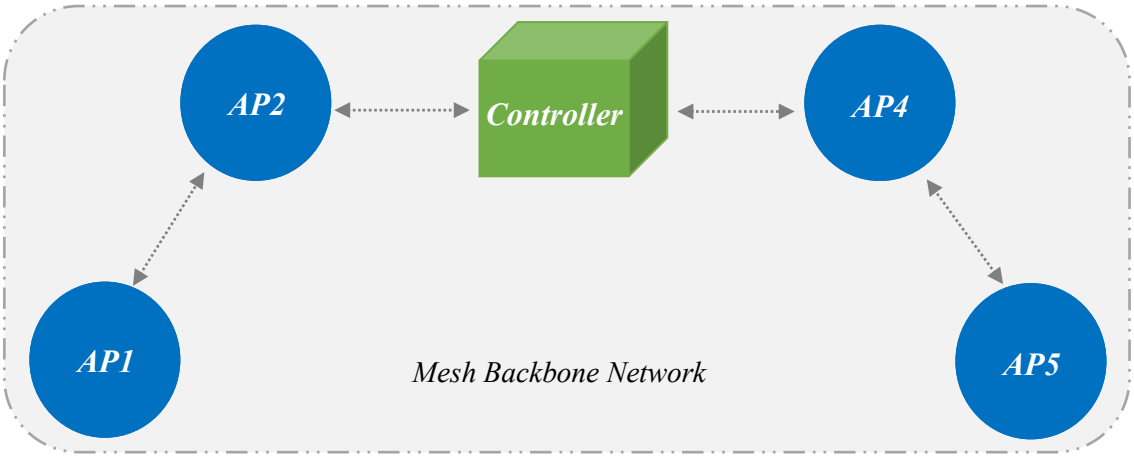


Figure 3.5: SDMWN Architecture with 4 APs

Figure 3.5 consists of 5 wireless nodes including 4 APs and the controller-station, in the second SDMWN architecture. This second architecture is similar to the original SDMWN architecture in Figure 3.1, except that previous stations have been replaced with APs (AP1 and AP5), having identical features as unchanged APs (AP2 and AP4). Also, messages (3 and 6), from Figure 3.4, have now become relevant due to the introduction of AP1 and AP5 as adjacent APs to AP2 and AP4 respectively.

Furthermore, Figure 3.5 shows four separate edges, representing the multi-hop link between AP1 and AP5, with AP2, STA3 and AP4 acting as intermediate nodes. However,

at the end of the link discovery process, this multi-hop network topology will less likely be the controller's interpretation of the topology. Instead the topology view of the controller, after AP and link discoveries, is illustrated in Figure 3.6 below:

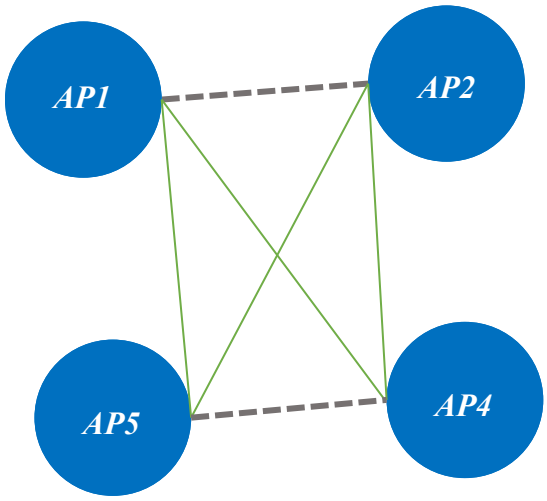


Figure 3.6: Link Discovery with Four Access Points

From Figure 3.6, the controller believes all APs are fully meshed with six (bi-directional) single-hop link or edges. The controller also believes each AP is a direct neighbour within communication range. However, this is not actually the case as distant APs in the (second) SDMWN architecture have multi-hop links between one another. Also, the controller is directly connected to AP2 and AP4 only, with multi-hop links to AP1 and AP5 instead.

Once again, the (virtual) network topology view of the controller is influenced by IEEE 802.11s operations, which facilitate the distribution and forwarding of LLDP packets and other control messages among APs and the controller in the mesh backbone network. Also, from Figure 3.6, the two (grey) broken lines represent physical links or edges in the mesh network. These are the links between AP1 and AP2, and AP4 and AP5. On the other

hand, the four remaining (green) solid lines represent virtual links that are conceived based on the IEEE 802.11s framework. The link discovery process is initiated periodically by the controller, to maintain up-to-date information about the state of the network, while the completion of link discovery sets the stage for (possible) station discovery.

3.4.3 Station Discovery

After AP and link discoveries between the controller and APs (in the mesh backbone network), we have STA discovery. This final stage of topology discovery, in SDMWN, is mostly reactive or on-demand because STA discovery usually occurs after stations generate some sort of traffic. Moreover, stations in the SDMWN architecture are not OF-enabled and therefore, are unable to communicate directly with the controller or participate in the OFDP (LLDP) packet exchange process that occurs during AP discovery. In that case, the OF-enabled APs also serve as intermediaries between stations and the controller. Like the packet forwarding process explained in Section 3.3, the controller can collect information about existing stations, interfaces and corresponding links to APs by using (STA) traffic – encapsulated in Packet-In messages and forwarded by APs.

Types of traffic that promote STA discovery in SDMWN include Logical Link Control (LLC) messages generated by stations to establish connections with APs [39], and various request/reply (query) messages that may be exchanged between stations. Both traffic types usually contain sufficient information, required for the controller to determine the existence of stations in the network. Using Figure 3.4 as a reference, the STA discovery process using a ping request/reply traffic is illustrated in Figure 3.7 below:

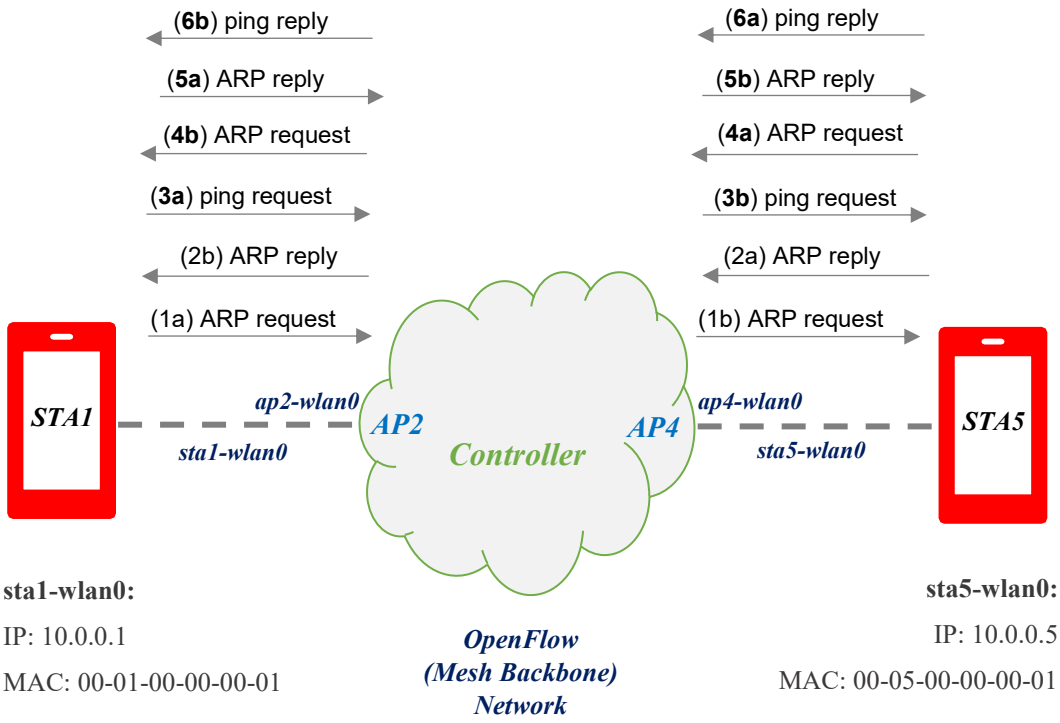


Figure 3.7: Station Discovery

The complete SDMWN architecture, after STA discovery, is illustrated in Figure 3.7 above. This shows STA1 and STA5 connected to AP2 and AP4 respectively, in the (OpenFlow) multi-hop network. Figure 3.7 also indicates pairs (a and b) of messages (1 to 6) exchanged between the stations, via the OpenFlow/mesh backbone network. The whole process involving a back and forth exchange of messages leads to STA discovery.

Alternatively, STA discovery can occur at an earlier stage, possibly during AP discovery and before the ping request/reply packets are exchanged. This can be due to LLC messages generated by each station when associating with the corresponding APs at the time of AP discovery. Just like we have in the mesh discovery and peering process earlier discussed, APs also send additional beacon frames to enable association with stations [32].

These standard 802.11 beacons are different and sent separately from the mesh beacons [33]. Upon detecting such beacons, each station then sends out LLC messages to establish connections with appropriate APs.

Moreover, the situation completely changes if a station stays idle (zero traffic) for a specified period (typically 10 seconds), then the controller eventually removes the station from its global topology database. For the STA discovery process in Figure 3.7, we assume both stations have not generated any traffic for a very long time – beyond an idle timeout period. Before STA1 can send a ping request to STA5, we already established in Section 3.2 that ARP request and reply messages (1a, 1b, 2a and 2b) must be exchanged between both stations, if such information is not already available. Eventually, when AP2 receives the ping request packet from STA1, it will encapsulate the packet in a Packet-In message before forwarding to the controller. This is based on our previous assumption that both stations have been idle for a very long time and therefore, any possible pre-installed flow entry must have expired during the idle period. When the controller receives the Packet-In message, it extracts and analyses the header information to determine the existence of STA1. Likewise, the controller also obtains new information about AP2, which includes the ap2-wlan0 interface's (ingress) Port ID, connecting STA1 to the OpenFlow network.

Notwithstanding, at this stage, the controller still has no knowledge about STA5 and instructs AP2 to flood the packet into the network. Due to this flooding action, the ping request packet eventually reaches AP4 – with no corresponding flow rules either – AP4 also forwards the packet back to the controller. Upon receiving the original packet again, the controller still has no information about STA5 and similarly instructs AP4 to flood the

ping request packet into the network as well. It is at this particular moment that STA5 finally receives the ping request packet, originally sent by STA1. After receiving the ping request packet, STA5 is obliged to respond to STA1 with a corresponding ping reply. Like the earlier process of STA1 obtaining STA5's MAC address, STA5 will also go through the ARP request/reply process – using messages (4a, 4b, 5a and 5b) to obtain STA1's MAC address. Once this process is complete, STA5 can finally send the ping reply packet

As expected, the ping reply packet from STA5 is initially first received by AP4 before it is forwarded to the controller, encapsulated in a Packet-In message. This action is based on the absence of flow rules to handle such packet as well. With the new ping reply packet from AP4, the controller also becomes aware of STA5's existence and is now able to obtain vital information about its parameters. Hence, with previous information the controller already has about STA1, the controller can now construct a global view of the entire SDMWN topology. Ultimately, the controller uses its global visibility to compute the best path for the ping reply packet to travel from STA5 to STA1 and installs appropriate flow entries on both APs' flow tables as well, to guide present and future communication between STA1 and STA5. And with the controller's (virtual) topology view, relevant flow entries are specifically installed on APs that are directly connected to the communicating stations. Routing between APs in the mesh network is handled by IEEE 802.11s instead.

This completes the (final) STA discovery process and overall topology discovery in our SDMWN architecture, albeit only considering a static SDMWN network. For the successful operation of SDMWN under mobility conditions, a robust mobility management mechanism is discussed in Section 3.5.

3.5 Mobility Management in SDMWN

The end of initial OFDP topology discovery in SDMWN brings up another critical area in ensuring that the controller constantly maintains and updates its global network view, particularly in the presence of external factors like mobility. Mobility events result in physical topology changes in the existing network, previously known to the controller. This requires the controller to subsequently make corresponding adjustments to its global network topology database, as soon as possible, for effective path computation.

In the mesh backbone, when topology changes occur, mesh nodes (i.e. APs and the controller) with identical mesh profiles can simply establish a new mesh peering with current neighbour nodes. Additionally, even when link breakage occurs during mobility, mesh nodes may keep the peer link status to facilitate quick reconnection, if possible [32]. These (transparent) IEEE 802.11s mobility provisions also translate into the controller's mobility management of APs. Therefore, in the OpenFlow network scheme of things, the mobility of APs is not a problem based on the controller's fully meshed view of connected APs. However, a link timeout after a sustained period of time is used to indicate possible link breakage in the OpenFlow network topology. This is required for maintaining plane-to-plane connectivity between the controller and AP(s). For this, echo request and reply messages are exchanged by the controller and adjacent APs as a keep-alive mechanism to preserve and verify the liveliness of the controller-AP session [38].

For mobility management in the complete SDMWN architecture that specifically involves stations, SDMWN employs a more advanced mobility management mechanism. This is achieved using a host tracking feature of the controller, responsible for regularly

monitoring conditions of stations and possible topology changes [41]. This host tracking process – coordinated by the controller – solely relies on the OF-enabled APs to inform the controller about the location of stations, through Packet-In messages containing messages originally generated by stations. A host tracker holds a profile for each station in the network. Therefore, when a Packet-In message arrives at the controller, the host tracker examines the message so as to determine and compare the station's current profile (including IP address, MAC address, Chassis ID and ingress Port ID on the connected AP) with the existing profile in its global network topology database. This helps the controller always maintain and update its global network view as required.

In the SDMWN architecture, two major events that can be explored as related to mobility management or host tracking are JOIN and MOVE events [42].

3.5.1 JOIN Event

The first event – JOIN – usually occurs when stations are newly introduced to the network or associating with APs, in the OpenFlow network, for the first time. In this case, the controller has no information about the current profiles of stations. Using Figure 3.5 as a reference, during an initial connection establishment process between STA1 and AP2, STA1 will send out LLC broadcast messages towards AP2's direction. This constitutes a service request for connection as LLC – a sublayer (upper layer) of the data link layer – provides a basic interface to higher layers like the network layer [39, 40]. It also provides the method for addressing stations across the network while also controlling the exchange of data between stations and APs.

Eventually, the LLC message is received by the controller as a Packet-In message. This is because when AP2 receives the LLC message with no match in its flow table, AP2 forwards the LLC message to the controller by default. Upon receiving and extracting the Packet-In message, the host tracker on the controller checks the controller memory for an existing similar node profile. If none exists as expected, a new node profile for STA1 is created. In this case, the controller concludes that a new station (STA1) has just joined the network. This host tracking procedure is the same for STA5 and is employed by the controller to determine the existence and state of stations in the network. Such information is also useful during the installation of flow entries about the new stations.

3.5.2 MOVE Event

The second event – MOVE – occurs while a station is already a member of the network but changes its position and consequently, its AP association. Like the JOIN event, during the process of associating with a new AP, the station will also send out LLC broadcast messages. Likewise, the LLC message is eventually received by the controller, through an AP. Upon examining the Packet-In message, the host tracker successfully identifies the node profile but notices varying location information regarding the station, especially in the Chassis ID and ingress Port ID of the connected AP. For example, in Figure 3.3, AP2 and AP4 have different Chassis IDs i.e. 1 and 2 respectively. This information is already known to the controller. Therefore, if STA1 moves from AP2 to associate with AP4, the host tracker will notice a change in the Chassis ID of the connected AP, although the Port IDs are identical. This mismatch in the node profile of STA1 basically reflects its current

association with AP4 instead of AP2 as before. In this case, the controller concludes that STA1 has moved to a new location or AP association and the controller updates the corresponding node profile and global topology information.

Furthermore, in a case that mobility occurs during an active communication between stations, the controller is required to update flow entries on appropriate AP(s). To handle such situation in SDMWN, most newly installed flow entry, except preconfigured flow entries like those used for LLDP packets, contain two key parameters – idle timeout and hard timeout. These parameters both control the removal of a flow entry from a flow table, so as to minimize the occurrence and impact of stale or irrelevant flows/flow entries, during communication between mobile stations [43, 44]. And while both timeouts perform identical tasks, they are quite different in execution. The idle timeout is a period (about 10 seconds) after which a flow entry is removed from any AP's flow table because there are no (subsequent) packets matching it (anymore). On the other hand, the hard timeout is a fixed period (about 20 seconds) after which the flow entry is removed from the AP's flow table, whether packets match it or not [43 44]. In either case, the first timeout (idle or hard) to be exceeded triggers the AP to remove the appropriate flow entry from its flow table.

To better understand how these timeouts support communication between (mobile) stations, as part of the mobility management provision in SDMWN, we can envisage two particular cases where the intervention of each timeout is crucial. The first/moving-sender case involves a source or sender-station moving and changing its AP association, while the moving-receiver case involves a destination or receiver-station moving and changing its AP association instead. At no time can both stations be associated with the same AP. And

in both cases, we consider a situation whereby the sender-station is continuously generating traffic, specifically intended for the receiver-station, in a unidirectional flow operation.

In the moving-sender case, when the sender-station moves from an old sender-AP to associate with a new sender-AP, the host tracker becomes aware of such event through LLC messages generated by the sender-station, and forwarded as Packet-In messages by the new sender-AP. Consequently, the controller updates its global topology database to reflect such topology change. During this process, the new sender-AP will also engage the controller to install flow entries, to handle new incoming traffic and for communication to resume between the two stations. Lastly, the idle timeout also kicks in to remove the stale flow entries on the old sender-AP, after a particular period of inactivity. And this completes the mobility management process for the moving-sender case in SDMWN.

In the moving-receiver case, when the receiver-station moves from an old receiver-AP to associate with a new receiver-AP instead, the host tracker again becomes aware of such event, while the controller also updates its global topology database to reflect such topology change. However, in this case, the new receiver-AP has no incentive to engage the controller for new flow entry installations. This is due to the lack of incoming packets, as the sender-AP still contains stale flow entries that direct packets to the old receiver-AP. This goes on until the hard timeout finally kicks in, after a fixed amount of time, to remove the stale flow entries on the sender-AP. Afterwards, the sender-AP immediately engages the controller to install new flow entries that address the (new) receiver-AP and location of the receiver-station. Likewise, the new receiver-AP engages the controller to install new flow entries, to handle the incoming traffic and to resume communication between the two

stations. Lastly, the idle timeout also kicks in to remove the stale flow entries on the old receiver-AP. And this completes the mobility management process for the moving-receiver case in SDMWN.

Based on the discussion above, we can conclude that it will take a longer period for communication to resume between stations in the moving-receiver case, compared to the moving-sender case. This is mainly because the new sender-AP in the moving-sender case has the incentive to immediately engage the controller for the new flow entry installations, without having to wait for a hard timeout period. Again, this is due to incoming packets sent from the source or sender-station.

3.6 Summary

In this chapter, we began by highlighting the practicality and mobility issues associated with existing SDN-based MWN solutions, and why such solutions may not compare well with distributed routing in traditional MWN architectures. Furthermore, we proposed an SDN-based MWN solution, called SDMWN, and argued that the SDMWN architecture compares well with traditional MWN architectures, mostly based on its practicality and existence of fully mobile nodes, in both control and data planes, in a completely wireless network. This framework and conditions have been put in place to promote fairness and clarity in our comparative analysis between centralized routing and distributed routing in their respective MWN architectures, while also attempting to tackle limitations related of other SDMWN proposals. After providing high-level descriptions of components and their connections in the SDMWN architecture, we finished by highlighting and describing major

aspects of SDMWN operations. These include centralized routing and packet forwarding, topology discovery and mobility management.

While SDMWN still inherits a few performance limitations like SPOF, network overhead and high latencies in the first few packets of new flows, commonly associated with the SDN network paradigm, we believe this is an improvement on existing SDN-based MWN architectures. Major advantages of SDMWN over existing SDN-based MWN solutions include: practicality in design, reliable plane-to-plane communication, minimal flow entry installation, full mobility, and seamless network integration. In contrast, the most visible disadvantage of SDMWN is related to the dual topology discovery process. And though topology discovery in SDMWN is independently performed by two separate entities (i.e. IEEE 802.11s and OFDP), IEEE 802.11s operations only occur in the MAC layer and are totally transparent to the upper layer entities.

All things considered, significant benefits of the SDMWN architecture make it an appropriate centralized routing representative for the comparative analysis with distributed routing in the traditional MWN architecture, to be discussed in Chapter 4.

Chapter 4

Comparative Analysis of MWN Architectures

4.1 Introduction

In previous chapters – Chapters 1, 2 and 3 – we established major differences between centralized routing and distributed routing in MWN and all networks in general. Likewise, after extensively reviewing the state-of-the-art and analyzing the merits and demerits of both approaches, we presented supporting arguments about the benefits of adopting the centralized routing approach in MWNs.

Taking a step further, we also proposed a Software Defined Multihop Wireless Network, identified as SDMWN. The SDMWN architecture is completely wireless and well-equipped with fully mobile nodes. This approach provides us with the opportunity of having an in-band controller, that can also become mobile if needed; quite similar in design to traditional MWN architectures such as VANETs. SDMWN is not only practical and forward-thinking but is also aimed at preserving the original features of SDN that make it impressive. Such features include: (standard) OFDP for topology discovery and OF-enabled APs as replacements for (conventional) switches. Likewise, the traditional MWN architecture employs standard OLSR features and parameters to ensure consistency.

In this chapter, we proceed to the final objective of this thesis, which involves conducting a fair comparative analysis between the centralized routing in our SDMWN architecture and distributed routing (with OLSR) in our traditional MWN architecture. The rest of this chapter is organized as follows. Section 4.2 covers respective implementations of the MWN architectures to be compared, including our proposed SDMWN architecture and traditional MWN architecture. Section 4.3 describes performance evaluation metrics, tests and data collection techniques to be employed all through our research. Section 4.4 provides detailed descriptions of our experiment designs and methodology for the performance evaluation of both MWN architectures. Finally, Section 4.5 provides general analyses and anticipated outcomes of the comparative analysis between centralized routing and distributed routing, in their corresponding MWN architectures.

4.2 MWN Implementations

The MWN architectures, to be compared, are both implemented on Mininet-WiFi (version 2.2.1d1), using already existing (internal and external) network elements. Mininet-WiFi, a fork of the Mininet SDN emulator, is a tool widely employed for high-fidelity experiments as it can emulate several wireless network architectures and scenarios [45]. These also include traditional and SDN-based MWN applications, under different network conditions.

For our SDMWN architecture, a POX controller (0.3.0 – dart version) is also incorporated as an add-on feature, interfacing with Mininet-WiFi to provide centralized routing services. POX, alone, is a networking software platform that can effectively serve as an OpenFlow controller in any SDN architecture [46]. It can be used in both wired and

wireless OpenFlow network environments, under static or mobile network conditions. Major components of the POX controller platform for implementing centralized routing and other SDMWN operations include: an *openflow.discovery* component for OFDP-based topology discovery [46], a *forwarding.l2_learning* component for flow installations using matching MAC and IP address fields [46], a *host_tracker* component for handling mobility in stations [46], an *openflow.keepalive* component for maintaining plane-to-plane connectivity [46], and a *misc.gephi_topo* component for Gephi-based visualization of the controller's (OpenFlow) network topology view [46]. Gephi (version 0.92) is a graph or network visualization and exploration tool [46]. The network parameters of most of the listed POX components are set to their default values, including an LLDP interval of 5 seconds, a keepalive (ICMP echo) interval of 1 second and a link timeout period of 10 seconds. Other major elements of our SDMWN architecture are all readily available on Mininet-WiFi instead. These include stations, OF-enabled APs, IEEE 802.11 and IEEE 802.11s services. Notably, a beacon interval of 0.1 second and a mesh beacon interval of 1 second are maintained for IEEE 802.11 and IEEE 802.11s operations respectively.

Alternatively, to support distributed routing in our traditional MWN architecture, an OLSR daemon (olsrd version 0.6.8) [47] is also integrated with Mininet-WiFi. OLSR is the representative distributed routing protocol, widely adopted among researchers, for evaluating the performance of (traditional) MWNs such as MANETs. In our application of OLSR, most of the protocol parameters are configured based on the recommendations in [12]. Particularly, key parameters such as the Hello interval and TC interval are set to 2 seconds and 5 seconds respectively.

4.3 Performance Metrics

For the comparative analysis between centralized routing in our SDMWN architecture and distributed routing in our traditional MWN architecture, the following performance metrics are selected to provide a comprehensive evaluation: Ping Success Rate (PSR), Round-Trip Time (RTT), Normalized Routing Overhead (NRO)and Convergence Time.

4.3.1 Ping Success Rate

PSR is the ratio of total number of ping packets successfully delivered to the destination(s) to total number of packets sent by the source(s). This provides valuable information about the reliability and competency of each routing method to successfully deliver data packets. PSR tests are conducted on Mininet-WiFi using the Ping utility. Ping is the most popular tool for measuring active network performance [48]; using ICMP echo request and reply messages to determine end-to-end connectivity. As it is difficult to determine the status of a transmitted packet, PSR is a two-way metric measured based on responses (ICMP echo replies) to the original ICMP echo requests. When comparing both routing protocols, all else being equal, we aim at higher PSR, indicating a superior protocol performance.

4.3.2 Round-Trip Time

RTT is also a two-way metric for judging the performance of both MWN architectures. It is the total time taken for a packet to travel from a specific sender-station to a specific receiver-station and back to the original source again. This time taken (in milliseconds) by a packet is affected by path length or hop count and several network delay factors, ranging

from queueing and congestion delay to packet loss and retransmission delay. The RTT performance of each routing method is also evaluated using Ping to determine total time taken for a sender node to receive ICMP echo reply to a corresponding ICMP echo request that originated from the same sender. RTT statistics are measured using a timer feature of the ping utility, which records the time elapsed when an ICMP echo reply packet arrives at the sender node, where the corresponding ICMP echo request packet was generated. In the end, just like we have for PSR, average RTT results (in milliseconds) are obtained directly from the ping terminal at the end of the ping tests. However, unlike PSR, we aim at lower RTT, indicating a superior protocol performance.

4.3.3 Normalized Routing Overhead

NRO is defined as the average number of control messages required per data packet delivered at the destination. Control messages – exchanged during topology discovery and other maintenance operations – play important roles for successful data transfer in both MWN architectures. However, the characterization of control messages in the traditional MWN architecture is quite straightforward compared to the SDMWN architecture. In the traditional MWN architecture with OLSR, relevant control messages comprise of Hello, TC and even ARP messages exchanged between stations in the network. On the other hand, with the incorporation of IEEE 802.11s and other IEEE 802.11 services in SDMWN, the classification of control messages is extended to cover these unique network entities. Consequently, control messages considered in the SDMWN architecture include: beacon, acknowledgements, mesh-beacon and LLC messages for IEEE 802.11 and IEEE 802.11s

operations; TCP, LLDP, OF {Hello, Feature Request, Feature Reply, Packet-Out, Packet-In, Stats Request, Stats Reply, Barrier Request, Barrier Reply, Flow-Modification, Port-Modification, Echo Request, Echo Reply} messages for OpenFlow operations; and to conclude, ARP messages exchanged between stations.

Unlike the variable control messages, data packets are basically ICMP echo request/reply packets exchanged between the sender/receiver nodes only, for both MWN architectures. Essential data required for computing NRO is collected specifically from a hwsim0 [51] interface, using Wireshark. The hwsim0 interface is a software interface, created by Mininet-WiFi, that copies all wireless traffic from all the virtual wireless interfaces in the network scenario [51]. Wireshark is the network analysis tool employed to capture packets in real time and to display them in a human-readable format for easy data collection [49]. To obtain final NRO results, the data collected is processed using a custom-developed AWK script. AWK is the data-driven scripting language for text processing, data extraction and reporting [50]. NRO for each routing method is calculated with AWK, using the formula below:

$$Normalized\ Routing\ Overhead\ = \frac{total\ number\ of\ control\ messages}{total\ number\ of\ data\ packets}$$

Results obtained provide information about the efficiency of each routing method, based on the average number of messages (frames and packets) required to successfully deliver a single data packet to a destination. In this case, a lower NRO result between the two routing methods suggests superior network performance.

4.3.4 Convergence Time

Convergence Time is originally defined as total time taken to complete the topology discovery process in a network. However, in mobile MWN architectures, where topology changes exist and are unpredictable, convergence time is further expanded as total time to reflect topology updates during link state events or changes in network condition. This includes the time elapsed between link failure detection and the time a new and stable path is restored [52]. However, these two events (i.e. link failure detection and path restoration) are very difficult to pinpoint in actual MWN environments. Therefore, an effective method to estimate such convergence time is by performing/analysing throughput tests between pairs of nodes. Throughput is a measure of total number of packets that are transmitted per unit time in seconds. It provides evidence about the level of utilization of available network resources during data transmission. With this, convergence time is obtained as the time (in seconds) from which a previously valid/active route between a specific source and a specific destination becomes invalid/ inactive (highlighted by a steep fall in throughput figures) to when a new valid/active route between the exact same source and the exact same destination is discovered (highlighted by a steep rise in throughput figures).

Throughput tests are easily conducted with iPerf – a simple and powerful tool that measures maximum achievable TCP and UDP bandwidth [53]. Subsequently, throughput performance data required to estimate the convergence times of both MWN architectures is obtained from Wireshark, using its I/O Graphs feature. Wireshark I/O Graph is a user-configurable graph tool, for illustrating patterns or trends of captured network packets [49]. The relevant events, measured on the Wireshark I/O Graphs, are the points emphasized by

steep downward slopes and corresponding steep upward slopes. Hence, mean convergence time between the communicating pair(s) of nodes is calculated using the equation below:

$$Convergence\ time = \frac{\sum_1^n (t_{pi} - t_{di})}{n}, \quad i = 1, 2, 3, \ldots, n$$

Drop time, t_{di}, is the time recorded on the graph indicating when previously stable throughput performance starts decreasing in a steep downward slope, peak time, t_{pi}, is the corresponding time recorded after t_{di} on the graph when peak throughput is reached after a steep upward slope, and n is the number of (t_{di}, t_{pi}) pairs or convergence events. Results obtained provide information about the speed and robustness of each routing approach. As high convergence time can increase packet loss in the network, a lower convergence time result between the two routing methods suggests superior network performance.

4.4 Experiment Design and Methodology

Using our performance evaluation metrics, tests carried out for the comparative analysis between SDMWN and traditional MWN are divided into two major types: static network experiments and mobile network experiments. Static network experiments include sanity checks to ensure as much fairness as possible, during the comparative analysis of both MWN architectures. We also have baseline tests, to study network activities or behaviours of both MWN architectures, which provide the groundwork for our comparative analysis. On the other hand, mobile network experiments are carefully designed to subject both MWN architectures to rigorous performance tests, under identical conditions.

4.4.1 Static Network Experiments

Our static network experiments mainly consist of sanity checks – in the form of throughput tests, and baseline tests – in the form control message rate and average hop-count tests. Therefore, in total, there are three major performance tests be carried out on both MWN architectures, under identical static network conditions. Starting with the throughput tests, these employ a line topology in which bi-directional, end-to-end throughput performance is tested under varying hop counts, for both MWN architectures. Key parameters used for the throughput tests are shown in Table 4.1 below:

Table 4.1: Static Network Emulation Parameters

Parameters	**Value**
Emulation time	180 s
Network area	305 m x 80 m
Number of nodes	5, 6, 7, 8, 9,10
Network topology	Line
Channel type	Wireless
MAC protocol	IEEE 802.11
Mode	g
Propagation loss model	Log-distance
Transmission range	40 m
Association control	Strongest-Signal-First
Routing protocols	OLSR, Shortest path & IEEE 802.11s
Traffic generator	iPerf (TCP)
TCP window size	85.3 Kbytes (default)
Measurement interval	0.5 s

The throughput tests are sanity checks to confirm that both MWN architectures are indeed comparable by determining whether similar interference exists, and wireless channels are utilized in a similar manner for both MWN architectures. From Table 4.1, each static network emulation will run for a total of 180 seconds – with the first 60 seconds dedicated to topology discovery or network convergence, while the remaining two intervals of 60 seconds each are for the bidirectional iPerf (throughput) tests under varying hop counts, for both MWN architectures. Additionally, the control message rate test can be performed concurrently with the throughput tests. This is achieved during the early 60-second period of topology discovery, only for the biggest network with 10 nodes. During this period, the total number of control messages generated in both (idle) MWN architectures is recorded.

With a horizontal distance of 25 meters between each node, a varying hop count in the line topology is achieved by increasing the network size or number of wireless nodes in the static network. And since we are testing end-to-end connectivity, which mostly involves end stations, the number of stations in the SDMWN architecture is kept constant at 3 stations; a station at each end of the line topology and the single remaining station is situated at the center of the line topology to house the controller. Therefore, a static network of size N for the SDMWN architecture consists of N-3 APs – serving as intermediate nodes. On the other hand, there is no need for such provisions in the traditional MWN architecture since it consists of stations only. This means a network of size N simply translates to N stations. Lastly, to obtain fine-grained throughput results, current TCP bandwidth is recorded at (the minimum) 0.5-second intervals which translates to 2 measurements per second, for both MWN architectures. Each set of experiments is repeated 10 times (under

identical conditions) to determine the statistical significance of our test results, within the 95% confidence interval. Models of the largest network for static network experiments, consisting of 10 nodes, are illustrated in Figure 4.1 and Figure 4.2 below:

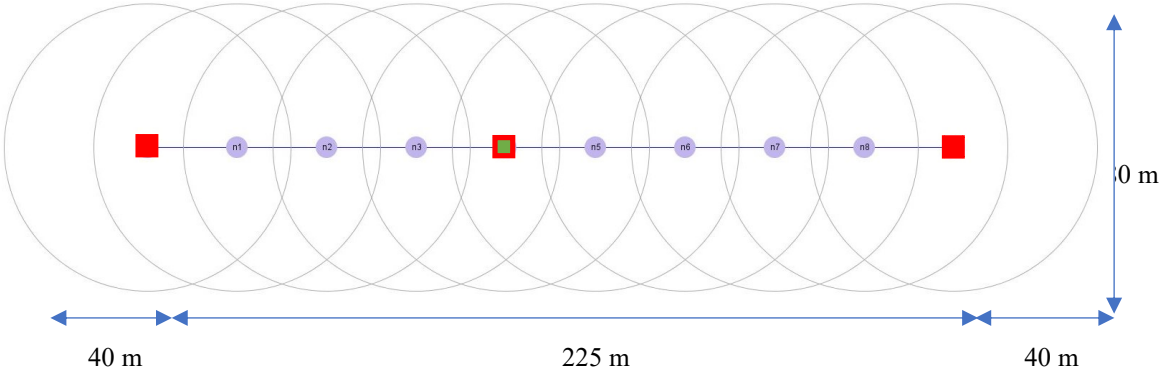


Figure 4.1: Static Network Model for SDMWN

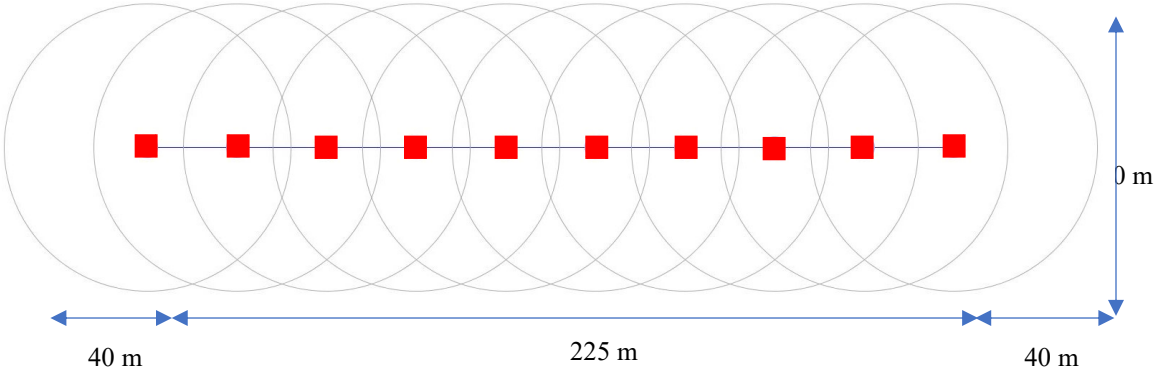


Figure 4.2: Static Network Model for Traditional MWN

Figure 4.1 shows the model of the largest static network experiment consisting of 10 nodes, including 7 APs, 2 stations and the controller-station in the SDMWN architecture. The two (red) square nodes are end-stations, the seven (blue) round nodes are forwarding-APs, and

the (red-green) square node (at the centre) is the controller. On the other hand, Figure 4.2 shows a corresponding model of the largest static network experiment in our traditional MWN architecture, consisting of 10 stations, represented by the (red) square nodes only.

To conclude our static network experiments, the final tests to be performed are the average hop-count tests. These are baseline tests to study the path selection behaviours of centralized routing and distributed routing, in their corresponding MWN architectures. Unlike the throughput and control message rate tests, the average hop-count tests employ a grid topology. Similar grid configurations, used for our mobile network experiments, are fully explained in the next section. For initial average hop-count tests, (grid) path lengths between the two diagonal vertices are measured under varying network sizes, for both MWN architectures. Next, for insights into more random events, a separate set of average hop-count test is performed using the largest grid network of 50 nodes to find average hop counts between 10 pairs of randomly-scattered nodes. With OLSR in our traditional MWN, path discovery between end stations is simply achieved using traceroute [48]. However, path discovery between end stations in SDMWN requires us to use the *iw* [51] utility, due to (transparent) IEEE 802.11s MAC layer operations between APs in the mesh backbone.

4.4.2 Mobile Network Experiments

Our mobile network experiments also employ a grid topology. This can act as a framework, to support the continued existence of (potential) single-hop and/or multi-hop links between communicating nodes, during the PSR, RTT, NRO and Convergence Time tests. Key parameters used for these performance tests are shown in Table 4.2 below:

Table 4.2: Mobile Network Emulation Parameters

Parameters	**Value**
Emulation time	350 s
Network area	205 m x 180 m
Number of nodes	10, 20, 30, 40, 50
Network topology	Grid
Grid area	25 m x 25 m
Channel type	Wireless
MAC protocol	IEEE 802.11
Mode	g
Propagation loss model	Log-distance
Transmission range	40 m
Association control	Strongest-Signal-First
Routing protocols	OLSR, Shortest path, IEEE 802.11s
Traffic generators	Ping (ICMP echo), iPerf (TCP)
Ping Packet size	56 bytes (default)
Ping interval	0.3 s (≈ 3 packets/s)
TCP window size	85.3 Kbytes (default)
Mobility model	RWP (with area constraints)
Node speed	10 m/s

From Table 4.2, each mobile network emulation is run for a total of 350 seconds with the first 100 seconds dedicated to topology discovery or network convergence while the remaining 250 seconds involve data transmission and performance testing under varying network densities, for both MWN architectures. Network density is varied by incrementing the network size or number of nodes in the network. For the SDMWN architecture that consists of stations and APs, the number of stations (including the controller-station) and

APs is allocated at a ratio of 2 to 3, in favour of APs, which make up the grid topology. Therefore, a network size of N for the SDMWN architecture will consist of 2N/5 stations and 3N/5 APs which instead translates to just N stations in the traditional MWN architecture. Each square component of the grid topology has a 25 m x 25 m square area with diagonal lengths of approximately 35 m. This grid measurement is carefully selected such that adjacent nodes have stable communication links within the 40m transmission range, between each other. In both MWN architectures, the main grid topology is formed by a group of 3N/5 nodes that are located at every grid point. For the SDMWN architecture, grids are strictly formed by APs while 2N/5 stations (including the controller-station) are randomly scattered around a restricted transmission area covered by the grid nodes. The traditional MWN architecture employs a similar technique in which 3N/5 stations are located on grid points while the remaining 2N/5 stations are scattered around the restricted transmission area provided by the grid nodes.

During PSR, RTT and NRO performance tests, using the ping utility, ICMP echo request and corresponding ICMP echo reply packets are exchanged between separate pairs of all randomly-scattered stations, in both MWN architectures. Throughout these tests, only one station from a pair of stations is responsible for initiating ping requests while expecting a corresponding ping reply from the target (pair) node. For the SDMWN architecture of size N, the N/5 (i.e. 2N/5 ÷ 2) pairs of stations exchanging ping packets also include the controller-station. Identical node pairs are selected for transmission in the traditional MWN architecture as well. In addition, a constant ping interval is selected at 0.3 seconds (≈ 3 ping packets per seconds) which translates to 3N/5 ping requests per second for N/5 pairs. For

the ping utility, only admin/root users can specify ping intervals less than 0.2 seconds as lower intervals are highly susceptible to network flooding; as indicated by an error message. Therefore, the 0.3-second interval is selected such that it is enough to ensure fine-grained results and observation during network performance testing; while also restricting packet flooding or network overload. Finally, for convergence time readings, single-flow iPerf tests are performed between two stations. Models of the largest network of 50 nodes, for mobile network experiments, are shown in Figures 4.3 and 4.4 below:

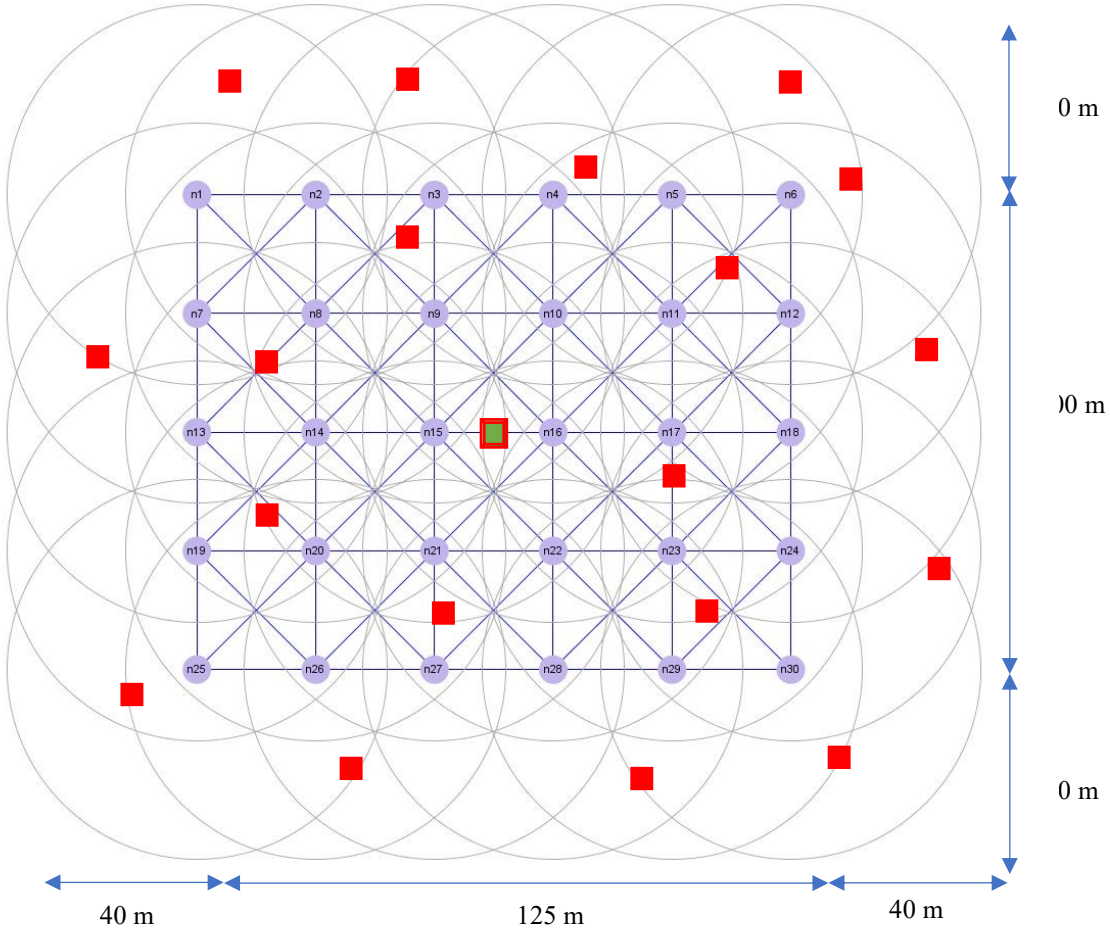


Figure 4.3: Mobile Network Model for SDMWN

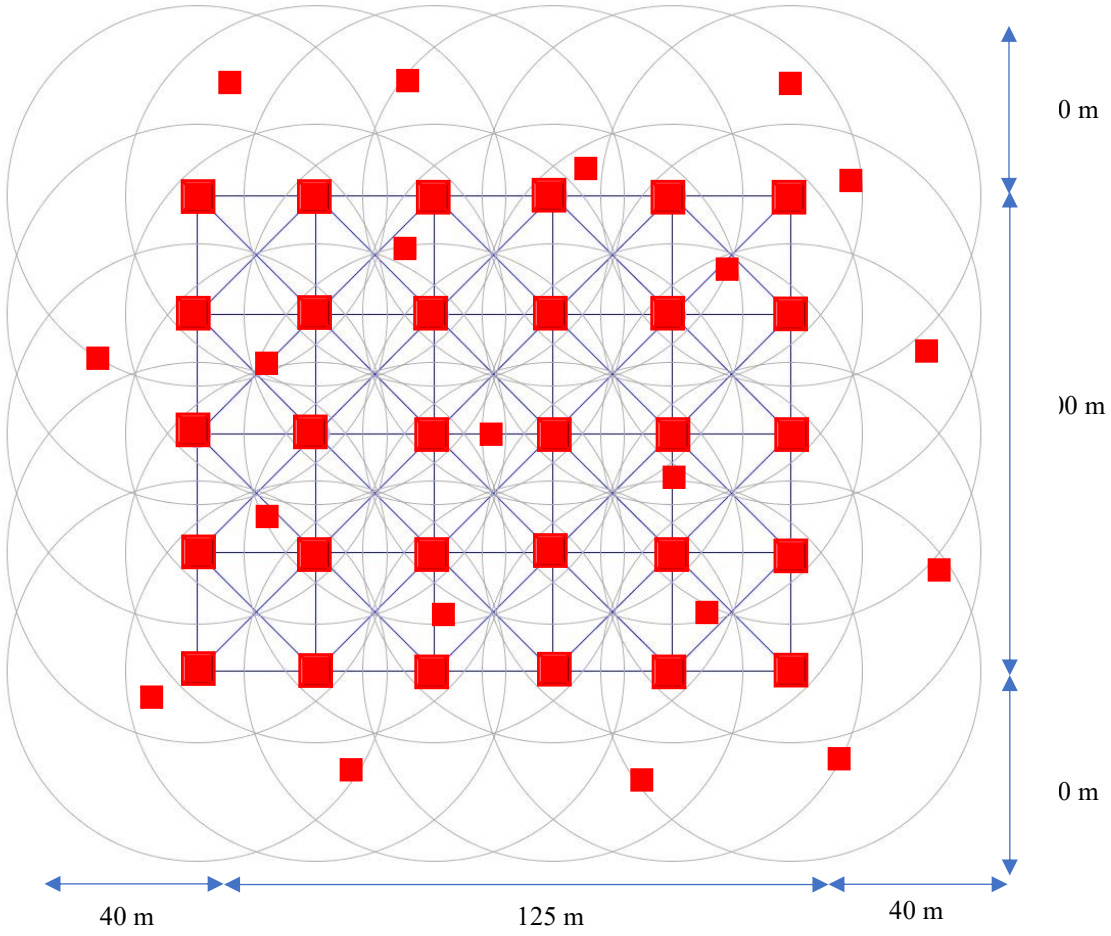


Figure 4.4: Mobile Network Model for the Traditional MWN

Figure 4.3 shows the model of the largest mobile network experiment consisting of 50 nodes, including 30 APs, 19 stations and the controller-station in the SDMWN architecture. It also shows the maximum 205 m x 180 m grid transmission area formed by (grid) APs, in a 6 x 5 grid network configuration. The nineteen (red) square nodes are the randomly-scattered stations, the thirty (blue) round nodes are the grid APs, and the (red-green) square node (at the centre) is the controller. Next, Figure 4.4 shows an equivalent model of the largest mobile network experiment for our traditional MWN architecture, consisting of 50 stations – 30 grid stations and 20 randomly-scattered stations. It also shows the maximum

205 m x 180 m grid transmission area formed by the grid stations, in a 6 x 5 grid network configuration. And though the grid stations may appear bigger than the randomly-scattered stations, all stations in the traditional MWN have identical physical/network features.

For the potential effects of mobility on the network performance of both MWN architectures to be well observed and analysed in a progressive manner, the mobile network experiment is further divided into a partial mobility scenario and a full mobility scenario. In the case of partial mobility, a set of nodes are mobile while others remain static, in both MWN architectures. For the SDMWN architecture, all randomly-scattered stations but the controller-station become mobile, within the grid transmission area, while the grid APs and the controller, earlier mentioned, remain static. Correspondingly, for the traditional MWN architecture, grid stations are static while the randomly-scattered stations but one (imitating the SDMWN controller) become mobile, within the grid transmission area as well.

Next, during full mobility, every node becomes mobile in both MWN architectures. These include all stations, APs and even the controller-station in the SDMWN architecture. The same system applies to the traditional MWN architecture, where all stations are mobile as well. And while there are no (visible) grid configurations in the full mobility scenario, mobile nodes are still constrained to the corresponding grid transmission areas of the respective partial mobility scenario. These area constraints are used in both scenarios to promote the existence of potential communication links, as long as possible.

For the sake of simplicity and general performance testing, mobile nodes in both mobility scenarios employ Random Waypoint (RWP) mobility model with area constraints and a constant node speed of 10 m/s. RWP is the mobility model most widely used among

researchers, for MWN experiments. It enables mobile nodes to move around freely within an area. The fixed node speed of 10 m/s is selected based on the relatively small grid area (25 m x 25 m) of the mobile network model in both MWN architectures. Therefore, it takes a period of about 2.5 to 3.5 seconds for a mobile node to travel from one grid point or area to another. And this whole procedure applies to both MWN architectures.

Mininet-WiFi has provisions for different mobility models and we are specifically leveraging its exclusive *net.startMobility* [51] function that comes with a standard RWP mobility feature. This function allows us to specify minimum and maximum speeds and (x,y) positions. Also, Mininet-WiFi governs node associations or connections either using calculated signal strength or load level metrics. For our experiments, we are employing the Strongest-Signal-First (SSF) [51] association control mechanism, to guide AP-station associations in SDMWN. Once again, to determine the statistical significance of our test results, each mobile network experiment is performed 10 times, using varying seed values to further introduce noise or randomness to the experiment. And ensuing mean and standard deviation values are used to obtain margins of error, within the 95% confidence interval. We have generally chosen to limit all tests to 10 runs due to the significant amount of time taken per emulation, and the high fidelity and lack of repeatability of the emulator [45].

4.5 Performance Expectation

Generally, the network performance of a routing protocol or method is almost guaranteed to worsen with increasing external events like mobility, and we can assume the same for our mobile network experiments when we progress from the partial mobility scenario to

the full mobility scenario. Therefore, we expect the network performance of centralized routing and distributed routing, in respective MWN architectures, to degrade during this process. Particularly, metrics like PSR and RTT are very susceptible to uncertainties presented by network topology changes as active links may be broken or lose quality (in terms of signal strength and throughput) during mobility events, and it may take some time for new/quality links to be obtained, if at all possible.

Furthermore, our preliminary inquiries and assessments suggest that our proposed SDMWN architecture, based on the centralized routing method, will perform considerably better in a number of ways than our traditional MWN architecture. This is especially true for the mobile network scenarios due to provisions put in place for mobility management in the SDMWN architecture. For instance, IEEE 802.11s operations efficiently ensure that connected APs in the mesh backbone are fully meshed, most of the time. Therefore, the controller is not bothered with AP mobility based on its (permanent) fully meshed view of connected APs in the OpenFlow network. Instead, the controller primarily handles station mobility using the host tracker, which can swiftly detect station movements as they associate with (new) APs.

In contrast, distributed routing with OLSR in our traditional MWN architecture depends on Hello and TC messages, exchanged between participating nodes, to detect topology changes in the network. This allows topology update information to be propagated across the entire network in a distributed manner. It also suggests that the mobility management procedure and subsequent network convergence period will be considerably longer with OLSR in traditional MWN, compared to centralized routing in

SDMWN. Moreover, we can estimate from the (plausible) mobility management procedure in the centralized routing approach, that the corresponding SDMWN architecture will outperform our traditional MWN architecture, in terms of PSR, under identical conditions.

However, in favour of distributed routing in our traditional MWN architecture, we estimate NRO will be lower, compared to centralized routing in the SDMWN architecture. Our assertion is backed by the fact that there are more control message types (e.g. beacons) generated at relatively higher frequencies in SDMWN, compared to our traditional MWN with OLSR. Similarly, in support of OLSR, MPRs, responsible for dispersing TC messages across the entire network, are carefully selected to reduce the number of control messages exchanged. This helps avoid unnecessary flooding, reduce redundant retransmissions and subsequently minimize overhead in our traditional MWN architecture.

Finally, we expect negative effects of network partitioning such as limited/zero connectivity, caused by mobility, to be more pronounced in the SDMWN architecture, compared to the traditional MWN architecture. This is mostly due to the general limitation of SDN, and consequently SDMWN, with regards to architectural requirements. One such requirement in SDMWN is that APs must have active paths or links to the controller, at every point in time, to maintain controller-AP sessions for plane-to-plane communication. Another requirement is that stations need to be associated with APs, at every point in time, for successful data communication. Based on these requirements, we believe the traditional MWN architecture is effectively more resilient against network partitioning than the SDMWN architecture. We also predict the architectural requirement concerning stations will eventually affect route selection and path lengths in SDMWN. This is because, in the

SDMWN architecture, neighbour or adjacent stations always need to communicate through associated AP(s), and possibly the controller, in a multi-hop manner. In contrast, with no such requirements in our traditional MWN architecture, neighbour or adjacent stations can communicate in a single-hop/direct manner, facilitated by distributed routing with OLSR. Furthermore, every station is a potential forwarding or intermediate node in the traditional MWN, with no path selection constraints. In summary, we generally expect the average hop-count or path-length statistics to be lower for distributed routing with OLSR, compared to centralized routing in SDMWN, under identical conditions.

4.6 Summary

In this chapter, we started by discussing the respective implementations of SDMWN and the traditional MWN to be compared. Next, we provided detailed descriptions of the performance metrics selected for evaluation in our comparative analysis of the SDMWN and traditional MWN architectures. After establishing the performance metrics to be tested and the methods of data collection for our study, we described our emulation setup and procedures for carrying out different performance tests in both MWN architectures, under identical conditions. To conclude, we discussed our predictions of emulation results from the comparative analysis between centralized routing in SDMWN and distributed routing in traditional MWN, to be presented and analyzed in the next chapter – Chapter 5.

Chapter 5

Emulation Results

5.1 Introduction

In this chapter, we present and compare performance test results of centralized routing in our SDMWN architecture against distributed routing in our corresponding traditional MWN architecture. This is the final step towards achieving the main goal of this thesis, which is to investigate whether centralized routing in SDMWN should be widely employed as an alternative for distributed routing in traditional MWN, to optimize MWN operations, control and performance. Results and analysis presented here are based on our designs, implementations, experiments and procedures, fully explained in the previous chapter.

The rest of this chapter is organized as follows. Section 5.2 presents emulation results from our static network experiments, including throughput, control message rate and average hop-count tests. These are subsequently followed by analyses and discussions on the performance behaviour of both MWN architectures. Section 5.3 presents emulation results from our mobile network experiments, including the partial and full mobility scenarios, as well as corresponding discussions on the comparative analysis of centralized routing versus distributed routing in our respective MWN architectures.

5.2 Static Network Results

The first set of static network results – obtained from our throughput tests – is illustrated in Figure 5.1 below. It is part of the sanity checks, to confirm that both MWN architectures are indeed comparable, which will allow objective conclusions to be drawn at the end of our comparative analysis. Also, such comparison is fair because there is only a single route, with the same number of hops and transmission ranges, between end-stations in both cases.

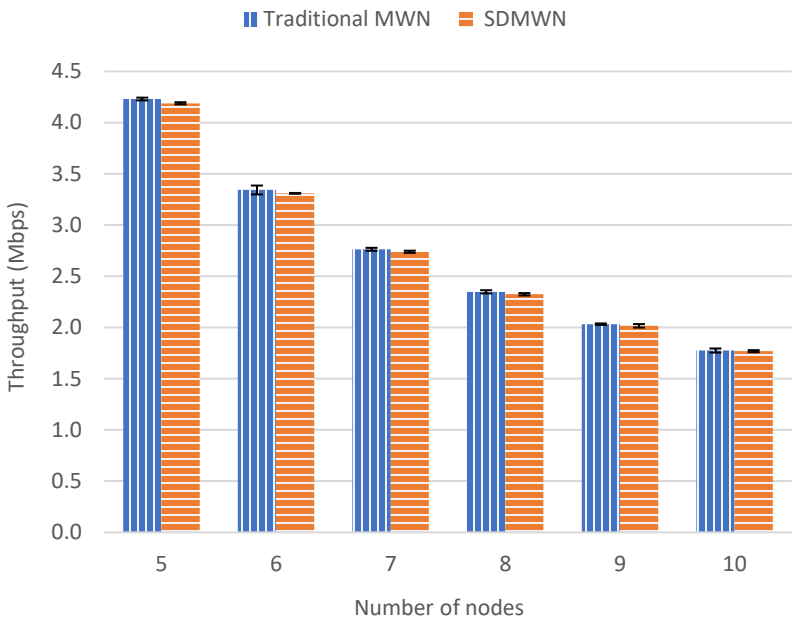


Figure 5.1: Static Network – Throughput Test

Figure 5.1 illustrates identical throughput statistics for both MWN architectures as hop count is varied in our static networks, irrespective of the differences in the design and setup of both MWN architectures. Such behaviour can be attributed to the single-route provision in both (static) MWN architectures. Additionally, external factors such as interference and signal strength, which affect throughput performance in our emulated environment, appear

to be consistent across both MWN architectures. This mainly confirms our initial theory that wireless channels are utilized in a very similar manner in both MWN architectures and provides us with confidence going forward in this study. As both MWN architectures are indeed comparable, it is fair to generalize our analysis of the throughput performance.

Interference in our emulated environment is based on wmediumd – a wireless medium simulation application – which operates under the concept of a single wireless medium. In this case, all wireless nodes in the network interfere with each another, irrespective of their positions [51]. Wmediumd emulates interference using existing signal strength levels and bit rates to calculate network attributes such as bandwidth, loss, latency and delay, based on a signal table provided in [54]. Signal strength, derived from Received Signal Strength Indication (RSSI), depends on the existing (log-distance) propagation model and the distance between adjacent nodes [51]. This wireless simulation approach is highly recommended for MWN implementations in Mininet-WiFi [51].

From Figure 5.1, throughput starts at approximately 4.2 Mbps which is the highest value attained in a static network of 5 nodes (i.e. 4 hops). Subsequent throughput values begin to decrease in a (rough) geometric sequence, all the way down to about 1.8 Mbps, as the number of nodes/hops in the network increases to 10 nodes (i.e. 9 hops). A reason for such behaviour in our emulated environment – based on wmediumd – is that network resources like the bandwidth/wireless channel are (equally) shared amongst all transmitting nodes in the network, irrespective of position. Therefore, throughput in the static network is inversely proportional to the total number of hops or transmitting nodes in the network. With this knowledge, we can also estimate the throughput for a single-hop transmission in

a static network of 2 nodes to be around 16 to 17 Mbps (i.e. 4.2 x 4 or 1.8 x 9). This is calculated using the throughput statistics of the smallest and largest static networks of 4 hops (or 5 nodes) and 9 hops (or 10 nodes) respectively. To confirm our prediction, we performed additional throughput tests between two adjacent nodes, where emulation results fell right within the (16 – 17) Mbps range. And this shows that our initial theory of bandwidth sharing in our emulated environment is indeed true. Moreover, our results fall close to an expected throughput of 18.676 Mbps, as seen using the *iw* [51] and *iwconfig* [51] utilities on Mininet-WiFi. The expected throughput, defined by wmediumd, is derived from the signal table for a bit rate of 24 Mbps (802.11g WLAN modulation standard) and RSSI of -82 dBm – both seen using *iw* [51] and *iwconfig* [51] as well.

The second set of results obtained during topology discovery in our static network experiments, for both MWN architectures with 10 nodes, are illustrated in Figures 5.2 and 5.3 below. These form part of our baseline tests, which investigates the number of control messages generated in the SDMWN and traditional MWN architectures respectively.

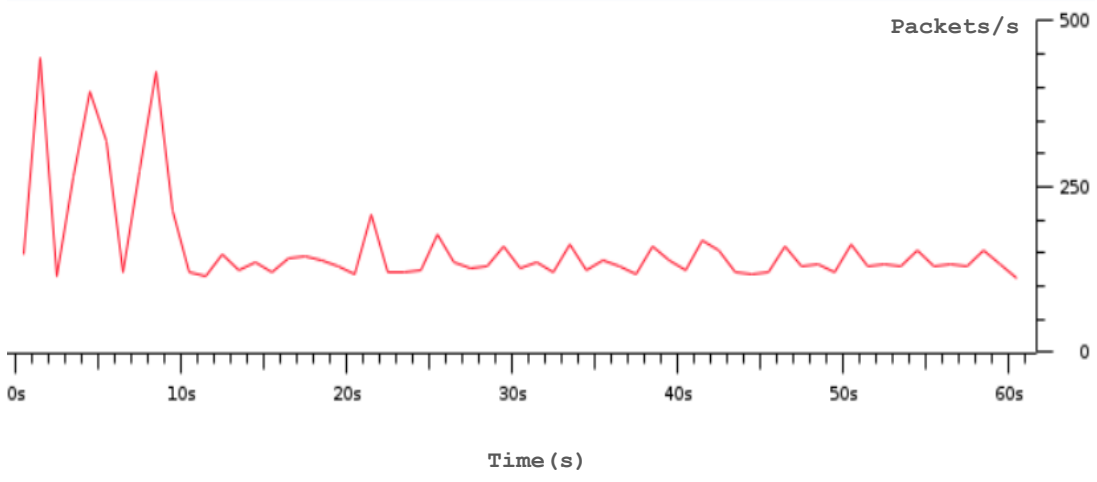


Figure 5.2: Static SDMWN – Control Message Rate

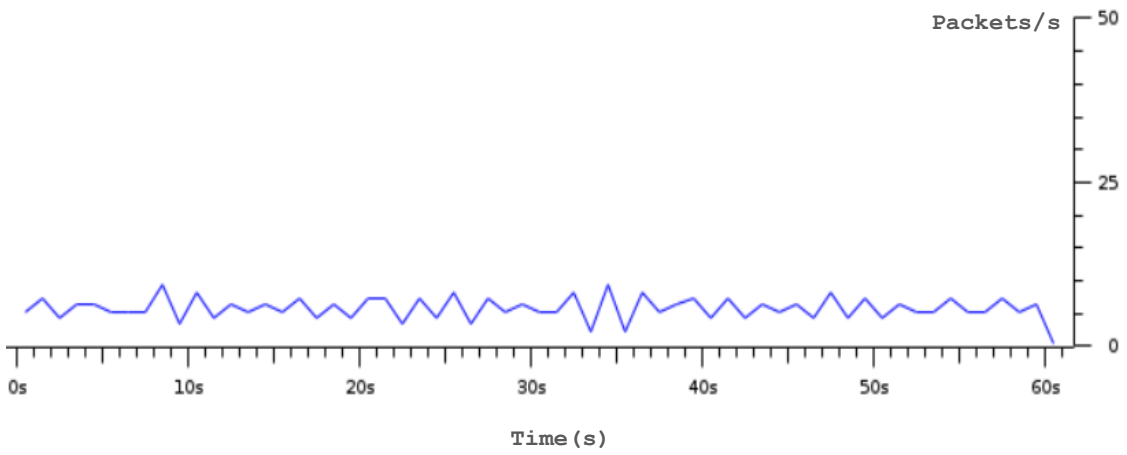


Figure 5.3: Static Traditional MWN – Control Message Rate

Figure 5.2 and Figure 5.3 provide an insight into the average rate at which control messages are generated/exchanged during topology discovery in the SDMWN and traditional MWN architectures individually. To provide accurate depictions of the average (control) message rate, these graphs were obtained directly from Wireshark, using its I/O Graph feature. For a high-level illustration of the control messages generated in both MWN architectures, Figure 5.4 is a composite bar graph of the MAC layer and upper layer control messages in SDMWN, followed by the Hello & TC messages in traditional MWN. These statistics were extracted directly from the Wireshark statistics-summary; with the SDMWN architecture, as expected, generating far more control messages than the traditional MWN architecture. For SDMWN, we further discovered that, out of 10,989 control messages generated during the 60-second topology discovery period, control activities in the MAC layer account for 6,997 or roughly 64% of such messages. This mainly consists of 4,116 beacons, generated by APs to facilitate AP-station associations. And it is about the same as the theoretical value of 4,200 beacons (i.e. 60 x 10 x 7) – estimated using the default interval of 0.1 second

or 10 beacons per second [55], for 7 APs in the SDMWN architecture. This default beacon interval delivers decent performances in most wireless applications [56].

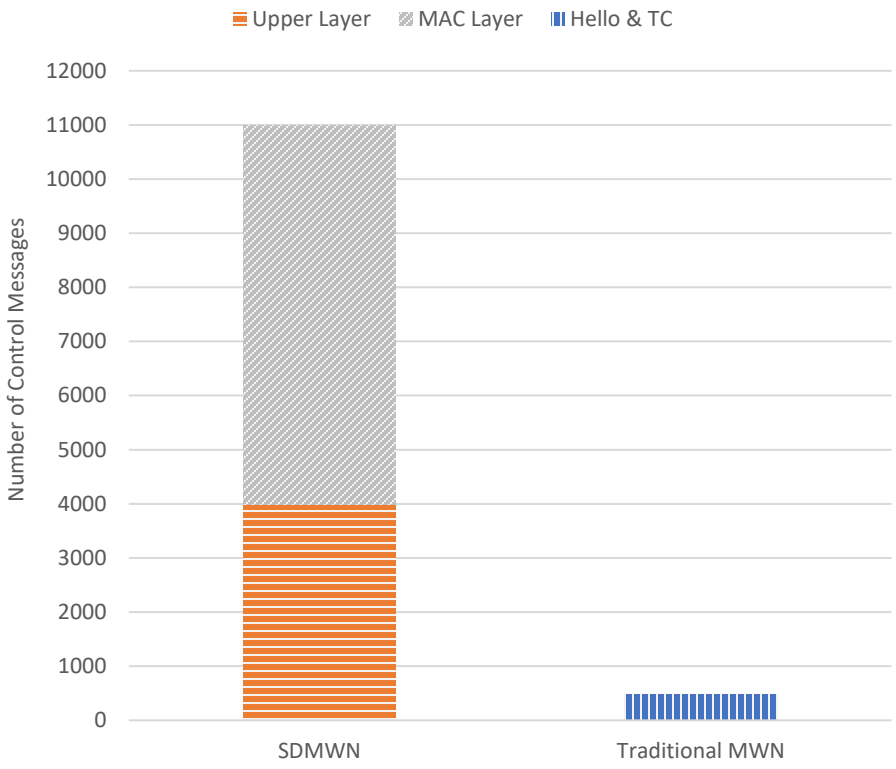


Figure 5.4: Static Network – Number of Control Messages

The next set of MAC layer control messages are 475 mesh beacons, for IEEE 802.11s mesh discovery and peering operations in SDMWN. This is approximately equal to an estimated value of 480 mesh beacons (i.e. 60 x 8). Mesh beacons are generated at 1-second intervals by 8 nodes (7 APs and the controller) that form the mesh backbone network. The last major set of (SDMWN) control messages in the MAC layer are IEEE 802.11 acknowledgement frames. The 2,260 acknowledgement frames are generated by the controller-station and its connected AP, as responses, to manage unicast upper layer (TCP and OF) exchanges.

For the upper layer control messages – 36% of control messages in the SDMWN architecture – there are 1,105 TCP messages and 2,887 OF messages. Also, due to the mesh connectivity provided by IEEE 802.11s, all 7 APs and the controller are involved in the upper layer exchanges. TCP messages are exchanged between APs and the controller, to establish and manage AP-controller sessions that support plane-plane connectivity. TCP connections are initiated by APs using SYN messages, which the controller responds to with corresponding ACK/SYN messages. Subsequent ACK messages are used to maintain such connections. On the other hand, OF message exchanges between the controller and APs are initiated by the controller. These mainly include 2,387 LLDP packets, which play a major role in OFDP topology discovery.

We can further estimate the expected number of LLDP packets exchanged during the entire 60-second period since we know that OFDP topology discovery is performed at every 5-second interval. We will also consider the fact that LLDP packets are encapsulated in Packet-Out and Packet-In messages, generated by the controllers and APs respectively. Furthermore, with each AP having two interfaces, the controller sends separate Packet-Out messages per interface. Taking another look at Figure 4.1, the controller is located in the middle of the 7 APs in the SDMWN architecture, where end APs are 3 and 4 links away on the left-hand side and right-hand side respectively. Consequently, for the Packet-Out messages, there should be 32 (i.e. $2\{1 + 2 + 3\} + 2\{1 + 2 + 3 + 4\}$) LLDP packets, sent by the controller and distributed among all 7 APs, in a single round of OFDP topology discovery. For the resultant LLDP packets, sent by each AP and received on every mesh interface, the total number of LLDP packet transmissions per (mesh) broadcast LLDP

packet is 8 (i.e. 7 APs plus the controller, all part of the mesh backbone). So, with 7 APs each sending out an LLDP (mesh) broadcast packet, there should be 56 (i.e. 8 x 7) LLDP packets, transmitted in a single round of OFDP topology discovery. Lastly, for the Packet-In messages, there should be 16 (i.e. {1 + 2 + 3} + {1 + 2 + 3 + 4}) LLDP packets sent by all 7 APs to the controller, per each LLDP (mesh) broadcast packet. So, with 7 APs each sending out an LLDP (mesh) broadcast packet, there should be 112 (i.e. 16 x 7) LLDP packets sent towards the controller, during a single round OFDP topology discovery. In the end, we expect 200 LLDP packets in total, for a single round of OFDP topology discovery This then translates to 2,400 (200 x 12) LLDP packets, for 12 rounds of OFDP topology discovery, which is about the same value as the 2,387 LLDP packets, obtained from the Wireshark statistics-summary, for the entire 60-second period.

Next, from Figure 5.2, there are noticeably high spikes with a peak (average) packet rate of about 450 packets per second, around the first 8 seconds of topology discovery. Afterwards, the average packet rate drops to a steady rate with peaks of about 150 packets per second, throughout the remaining 52 seconds of the topology discovery period. This high control message rate or spike can be attributed to initial interactions between the controller and APs, during initial AP-controller session establishments and (one-time) AP discovery stage. During the initial topology discovery phase in SDMWN, there is an above-average exchange of TCP and OF packets, which includes the exchange of information about AP capabilities. Afterwards, there is a relatively steady period of control traffic for maintaining AP-controller connectivity. To support our assertions, we took a closer look at control messages generated during this 8-second surge and discovered about a 10% shift

in the ratio of MAC layer to upper layer control messages, when compared to the (64% to 36%) ratio obtained for the entire 60-second period. This 10% shift, towards the upper layer control messages, is due to increased TCP and OF network activities during the first 8 seconds, while major MAC layer message rates like beacons and mesh beacons remained constant throughout the entire topology discovery period.

On the other hand, Figure 5.3 and Figure 5.4 show much fewer control messages generated during topology discovery in our traditional MWN architecture, compared to the SDMWN architecture. Also, topology discovery analysis is less complicated in this case as it only involves OLSR (Hello and TC) control messages. From the Wireshark statistics-summary, there are 473 control messages, generated by every station running OLSR in the traditional MWN architecture. Though our Hello and TC intervals are set to 2 seconds and 5 seconds respectively, it is difficult to use such information to estimate the total number of control messages in the traditional MWN architecture. This is because OLSR employs a unified packet format for communication, and one or more Hello/TC messages can be encapsulated within a single packet [12]. This mode of operation was verified with Wireshark, where packets contained multiple control messages with the same header format but varying originator or neighbour addresses. However, we can use these (2-second and 5-second) intervals to establish an upper limit for the total number of control messages in our traditional MWN, assuming each Hello/TC message is represented by a single packet. Therefore, with 10 nodes generating Hello packets at 2-second intervals, we expect a total of 300 (i.e. 10 x 60/2) Hello messages at most, over the 60-second period. And with 8 MPR nodes generating TC messages at 5-second intervals and all MPR nodes re-

broadcasting each TC message, we expect a total of 768 (i.e. 8 x 8 x 60/5) TC messages at most, over the 60-second period as well. These values translate to a maximum total of 1068 control messages expected in our traditional MWN architecture.

From the results and analysis, we can easily conclude that there are far more control messages generated in the SDMWN architecture, compared to the traditional SDMWN architecture. In the SDMWN architecture, an average control message rate of 183 packets per second is estimated for all nodes, over the entire 60-second period. However, this includes the 8-second period of peak control message rates. And with this 8-second surge being a one-time cost, we can separately estimate the control message rate during the subsequent steady state. For the remaining 52 seconds of steady control traffic in SDMWN, with roughly 8,194 control messages, the new control message rate translates to about 137 packets per second. On the other hand, the traditional MWN architecture has an average control message rate of 6 packets per second, over the entire 60-second period. However, this estimation is based on the Wireshark statistics-summary, where one or more Hello/TC messages are encapsulated into a single packet. For the estimated upper limit or maximum total of 1,068 control messages, estimated for the same 60-second period, we obtain a maximum control message rate of about 18 packets per second.

Using the control message statistics obtained for the best/worst case scenarios, we see that the SDMWN architecture generates about 8 to 30 times more control messages than the traditional MWN architecture, at any given period in time. The lower bound (8) is obtained from the ratio of the (lower) steady control message rate in SDMWN to the upper limit control message rate in our traditional MWN (i.e. 137/18). And the upper bound (30)

is obtained from the ratio of the overall control message rate (that includes the 8-second surge) in SDMWN to control message rate for combined TC & Hello messages in our traditional MWN (i.e. 183/6).

In addition, we can conclude that the SDMWN architecture will always generate more control messages than our traditional MWN architecture, even as network size grows. Our conclusion is based on the fact that both MWN architectures have control message rates that grow quadratically with a significant number of nodes in the network. For SDMWN, the number of LLDP packets (particularly LLDP broadcasts and resulting Packet-In messages) grows quadratically with the number of APs in the mesh backbone, even as other control messages like beacons and mesh beacons grow linearly. In the same way, TC messages – generated by each MPR and re-broadcasted by every other MPR in our traditional MWN – grow quadratically with the number of MPRs, even as Hello messages grow linearly with network size.

The final set of results, in our static network experiments, are the hop-count or path-length tests for both MWN architectures, illustrated in Figure 5.5 and Figure 5.6 below. From Figure 5.5, distributed routing with OLSR in our traditional MWN architecture constantly adopts the (ideal) shortest possible path between vertex stations, as network size is varied. This is largely expected because of its shortest path or Dijkstra algorithm, which only employs hop count as its only path selection metric. However, this is not the case in SDMWN where longer path lengths involving more hops or nodes (about 1.25 times our traditional MWN) are typically selected. One feature common to both MWN architectures is that the average grid hop count becomes steady – from 30 to 50 nodes – after a steep

rise. This hop-count pattern is attributed to our grid configurations for 30, 40 and 50 nodes, having 18 (6 x 3), 24 (6 x 4) and 30 (6 x 5) grid nodes/configurations respectively. For all three grid configurations, the shortest possible path length between the diagonal vertices is exactly 7, while the longest possible path lengths range from 9 to 11. This steady hop-count pattern is further highlighted by lower bounds of the confidence intervals in the path-length statistics, for 30, 40 and 50 nodes in SDMWN. And though the shortest possible path of 7 is never attained during this steady hop-count period in SDMWN, the lower bounds (indicating the most likely shortest path lengths) are about the same constant value of 7.8. This is unlike the corresponding upper bounds – indicating the most likely longest path lengths. Variations in the upper bounds are much more pronounced because the longest possible path lengths, for the steady path-length period, vary from 9 to 11.

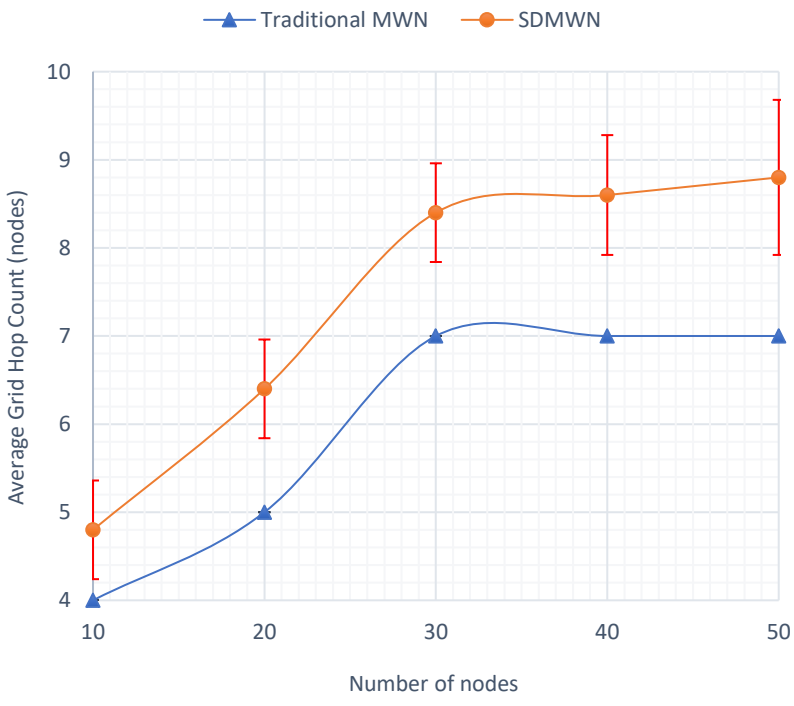


Figure 5.5: Static Network – Average Grid Hop Count

On the other hand, there are no confidence intervals for the grid hop-count statistics of the traditional MWN architecture. This is because, unlike SDMWN, optimum shortest paths (based on hop count only) are always selected by OLSR. A reasonable explanation for the unusual hop-count behaviour in SDMWN is based on IEEE 802.11s, which provides the multi-hop communication framework between APs in the mesh backbone. Unlike OLSR, IEEE 802.11s uses the composite ALM metric – a radio-aware path selection metric that estimates link quality, by considering packet loss probability and bit rate of the link, in addition to hop count. Therefore, the shortest possible path is not always selected as the best path in this case.

Based on our (25 m x 25 m) square grid configurations, we can deduce that the diagonal links of 35 m must be leveraged as much as possible to obtain the shortest possible paths between distant vertices. This is because, in the grid topology, a diagonal link (35 m) between two diagonally adjacent nodes is equivalent to a pair of horizontal and vertical links (25 m each) between the same diagonally adjacent nodes. And this is the technique widely employed by OLSR in the traditional MWN architecture. In contrast, IEEE 802.11s in the SDMWN architecture rarely selects the (35 m) diagonal links. Instead, IEEE 802.11s frequently opts for the shorter vertical/horizontal links (25 m each). Since bandwidth, loss, latency delay and consequently and link quality in our emulated environment (based on wmediumd) are estimated using the distance between nodes [51], this explains why the (shorter) horizontal and vertical links can be selected over the (longer) diagonal links. It also shows that (IEEE 802.11s) ALM metric tries to find a balance between hop count and link quality, as much as possible.

In theory, the IEEE 802.11s path selection technique in SDMWN increases RTT and energy consumption in (large) networks because it requires more participating nodes and networks resources, compared to OLSR in traditional MWN. Nonetheless, judging by wmediumd's mode of operation, we believe this approach can still be useful in terms of overall throughput performance. To support our claim, we performed additional throughput and RTT tests, using a simple static network of 2 to 3 nodes, in a line topology. Based on our knowledge that one diagonal link is equivalent to a pair of horizontal and vertical links, in the grid topology, we measured the throughput performance for 2 nodes connected by a single link of 35 m, and 3 nodes with two (multi-hop) 25 m links. The additional tests demonstrated higher throughput statistics, about 8 Mbps, for the network with two 25m links. As expected in a two-hop network, this translates to an estimated 50% drop in the maximum achievable throughput of about 17 Mbps – which we already established during our sanity checks. On the other hand, the network with the single 35 m link has a lower throughput of about 6 Mbps, which translates to a higher 65% drop, using the maximum achievable throughput as the reference throughput as well. In our RTT tests, we discover that average RTT, starting at about 1 ms to 2 ms in both cases, increases by about 0.5 ms, for a new (25 m) link added This subsequently validates our initial assertion that the path selection technique in IEEE 802.11s outperforms that of OLSR, in terms of throughput performance. However, this comes at a relatively small increase in RTT.

Lastly, from Figure 5.6, distributed routing with OLSR in our traditional MWN architecture also demonstrates lower average hop-count statistics, compared to centralized routing in the SDMWN architecture.

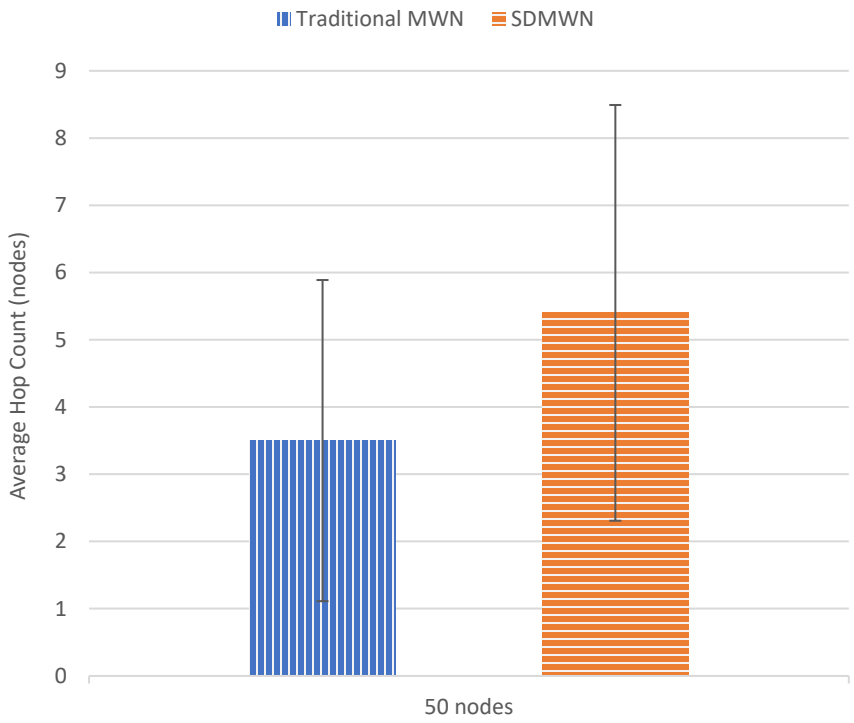


Figure 5.6: Static Network – Average Hop Count

This additional hop-count test, performed with the largest static (grid) network of 50 nodes, addresses more random network states. In this case, the average hop count or path length in SDMWN is about 1.5 times that of our traditional MWN. However, this (new) factor is considerably larger than the previous factor of 1.25, obtained for the initial (grid) hop-count test. A simple explanation for this increase is based on the enforcement of the SDMWN architectural requirement, whereby stations cannot communicate directly with each other and can only communicate using APs as intermediaries, even when such stations fall within transmission range of each other. In contrast, there is no such architectural requirement in our traditional MWN architecture, as adjacent stations can communicate directly with each other and every station, not necessarily the grid stations, can serve as an intermediary node.

5.3 Mobile Network Results

This section compares and analyses the network performance of both MWN architectures, in terms of convergence time, PSR, RTT and NRO. These metrics are observed under varying network sizes, for partial mobility and full mobility scenarios.

5.3.1 Partial Mobility Scenario

The partial mobility scenario is designed such that the availability of potential links between every node is always guaranteed, via the grid network topology. The first set of results, presented for the partial mobility scenario, are the convergence time performance statistics of both MWN architectures, illustrated in Figure 5.7 below.

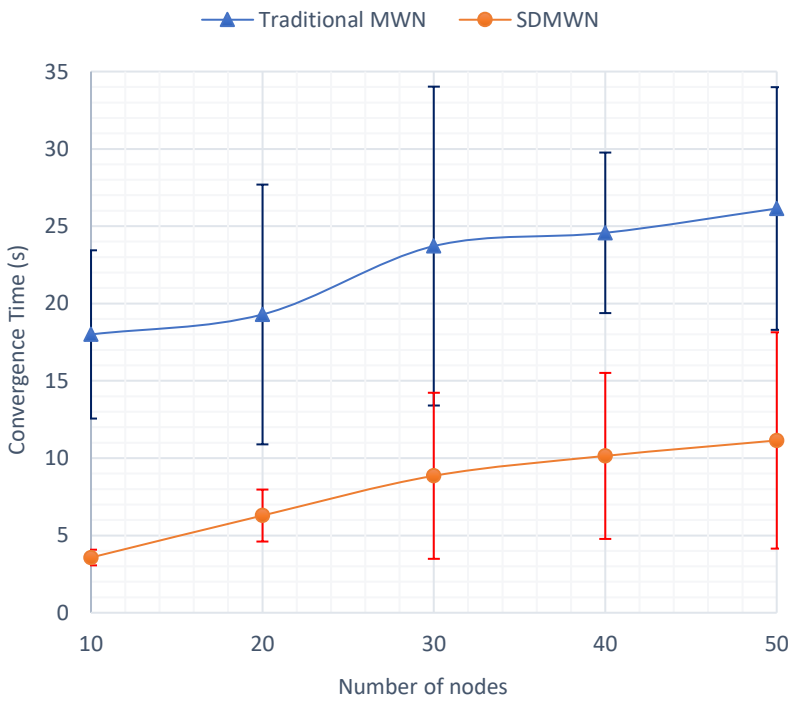


Figure 5.7: Partial Mobility Scenario – Convergence Time

We are starting with convergence time because the convergence time performance of each routing approach, in their respective MWN architectures, is expected to have a significant impact on the outcome of other performance tests. From Figure 5.7, centralized routing in the SDMWN architecture generally outperforms distributed routing in the traditional MWN architecture. This suggests that the average time taken for relevant topology updates, due to mobility, to reflect across the network, and then possibly allow data transmission is shorter in SDMWN, compared to traditional MWN. More specifically, looking at the distance between reference points on the graph, we can estimate that convergence time is approximately 15 seconds shorter with centralized routing in SDMWN, compared to distributed routing in traditional MWN. Nonetheless, one feature common to both routing approaches is that network size has no significant effect on convergence time. We reached this conclusion based on the overlapping confidence intervals, for the convergence time statistics of each routing approach. This shows that the difference in the convergence time performances, under varying network sizes, is not statistically significant, for each MWN architecture. An explanation for such an outcome is provided later in our analysis.

With grid nodes remaining static throughout the partial mobility scenario, for both MWN architectures, mobility management and network convergence focus mainly on the (randomly-scattered) mobile stations. And this is where the major benefits of SDMWN's centralized routing approach are most evident. With no resultant topology changes in the mesh backbone and besides the mobile stations having to locally discover and re-associate with new APs, only the SDMWN controller is responsible for (globally) detecting topology changes and enforcing topology updates across relevant parts of the network. This method

of operation is particularly true when the communicating pair of stations are distant or at least 3 hops away from each other, while connected to separate APs. Otherwise, when the communicating pair of stations are connected to the same AP, mobility management and network convergence only occur locally. So, for each (distant) station moving from one AP to another, only a fixed set of 3 nodes (i.e. the station, AP and controller) will actively take part in detecting topology changes and sending topology updates to relevant parts of the SDMWN architecture. On the other hand, with OLSR in our traditional MWN, there is a relatively higher number of nodes – including the static/grid nodes – involved in detecting similar topology changes and sending topology updates across the entire network instead. This also considers the fact that, even though topology changes are usually detected locally by neighbours of the mobile station, OLSR still requires topology updates to be distributed across the entire traditional MWN architecture. Consequently, longer periods of time are taken by the relatively higher number of participating nodes to propagate topology updates across the entire network, compared to having just 3 nodes operating within smaller areas in the SDMWN architecture. And the distinct network behaviours underscore why mobility management is more effective in SDMWN, resulting in much faster convergence times, compared to the distributed approach in our traditional MWN.

Our argument is supported by the MOVE event in SDMWN, as the mobile stations disassociate/associate with (new) APs, during the partial mobility scenario. For AP-station association, mobile stations immediately send out LLC messages to the APs, which are eventually received by the controller. The host tracker of the controller then uses the information contained in the LLC messages to detect the topology changes, within a short

period of time. The estimated period for the host tracker to detect such topology changes in SDMWN is effectively less than 1 second. This assertion is based on the default beacon interval of 0.1 second, for IEEE 802.11 beacons generated by APs. With this IEEE 802.11 provision, mobile stations can swiftly sense and associate with (new) APs, which also translates to the host tracker being immensely effective at detecting topology changes. The efficiency of the host tracker is also facilitated by the fact that the controller and APs are static members of the mesh backbone network, thereby having stable routes between each other. However, the efficiency of the host tracker does not account for the overall network convergence time in SDMWN. This is because the controller must subsequently enforce topology updates on relevant areas of the network, by performing flow installations on appropriate APs, to complete the network convergence process. And this additional period used for flow installations can range from about 0 to 20 seconds, based on the moving-sender/moving-receiver cases, also described for the MOVE event. The moving-sender case is much more effective because flow installations are immediately performed, unlike the moving-receiver case, which depends on the hard timeout for new flow installations. These variations in the flow installation period, particularly during the moving-receiver case, is a plausible explanation as to why we have overlapping confidence intervals in the convergence time statistics for the centralized routing approach in SDMWN.

For the traditional MWN architecture, where mobility management and network convergence require more time and resources, the estimated period for locally detecting topology changes can be as much as 6 seconds. Our assertion is based on a neighbour-state holding time, which specifies a timeout period for received Hello messages, after which

the neighbour-state information becomes obsolete. [59, 60]. And while the mobile stations can easily detect and be detected by new neighbours, through Hello messages, it can take a relatively long time for its old neighbours to locally detect such topology change, using the timeout period. By default, this period is set to three times the Hello interval [12, 59]. However, this period does not account for the overall convergence time in the traditional MWN as well. To complete the network convergence process in the traditional MWN architecture, MPRs must globally propagate topology updates, by subsequently dispersing TC messages across the entire network. Each MPR periodically generates a TC message every 5 seconds, which is then (re-)broadcasted throughout the entire network, by all other MPRs. The additional amount of time – required for dispersing TC messages across the entire network – is the major reason why the overall convergence time is approximately 15 seconds longer in the traditional MWN, compared to SDMWN. And even though extra TC messages may be sent at much higher frequencies than the default interval, during special cases such as changes to the MPR Selector set [12], the traditional MWN is highly unlikely to match the convergence time performance of the SDMWN architecture. Again, this is because the centralized approach in SDMWN generally involves fewer nodes, without requiring the dispersal of topology updates across the entire network.

Lastly, for the overlapping confidence intervals in the convergence time statistics of the traditional MWN, this can be attributed to the stations moving out of range, just after the last Hello message or just before the next Hello message. In the first case, it takes a relatively long time to (locally) detect the topology changes. In contrast, such detection occurs sooner, for stations moving out of range just before the next Hello message.

The next sets of results, presented for the partial mobility scenario, are the PSR and RTT performance statistics of both MWN architectures, illustrated in Figures 5.8 and 5.9 respectively. Since both performance metrics are largely influenced in a similar manner, by the same network degradation factors, they can then be analysed collectively.

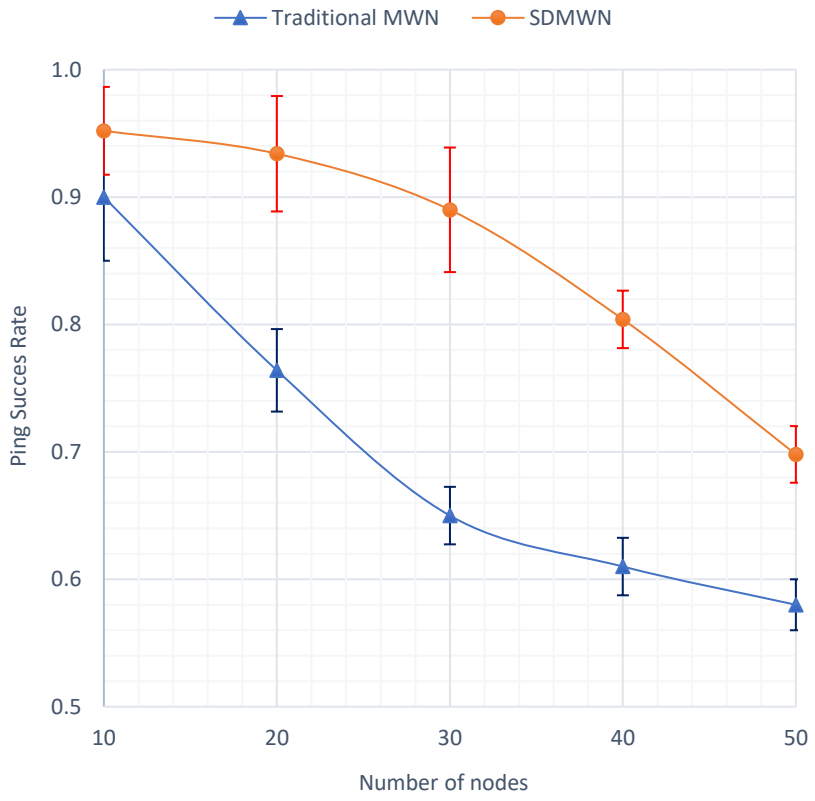


Figure 5.8: Partial Mobility Scenario – PSR

From Figure 5.8 and Figure 5.9, the respective PSR and RTT performance statistics, for each MWN architecture, significantly deteriorate as network size increases. As previously confirmed in our baseline tests, path lengths and the corresponding time taken for packets to travel between communicating pair of nodes increase with growing network sizes. This particularly affects the RTT performance metric, as the period and probability of successful

packet delivery worsen with time. This is due to network uncertainties associated with link breakages and network convergence, in the partial mobility scenario.

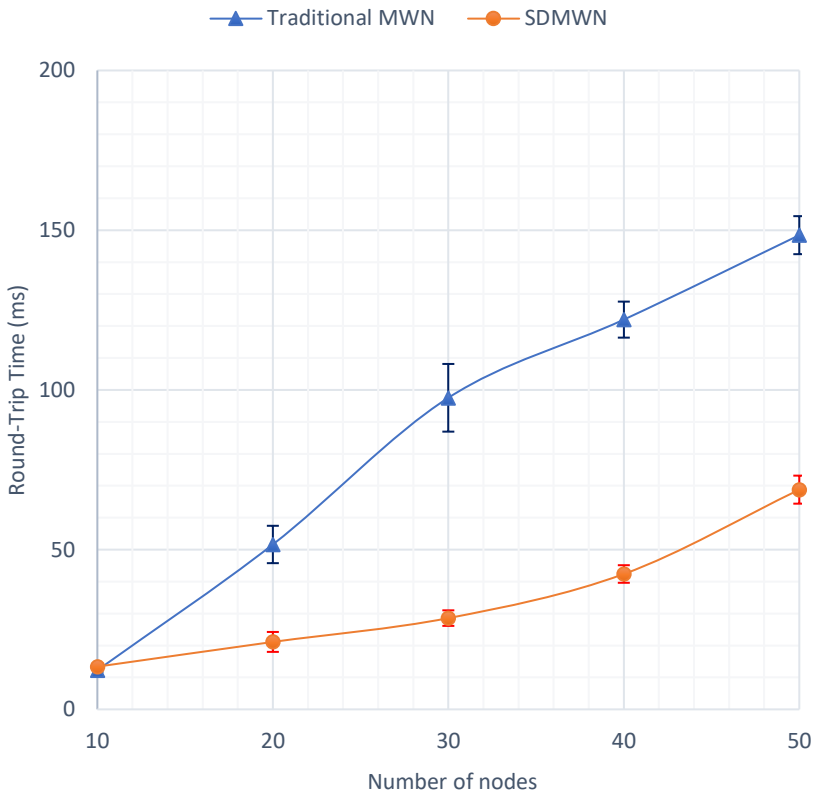


Figure 5.9: Partial Mobility Scenario – RTT

Other features that both MWN architectures have in common are the similar PSR and RTT statistics, for the smallest network size of 10 nodes. This behaviour is more pronounced for the RTT performance, where both approaches methods have identical starting points on the graph. And for the PSR performance, the performance difference is very small, compared to differences in the larger networks. However, as the network size grows, centralized routing in SDMWN outperforms distributed routing in our traditional MWN, with average differences of about 0.15 (15%) and 65 ms in the PSR and RTT performances respectively.

We have various explanations to support such outcomes. The most plausible explanation is based on SDMWN's architectural requirement, which has no provision for adjacent stations to communicate directly with each other, except via the AP infrastructure. Therefore, in the partial mobility scenario, where grid APs and the controller in the mesh backbone are static, this restricts the occurrence of topology changes and resulting link breakages to the mobile stations only. In this case, the more stable links of the grid network are mostly employed during communication. However, for traditional MWN, adjacent stations can simply communicate directly with each other and every station is a potential forwarding/intermediate node. And while there are also static/grid stations with more stable links, in the traditional MWN architecture, mobile stations are not required to select such links or stations as intermediaries. So, the possibility of link breakages and subsequent topology changes occurring between communicating stations is much higher in the traditional MWN, compared to SDMWN. These network disruptions and uncertainties negatively impact the PSR and RTT performances.

Next, the superior PSR and RTT performances in SDMWN, over the traditional MWN, can also be attributed to the quality of links selected for communication. However, this is not necessarily because of a (standard) feature or provision of the centralized routing approach, as opposed to the distributed routing approach. This is basically a feature of the (underlying) IEEE 802.11s protocol/ALM path selection metric in the mesh backbone. We already established in our grid hop-count test that IEEE 802.11s usually selects the shorter (25 m) links of relatively high quality and throughput. And since communication between mobile stations in SDMWN is always done via static/grid APs in the mesh backbone, this

ensures that such quality links are often selected. Similarly, based on the SSF association control mechanism, the mobile stations generally choose to associate with the closest APs or the APs with the strongest links. This then suggests that the first and last hops in each end-to-end connection are of high quality as well. On the other hand, OLSR in traditional MWN is prone to selecting lower quality links, based on its (standard) hop-count metric. When mobile stations move apart from each other, resulting in unstable links, before possible link failure, OLSR may still select such links, based on hop count and irrespective of link quality. Also, even when the static/grid stations are employed as intermediaries, the longer 35m diagonal links are typically selected. And this particularly contributes to the superior PSR performance of SDMWN, over the traditional MWN. Such behaviour also explains why initial PSR and RTT statistics, for both MWN architectures, are much closer in the smallest-sized network of 10 nodes, before growing much wider as the network size increases. There are fewer links involved in this case and path length is relatively smaller. The average path length in the smallest network can be as low as 1, for adjacent stations in the traditional MWN architecture, and 2 for stations associated with the same APs in the SDMWN architecture. So, it is fair to say that the effects of topology changes, link quality and link failures are minimal in this case.

All things considered, it is highly unlikely that the distributed routing approach would surpass or even match the superior PSR and RTT performances of the centralized routing approach, irrespective of the (underlying) ALM path selection metric in SDMWN. This assertion is primarily based on SDMWN's architectural requirements, whereby the mobile stations are required to communicate via the (grid) AP infrastructure only. To begin

with, the (standard) SDN feature or architectural requirements positively impacts mobility management and overall network convergence, during partial mobility, as earlier shown in the convergence time performance results. So, the superior PDR and RTT performances in the SDMWN can partially be attributed to its superior convergence time performance. This is because the average time elapsed between (active) link failure and network convergence, for data communication between the mobile stations is relatively shorter in SDMWN, compared to our traditional MWN. And such network behaviour greatly favours both PSR and RTT performance metrics, which both significantly worsen with time. Lastly and more importantly, due to the SDMWN architectural requirement and regardless of the ALM path selection metric in the mesh backbone, fixed and consequently stable paths will naturally be employed in SDMWN, unlike the traditional MWN. This is because the mobile stations in SDMWN always communicate via the grid APs (and the controller), which provide the fixed or stable paths, due to zero topology changes in the grid network. In contrast, without such architectural requirement, the mobile stations in the traditional MWN do not always use the stable/fixed paths, provided by the grid stations. Therefore, the mobile stations are susceptible to dynamic and relatively unstable paths, provided by other mobile stations.

The final set of results, presented for the partial mobility scenario, are the NRO performance statistics of both MWN architectures, illustrated in Figure 5.10 below. This indicates the number of control messages generated per successful data packet delivery. From Figure 5.10, it is evident that the NRO statistics of both MWN architectures increase with growing network sizes. This means that the number of control messages involved in mobility management and network convergence, for the successful delivery of data packets

in both MWN architectures, increases with the number of participating nodes in the network. Also, the differences in the NRO performances results are statistically significant, judging by the non-overlapping confidence intervals, for each routing approach.

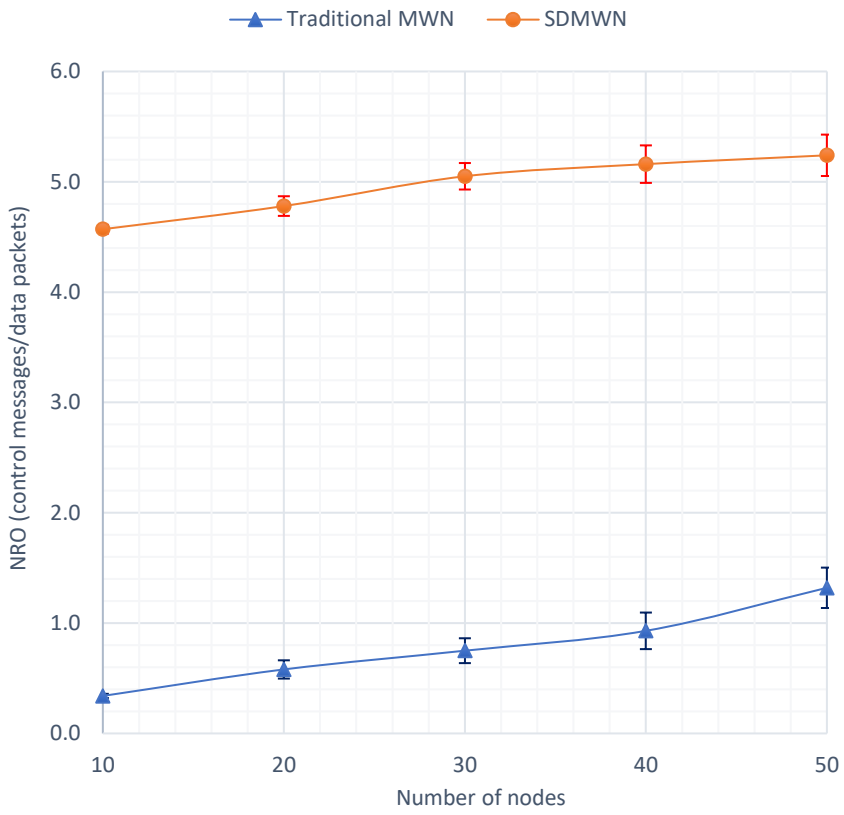


Figure 5.10: Partial Mobility Scenario – NRO

In SDMWN, the increasing NRO performance statistics, with the growing network size, can be attributed to a corresponding increase in the number of APs in the mesh backbone. Based on our design, APs make up 60 % of the SDMWN architecture, for all network sizes. This mostly leads to a quadratic growth in the number of (LLDP) control messages exchanged in the SDMWN architecture, as confirmed in one of our baseline tests. For the

mobile stations in SDMWN, the introduction and following increase of mobile stations also contribute to the increasing number of control messages exchanged in the network. This is because additional control messages, such as LLC messages, are generated for increasing mobility management and network convergence events, also leading to a corresponding increase in the number of flow installations performed by the controller. A similar argument can also be made for the traditional MWN, as the number of (TC) control messages grows quadratically with the increasing number of MPRs. From snapshots of the grid topologies in the partial mobility scenario and supporting arguments in [57], we further deduced that the average number of MPRs is about 33% of the total number of nodes in the traditional MWN architecture. And this percentage was fairly constant, under varying network sizes, which suggests that the number of MPRs grows linearly with network size.

Finally, the NRO performance of the centralized routing approach in SDMWN is unsurprisingly higher or worse than that of distributed routing in the traditional MWN. Our expectation is based on deductions from one of our baseline tests, where the average control message rate in the SDMWN architecture was around 8 to 30 times that of traditional MWN architecture. Taking another look at reference points on Figure 5.10, we estimated the average NRO performance statistics of SDMWN to be roughly 9 times more than that of its distributed counterpart. However, this factor, 9, is much closer to the lower bounds of the (8 to 30) range, obtained for the average control message rate tests. One plausible explanation for such outcome points to the decrease in the TC interval and a corresponding increase in the frequency or amount of (TC) control messages dispersed across the entire network. And such behaviour mostly occurs due to special cases, such as changes to the

MPR Selector set, introduced by topology changes and network convergence in the partial mobility scenario. In this case, additional TC-messages are generated and (re-)broadcasted by the MPRs, to increase the network's reactiveness to link failures. Another visible explanation for the relatively low NRO factor is based on the inferior PSR performance of the traditional MWN architecture, compared to the SDMWN architecture. With fewer packets successfully delivered in the traditional MWN architecture, this further increases the resultant NRO performance statistics.

5.3.2 Full Mobility Scenario

The full mobility scenario is largely intended to test the robustness and resilience of the two routing approaches and corresponding MWN architectures, under identical conditions. Moreover, unlike the partial mobility scenario, this setup does not guarantee the availability of potential links between every node, as all nodes are mobile in both MWN architectures. The first set of results, presented for the full mobility scenario, are the convergence time performance statistics of both MWN architectures, illustrated in Figure 5.11 below. From Figure 5.11, distributed routing in our traditional MWN generally outperforms centralized routing in SDMWN. Unlike the partial mobility scenario results, this average time taken for relevant topology updates, due to full mobility, to reflect across the network, and then possibly allow communication is (about an average of 11 seconds) shorter in our traditional MWN architecture, compared to the SDMWN architecture. An explanation for SDMWN's poor performance is provided later in our analysis. Nonetheless, similar to the performance behaviour during partial mobility, network size has no significant effect on convergence

time performance of both MWN architectures. This is based on the overlapping confidence intervals, for each routing approach as well. And the reasons provided for the overlapping confidence intervals, during partial mobility, also apply to the full mobility scenario.

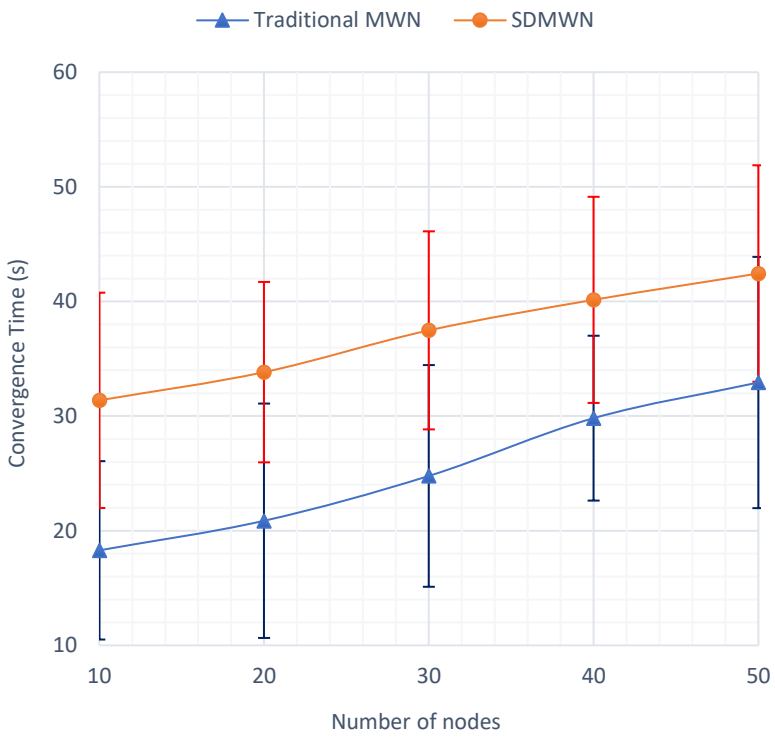


Figure 5.11: Full Mobility Scenario – Convergence Time

In addition, the overall convergence time performance generally deteriorates during the full mobility scenario, with respect to our earlier partial mobility results. The performance drop is mainly attributed to the occurrence of network partitioning events, in both MWN architectures, expected during full mobility. However, the effects of this type of events are more pronounced for centralized routing in SDMWN, with a time-increase factor of about 4.6, compared to distributed routing in the traditional MWN architecture, with a relatively

low time-increase factor of about 1.2. These average values are calculated with respect to the partial mobility scenario results. A plausible explanation for the higher time-increase factor in the convergence time performance of centralized routing is based on SDMWN's architectural requirements, particularly the one concerning plane-to-plane communication. And while this SDN feature is largely responsible for SDMWN's superior convergence time performance in the partial mobility scenario, it is at the same time harmful in the full mobility scenario. This is because APs require active links to the controller, to establish and maintain the AP-controller sessions, for effective plane-to-plane communication, and subsequent mobility management and network convergence events. During full mobility, (mobile) AP connectivity to the (mobile) controller is susceptible to link breakages and then network partitioning of relevant areas of the network. Such events disrupt plane-to-plane communication, thereby increasing the time taken for mobility management and network convergence, globally performed by the controller. From our inspection of the controller's activities, through the (POX) console, we also recorded about 3 to 5 separate periods – with no active AP-controller session or communication, due to mobility, link breakage and network partitioning events. This 3 to 5 range, for the number of periods that the controller was inaccessible, is constant for all network sizes. This can be attributed to the fact that the network density, or the ratio of the number of nodes to the network area, also remained constant under varying network sizes. Therefore, we can conclude that the negative impact of network partitioning is constant, under varying network sizes, in the SDMWN architecture. This conclusion also supports the argument that network size has no significant effect on the convergence time performance in the SDMWN architecture.

As earlier stated, the negative impact of network partitioning on the convergence time performance of the distributed routing approach is relatively low, with a time-increase factor of about 1.2, compared to the centralized routing approach. And this suggests that the traditional MWN is resilient and better equipped to operate in the full mobility scenario, over SDMWN. This can be attributed to the fact that every node in the traditional MWN is a potential forwarding node, as every node builds its own routing table. So, as long as the communicating pair of nodes is in the same partition, potential routes can be found with distributed routing, using nodes in that same partition. In contrast, for centralized routing, if the SDMWN controller is in a different partition, away from the communicating pair of nodes, potential routes (requiring controller-AP flow installations) cannot be found.

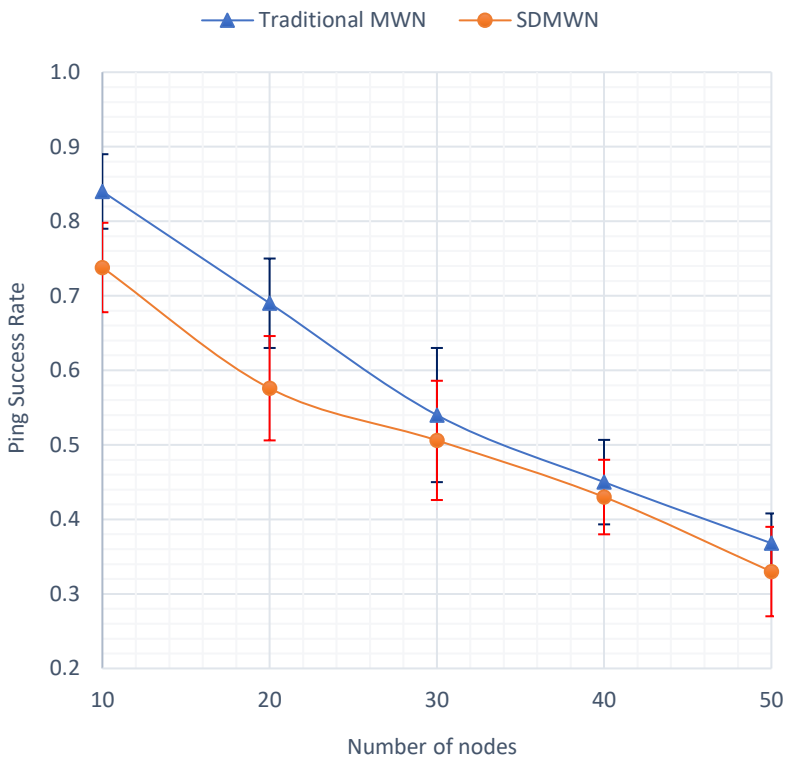


Figure 5.12: Full Mobility Scenario – PSR

The next set of results, presented for the full mobility scenario, are the PSR and RTT performance statistics, for both MWN architectures, illustrated in Figure 5.12 and Figure 5.13 respectively. Once again, these performance metrics are analysed collectively based on their shared characteristics.

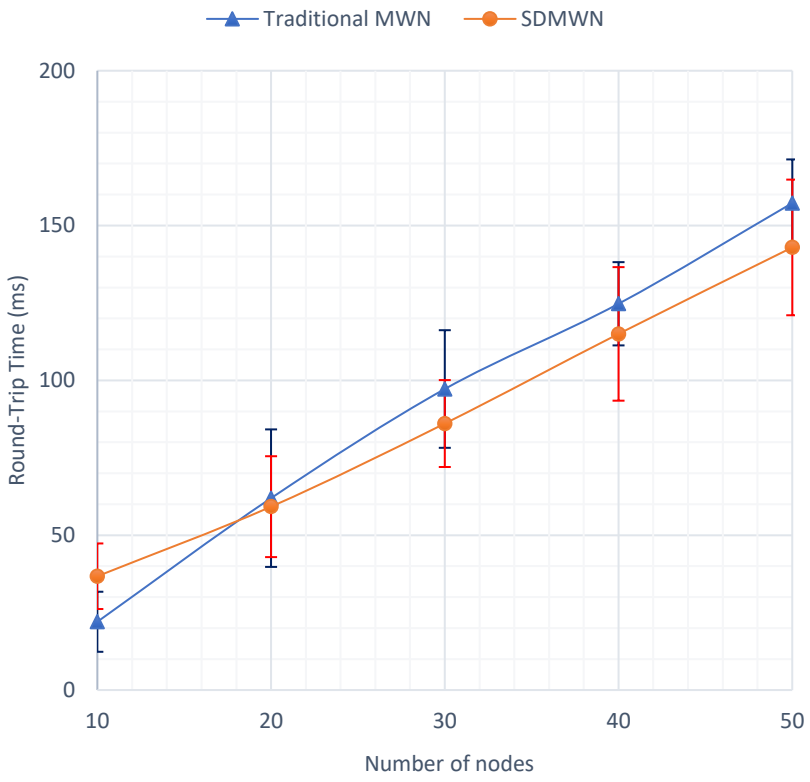


Figure 5.13: Full Mobility Scenario – RTT

From Figure 5.12 and Figure 5.13, the respective PSR and RTT performance statistics of both MWN architectures deteriorate as network size increases. And similar to the partial mobility scenario, such performance behaviour is attributed to the increasing path lengths and corresponding time taken for packets to travel between communicating pair of mobile nodes, as network size increases. However, unlike the partial mobility scenario, the PSR

and RTT performance differences for both routing approaches are much closer and, in most cases, not statistically significant. This can be attributed to the relatively high deterioration in the overall convergence time performance of the centralized routing approach, which is less suited to the full mobility scenario, compared the partial mobility scenario. And since the period and probability of successful packet delivery both deteriorate with time, the undesirable effects of full mobility (such as amplified packet delays and losses) on the PSR and RTT performance, are more pronounced in the SDMWN architecture, compared to the traditional MWN architecture. These assertions are supported by the approximately 40% drop in the average PSR performance of the SDMWN architecture, compared to a relatively low 17% drop in the equivalent traditional MWN architecture. In the same way, the average RTT performance of the SDMWN architecture deteriorates by a factor of about 2.5, compared to a factor of about 1.1, for the traditional MWN architecture. Again, these average values are all calculated with respect to the partial mobility scenario statistics.

Finally, taking another look at Figure 5.12 and Figure 5.13, the distributed routing approach slightly outperforms the centralized routing approach, in terms of PSR and RTT performances, for the smallest networks of 10 to 20 nodes. These particular PSR and RTT performance differences are indeed statistically significant, judging by the non-overlapping confidence intervals. However, as the network size grows from around 20 to 50 nodes, the PSR and RTT performance differences shrink and become statistically insignificant, even with the shorter convergence time performance in the traditional MWN. This performance behaviour can be attributed to the distributed routing operation, concerning the propagation of global topology updates across the entire mobile network. Since the number of MPRs

and subsequent TC message (re-)broadcasts increase with network size, these consequently reduce the odds and increase the period for successful completion of mobility management and network convergence operations, in the entire mobile network. And even though the traditional MWN generally has a faster convergence time than SDMWN, this does not completely translate to superior PSR and RTT performances in the full mobility scenario. A plausible explanation for such PSR and RTT outcomes is related to the fact that the convergence time performance difference of 11 seconds, during full mobility, is relatively low, compared to the convergence time performance difference of 15 seconds in the partial mobility scenario, where SDMWN's superior convergence time partially contributes to its undeniably superior PSR and RTT performances. In the end, both MWN architectures appear to match each other, in terms of PSR and RTT performance, during full mobility.

The final set of results, presented for the full mobility scenario, are the NRO performance statistics of both MWN architectures, illustrated in Figure 5.14 below. From Figure 5.14, the NRO performance statistics of both MWN architectures increase with the growing network size. So, similar to the partial mobility scenario, the number of control messages used for the mobility management, network convergence and successful delivery of data packets in both MWN architectures, increases with the number of (mobile) nodes in the network. The differences in the NRO performances results are also statistically significant, judging by the non-overlapping confidence intervals. Furthermore, the NRO performance of the centralized routing approach in SDMWN is unsurprisingly worse than that of distributed routing in the traditional MWN. And the same reasons provided for such outcomes, during the partial mobility scenario, also apply to the full mobility scenario.

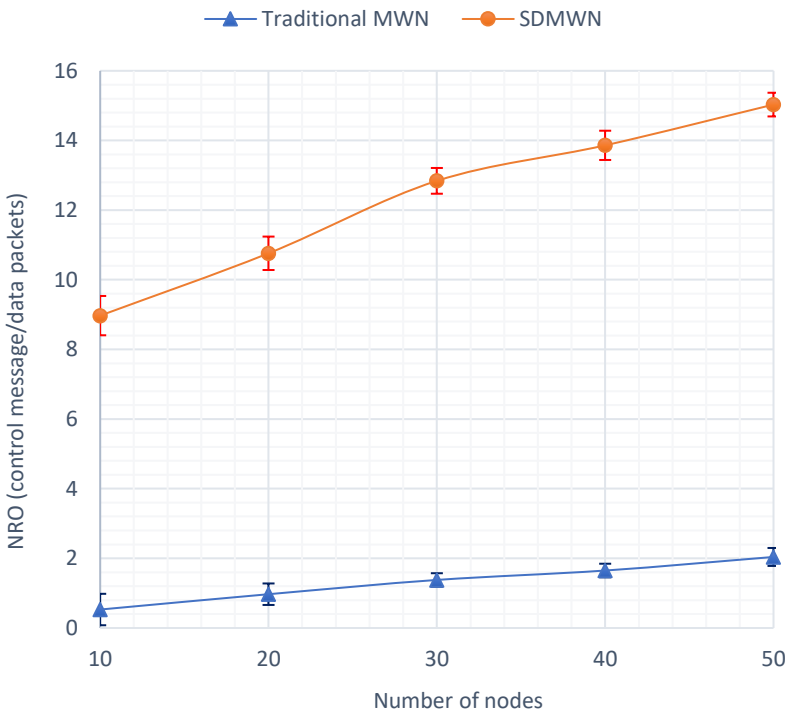


Figure 5.14: Full Mobility Scenario – NRO

Compared to the partial mobility scenario, there is a general rise in the NRO performance statistics of both MWN architectures. This is due to the increased topology changes, link breakages and network convergence events, associated with full mobility. Additionally, because there are fewer data packets successfully delivered, during full mobility in both MWN architectures, this also increases the resultant NRO performance statistics. In the case of distributed routing, additional TC messages are generated at higher frequencies, due to more mobility management and network convergence operations in the full mobility scenario, over the partial mobility scenario.

In the case of the centralized routing approach, the negative impact of full mobility on the NRO performance is more pronounced because its NRO performance deteriorates

more rapidly. This argument is further strengthened by the average NRO performance statistics of SDMWN, which is roughly 11 times more than that of the distributed routing approach. And this shows a substantial increase from the previous factor of 9, obtained for the partial mobility scenario. One plausible explanation for this outcome is based on the (mobile) controller having to perform additional AP-controller session establishments, due to network partitioning events. Besides, from previous baseline tests, we already confirmed that there is an above-average control message rate during (initial) AP-controller session establishments. And while this is more or less a one-time cost in the partial mobility scenario, we confirmed that such a process occurs about 3 to 5 times during full mobility, thereby contributing to the higher NRO performance statistics. Lastly, another reason for the increased NRO performance statistics of the centralized routing approach is based on IEEE 802.11s mobility management operations, due to the introduction of mobile nodes in the mesh backbone network. Therefore, there are more mesh discovery, peering and path computation operations the mesh backbone. This is unlike the partial mobility scenario, where all members of the mesh backbone are static, with no resultant link breakages.

5.4 Summary

In this chapter, we presented the emulation results, obtained from our comparative analysis between centralized routing in the SDMWN architecture and distributed routing in the traditional MWN architecture. We started with the static network experiments, consisting of sanity checks and baseline tests, to investigate the network behaviour of each MWN architecture. These provided the groundwork for our comparative analysis, useful during

subsequent mobile network experiments. Additionally, in this section, we demonstrated the identical throughput statistics of both MWN architectures, while the SDMWN architecture exhibited a higher average control message rate and increased average (grid) hop count. Next, during the mobile network experiments, we subjected both MWN architectures to rigorous performance tests, under identical conditions. For the partial mobility scenario, besides its high NRO cost, the centralized routing approach in the SDMWN architecture was evidently the superior routing approach. Finally, for the full mobility scenario, the (superior) performance of the centralized routing approach diminished substantially, at a higher rate than the (more resilient) distributed routing approach. In the end, both routing approaches performed rather poorly, particularly in terms of PSR and RTT, with no clear favourite. General conclusions and possible steps for improving the SDMWN architecture and style of operation, to be more resilient and better equipped to handle full mobility environments, are presented in the final chapter – Chapter 6.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we begin by highlighting the benefits of the centralized routing approach and potentials of the SDN-based MWN architecture, over distributed routing in the traditional MWN architecture. Such benefits include network manageability, flexibility, extensibility and global network visibility, to reduce network complexity and improve overall network performance. Similarly, we also highlight the few limitations of the centralized routing approach and SDN-based MWN architecture, most especially the SPOF and plane-to-plane connectivity issues, which in turn makes the traditional MWN architecture more resilient to unpredictable network conditions. We then take a critical look at some of the existing or relevant SDN-based MWN proposals, to identify significant limitations of such solutions, particularly concerning the feasibility of real-life application and execution. With all these in mind, we subsequently put forward our SDN-based MWN architecture, called SDMWN, that mainly addresses the practicality and mobility issues in existing SDN-based MWN solutions. The practicality of SDMWN relates to its full mobility capabilities, including the controller, in a completely wireless network. This is realized by employing IEEE 802.11s,

as the underlying protocol, to facilitate SDN operations. Nonetheless, SDMWN manages to preserve original features of the (wired) SDN architecture, including (standard) OFDP for topology discovery and OF-enabled APs as replacements for (conventional) switches.

In the same way, we also put forward a comparable traditional MWN architecture, using OLSR as the distributed routing candidate, to properly assess the performance of the centralized routing approach in SDMWN, and its ability to serve as an effective substitute. For the sake of clarity and objectivity, our traditional MWN also operates with the standard or recommended OLSR features and parameters. Equally, both MWN architectures are put through the same (static and mobile) network performance tests, under identical conditions. With these measures and procedures, we are able to appropriately deduce the extent to which the performance results of both MWN architectures can be attributed to the different routing approaches and MWN architectures, and the possible roles played by other (internal or external) features, such as the native protocol parameters and other supporting protocols.

In the end, we discover that SDMWN clearly outperforms the distributed routing approach in our traditional MWN, with about 15% and 65 ms performance gains for PSR and RTT respectively, when operating in a controlled (mobile) network environment, such as the partial mobility scenario. By ensuring the availability of (potential) communication links between every node, during the partial mobility scenario, we are able to eliminate, to a large extent, potential SPOF and plane-to-plane connectivity issues earlier mentioned. We mainly attribute the superior (PSR and RTT) SDMWN performance to the efficiency of the centralized routing approach, even with the presence of IEEE 802.11s and its ALM routing metric. There are two major reasons for this conclusion. The first being the efficient

mobility management and network convergence procedures in SDMWN. These typically involve a set of 3 nodes (i.e. the mobile station, grid AP and controller) in specific areas of the network, without the need to propagate topology updates across the entire network, as required with the distributed routing approach in our traditional MWN. Consequently, with the centralized routing approach, SDMWN consumes less time and resources, for mobility management and network convergence, during partial mobility. Secondly, mobile stations in SDMWN are required to communicate via the static or grid APs of the partial mobility scenario, using fixed/stable paths, as opposed to mobile stations in the traditional MWN, which are rather susceptible to dynamic and relatively unstable paths – provided by other mobile stations. This method of operation in SDMWN is totally unrelated to the underlying IEEE 802.11s protocol and its ALM path selection metric. Instead, our argument is solely based on the architectural requirement of SDMWN and SDN architectures in general, that is not found in the distributed routing approach of the traditional MWN architecture.

To conclude, the reasons given above refer to exclusive features of the centralized routing approach or the general SDN architecture, which largely improve the PSR and RTT performances of SDMWN, during partial mobility. Though, such performance comes at a much higher cost of network overhead, as there are more control message types – generated at higher frequencies in SDMWN, compared to the traditional MWN architecture. Then, for the full mobility scenario, the (superior) PSR and RTT performances of the centralized routing approach are mostly impacted by the uncontrolled mobile network environment. The reason is that, with no guarantee of potential communication links between every node, the possible network partitioning of the controller gives rise to the SPOF and plane-to-

plane connectivity issues. On the other hand, with no such concerns in the traditional MWN, the distributed routing approach is more resilient and less impacted by the negative effects of the full mobility scenario. Ultimately, based on these results and deductions, we strongly support the idea that SDN-based MWNs (such as SDMWN) should be widely employed as replacements for traditional MWNs, albeit under controlled mobile network conditions only. This then suggests that, even with the current capability and vast potential of the centralized routing approach in SDMWN, there are still significant steps and modifications that can be put towards its architecture and method of operations, particularly regarding network resilience, for it to be fully considered as a successor to the distributed routing approach in the traditional MWN. More so, we believe all these can be achieved without doing away with the original features of SDN, that make it remarkable.

6.2 Recommendations and Future Work

There is a wide array of improvements that can be made to the centralized routing approach in SDMWN, including some features that have already been implemented in existing SDN-based MWN proposals, to make the architecture more resilient to uncontrolled (mobile) network conditions. Moreover, many of these existing solutions are focused on addressing the SPOF, plane-to-plane connectivity and resultant network partitioning issues affecting SDN-based MWN. For instance, solutions such as [25], OLSR_SDN [26], [27] and SDN MANET [28] operate under the provision that the transmission range of the controller covers the entire network area, and consequently every node in the data plane. However, as earlier mentioned, there are lingering concerns about the practicality and scalability of

such solutions. Furthermore, other solutions like wmSDN [21], [24] and [25], with backup distributed routing mechanisms, to protect against possible controller failure, do not promote the original features of the centralized routing approach/SDN architecture.

Hence, to build upon the centralized routing approach in SDMWN, we recommend a provision be made to elect substitute or temporary controllers in specific areas or regions of the network, as backup controllers, whenever the primary or initial controller becomes unreachable. This recommendation can make the centralized routing approach in SDMWN more resilient to uncontrolled (mobile) network conditions, like the full mobility scenario, by mitigating the SPOF, plane-to-plane connectivity and subsequent network partitioning issues. In addition to such SDMWN provision, we suggest there should be only one active controller in a particular network or network partition, at every point in time, to maintain the unique qualities of the SDN architecture. We also expect these SDMWN enhancements to come at a higher cost of network overhead in the SDMWN architecture.

Next, with the new SDMWN provision in place, a more interesting scenario will be to implement comparable load balancing algorithms for both routing approaches, before testing the network performance of their respective MWN architectures. Load balancing is a methodology for distributing traffic load appropriately, across two or more nodes/paths for effective communication, to mitigate the potential effects of link failures in the network. For the centralized and distributed routing approaches, load balancing may improve energy efficiency, network resource utilization, traffic congestion, and so on, in their respective MWN architectures. Last but not least, it will also be interesting to test the scalability of both routing approaches and corresponding MWN architectures, using larger network sizes

of about 70 to 100 nodes, to determine whether/how the (superior) SDMWN performance is impacted by potential overload or bottleneck in the controller, compared to OLSR in the traditional MWN architecture. And such tests may also be extended for specific use cases like VANETs, using the Manhattan grid mobility model with varying node speeds.

References

[1] McKeown, Nick, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. “OpenFlow: enabling innovation in campus networks.” ACM SIGCOMM Computer Communication Review 38, no. 2 (2008): 69-74.

[2] Farhady, Hamid, HyunYong Lee, and Akihiro Nakao. “Software-defined networking: A survey.” Computer Networks 81 (2015): 79-95.

[3] Haque, Israat Tanzeena, and Nael Abu-Ghazaleh. “Wireless software defined networking: A survey and taxonomy.” IEEE Communications Surveys & Tutorials 18, no. 4 (2016): 2713-2737.y

[4] Jarraya, Yosr, Taous Madi, and Mourad Debbabi. “A survey and a layered taxonomy of software-defined networking.” IEEE communications surveys & tutorials 16, no. 4 (2014): 1955-1980.

[5] Kobo, Hlabishi I., Adnan M. Abu-Mahfouz, and Gerhard P. Hancke. “A Survey on Software-Defined Wireless Sensor Networks: Challenges and Design Requirements.” IEEE Access 5, no. 1 (2017): 1872-1899.

[6] Rawat, Danda B., and Swetha Reddy. “Recent advances on software defined wireless networking.” In SoutheastCon, 2016, pp. 1-8. IEEE, 2016.

[7] Alotaibi, Eiman, and Biswanath Mukherjee. “A survey on routing algorithms for wireless ad-hoc and mesh networks.” Computer networks 56, no. 2 (2012): 940-965.

[8] Holzle, U. “Inter-Datacenter WAN with centralized TE using SDN and OpenFlow.” Open Networking Summit (2012).

[9] Cox, Jacob H., Joaquin Chung, Sean Donovan, Jared Ivey, Russell J. Clark, George Riley, and Henry L. Owen. “Advancing Software-Defined Networks: A Survey.” IEEE Access 5 (2017): 25487-25526.

[10] Zahmatkesh, Afsane, and Thomas Kunz. “Software defined multihop wireless networks: Promises and challenges.” Journal of Communications and Networks 19, no. 6 (2017): 546-554.

[11] Kannhavong, Bounpadith, Hidehisa Nakayama, Nei Kato, Abbas Jamalipour, and Yoshiaki Nemoto. “A study of a routing attack in OLSR-based mobile ad hoc networks.” International Journal of Communication Systems 20, no. 11 (2007): 1245-1261.

[12] Clausen, Thomas, and Philippe Jacquet. Optimized link state routing protocol (OLSR). No. RFC 3626. 2003.

[13] Vats, Kuldeep, Monika Sachdeva, Krishan Saluja, and Amit Rathee. “Simulation and performance analysis of OLSR routing protocol using OPNET.” International Journal of Advanced Research in Computer Science and Software Engineering 2, no. 2 (2012).

[14] Huhtonen, Aleksandr. “Comparing AODV and OLSR routing protocols.” Telecommunications Software and Multimedia (2004): 1-9.

[15] Adjih, Cedric, Thomas Clausen, Philippe Jacquet, Anis Laouiti, Paul Muhlethaler, and Daniele Raffo. “Securing the OLSR protocol.” In Proceedings of Med-Hoc-Net, pp. 25-27. 2003.

[16] Yang, Mao, Yong Li, Depeng Jin, Lieguang Zeng, Xin Wu, and Athanasios V. Vasilakos. “Software-defined and virtualized future mobile and wireless networks: A survey.” Mobile Networks and Applications 20, no. 1 (2015): 4-18.

[17] Kreutz, Diego, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. “Software-defined networking: A comprehensive survey.” Proceedings of the IEEE 103, no. 1 (2015): 14-76.

[18] Azzouni, Abdelhadi, Nguyen Thi Mai Trang, Raouf Boutaba, and Guy Pujolle. “Limitations of openflow topology discovery protocol.” arXiv preprint arXiv:1705.00706 (2017).

[19] Wang, Junfeng, Yiming Miao, Ping Zhou, M. Shamim Hossain, and Sk Md Mizanur Rahman. “A software defined network routing in wireless multihop network.” Journal of Network and Computer Applications 85 (2017): 76-83.

[20] Wang, Junfeng, Ping Zhai, Yin Zhang, Lei Shi, Gaoxiang Wu, Xiaobo Shi, and Ping Zhou. “Software Defined Network Routing in Wireless Sensor Network.” In Cloud, Computing, Privacy in New Computing Environments, pp. 3-11. Springer, Cham, 2016.

[21] Detti, Andrea, Claudio Pisa, Stefano Salsano, and Nicola Blefari-Melazzi. “Wireless mesh software defined networks (wmSDN).” In Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on, pp. 89-95. IEEE, 2013.

[22] Fu, Hao, Yuan-an Liu, Kai-ming Liu, and Yuan-yuan Fan. “An SDN-based congestion-aware routing algorithm over wireless mesh networks.” In Wireless Communication and Sensor Network: Proceedings of the International Conference on Wireless Communication and Sensor Network (WCSN 2015), pp. 111-119. 2016.

[23] Zhu, Ming, Jiannong Cao, Deming Pang, Zongjian He, and Ming Xu. “SDN-based routing for efficient message propagation in VANET.” In International Conference on Wireless Algorithms, Systems, and Applications, pp. 788-797. Springer, Cham, 2015.

[24] Ku, Ian, You Lu, Mario Gerla, Francesco Ongaro, Rafael L. Gomes, and Eduardo Cerqueira. “Towards software-defined VANET: Architecture and services.” In Ad Hoc Networking Workshop (MED-HOC-NET), 2014 13th Annual Mediterranean, pp. 103-110. IEEE, 2014.

[25] Ku, Ian, You Lu, and Mario Gerla. “Software-defined mobile cloud: Architecture, services and use cases.” In Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International, pp. 1-6. IEEE, 2014.

[26] Labraoui, Mohamed, Michael Mathias Boc, and Anne Fladenmuller. “Software defined networking-assisted routing in wireless mesh networks.” In Wireless Communications and Mobile Computing Conference (IWCMC), 2016 International, pp. 377-382. IEEE, 2016.

[27] Labraoui, Mohamed, Charalampos Chatzinakis, Michael Mathias Boc, and Anne Fladenmuller. “On addressing mobility issues in wireless mesh networks using software-defined networking.” In Ubiquitous and Future Networks (ICUFN), 2016 Eighth International Conference on, pp. 903-908. IEEE, 2016.

[28] Hans, C. Yu, Giorgio Quer, and Ramesh R. Rao. “Wireless SDN mobile ad hoc network: From theory to practice.” In Communications (ICC), 2017 IEEE International Conference on, pp. 1-7. IEEE, 2017.

[29] Rault, Tifenn, Abdelmadjid Bouabdallah, and Yacine Challal. “Energy efficiency in wireless sensor networks: A top-down survey.” Computer Networks 67 (2014): 104-122.

[30] Salman, Ola, Raghid Morcel, Obada Al Zoubi, Imad Elhajj, Ayman Kayssi, and Ali Chehab. “Analysis of topology based routing protocols for VANETs in different environments.” In Multidisciplinary Conference on Engineering Technology (IMCET), IEEE International, pp. 27-31. IEEE, 2016.

[31] Pei, Guangyu, Mario Gerla, and Tsu-Wei Chen. “Fisheye state routing: A routing scheme for ad hoc wireless networks.” In Communications, 2000. ICC 2000. 2000 IEEE International Conference on, vol. 1, pp. 70-74. IEEE, 2000.

[32] Hiertz, Guido R., Dee Denteneer, Sebastian Max, Rakesh Taori, Javier Cardona, Lars Berlemann, and Bernhard Walke. “IEEE 802.11 s: the WLAN mesh standard.” IEEE Wireless Communications 17, no. 1 (2010).

[33] Henry, Jerome, and Marcus Burton. “802.11 s Mesh Networking.” CWNP Enterprise Wi-Fi Whitepaper (2011).

[34] Carrano, Ricardo C., Luiz CS Magalhães, Débora C. Muchaluat Saade, and Célio VN Albuquerque. “IEEE 802.11 s multihop MAC: A tutorial.” IEEE Communications Surveys & Tutorials 13, no. 1 (2011): 52-67.

[35] Alharbi, Talal, Marius Portmann, and Farzaneh Pakzad. “The (in) security of topology discovery in software defined networks.” In Local Computer Networks (LCN), 2015 IEEE 40th Conference on, pp. 502-505. IEEE, 2015.

[36] Pakzad, Farzaneh, Marius Portmann, Wee Lum Tan, and Jadwiga Indulska. “Efficient topology discovery in software defined networks.” In Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on, pp. 1-8. IEEE, 2014.

[37] Ochoa Aday, Leonardo, Cristina Cervelló Pastor, and Adriana Fernández Fernández. “Current trends of topology discovery in OpenFlow-based software defined networks.” (2015).

[38] Flowgrammable website, last accessed July 31, 2018, http://flowgrammable.org/sdn/openflow/message-layer/

[39] “IEEE 802 Logical Link Control”, Computer Science 07.442 IEEE 802 LAN standards, 9 May 2000, http://docshare01.docshare.tips/files/18359/183593941.pdf

[40] Computer Networking Demystified website, last accessed July 31, 2018, http://computernetworkingsimplified.in/data-link-layer/components-data-link-layer-llc-mac/

[41] Khan, Suleman, Abdullah Gani, Ainuddin Wahid Abdul Wahab, Mohsen Guizani, and Muhammad Khurram Khan. "Topology discovery in software defined networks: Threats, taxonomy, and state-of-the-art." IEEE Communications Surveys & Tutorials 19, no. 1 (2017): 303-324.

[42] Hong, Sungmin, Lei Xu, Haopei Wang, and Guofei Gu. "Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures." In NDSS. 2015.

[43] Azzouni, Abdelhadi, Othmen Braham, Nguyen Thi Mai Trang, Guy Pujolle, and Raouf Boutaba. "Fingerprinting OpenFlow controllers: The first step to attack an SDN control plane." arXiv preprint arXiv:1611.02370 (2016).

[44] Liu, Lei, Takehiro Tsuritani, Itsuro Morita, Hongxiang Guo, and Jian Wu. "Experimental validation and performance evaluation of OpenFlow-based wavelength path control in transparent optical networks." Optics Express 19, no. 27 (2011): 26578-26593.

[45] Fontes, Ramon R., Samira Afzal, Samuel HB Brito, Mateus AS Santos, and Christian Esteve Rothenberg. "Mininet-WiFi: Emulating software-defined wireless networks." In Network and Service Management (CNSM), 2015 11th International Conference on, pp. 384-389. IEEE, 2015.

[46] The POX network software platform website, last accessed July 31, 2018, https://noxrepo.github.io/pox-doc/html/

[47] OLSR daemon website, last accessed August 9, 2018, http://www.olsr.org/mediawiki/index.php/Main_Page

[48] Huston, Geoff. "Measuring ip network performance." The Internet Protocol Journal 6, no. 1 (2003): 2-19.

[49] Lamping, Ulf, Richard Sharpe, and Ed Warnicke. "Wireshark User's Guide for Wireshark 2.1." (2014).

[50] Barlow, D., A. D. Robbins, P. H. Rubin, R. Stallman, and P. V. Oostrum. "The AWK Manual." (1995).

[51] dos Reis Fontes, Ramon, and Christian Esteve Rothenberg. “The User Manual.” (2018).

[52] Patil, Annapurna P., Narmada Sambaturu, and Krittaya Chunhaviriyakul. “Convergence time evaluation of algorithms in MANETs.” arXiv preprint arXiv: 0910.1475 (2009).

[53] iPerf website, last accessed August 9, 2018, https://iperf.fr/

[54] Wmediumd signal table, last accessed August 14, 2018, https://github.com/ramonfontes/wmediumd/blob/mininet-wifi/tests/signal_table_ieee80211ax

[55] Velayos, Héctor, and Gunnar Karlsson. Techniques to reduce IEEE 802.11 b MAC layer handover time. Vol. 3. TRITA-IMIT-LCN, 2003.

[56] Geier, Jim. “802.11 Beacons Revealed.” Wi-Fi Planet (2002).

[57] Nguyen, Dang, and Pascale Minet. “Analysis of MPR Selection in the OLSR Protocol.” In null, pp. 887-892. IEEE, 2007.

[58] Mahfoudh, Saoucene, and Pascale Minet. “An energy efficient routing based on OLSR in wireless ad hoc and sensor networks.” In Advanced Information Networking and Applications-Workshops, 2008. AINAW 2008. 22nd International Conference on, pp. 1253-1259. IEEE, 2008.

[59] Huang, Yangcheng, Saleem N. Bhatti, and Daryl Parker. “Tuning olsr.” In Personal, Indoor and Mobile Radio Communications, 2006 IEEE 17th International Symposium on, pp. 1-5. IEEE, 2006.

[60] Hosek, Jiri, and Karol Molnar. “Investigation on OLSR routing protocol efficiency.” Recent Advances in Computers, Communications, Applied Social Science and Mathematics, ISBN (2011): 978-1.