Design and Implementation of a Software Defined Network Based Wireless Network Controller

Submitted in partial fulfillment of the requirements
for the degree of

Bachelor of Technology
in *Electrical Engineering*

&

Master of Technology
in *Communication and Signal Processing*

by

Ojas Kanhere
Roll No. 12D070002

Supervisor:

Prof. Abhay Karandikar

Department of Electrical Engineering
Indian Institute of Technology Bombay

2017

Dissertation Approval

This thesis entitled "Design and Implementation of a Software Defined Network based Wireless Network Controller" by Ojas Kanhere (12D070002) is approved for the degree of Master of Technology in Communication Engineering.

Examiners

Supervisor

Chairperson

Date : 20/6/2017

Place : Mumbai

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Ojas Kanhere
12D070002

21/6/2017

Contents

List of Figures **viii**

1 Introduction **1**

1.1 The SDN-based approach to network management and control 1

1.2 OpenFlow 2

1.3 SDN in Wireless Networks 3

1.4 Organisation of the Thesis 3

2 An SDN-based multi-RAT controller **5**

2.1 Middleware Description 7

3 Control of WLAN Networks **9**

3.1 SDN based Admission Control in WLAN Networks 9

3.1.1 AP Discovery 10

3.1.2 Authentication 11

3.1.3 Association 11

3.1.4 802.1x Authentication 11

3.2 Quality of Service (QoS) in WLAN 11

3.2.1 Dynamic Virtual Local Area Network (VLAN) assignment 12

3.3 WLAN Control in the context of Multi-RAT architecture 12

4 Management of WLAN Networks **15**

4.1 WLAN Management Protocols and their Drawbacks 15

4.1.1 SNMP 15

4.1.2 CAPWAP 16

4.1.3 TR-069 17

4.2 Management of Network devices using NETCONF 18

5 Implementation of the Controller 21

6 ns-3 simulations 25

6.1 Introduction 25

6.2 Network Topology Description 25

6.2.1 Traffic routing in ns-3 27

6.2.2 Link Capacity 27

6.2.3 Error Models 27

6.2.4 Parameters 28

6.3 Admission Control via Asssociation Request Forwarding 28

6.3.1 Implementation Details 28

6.4 Simulation of Netconf Messages 29

6.5 Test Cases and Results 30

6.5.1 Simulation Test Cases 30

7 Conclusions and Future Work 39

A Yang Model for WLAN APs 41

B Proposed Table of Contents for P1930.1 43

C OpenWrt configuration files 45

List of Figures

2.1 SDN based Multi-RAT wireless controller network 6
2.2 Middleware Architecture 6
2.3 IEEE P1930.1's homepage 8
3.1 Call flow for Admission to a WLAN Network, controlled by the multi-RAT controller 13
4.1 Example of a transaction between the ACS and AP in TR-069 18
5.1 Box diagram of implementation setup 21
5.2 Implementation Setup 24
6.1 An SDN based WLAN controller 26
6.2 A Non-SDN based WLAN controller 26
6.3 Passage of an association message through the OSI stack of network nodes 29
6.4 Test Case 1 topology 31
6.5 Average Station Throughput Comparision, Test Case 1 31
6.6 Test Case 2 topology 32
6.7 Average Station Throughput Comparision, Test Case 2 32
6.8 Average Setup Time Comparision, Test Case 2 33
6.9 Test Case 3 topology 33
6.10 Average Station Throughput Comparision, Test Case 3 34
6.11 Average Setup Time Comparision, Test Case 3 34
6.12 Test Case 4 topology 35
6.13 Average Setup Time Comparision, Test Case 4 36
6.14 Average Station Throughput Comparision, Test Case 4 36
6.15 Test Case 5 topology 37

6.16 Average Station Throughput Comparision, Test Case 5 38

6.17 Average Setup Time Comparision, Test Case 5 38

Acknowledgements

I would like to thank Professor Karandikar for the opportunity given to me to work on this project and for his guidance. I would also like to thank Mr. Pranav for the invaluable inputs he has provided.
I thank my lab mates, Rohan, Abhishek, Priyanka, Akshata, Arghyadip and Indu for their support in doing this project.

Ojas Kanhere

Abstract

Wireless Local Area Networks of today require a dense deployment of wireless Access Points , to handle the surge in the amount of traffic generated by an ever-increasing number of data users. In such scenarios, it is useful to control the Access Points via a central controller, to manage issues such as inter-Access Point interference and handover. In a Software Defined Network, since the data plane and control plane are separate, centralized management is facilitated. This thesis describes a way to bring the Software Defined Network paradigm into Wireless Local Area Networks. The control and management functions that need to be implemented in the wireless scenario are described. Implementation details for the WLAN controller have been provided. Finally, the thesis provides a quantitative measure of the gain attained in various network parameters such as throughput and wireless station setup time by bringing in the SDN paradigm, via simulations. It has been verified that the SDN solution scales well, as the number of Access Points is increased.

List of Acronymns

AP	Access Point
SDN	Software Defined Network
RIP	Routing Information Protocol
OSPF	Open Shortest Path First
TCP	Transmission Control Protocol
CAPWAP	Control and Provision of Wireless Access Points
WLAN	Wireless Local Area Network
RAT	Radio Access Technology
LTE	Long Term Evolution
STA	Station
QoS	Quality of Service
VLAN	Virtual Local Area Network
AAA	Authentication, Authorization and Accounting
IP	Internet Protocol
MAGW	Management Gateway
PNG	Packet Network Gateway
SNMP	Simple Network Management Protocol
ns-3	network simulator 3
P2P	Point to Point

Chapter 1

Introduction

The Government of India aims to provide primary broadband connectivity to 600 million homes across the nation, at the rate of 2Mbps by 2020[1]. To provide such high data requirements, urban regions shall be divided into small cells, thereby increasing the network capacity. Along with fixed data traffic, the steady increase of cellular internet connections has lead to a growth in mobile traffic. To address the demand for mobile internet and due to the lack of availablity of surplus cellular spectrum, service providers are increasingly looking to offload traffic to fixed or Wi-Fi networks. Because of its low cost and ubiquituous nature, Wi-Fi is the last-mile technology of choice. To provide the required capacity to enable operator offloading, an ultra-dense deployment of Wi-Fi will be set up. Wi-Fi shall provide primary nomadic broadband and an umbrella coverage of cellular technology will provide mobility support.

Owing to the fact that in deployment, there shall now be a very large number of Access Points (APs), a centralized, scalable controller is required. A Software Defined Network (SDN) based paradigm will address this need.

1.1 The SDN-based approach to network management and control

Switches consist of two components:- the data plane and the control plane. The Control Plane builds the Forwarding Table, required by the data plane, using routing tables provided to it. Routing protocols, such as the Routing Information Protocol (RIP) and the Open Shortest Path First (OSPF) protocol are implemented here. The data plane does the actual forwarding of packets entering the switch. SDN is a paradigm that reorganizes the network architecture by separating the data plane and control plane. It reduces the computational load on various network devices. Instead, all decisions are taken by the controller. SDN allows the network administrator a global view of the network, and provides a way to do real time changes in the network.

SDN introduces an abstraction layer, separating network intelligence from physical devices. This allows parallel development of software and hardware and makes it easier to experiment with new ideas and protocols.

1.2 OpenFlow

The OpenFlow[2] protocol is a commonly used protocol for the southbound interface of the controller. Messages are sent via the Transmission Control Protocol (TCP) transport protocol, on port 6633.

Traditional networking deals with packets of data. OpenFlow deals with Flows. A Flow is a stream of packets, through which applications deliver services. OpenFlow creates “flow rules”, for a set of flows. These flow rules “match” certain fields of the flows, like source and destination MAC address, VLAN ID, source and destination Internet Protocol (IP) address etc, against certain fixed values. The flow rules are then pushed to network switches, on the basis of which forwarding tables are generated. When a “match” occurs, the flow rule specifies an action which must be taken by the OpenFlow Switch. These actions can discard, modify forward or even queue the packet. If a flow matches no flow rule, an event called a “table miss” occurs. When a “table miss” occurs, the packet is either dropped or forwarded to the controller. It is assumed that the central controller will know what to do with the packet. The flows can then be added/deleted/modified in real time by the OpenFlow Controller.

1.3 SDN in Wireless Networks

The SDN paradigm has already been successfully implemented in wired networks. We hope to bring the advantages of SDN to wireless networks as well. To do so, we have centralized both the management plane and the control plane of the network. The data plane is located on physical switches (and APs), unlike the Control and Provision of Wireless Access Points (CAPWAP[3;4]) protocol (an existing management protocol for configuration of APs), where all data messages must be forwarded to the centralized controller. Such forwarding unnecessarily loads the network. Instead, it has been proposed to locally break out all data traffic.

The forwarding table of the data plane is built by the control plane via the OpenFlow protocol, as described above. The switches (and APs) are configured by the management plane via the NETCONF[5] protocol (a detailed description of NETCONF is given in Chapter 4).

1.4 Organisation of the Thesis

In this chapter, we have discussed the concepts of SDN, have talked about OpenFlow, a common southbound interface protocol used in SDN based wired controllers. We have talked about the need to bring the SDN paradigm to wireless networks. The rest of the thesis is organised as follows. Chapter 2 describes the design of an SDN based Wireless Controller. Chapter 3 and 4 describe the Control and Management Tasks that the wireless SDN controller must handle. Chapter 5 describes our implementation of an SDN based wireless controller, and the design of our in-lab testbed. In Chapter 6, we have described how we have modified ns-3 and used it to simulate the gains achieved when an SDN based Wireless Local Area Network (WLAN) controller is used, over a Non-SDN based WLAN controller. Finally, in Chapter 7 we conclude the thesis and describe the future work that needs to be carried out.

Chapter 2

An SDN-based multi-RAT controller

The Multi-Radio Access Technology (RAT) based wireless controller, shown in Figure 2.1 has been designed by us, so that a single communication system may provide services, typically provided by multiple systems.

Across different RAT, there are certain common features which are abstracted by the controller and controlled by a common controller block. For example, all access request messeges (the association request in WLAN and the attach request in Long Term Evolution (LTE) networks) could be controlled by a single controller module. RAT specific features are controlled by the RAT specific modules.

The management plane is common to all RAT and deals with device configuration and error detection. Each separate network device has its own YANG model, which is then configured by the management plane, via the NETCONF protocol.

The Controller will periodically collect statistics from the network. Based on this information, specific RAT module application will get triggered, which will then perform the required function. The RAT module then communicates with the network node, via the southbound interface of the controller middlware.

Time critical functions, such as scheduling will still be carried out at the network node. However, following the SDN paradigm of data plane and control plane separation, all other deferrable functions shall be handled remotely, at the controller.

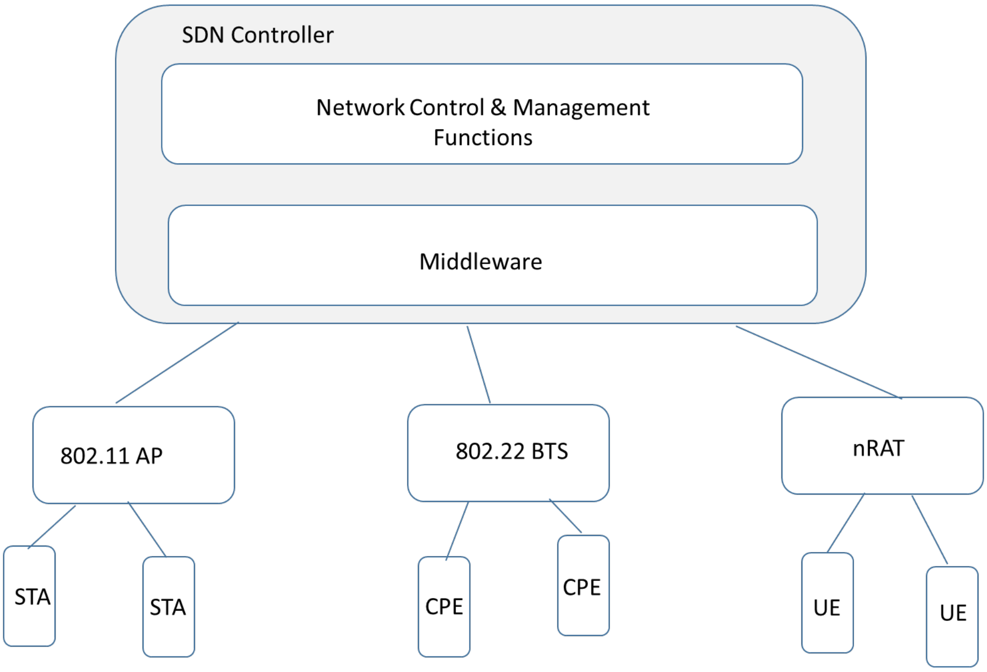


Figure 2.1: SDN based Multi-RAT wireless controller network

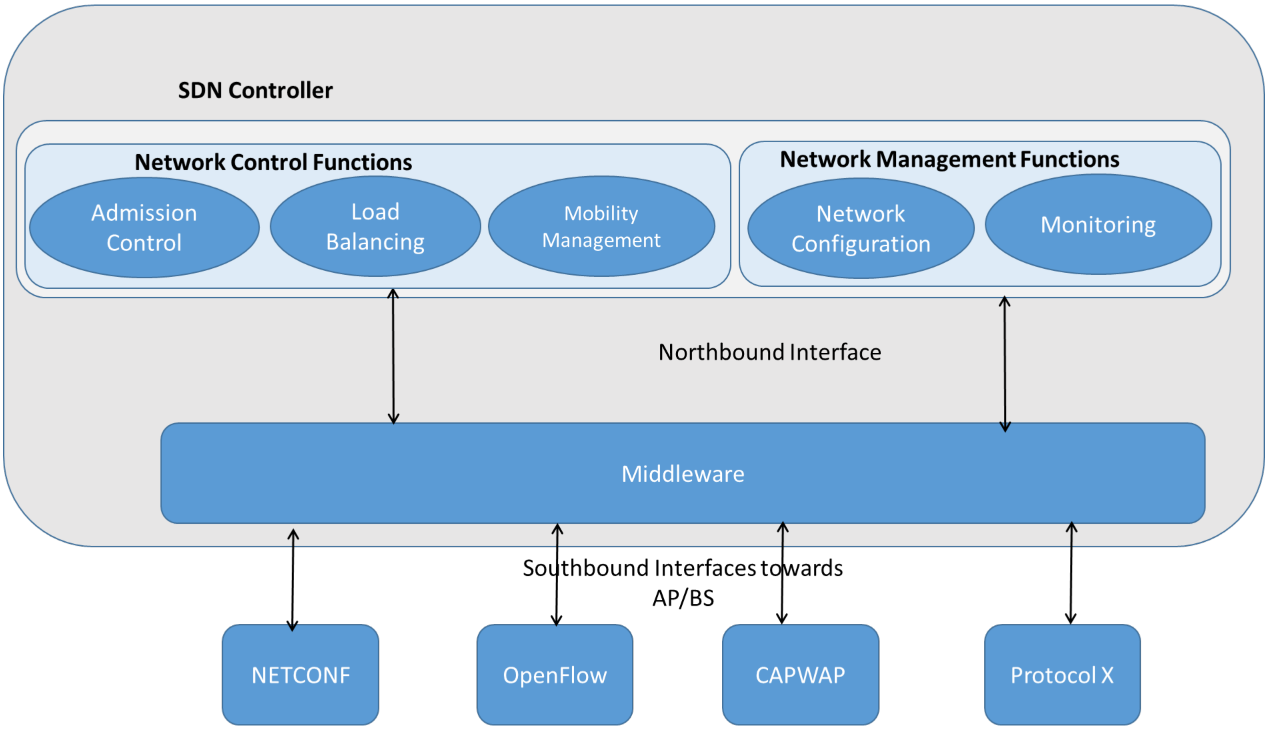


Figure 2.2: Middleware Architecture

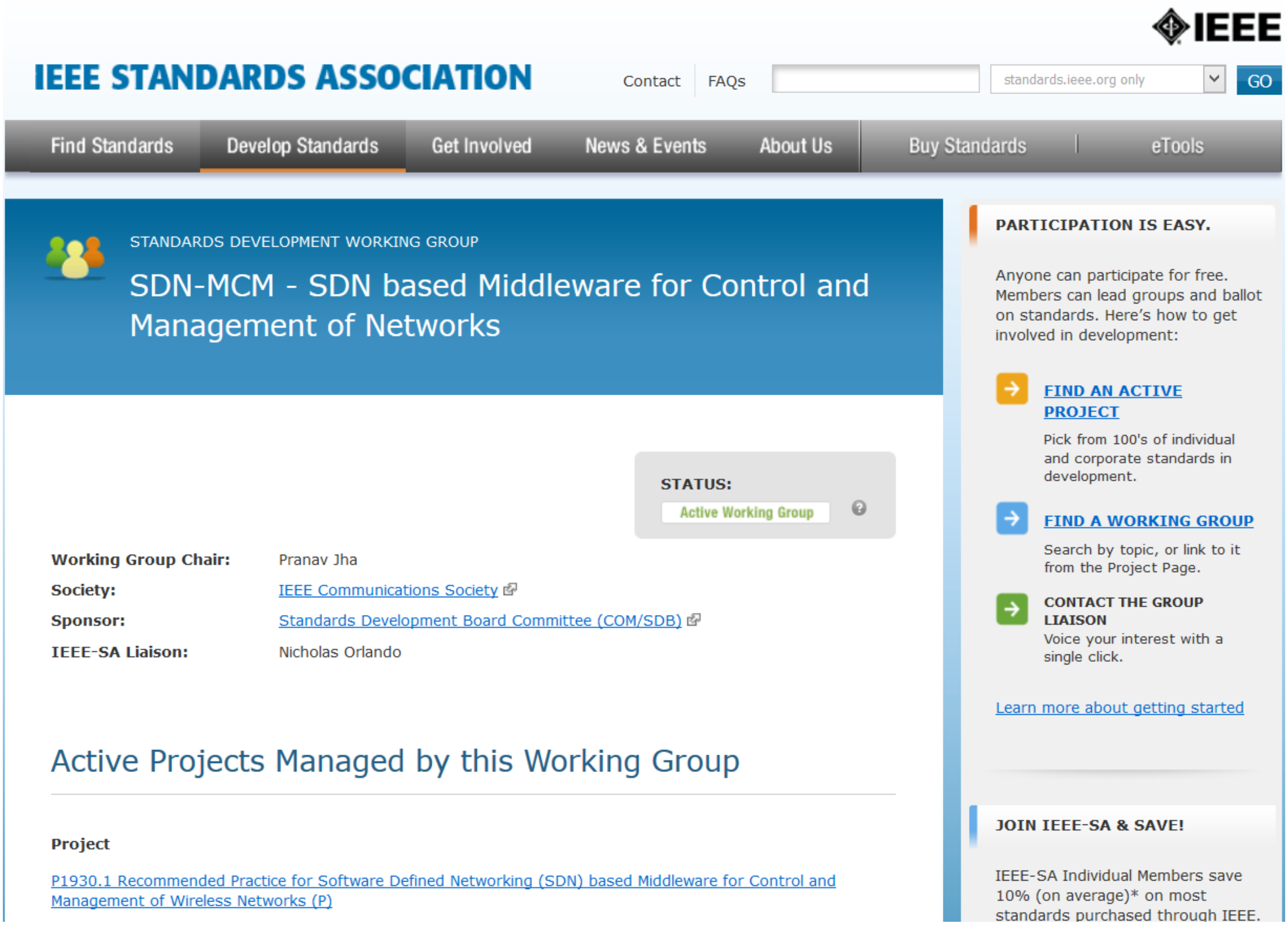
2.1 Middleware Description

The Middleware acts like an aggregator and consists of an Abstraction Layer to expose the capabilities of the underlying RATs. The Abstraction Layer has a Northbound interface towards network applications and a Southbound Interface towards the Control and Management protocols. It also consist of an East-West interface for inter-controller communication.

At the time of writing this thesis, P1930.1 (Recommended Practice for SDN based Middleware for Control and Management of Wireless Networks) is being developed by IEEE Work Group 1930, in which we are participating. We intend to standardise the north-bound interface, between the middleware and the Network Control and Management Applications. This would allow third-party vendors to develop applications independently.

At this stage, we believe that it would not be wise to standardize the southbound interface, as several vendors are already using their own custom control protocols (which may or may not be SDN based). A standardization of the southbound interface would require modifications in currently manufactured controllable access points. Such radical change may not be acceptable. Also, standardization activities are already taking place for a southbound interface in WG 802.11. Hence it was felt that it would be prudent to harmonize P1930.1's current activities with those of WG 802.11.

The middlware could have been placed on the Access Point as well. However, we believe that it would be better to plcae the middleware on the controller because it provides us flexibility to use existing Southbound protocols such as Openflow, NETCONF, TR-069 etc.



STANDARDS DEVELOPMENT WORKING GROUP

SDN-MCM - SDN based Middleware for Control and Management of Networks

STATUS: Active Working Group

Working Group Chair:	Pranav Jha
Society:	IEEE Communications Society
Sponsor:	Standards Development Board Committee (COM/SDB)
IEEE-SA Liaison:	Nicholas Orlando

Active Projects Managed by this Working Group

Project

P1930.1 Recommended Practice for Software Defined Networking (SDN) based Middleware for Control and Management of Wireless Networks (P)

PARTICIPATION IS EASY.

Anyone can participate for free. Members can lead groups and ballot on standards. Here's how to get involved in development:

FIND AN ACTIVE PROJECT

Pick from 100's of individual and corporate standards in development.

FIND A WORKING GROUP

Search by topic, or link to it from the Project Page.

CONTACT THE GROUP LIAISON

Voice your interest with a single click.

Learn more about getting started

JOIN IEEE-SA & SAVE!

IEEE-SA Individual Members save 10% (on average)* on most standards purchased through IEEE.

Figure 2.3: IEEE P1930.1's homepage

Chapter 3

Control of WLAN Networks

The Control Plane is responsible for the configuration of the forwarding plane. Its tasks include flow management for routing, admission and mobility control and authentication of devices

The control plane requires low latency, due to the nature of its functions, else it may act as the bottleneck of the network.

3.1 SDN based Admission Control in WLAN Networks

A need for admission control in WLAN networks was felt, for effective utilization of network resources. In cellular networks, when a base station is heavily loaded, and there exists another lightly loaded base station whose coverage area spans the wireless station (STA), the STA is instructed to connect to the lightly loaded base station. No such mechanism is available in WLAN networks - the decision of which AP to connect to lies solely with the STA.

The default behaviour of STAs is to connect to the AP having maximum signal strength. The AP is not able to provide any information to the STA, regarding its current load. This shortcoming would prove especially problamatic in dense WLAN deployments, where the STA may lie in the coverage area of several AP.

To address this issue, of Admission Control has been implemented - on reaching a threshold of connected STA, the AP will be instructed by the central controller to stop allowing connections from additional STA. On being denied a connection, the STA will then attempt to connect to a different AP. This will in effect allowing the Controller to decide which AP the STA must connect to.

As per the 802.11(2016) standard, when a network device is to join a WLAN network, the following steps need to be carried out:

- AP Discovery
- Authentication
- Association
- 802.1x Authentication

3.1.1 AP Discovery

There are two methods, through which the WLAN STA may select a AP, through which it is to connect to the WLAN: Active scanning and Passive scanning.

In **Passive Scanning**, the STA listens for the beacon frame, sent by the AP at regular intervals. Based on the information provided by the beacon frame, the STA selects an appropriate AP to connect to.

In **Active Scanning**, the STA broadcasts a Probe Request on a particular frequency channel, and listens for a probe reponse from APs that are broadcasting at that particular frequency.

Admission control through a centralized controlling entity is not possible at the stage of AP Discovery. The beacon frame is broadcasted - it is not sent to only a select class of STA. Any STA in the vicinity of the AP can receive the beacon frame.

The STA waits for a probe response, at a particular frequency, for only 10ms[6], after which it hops to the next frequency band and sends a new probe request. Hence, if admission control were to be done in the AP Discovery phase, the Probe request must be forwarded by the AP, to the central controller *and* the controller must send back a response within 10ms. The latency requirements of this step prevent implementation of effective admission control.

3.1.2 Authentication

802.11 Authentication is an artifact left behind from the times before a searate authentication server was used for authentication (as is the case with 802.1x authentication). By default, Open Authentication is performed - the client sends an Authentication Request, to which the AP replies with an Authentication Response - to accept the STA.

The STA always expects the AP to accept its Authentication Request. Hence, it was felt inappropriate to include Admission Control at this step.

3.1.3 Association

On determining which AP it wishes to connect to, the STA sends an Association Request. In traditional 802.11 networks, the AP must then send back an Association response. To implement Admission Control, it has been decided to send the Association request to the central controller, which will generate a reponse based on policy. The response will then be communicated to the STA, via the AP.

3.1.4 802.1x Authentication

This is the actual authentication step, implemented in modern WLAN networks. However, 802.1x authentication is a time consuming procedure, hence it was decided not to implement admission control at the authentication step.

3.2 Quality of Service (QoS) in WLAN

In wired networks, QoS can be enforced via VLAN tagging. Devices are assigned to a Virtual LAN (VLAN), based on the port to which they are connected, or based on the MAC address of the device. Based on the VLAN the STA are assigned to, packets originating from the STAs get tagged. Based on these VLAN tags, traffic is sent to corresponding queues on network switches. Thus, packets are transmitted by the switches based on their priority level.

We have developed a simple way to enforce QoS in WLAN to provide graded service to connected users, as per operator policy, based on VLAN tagging.

3.2.1 Dynamic Virtual Local Area Network (VLAN) assignment

802.1x Authentication is a method for per-user or per-device based authentication. The Extensible Authentication Protocol - Protected Extensible Authentication Protocol (EAP-PEAP) method has been used for user-based authentication, in which a centralized RADIUS server authenticates the users.

The RADIUS server acts as an Authentication, Authorization, and Accounting (AAA) server, granting the STA access into the network. Owing to the fact that PEAP is a username-password based authentication system, STA can be divided into classes, based on their credentials. Based on the credentials provided by the STA, the radius server indicates to the AP, which VLAN the STA must join. The AP tags traffic originating from STA with its VLAN. All STA belonging to the same VLAN connect to the same interface on the AP.This was done by creating sub-interfaces for each VLAN, within the wireless interface.

By placing different users into different VLAN, QoS can be implemented at the edge routers. As in the case of the wired network, packets with different VLAN tags are sorted into different hardware queues on the edge routers. This allows the routers to give bandwidth and delay guarantees.

3.3 WLAN Control in the context of Multi-RAT architecture

In a multi-RAT wireless network architecture, there will be two additional nodes, not normally present in a standard 802.11 based network:

- The Management Gateway(MAGW) : The MGW acts as a mobility anchor, saving context required for LTE-Wifi handover.

- The Packet Network Gateway (PNG) : This gateway serves as an entry point for the packet data network to which the wireless station is connected.

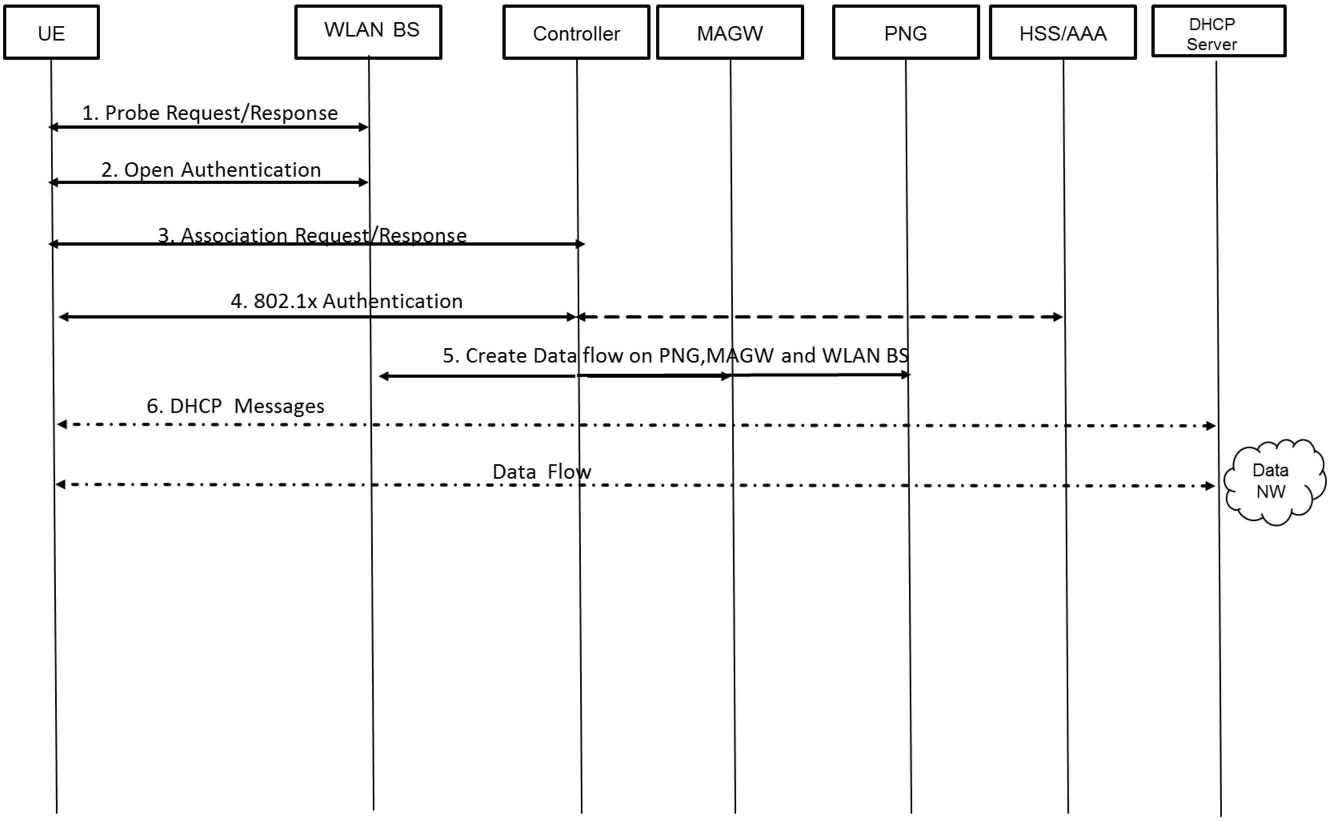


Figure 3.1: Call flow for Admission to a WLAN Network, controlled by the multi-RAT controller

Chapter 4

Management of WLAN Networks

The Management Plane deals with functions such as Configuration of Network devices, Network Monitoring and Error Detection.
Unlike the Control Plane, there are no strict latency requirements of the Management Plane.
There are several protocols already developed which may be used to configure Network devices, such as Simple Network Management Protocol (SNMP), Control and Provision of Wireless Acces Points(CAPWAP), TR-069 and NETCONF.
NETCONF has been proposed to be used as the protocol to configure parameters of Network Devices.

4.1 WLAN Management Protocols and their Drawbacks

4.1.1 SNMP

SNMP is a legacy protocol, running over UDP. The protocol describes two types of devices:
The Manager sends UDP GET/SET requests, to gather information about devices in the network.
The Agents first create a database on each network device called the Management Information Base (MIB). They are then responsible for storing device information in these databases. They send UDP GET/SET responses

In addition, asynchronous TRAP messages may be sent by the agents to the controller. However, inability of SNMP to scale[7] meant it was ill-advised to continue with SNMP. The absence of a reliable transport mechanism in SNMP means a much larger number of transactions are required to configure a particular parameter. Also, there is no means of configuration validation.

4.1.2 CAPWAP

CAPWAP is a viable option, which may be used to manage WLAN networks. However, there are no means available to configure multiple device parameters concurrently in CAPWAP, making it less efficient in comparision. Several options in CAPWAP are left open to the implementor, which may lead to interoperability issues. Additionally, there are IPR issues with CAPWAP, which may proove to be an obstacle. CAPWAP calls the wireless controller an Access Controller (AC).
There are two types of messages sent by the AC- the CAPWAP Data messages and the CAPWAP control messages. The CAPWAP control messages are used by the AC to manage the AP. These messages include

1. The “Join” message, sent when the AP wishes to connect to the AC

2. The AP Configuration Management message, used by the AC to configure the AP (although the AP may store its configuration locally)

Modes of operation

1. Split MAC

   Layer 2 data and management frames are forwarded to the AC from the AP. All time sensitive tasks are still done by the AP (like Beacon generation, Probe Response, Power Management). However, Distribution and Integration services are handled by the AC. A drawback of this approach is that data needs to be tunneled to the AC before being sent to the external network.

   Data packets are encapsulated as 802.3 frames and sent to the AC as CAPWAP Data messages(over a separate UDP port). This may not be efficient (or scalable) when a large number of AP need to be controlled by a single AC.

2. Local MAC
   Integration services exist on the AP itself, while distribution services exist on the AC or the AP. All MAC functions terminate on the AP, giving rise to the term "fat AP" for those APs that are controlled by CAPWAP with local MAC. The AC is still informed about MAC events, the 802.11 messages are forwarded to the controller. If distribution and Integration services may be handled by the AP itself, tunneling of data is not required.

Association/Reassociation and Dissociation

In CAPWAP, association requests by STA to AP must be forwarded to the AC. This gives the AC an idea of the general location of all users in the network, which in turn can help it in its balancing tasks. The AC can also force the AP to send a dissociation message to the STA.

Low loaded AP may be asked to increase their tx power. Since the STA first tries to connect to the AP with greatest tx power, this method allows more STA to be connected to the lightly loaded AP, bringing about Load Balancing.

4.1.3 TR-069

TR-069[8] is a CPE WAN Management Protocol, for secure auto-configuration of Consumer Premises Equipment (CPE). It defines the protocol, message structure, session rules and various RPCs required.

It is based on Hyper Text Transfer Protocol (HTTP) posts and responses. Inside the HTTP post/response are SOAP envelopes, which contain SOAP requests and responses. A SOAP message is an XML document consisting of the Envelope, Header, Body and Fault (errors, if any).

Envelopes are numbered between successive HTTP posts and responses, and have to be responded to in the same order as they come.

The connection is initiated by the CPE, via the Inform request. The CPE and the ACS then trade requests(and responses).Among these requests by the ACS is the Set Parameter Values Request, by which various parameters of the CPE can be adjusted. Connection is terminated by the CPE when all responses have been replied to and when there are no remaining requests.

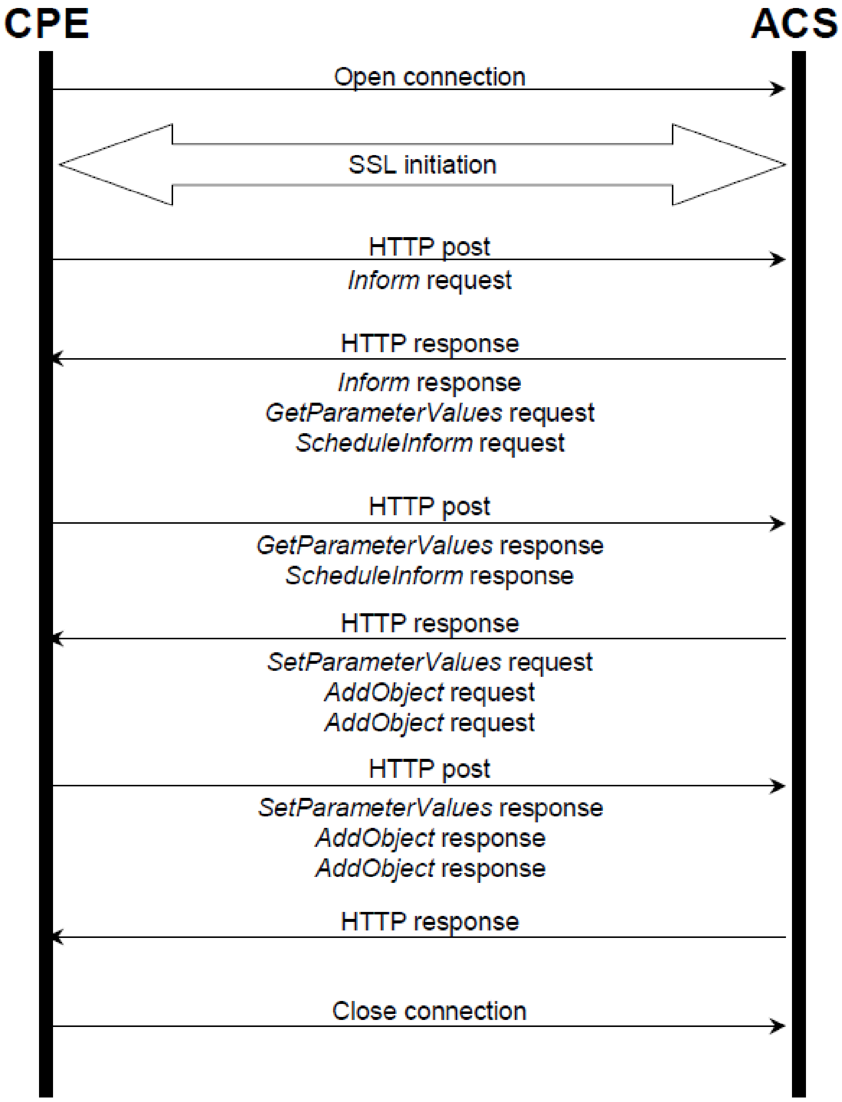


Figure 4.1: Example of a transaction between the ACS and AP in TR-069

The AC can also simulate initiation of a connection by sending a Connection Request Notification.

The average time required for TR-069 transactions is the most. It is sensitive to delays and packet losses.[9]

4.2 Management of Network devices using NETCONF

The NETCONF protocol provides a scalable means to configure network devices. The NETCONF protocol is composed of the NETCONF manager (the TCP client located on the controller) and NETCONF servers (devices). NETCONF uses an RPC based mechanism for communication between the client and the server.

All NETCONF messages are encoded in XML. This enables messages to be sent in any order - the application need not rely on the specific location of the data it requires to extract. It only needs to read the associated data label. All NETCONF messages are encrypted by SSH, allowing for server-client authentication, before any data is sent. It uses TCP as the transport mechanism, allowing for a packet transmission guarantees and more efficient use of the medium.

When the manager and server initially connect, they exchange the <*hello*> message, to declare their capabilities. This enables the manager to know which RPC operations it may submit to the devices.This is in turn determined by the YANG data models loaded on the devices.

A data model is a precise and explicitly defined structure of data, how it is stored and the relations between various data elements. YANG was a data modelling language defined specifically for NETCONF. It allows for easy validation of configuration requests and is human-readable. In addition, YANG has reusable types and groupings of data, which allows for easy creation of new data models.

Since we have used NETCONF to manage WLAN APs, we first created a YANG model for an AP. Please refer to Appendix A, for a detailed description of the data model created by us.

Configuration Management

The NETCONF manager sends a request, enclosed in an RPC element. Requests could be of the type <*get*>*,* <*get-config*>*,* <*edit-config*>*or* <*close-session*>*.*

When the NETCONF server receives this request, an appropriate callback function is executed. This callback function then handles the operation requested by the manager. In response to the request from the manager, the server must also send a RPC reply message - the <*ok*> element is sent on successful configuration,

For example, the request could be to change the transmission power of the AP, which could be implemented via the <*edit-config*>request. On running the appropriate callback function, the AP would then send an <*ok*> response to the Management application.

Implementation of callback functions may be different for AP deployed by different vendors. A generic data model has been defined for all AP. Different types of AP can thus be simultaneously managed by a single management application.

Error Detection

If a rpc request fails, the rpc reply element contains an <*rpc-error*> element. On encountering an error, the NETCONF server is made to rollback to the previous error-free state. On rollback, none of the configurations in the transaction hold. This makes it easy for the manager to keep track of the state of the servers, and reduces the number of status messages that are needed to be exchanged, since only one status message is required per transaction.

Chapter 5

Implementation of the Controller

A small scale, in-lab test bed was set up, to verify that the SDN-based controller architecture works correctly.

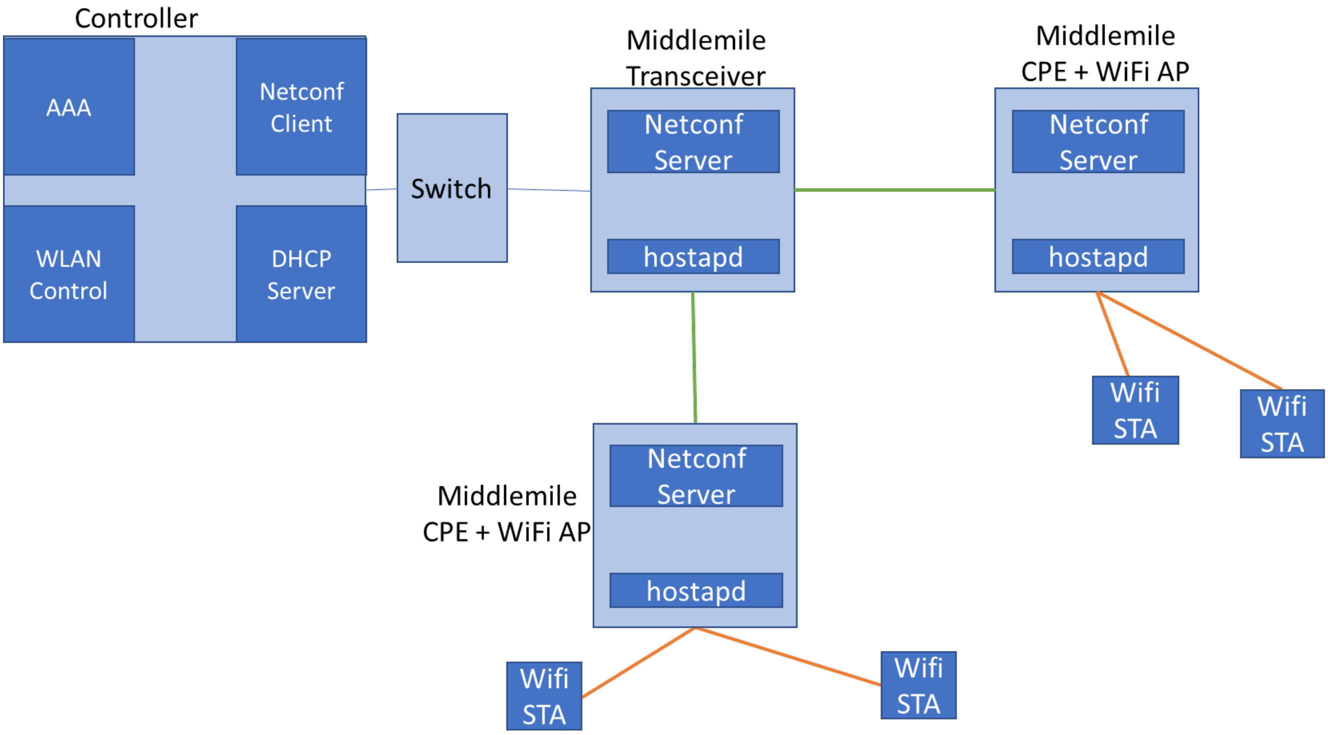


Figure 5.1: Box diagram of implementation setup

We used 3 Mikrotik RB433AH boards with OpenWrt's Chaos Calmer (15.05) installed. The WLAN Controller and an authentication, authorization, and accounting (AAA) server were set up on a Linux desktop. A wired router (Asus RT-N1 3U) was used, to

connect the controller with the Mikrotik boards.The router also acts as a DHCP server. We've made an in-lab test bed demonstrating how the network providing wireless coverage to Palghar and its surrounding villages would be deployed and controlled by our WLAN Controller. All boards have hostapd installed on them, so that they may function as wireless nodes. A Doodle Labs wireless card was installed in one slot of the MikroTik board. The card transmits and receives data in the 530 Mhz band. A middlemile link was created between one board, which acted like a middlemile base station and the other two boards, which acted like the middlemile Receivers.
In the other slot of the boards acting as middlemile receivers, a normal 802.11 g card was installed. The board bridged data from the middlemile link to the Wireless link and back.

Down-Shifting Frequency of the Doodle Card Transceiver

Since IIT-B has received permission to use the 510 Mhz band for middlemile data transmission, first the frequency of transmission of the doodle board needed to down-shifted by 20 Mhz By default, the frequency at which the doodle board transmits is given by
f = 530 + (channel number)*5 Mhz,
where 530 is the central frequency of transmission and the channel number can be set to an integer between 1 and 12.
By setting the central frequency to 510 Mhz, and creating a new regulatory domain, in which such configuration is permissable, we have forced the board to transmit in the 510 Mhz band.

802.1x authentication

A WLAN where 802.1x authentication is carried out comprises of three components:

- The supplicant: the STA which wishes to gain access to the WLAN
- The access device: the AP through which the STA will send its traffic to the WLAN
- The AAA server: This device handles requests for access to the network

When a device connects to a network, it enters the unauthorized state. Initially all data traffic is blocked by this AAA server. The only traffic that is allowed to be sent by the device is authentication data to the server.

The supplicant sends an EAP-start message. The access device sends an EAP-request identity message. The supplicant's EAP-response packet with the supplicant's identity is "proxied" to the authentication server by the access device.

Since we are using EAP-PEAP, mutual authentication between the supplicant and the server is done. The authentication server send its credentials to prove itself to the supplicant and asks the supplicant to authenticate itself. The supplicant checks the server's credentials and then sends its credentials to the server. The AAA server has a user database, in which all user credentials are saved. In addition, information about which VLAN the supplicant must join is stored. Based on these credentials, the supplicant is either accepted or rejected.

If the supplicant is accepted, the access device changes the virtual port with the supplicant to an authorized state allowing full network access to that end user. If the supplicant is the first device belonging to a particular VLAN class, the server directs the access device to create a sub-interface. All subsequent supplicants belonging to that particular VLAN, connect to the access device through that sub-interface. Finally, all traffic entering that interface is tagged with the appropriate VLAN, enabling QoS enforcement.

The AAA server we have implemented on the Linux desktop uses FreeRADIUS, a free open source RADIUS server. (RADIUS is the networking protocol through which 802.1x is implemented)

The WLAN Controller

There are three components of the WLAN Controller -

- The wired OpenFlow Controller: This module of the controller was implemented using Open Daylight (ODL), a freely available SDN Controller. Proactive flows were set by the OpenFlow Controller. Traffic entering the board from the sub-interfaces of wlan0 is forwarded to the WAN port of the board.

- The Netconf module: Netopeer[10] has been used to implement the Netconf manager on the controller and the corresponding server on wireless nodes. When a NETCONF message is received by the netopeer server on the AP, a Unified

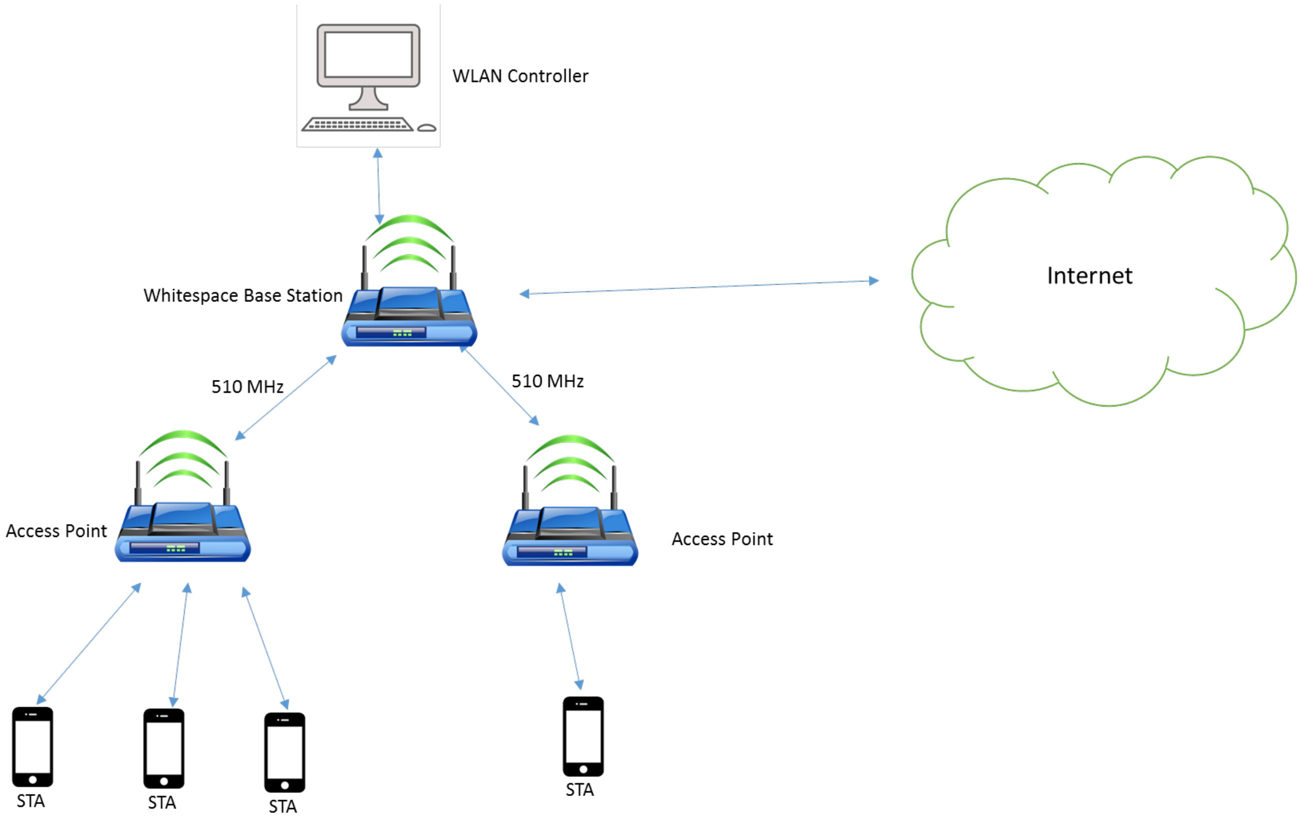


Figure 5.2: Implementation Setup

Configuration Interface (UCI) command is called to perform the requested function. UCI commands allow the controller to modify the config files on the OpenWrt boards. Any alternate interface may also be used to configure non-OpenWrt boards.

- The northbound applications: Admission Control was implemented for load balancing.

When the STA sends an Association request to a AP, the Association request is forwarded to the WLAN controller, in the form of a TCP message. A server running on the Controller reads this message.

In the current implementation, a list of clients is stored at the WLAN controller, based on which admission control decisions are made. The controller sends back a TCP message to the AP, indicating whether or not to accept the STA

We used a YANG model to configure the AP's SSID, transmission power, channel and to enable dynamic VLAN.

Chapter 6

ns-3 simulations

6.1 Introduction

In the current literature, studies have been done, describing the qualitative benefits of using an SDN based controller, over a Non-SDN based controller. Through simulations in network simulator-3 (ns-3)[11], we have arrived at a measure of the gain attained, if the SDN principles are used to design the wireless network.

Consider two different types of WLAN controllers - an SDN based WLAN controller and a Non-SDN based WLAN controller. In an SDN based WLAN controller topology, since there is a separation of data traffic and control traffic, data from various network switches gets routed directly to the external gateway.

However, in a Non-SDN based WLAN controller topology, all data must be routed through the controller.

6.2 Network Topology Description

The 5G project at IIT-B has been studying how to use an SDN based controller, to control access points deployed in villages, which are connected to the fibre POP via a middlemile. Hence, our topology models the same real-life network.

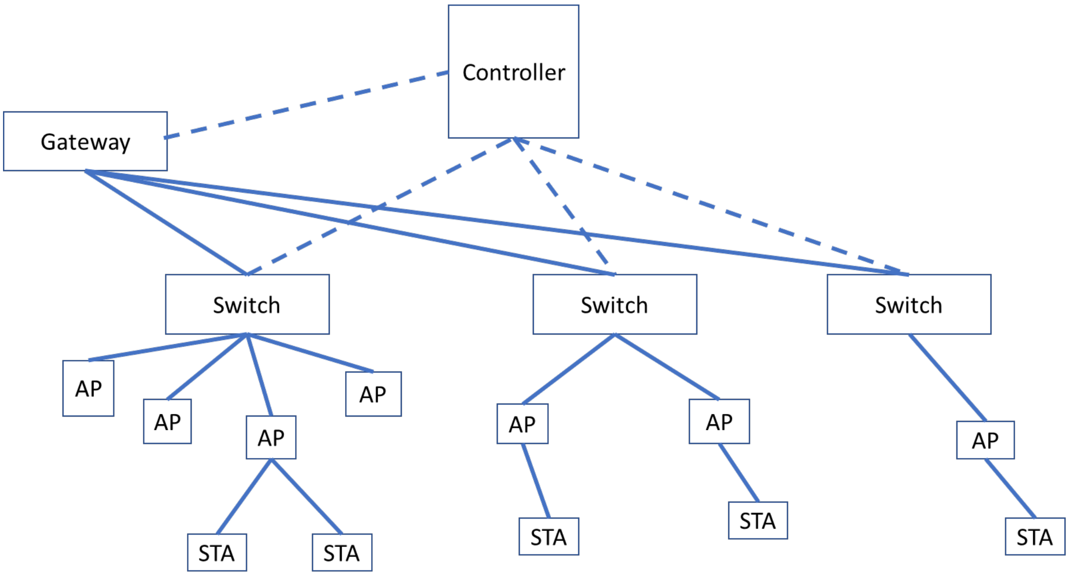


Figure 6.1: An SDN based WLAN controller

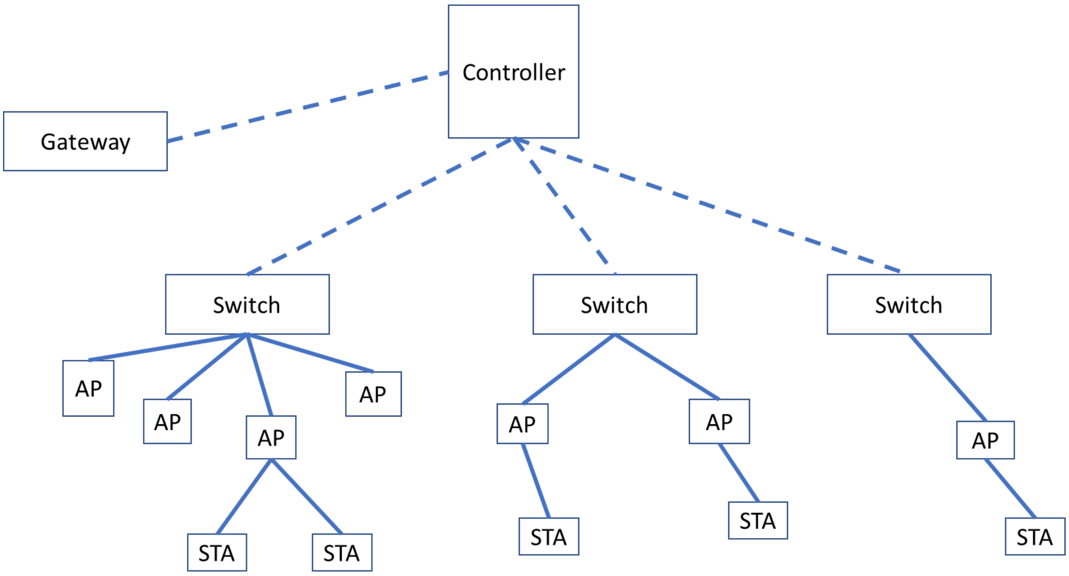


Figure 6.2: A Non-SDN based WLAN controller

6.2.1 Traffic routing in ns-3

There are two ways through which traffic can be routed in ns-3: global routing and static routing.
In global routing, all optimal routes in the network are determined. The routing tables of all nodes are set during network configuration (before the actual simulation starts). Although this feature is convenient to use, it does not scale well, beyond a few hundred network nodes. Instead, we have used static routing, which dramatically reduces the time ns-3 takes to configure the network.
In static routing, explicit routing paths are given. To further save on time, we have only set up those routing paths, which we know will be used. For example, since we are not simulating station to station communication, we have installed no routing rules, corresponding to such paths.

6.2.2 Link Capacity

In test case 1-3 each base station - access point has a link with a capacity of 40 Mbps. The Government of India has provided 20 MHz of spectrum to be used for the middlemile link. Assuming a spectral efficiency of 2, the link can support a data rate of 40Mbps. Further, at most 4 non-interfering antennas may be installed on the middlemile base station. Hence, in test case 4 each base station - access point has a link with a capacity of 10 Mbps.
Each wireless station is sending data at a constant bit rate of 2 Mbps. All other point to point (P2P) links have a capacity equal to the worst case traffic which could possibly go through them.

6.2.3 Error Models

The Yans error model has been used for wireless links. BER of 10^{-8} (which comes to about 0.44% PER, with packet size of 5KB) for all (wired) links. The Rate Error Model was used to implement this.

6.2.4 Parameters

We have measured the following parameters, in order to compare the performance of the SDN and Non-SDN based controllers:

- **Throughput**: The average throughput experienced by stations connected to the Access Points in the network, measured by counting the number of data packets received by the stations.
- **Set-up Time**: The time from when a wireless station sends an association request to when it receives an association reply and joins the WLAN network

Throughput has been measured using "Flowmon", a tool in ns-3 to measure packet statistics, while Set-up time is measured using ns-3's time object, which schedules various network events.

6.3 Admission Control via Asssociation Request Forwarding

To take advantage of the global view of the network provided by a controller based architecture, admission control has been implemented for 802.11 WLAN networks.
Whenever a station comes in the vicinity of an Access Point(AP), it sends an association request. In regular 802.11 WLAN networks, the AP responds with an association response.
In our controller based WLAN network, we have forwarded the association request to the controller, which then sends back a response to the AP, to determine whether the station may associate. This decision could be taken by the controller in such a way, so as to uniformly distribute the stations across all APs.

6.3.1 Implementation Details

The association request is a MAC layer message. Hence, conventionally, the L2 layer of the AP handles these messages. However, the WLAN controller may not lie in the same L2 broadcast domain. The packet must be routed over an L3 network, and hence must have an IP header.

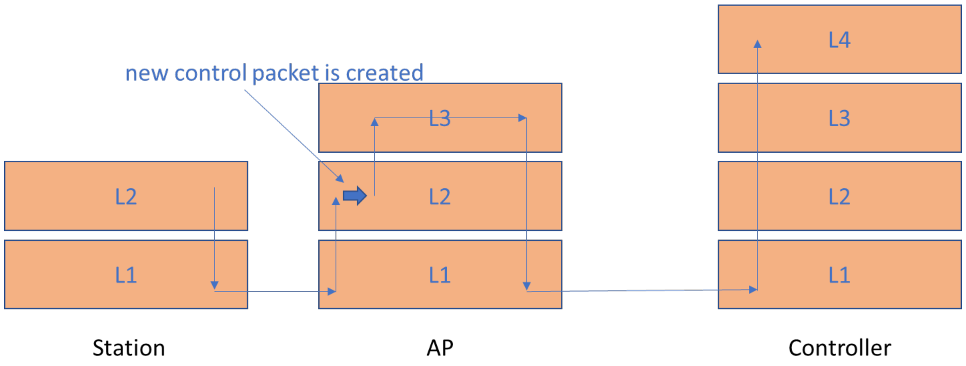


Figure 6.3: Passage of an association message through the OSI stack of network nodes

When the AP receives the association request, the stations MAC address is extracted from the request frame, and added as payload into a new control message. This control message is then encapsulated by a UDP header, followed by an IP header. The packet is then forwarded up by the AP, and the L3 layer of the AP then routes the packet to the controller.

The controller has a UDP application running on it, listening on port 10, which then responds affirmatively to the association request, sending back a UDP packet to the AP. In the future, an algorithm will be running on the controller, to determine whether the station may associate.

6.4 Simulation of Netconf Messages

Netconf messages are sent from the APs to the controller every 3 seconds. Currently, we are only sending one statistic to the controller the number of stations currently associated with the AP. However, other statistics can be easily added, based on the requirements of the algorithm running on the controller.

6.5 Test Cases and Results

6.5.1 Simulation Test Cases

In all test cases, constant bit rate traffic (2Mbps) is generated by stations, destined to the gateway of the network. In test cases 1-3, the AP-middlemile base station capacity has been set to 40 Mbps. In test case 4, this capacity has been reduced to 10 Mbps, to better model the capability of an individual middlemile link, assuming that atmost 4 non-interfering antennas can be placed at the middlemile base station.
In test case 1, stations and APs are positioned on a grid. In test cases 2,3 and 4, the stations were randomly positioned in a circle with a radius of 30m, around various AP. A threshold of 1.5 Mbps (3dB below 2 Mbps) has been set as the acceptable minimum throughput per station. Specifically, gain due to the SDN based controller has been defined as:

$$\%\text{Gain} = \frac{AP_{SDN} - AP_{Non-SDN}}{AP_{Non-SDN}},$$

where AP_{SDN} and $AP_{Non-SDN}$ are respectively the maximum number of APs in the SDN and Non-SDN based WLAN controller based network, beyond which the per-station throughput drops below the threshold.

Case 1

We first considered a lightly loaded test case. We used 802.11 b based APs, with 5 stations per AP. We plotted the variation in throughput as the number of APs per switch were increased. Since there are at most 200 APs, with 5 stations per AP, through any link in the network, atmost 2 Gbps of data may flow (specifically, 2 Gbps of data will flow through the controller-gateway link in the Non-SDN based controller topology)

A gain of 143% was observed in this test case.

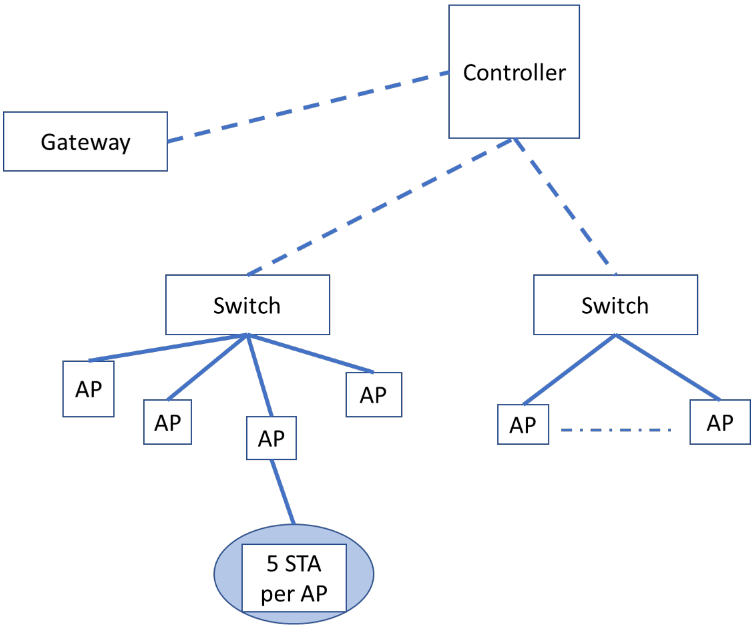


Figure 6.4: Test Case 1 topology

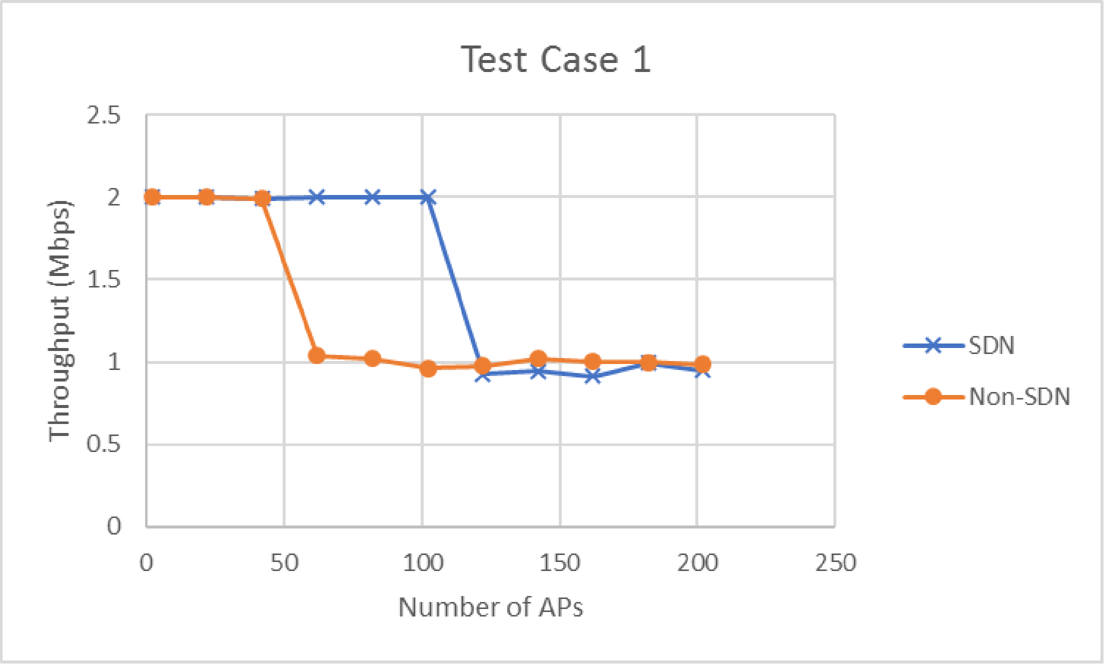


Figure 6.5: Average Station Throughput Comparision, Test Case 1

Case 2

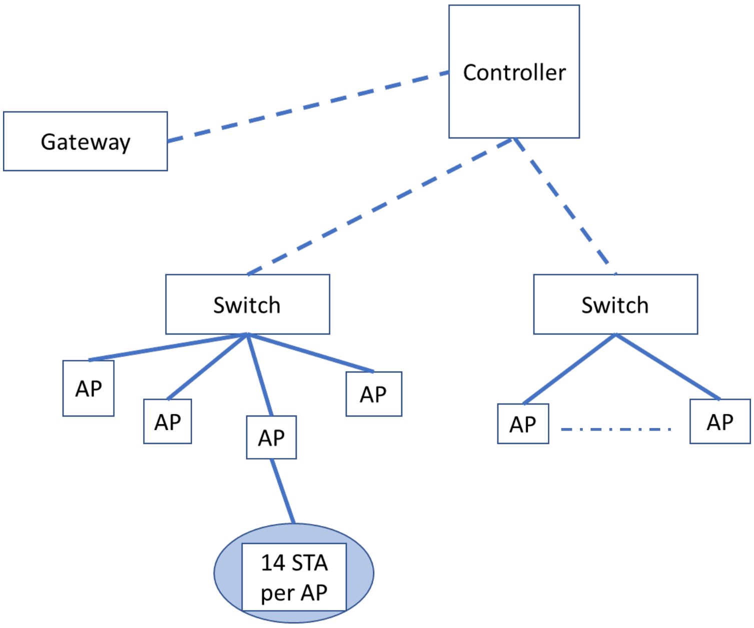


Figure 6.6: Test Case 2 topology

We used 802.11 n based APs, with 14 stations per AP. We plotted the variation in throughput as the number of APs per switch were increased. There are at most 122 APs, with 14 Stations per AP. $122 \cdot 14 \cdot 2$ Mbps = 3416 Mbps . Hence, 4 Gbps is the link capacity provided (allowing for some extra capacity)

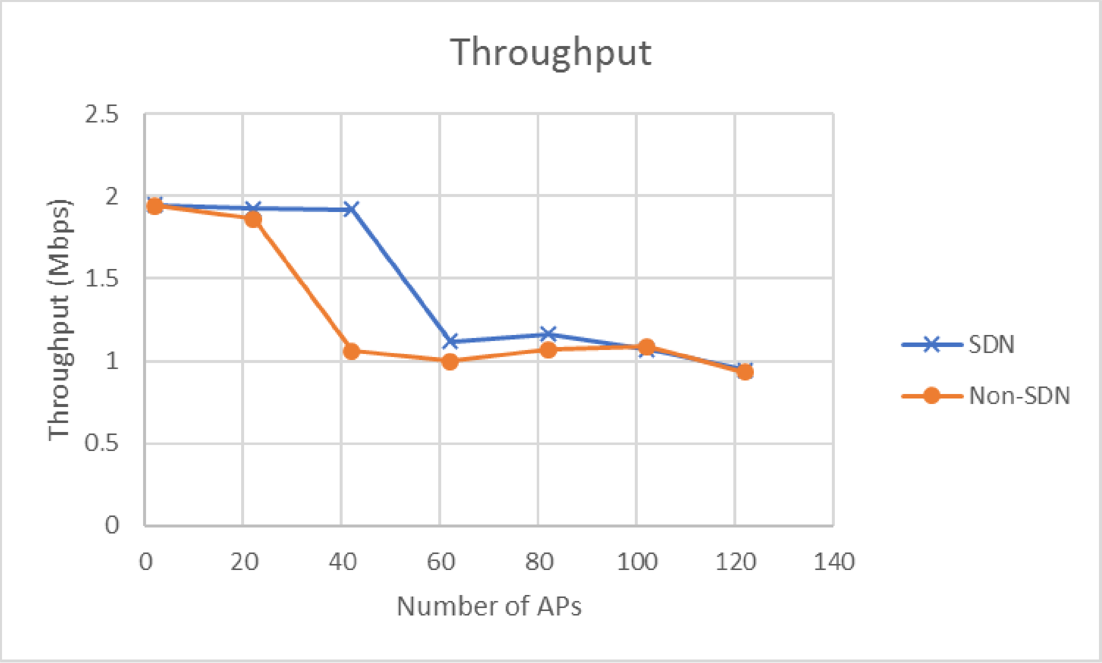


Figure 6.7: Average Station Throughput Comparision, Test Case 2

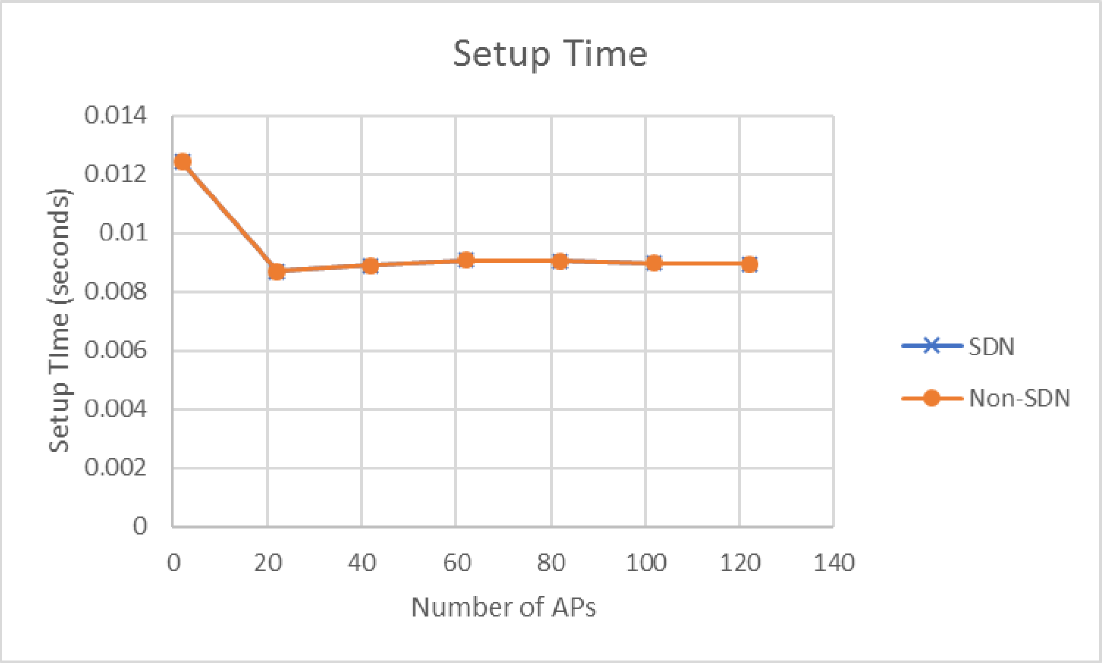


Figure 6.8: Average Setup Time Comparision, Test Case 2

A gain of 90.1% was observed in this test case.

Case 3

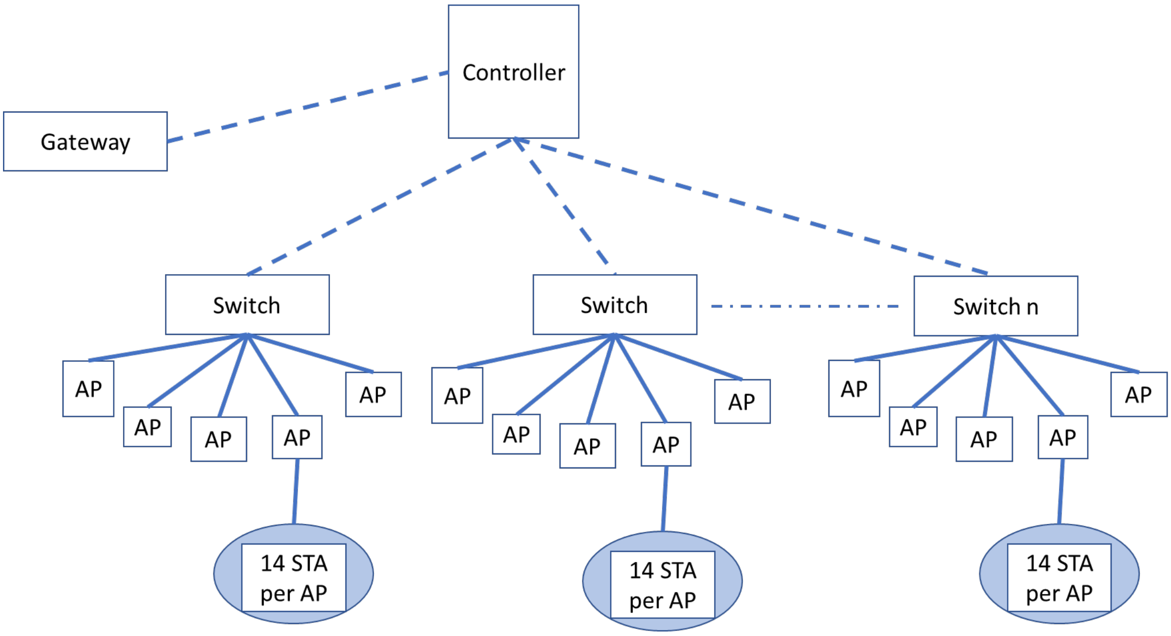


Figure 6.9: Test Case 3 topology

Instead of increasing the number of APs per switch, we increased the number of switches in the network and plotted the variation in throughput. There are at most 185

APs, with 14 Stations per AP. $185 \cdot 14 \cdot 2$ Mbps $= 5180$ Mbps . Hence, each link has a capacity of 6 Gbps.

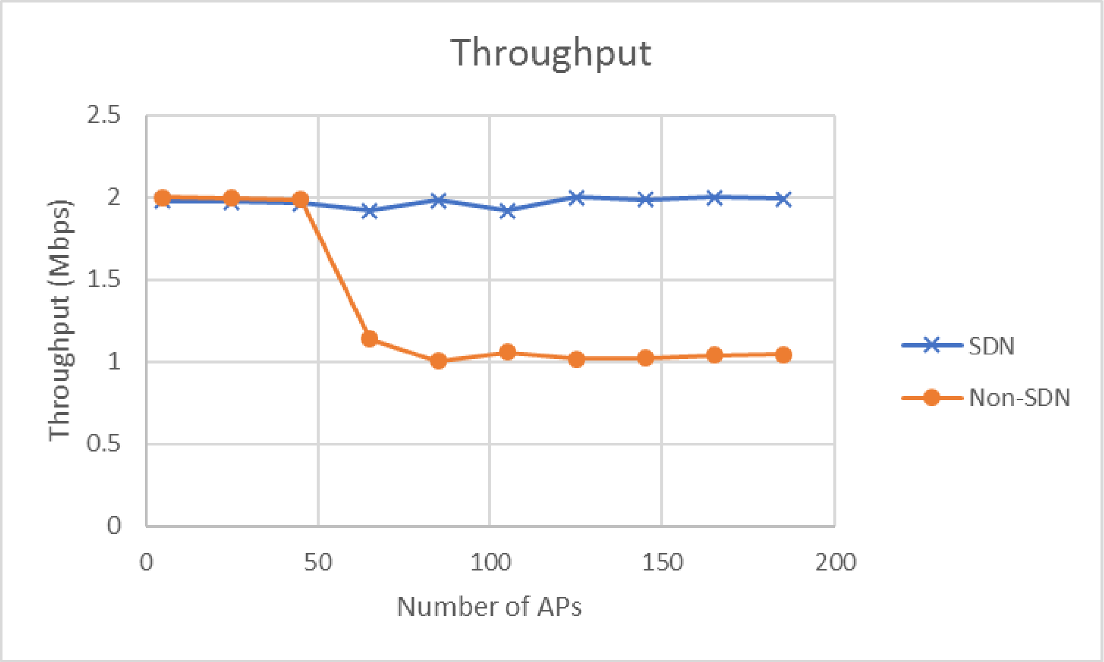


Figure 6.10: Average Station Throughput Comparision, Test Case 3

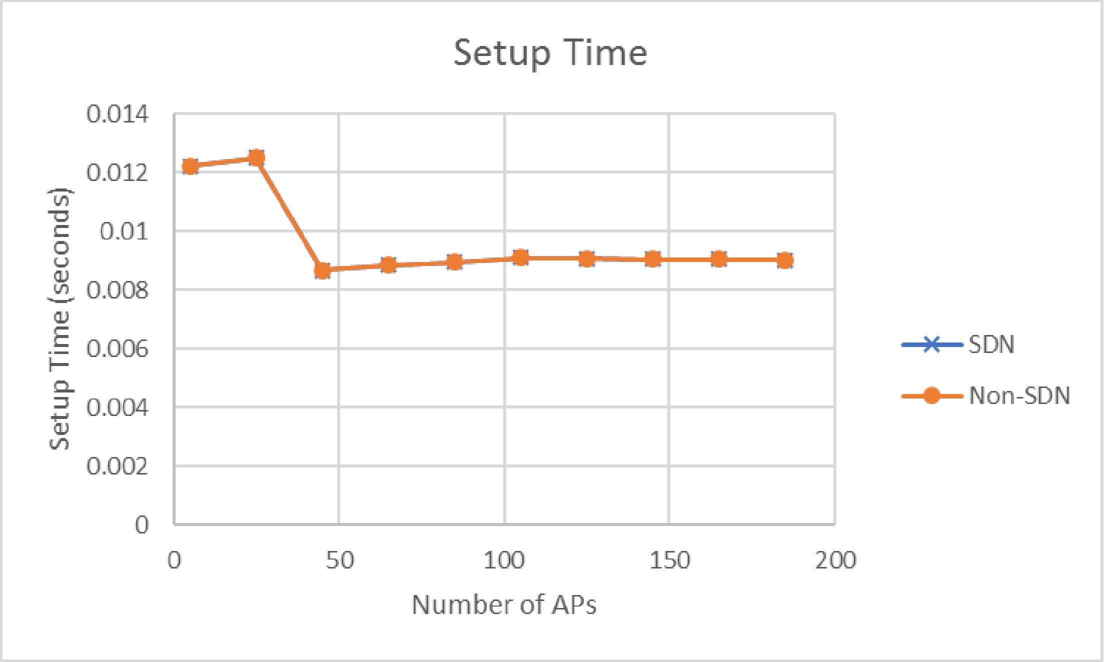


Figure 6.11: Average Setup Time Comparision, Test Case 3

Case 4

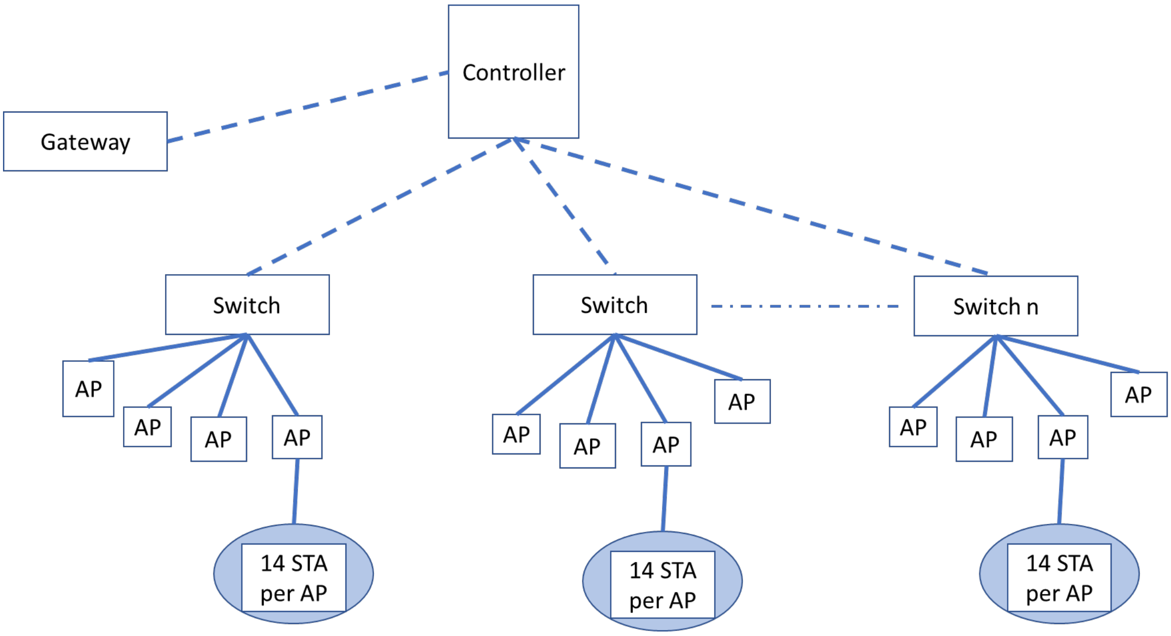


Figure 6.12: Test Case 4 topology

Now we set the number of APs per switch to be 4. The 40 Mbps spectrum is equally divided amongst all AP-middlemile base station pairs, resulting in a per-link capacity of 10 Mbps. There are at most 185 APs, with 14 Stations per AP. $185 \cdot 14 \cdot 2$ Mbps $= 5180$ Mbps . Hence, each link has a capacity of 6 Gbps.

As always, there is no difference in setup time.

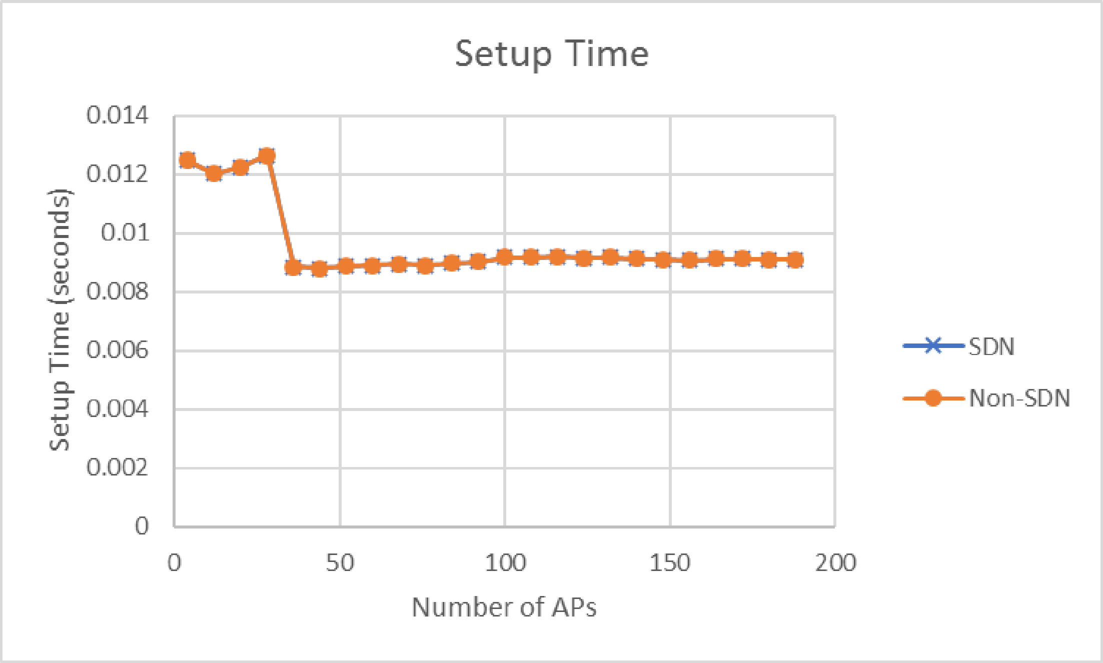


Figure 6.13: Average Setup Time Comparision, Test Case 4

As in case 3, **no drop in throughput** is observed, when the network is controlled by an SDN based controller. However, the same can't be said for the Non-SDN based controller.

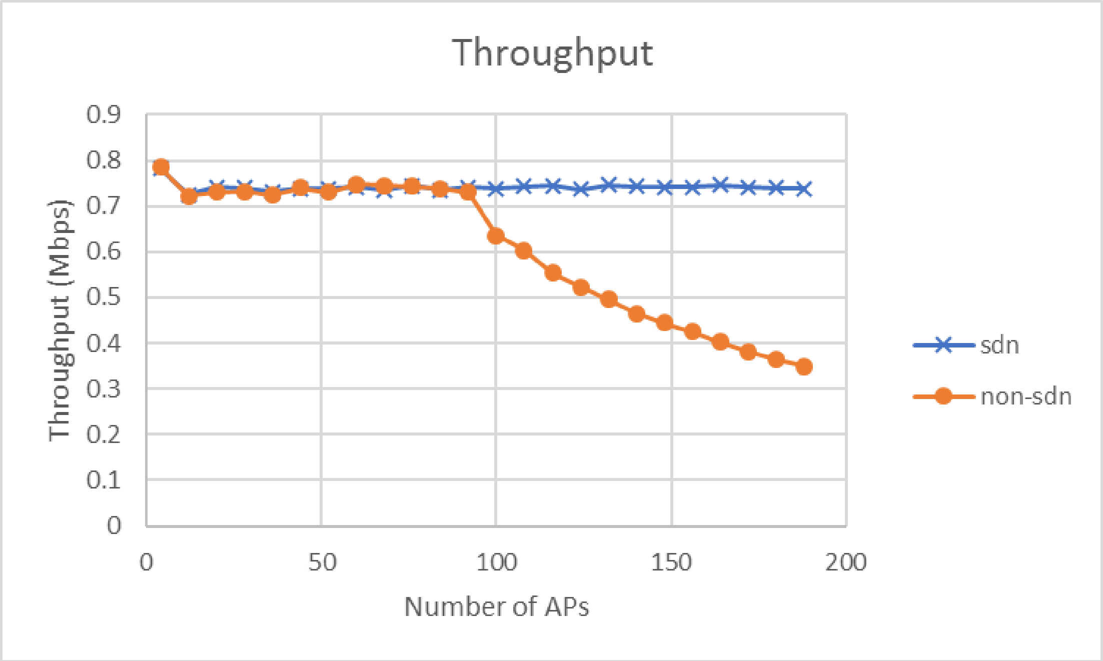


Figure 6.14: Average Station Throughput Comparision, Test Case 4

Note: In case 3, when the Non-SDN based controller was used, there was a bottleneck *in the network between the controller and the gateway.*

In case 4, this bottleneck appears in the AP-middlemile base station link.

Case 5

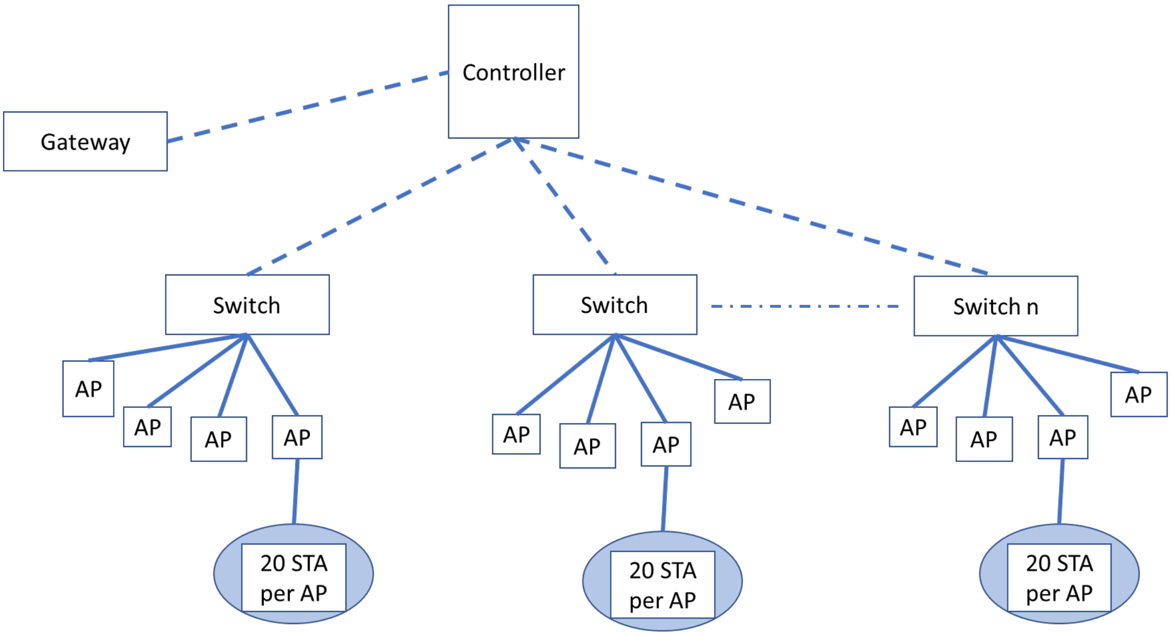


Figure 6.15: Test Case 5 topology

There are now 20 APs per switch, each generating data at a rate of 0.5 Mbps, to ensure that the AP-switch link is no longer the network bottleneck. We have scaled up the simulations in this test case - simulating upto 1000 Aps and a total of 20000 STA

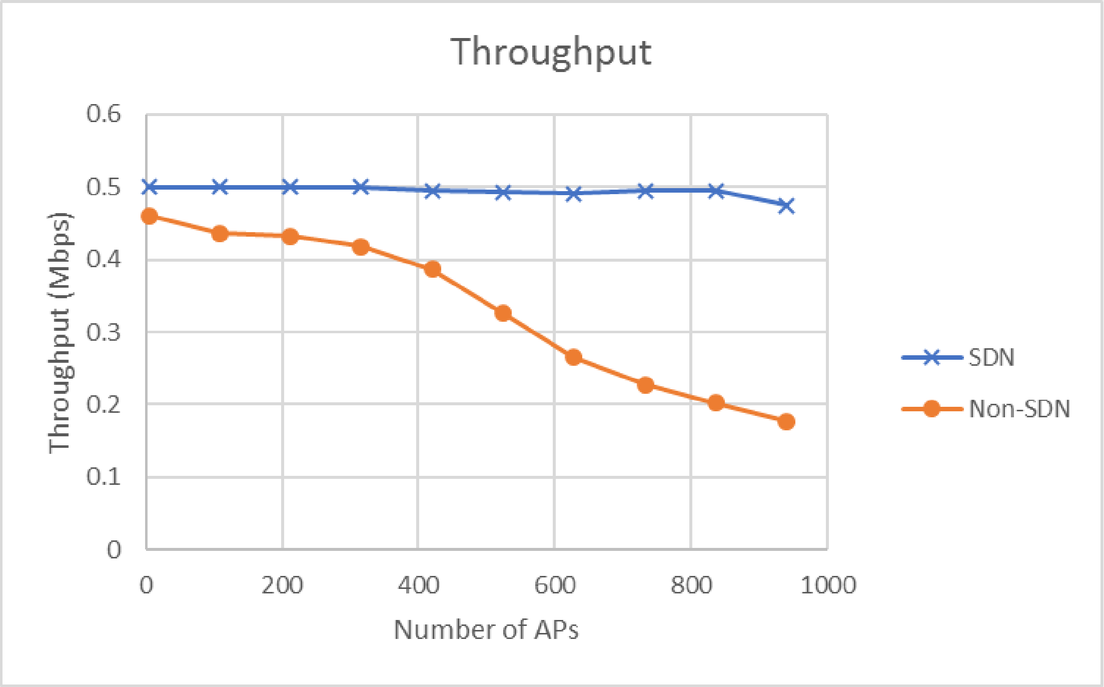


Figure 6.16: Average Station Throughput Comparision, Test Case 5

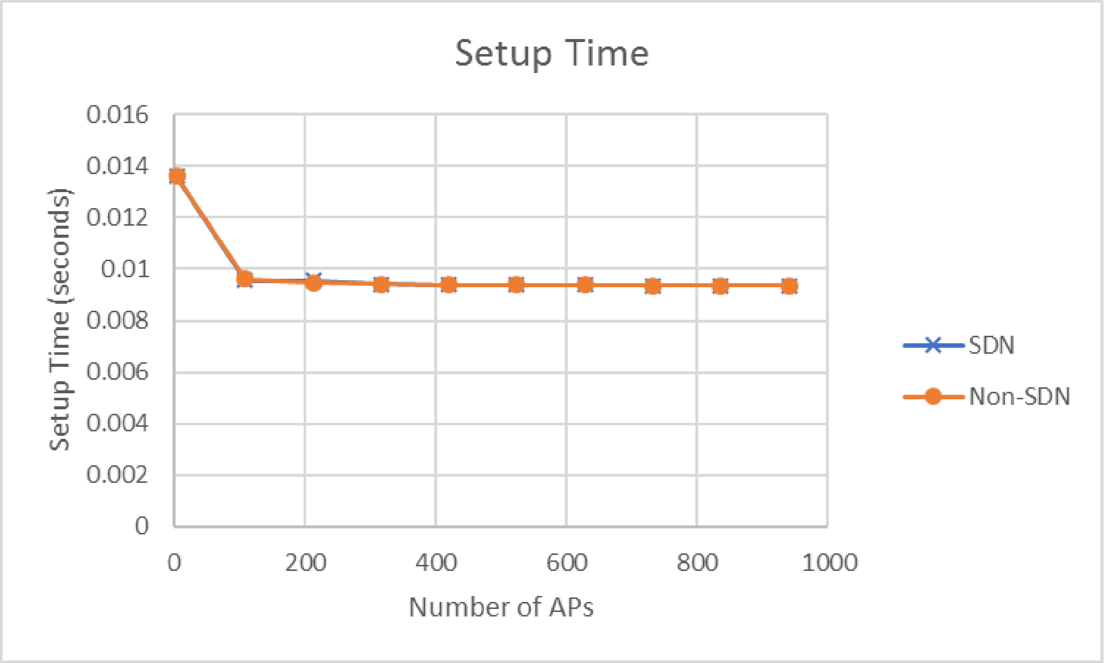


Figure 6.17: Average Setup Time Comparision, Test Case 5

Chapter 7

Conclusions and Future Work

SDN based networks have been known to increase wired network throughput. In this thesis, we have brought the SDN paradigm to wireless networks.
In chapter 2, we described a SDN-based wireless controller architecture. In chapter 3 and 4, we described how the Control plane and Management plane respectively, of the WLAN controller must be designed. Details of how the WLAN controller was implemented in-lab wer given in chapter 5. In chapter 6, a qualitative study was done via simulations, and it was noted that indeed, SDN based WLAN controllers perform better than their Non-SDN based counterparts.

Future work will include :

- Deploying an SDN controller to control a large scale WLAN test bed
- Develop Algorithms which take advantage of the global view provided by the SDN controller
- Continue to participate in IEEE standardization activities, specifically to participate in WG P1930.1 : SDN based Middleware for Control and Management of Networks

Appendix A

Yang Model for WLAN APs

We have created the following Yang model for wireless Access Points. The YANG model can be easily extended to include other configurable parameters of the Access Point.

```
module "openwrt-network"
{
  namespace "urn:ietf:params:xml:ns:yang:openwrt-network" ;
  prefix "net" ;
  organization "iitb" ;
  description "Module for controlling openwrt network parameters" ;
  revision "2017-06-04"
  {
    description "Initial commit" ;
  }

  container "default_config"
  {
    config "true";
    leaf "ssid"
    {
      type "string" {}
    }
    leaf "key"
    {
      type "string" {}
    }
    leaf "ctrl_ip"
    {
      type "string" {}
    }
    leaf "vlanid"
    {
      type "uint8" {}
    }
  }

  container "add_vlan"
  {
    config "true";
    leaf "vlanid"
    {
      type ”uint8” {}
      description ”specify vlanid” ;
    }
  }

  container ”wifi_parameters”
  {
    config ”true” ;
    leaf ”channel”
    {
      type ”uint8” {}
    description ”sets the transmission channel number” ;
    }
    leaf ”txpower”
    {
      type ”uint8” {}
      description ”set the transmission power” ;
    }
  }
}
```

Appendix B

Proposed Table of Contents for P1930.1

We propose that standard P1930.1 should cover the following topics:

1. **Overview**
  1.1. Scope
  1.2. Purpose
  1.3. Supplementary information on purpose
  1.4. Word Usage
2. **References**
3. **Definitions, Abbreviations, Acronyms**
4. **General Description**
  4.1. System Architecture
  4.2. Components of the Middleware
    4.2.1. Abstraction Layer
    4.2.2. North Bound Interface
    4.2.3. South Bound Interface
    4.2.4. Peer to Peer Interface
5. **Protocols**
  5.1. Management Protocols
  5.2. Control Protocols
6. **Protocol Messages**
  6.1. Management Plane
    6.1.1. Controller Discovery
    6.1.2. Device Access and Authorization
    6.1.3. Session Management
      6.1.3.1. Session Establishment
      6.1.3.2. Session Termination
    6.1.4. Device Management
      6.1.4.1. Firmware Upgrade
      6.1.4.2. Reset
      6.1.4.3. Reboot
      6.1.4.4. Configuration download
      6.1.4.5. Device Configuration
        6.1.4.5.1. Get Attribute

6.1.4.5.2. Set Attribute
6.1.4.6. Device Operation
6.1.4.6.1. Error Indication
6.1.4.6.2. Generic Indication
6.2. Control Plane
6.2.1. Connection Control
6.2.1.1. Association Setup & Admission Control
6.2.1.2. Authorization & Authentication
6.2.1.3. Mobility Control
6.2.2. Traffic/Service Flow Management
6.2.2.1. Flow Creation
6.2.2.2. Flow Deletion
6.2.3. QoS Control
7. **Abstraction Layer**
8. **North Bound Message Interface**
8.1. Message Structure
8.2. Description of fields
9. **Peer to Peer Interface**
10. **Security**
10.1. Authentication (for APs)
10.2. Encryption

Appendix C

OpenWrt configuration files

Given below are the board configurations required to set up dynamic VLAN and to configure Mikrotik boards as Whitespace BTS and WLAN APs

```
Board in BTS AP Mode
/etc/config/network
config interface 'lan'
        option ifname 'eth0'
        option type 'bridge'
        option proto 'static'
        option ipaddr '192.168.1.2'
        option netmask '255.255.255.0'
        option gateway '192.168.1.1'
config switch
        option name 'switch0'
        option reset '1'
        option enable_vlan '1'
config switch_vlan
        option device 'switch0'
        option vlan '1'
        option ports '1 2 5'


/etc/config/openflow
config 'ofswitch'
        option 'dp' 'dp0'
        option 'dpid' '000000000002'
        option 'ofports' 'br-lan'
        option 'ofctl' 'tcp:192.168.1.83:6653'
        option 'mode'  'outofband'
```

```

Board in BTS STA mode and Wi-Fi AP mode.
/etc/config/network
config interface 'wan'
        option ifname 'eth0'
        option type 'bridge'
        option proto 'static'
        option ipaddr '192.168.1.3'
        option netmask '255.255.255.0'
        option gateway '192.168.1.1'
config interface 'vlan1'         # Defining vlan
       option ifname 'eth1.1'    # Adding a sub-interface to eth1
       option type 'bridge'
       option proto 'dhcp'
config interface 'vlan3'          # Defining vlan
       option ifname 'eth1.3'     # Adding a sub-interface to eth1
       option type 'bridge'
       option proto 'dhcp'
config switch
       option name 'switch0'
       option reset '1'
       option enable_vlan '1'
config switch_vlan
       option device 'switch0'
       option vlan '1'
       option ports '1 2 5t'
config switch_vlan
       option device 'switch0'
       option vlan '3'
       option ports '5t'

/etc/config/wireless
config wifi-device 'radio1'
        option type 'mac80211'
        option channel '1'
        option hwmode '11g'
        option path 'pci0000:00/0000:00:14.0'
        option chanbw '20'
        option txpower '27'
        option log_level '0'
config wifi-iface
        option device 'radio1'
        option network 'wan'
        option mode 'sta'
        option ssid 'IITB-Whitespace'
        option encryption 'none'
        option wds '1'
        option rts '0'
config wifi-device 'radio0'
        option type 'mac80211'
        option hwmode '11g'
        option path 'pci0000:00/0000:00:13.0'
        option htmode 'HT20'
        option txpower '20'
        option channel '11'
config wifi-iface
        option device 'radio0'
        option mode 'ap'
        option ssid 'AP1'
        option encryption 'wpa2'
        option server '192.168.1.83'   #IP of radius server
        option key 'sharedsecret'
        option dynamic_vlan '2'        #VLAN checking is mandatory
        option vlan_tagged_interface 'eth1'
        option vlan_bridge 'br-vlan'
        option vlan_naming '0'

/etc/config/openflow
config 'ofswitch'
       option dp 'dp0'
       option dpid '000000000001'
```

```
        option ofctl 'tcp:192.168.1.83:6653' #IP of Openflow controller
        option ofports 'br-vlan1 br-vlan3 br-wan' #OpenFlow ports
        option mode 'outofband'
```

Bibliography

[1] B. Hedstrom, A. Watwe, and S. Sakthidharan, “National Telecom Policy,” *Department of Telecommunications*, 2012.

[2] “Openflow switch specifications.” Version 1.5.1.

[3] P. Calhoun, M. Montemurro, and D. Stanley, “Control and provisioning of wireless access points (capwap) protocol specification,” RFC 5415, IETF, March 2009.

[4] P. Calhoun, M. Montemurro, and D. Stanley, “Control and provisioning of wireless access points (capwap) protocol binding for ieee 802.11,” RFC 5416, IETF, March 2009.

[5] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, “Network configuration protocol (netconf),” RFC 6241, IETF, June 2011. http://www.rfc-editor.org/rfc/rfc6241.txt.

[6] I. Ramani and S. Savage, “Syncscan: practical fast handoff for 802.11 infrastructure networks,” in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 1, pp. 675–684, IEEE, 2005.

[7] B. Hedstrom, A. Watwe, and S. Sakthidharan, “Protocol Efficiencies of NETCONF versus SNMP for Configuration Management Functions,” *University of Colorado, Master Thesis*, 2011.

[8] “TR-069, CPE WAN Management Protocol.” https://www.broadband-forum.org/technical/download/TR-069.pdf. Technical Report, DSL Forum.

[9] M. Słabicki and K. Grochla, “Performance evaluation of SNMP, NETCONF and CWMP management protocols in wireless network,” in *Electronics, Communications and Networks IV: Proceedings of the 4th International Conference on Electronics, Communications and Networks (CECNET IV), Beijing, China, 12–15 December 2014*, p. 377, CRC Press, 2015.

[10] “Netopeer.” https://github.com/CESNET/netopeer.

[11] “ns-3.” https://www.nsnam.org/ns-3-26/.

[12] M. Bernaschi, F. Cacace, G. Iannello, M. Vellucci, and L. Vollero, “OpenCAPWAP: An open source CAPWAP implementation for the management and configuration of WiFi hot-spots,” *Computer Networks*, vol. 53, no. 2, pp. 217 – 230, 2009. QoS Aspects in Next-Generation Networks.

[13] S. Min, S. Kim, J. Lee, B. Kim, W. Hong, and J. Kong, "Implementation of an openflow network virtualization for multi-controller environment," in *2012 14th International Conference on Advanced Communication Technology (ICACT)*, pp. 589–592, Feb 2012.

[14] K.-K. Yap, R. Sherwood, M. Kobayashi, T.-Y. Huang, M. Chan, N. Handigol, N. McKeown, and G. Parulkar, "Blueprint for introducing innovation into wireless mobile networks," in *Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures*, pp. 25–32, ACM, 2010.

[15] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley, "Facilitating access point selection in ieee 802.11 wireless networks," in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pp. 26–26, Usenix Association, 2005.

[16] I. Papanikos and M. Logothetis, "A study on dynamic load balance for ieee 802.11 b wireless lan," in *Proc. COMCON*, vol. 2001, 2001.

[17] "IEEE standard for Information technology-Telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," *IEEE Std*, vol. 802, no. 11, 2016.

[18] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, "Software-defined networking (sdn): Layers and architecture terminology," RFC 7426, RFC Editor, January 2015. `http://www.rfc-editor.org/rfc/rfc7426.txt`.