UNIVERSIDADE FEDERAL FLUMINENSE

JUAN LUCAS DO ROSÁRIO VIEIRA

An SDN-based access point virtualization solution for multichannel IEEE 802.11 networks

NITERÓI

2020

UNIVERSIDADE FEDERAL FLUMINENSE

JUAN LUCAS DO ROSÁRIO VIEIRA

An SDN-based access point virtualization solution for multichannel IEEE 802.11 networks

Thesis presented to the Computing Graduate Program of the Universidade Federal Fluminense in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. Area: Computer Systems.

Advisor:
Diego Gimenez Passos

NITERÓI

2020

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

```
V657a   Vieira, Juan Lucas do Rosário
          An SDN-based access point virtualization solution for
        multichannel IEEE 802.11 networks / Juan Lucas do Rosário
        Vieira ; Diego Gimenez Passos, orientador. Niterói, 2020.
          74 f. : il.

          Dissertação (mestrado)-Universidade Federal Fluminense,
        Niterói, 2020.

        DOI: http://dx.doi.org/10.22409/PGC.2020.m.12720601799

          1. Wireless Networks. 2. Virtual Access Points. 3. Software
        Defined Networking. 4. Channel Switch Announcement. 5.
        Produção intelectual. I. Passos, Diego Gimenez, orientador.
        II. Universidade Federal Fluminense. Instituto de
        Computação. III. Título.

                                              CDD -
```

Bibliotecária responsável: Fabiana Menezes Santos da Silva - CRB7/5274

Juan Lucas do Rosário Vieira

An SDN-based access point virtualization solution for multichannel IEEE 802.11 networks

Thesis presented to the Computing Graduate Program of the Universidade Federal Fluminense in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. Area: Computer Systems.

Approved in March, 2020.

THESIS DEFENSE COMMITTEE

[illegible]

Prof. Diego Passos, D.Sc. – Advisor, UFF

[illegible]

Prof. Célio Albuquerque, Ph.D. – UFF

[illegible]

Prof. Katia Obraczka, Ph.D. – University of California, Santa Cruz

Niterói

2020

"Nothing in life is to be feared, it is only to be understood. Now is the time to understand more, so that we may fear less." — Marie Curie.

Acknowledgements

First, I would like to thank my family for all the support and patience during this journey. I am thankful to my mother, Fatima — my greatest example of love, kindness, and generosity — and my father, Antonio, for encouraging me to pursue my dreams and providing me with financial and emotional aid. To my brother Jean for cheering me to believe in my abilities. And, to my nephew Arthur, who brightened my days with his smile.

I would like to thank all the professors at the Institute of Computing that make our university even better, for all the lessons acquired during my academic life. Especially to my advisor Professor Diego Passos, who is always willing to help and share his vast knowledge and who I look up to as a professional.

I also would like to thank my colleagues at Universidade Federal Fluminense and Laboratório Mídiacom, that were available to assist and made this journey lighter with the relaxation moments and laughs.

Finally, I would like to thank CAPES for the scholarship and also TAESA and P&D ANEEL (PD-07130-0053/2018) for the financial support.

Resumo

A popularização de dispositivos móveis trouxe novas aplicações para as redes de comunicação. No entanto, à medida em que avançamos para um cenário sem fio cada vez mais denso, com um número maior de objetos conectados, problemas como o desbalanceamento de carga nos nós e colisões entre transmissões se tornam mais graves. Além disso, embora o *handover* realizado pelo cliente seja ineficiente para mitigar esses desafios, decisões de migração baseadas na utilização de toda a rede permitiriam um melhor gerenciamento de seus recursos. Este trabalho propõe uma solução de virtualização de pontos de acesso baseada no paradigma de *Software Defined Networking* para permitir a migração de clientes realizada pela infraestrutura de rede, com base em um escopo global. Diferentemente de outras soluções encontradas na literatura, a proposta oferece suporte a migrações multicanal por meio do mecanismo de *Channel Switch Announcement* do IEEE 802.11h sem restringir a utilização dos canais nos pontos de acesso. Para demonstrar a viabilidade de tal abordagem, dados experimentais sobre o comportamento de vários dispositivos diferentes em face deste mecanismo são apresentados. A solução completa de virtualização também é avaliada, revelando que o *handover* de estações não ocasionou atrasos ou perdas de pacotes significativos nas conexões dos clientes, enquanto proporcionou maior flexibilidade no gerenciamento de redes sem fio IEEE 802.11.

Palavras-chave: Redes sem Fio, Pontos de Acesso Virtuais, Redes Definidas por Software, Channel Switch Announcement.

Abstract

The popularization of mobile devices has brought several new applications to communication networks. However, as we move into an increasingly denser scenario with a more significant number of connected objects, problems such as load unbalance and collisions between transmissions become more severe. Moreover, while client-based handoff is inefficient to mitigate these issues, network-wide migration decisions would allow better management of network resources. This work proposes an access point virtualization solution based on the SDN paradigm to allow client handover conducted by the network, based on a global scope. Differently from other solutions found in the literature, the proposal supports multichannel migrations by means of the IEEE 802.11h Channel Switch Announcement without restricting the channel utilization by the access points. To demonstrate the feasibility of such an approach, we present experimental data regarding the behavior of several different devices in face of this mechanism. We also evaluate our complete virtualization solution, which reveals that the handover of stations did not lead to significant delays or packet losses in the clients' connections while providing greater flexibility to the management of IEEE 802.11 wireless networks.

Keywords: Wireless Networks, Virtual Access Points, Software Defined Networking, Channel Switch Announcement.

List of Figures

2.1 Example of an infrastructure-based IEEE 802.11 network. 7

2.2 Example of an IEEE 802.11 network with several devices operating in ad-hoc mode. 8

2.3 Examples of Basic Service Sets. On the left, two infrastructure BSSs joined by a Distribution System, creating an ESS. On the right side, an independent BSS formed by three nodes. 10

2.4 Non-exhaustive timeline of the IEEE 802.11 standard evolution. 11

2.5 Format of a Channel Switch Announcement element in an IEEE 802.11 beacon (Adapted from [4]). 13

2.6 Overview of the Software Defined Networking architecture. 14

2.7 Overview of the centralized, distributed and hierarchical control planes of a SDN. 16

4.1 Overview of the virtualization solution. Each physical AP hosts one or more VAPs. The controller manages the physical and virtual APs. 26

4.2 Example of VAP migration managed by the controller. The VAP 1 is transferred from AP1 to AP2. 27

4.3 Steps of the multichannel virtual access point migration procedure. 28

4.4 Conceptual diagram of the developed controller. 32

5.1 Scenarios used to compare a channel switch with announcement versus without any warning to the station. 37

5.2 How the throughput measured by a station is affected by the AP switching channels with and without the CSA mechanism. The confidence interval is based on 10 executions with a confidence level of 95%. 38

5.3 Scenario for the evaluation of the overhead caused by multiple BSSs in a single physical AP. 41

5.4 Comparison of the average throughput of each station. The confidence interval is based on a confidence level of 95%. 41

5.5 Jain's fairness index of each BSS configuration of the experiment. 42

5.6 Scenario used to evaluate the proposed multichannel virtualization solution. It is composed of three wireless clients, two physical access points and one controller. 43

5.7 Round trip time perceived by each station during the experiment. A moving average with a sliding window of 30 points was used to smooth the data variation. The migrated VAP is moved between physical APs ten times. The greyed out area represents the periods in which the migrating VAP was hosted by the AP in channel 6. 44

5.8 Overview of the topology used for throughput, outage time and packet loss evaluation of the proposed solution. 45

5.9 Average TCP and UDP throughput experienced by the station in three different handoff scenarios. The confidence interval is based on 30 executions with a confidence level of 95%. 46

5.10 Average outage times experienced by the station in three handoff scenarios. The 95% confidence intervals are based on 30 executions of the experiment. 47

5.11 Median outage times experienced by the station in three handoff scenarios. The median is based on 30 executions of the experiment. 48

5.12 Average packet loss percentage experienced by the station through the duration of the experiment (40 seconds). The confidence interval is based on 30 executions with a confidence level of 95%. 49

5.13 Average packet loss experienced by the station before, during and after the handover and the average number of packets transmitted during the experiment. The confidence interval is based on 30 executions with a confidence level of 95%. 50

5.14 Average TCP and UDP packet rate measured in the source. The confidence interval is based on 30 executions of the experiment. 51

List of Tables

3.1 Comparison between related work and the proposed solution. 24

4.1 Station-related information that are accessible from the application plane. . 34

4.2 Commands supported by the REST API of the controller. 35

5.1 Behavior of computers and laptops with different hardware and operating systems upon receiving a Channel Switch Announcement IE. 39

5.2 Behavior of different smartphones and tablets upon receiving a Channel Switch Announcement. 39

Acronyms

ANATEL	Agência Nacional de Telecomunicações	11
AP	Access Point	1
API	Application Programming Interface	21
ARP	Address Resolution Protocol	21
BSS	Basic Service Set	8
BSSID	Basic Service Set Identifier	8
CLI	Command Line Interface	31
CSA	Channel Switch Announcement	2
DFS	Dynamic Frequency Selection	12
DS	Distribution System	9
ECSA	Extended Channel Switch Announcement	13
ESSID	Extended Service Set Identifier	9
ESS	Extended Service Set	9
FCC	Federal Communications Commission	11
HTTP	Hypertext Transfer Protocol	34
IBSS	Independent Basic Service Set	8
IEEE	Institute of Electrical and Electronics Engineers	5
IE	Information Element	3
IP	Internet Protocol	48
ISM	Industrial, Scientific and Medical	11
IoT	Internet of Things	9
JVM	Java Virtual Machine	32
LAN	Local Area Network	5
MAC	Medium Access Control	10

NIC	Network Interface Card	6
NTP	Network Time Protocol	41
NOS	Network Operating System	14
QoE	Quality of Experience	43
QoS	Quality of Service	17
REST	Representational State Transfer	33
RSSI	Received Signal Strength Indicator	18
RTT	Round Trip Time	42
SDN	Software Defined Networking	2
SSID	Service Set Identifier	9
TBTT	Target Beacon Transmission Time	13
TCP	Transmission Control Protocol	37
TPC	Transmit Power Control	12
UDP	User Datagram Protocol	45
URI	Uniform Resource Identifier	34
VAP	Virtual Access Point	2
VoIP	Voice over IP	13
WLAN	Wireless Local Area Network	1

Contents

1 Introduction **1**

1.1 Contributions 3

1.2 Text Outline 3

2 Background **5**

2.1 IEEE 802.11 and Wi-Fi 5

2.1.1 Architecture 6

2.1.1.1 Infrastructure Architecture 6

2.1.1.2 Ad-hoc Architecture 7

2.1.2 Basic Service Set 8

2.1.2.1 Basic Service Set Identifier 8

2.1.2.2 Service Set Identifier 9

2.1.3 Extended Service Set 9

2.1.4 Evolution 9

2.1.5 Spectrum Management Services 11

2.1.5.1 Channel Switch Announcement 12

2.2 Software Defined Networking 14

2.2.1 Architecture 14

2.2.1.1 Centralized and Distributed Control Planes 15

2.2.2 OpenFlow 16

3 Related Work **18**

3.1 Client-side virtualization 18
3.2 Network-only modification 19
3.3 Access Point Virtualization 20
3.3.1 Multichannel support 22
3.4 Overview 23

4 A Lightweight Virtualization Solution 25

4.1 Architecture 25
4.2 VAP Handover 27
4.3 Implementation 29
4.3.1 Access Points 29
4.3.2 Southbound API 31
4.3.3 Control Plane 32
4.3.4 Northbound API 33

5 Evaluation 36

5.1 CSA performance assessment 36
5.2 CSA client behavior 38
5.3 Solution Evaluation 40
5.3.1 VAP Overhead 40
5.3.2 Handover Benchmark 42
5.3.2.1 Round Trip Time 43
5.3.2.2 Throughput, Outage Time and Packet Loss 44

6 Conclusion 53

6.1 Limitations 54
6.2 Future work 55

References **57**

Chapter 1

Introduction

The increasing availability of portable computers led to the necessity of connecting devices providing better mobility support. In contrast to its wired counterpart, Wireless Local Area Networks (WLANs) have improved the convenience in communication between devices by dispensing the need for wired connection and paved the way for new applications, enabling the exchange of data in several scenarios.

Currently, WLANs based on the IEEE 802.11 standard — often referred to as Wi-Fi networks — are widespread, being found in several locations and supported by a plethora of devices, such as light-bulbs, fridges, video-games, TVs, smartphones and laptops. As we move to even denser wireless networks — with a higher number of Access Points (APs) and wireless clients using the same shared medium — tasks such as load balancing, interference control, and network resource optimization need to be done to avoid the performance degradation of the network.

In traditional IEEE 802.11 networks, the wireless station is responsible for the handoff decision [6]. The handoff is a term used to refer to the process in which a wireless client disassociates from the current access point and associates with a new AP with the goal of improving its connection to the network. The decision is usually limited to the analysis of the signal generated by each AP as received by the client radio [6]. This migration[1] is a costly process, as it requires the station to perform a scan of the wireless medium, followed by the association with a new AP. Thus, the time required to execute these steps can lead to the interruption of communication for several seconds. In addition, the variability of the signal perceived by the radio of a station can lead to sequential reassociations [6], which might severely jeopardize applications that require the timely delivery of packets.

[1]In this work, we use the words *migration* and *handover* to refer to the same handoff process.

Furthermore, characteristics such as the current load of the AP or the interference between transmitting devices are not known by the station. These additional pieces of information could improve the AP selection process by providing a broader view of the network state. Furthermore, decisions could be based on what is globally best for the network — instead of what is locally best for the station —, aiming at better resource utilization and mitigation of interference.

Several works have aimed at bringing improvements to the handover process in wireless networks. Some authors propose modifications to wireless stations [18], allowing them to connect to multiple APs simultaneously. However, this approach results in interoperability issues with current mobile devices. Another strand [28, 7, 13, 29] uses the concept of access point virtualization to transfer the handover responsibility to the WLAN, similarly to how migrations are handled in cellular networks [11]. In these solutions, each station is associated with a Virtual Access Point (VAP) which might, for example, be transferred between physical access points when convenient.

However, part of the access point virtualization solutions restricts the channel utilization of APs in the wireless networks [13, 29, 27, 30] — either requiring neighbor APs to operate in different channels or all APs to operate in a single channel. Other solutions are only able to make limited-scope handover decisions, based on the information (*e.g.*, signal strength experienced by the APs) exchanged between a few close access points [7].

The addition of a central controller connected to the physical access points of a wireless network allows client[2] handover to be performed based on the overall network utilization. In comparison to the limited-scope choices — such as client-based or those based on a single AP — these network-wide handover decisions enable better resource management and new applications. As an example, if node utilization data is accessible by the control entity, a load-balancing algorithm can be performed and clients migrated in order to better distribute the network load among the APs. Likewise, if the goal is to reduce the energy consumption of the network, clients can be migrated to cluster them in fewer APs, allowing the shutdown of unused APs.

This work proposes a lightweight access point virtualization solution for IEEE 802.11 networks that employs the Channel Switch Announcement (CSA) [4] mechanism to enable the handover of clients between physical access points even if they operate on separate channels. The proposed architecture also includes a control entity for the centralized decision of client handover, borrowing from the concept of Software Defined Networking

[2]In this work, we use stations and wireless clients as synonyms.

(SDN). This paradigm consists of the disassociation of the control and data planes in forwarding devices, with the centralization of the control in an entity called *controller*, which is responsible for the management of the network [24].

1.1 Contributions

Below, we summarize the main contributions of this thesis:

- a virtualization solution that enables network-based handover of stations between access points operating in different channels;
- demonstration, through multiple experiments, that the proposed solution is, in fact, valid and that performs no worse than the traditional station-driven handover scheme of IEEE 802.11 networks;
- analysis of the behavior of several computers and mobile devices upon receiving a beacon with the CSA Information Element (IE); and
- an implementation[3] of the proposed solution composed by an AP-side software and the control entity.

1.2 Text Outline

The remainder of this thesis is organized as follows. Chapter 2 reviews essential concepts that help understanding the proposed solution. It presents the architectural aspects of the IEEE 802.11 standard, its spectrum management services, and its evolution through the publishing of new amendments and revisions. Also, it covers the architecture of the SDN paradigm and the OpenFlow protocol.

Chapter 3 covers the literature related to our work. It highlights proposals that aim at reducing the overhead of station handover and improving the flexibility of WLANs. At the end of the chapter, an overview and comparison of the mentioned works is presented.

Chapter 4 describes the virtualization solution proposed in this work. It includes architectural, behavior, and implementation aspects of our solution and how SDN and the CSA mechanism were employed in it.

[3]Available at: https://github.com/juanlucasvieira/VAP-SDN

Chapter 5 covers the evaluation of our work. First, a performance and behavior analysis regarding the CSA mechanism is presented. Then, we assess the impact of our solution by exhibiting the performed experiments and their results.

Finally, Chapter 6 concludes this thesis. It discusses the benefits, limitations, and applications of our work. It also addresses further enhancements regarding the proposed solution.

Chapter 2

Background

The history of wireless data communication dates from the late 1800s, with the creation of wireless telegraphs [12]. Wireless communication based on packet-switching appeared only in the 1970s, with the establishment of ALOHAnet [26], at the University of Hawaii. However, the popularization of this type of communication only occurred with the emergence of mobile devices, which started the dissemination of WLANs as a more convenient and mobility-friendly extension of Local Area Networks (LANs).

Currently, the IEEE 802.11 standard is the foundation of most modern WLANs — being available in several locations and mobile devices. Since our solution is focused on networks based on this standard, we first review the main concepts behind its architecture, which will clarify some aspects of our proposal. Then, we present an overview of the evolution of the standard over the years. Also, our solution depends on the IEEE 802.11h's CSA mechanism to allow seamless handover of stations between physical APs operating in different channels. Thus, its operation will be explained with more detail at the end of the first section. The proposed architecture also relies on the SDN paradigm. Therefore, we present a review of its key concepts and protocols in the second section of the chapter.

2.1 IEEE 802.11 and Wi-Fi

In 1990, the Institute of Electrical and Electronics Engineers (IEEE) began the discussion to develop a standard for WLANs. Seven years later, the earliest version of the IEEE 802.11 standard was published [16], which brought several specifications for the physical and link layers of these wireless networks. However, the first-released products that implemented the standard suffered from severe interoperability problems [25].

To resolve the problems caused by discrepancies in the implementation of the standard, several manufacturers created, in 1999, a working group that would be responsible for certifying and verifying the interoperability of IEEE 802.11 compatible devices, which was later called the Wi-Fi Alliance[1]. Because of this association between Wi-Fi certification and the IEEE 802.11 standard, WLANs have become popularly known as Wi-Fi networks.

2.1.1 Architecture

The IEEE 802.11 standard defines different architectures — and operating modes — that can be adopted by the WLAN depending on the purpose of the network:

- the Infrastructure architecture; and
- the Ad-hoc architecture.

The specific characteristics and restrictions of each architecture will be addressed in the following sections.

2.1.1.1 Infrastructure Architecture

The IEEE 802.11 infrastructure WLAN defines a hierarchical architecture in which each wireless node is either an AP or a station.

Stations operate in managed mode and are considered wireless clients since they are devices with an integrated wireless Network Interface Card (NIC) that want to communicate in the network. In contrast, the APs, or base stations, are considered part of the network infrastructure, since their only purpose is to provide connectivity to stations. Their wireless NIC must operate in the master mode. As an analogy with its wired counterpart, an AP can be seen a network switch that connects several computers in a LAN [20]. Some APs might also contain wired interfaces to establish a connection between wireless and wired networks.

In an infrastructure-based network (see Figure 2.1), the communication between wireless clients is always relayed by the AP. Therefore, stations must be associated with an AP to exchange data. This association between the AP operating as master and stations

[1]`https://www.wi-fi.org/who-we-are/history`

operating in managed mode creates a link in which an AP will only forward data transmitted by an associated client, and the client will only process data received from the currently associated AP. Each station may associate with only one AP at a time.

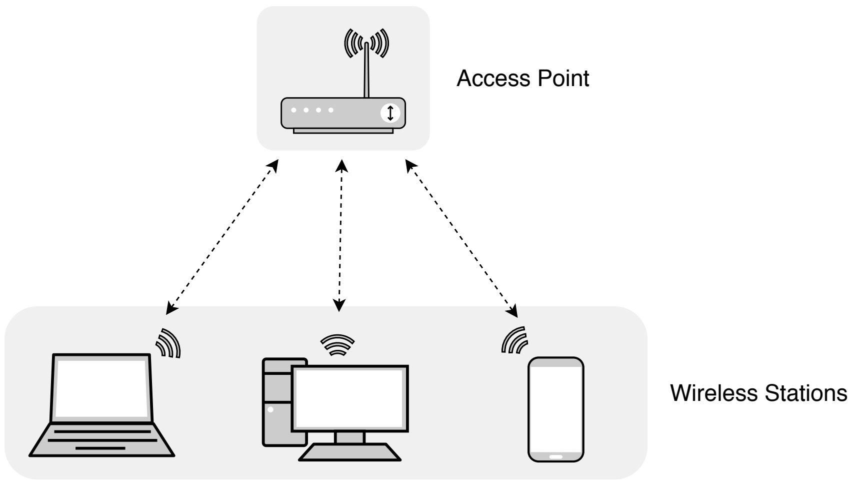


Figure 2.1: Example of an infrastructure-based IEEE 802.11 network.

The infrastructure architecture is the most commonly used type of IEEE 802.11 network and can be easily found in domestic and commercial buildings to provide Internet access to its users. However, in some scenarios (*e.g.*, remote, hard to reach areas), it is not feasible or desirable to use a network infrastructure to exchange data between wireless devices. Therefore, the IEEE 802.11 standard also allows the utilization of an ad-hoc architecture.

2.1.1.2 Ad-hoc Architecture

In contrast with the previous architecture, devices with a wireless NIC operating in ad-hoc mode can transmit data directly to each other without requiring an association with an AP, as shown in Figure 2.2. The ad-hoc architecture is best suited for occasional device communication as it does not require the use of an infrastructure to relay data transmitted between devices.

Previously, the IEEE 802.11 standard only provided support for direct data transmission from source to destination, in a single-hop manner. This limitation implied a constraint in which the communication between devices was limited by the radio range of the nodes. Either the devices would have to be in the range of each other, or a routing protocol implemented at the upper layers would be responsible for making the devices route packets in a multi-hop manner [25].

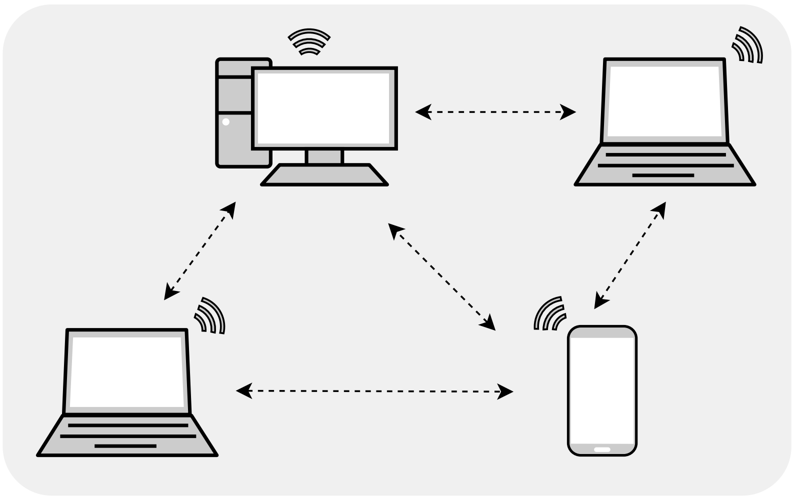


Figure 2.2: Example of an IEEE 802.11 network with several devices operating in ad-hoc mode.

Due to this need for establishing routes and transmitting packets through intermediate nodes, the IEEE 802.11s amendment was formulated, defining a mesh-like architecture and layer-2 routing algorithms that enabled multi-hop data delivery to one or multiple destinations in ad-hoc networks [8].

2.1.2 Basic Service Set

A set of network nodes that are capable of communicating directly in an ad-hoc manner or through an access point (AP) is called a Basic Service Set (BSS), which is the basic building block of an IEEE 802.11 network [5]. Two common types of BSS are:

- Infrastructure BSS: a set composed of an AP and the stations associated with it; or
- Independent Basic Service Set (IBSS): a set composed of network nodes that communicate using the ad-hoc mode.

2.1.2.1 Basic Service Set Identifier

An IEEE 802.11 network can consist of one or multiple BSSs. The Basic Service Set Identifier (BSSID) is a 6-octet MAC address-like number used to differentiate one BSS from others.

In an Infrastructure BSS the BSSID is usually derived from the MAC address of the wireless NIC of the AP, whereas in an IBSS the individual/group bit is set to 0, the universal/local bit of the address is set to 1 and the remaining 46 bits are usually randomly generated. The IEEE 802.11 standard also specifies a wildcard BSSID — composed of all 48 bits set to 1 — used in specific situations to represent all BSSs of the network [5].

2.1.2.2 Service Set Identifier

The Service Set Identifier (SSID) is also a identifier for BSSs. However, in contrast to the BSSID, it has a variable structure that can assume up to 32 octets. The standard also specifies a wildcard for SSIDs which is represented as a empty field.

Since the SSID is usually a human-readable identifier, this field is often referred to as the WLAN “name” in the user interface of the devices. The SSID is a field used in many management operations, such as broadcasting the existence of the WLAN to nearby devices.

2.1.3 Extended Service Set

In certain scenarios, a single BSS might not be sufficient to provide connectivity to a building or a large household. Thus, the IEEE 802.11 standard specifies that multiple BSSs can be connected to expand the coverage area or to increase the capacity of a IEEE 802.11 network.

A set of connected BSSs receives the name of Extended Service Set (ESS). An ESS can achieved through the interconnection of the APs of each BSS by a wired infrastructure — for example, a set of layer-2 switches. This architectural component used to connect multiple BSSs is defined by the standard as a Distribution System (DS)[2]. In an ESS, the SSID is referred to as an Extended Service Set Identifier (ESSID), and each AP contained in it must use the same ESSID. Figure 2.3 shows an example of an ESS with two BSSs.

2.1.4 Evolution

The IEEE 802.11 standard is constantly evolving. Since its first publication in 1997, several amendments [2] have been created with the aim of bringing improvements — such as higher data rates and improved security — or to modify the standard to deal with different legislation or scenarios. Vehicular networks, Internet of Things (IoT), and mesh networking are some applications considered by these extensions. Below, we present the

[2]The DS is not considered part of the ESS.

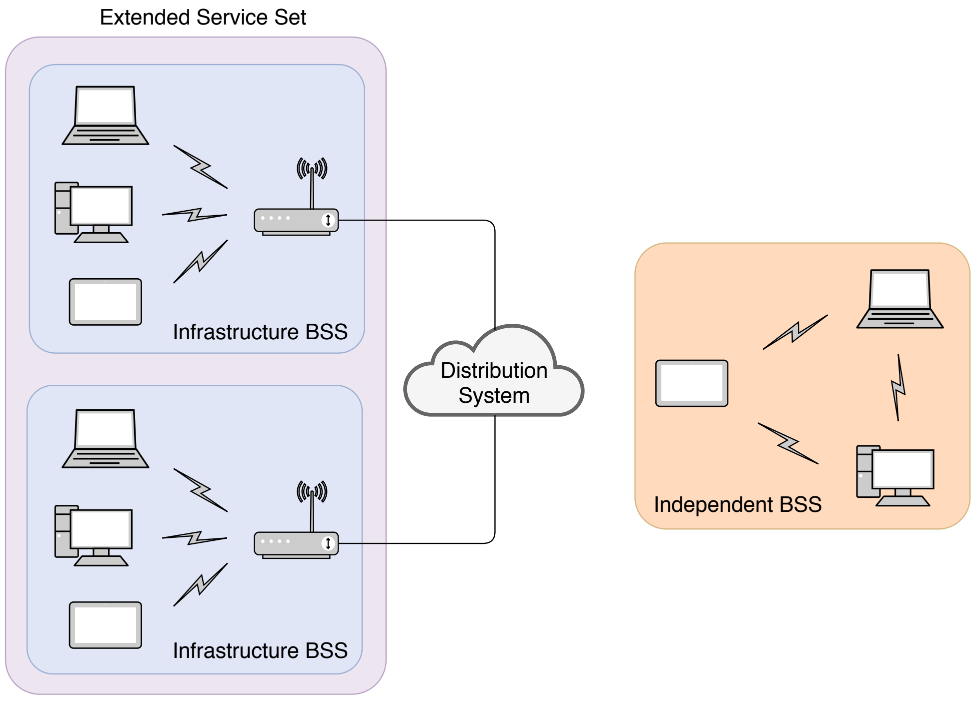


Figure 2.3: Examples of Basic Service Sets. On the left, two infrastructure BSSs joined by a Distribution System, creating an ESS. On the right side, an independent BSS formed by three nodes.

main amendments to the standard:

- **IEEE 802.11a** - Improvements to the physical layer that enabled rates up to 54 Mb/s for the 5 GHz band.
- **IEEE 802.11b** - Improvements to the physical layer that enabled rates up to 11 Mb/s for the 2.4 GHz band.
- **IEEE 802.11d** - Addresses operation in additional regulatory domains.
- **IEEE 802.11g** - Enhances data transfer rates up to 54 Mb/s for the 2.4 GHz band.
- **IEEE 802.11h** - Improvements in spectrum and transmission power management.
- **IEEE 802.11i** - Security enhancements for the Medium Access Control (MAC) layer.
- **IEEE 802.11n** - Enhances data transfer rates up to 600 Mb/s for the 2.4 GHz and 5 GHz bands.
- **IEEE 802.11p** - Includes specifications for vehicular communication.
- **IEEE 802.11s** - Includes specifications for mesh networking.

- **IEEE 802.11ac** - Enhances data transfer rates up to several gigabits per second for the 5 GHz band.
- **IEEE 802.11ad** - Enhances data transfer rates up to several gigabits per second for the 60 GHz band.
- **IEEE 802.11ah** - Enables support for 900 MHz operation and includes several mechanisms to deal with dense deployments, focusing on the communication of IoT devices.
- **IEEE 802.11ax** - Currently in progress. Modifies the physical and medium access control layers for higher efficiency and higher data transfer rate. Marketed as Wi-Fi 6.

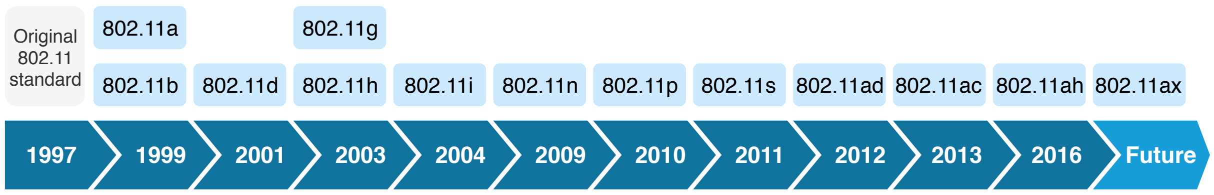


Figure 2.4: Non-exhaustive timeline of the IEEE 802.11 standard evolution.

Figure 2.4 shows a timeline of the IEEE 802.11 standard and the publication date of the aforementioned amendments. Since 1997, four revisions of the original version were published in 1999, 2007, 2012 and 2016, which incorporated several amendments[3] into the base standard. However, being included in the base standard does not imply that new released Wi-Fi enabled devices must support all the functionalities introduced by the amendments, considering that some of these extensions are not mandatory and may vary according to the vendor.

2.1.5 Spectrum Management Services

The IEEE 802.11 networks operate in the Industrial, Scientific and Medical (ISM) radio band, which can be used without requiring the acquisition of a license. Despite being license-free, the ISM band is under regional regulations of governmental agencies — such as the Federal Communications Commission (FCC) in the USA or the Agência Nacional de Telecomunicações (ANATEL) in Brazil. These regulations impose shared medium utilization restrictions to avoid interference between radio-based services. These restrictions

[3]Amendments *a*, *b*, *d*, *g*, *h*, *i*, *j*, *e*, *k*, *r*, *y*, *w*, *n*, *p*, *z*, *v*, *u*, *s*, *ae*, *aa*, *ad*, *ac*, *af* were incorporated into the base standard.

vary according to the country, since the same communication service (*e.g.*, cellular network, TV broadcasting) may use different frequencies depending on the country in which it is operating. Therefore, changes need to be made to enable the interoperability of these wireless networks with other radio-based communication.

To avoid co-channel operation between radars operating in the 5 GHz band and satisfy regulatory requirements of WLANs operating in this same band, the IEEE 802.11h extension was created [4]. This amendment provides several spectrum management services to IEEE 802.11 networks, such as Transmit Power Control (TPC) and Dynamic Frequency Selection (DFS). These services permit, for example, the specification of a maximum transmit power for a channel or to discontinue operation in a specific channel after a radar using the same frequency is detected.

In our proposal, we used a mechanism introduced by the IEEE 802.11h amendment, which is part of the DFS service, whose primary purpose is to assist channel switching after a radar detection.

2.1.5.1 Channel Switch Announcement

The Channel Switch Announcement is a mechanism introduced in the IEEE 802.11h amendment which can be used by a mesh station, an AP in an infrastructure BSS or a station in an IBSS to announce that it will change its channel of operation [4].

Regarding its utilization in an infrastructure BSS, the CSA allows warning associated clients that the AP's operating frequency will change. From the moment the AP first announces a channel switch until the actual frequency change, the access point may also block transmissions from nearby stations (an option that is informed within that CSA).

The CSA IE[4] is the structure used to communicate the channel switch. This element can be added to beacons, probe responses or action frames transmitted by APs. Stations operating in managed mode, however, shall not transmit the CSA IE. As can be seen in Figure 2.5, the format of the CSA IE consists of five pieces of information:

- the *Element ID* field identifies the type of the IE, which, in the case of a CSA, is represented by 37;
- the *Length* field specifies the number of following octets that are part of the IE, which is 3 octets in this example;

[4]An Information Element is a variable data structure, usually included in management frames, used by several functionalities of the 802.11 standard.

- the *New Channel Number* field indicates the number of the new channel to which the particular radio will transfer;
- the *Channel Switch Count* field indicates the number of Target Beacon Transmission Times (TBTTs) until the channel change is performed; and
- the *Channel Switch Mode* field indicates potential restrictions on associated stations' transmissions: a value of 1 denotes a transmission constraint, while the value 0 indicates that the stations can continue to transmit data during the period of transition announcement. In an IBSS, the station may treat this field advisory.

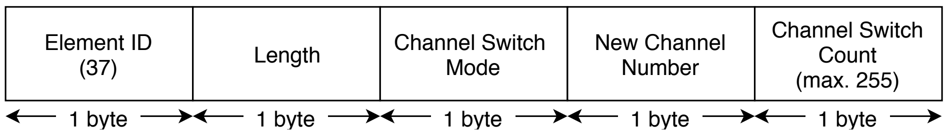


Figure 2.5: Format of a Channel Switch Announcement element in an IEEE 802.11 beacon (Adapted from [4]).

The IEEE 802.11 standard also provides an Extended Channel Switch Announcement (ECSA) mechanism. In addition to the CSA, it also permits specifying the new operating class[5] to which the transmitter will switch, allowing, for example, to change the bandwidth of the channel. Likewise, the ECSA IE contains all the fields of the CSA IE, with the inclusion of the *New Operating Class* field, which indicates the number of the operating class after the switch.

If an access point performs a channel switch operation without using the CSA mechanism, associated stations would need first to detect the unavailability of the AP, re-scan the wireless medium to find out which channel the AP in question has switched to and later perform the reassociation and reauthentication processes. This can cause considerable delay [6], jeopardizing applications that demand timely transmission of packets (*e.g.*, streaming, Voice over IP (VoIP)). The use of the CSA mechanism can prevent the stations from repeating this reassociation procedure with the access point, significantly reducing the delay of this operation. Besides, considering the handover of wireless stations, the inclusion of the information element in beacon frames enables the handover of clients between access points that are operating on different channels, as further explained in Section 4.1.

[5]The operating class defines a set of permitted values and behavior limits for radio operation in a regulatory domain.

2.2 Software Defined Networking

The Software Defined Networking paradigm proposes the separation of the data and the control planes in a network. Unlike traditional networks, in which each switch, router or access point is responsible for decision making and packet forwarding, in this architecture, duties such as routing and overall management decisions are transferred to an entity that possesses a global view of the network, known as the *controller* or Network Operating System (NOS) [14].

2.2.1 Architecture

The decoupling promoted by the SDN paradigm consists of dividing routing and network management roles into three main architectural components: the data, control and application planes, as shown by Figure 2.6.

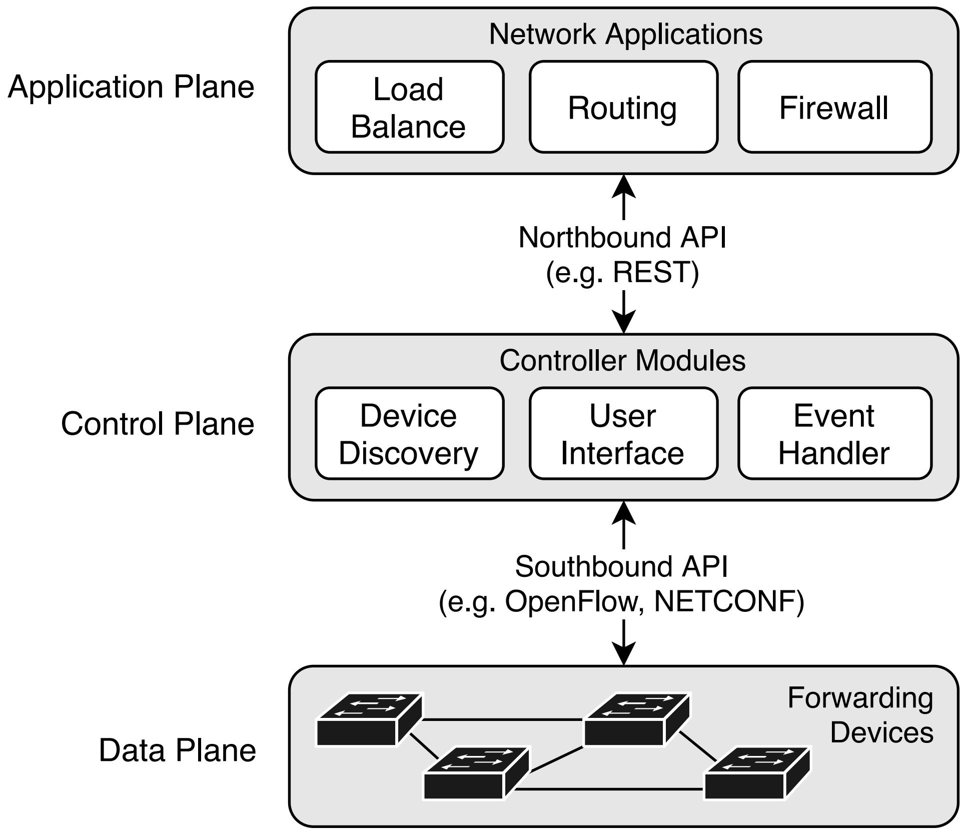


Figure 2.6: Overview of the Software Defined Networking architecture.

The *data plane* is constituted of network nodes responsible for forwarding packets in the network. These nodes do not make any decisions on the network. They handle incoming packets by taking actions defined by forwarding rules stored in their data structures.

The *control plane* consists of the controller and its modules, which usually provide several network services to facilitate the management of the network, such as device discovering and statistics collecting. The controller communicates with the forwarding devices through a *Southbound API*, such as the *OpenFlow* [21] or NETCONF protocols, which defines a set of control messages that can be exchanged between the device and the controllers. These messages allows the controller to acquire information regarding the data plane and to install or remove packet forwarding rules in the forwarding devices while the network is in operation. The control plane usually provides *Northbound API*, which contains several functions and abstractions that can be used by the application plane.

Finally, the *application plane* consists of network applications that run on top of the controller, using its exposed abstractions. These applications might perform tasks such as load balancing, routing, and intrusion detection.

2.2.1.1 Centralized and Distributed Control Planes

The control plane of an SDN can be implemented using a centralized or distributed approach. In the first case, a single server running the NOS is employed on the network and is responsible for managing all devices in the data plane.

In the distributed approach, multiple servers running the controller are employed on the network. Each local controller is responsible for the control of flows that travel on a subset of switches of the data plane. In order to maintain consistency between the rules and the global view of the network, the controllers communicate with one another through *eastbound* or *westbound* interfaces [19]. There are also hierarchical approaches, which consider a global controller responsible for all local controllers. Figure 2.7 presents an overview of these three approaches for the control plane.

The centralized approach allows centralized network control in one place, eliminating the burden of employing mechanisms that ensure consistency of data plane information between multiple controllers. However, this approach has a single point of failure and can render the network completely inoperable if the central controller experiences problems. The distributed approach provides better network scalability since the management load of the nodes is distributed among the multiple instances of the controllers. Besides, it is more resilient, since the network load of a failed controller could be transferred to the

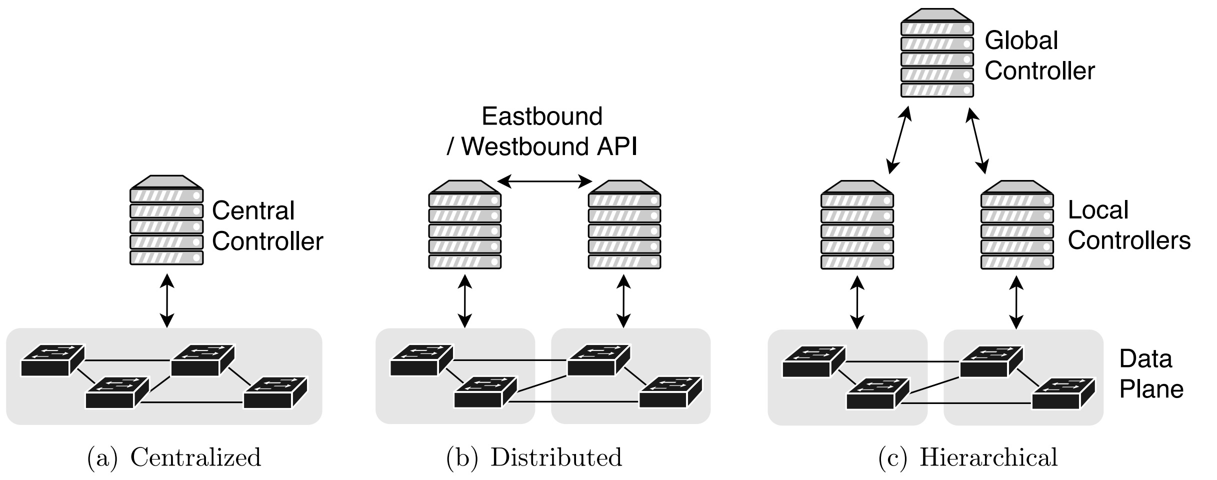


Figure 2.7: Overview of the centralized, distributed and hierarchical control planes of a SDN.

other controllers in the network.

2.2.2 OpenFlow

Although it was initially thought as a way for researchers to execute experimental protocols in traditional networks, OpenFlow [21] became a popular communication protocol between the data plane and control plane.

OpenFlow defines commands that can be used to insert and modify forwarding rules in the flow tables of the switches managed by the controller. These rules are based on a match-action approach, in which the header of a packet is compared to the match field of the rule and, in case of a positive match, the specified action (*e.g.*, forward the packet through a specified port) of the rule is taken. If the switch cannot find a matching rule for a incoming packet, it sends a *packet-in* request to the controller (*i.e.*, it asks the controller what should be done with the packet). Besides defining the exchange of messages between switches and controllers, OpenFlow also specifies the components that an OpenFlow-enabled switch must have, such as a *Flow Table*, which is a structure to store the rules installed by the control plane.

Since its first release in 2009, the OpenFlow protocol has received several updates to include new features [3]. As examples, version 1.1 includes the fast-failover functionality, which allows the specification of alternative actions in case of a link failure, and version 1.3 allows the inclusion of meters in switches, capable of discarding or remarking packets that exceed an established throughput threshold.

The SDN concept has attracted the attention of researchers and companies due to its potential to provide flexibility and to facilitate the management of communication networks. The separation of the control and data planes enables the centralization of traffic control and allows new protocols and applications to be implemented in a network without the need to modify forwarding devices. Its use has shown gains in terms of resource utilization [17] and in Quality of Service (QoS) provisioning [10]. Consequently, it is natural to seek the same advantages of this paradigm in wireless networks.

However, the OpenFlow protocol and the SDN architecture were originated with wired infrastructures in mind. Access points and wireless clients introduce several new challenges — particularly those inherent from the shared nature of the wireless medium that can negatively impact network performance, such as interference between transmitters and hidden terminals. Therefore, changes to the original paradigm were needed to improve the management potential of the control plane and the optimization of resource utilization in wireless networks. Several works [15] have extended SDN and OpenFlow by adding specific functionalities that allow the fine tuning of the parameters of the transmitters (*e.g.*, transmission power control, operation channel switching, temporary radio deactivation) in a wireless network.

Chapter 3

Related Work

In this chapter, we highlight the related works in the literature that address infrastructure-driven handover, access point virtualization, and seamless client mobility. We first review the proposals that try to circumvent shortcomings related to client-based handover, then we extend the discussion introducing works that propose access point virtualization solutions to improve the overall performance of Wi-Fi networks. At the end of the chapter, we present a comparison between the highlighted solutions.

3.1 Client-side virtualization

In traditional IEEE 802.11 networks, the wireless client typically decides to handoff when it begins to detect the degradation of the Received Signal Strength Indicator (RSSI) from the AP currently providing the connection [6]. During the handoff process, the communication between a host and the network may be interrupted for several milliseconds until the client associates with a new access point. Besides, the selection of a new AP during the handoff is made to the benefit of the station itself, *i.e.*, the wireless client does not make decisions aimed at delivering performance gains to the network.

To minimize the overload caused by client handoffs, the authors in [18] propose a trigger-based dynamic load balancing mechanism for WLANs using station-side virtualization. In their work, the wireless NIC of the station is virtualized, enabling the client to be connected to multiple access points at the same time. Their virtualization technique is based modifications on MAC layer of the wireless card driver, allowing it to keep state information for each associated AP. From that point, clients and APs send status messages to a server that will monitor the traffic conditions continuously. Instead of recalculating the optimal topology periodically, their solution recomputes an optimal association topology only when a bottleneck is detected by their monitoring mechanism, based on the current bandwidth usage of the nodes. After that, an AP selection algorithm is executed to balance the resource utilization of the network. Then, control messages — proposed by the authors — are sent to inform the client from which AP it should transmit or receive data (*i.e.*, send or receive packets through one of the client's virtual interfaces). The authors also evaluate their solution — using the ns-3 simulator [1] —, reporting an increase of 11% in the aggregated throughput of the network, in comparison to the traditional IEEE 802.11 AP selection behavior.

Despite the performance benefits presented in their work, new features need to be included in wireless stations, increasing the deployment cost of the network. Also, even if these modifications were economically feasible, many current Wi-Fi devices would not benefit from the throughput gains, since they would not be able to connect to multiple APs simultaneously. Thus, client-side or protocol modifications are not feasible or desirable in many scenarios.

3.2 Network-only modification

Another strand of research focuses on delivering improvements by proposing the modification of the behavior or architecture of wireless networks without requiring changes to wireless clients.

In [23], the authors propose DenseAP, an architecture for enterprise Wi-Fi networks. Their primary focus is to design a practical solution that enhances the performance of dense IEEE 802.11 scenarios, leaving current wireless clients unchanged. In their solution the APs also do not expose their SSID, requiring the clients to send probe requests during their active scanning phase. Each probe request that is received by the APs is sent to a centralized controller, which selects the AP that will reply and provide network connection to the client, *i.e.*, the AP to which the station will be associated. The selection is made by an algorithm that estimates free air time and expected transmission rate, based on periodic reports that the APs send to the DenseAP controller, containing current traffic, signal, and channel conditions and a list of associated stations. The controller is also capable of handing clients off from one AP to another, aiming at overload reduction, which is accomplished through the execution of a load balancing algorithm. In this case, the source AP sends a disassociation frame to the client, and the destination AP replies to the upcoming probe request. Although their evaluation has shown throughput gains in dense scenarios, the clients undergo a significant disruption time — around 1.5 seconds — during the handoff for load balancing purposes, since the station still needs to scan the medium and re-associate with the target AP. Also, new stations may experience significantly longer delays to associate with an AP, considering that the controller needs to collect and calculate the signal strength of each probe request sent by each client.

3.3 Access Point Virtualization

Several works utilize the concept of access point virtualization to provide improvements for wireless networks.

Grunenberger and Rousseau [13] proposed a mechanism to enable seamless client mobility on IEEE 802.11 networks, in which the handoff between APs is entirely transparent for the stations. This is achieved by transferring the handoff responsibility to the infrastructure through the virtualization of APs. In their proposal, each station has its own VAP, physical APs can host multiple VAPs, and when a client moves, its VAP should be moved together to a nearby AP, in a way that the association between the client and the VAP is maintained. When the signal strength of the client on the current AP is weaker than the signal received by another AP in the network, control messages are exchanged between both APs to transfer the VAP information. Their solution does not require any modification to the clients. However, it requires that every AP — and consequentially every client — operates in the same channel, which could aggravate interference problems in dense scenarios.

In [28], the authors propose Odin, an SDN framework to facilitate the deployment and the programming of services and functionalities in enterprise WLANs. Their solution also utilizes AP virtualization as an abstraction to simplify the development of Odin applications, suppressing the concerns about changes in the endpoint link, *i.e.,* the link between a client and the network. The developed applications are executed on top of the Odin Master, which is an application on top of an OpenFlow controller. The authors highlight that their framework supports seamless mobility, since the infrastructure is able to handoff clients without requiring re-associations. Similarly to other above-mentioned proposals, Odin utilizes RSSI as a metric to start the handoff procedure and does not require any client-sided modifications.

Ethanol [22] is an architecture that extends the SDN paradigm to allow global control of QoS, client mobility, and security for dense IEEE 802.11 networks. The Ethanol controller manages features of the APs, such as client association, VAPs and current state of links, and can be executed on the cloud or in a computer in the wired part of the network. The controller also supports the OpenFlow protocol, which allows it to control OpenFlow-enabled switches. The Ethanol agent introduces a management interface for APs, providing an Application Programming Interface (API) for wireless link control and for QoS definition in the wired ports. The agent is executed on modified commodity wireless routers and receives commands specified by the Ethanol protocol from the controller via a secure connection. The authors also implemented a prototype to evaluate their architecture. Besides providing wired traffic prioritization and reducing Address Resolution Protocol (ARP) overhead in wireless links, their prototype was able to control the association of clients based on current AP load by forcing the station to associate with the AP that contains fewer connected stations, *i.e.,* new association requests are denied if the AP has more clients connected than another AP in the range of the client. Although their control model involves access point virtualization, the work does not detail the virtualization process or if their VAPs are migrated between physical APs. Also, the authors highlight that only part of the functionalities was implemented due to time restrictions and hardware and software constraints. The absence of additional exploration of the proposed functionalities in their prototype might obscure some limitations of their work.

With the goal of improving client mobility and the ergonomics of Wi-Fi networks, the authors in [27] have developed a solution that allows clients to deploy virtual OpenFlow APs and virtual SDN controllers on the fly. The authors highlight that, even though multiple VAPs share the same physical AP, their solution provides complete isolation between instances, allowing them to operate independently in the same wireless NIC. Despite the claimed improvements in scalability, security and flexibility, they do not benchmark the proposed architecture to evaluate the real benefits of the solution in handover scenarios.

Zeljković et al. [29] propose an SDN-based solution that uses VAPs to allow the handover of clients. Rather than wait for the deterioration of the network, their solution estimates the performance of the nodes and hands clients off proactively, preserving the current QoS. In this work, the handover decision is based on an algorithm that takes into account several metrics, such as RSSI, current traffic load, AP capacity and client mobility. Their evaluation reveals a reduction in the number of handovers and an increase in average throughput in comparison to a reactive algorithm that triggers the handover when the RSSI value drops below a threshold and MAX RSSI – a handover algorithm that keeps a station associated with the AP with the highest RSSI.

In dense scenarios, better performance can be achieved when nearby APs operate in orthogonal channels, due to the reduction of collisions caused by simultaneous transmissions in overlapping frequencies. However, none of the aforementioned works addresses the use of APs operating on different channels, which is a limiting factor in such scenarios: either all APs would have to operate in the same channel — which would aggravate the problems of the shared medium — or the handover of stations would have to be restricted to a subset of target physical APs.

3.3.1 Multichannel support

In [7], the authors seek to solve the above-mentioned channel limitations by introducing the idea of Multichannel Virtual Access Points, which uses the CSA mechanism and a wired infrastructure for inter-AP message exchange. In their proposal, the migration of a client occurs when a AP detects that the client's signal is below a threshold. Then, the AP sends scan requests to neighbour APs, which change their channels temporarily to the station's channel and reply to the requests with the perceived signal strength of the station. Afterward, the current AP migrates the VAP to the physical AP with the highest reported signal and makes the station change its frequency to the channel of the new AP. Despite allowing client handover between APs operating in different channels, the lack of global view and centralized control limits the scope of the handover decisions, which are solely based on the RSSI.

CloudMAC [9] is an OpenFlow based architecture in which virtual access points are deployed in a cloud computing environment, providing flexibility and scalability of IEEE 802.11 networks. In this work, physical APs only forward MAC frames, whereas tasks such as management frame creation and MAC data processing are held by VAPs in a virtual machine. The authors also briefly mention the possibility of using CSA for the AP exchange process, without further exploring the utilization of this mechanism in their solution.

Our work shares some similarities to the BigAP [30] proposal, like the use of the CSA IE to enable multichannel station handover and the employment of a controller to coordinate the migration process. However, their work considers the utilization of one single BSS throughout the access points of the whole network, requiring nearby APs to operate on different channels to avoid problems inherent of the BSS replication (*e.g.*, duplicates due to two APs receiving a frame from the station and forwarding it through the network). This requirement might lead to performance degradation if other unmanaged nearby networks heavily utilize channels that must be used by one of the managed APs to cover a certain area. Their work also requires APs with two wireless interfaces, one to serve as an AP and another running in promiscuous mode to monitor the wireless activity of nearby devices, which increases the cost of deployment of their solution and can be a limiting factor for scenarios that focus on low-budget networks.

3.4 Overview

Despite the many efforts to reduce the handover overhead and increase the performance of WLANs through the virtualization of network components, there is still need for a lightweight solution that is able to operate in commodity hardware and fully support multichannel handovers, without significant restrictions in frequency utilization. Also, delivering global view and control through an abstraction interface improves network flexibility, allowing the employment of different handover algorithms that rely on various network state information.

Table 3.1 presents an overview of the aforementioned proposals and compares them according to the following characteristics:

- *Client-side modification*: highlights whether or not the solution requires software and/or hardware modification in the wireless client;
- *Network-based Handover*: highlights if the proposal enables client handover initiated by the network infrastructure;
- *Virtualized Access Point*: highlights if the proposal utilizes the concept of virtual access point;
- *Controller-based*: highlights if the proposal relies on a controller entity to manage the network;
- *Multichannel Support*: highlights if the proposal supports the handover of clients between APs operating in different channels; and
- *Channel Restriction*: for proposals with multichannel support, highlights if the solution restricts channel utilization.

Table 3.1: Comparison between related work and the proposed solution.

Work	Description	Client-side modification	Network-based Handover	Virtualized Access Point	Controller based	Multichannel Support	Main limitation
[18]	Minimizes client handoff overhead by virtualizing the station NIC, allowing it to be connected to multiple APs.	Yes	Yes	No	Yes	No	Requires station-side modifications
[23]	Increases the performance of dense enterprise WLANs by handing over clients and estimating free air time and transmission rate.	No	Yes	No	Yes	No	No multichannel support
[13]	Proposes an architecture to enable seamless RSSI-based client mobility through inter-AP control message exchange.	No	Yes	Yes	No	No	No multichannel support
[28]	Proposes a framework to ease the development and employment of network services by providing link abstractions at the network edge.	No	Yes	Yes	Yes	No	No multichannel support
[22]	Provides an architecture that extends the SDN paradigm to improve QoS, client mobility and security in dense WLANs.	No	Yes	Yes	Yes	No	No multichannel support
[27]	Improves the mobility and flexibility of Wi-Fi networks by deploying VAPs and virtual SDN controllers on-the-fly.	No	No	Yes	Yes	No	No multichannel support
[29]	Proposes a SDN-based proactive handover solution based on multiple network metrics, such as current load, RSSI and client mobility.	No	Yes	Yes	Yes	No	No multichannel support
[7]	Proposes the concept of Multichannel Virtual Access Point to enable the handover of stations between APs operating in different frequencies.	No	Yes	Yes	No	Yes	No central controller
[9]	Proposes an architecture in which management frame creation and MAC data processing are held by VAPs deployed in a cloud environment.	No	Yes	Yes	Yes	Yes[a]	Higher delays for MAC frame processing
[30]	Aims at improving mobility and QoE by proposing an architecture in which a single BSS is replicated throughout all access points in the network.	No	Yes	Yes[b]	Yes	Restricted	Nearby APs must operate in distinct channels
Our Proposal	Enables seamless multichannel handover of stations, initiated by the network, providing abstractions that allow applications to define their own handover decisions.	No	Yes	Yes	Yes	Yes	Limited number of VAPs being executed by one physical AP

[a]The CloudMAC solution only cites the CSA as an enabling technology to allow the station handover between APs that operate in different channels.
[b]Here we considered the single BSS to be a single a VAP, since the BSS information is replicated in all physical access points.

Chapter 4

A Lightweight Virtualization Solution

Our work proposes a solution that allows the station handover to be initiated by the network infrastructure, while being totally transparent from the client's point of view. For this, we rely on the AP virtualization and SDNs paradigms. Besides, by using the CSA as an enabling mechanism, our solution supports handover between APs operating on different channels.

In this chapter we present the architectural components of our virtualization solution and how the CSA mechanism is incorporated to enable multichannel handover of clients. We also detail some aspects of the implementation of the prototype used for the evaluation.

4.1 Architecture

The architecture of our multichannel virtualization solution consists of three main components:

- the virtual APs;
- the physical APs; and
- the controller.

The concept of AP virtualization can be found in the literature to represent different ideas for an entity that performs tasks or assumes the role of an AP. In our proposal, a VAP consists on encapsulating the information of a BSS and its associated station to allow the portability and transfer of this data between different physical APs. We call physical APs the nodes that host one or more VAPs, connecting multiple clients wirelessly to the network. Different from traditional IEEE 802.11 networks, in our proposal the BSS contained in a VAP accepts the association of at most one station. Therefore, each connected client has their own VAP. This allows VAP migrations to be performed without interfering with other stations connected to the network.

The controller, connected to physical APs through a wired network, is an entity capable of requesting state and usage information of the access points and their associated clients. Furthermore, the controller is responsible for orchestrating the process of VAP migration by exchanging control messages with the physical APs. Also, it has an interface that provides functions and AP-state information to external applications. Figure 4.2 presents an overview of the architecture of the solution.

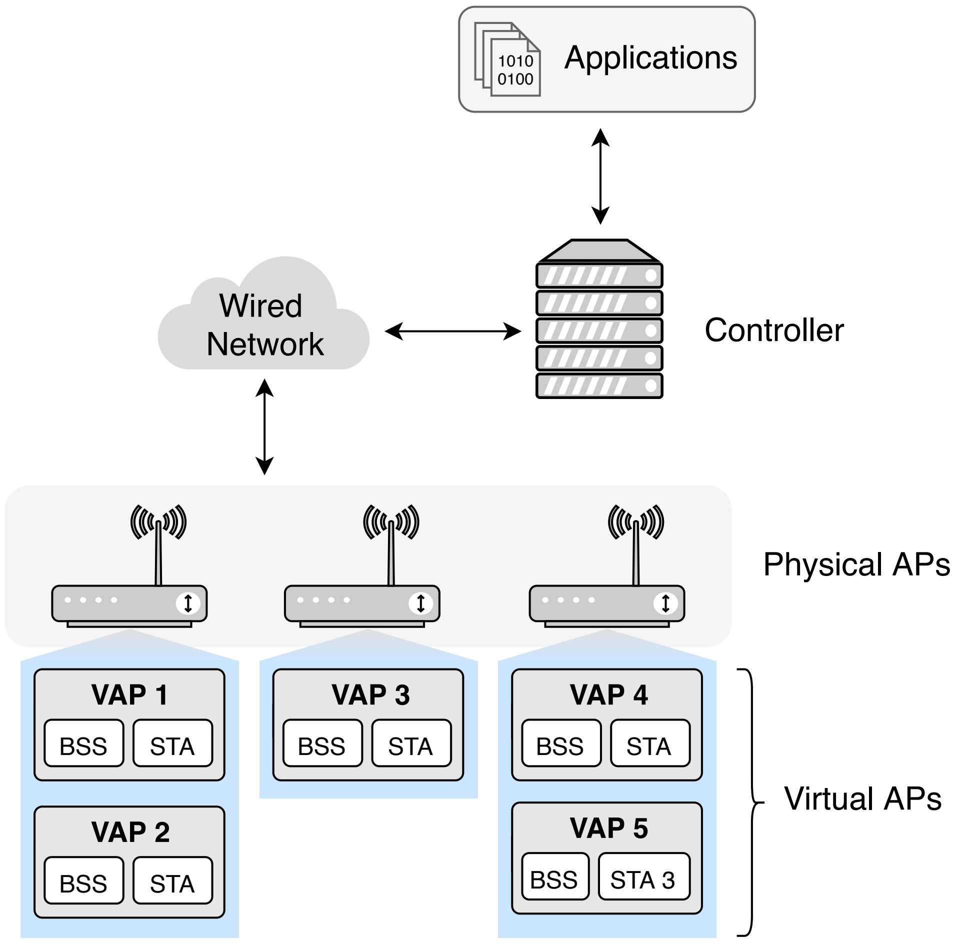


Figure 4.1: Overview of the virtualization solution. Each physical AP hosts one or more VAPs. The controller manages the physical and virtual APs.

It is outside the scope of this work to define in which cases the migration process should be triggered (*i.e.*, the controller does not implement a handover decision algorithm). Instead, the primary purpose of this proposal is to make the handover decision more flexible by establishing an interface for the data plane (physical APs) that provides abstractions for applications running on top of the controller at the control plane. This concept allows VAPs — and, by extension, wireless stations — to be migrated based on different decision algorithms, implemented by these applications using the functions provided by the framework. The objectives of these algorithms may include support for seamless station handoff or load balance between physical APs and between channels.

4.2 VAP Handover

The VAP handover procedure begins when the controller receives a request through its API. It consists of the exchange of control messages between the controller and the source and destination APs — which might be operating in the same or different channels — regarding the information of the target VAP and its associated station. Figure 4.2 provides an example of a VAP migration between two physical APs, orchestrated by the controller.

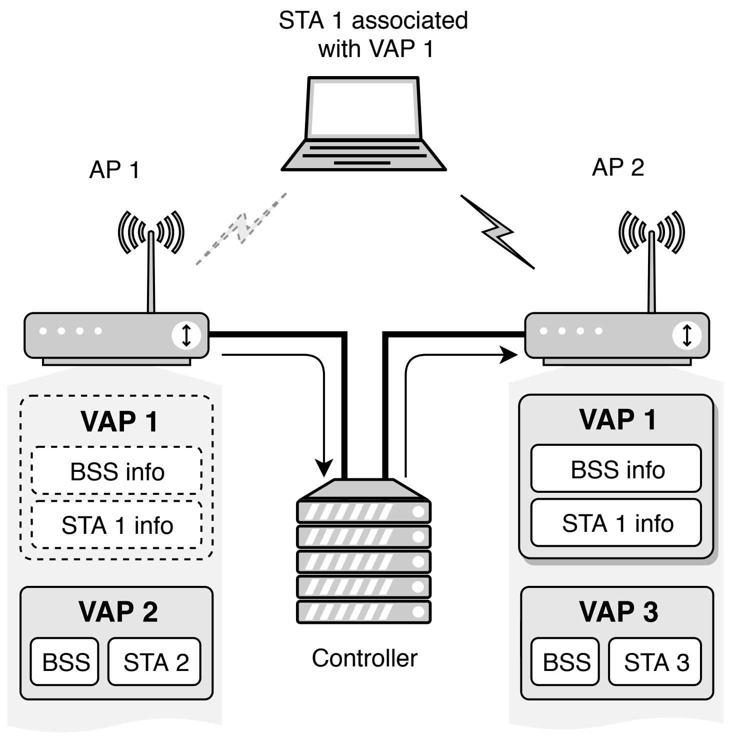


Figure 4.2: Example of VAP migration managed by the controller. The VAP 1 is transferred from AP1 to AP2.

As previously stated, the CSA is a key mechanism to allow inter-channel migration. During the handover procedure involving APs in different channels, the source AP — *i.e.*, the one to which the client is currently associated — must include the CSA IE in the beacons of the VAP to be transferred. The *New Channel Number* is set to correspond to the channel of the destination AP. The goal of this approach is prompting the station to change its operating frequency to match the channel of the new physical AP which will host the VAP after the migration is complete, maintaining the association.

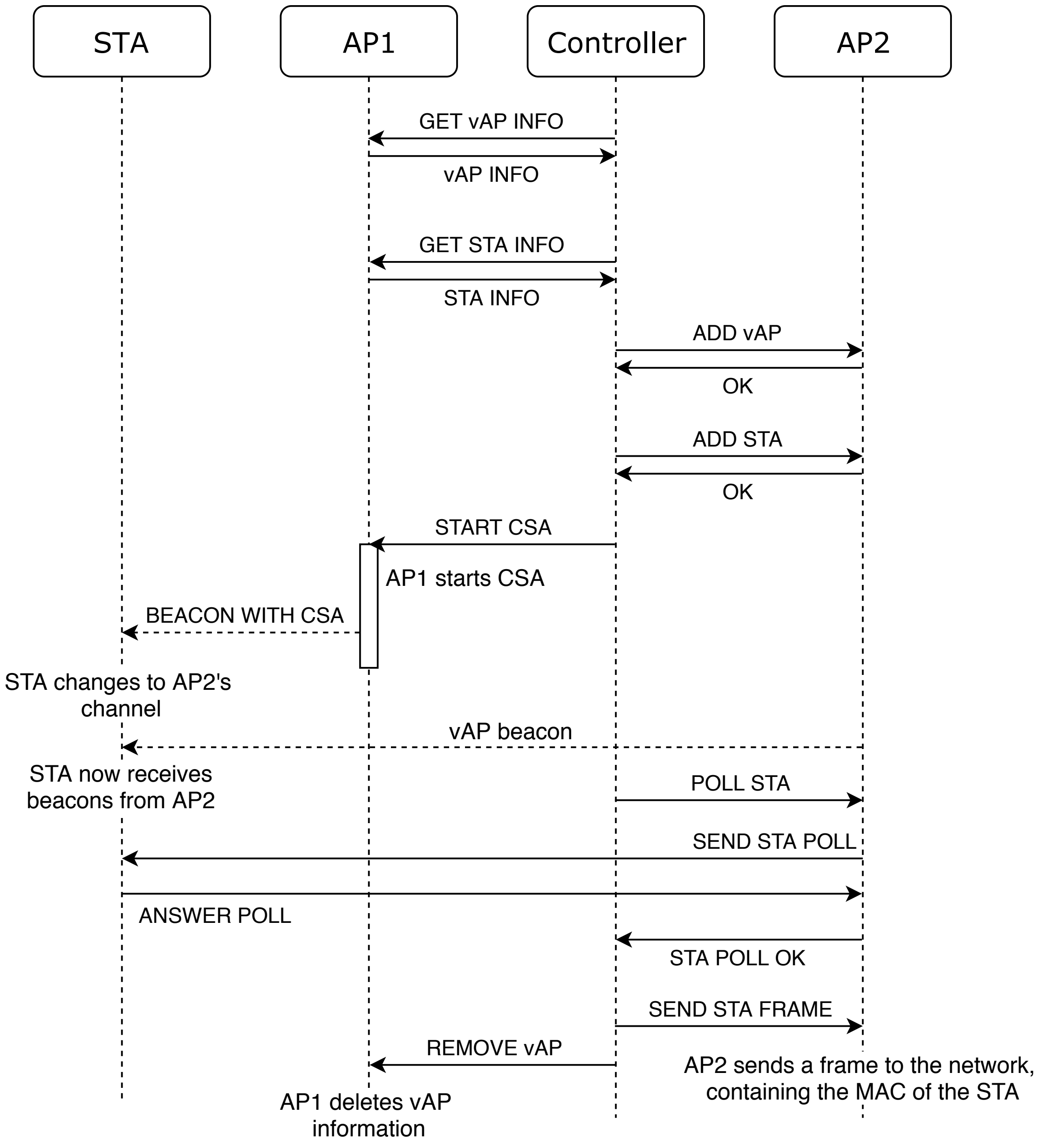


Figure 4.3: Steps of the multichannel virtual access point migration procedure.

The multichannel migration process is performed based on the steps described in Figure 4.3:

1. The controller requests from AP1 the status information regarding one of its VAPs and the station (STA) connected to it;

2. The controller sends the received VAP information to AP2, which creates a VAP[1]

[1]The term VAP is used to address the virtual access point responsible for the station (STA) experi-

with the same configuration of the original VAP;

3. The controller sends a client addition command to AP2 including the STA association information in the newly created VAP;
4. The controller sends a start CSA command to AP1, inducing it to announce a channel switch to the channel of AP2;
5. AP1 starts including the CSA information element in the beacons of the VAP;
6. The STA receives the VAP beacon with the CSA and switches to the channel of AP2;
7. The STA starts using AP2, without disassociating from its VAP;
8. The controller sends a request to AP2, to verify that the STA is connected to the new AP. Upon receiving an answer from the STA, AP2 reports the connectivity success to the controller.
9. AP2 broadcasts the new location of the STA, by sending a frame to the distribution system containing the MAC address of the STA.
10. AP1 deletes the VAP information, upon receiving a removal command from the controller.

Steps 4, 5, and 6 are not required when the destination AP is in the same channel as the source AP.

4.3 Implementation

To better assess the feasibility of our solution, we have developed an implementation for the controller and access points, as well as the northbound and southbound APIs. In this section, we detail the implementation aspects of the software executed by data plane devices, up to the control plane and its interface.

4.3.1 Access Points

In some works, station changes are proposed to improve the performance of wireless networks. However, these changes reduce the feasibility of deploying the solution as

encing the handover.

manufacturers would need to modify the hardware of these devices. In order to maintain compatibility, we aimed for the full interoperability of the proposed solution with existing mobile devices without relying on changes to them.

Access points, in comparison, are more feasible to adapt because they are elements of network infrastructure. However, hardware modifications to these devices are also undesirable. Therefore, another goal was to deploy our solution on devices with commercially available wireless NICs. Also, to facilitate its implantation in resource-constrained APs, our solution must be lightweight, without relying on multiple libraries to perform the required functions.

To fulfill the goals mentioned above, we decided to use the HostAPD 2.7 source code as a base for our AP implementation. HostAPD[2] is an open-source software under the BSD license that allows Wi-Fi network interfaces to be used as access points, as well as providing authentication mechanisms. Its code is highly portable, being supported by multiple platforms and embedded devices, such as OpenWRT-supported APs. In the proposed solution, the VAPs are implemented as multiple BSSs over the same wireless interface, created and managed by HostAPD. However, the HostAPD, as is, does not provide all the functionalities required by our solution.

To allow the transfer of BSS status information and its associated station, new features have been added to HostAPD's source code. By default, the software has native support for channel exchange commands by including the CSA element in the beacons of all VAPs that are on the same wireless interface and by changing the channel of the physical interface. This behavior is undesirable when only one VAP is migrated, since all VAPs will advertise the channel switching. Therefore, modifications were made in the beaconing functionality so that only the VAP to be migrated includes the CSA element in its beacon, avoiding unwanted changes in other VAPs.

During the migration process, the destination AP needs to receive information from the migrating station and register it as associated with its VAP. After that, the AP needs to be ready to receive or transmit data. HostAPD 2.7 allows defining a fake association for a specified MAC address for test purposes. However, this functionality is not enough to simulate the station association, since more hardware-related information is needed to allow the communication between the AP and the station. Therefore, we modified the HostAPD code to allow arbitrary registration of station information (*e.g.*, MAC address, supported transmission rates, and other capabilities), which is otherwise acquired by the

[2]Available at: https://w1.fi/hostapd/

AP during the association procedure, so that no client association is required after the channel switching. Our solution also imposes a relationship of one station to one VAP. However, the maximum number station per BSS is a configurable parameter in HostAPD. Therefore, it was not necessary to modify the code to limit the number of associated stations to one.

Regarding the initialization of the network, in our implementation, each AP starts with an "empty" VAP in which a client might associate. When a station associates with this VAP, the AP instantiate another empty VAP, to allow the association of other stations. This process is repeated until the limit of VAPs that can be held by an AP, addressed in Section 6.1.

To forward wireless traffic to the wired network, we needed to configure a bridge at the physical APs. We accomplished this by developing a script that utilizes the *brctl* command[3], which allows the creation and removal of bridges in Linux-based systems. The script builds a bridge to connect the wired interface to virtual wireless interfaces of each VAP that will be dynamically created or removed by the HostAPD.

4.3.2 Southbound API

Similarly to the implementation for Access Points, we have used the HostAPD Command Line Interface (CLI) as the foundation of our *southbound* API. The HostAPD CLI already provides a plethora of commands that allow the management of HostAPD through user-input commands directly from the provided client-sided application or received via a network socket. Still, we needed to extend it to implement the necessary functionality between the data and control planes of our solution. The performed modifications are detailed below:

- SEND_CSA - This command modifies the behavior of the channel switch command. It causes the AP to inject the CSA IE only in the beacons of the specified BSS, without actually changing the channel of the AP as the default command does.
- ADD_STA_P - This command was implemented to allow the injection of a station in the destination AP by the controller without requiring a new association of the wireless client. The parameters of this command include the association identifier, MAC address, supported rates, and hardware capabilities of the station.

[3]Available at: https://linux.die.net/man/8/brctl

- ADD_BSS - This command modifies the behavior of the default BSS inclusion command to allow the configuration of new BSSs through the transmitted parameters in the command message. The default command only allows the creation of a new BSS upon the prior specification of a configuration file in the storage of the AP, which is pointed by the command.

- Transaction Identifier - A number added to the command messages to distinguish each request sent by the controller. It helps the control plane identify to which request a received response belongs.

4.3.3 Control Plane

For the control plane, we have developed an application using the Java language. Since the Java environment relies on a Java Virtual Machine (JVM) as a middleware between the operating system and the application, Java-based applications can be run in multiple platforms, such as Linux or Windows computers, extending the support of our control plane. Below and in Figure 4.4, we present an overview of the main modules of the controller:

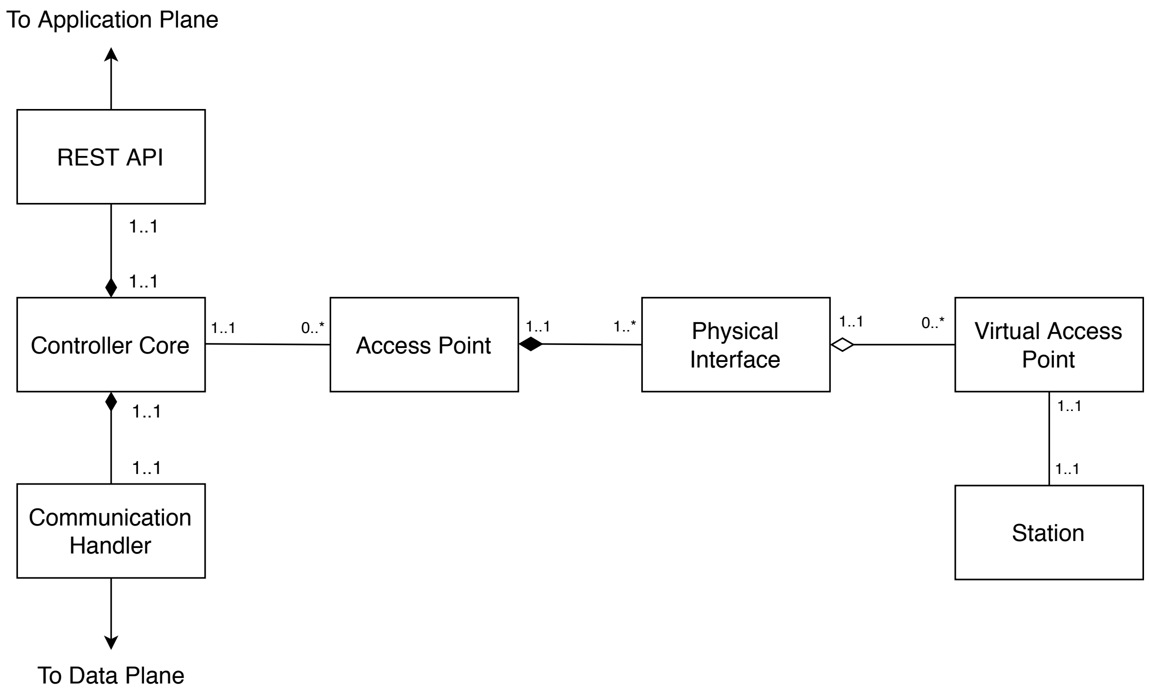


Figure 4.4: Conceptual diagram of the developed controller.

- REST API - Implementation of the Northbound API. It provides information regarding the data plane and migration functions.

- Controller Core - Manages the registered APs, requesting their state information. It is also responsible for handling the VAP migration process.
- Access Point - Object representation of an AP which stores its multiple physical wireless NICs and manages the control interface of each of them.
- Physical Interface - Object representation of a wireless NIC of an AP which stores its hardware information. Each physical interface may manage multiple VAPs.
- Virtual Access Point - Object representation of a VAP that stores information related to its BSS and its associated station.
- Station - Object representation of a station. Stores association parameters, such as the hardware capabilities, supported rates, listen interval, and connection statistics of a wireless client.
- Communication Handler - Manages all the structures needed to send or receive network messages. Builds new requests to be sent by the controller, waiting for correspondent answers. It also processes the events that are sent by the APs.

In our implementation, the controller is able to rollback a VAP migration in case of failure or timeout of the injection requests of the VAP and its station or if the AP fails to send the beacons with the CSA IE.

4.3.4 Northbound API

For the design of our *Northbound* API, we aimed for a relatively simple interface in which other applications could easily interact with our control plane. Then, we decided to implement it using Representational State Transfer (REST), since it is a robust, well-known architecture that allows the interoperability of multiple web-based services. For that, we used the Spring framework[4], which facilitates the development of REST web services.

The developed *Northbound* API allows applications to gather information and send requests to the control plane. Regarding the data plane information provided by the *Northbound* API, it is possible to retrieve, for example, the available wireless NICs of an AP or the number of stations currently being served by each AP of the data plane. Besides these, Table 4.1 shows several station-related data that is available to the application

[4]Available at: https://spring.io/

Table 4.1: Station-related information that are accessible from the application plane.

Information	**Description**
RX Packets	Number of packets received by a station.
TX Packets	Number of packets transmitted by a station.
RX Bytes	Number of bytes received by a station.
TX Bytes	Number of bytes transmitted by a station.
Supported Rates	Data rates supported by the station.
Inactive Time	Time in milliseconds in which the station is inactive.
Signal	Signal strength between the station and the AP.
RX Rate	Link bit rate used for data reception.
TX Rate	Link bit rate used for data transmission.
Connected Time	Time in which the station is connected to the AP.

plane. With these data, an load balancing application can, for example, derive the total load of an AP by combining the data from each station that the AP is serving and handover clients to better distribute the load of the network.

The application plane can send requests and receive data via simple Hypertext Transfer Protocol (HTTP) methods with the specified Uniform Resource Identifier (URI), as shown in Table 4.2.

Table 4.2: Commands supported by the REST API of the controller.

Command	**Description**	**HTTP Method**	**URI**
Register AP	Registers an AP to be managed by the controller, given its IP address and port.	POST	/register/ap/<ap-id>/<ap-ip>/<ap-port>
Unregister AP	Unregisters an AP from the control plane, given its identifier.	DELETE	/ap/<ap-id>
Get all APs	Retrieves information of all registered APs, including their NICs, VAPs and associated stations.	GET	/ap/all
Get AP	Retrieves information of a specified AP, including its NICs, VAPs and associated stations.	GET	/ap/<ap-id>
Get VAP	Retrieves information of a specified VAP, including the associated station.	GET	/vap/<vap-id>
Create VAP	Creates a new default VAP at the specified NIC of an AP.	POST	/create/vap/at/<ap-id>/<interface>
Delete VAP	Removes the specified VAP from the AP.	DELETE	/vap/<vap-id>
Migrate VAP	Migrates the specified VAP from its source AP to the destination AP.	POST	/migrate/<vap-id>/from/<source-ap-id>/to/<destination-ap-id>
Migrate VAP to NIC	Migrates the specified VAP of the source AP to the specified NIC of the destination AP.	POST	/migrate/<vap-id>/from/<source-ap-id>/to/<destination-ap-id>/at/<interface-name>

Chapter 5

Evaluation

In this chapter, we present the experiments held during the development of this work to validate and increase our understanding concerning the proposed solution. In each experiment, we briefly discuss its purpose and methodology, describing the scenarios and tools that we utilized. Then, we present the obtained results and the conclusions that can be drawn from them.

Being the CSA a crucial mechanism of the proposed multichannel virtualization solution, we start our evaluation with an analysis of the performance of this mechanism and of the behavior of several client devices when they receive a frame with the CSA IE. Later, we present a benchmark of the proposed solution.

5.1 CSA performance assessment

To deepen our knowledge about the impact of the CSA mechanism, we held an experiment to compare the performance of a channel change by an AP with and without advertising it the associated stations.

For this analysis, we created a simple topology as shown in Figure 5.1 in which a laptop with an embedded Wi-Fi card uses HostAPD to create an access point. The laptop also runs DNSMasq[1] to assign IP addresses to a Raspberry 3 Model B running Kali Linux 2018.4 with Nexmon acting as a station. The *iperf*[2] tool was used to measure TCP throughput between the station and the AP, which used the IEEE 802.11b mode. In this test, the AP performed a channel switch after 30 seconds. In the first scenario, the CSA mechanism was used with a *Channel Switch Count* number of 5 TBTTs, while

[1]DNSmasq is a DNS forwarder and DHCP server software.
[2]Available at: https://sourceforge.net/projects/iperf2/

in the second scenario, the AP simply changed its channel without any kind of warning to the station.

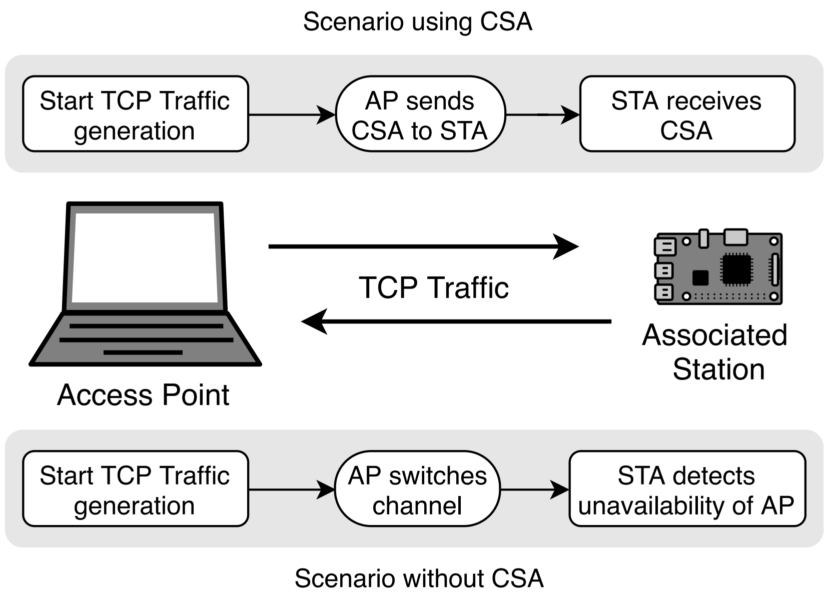


Figure 5.1: Scenarios used to compare a channel switch with announcement versus without any warning to the station.

The obtained results are shown in Figure 5.2. In the scenario without CSA the wireless station had to detect the absence of the AP in the original channel, perform a new scan to find out the new channel and, finally, request a reassociation with the AP. All that process caused the network throughput to remain at zero for almost 8 seconds, followed by a slow recovery, as can be seen in Figure 5.2 (likely due to the congestion control algorithm of Transmission Control Protocol (TCP) severely decreasing its congestion window due to the burst of losses during the disconnection period). With CSA, even though the connection manifested a momentary decrease in throughput during the channel switch of the station, the downtime was significantly lower, as well as its effect on TCP compared to the previous scenario.

This result emphasizes that the CSA mechanism considerably reduces the delay for channel *rendezvous*. Consequently, when applied to the handover of stations between access points in different channels, the inclusion of CSA IE might also reduce the connection reestablishment delay after the station handover process.

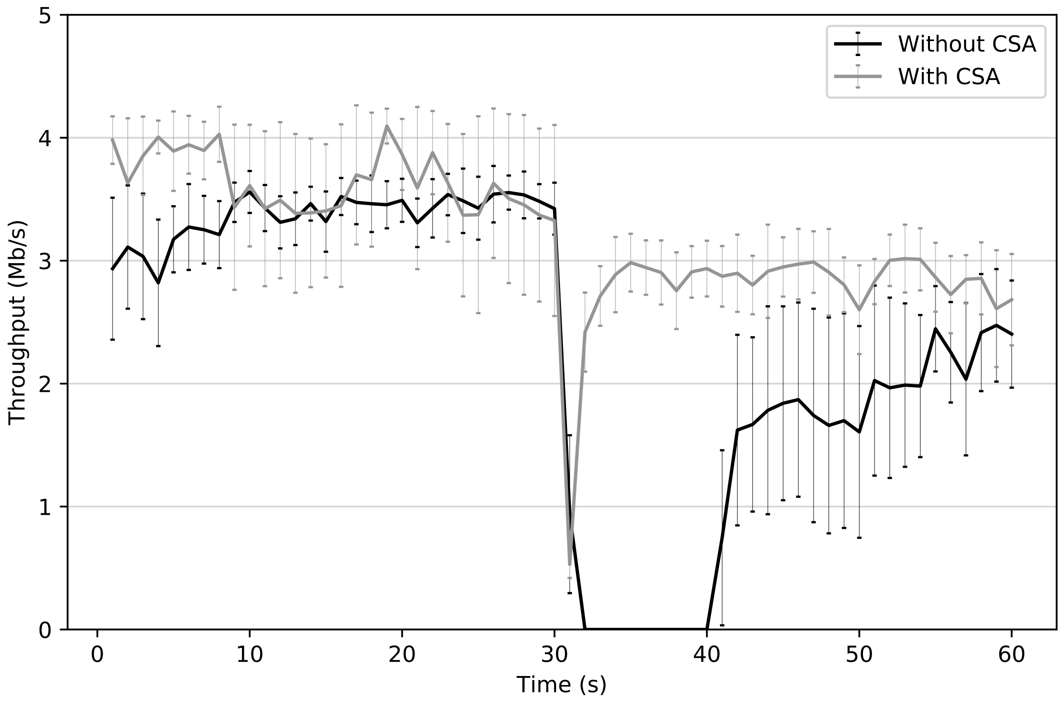


Figure 5.2: How the throughput measured by a station is affected by the AP switching channels with and without the CSA mechanism. The confidence interval is based on 10 executions with a confidence level of 95%.

5.2 CSA client behavior

The IEEE 802.11 specification establishes a series of rules related to the transmission of frames with the CSA IE. However, the standard does not necessarily state that a station must change its frequency to the advertised channel upon receiving a frame with the CSA IE. According to the standard, wireless stations might prefer to handoff to another BSS instead of following the AP to the new advertised channel.

In the context of the proposed solution, such flexibility in the reaction to a CSA could lead to a disassociation of the station during the migration of a VAP, increasing the time of disconnection. Therefore, we conducted an analysis to investigate the behavior of wireless devices made by popular manufacturers. Our goal was to verify if these devices follow the AP in the channel switch when they receive the CSA IE and if they respect the transmission restriction imposed by the AP during the announcement period.

In this scenario, we used a laptop executing Ubuntu 18.04.02 LTS, HostAPD, and DNSMasq. The *ping* command was used to generate traffic between the AP and the devices. The AP used the 2.4 GHz band and performed frequency changes between channels 1 and 6 using the CSA mechanism. A higher *Channel Switch Count* number of

Table 5.1: Behavior of computers and laptops with different hardware and operating systems upon receiving a Channel Switch Announcement IE.

Device	**Wireless Card**	**Driver**	**Operating System**	**Reassoc.**	**Blocks TX**
Acer Aspire E1-471-6404	Qualcomm Atheros AR9485	ath9k	Ubuntu 18.04.2 LTS	No	No
Dell Inspiron i14-5448-B30	Intel Wireless AC 7265	iwlwifi	Ubuntu 18.04.2 LTS	No	Yes
		netwtw04.sys	Windows 10 (1809)	Yes	Yes
Dell Inspiron i15-7572-A30S	Qualcomm Atheros QCA6174	ath10k_pci	Ubuntu 18.04.1 LTS	No	Yes
		qcamain10x64.sys	Windows 10 (1803)	Yes	No
Dell Inspiron 13-7359	Intel Dual Band Wireless AC 3165	iwlwifi	Ubuntu 18.04.02 LTS	No	Yes
		netwtw04.sys	Windows 10 (1809)	Yes	Yes
Raspberry Pi 3 Model B	Broadcom BCM43438	brcmfmac	Kali 2018.4 with Nexmon	No	No
			Raspbian 9	No	No

Table 5.2: Behavior of different smartphones and tablets upon receiving a Channel Switch Announcement.

Device	**Operating System**	**Reassoc.**	**Blocks TX**
Ipad Air 2 (A1566)	iOS 12.3.1	No	Yes
Samsung Galaxy S10 (SM-G973F)	Android 9	No	Yes
Samsung Galaxy Tab A (SM-P585M)	Android 8.1	Yes	No
Motorola Moto G4 Play (XT1600)	Android 7.1.1	Yes	No
Samsung Galaxy S6 (SM-G925F)	Android 7.0	No	No
LG Optimus L5 (E612F)	Android 4.1.2	Yes	No
Samsung Galaxy Tab 7.7 (GT-P6800)	Android 4.1.2	Yes	Yes

200 TBTTs was used to highlight the client's blocking behavior — since, with a very small number, the transmission restraint might not be perceived — and to avoid the case in which the station was not aware of the channel switch due to losses of beacons containing the CSA IE.

Table 5.1 presents the results obtained with different computers and laptops as clients — using different hardware and software —, while Table 5.2 displays results for smartphones and tablets of popular brands. In both cases, the *Reassociation* column is related to the need for stations to reassociate with the AP — after, possibly, re-scanning the wireless medium — once channel switching is triggered. The *Blocks TX* column shows which devices stopped their transmission when they received a CSA frame with the *Channel Switch Mode* field equal to 1.

None of the analyzed devices preferred to associate with another BSS. However, they exhibited different behaviors concerning the compliance with the transmission blocking command and the need to reassociate with the original AP. In Table 5.1, it can be seen that three mobile computers requested a reassociation with the AP when they were running Windows. It is interesting to note, however, that in the Ubuntu environment, these same devices did not require reassociation and continued to send packets as soon as they switched channels. This suggests that potentially incompatible devices may only require software updates to support CSA without requiring physical modifications to their hardware. Furthermore, in some situations, the device performed a reassociation (value Yes) and also respected the transmission constraint command (value Yes). This behavior implies that wireless stations that recognize the CSA mechanism may still request a reassociation to the AP after channel switching.

5.3 Solution Evaluation

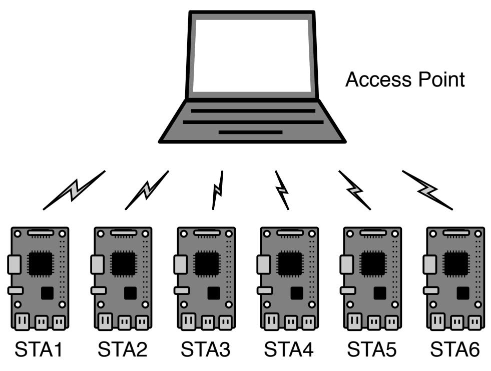
Despite providing new functionalities for IEEE 802.11 networks, it is essential to assess the quantitative impacts of our proposal. Therefore, in the following sections we present experiments that evaluates several aspects of our solution.

5.3.1 VAP Overhead

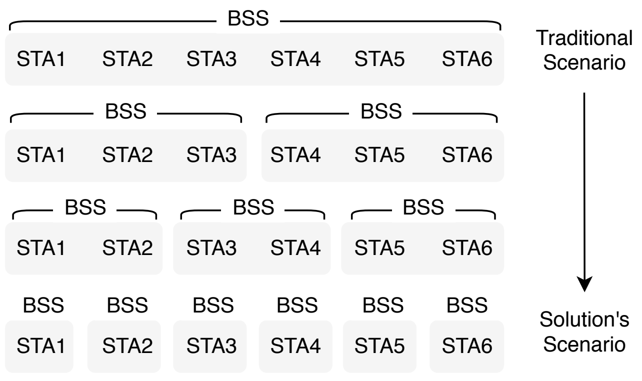
Unlike a traditional IEEE 802.11 network where a BSS can contain multiple associated stations, the proposed solution specifies that each VAP accepts only one station. This specification permits each client to have its own VAP, allowing the VAP migration at any time without interrupting the connection of other stations associated with the AP. Therefore, physical APs must host multiple VAPs, and since each VAP originates the creation of a BSS, the AP NIC must host multiple BSSs at the same time.

The inclusion of extra BSSs to be run by the NIC increases the resource utilization of the shared medium and the AP since a higher number of management frames needs to be processed or transmitted (*e.g.*, rather than sending a single beacon to all associated stations, for each BSS, a beacon will be sent to the associated station). An experiment with multiple stations was held to analyze the performance impact of increasing the number of BSSs hosted by an AP.

For this experiment, we used 6 Raspberry Pi 3 Model B as stations and an Ubuntu laptop with an Atheros AR9485 NIC as an AP. All of them were operating in IEEE 802.11g mode. First, we started with all six stations associated to only one BSS. Then, in each step of the experiment, the number of BSSs was increased in a way that the number of associated stations would be equally divided among them (*i.e.*, each BSS would contain the same number of clients). With the goal of analyzing possible benefits in reducing the amount of beacons sent by each BSS of the AP in a period of time, we also varied the time interval between beacon transmissions for each step of the experiment. The *iperf* tool was used in each station to generate TCP traffic from the clients to the network. The Network Time Protocol (NTP) was used to synchronize the clocks of stations and the AP. The scenario used for this evaluation can be seen in Figure 5.3.



(a) Topology


BSS
STA1 STA2 STA3 STA4 STA5 STA6
Traditional Scenario
BSS BSS
STA1 STA2 STA3 STA4 STA5 STA6
BSS BSS BSS
STA1 STA2 STA3 STA4 STA5 STA6
BSS BSS BSS BSS BSS BSS
STA1 STA2 STA3 STA4 STA5 STA6
Solution's Scenario

(b) Station association map in each step

Figure 5.3: Scenario for the evaluation of the overhead caused by multiple BSSs in a single physical AP.

In this experiment, each step was repeated 60 times and, at each run, the throughput experienced by the stations was recorded for 50 seconds. Then, the average of the observed values was calculated. Figure 5.4 shows the average and aggregate throughput experienced by each station for each configuration.

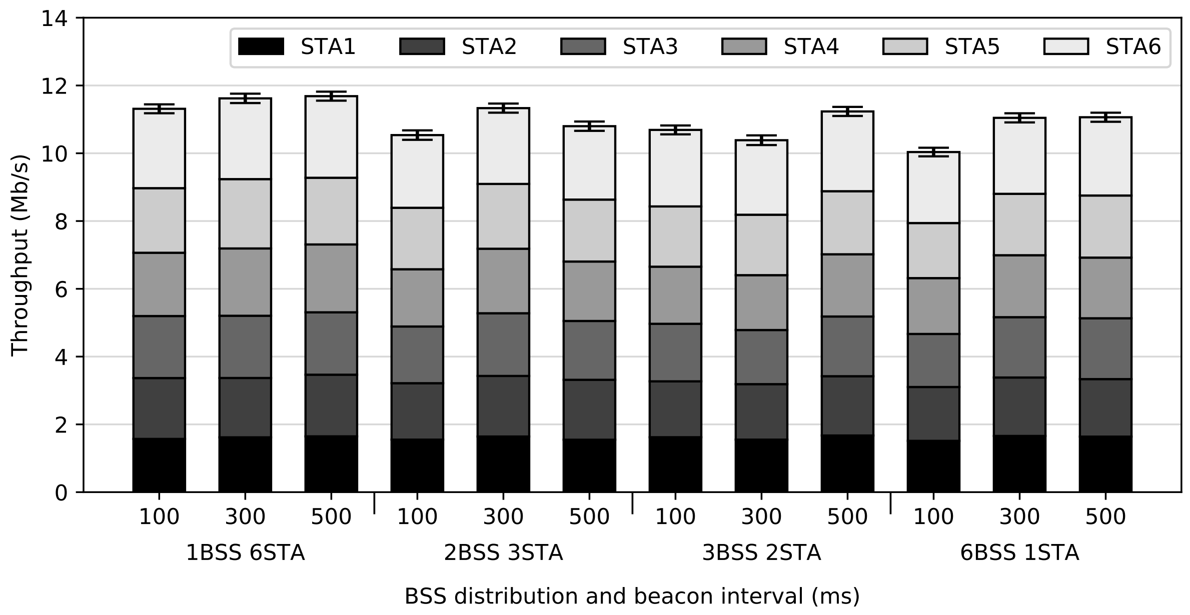


Figure 5.4: Comparison of the average throughput of each station. The confidence interval is based on a confidence level of 95%.

Comparing the configurations with beacon intervals of 100 ms, one can observe that the aggregated throughput decreased when the number of BSSs increased, which would corroborate the idea that performance degrades with the number of BSSs per AP. However, with a beacon interval of 300 ms, for instance, this behavior is not observed since the throughput in the last configuration is higher than in the third one, even with the double of BSSs being executed by the AP.

For the first and the last BSS configurations, increasing the transmission interval of beacons resulted in a throughput increase. Still, the same behavior is not observed for the second and third configurations. Also, despite STA6 achieving slightly higher throughput, the results show that the achieved throughput for each BSS configuration is somewhat fairly divided between the six stations, with a Jain's fairness index no worse than 0.98, as shown in Figure 5.5.

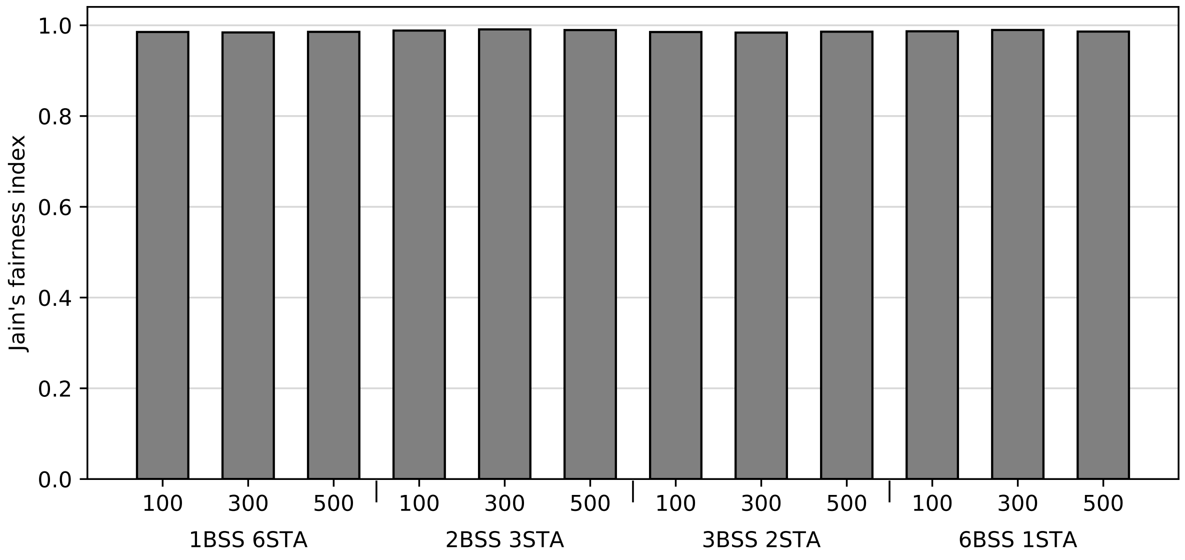


Figure 5.5: Jain's fairness index of each BSS configuration of the experiment.

As a conclusion of this experiment, although the first configuration achieved slightly higher throughput, the increase in the number of BSSs did not cause significant performance degradation. The experiment attests that the proposed solution is capable of achieving throughput rates comparable to commonly used WLAN scenarios while providing greater flexibility for the management of IEEE 802.11 networks.

5.3.2 Handover Benchmark

The station handover phase performed by the network is a key point of the proposed solution. In this section, we present experiments that measure VAP migration performance in relation to four different metrics: Round Trip Time (RTT), throughput, outage time, and packet loss.

5.3.2.1 Round Trip Time

The handover of stations might delay and disrupt network connections. In time-based applications, such as VoIP and video conference, these latency issues may severely deteriorate the Quality of Experience (QoE) of users. Therefore, we begin the handover evaluation investigating the impact of multiple VAP migrations toward the RTT of an ongoing connection.

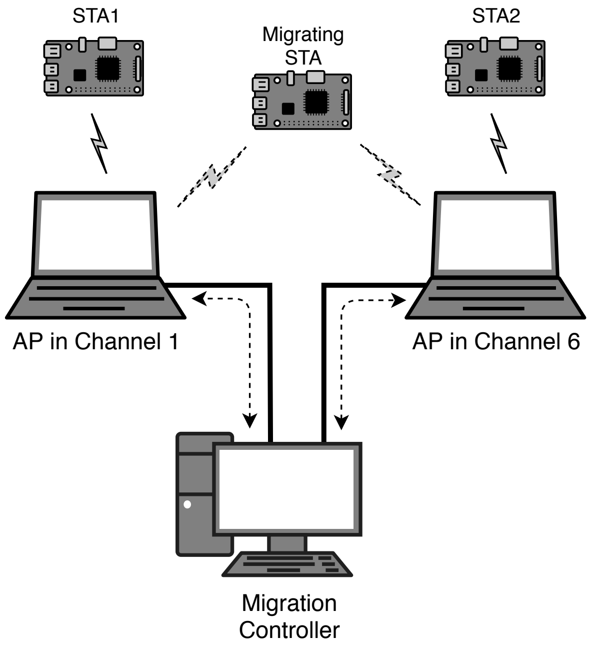


Figure 5.6: Scenario used to evaluate the proposed multichannel virtualization solution. It is composed of three wireless clients, two physical access points and one controller.

For the evaluation of the RTT achieved during the migration of a VAP, we utilized three Raspberry Pi 3 Model B as wireless clients, two laptops running instances of the proposed virtualization solution acting as physical access points on channels 1 and 6, and a desktop serving as a controller. Two of the stations were used as baselines of the performance of each channel. They were associated to VAPs operating in each of the two physical access points. The third station and its respective VAP was transferred between the two physical access points ten times during the experiment. The migrations were started arbitrarily by sending requests to the APs, without relying on a decision mechanism. All clients ran the *ping* command to measure the round trip time of the communication between the station and its VAP. We used a *Channel Switch Count* number of 15 TBTTs. An illustration of the scenario can be seen in Figure 5.6.

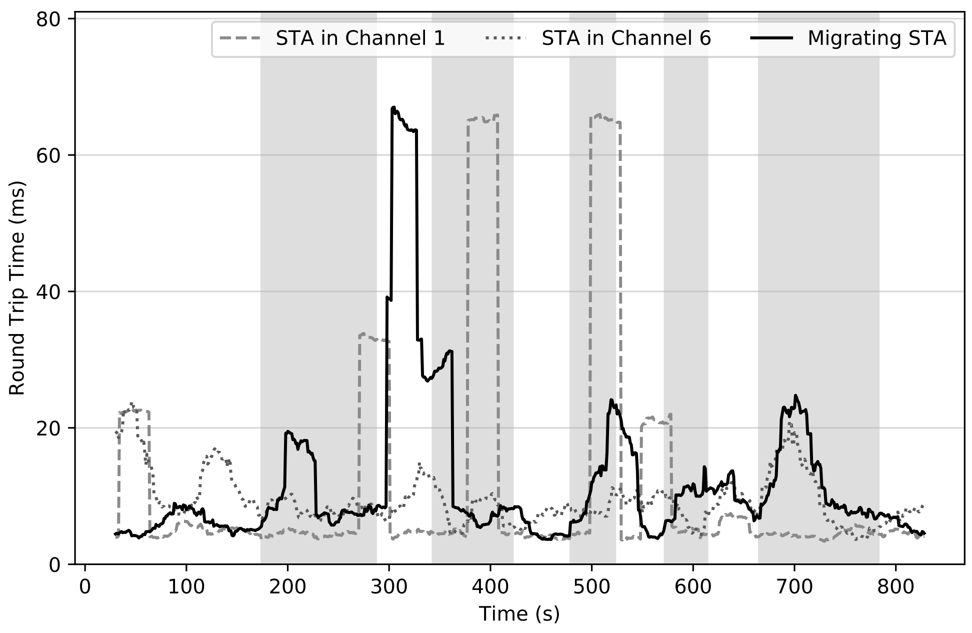


Figure 5.7: Round trip time perceived by each station during the experiment. A moving average with a sliding window of 30 points was used to smooth the data variation. The migrated VAP is moved between physical APs ten times. The greyed out area represents the periods in which the migrating VAP was hosted by the AP in channel 6.

As shown by the results illustrated in Figure 5.7, a peak in the RTT of the migrating station connection can be observed around 300 s when the station changes to channel 1. However, the stationary station in channel 1 experiences a similar behavior around 400 and 500 s, which denotes a performance degradation in channel 1. Thus, throughout the experiment, the migrating station experiences RTTs that are compatible with the stationary stations and the performance of each channel. Therefore, the handover of a client does not introduce significant delays in the communication, confirming that the solution is, in fact, capable of performing multichannel handover without interrupting the connection.

5.3.2.2 Throughput, Outage Time and Packet Loss

In addition to quantifying the impact of the VAP migration process, in the following experiments, we compared the proposed solution to two other handover scenarios. In the first scenario, the AP to which the station is currently associated broadcasts a deauthentication frame. In the second case, the current AP is turned off without transmitting any warning to the station. In both cases the client is forced to associate with the second available AP. Throughout the section, we referred to the prior scenario as *Deauth*, while the second was called *No Deauth*.

For all the scenarios mentioned above, we have used the same network. As illustrated in Figure 5.8, the experiment topology comprises two laptops with Atheros NICs running our modified version of HostAPD acting as a physical AP operating in the IEEE 802.11g mode on channels 1 and 11 of the 2.4 GHz band. An Ubuntu PC was used to execute our controller implementation. We also utilized a desktop to generate traffic in the network. All these devices were connected to each other through a switch. Finally, a Raspberry Pi 3 Model B was used as a station to where traffic is being sent. The *at*[3] command was used to start the experiments at the source and the destination at a specified time after a clock synchronization using the NTP. To generate and capture the transmitted traffic we used respectively the *iperf* and *tcpdump* tools. We repeated the experiment using both TCP and User Datagram Protocol (UDP) as transport protocols for the injected traffic. In both cases, *iperf* was configured to generate at rate of 30 Mb/s at the application layer. In this analysis, the experiment was repeated multiple times and each run lasted 40 seconds. The controller was configured to migrate the station and deactivate the BSS running on the AP in channel 1 after 20 seconds of execution.

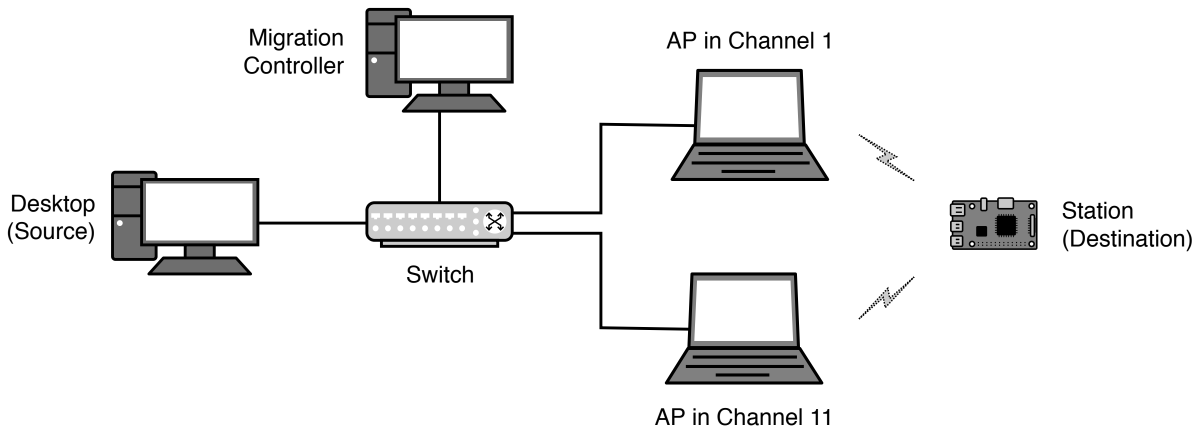
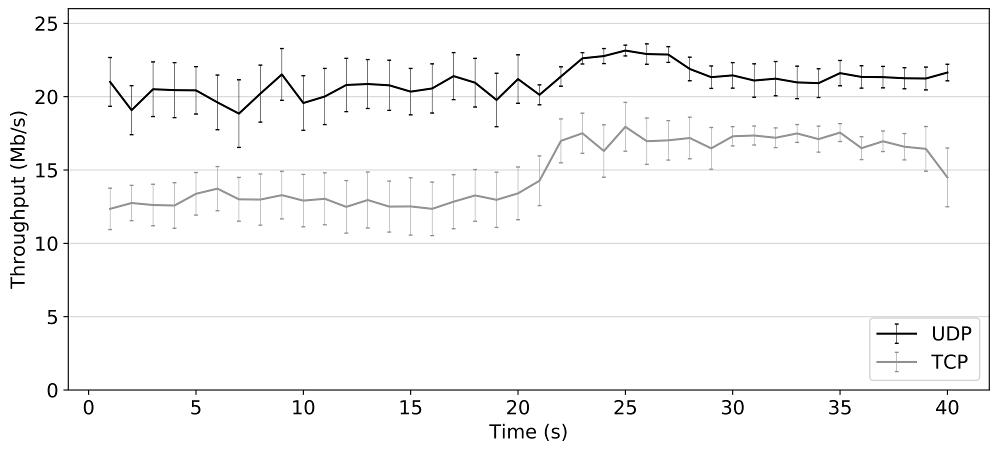


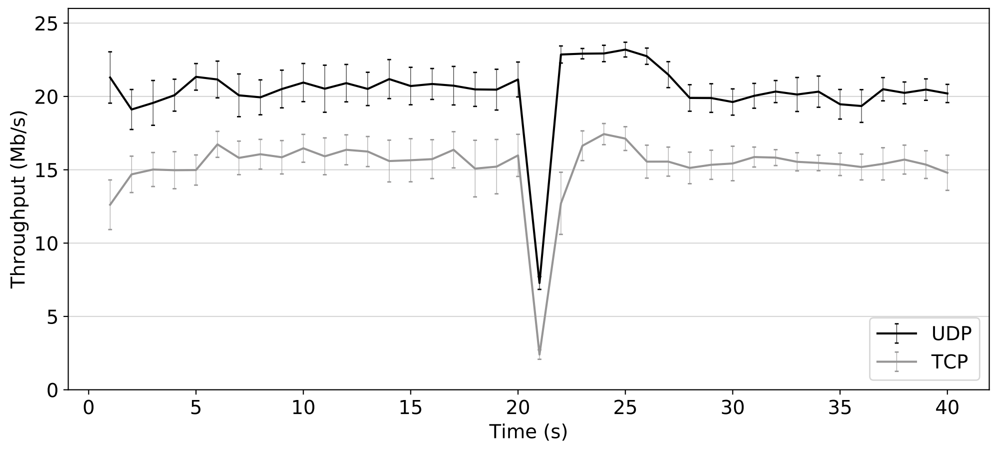
Figure 5.8: Overview of the topology used for throughput, outage time and packet loss evaluation of the proposed solution.

Figure 5.9 shows the average throughput experienced by the station for each handoff scenario. In Figure 5.9 (a), one can observe that there was no significant throughput drop during the migration period (*i.e.*, no valleys can be seen around 20 seconds of execution). Moreover, it is interesting to note that after migration, TCP throughput increased. This behavior can be explained by the fact that channel 11 was less busy than channel 1, and thus TCP was able to increase its packet transmission rate. In Figure 5.9 (b), a noticeable throughput drop occurs after 20 seconds of execution, which affects TCP more than UDP. Finally, in the Figure 5.9 (c), the throughput for both protocols drops to zero

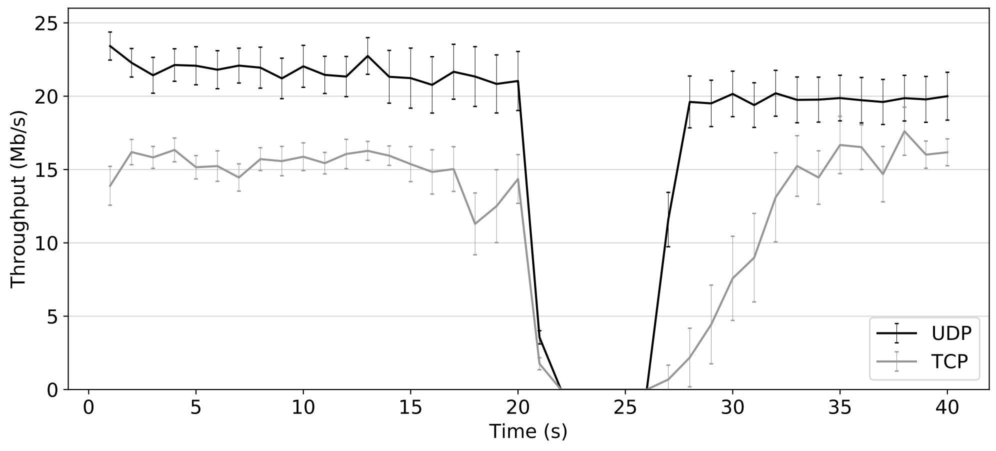
[3]Available at: https://linux.die.net/man/1/at



(a) Our Proposal



(b) Deauth Scenario



(c) No Deauth Scenario

Figure 5.9: Average TCP and UDP throughput experienced by the station in three different handoff scenarios. The confidence interval is based on 30 executions with a confidence level of 95%.

during several seconds, with a slow recovery of the TCP throughput due to its congestion control.

Following the throughput analysis, we also addressed the connection outage[4] time caused by the handover, which, in our experiment, comprises the delay between the last TCP/UDP packet received through the AP in channel 1 until the first packet received through the AP in channel 11. The *tshark* tool was used to process the traffic exchanged between source and destination. Figure 5.10 shows the average outage time for TCP and UDP traffic.

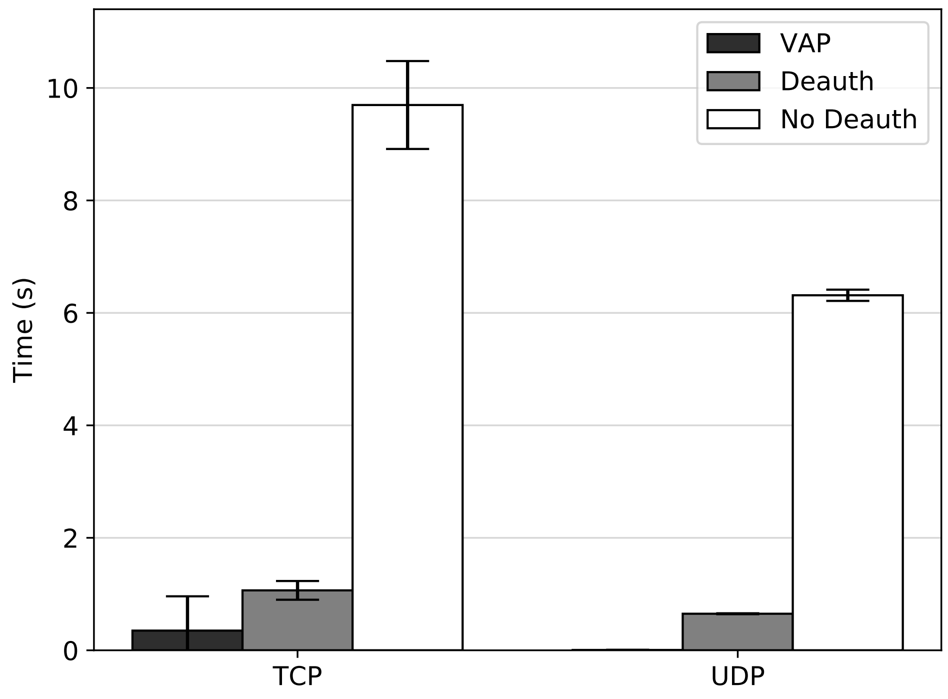


Figure 5.10: Average outage times experienced by the station in three handoff scenarios. The 95% confidence intervals are based on 30 executions of the experiment.

The results show an average outage time of 350 ms for TCP and less than 6 ms for UDP using our solution. In comparison to the *Deauth* approach — which is similar to a typical station-based handover — this represents a reduction of around 67% in the average outage time for TCP connections and 99% for UDP. Also, not advertising the BSS deactivation increases the average outage time approximately 9 times, as can be noticed by examining the *Deauth* and *No Deauth* scenarios.

Comparing both transport protocols, our proposal achieved a TCP outage time significantly higher than that of UDP. This occurred because in one of the 30 executions

[4]The outage period refers to the amount of time in which the client is not able to receive or transmit data because it is still changing its channel (in the scenario of the proposed solution) or it is not yet associated with the other available AP after disassociating from the first AP (in the other two scenarios).

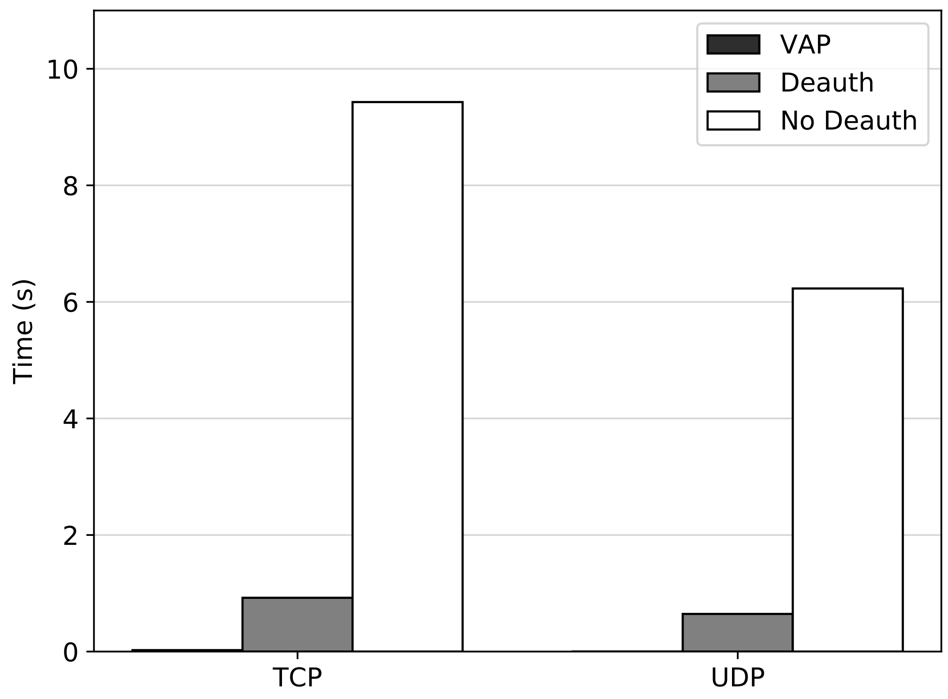


Figure 5.11: Median outage times experienced by the station in three handoff scenarios. The median is based on 30 executions of the experiment.

the outage time was unexpectedly high, with the TCP traffic taking several seconds to resume after the handover. Although the cause for this is unknown, we investigated this particular result and found that the proposed solution behaved as expected, since the station received the CSA IE and started receiving frames from the new physical AP without requiring a new association with its VAP. Figure 5.11 presents the median outage time as an alternative metric, which helps to reduce the outlier interference in the visualization of the results. In comparison to the *Deauth* scenario, our solution resulted in a reduction of approximately 97% in the median outage time using TCP. We should highlight that, even on average — *i.e.*, with the effect of the outlier —, our solution achieved a satisfactory outcome.

For the packet loss analysis we repeated the same methodology used in the two previous experiments. However, we also used a sniffer in the desktop node (*i.e.*, the source of the traffic) to analyze the packet rate at the traffic source. Since both TCP and UDP operate over the Internet Protocol (IP), the identification field in the IP header was used for packet loss computation. If a gap in the values of this field is detected between two consecutive packets, then packets were considered lost.

Figure 5.12 shows the average percentage of packets lost during the experiment. As can be seen, in all cases the UDP traffic had significantly more packets lost than TCP.

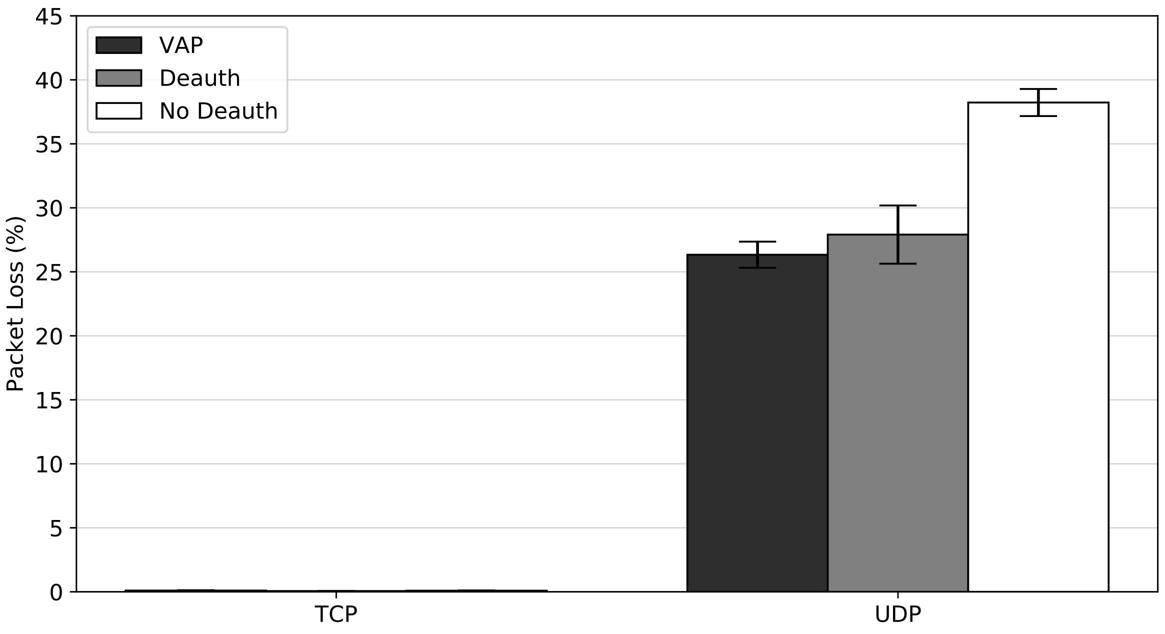
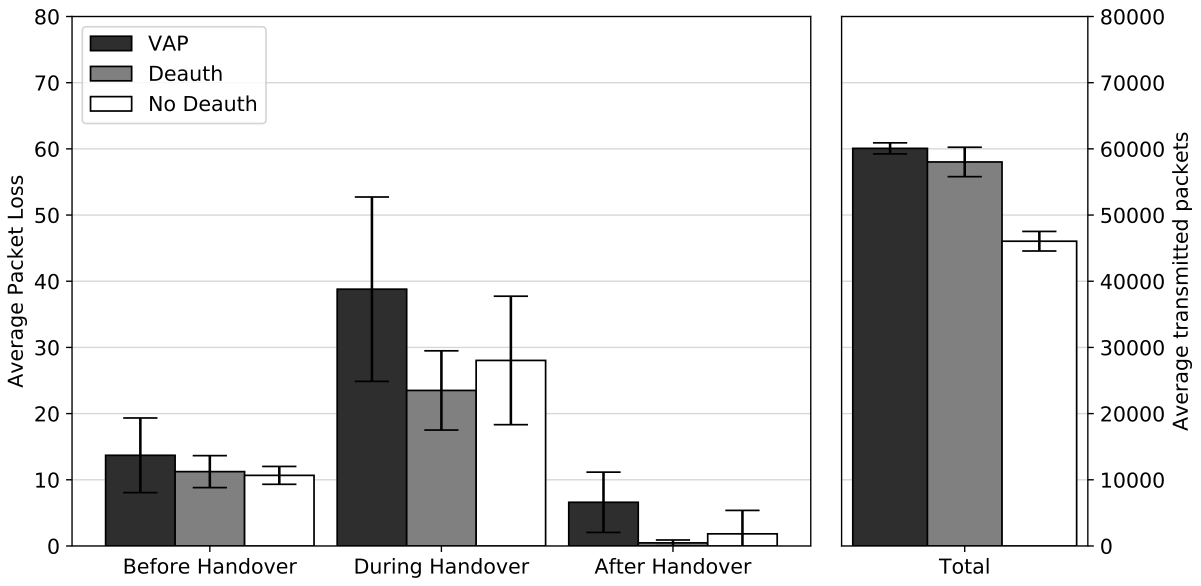


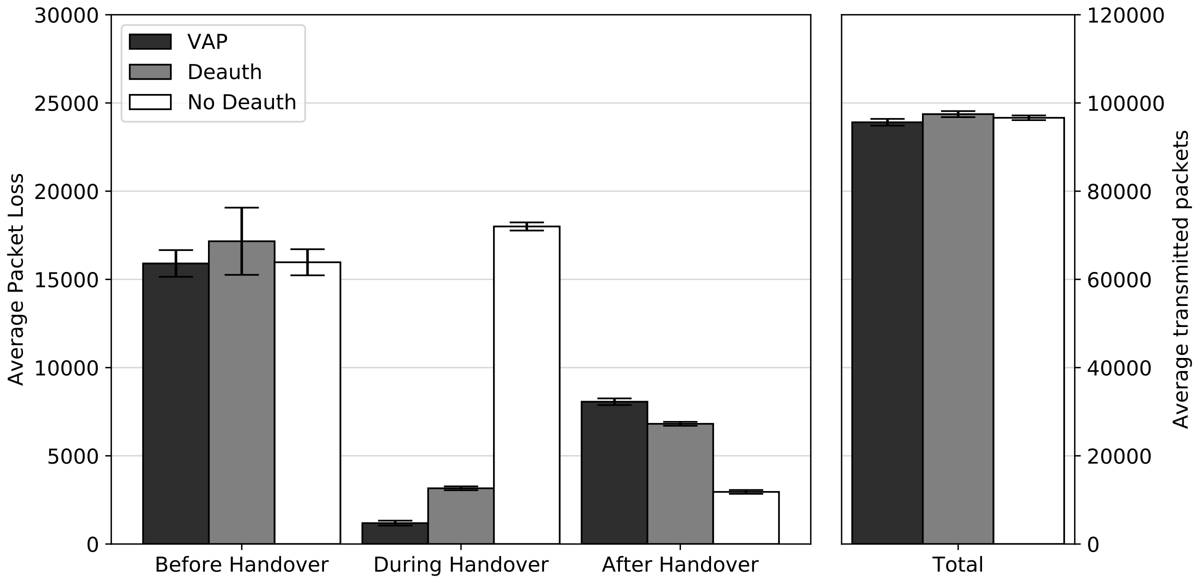
Figure 5.12: Average packet loss percentage experienced by the station through the duration of the experiment (40 seconds). The confidence interval is based on 30 executions with a confidence level of 95%.

This behavior is expected since, unlike UDP, TCP has a congestion control mechanism that reduces its transmission rate when acknowledgments for sent packets are not received. Consequently, the number of lost packets is also reduced. Moreover, one can observe that the proposed solution obtained a lower packet loss rate with UDP. In contrast, our mechanism exhibited a slightly higher amount of lost packets for TCP traffic. However, this difference was within the confidence interval of the other two scenarios. Although this result provides an overview of packet losses in this experiment, it does not paint a full picture. Notice that this experiment comprises three different stages: before, during and after the handover. Thus, in order to completely understand the behavior of each approach, it is important to scrutinize the losses during each of these stages. Figure 5.13 presents the average number of lost packets in each stage of the experiment and the average number of transmitted packets.

In relation to UDP, our proposal resulted in less packet losses during the handoff phase when compared to other scenarios, exhibiting an average of less than 2%. In comparison, the Deauth scenario had 161% more losses. On the other hand, in the *No Deauth* scenario most packets were lost during the handover phase.

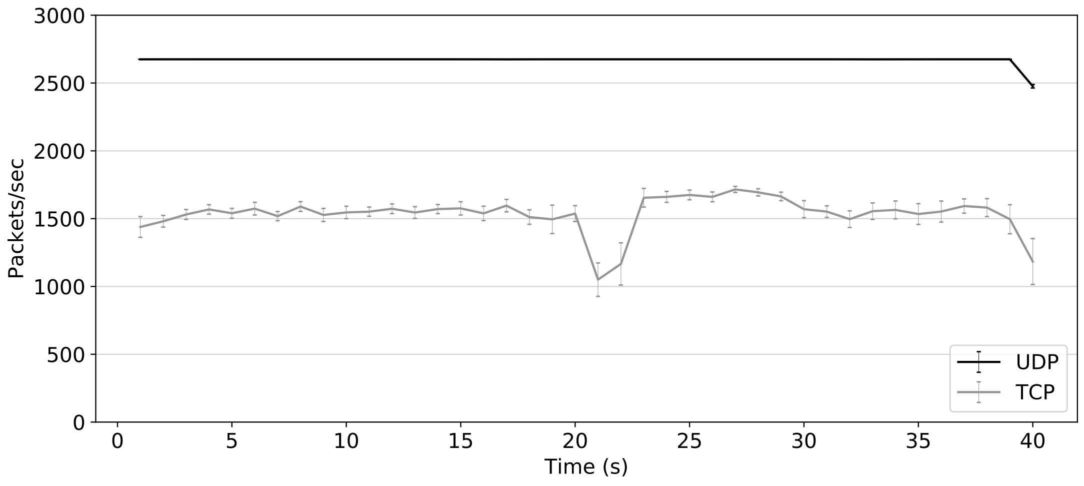


(a) TCP

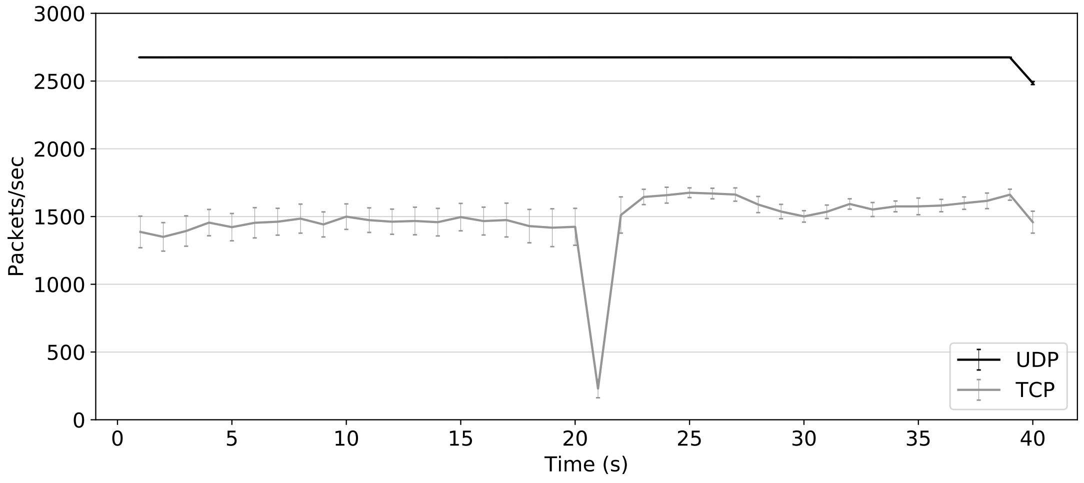


(b) UDP

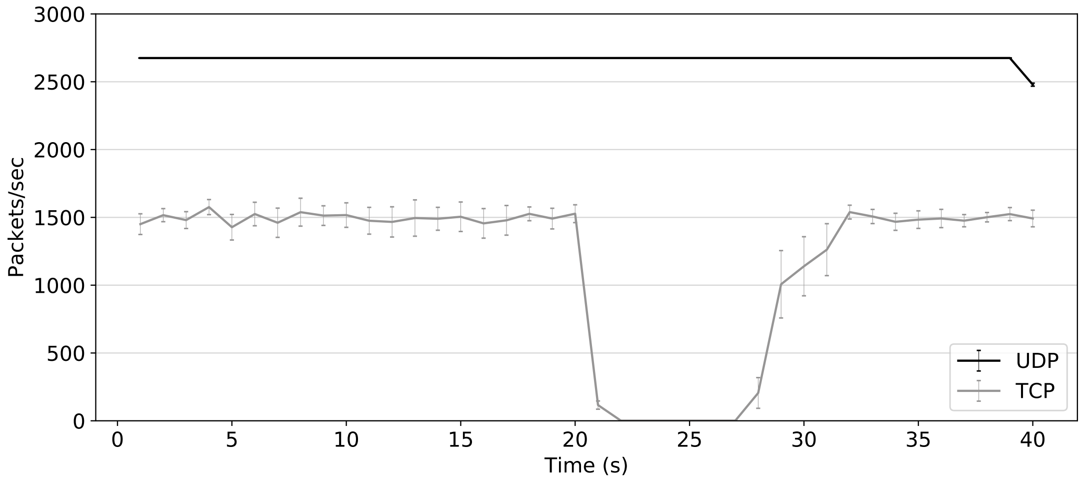
Figure 5.13: Average packet loss experienced by the station before, during and after the handover and the average number of packets transmitted during the experiment. The confidence interval is based on 30 executions with a confidence level of 95%.



(a) Our Proposal



(b) Deauth Scenario



(c) No Deauth Scenario

Figure 5.14: Average TCP and UDP packet rate measured in the source. The confidence interval is based on 30 executions of the experiment.

With TCP, however, our solution resulted in the highest packet loss rates during the handover. Nevertheless, this adverse result is due to TCP's congestion control mechanism. As shown in Figure 5.14, during the handover process, TCP's transmission rate with our solution was higher than that of the two other scenarios. Consequently, more packets were lost in the brief disconnect time. In the other two scenarios, however, packet transmission rates drop significantly, resulting in fewer losses. With UDP, conversely, the send rate remains constant throughout the execution, and the number of lost packets is proportional to the outage time, which corroborates the relationship TCP's transmission rate and packet loss during the handover.

Comparing the three approaches, the *No Deauth* scenario obtains the worst performance. Because no advertising is transmitted before the BSS deactivation, the station first needs to detect that the AP is not available, scan nearby BSSs and then associate with a new AP. Sending a deauthentication frame — as in the *Deauth* scenario — eliminates the burden of detecting the AP outage, reducing the delay of until the station re-connects to the network. Although our solution also presents a delay during the migration phase, the station can transmit or receive packets through the new associated AP as soon as it changes channels, without the need for re-association or medium scan. Therefore, the number of lost packets is significantly lower in comparison to prior cases.

Chapter 6

Conclusion

In this work, we presented a lightweight AP virtualization solution which relies on a central controller to enable inter-channel handover of clients based on a network-wide view rather than the constrained view of the station.

Our proposal allows the transparent handover of wireless stations between multiple access points, as the client remains associated with the same BSS (or VAP) during the handover process. Despite not addressing when the migration should occur, the abstraction layer provided by our controller allows the handover decision to be made based on different metrics, providing flexibility to applications that can implement their own handover and AP selection algorithms on the top of our solution. In addition to the mentioned benefits, the use of a controller makes room for several new applications.

Since the solution uses IEEE 802.11h's CSA mechanism to allow multichannel AP migration, we have also analyzed the behavior of multiple devices upon receiving a frame with the CSA IE. The investigation has shown that different stations react differently to the mechanism, sometimes requiring reassociation with the AP. Still, most of the evaluated devices present some form of support to CSA and channel switch is faster with it.

That support allowed us to implement a prototype of our proposed architecture on top of HostAPD. Based on that implementation, we carried on experiments to assess performance aspects of the migration. Our results show that our proposed virtualization solution indeed allows seamless migration of client stations between physical APs, with little to no impact in terms of delay and packet losses even when the migration involves a channel switch. As such, we believe that our proposal increases the flexibility of wireless networks, allowing network-wide optimization tasks.

A well-known problem of IEEE 802.11 networks is when a client, in the border of the coverage area of two or more APs, switches continuously from one AP to another due to the variations of signal in a phenomenon known as *ping-pong effect* [6]. The proposed solution could be used to mitigate this problem by moving the VAP along with the client, avoiding the station to perform handoffs between BSSs.

Considering a dense wireless scenario, multiple wireless clients may try to connect to the same closest AP, even if there are other idle APs in the vicinity. With the proposed solution, by detecting such a situation, a controller could perform load management by migrating stations to nearby access points in order to balance the resource utilization of the wireless stations. The same technique can be applied to mitigate interference between transmitters on the same channel, migrating clients to APs operating in non-overlapping channels to reduce collisions during transmissions. Also, the energy consumption of a WLAN can be reduced by migrating stations from sub-utilized APs, concentrating these clients at fewer APs of the network, which allows the deactivation of the radio of the unused infrastructure devices.

6.1 Limitations

The proposed virtualization solution has a strong dependency on the CSA mechanism. Although CSA has been part of the IEEE 802.11 standard since 2007 [5], legacy devices may not perform the channel switch upon receipt of the CSA IE. Still, the proposed solution is compatible with these legacy devices. The main difference is that, after a migration, they need to re-scan and reassociate with the VAP on the new physical access point. As a result, these devices might experience considerably longer delays and packet losses during the handover process when compared to fully CSA-compliant devices. Furthermore, as shown in Section 5.2, devices that recognize the CSA may also require a reassociation, which might increase the total handover time.

Another limitation is related to the heterogeneity of the devices that act as access points. The number of concurrent BSSs supported by a wireless interface varies depending on the model and manufacturer of the network card. Therefore, since the virtualization solution consists of an encapsulation of BSSs, the number of VAPs supported by each physical access point is directly limited by how many BSSs the interface supports. That limitation can be mitigated by the addition of multiple wireless interfaces in each physical access point, although that will increase the cost of deployment.

In our tests, duplicated frames were received by a station when migrations between APs within the same channel occurred. Although this behavior did not cause any problems in the communication between devices, duplicates are an undesirable characteristic. Also, this work currently does not address the redirection of the packets — destined to a station — that are in the buffer of the source AP after the client is migrated to another physical AP. Furthermore, in our implementation, the AP keeps transmitting beacons for each occupied and available VAP. Because the solution does not indicate which VAP is vacant, a nearby station, seeking connection to the network, might attempt to connect to several occupied VAPs before successfully associating with the available VAP.

6.2 Future work

As future enhancements for this research, we intend to extend the solution by integrating it to an OpenFlow controller — such as ONOS[1] or OpenDayLight[2]. That integration will allow packets in the wired distribution system to be immediately forwarded to the correct physical AP when a VAP is migrated through the installation of flow rules on the switches. Although the use of an SDN controller is not strictly necessary to accomplish this redirection, we envision that such inclusion will reduce the amount of packets that are lost during the handover process. Besides that, the inclusion of an OpenFlow controller would greatly increase the versatility of the solution. Also, a single controller is responsible for managing all APs of the network. Therefore, we intend to analyze the scalability of the solution concerning the demands of the application plane (*e.g.*, how often the application plane requests data plane information, which will vary according to the type of application).

We also plan to devise some kind of synchronization mechanism among the physical APs involved in migrations to avoid the reception of duplicated frames by the station when these APs are operating on the same channel. We intend to build solutions for load balancing, mobility management and network power saving that rely on the proposed architecture. These can be implemented as algorithms of the application plane which communicate with our control plane. In addition, to facilitate the association of stations that want to connect to the network, we intend to modify the beaconing behavior of occupied VAPs, by increasing the interval between beacon transmissions or disabling the beaconing of these VAP. This would make the available VAP more apparent during the

[1]Available at: https://onosproject.org/
[2]Available at: https://www.opendaylight.org/

scan process of nearby stations, in comparison to the occupied ones. We also intend to further investigate the restriction regarding the number of BSS in a single wireless NIC and how to overcome this limitation.

References

[1] Network Simulator ns-3. https://www.nsnam.org/. Accessed: 2019-11-21.

[2] Official IEEE 802.11 working group project timelines. http://www.ieee802.org/11/Reports/802.11_Timelines.htm. Accessed: 2019-12-28.

[3] OpenFlow Specifications. https://www.opennetworking.org/software-defined-standards/specifications. Accessed: 2019-12-31.

[4] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Spectrum and Transmit Power Management Extensions in the 5 GHz Band in Europe. *IEEE Std 802.11h-2003 (Amendment to IEEE Std 802.11, 1999 Edn. (Reaff 2003))* (Oct 2003), 1–75.

[5] 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 802.11-2016 (Revision of IEEE Std 802.11-2012)* (Dec 2016), 1–3534.

[6] Balbi, H.; Passos, D.; Carrano, R. C.; Magalhães, L. C. S.; Albuquerque, C. V. N. A Case Study of Association Instability in Dense IEEE 802.11 Networks. In *International Symposium on Computers and Communications (ISCC)* (2019).

[7] Berezin, M. E.; Rousseau, F.; Duda, A. Multichannel virtual access points for seamless handoffs in IEEE 802.11 wireless networks. In *IEEE Vehicular Technology Conference* (may 2011), IEEE, pp. 1–5.

[8] Carrano, R. C.; Magalhães, L. C. S.; Saade, D. C. M.; Albuquerque, C. V. N. IEEE 802.11s Multihop MAC: A Tutorial. *IEEE Communications Surveys Tutorials 13*, 1 (First 2011), 52–67.

[9] Dely, P.; Vestin, J.; Kassler, A.; Bayer, N.; Einsiedler, H.; Peylo, C. CloudMAC — An OpenFlow based architecture for 802.11 MAC layer processing in the cloud. In *2012 IEEE Globecom Workshops* (Dec 2012), pp. 186–191.

[10] Dutra, D. L. C.; Bagaa, M.; Taleb, T.; Samdanis, K. Ensuring End-to-End QoS Based on Multi-Paths Routing Using SDN Technology. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference* (Dec 2017), pp. 1–6.

[11] Ekiz, N.; Salih, T.; Kucukoner, S.; Fidanboylu, K. An overview of handoff techniques in cellular networks. *International journal of information technology 2*, 3 (2005), 132–136.

[12] Federal Communications Commission. A Short History of Radio. `https://transition.fcc.gov/omd/history/radio/documents/short_history.pdf`. Accessed: 2019-12-24.

[13] Grunenberger, Y.; Rousseau, F. Virtual access points for transparent mobility in wireless LANs. In *IEEE Wireless Communications and Networking Conference, WCNC* (apr 2010), IEEE, pp. 1–6.

[14] Gude, N.; Koponen, T.; Pettit, J.; Pfaff, B.; Casado, M.; McKeown, N.; Shenker, S. NOX: towards an operating system for networks. *ACM SIGCOMM Computer Communication Review 38*, 3 (2008), 105–110.

[15] Haque, I. T.; Abu-Ghazaleh, N. Wireless software defined networking: A survey and taxonomy. *IEEE Communications Surveys & Tutorials 18*, 4 (2016), 2713–2737.

[16] IEEE. 25th Anniversary of IEEE 802.11™. `https://standards.ieee.org/events/802-11.html`. Accessed: 2019-12-24.

[17] Jain, S.; Kumar, A.; Mandal, S.; Ong, J.; Poutievski, L.; Singh, A.; Venkata, S.; Wanderer, J.; Zhou, J.; Zhu, M., et al. B4: Experience with a globally-deployed software defined WAN. In *ACM SIGCOMM Computer Communication Review* (2013), vol. 43, ACM, pp. 3–14.

[18] Kawada, M.; Tamai, M.; Yasumoto, K. A trigger-based dynamic load balancing method for WLANs using virtualized network interfaces. In *2013 IEEE Wireless Communications and Networking Conference (WCNC)* (April 2013), pp. 1091–1096.

[19] Kreutz, D.; Ramos, F. M. V.; Veríssimo, P. E.; Rothenberg, C. E.; Azodolmolky, S.; Uhlig, S. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE 103*, 1 (Jan 2015), 14–76.

[20] Kurose, J. F.; Ross, K. W. *Computer Networking: A Top-Down Approach*, 5 ed. Pearson, 2009.

[21] McKeown, N.; Anderson, T.; Balakrishnan, H.; Parulkar, G.; Peterson, L.; Rexford, J.; Shenker, S.; Turner, J. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communication Review* (2008).

[22] Moura, H.; Bessa, G. V. C.; Vieira, M. A. M.; Macedo, D. F. Ethanol: Software defined networking for 802.11 Wireless Networks. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)* (May 2015), pp. 388–396.

[23] Murty, R.; Padhye, J.; Chandra, R.; Wolman, A.; Zill, B. Designing High Performance Enterprise Wi-Fi Networks. In *NSDI* (2008), vol. 8, pp. 73–88.

[24] Nunes, B. A. A.; Mendonca, M.; Nguyen, X.-N.; Obraczka, K.; Turletti, T. A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications Surveys & Tutorials 16*, 3 (2014), 1617–1634.

[25] Passos, D.; Balbi, H.; Carrano, R. *Tecnologias de Redes Sem Fio. Escola Superior de Redes*, 2016. In Portuguese.

[26] Schwartz, M.; Abramson, N. The Alohanet - surfing for wireless data. *IEEE Communications Magazine 47*, 12 (Dec 2009), 21–25.

[27] Stiti, O.; Braham, O.; Pujolle, G. Virtual Openflow-based SDN Wi-Fi access point. In *2015 Global Information Infrastructure and Networking Symposium (GIIS)* (Oct 2015), pp. 1–3.

[28] Suresh, L.; Schulz-Zander, J.; Merz, R.; Feldmann, A.; Vazao, T. Towards programmable enterprise WLANS with Odin. In *Proceedings of the first workshop on Hot topics in software defined networks - HotSDN '12* (2012), ACM Press, p. 115.

[29] Zeljković, E.; Marquez-Barja, J. M.; Kassler, A.; Riggio, R.; Latré, S. Proactive Access Point Driven Handovers in IEEE 802.11 Networks. In *2018 14th International Conference on Network and Service Management (CNSM)* (Nov 2018), pp. 261–267.

[30] Zubow, A.; Zehl, S.; Wolisz, A. BIGAP — Seamless handover in high performance enterprise IEEE 802.11 networks. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (April 2016), pp. 445–453.