Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2020

Inês Mariana Lemos Lopes

Conectividade Definida por Software em Ambientes Móveis

Software-Defined Connectivity in a Mobile Environment

Universidade de Aveiro
2020
Departamento de Eletrónica,
Telecomunicações e Informática

Inês Mariana Lemos Lopes

Conectividade Definida por Software em Ambientes Móveis

Software-Defined Connectivity in a Mobile Environment

“*Unless someone like you cares a whole awful lot, nothing is going to get better. It's not.*”

— Dr. Seuss in "The Lorax"

Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2020

Inês Mariana Lemos Lopes

Conectividade Definida por Software em Ambientes Móveis

Software-Defined Connectivity in a Mobile Environment

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor João Paulo Silva Barraca, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Diogo Nuno Pereira Gomes, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president	Professor Doutor Arnaldo Silva Rodrigues de Oliveira Professor Auxiliar, Universidade de Aveiro
vogais / examiners committee	Professor Doutor Pedro Miguel Alves Brandão Professor Auxiliar, Faculdade de Ciências da Universidade do Porto
	Professor Doutor João Paulo Silva Barraca Professor Auxiliar, Universidade de Aveiro

agradecimentos / acknowledgements

Agradeço ao Professor Doutor João Paulo Barraca por toda a disponibilidade, pelos conhecimentos transmitidos, apoio e motivação dados para que eu conseguisse ultrapassar dificuldades e melhorar o meu trabalho. Ao Instituto de Telecomunicações de Aveiro, pelos recursos disponibilizados e por proporcionar as condições necessárias à realização desta dissertação.

Um grande obrigada aos meus colegas, que me ajudaram a ultrapassar momentos difíceis e que deixaram marca ao tornarem o meu percurso académico inesquecível. Guardarei para sempre todas as boas memórias.

Para finalizar, quero agradecer às pessoas mais importantes da minha vida, que sempre me apoiaram incondicionalmente: a minha família. Agradeço aos meus irmãos, Sara e João, por me terem animado quando me sentia em baixo e por me terem dado forças para continuar nesta batalha. À minha mãe, Leonor, dedico o maior agradecimento de todos, por ter sido a pessoa com quem sempre pude contar, nos bons e nos maus momentos, e mais importante que tudo, por me ter dado a oportunidade de frequentar o ensino superior na universidade que eu queria. Nunca irei esquecer todo o apoio que me deram ao longo destes anos para que eu conseguisse alcançar esta meta.

Palavras Chave

Internet em comboios, alta mobilidade, congestionamento de rede, balanceamento de carga, redundância de gateway, rede definida por software, Openflow.

Resumo

O acesso à Internet de banda larga em comboios tornou-se num serviço esperado por parte dos passageiros e o aumento na exigência de qualidade tem constituído um desafio para os prestadores de serviços. Existem soluções de acesso à Internet em comboios que carecem da flexibilidade e redundância necessárias para uma melhor qualidade de serviço na rede. Assim, esta dissertação estuda duas novas soluções de balanceamento de carga, uma distribuída e outra centralizada. Numa rede de comboio emulada, routers de cada carruagem comunicam as suas estatísticas de rede para outros nós da rede, para posteriormente serem tomadas decisões de balanceamento de carga. Na solução distribuída, cada router tem capacidade de tomar decisões de balanceamento de carga, enquanto que na solução centralizada, um controlador dentro do comboio toma essas decisões. Esta última solução baseia-se num balanceamento de carga em ambiente SDN. Neste sistema, uma entidade remota agrega o tráfego dos passageiros e encaminha-o para uma rede externa, ou para a Internet. Os sistemas devem proporcionar uma qualidade de experiência melhorada aos passageiros do comboio e a largura de banda total disponível deve ser distribuída mais uniformememente por todas as carruagens. Requisitos foram estabelecidos para os sistemas, que foram desenvolvidos após uma análise aprofundada das soluções de acesso à Internet em comboios encontradas na literatura. Após a obtenção dos resultados dos testes de desempenho, concluiu-se que as duas soluções melhoram o QoE dos passageiros, desde que as condições da rede sejam aproximadamente constantes durante um certo período.

Keywords

Internet access on board, high mobility, network congestion, load balancing, gateway redundancy, software-defined networking, Openflow.

Abstract

Broadband Internet access on trains has become an expected service for passengers, and the rise in quality demand has been posing a challenge to service providers. There are solutions for Internet access on trains that lack the flexibility and redundancy needed for an improved QoS in the network. Thus, this dissertation studies two new load balancing solutions, one distributed and the other centralized. In an emulated train network, routers of each car communicate their network statistics to other nodes of the network, to be used in load balancing decisions. In the distributed solution, each router has the capacity to make load balancing decisions, while in the centralized solution, an onboard controller makes those decisions. The latter solution is based on load balancing in an SDN environment. In this system, a remote entity aggregates the passengers' traffic and forwards it to an external network, or to the Internet. The systems must provide an improved quality of experience to the passengers on the train and the total avaliable bandwidth should be distributed evenly by all cars. Requirements were established for the systems, which were developed after an in-depth analysis of solutions for Internet access on board found in the literature. After obtaining the results of the performance tests, it was concluded that the systems improve the QoE of the passengers, as long as the network conditions are approximately constant for an extended period.

Contents

List of Figures

List of Tables

Acronyms

AP	Access Point
API	Application Programming Interface
CIDR	Classless Inter-Domain Routing
CPE	Customer Premises Equipment
DNS	Domain Name System
eNodeB	Evolved Node B
GRE	Generic Routing Encapsulation
IP	Internet Protocol
JSON	JavaScript Object Notation
LACP	Link Aggregation Control Protocol
LAN	Local Area Network
MAC	Media Access Control
MEC	Multi-access Edge Computing
NAT	Network Address Translation
OS	Operating System
OVS	Open vSwitch
OVSDB	Open vSwitch Database
QoE	Quality of Experience
QoS	Quality of Service
RoF	Radio-over-Fiber
RSRP	Reference Signal Received Power
RSRQ	Reference Signal Received Quality
RTT	Round-Trip Time
SDN	Software-Defined Networking
STA	Station
TCP	Transmission Control Protocol
UCI	Unified Configuration Interface
VM	Virtual Machine
WAN	Wide Area Network
WLAN	Wireless Local Area Network

1 Introduction

"Even if you are on the right track, you'll get run over if you just sit there."
- Will Rogers

Nowadays, nearly every person owns a smartphone, tablet, or computer with 3G/4G and Wi-Fi (IEEE 802.11) capabilities that allows them to be connected most of the time, if not all the time, to the Internet. Also, society has the constant need to be well-informed of the news, social updates, and other information that the Internet provides them. Due to the mobile nature of smartphones, smartwatches, and other technologies that connect to the Web, they follow along with its users every day.

Transportation companies try more and more to meet the requirements of an increasingly interconnected world, to allow passengers to spend their trip time working or entertaining themselves, and also to keep up with competitors. For this reason, in a world with various transportation companies, a good and efficient Internet connection may be a decisive factor when choosing the next trip. Thus, Internet services have become a must on modern public transports, and the rise of Internet access' quality demand has been posing a considerable challenge to service providers. As explained in the State of the Art, there are numerous solutions for providing Internet access on trains, but the most widely used is broadband access using the cellular network. However, this technology also poses significant difficulties, due to the high speed achieved by some trains, linked to a rapidly changing signal quality, and the limited coverage in some locations, among others. A good and fast Internet connection can also be related to safety since it allows operation centers to monitor the train-related information and receive close to real-time updates.

It is worth noting that an increasing number of Internet users resorts to streaming services for entertainment or video calls for work meetings, for instance. These online applications usually take up the most bandwidth and, since it is shared between all passengers in a car, if not shared equally or used responsibly, network congestion becomes very common.

So, the above described increasingly popular pattern of Web usage is a big challenge in a network environment of limited bandwidth and rapidly changing signal strength.

1.1 MOTIVATION

Some network architecture solutions for Internet access on trains in the railway industry and others found in the literature, consist of placing one or more wireless routers in each car, which will then connect to an access network. These solutions may lack in control and management components (local or remote), with an isolation of the routers in each car, only serving their own passengers. Thus, the routers are never aware of their operational peers.

As expected, this poses a significant problem for passengers in the car, mainly because the router's gateway becomes a single point of failure that is susceptible to increasingly slower Internet connection speed or worse, total disconnection.

In this network topology, routers do not balance their client's traffic, and they do not report their uplink status to some local or remote monitoring entity that could easily control them and avoid the situation of cars without an Internet connection or slow Internet speed for their passengers.

The mentioned problems of this network solution illustrate the need for a new network architecture that would take into consideration the existing issues and improve the management of the network elements significantly so that they can cooperate and report their current status to peers.

1.2 OBJECTIVE

The main objective of this dissertation is to develop load balancing mechanisms that can assure that all passengers of the train have an uniform access to the Internet with increased QoE. After further improvements and adaptations, the solution could be used in a real-world train's network.

In order to fulfill this goal, it was first developed a distributed load balancing system, used to evaluate and validate the reliability of such a load balancing algorithm in an environment of Linux-based routing devices. The analysis and evaluation of the performance of the system allowed to conclude it could bring positive improvements to the train network. However, its issues had to be tackled in a more complex environment, which is the case of the developed centralized load balancing system.

The centralized solution consists of a load balancing algorithm implemented in an SDN [1] environment. This solution aims to bring greater flexibility and redundancy to a train's network, as well as increase the number of web services the passengers in some cars can access, and decrease the time it takes to access them.

Testing scenarios were described for both solutions, to further evaluate the degree to which the QoE of the passengers would increase with their usage and which could be the drawbacks of both solutions. The testing scenarios varied in the type of access by clients and the network degradation applied to the gateways of the network. The main results that evaluate the QoE associated with the systems are the number of downloads of a website per client and the total time it took to download them completely.

1.3 STRUCTURE OF THE DOCUMENT

The remainder of this document is structured like so:

- **Chapter 2 - State of the Art:** state of the art and related work is presented in this chapter, including key-concepts of network access inside trains and high-speed trains, as well as of SDN, Openflow and load balancing algorithms. Similar network architectures already deployed in trains or theoretical architectures found in the literature are also described.
- **Chapter 3 - Solution for Load Balancing in Trains:** introduces a reference network architecture and the proposed load balancing solution to minimize the negative effects of its associated problems. Also, the requirements of the system, the centralized solution's network architecture, and network elements needed for the normal operation of the system. It is also presented the flow of the messages exchanged by the system's elements.
- **Chapter 4 - Implementation:** first, a summary of the technologies used in this dissertation was made in order to contextualize the reader who may not be familiar with them. Following, an in-depth description of the implementation of the systems in a virtual environment details the distributed load balancing mechanism, followed by the centralized mechanism in an SDN environment.
- **Chapter 5 - Evaluation and Results:** Diverse testing scenarios were used to evaluate the performance of the solutions. The chapter contains a discussion of all the obtained results and of the possibility of a real-world implementation of the system.
- **Chapter 6 - Conclusions:** contains the main conclusions of the dissertation, including its limitations. It is also of great interest to discuss the future work, which can be found at the end of this chapter.

2 State of the Art

2.1 Introduction

The purpose of this chapter is to introduce the reader to concepts that are relevant in the scope of this dissertation and related work found in the literature. Section 2.2 covers various topics regarding Internet access on trains, such as terrestrial technologies for broadband access on trains, architectures described in the literature assessing train-to-infrastructure, inter and intra-car communication, future technologies, communication issues and challenges of Internet provisioning. Next, section 2.3 explores the Software-Defined Networking and OpenFlow concepts and architectures. Sections 2.4 and 2.5 describe, respectively, the gateway redundancy concept with related protocols and link and traffic aggregation concepts with associated protocols and solutions found in the literature. Finally, section 2.6 contains an overview of load balancing algorithms used in traditional networks and SDN.

2.2 Internet access on trains

Nowadays, there are millions of mobile devices connected to the Internet. Onboard Internet services are becoming a need in modern trains, and the rise of Internet access quality demand has been posing a challenge to service providers, due to the high speeds achieved by some trains (higher speed equals to more handovers per unit of time), the rapidly changing signal quality, the limited coverage, among others.

Since trips tend to be long, people feel the need to spend this time working or entertaining themselves. An efficient Internet connection may be a decisive factor when clients choose a type of transportation.

So, Internet users have high demands for service quality and Internet speed. Passengers not only require mobile coverage, but they also want to successfully and efficiently use various broadband services inside trains.

Two major options to be considered when providing Internet access onboard are:

- **Satellite architectures:** satellites are a solution used to provide onboard Internet services and were the first solution to be put into practice [2]. This technology has some advantages and also some limitations. It is simple to deploy because, as there is no need for a terrestrial network, there is also no need for an agreement between the railway operators and the network infrastructure administrators. Also, their coverage is broader than in terrestrial networks.

  Some disadvantages include the bandwidth limitation, latency, and the lack of connectivity when trains are inside tunnels [3]. It is also usually necessary to place big antennas on trains' roofs in order to communicate with the satellites, which has an impact on aerodynamics [2].

  Distinct types of satellites are available: Geostationary Orbit, Low, and Medium Earth Orbit [4]. They use different frequency bands and may provide unidirectional or bidirectional communications. Geostationary Orbit satellites can cover a wide geographical area and provide broadband connectivity for mobile users. For this reason, they are used in many communication and broadcasting systems [5].
- **Terrestrial architectures:** terrestrial networks may rely on existing networks, like the public cellular network (2G, 3G, 4G, and soon, 5G), or may require ground infrastructure to be deployed, which is the case of leaky coaxial cable, optical solutions, Radio-over-Fiber, Wi-Fi and WiMAX [4].

2.2.1 TERRESTRIAL SOLUTIONS FOR BROADBAND INTERNET ACCESS

In this section, some technologies that were proposed as solutions to provide broadband access to train passengers are described. The authors of [5] and [6] compiled a variety of methods for broadband access on trains. Further reading of these works is advised for the detail they present. The following compilation of terrestrial technologies (nonproprietary) takes into consideration the information in these two works and others on the theme:

Radio-over-Fiber (RoF)

Radio-over-Fiber consists of the use of an optical fiber link to transmit modulated Radio Frequency signals.

The Distributed Antenna System is an application of RoF technology. In that system, Remote Antenna Units are connected to a central processor via RoF links [7]. Refer to [8] for a review on RoF communication system.

The authors of [9] proposed the moving cell concept to reduce the handover time of passengers' stations in high-speed trains. They came up with this concept by using RoF-based DAS architecture. During a normal handover process, the STAs are required to adapt their frequency. Instead, with the "moving cell" concept, the Base Stations will track the movement of the trains to reconfigure their operating frequency subsequently, thus maintaining the communication links with the train passengers.

In [7], the authors propose a solution named "RADIATE" (RADio-over-Fiber as AnTenna Extender), to provide broadband Internet services in high-speed trains. The architecture is depicted in figure 2.1. This solution explores both the cellular network and the moving cell concept (explained previously). The architecture consists of wireless APs inside the cars (radio interfaces of multiple access technologies), an on-roof antenna system (using fiber links), which communicates with the cellular network and the RADIATE control system (RCS). The RCS is in charge of scheduling of passengers' traffic (fairness and distribution of workload) and optimization and control of the antenna system.

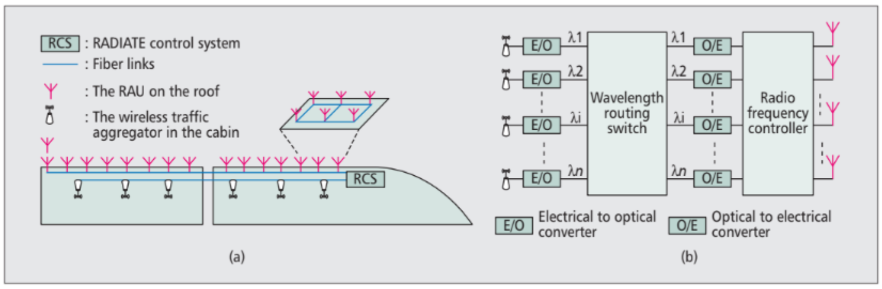
RCS : RADIATE control system
: Fiber links
: The RAU on the roof
: The wireless traffic aggregator in the cabin
RCS
(a)
E/O
λ1
λ2
λi
λn
Wavelength routing switch
O/E
Radio frequency controller
E/O Electrical to optical converter
O/E Optical to electrical converter
(b)

Figure 2.1: a) the RADIATE architecture and b) the RADIATE control system. Retrieved from [7].

In [10], an integrated WiMAX-RoF system was proposed and deployed along a high-speed railway track of the company Taiwan High-Speed Rail Corporation. In the proposed topology, base stations are placed along the tracks and connected by a broadband optical fiber backbone. They are managed by an Access Service Network Gateway whose main functions include establishing a connection between the BS and the WiMAX Customer Premises Equipment (CPE), deciding on handover events, and sending requests to the Connectivity Service Network (the element providing Internet services). RoF technology is used to cope with the signal attenuation (or loss of signal) inside the tunnels, extending the cell coverage and avoiding disconnection. The authors also propose a communication solution consisting of ground-to-train communication using WiMAX, communication inside the cars using IEEE 802.11b/g, and between cars using IEEE 802.11a. They consider that a WiMAX-RoF system "is a powerful solution for providing broadband Internet to the fast-moving train passengers".

Although associated with improved coverage and capacity in short-range communications, a big disadvantage of RoF-based solutions is the cost of deploying fiber along the railway.

Optical solutions

Free space optics (FSO), a form of optical communication, uses light in various frequencies of the spectrum to carry out a signal. Li-Fi [11] is an example of a FSO system. These types of communications achieve data rates of 1 Gbps and up, are immune to electromagnetic interference, and cannot penetrate walls. An FSO communication system has three main elements [12]: a transmitter sending the optical signals, a free-space transmission channel, and a receiver of the transmitted signals.

In [13], the authors evaluate the performance of an FSO system for broadband Internet access on trains, presenting a mathematical modeling for a ground-to-train FSO communication link. They also depict a typical FSO network architecture deployment along a railway (refer to figure 2.2). The authors state that "FSO technology with the proposed system modeling can be an alternative to provide a high bandwidth broadband access to high-speed trains".

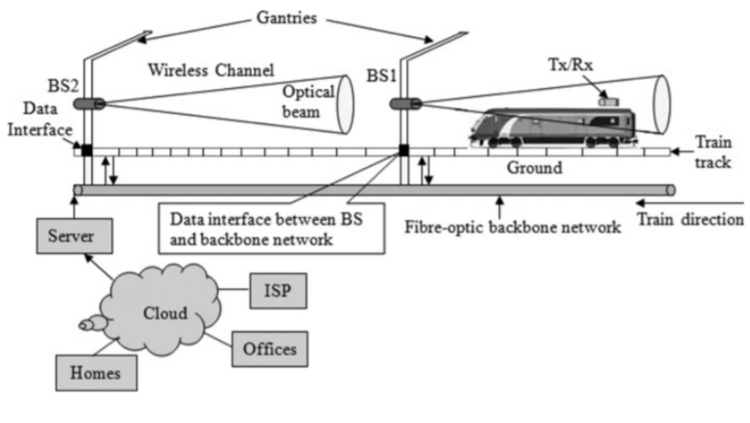

Figure 2.2: Typical ground-to-train FSO communications system, retrieved from [13].

In a recent study, [12], the authors investigated two different laser beam modalities (narrow and wide beams) for FSO communication in ground-to-train communications (for high-speed trains - 300km/h). They compared both and presented their advantages and disadvantages in different conditions, such as weather effect, range, light intensity, coverage length, security, parallel beams, among others.

Leaky Coaxial Cables (LCX)

In [14], a communication architecture was proposed for bullet trains in Japan, considering technical challenges of providing broadband Internet access to high-speed trains. The authors explain that Leaky Coaxial Cable has been utilized in Japan for radio communications on trains, being laid along railways, tunnels, subways, underground facilities, and other places where normal radio communication is not possible. However, at least at the time the work was written, it was used only for communication between operators (control center) and train drivers, to manage the train services. The authors also developed and tested (in an outdoor environment) an LCX communication system. The results indicated that data rates of up to 768 kbps could be achieved using LCX, which is not enough to satisfy bandwidth requirements for most Internet services.

A major disadvantage of LCX solutions is the cost of the deployment of the cable along the tracks.

IEEE 802.11-based Architectures (Wi-Fi)

Wi-Fi is a technology that allows good performance and good resistance to the train's high velocity. It can be used to link cars on the train into a computer network and provide Internet access to passengers. It is largely used nowadays inside trains since most devices have wireless communication capabilities. It also has the advantage that there is no need to wire the train or rewire it in case of reconfiguration.

In the architecture proposed on [15], users are connected to the Internet via a satellite gateway, through a Train Server. However, the server also has gateway functions to connect to other networking technologies, in case of a satellite link outage or when an 802.11 connection may be more convenient. The network architecture is depicted in figure 2.3.

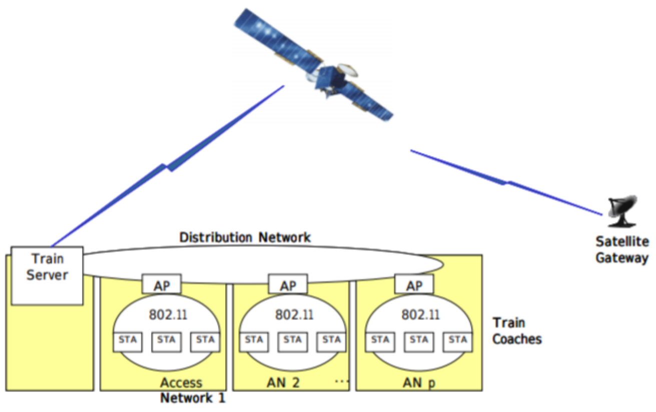

Figure 2.3: Onbard network architecture, retrieved from [15].

The authors propose two architectures for the implementation of a wireless distribution network. In both of them, there is a wireless access point in each car and several STAs (user terminals) trying to connect to the Internet.

- In the first topology, adjacent cars are connected using separate wireless links. Those links can be based on 802.11b or more advanced technologies, like 802.11a. The antennas should be located outside the cars to avoid signal attenuation (a result of the cars' infrastructure);
- In the second topology, APs are arranged in each car so that each one serves as a client STA for the AP in the previous car, but also as a host for the STAs within their car. In other words, the access network is also used as a distribution network.

The authors conclude by stating that their topologies must be tested on the real-world to measure how interference and propagation issues affect the performance and the feasibility of the architectures.

In [16], the authors propose a network architecture where onboard connectivity is provided to passengers by placing Wi-Fi APs inside each car. The APs connect to the cellular network using train-to-ground radio-access terminals (T-RATs). Figure 2.4 depicts the proposed architecture as well as onboard connectivity upon handover.

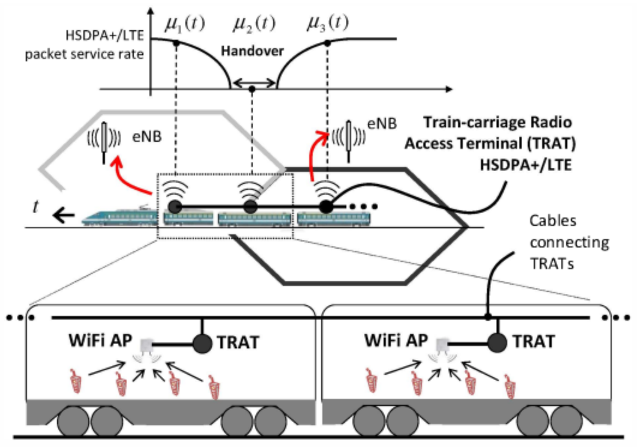

Figure 2.4: High-speed train scenario and discontinuous service in a T-RAT due to handover, retrieved from [16]

WiMAX-based Architectures

This wireless technology based on the IEEE 802.16 set of standards has a bandwidth capacity that makes it suitable for broadband Internet access on high-speed trains.

In [17], a novel (2008) technology was introduced: the SWiFT (Seamless Wireless Internet for Fast Trains), a three-tier architecture. It is based on the deployment of WiMAX (IEEE 802.16m) base stations along the railway network (Level-1). Each base station (IEEE 802.16m) consists of directional antennas, and an optical backbone is used to link the base stations together (Level-2) and to the Internet. WLAN (802.11e) access points are placed inside the train cars for onboard network (Level-0). In this architecture, each car of the train will have multiple users connecting to one or two Internet gateways. Therefore, handoffs are simplified since the Internet gateway acts as a single subscriber STA. The overhead and latency caused by handoff are, thus, reduced.

The authors of [18] compared different technologies for network connection (HSPA, E-UTRA, WiMAX). They concluded that WiMAX base stations provide acceptable coverage for several data rates and that Wi-Fi technology is only suited for connectivity inside the train. Nevertheless, 3G and 4G technologies still offer the best coverage.

Finally, in [19], the authors present a table with a comparison of Internet services on trains, in which they state that the middle-speed trains "Narita Express" of the East Japan Railway Company (JR-EAST) use WiMAX to provide Internet connection, with a maximum bandwidth of 40 Mbps.

It is important to note that WiMAX, even though possessing the capabilities needed to provide adequate Internet access, has been overshadowed by the current LTE standard.

2.2.2 COMMUNICATION WITH THE CELLULAR NETWORK

In this dissertation, broadband access through the public cellular network is in the spotlight. In the current days, most smart personal devices are multi-radio and capable of both cellular 3G and 4G communications. They are also capable of communicating through Wi-Fi.

A simplified network architecture for trains would be not to install any additional equipment, which means the passengers would have to connect directly to the cellular network. However, this architecture delivers a decreased service quality to the passengers, mainly due to the attenuation and loss of radio signals.

Some network architectures found in the literature to provide broadband Internet access to users on trains are described next.

The work [20] presents an LTE-based solution to support high throughput and continuous multimedia services on high-speed trains. The solution is based on a Cell Array, which organizes cells in a smart way along a railway (refer to figure 2.5). The architecture used to provide Internet on board consists of the use of two LTE femtocells within each car. Each femtocell covers half of a car and up to 50 simultaneous users. Two directional antennas at both ends of the train are used to connect to distant eNodeBs and thus increase the number of cells the train may be connected to (simultaneously). Communication inter-car is provided through optical fiber. The femtocell base station has two interfaces working within two different frequency bands. One interface is used for car-to-infrastructure communications (sends requests and receives the aggregated download traffic from the infrastructure LTE network), the other is used to communicate with the mobile stations within the cars.

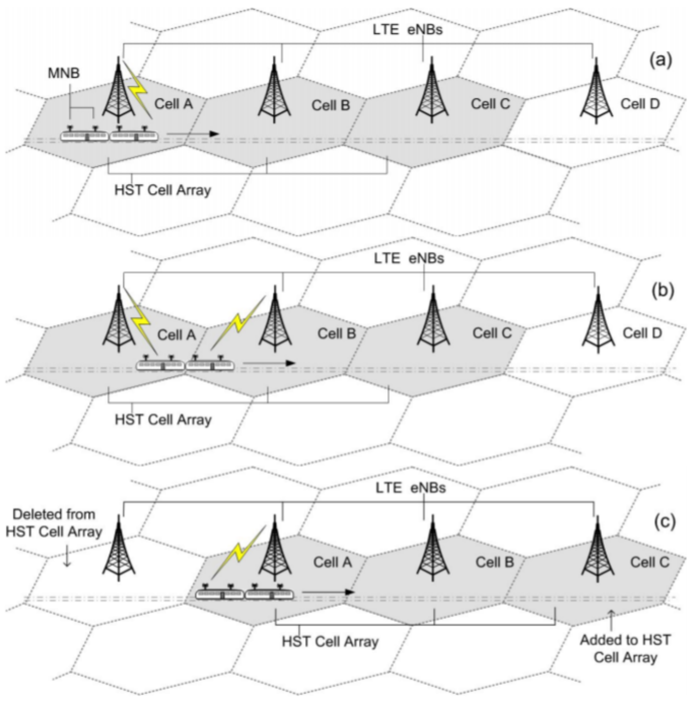

Figure 2.5: Cell Array reconfiguration, retrieved from [20].

In [21], studies were performed taking into account an architecture where there is a train access terminal, or aggregation router, that provides clients with Wi-Fi access (LAN side), aggregates traffic, and transports it via rooftop antennas (WAN side), using mobile broadband access or satellite. A similar architecture is considered in [18]. Here, the authors propose a topology where a base station transmits to a receiving antenna on the roof of each train car, then the receiving antenna connects to Wi-Fi APs inside the cars and wireless reception is finally ensured inside them. The work analyzes train communications based on real-life measurements, which were captured at train aggregation routers.

In [3], each car of a train is connected to the LTE network through one train-to-ground radio-access terminals (T-RAT). The onboard Internet access is provided by a LTE femtocell, one per car. The femtocell access points are connected through a dispatcher to the multiple T-RATs of the train. This dispatcher will allow forwarding of traffic between cars according to the status of the T-RATs' queues and service capability.

The T-RATs receive packets from onboard dispatchers and send them to the cellular network.

The work [6], presents a reference architecture to guide the discussion on broadband Internet access on trains (refer to figure 2.6). The architecture uses gateways in each car to build a train-level network. Broadband access is provided through a train access terminal (TAT), which can support one or more technology types. It connects to the access network using an antenna on the rooftop of one chosen car. The incoming signal from TAT is forwarded to gateways located in each car, which will then forward it to APs (one per car). Wi-Fi (IEEE 802.11) is the preferred technology used to provide connectivity to passengers, but a wired network is also feasible.

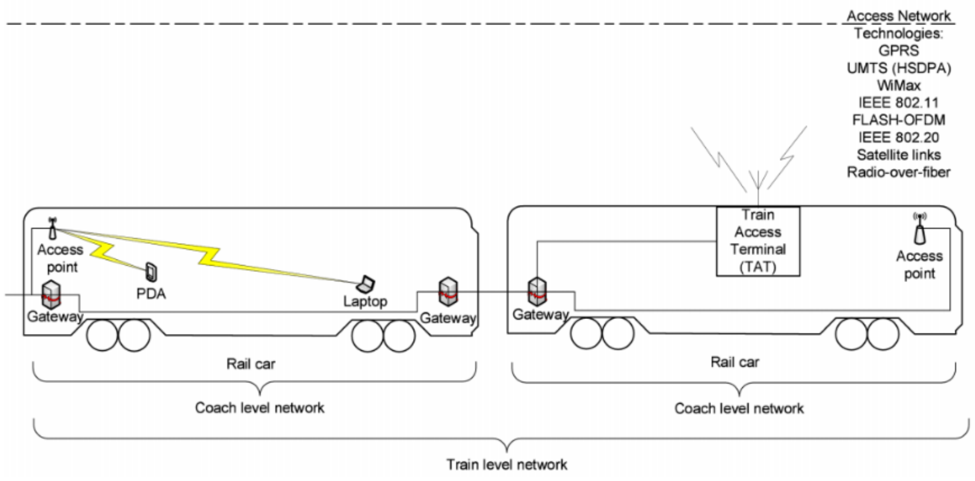

Figure 2.6: Architecture for Internet connectivity in a train, retrieved from [6].

The authors remark some benefits of using this architecture, including:

- The cellular network system is not strained while attempting to make handovers for many fast-moving users simultaneously;
- TAT can combine different access technologies and smartly select a better means for communication between the train and the access network.

Diving into real-world trains' Internet access, the website [22] provides extensive information about how numerous railway companies based in the UK provide Wi-Fi access to their passengers. The web page explains that Wi-Fi is widely available in trains doing long-distance routes but is uncommon for short-distance routes. The tendency is to provide free Wi-Fi, if available.

Comparing the different network architectures described before, it is possible to conclude that the communication inside cars will bring better results if achieved through Wi-Fi (IEEE 802.11), and the communication between cars, if necessary, should be achieved through a high data rate and low latency wireless network (refer to [23]). This is because wiring a train for network access is costly, and rewiring would be necessary in case of reconfiguration of the train topology.

2.2.3 FUTURE TECHNOLOGIES

Technology is constantly changing, and the railway industry will have to evolve with it. Some emerging technologies that could be used for Internet access on trains are analyzed next.

Currently, LTE can provide data rates of up to 1 Gbps [24] (peak data rate in low mobility scenario). 5G networks, in contrast, are able to provide users with data rates of up to 20 Gbps in urban areas [25].

Considering the existent access network technologies, 5G would be a good improvement for train-to-ground communication. It will be based on the foundation created by 4G LTE [24]. This near-future communication generation will increase the capacity and speed of networks in order to provide a more reliable connection to an increasing number of mobile users.

Considered to be an important technology to enable the evolution to 5G, Multi-access Edge Computing (MEC) will provide cloud-computing capabilities to the edge of the mobile network and, therefore, will be closer to mobile subscribers [26]. This technology will reduce latency, ensure higher efficiency in network operation and service delivery, and lastly, provide a better QoS and QoE. Refer to [26] for more detailed information on MEC.

Wi-Fi is the most common WLAN deployment inside trains, using APs for users' connection [4]. However, Li-Fi and WiGig are two suitable candidates for future intra-car communication.

Li-Fi (Light-Fidelity), standard IEEE 802.15, is a Visible Light Communication system. Thus, its frequency spectrum is located in the visible light band and can provide users with a data rate superior to 10 Gbps [27]. One characteristic of this technology is that the Internet cannot be accessed without a light source, which can limit its deployment location since light cannot penetrate opaque physical barriers (unlike radio frequency signals) [28]. This could be an advantage since there is no interference from one room to another; however, other sources of light may interfere with the signal.

WiGig (Wireless Gigabit), standard IEEE 802.11ad, operates at the 60GHz band [4]. It provides high-speed, low latency connections and throughput up to 7Gbps (distance up to 10 meters), which makes it a candidate for inter-car communications.

2.2.4 COMMUNICATION CHALLENGES AND ISSUES

The communication between the train or user and the mobile network suffers from delay, limited bandwidth, packet loss, or even prolonged connection loss. These might be due to:

1. **Frequent handovers:** The high frequency of handovers associated with the high speed of the trains greatly increases the probability of service interruption and overall delay, reducing QoS. The most affected services can be, for instance, streaming or other multimedia-related services that demand both high throughput and a stable connection.
2. **Weak or inexistent cellular coverage:** Companies which deploy public cellular networks' infrastructure need to profit from the network, which is why most recent cellular technologies like 4G are first deployed in places that will provide a better return of investment, such as more populated areas (cities) [29]. So, taking into account that railways are installed in a variety of types of locations (cities, rural areas, among others), Internet access will not always have the same quality during a trip and may even be nonexistent. Trains may travel through remote locations between cities during the trip.
3. **Blocking of radiofrequency:** if the mobile user connects directly to the cellular network, the signal to/from the base station as to penetrate the car, which may result in a loss of up to 24 dB [3], [29]. This issue is tackled when devices communicate within a train's local access network (as described in previous sections).
4. **Number of active passengers:** the more passengers access the Internet inside a train, the less bandwidth will be available for each. Inside a house, a reduced number of people (for instance, four) may be accessing the Internet. In a train, the number of passengers trying to access the Internet may be up to hundreds per gateway. It is observed an increase in the load of the network and bottlenecks may occur in some cars, causing significant service response delay. Moreover, each passenger may have more than one device connected to the network.

Thus, to improve the quality of services inside trains, cellular and railway companies must plan the deployment of the network very carefully, also taking into account the variety of architectures for the distribution link. The choices they make will have a direct impact on the challenges stated above.

In [29], the following scenario of communication is proposed:

- Considering 130 to 180 passengers in each car, if half of those passengers demanded real-time HD video (720p or 1080p), a bandwidth of up to 3.6 GHz would be required. The authors state that this bandwidth requirement cannot be fulfilled by LTE (4G), which only makes use of 20 MHz.

Another possible scenario would be:

- In a train with a total of 300 seats, the train is full and half of the passengers (150) want to watch an HD video from a streaming service, in this case, Netflix. For this video quality, it is recommended by the company to allocate 5 Mbps to each device [30]. LTE has a peak speed of 100 to 1000 Mbps [31], and the 3GPP website states that the peak for LTE is 300 Mbps [32].

Thus, the same conclusion is obtained for this scenario: LTE will not be able to cope with the ever-increasing demand of video streaming (or other high-throughput traffic), not only of train passengers but of all connected users around the world. The key to follow this demand would be, naturally, to switch to faster communication systems like 5G.

There are two ways to tackle communication issues and prevent bottlenecks, assuming a 3G/4G access network, while also providing a good quality of service. First, a railway company may limit the bandwidth and data usage of the passengers. Second, the traffic can be processed according to an assigned priority (for instance, low priority would include streaming services which consume a lot of capacity).

Some railway companies have data usage limits per customer for both downlink and uplink, and sometimes they even block services such as audio and video streaming. A few examples are enumerated next:

- **Trenitalia c2c Limited**: in trains of this English railway company, each passenger has a limited data usage of 100 MB (3G and 4G mobile network) in a day [33].
- **Deutsche Bahn**: a German company, offers different Wi-Fi services depending on the class the passenger travels in. In the first class, passengers may watch videos and listen to music, as well as send and receive large e-mail attachments [34]. Second class passengers, on the other hand, have their data usage limited. The website of the company calls it a basic Internet service, only allowing access to low-throughput websites and services.
- **Eurostar**: according to the company's Terms and Conditions, in [35], the Wi-Fi service "is intended to support general web browsing activities and use of the train's pre-loaded entertainment content". The company's Wi-Fi service does not allow music or video streaming and download of large files. Also, additional speed restrictions may be applied.
- **Queensland Rail**: to ensure a good quality of the Wi-Fi services, in this Australian company's trains, each passenger has a limit of 20 MB of data usage per session [36].

The cost of data transmission is an important aspect when applying the limitations described above. An increasing number of users want to connect to the Internet using the trains' network, which means that more data needs to be transmitted, resulting in an increased cost of the service paid by railway operators [37]. The cost of mobile Internet varies from country to country.

2.3 SOFTWARE-DEFINED NETWORKING

Traditional IP networks are highly dynamic and complex. Thus, managing and configuring them becomes a major challenge, particularly the high-level policies/control functions implementation, since these must be specified in low-level configuration.

Nowadays, network operators must deal with an increasing set of complex network policies and tasks, although only having a limited and constrained set of low-level configuration commands. Each network device must be, therefore, configured with low-level or vendor-specific commands, and because the state of the network changes continuously, its configuration must also change. Ad hoc scripts and external tools may be used to reconfigure the network [38]. However, frequent reconfigurations may lead to errors that may compromise the network.

An additional disadvantage of the management of traditional networks is that current networks are vertically integrated [39]. The control plane, responsible for network traffic management decisions, and the data plane, that forwards traffic according to previous decisions by the control plane are both inside each network device.

Software-Defined Networking (SDN) is a relatively new paradigm that tackles the limitations of traditional networks mentioned above. It provides both centralized control and a global view of the network. This mechanism decouples the control plane from the devices that forward traffic (data plane). Network devices, such as switches and routers, become mere forwarding elements while a centralized controller manages the whole network, simplifying its reconfiguration and policy enforcement and facilitating optimization. Flexibility and resource management is also improved.

In an SDN architecture, forwarding decisions are flow-based instead of destination-based [39]. A flow is, therefore, a sequence of packets forwarded from a source to a destination. The packets are associated with identical service policies in the forwarding devices.

Although Internet access on trains presents numerous issues and challenges, as explained in the previous section, it is still possible to manage the network in order to tackle or minimize their effects. Thus, an SDN architecture can be of significant value for efficient management and control of a train network, while increasing the QoS.

The SDN architecture is divided into three layers [40]: application layer, controller layer (control plane), and forwarding layer (data plane). These layers communicate with each other via APIs. The layers and their communication interfaces are described next, with the diagram of figure 2.7 depicting an overview of the SDN architecture.

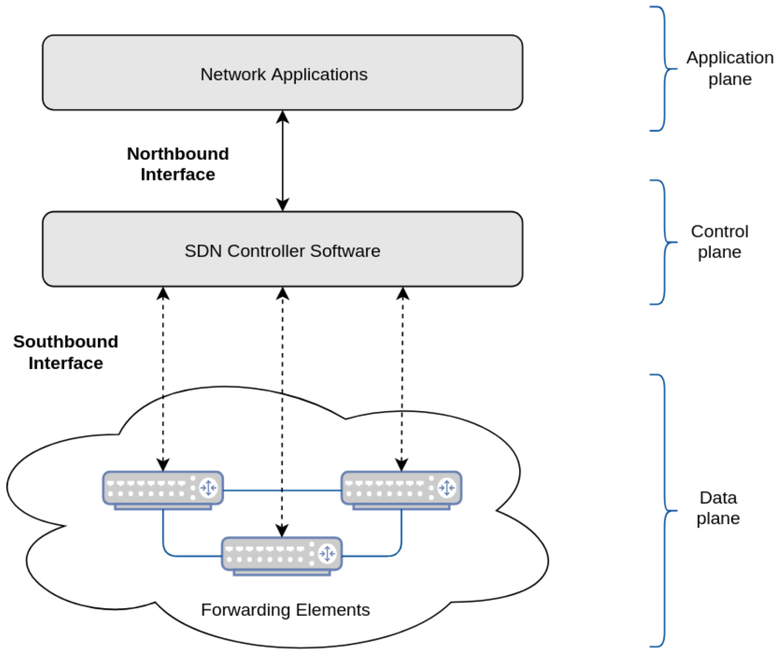

Figure 2.7: SDN architecture diagram

- **Data plane:** contains all types of forwarding devices, such as routers and switches, which are interconnected wirelessly or through physical wires.
- **Control plane:** logically centralized in a server, the control plane uses southbound interfaces to program and manage forwarding devices of the data plane, defining rules and instructions on how they should forward the traffic. A controller is, therefore, the "brain" of the network.

  Some well-known SDN controllers are: OpenDaylight, Floodlight, NOX/POX, Ryu, among others.
- **Application plane:** contains network applications and services that, through the northbound interface, define policies for the control and the management of the network behavior. These applications use the controller to collect information about the network.
- **Northbound Interface:** typically, an API is provided to application developers, to establish a common interface for the development and implementation of management applications and high-level control programs. This interface is used for the communication between the application plane and the control plane. It abstracts low-level instruction sets that are used by southbound interfaces

to program the forwarding devices. RESTful APIs are typically used for the communication between the application and control planes [40].

- **Southbound Interface:** this interface provides a protocol that formalizes the communication and management between the controller and physical/virtual switches, routers, and other low-level nodes of the network. The southbound API defines the instruction set of the forwarding devices. OpenFlow is an example of this interface.

Two interesting works, [39] and [41], explore deeply the SDN concept. In the first one, the authors provide an extensive review of the concepts, challenges, and existing solutions of the SDN architecture and its components. The second contains an overview of programmable networks and examination of the SDN architecture, as well as the OpenFlow standard, which is also explored in the first work and will be described in the next section.

2.3.1 OPENFLOW

OpenFlow is an open protocol that was proposed to standardize the communication between an SDN Controller and the forwarding plane (network devices), and thus control flow transfer by programming the flow table(s) of switches.

A controller makes control and management decisions in a plane separate from the forwarding devices, as explained before. The nodes of the data plane are only required to match the incoming packets and forward them according to predefined rules (established by the controller), simplifying the process [42].

This open-source model removes the limitations imposed by commercial solutions to researchers and supports high-performance and low-cost implementations.

The main components of the OpenFlow architecture are:

- **OpenFlow Switch:** these switches use rules defined in their flow tables (local databases) to forward packets. A flow table consists of a list of flow entries, and each entry has fields such as counters, priority, instructions, match fields, among others.

  Incoming packets are compared against match fields of each entry. In the case of a match, the packet is processed according to the instructions in that entry.

  The use of aggregation schemes, like wildcard rules, helps to deal with limited flow table size. Multiple rules may be aggregated into one, reducing not only communication between the controller and the switch but also the number of redundant flow entries that would otherwise express the same semantics.

- **Secure channel:** The secure channel is the interface that connects the controller to all switches. This channel is used by the controller to manage the switches and also exchange packets with them.
- **OpenFlow Controller:** the controller consists of a software program with the ability to manipulate the switches' flow tables. It uses the OpenFlow protocol to connect and configure the network nodes.

A survey of this protocol and a detailed explanation of its components and capabilities can be found in [42].

2.4 LINK AND TRAFFIC AGGREGATION

Link aggregation is a technique used in a network to combine multiple connections into a single one, thus increasing the system throughput and improving the overall system's performance. Some aggregation solutions found in the literature are described next.

In [43], the authors present an implementation of Link Aggregation Control Protocol (LACP) (IEEE 802.1AX) using the Ryu controller and Open vSwitch. The results demonstrate how this SDN topology can use link aggregation to solve the link failure problem. LACP is a protocol that enables the configuration of link aggregation in traditional switches. It allows the bundling of several Ethernet links to form a single logical channel. It is, thus, an active monitoring protocol enabling devices to add or remove links from the LAG (Link Aggregation Group). A network administrator must perform the configuration manually, and the number of physical interfaces in a bundle cannot be greater than eight [44].

The authors of [45] describe the design of an implementation of a Virtual Aggregation method. This method provides scalability to a network by dividing and distributing a forwarding table over multiple forwarding elements in an SDN network (using OpenFlow and NOX).

The work [46] proposes a service for traffic flow aggregation. Its main purpose is to reduce the number of OpenFlow rules (flow entries) needed in the devices without impacting control logic. After analyzing the results of the tests to the service on topologies specific to the backhaul network, the authors concluded by stating that dynamically aggregating traffic flows into bigger flows would reduce the number of rules in the devices up to 48% (approximately). With smaller flow tables, the devices can process a higher amount of traffic. Similar to this work is [47], where the authors propose using a flow aggregation method to minimize the number of flows, using a heuristic algorithm.

A link aggregation architecture in SDN was defined in [44]. It allows automatic, scalable, and self-adaptive link aggregations. Three link aggregation algorithms were created to evaluate the architecture: Hash Table, Traffic Analysis, and Virtual Round-Robin. They were tested in an environment consisting of an Open vSwitch used for packet switching, and a Ryu controller to control the switches.

- **Hash Table:** upon receiving the first flow packet, a switch will retransmit it to the controller, which will identify its flow fields and calculate a hash, determining which interface should be used to transmit data from that flow. It will then insert a flow entry into the switch's flow table.
- **Traffic Analysis:** the switch sends the first flow packet to the controller, and it will identify the bandwidth utilization of each aggregated interface, to decide about which interface should be used to transmit data from that flow. Accordingly, a flow entry is inserted into the switch's flow table.
- **Virtual Round-Robin:** the controller creates rules for all existent sources and targets using all aggregated interfaces. The amount of rules created is thus decreased, comparing to the two previous algorithms.

2.5 GATEWAY REDUNDANCY

Current network designs must include redundancy of its elements so as to achieve high availability and reliability of the system. Single points of failure should be nonexistent so as not to compromise the network.

One of the most critical elements that should be redundant in a network is the gateway. If a failure occurs in the gateway, communication with external networks is not possible. Inside a train, passengers may be denied access to the Internet due to a faulty gateway and lack of redundancy.

The work [48] explores and compares solutions for default gateway redundancy, focusing on the First Hop Redundancy Protocols (FHRP). These redundancy protocols, which enable the establishment of a fault-tolerant default gateway, are explained next.

The Hot Standby Router Protocol (HSRP) [49] was designed by Cisco to enable two or more routers to create a virtual router with associated virtual IP and MAC addresses. One router is responsible for packet routing (active state), and a backup router (standby state) will take the role of the active router in case of failure. The protocol allows clients to access external networks even when they cannot obtain the IP address of the default gateway.

The Virtual Router Redundancy Protocol (VRRP) [50] also solves the problem of single gateway failure. One virtual router with virtual IP is created. A master router is responsible for forwarding packets that were sent to the address of that virtual router. Multiple backup routers monitor the activity of the master and must be ready to assume its role in case of failure.

Finally, Gateway Load Balancing Protocol (GLBP) [51], a Cisco proprietary protocol, adds a load balancing component to the gateway redundancy, which means it can load balance over multiple gateways. In GLBP, a virtual router uses one IP address and up to four MAC addresses.

A router is selected as an AVG (Active Virtual Gateway), and a standby router is ready to replace it in case of failure. The AVG router will assign different virtual MAC addresses to other routers, named AVF (Active Virtual Forwarder). It will receive ARP requests sent to the virtual default gateway and respond to them by indicating different MAC addresses belonging to the other routers (AVF). As the name suggests, these routers will be used to route incoming packets.

The mechanism of load balancing can be configured using the following techniques: round-robin, host-dependent, and weighted.

2.6 LOAD BALANCING

Load balancing is a technique to distribute the local workload evenly across all nodes in the computer network, improving response time and resource allocation. Using this mechanism, the overall performance of the network may be improved, along with a fair computing resource distribution.

The basis of load balancing algorithms involves one or more nodes in a network that forward traffic to other nodes according to various metrics and the node status (availability, load, performance, among others).

Table 2.1 describes common load balancing techniques. These are widely known primitive strategies that are still used today and inspired the creation of other load balancing algorithms.

Technique	**Type**	**Description**
Round-robin	Static	Each request is forwarded to servers sequentially and circularly. This means that sequential servers will process each request, and when the last server receives a request, the assignment jumps again to the first server.
Weighted round-robin	Static	This technique is similar to the previous, except that a weight is assigned to each server based on its capacity and efficiency to process a request. Therefore, requests are assigned to servers from higher to lower weight value.
Least-connection	Dynamic	This technique will avoid the overload of servers by choosing the server with the least active connections to service an incoming request. Thus, the load balancer must monitor continuously the transactions of the servers in the system.
Weighted least-connection	Dynamic	The least-connection technique is used for load distribution, taking into account a previously assigned server weight.
Random	Static	Requests are assigned to any randomly chosen server of the system.
Source IP hash	Static	The selection of a server is based on a hash of the source IP address of the request.

Table 2.1: Load balancing algorithms

2.6.1 LOAD BALANCING ALGORITHMS

In this section, some load balancing techniques found in the literature are described. These techniques and the ones in section 2.6.2 are suitable to be implemented in a train network (after some adaptations) and comprise various types: static, dynamic, random, centralized, decentralized, among others.

In [52], three distributed load balancing solutions have been identified and described. These algorithms are to be applied in large scale cloud systems [53].

The first is the Honeybee Foraging algorithm. It uses a group of servers that are organized into virtual servers, and these serve a "virtual service queue" of requests.

The second algorithm is Biased Random Sampling. Here, "the load on a server is represented by its connectivity in a virtual graph". This algorithm is implemented using that same graph, which contains as many inward edges as available resources. So, when a node in the system executes a new job, it will remove an inward edge (one less resource), and when it completes the job, it will create a new inward edge (one more resource). This node is the last one in the "sampling walk".

Finally, the third algorithm, Active Clustering, is a self-aggregation load balancing technique that groups similar instances of the network together (similar services) and thus optimizes job assignments.

The authors of [54], propose a decentralized content-aware load balancing technique, named Workload and Client Aware Policy, WCAP. Content-aware algorithms use the content requested as a basis to schedule a request. This technique is to be applied in distributed computing environments (grid, cloud, cluster, among others). This approach considers a USP (Unique and Special Property) for each computing node and request. It helps the scheduler to decide on the most suitable node for the processing of requests.

A suitable node for the processing of a request is searched among a reduced list of nodes with the desired USP, which will improve the performance of searching for a suitable node. Therefore, each node is assigned a request that best fits its capability and specialization, reducing the processing time. At each node, the Round Robin algorithm was used as a technique to process the requests in a queue. According to the authors, the performance improvement of the search for a suitable node increases the overall performance of the system.

In [55], it is proposed a new load balancing algorithm that works by controlling the size of the APs' coverage range (or WLAN cells). Thus, the load of congested APs is reduced by forcing users on the limit of their coverage to shift to less congested adjacent APs, as seen in figure 2.8. This is achieved by reducing the transmission power of the APs.

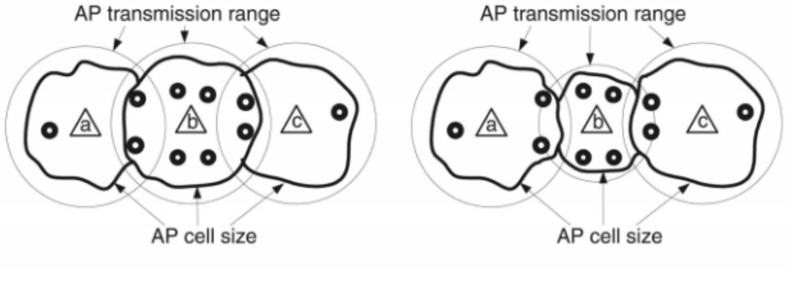

Figure 2.8: Left: APs transmitting with equal power level. Right: AP b adjusts transmission power level. Retrieved from [55].

The authors of [3] propose an architecture for Internet access on trains and designed two load balancing algorithms for it, focused on possible handover events along the route and relief of connectivity loss. These algorithms have the common goal of providing a fair service provisioning, or a good QoS.

The network architecture is depicted in figure 2.9. The dispatchers deployed in each car are a crucial element to the load balancing mechanism. They coordinate the number of packets to be forwarded to other cars when T-RATs are out-of-service and a handover event arises. To exchange packets and thus redirect the traffic, dispatchers are interconnected through optical fiber.

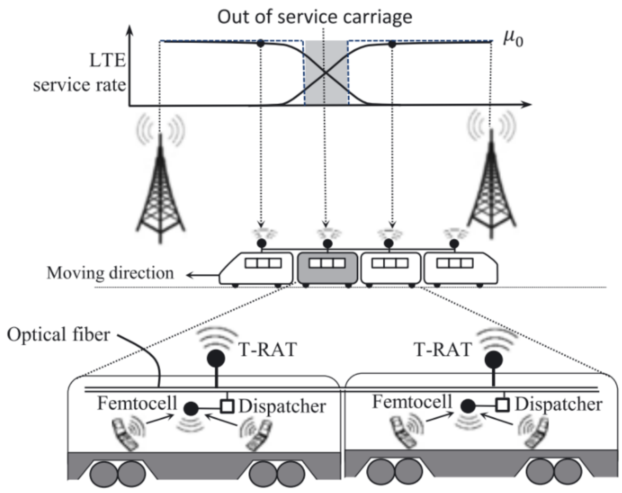

Figure 2.9: Network architecture for high-speed trains, retrieved from [3].

In the first algorithm, Fully Cooperative Load Balancing (FCLB), an out-of-service dispatcher will forward a fraction of “backlog” traffic to every other dispatcher located in different cars. That fraction is the result of a mathematical model described by the authors. There is also an exchange of queue status information among the dispatchers.

The second algorithm, Partially Cooperative Load Balancing (PCLB), is similar to the previous, but here the “backlog” traffic is only forwarded to the dispatchers located on adjacent cars (adjacent to the out-of-service dispatcher). One advantage is that signaling is significantly reduced (one-hop solution).

This algorithm is also described in [16]. The cars’ systems are organized in overlapped groups of 3 cooperating nodes, where the node N will exchange packets with the nodes N-1 and N+1. The forwarding of packets from one node (T-RAT queue) to another is directly proportional to the size difference between both queues.

The authors of [56] propose the Central Load Balancing Policy for Virtual Machines (CLBVM). This load balancing policy contains a central dispatcher and intends to distribute the network load uniformly. It makes the load balancing decisions based on global state information of the system.

Each VM has a unique identification, and multiple VMs are hosted by a server (named pServer). Aggregate CPU load and system utilization by guests is continuously collected. The gathered data is sorted into Heavy (H), Moderate (M), or Light (L) load categories. Messages are exchanged with a Master Server, which takes load balancing decisions periodically. If servers are unevenly loaded, it will balance heavily loaded servers with lightly loaded servers until no further migration is necessary.

The works [53], [57], [58], [59], [60] review load balancing techniques in cloud computing. In addition, the first three compare the techniques in terms of metrics such as throughput, performance, fault tolerance, scalability, among others.

2.6.2 LOAD BALANCING IN SDN

Load balancing in Software-Defined Networking is an approach to network load balancing that eliminates the need for protocols at the hardware level, allowing a higher performance of the network, improved management and status report. An SDN-based load balancer has control over the entire network, which brings some advantages: lower processing time, lower cost, flexibility, greater scalability, and higher reliability.

Diverse load balancing techniques applied to Software-Defined Networking are described in the literature. Some examples are presented next.

In [61], the authors present a load balancing mechanism for SDN where, first, a dynamic least-loaded server policy is used to choose the server that will process a request, and then, the Ant Colony System (ACS) algorithm is applied to find the best path to the least-loaded server chosen. This algorithm is a variant of the Ant Colony Optimization (ACO).

Ant Colony Optimization algorithms include the following steps:

1. Forward ants try to discover new paths to the servers and will gather information on the current state of the existing known paths (forward update);
2. If, meanwhile, one of the ants reaches a destination node, backward ants will follow the previously explored path and reach the source node.
3. During this backward trip, routing tables of the nodes along the explored path are updated (backward update).

By using SDN, both the forward and the backward updates can be eliminated since the SDN controller has a global view of the network.

The authors' proposed architecture consists of a network of OpenFlow switches, virtual hosts, and an OpenFlow controller. The load balancer is implemented as one of the controller's modules. In their proposed ACS-based algorithm, all servers of the system must report their current CPU status to the SDN controller so that it dynamically determines the best-suited server (least loaded).

The authors of [62] implemented a Round-Robin load balancing mechanism for SDN. The architecture consists of an OpenFlow switch network and multiple servers connected to the switch's ports. The chosen SDN controller is POX, and it maintains a list of the live servers.

Upon the arrival of a client request packet to the OpenFlow switch, it will analyze the information in the packet header and compare it against its flow entries. If the information matches an entry, the switch will forward the packet to a server according to the load balancing strategy. If the information does not match an entry, the packet is forwarded to the controller, which will decide what to do with it. The authors compare two load balancing strategies, Round-Robin and Random, using the results of total transactions per second and average response time of the server.

In [63], the authors propose a dynamic algorithm, directed to link utilization optimization in Data Center Networks. The algorithm searches for the shortest paths from a node to the others and computes the link's cost. Also, traffic flows are ordered according to their priority. Dijkstra's algorithm is used to find multiple paths of equal length and thus reduce the search to a smaller area in the network topology. In case of congestion of a path, it will be replaced with the newly discovered best route. This route is, therefore, associated with a minimum link cost and low traffic flow.

The simulation results reveal that the proposed algorithm achieves lower delays, higher throughput, and lower packet loss.

The authors of [64] analyze, evaluate, and implement some SDN-based load balancing strategies, namely: round-robin, weighted round-robin, dynamic algorithm, and flow statistic algorithm. Load balancing parameters like response time, throughput, and availability were utilized for comparison and experimental evaluation of the previous algorithms.

In [65], it is proposed an algorithm that avoids congestion in SDN environments, by rerouting a minimum number of flows and thus reduce the network nodes' overhead.

When a new flow arrives, it will be routed according to the current network condition, which is being monitored by the controller, therefore preventing switch overload and congestion in the network. However, when the average link utilization surpasses a threshold, the controller predicts congestion on the link, then the proposed algorithm chooses a minimum number of flows and reroutes them to the best-suited backup paths. These paths will not be congested after the addition of the rerouted flows.

Regarding the results of the experiments, this algorithm leads to improved throughput and decreased packet loss.

In [66], two algorithms were implemented. The network architecture consists of a POX controller, an OpenFlow switch, multiple servers, and clients connecting to the ports of the OpenFlow switch.

The first algorithm, Direct Routing Based on Server Load, chooses the least loaded server (CPU) to process a request. Servers inform the load balancer about their load, and the one possessing the lowest value is chosen. The second, Direct Routing Based on Server Connection, chooses the server with a minimum of active connections. If more than one server has the same value, the server with the lowest identifier is chosen.

Although both algorithms are similar to already existent ones, they distinguish themselves by the fact that the load balancer is not involved in the return message from the web server to the client. The server will, thus, respond directly to the client, improving performance and decreasing latency.

In [67], load balancing strategies in SDN present in the literature are reviewed and compared. The techniques are divided into deterministic and nondeterministic. It is an extensive and informative survey, describing and discussing the benefits, weaknesses, and challenges of each technique. The work [68] does a similar but less extensive review.

2.7 CHAPTER CONSIDERATIONS

This chapter describes important concepts regarding broadband access on trains, load balancing, SDN, among others. In each section, the related work found in the literature was presented and discussed, and the most relevant and informative works were explored.

First, there was a description of topics regarding broadband access on trains, including access technologies, train network architectures, issues associated with Internet access on trains, and future technologies.

Software-Defined Networking concepts were explained, and its architecture depicted. OpenFlow, an important standard in SDN, was also covered with a brief overview of its components.

Link aggregation and gateway redundancy were covered since they improve the performance and reliability of network systems by increasing their fault tolerance.

Finally, to explore the full potential of a network, its nodes' resources should be used in an efficient and organized way by using load balancing techniques. Since SDN and load balancing are the most relevant areas of this dissertation, load balancing algorithms used in SDN topologies were described.

Internet access on trains presents a great challenge due to limited bandwidth, degradation of radio signals, weak mobile network coverage, among others. These contribute to a decrease in the QoS of the system and thus, decrease in the QoE of passengers. However, a properly-defined network architecture with distributed or centralized management and control of the APs inside the train may attenuate the effects of the referred issues.

3

Solution for Load Balancing in Trains

In the delay commute of many people, Internet access can be a deciding factor when choosing the transportation type. However, equal access to the Internet is not always guaranteed to all passengers traveling in public transportation and the lack of management of the network may lead to a degraded experience.

On section 3.1 it is described an example of network architecture on trains, explored in the literature, and adopted in real-world trains. The issues with this architecture led to the beginning of this dissertation's work, which consists of, generically, new load balancing mechanisms that improve the quality of Internet access to passengers on a train. Two approaches to a load balancing mechanism were developed, in a distributed and a centralized environment. A single point of failure is to be avoided in the gateways of the train, fault-tolerance increased and resource utilization improved.

Section 3.2 describes the functional and operational requirements of the developed systems. Then, section 3.3 contains an overview of the architecture of the system, and section 3.4 describes the system's messages exchange and processes. Finally, section 3.5 proposes a load balancing solution in an SDN environment.

3.1 TRAIN NETWORK SCENARIO

Trains that provide Internet access to their passengers do it by setting up a Wi-Fi service. With this service, the passengers do not need to use their mobile data. As explored in the state of the art, some railway companies limit the data usage of their passengers, but others do not impose that limit.

A typical network architecture consists of placing one or more APs in each car, to which passengers will connect to, and set up a connection to the cellular network or a satellite link. In the adopted network architecture, there is only one AP per car.

As described in the state of the art, the cellular network is widely used for Internet access on trains. However, it poses several issues, mainly due to handover latency and signal quality degradation. These are due to the speed of the train, and also the remote locations long-trip trains usually go through, with inadequate coverage or far from base stations.

For the development of the load balancing solutions, it is considered a network without a centralized or distributed management, where each access router is isolated in the car and does not communicate with other routers or another local or remote entity. The routers never report their network state, and there is no redundancy whatsoever. This network architecture is prone to issues like disconnection of a whole car of the train or unfair distribution of the entire network capacity of the train. The developed solution aims to mitigate these problems.

3.2 REQUIREMENTS

Features of a system are developed according to predefined system requirements. These requirements encompass mainly system and management demands.

First, the system's functional requirements will be described. These are a description of the service(s) that the software system must offer. Then, nonfunctional requirements highlight conditions that can be used to analyze and evaluate the operation of the system.

It is crucial to note that these requirements do not solve the majority of the problems inherent to the type of Internet access on trains detailed in previous sections. However, they are needed to develop the system in a way that improves the negative effects of those problems.

3.2.1 FUNCTIONAL REQUIREMENTS

The systems must be developed following guidelines and rules, in order to be functional during their execution and in anomalous situations. Next are described some functional aspects regarding the load balancing systems and respective networks.

- **REQ-1:** The load balancing mechanism must be aware of the existing active access routers.

  For the correct operation of the system, nodes of the system must be aware, at all times, of the currently operating routers and the ones who are not active. In the centralized solution, only the controller needs this information.

- **REQ-2:** Statistics collected in each access router are reported to other nodes in less than one second.

  A train's speed is, most of the time, below its maximum speed and when arriving at stations, it slows down significantly until the train stops. The collection of statistics in the routers and their communication needs to be updated regularly in other nodes. Considering a train with 150 meters of length and 6 cars, with an operating speed of 250 km/h. The time it takes for the router inside car N to arrive at the position of car N-1 is exactly 360 milliseconds. On the other hand, if the train's speed lowers to 100 km/h, that value increases to 900 milliseconds. Thus, statistics should be collected and reported to other nodes in less than one second.
- **REQ-3:** Messages exchanged for reporting of statistics, between an access router and other nodes of the network, should only contain network information. This is to avoid operation overhead.
- **REQ-4:** The report of rapidly changing metrics and the forward of traffic (due to load balancing) should use high-speed links for the least latency on communication.

  The connection between the routers and between the routers and the SDN controller should be either a wired high-speed link, a wireless high-speed link (like 60GHz Wi-Fi), or the combination of both.
- **REQ-5:** NAT of the passengers' traffic should be performed only in a remote aggregator node.

  If NAT is performed in the gateway of each router, which is what happens in the proposed train scenario, the forwarding of traffic between congested and backup routers would result in a break of the TCP sessions.
- **REQ-6:** Connection to the aggregator node should be ensured by, at least, half of the existing access routers.

  The aggregator is crucial for the regular operation of the whole network system, and at all times, the routers that are operating normally or serving as backup must have an active link to the aggregator. The number of links must be at least N/2, where N is the number of existing routers, considering that one congested router must forward traffic to one backup router, and one backup router will serve one and only one congested router.
- **REQ-7:** The software developed must be compatible with the access routers' OS. The operating system used will be looked into in more detail in chapter 4.

3.2.2 NONFUNCTIONAL REQUIREMENTS

The following requirements are used to evaluate the usability and operation of the load balancing systems.

- **REQ-1:** All access routers must be accessible and manageable through the SDN controller machine.

  The ability to manage the routers remotely is a centralized feature that makes it easier for a system administrator to manage the train network elements.
- **REQ-2:** An improperly configured access router should not compromise the operation of the load balancing system.

  The system cannot allow an incorrectly configured router to compromise the load balancing algorithm. The router should be left out of the system.
- **REQ-3:** Each access router can forward its passengers' traffic to another router.

  A network connection must be established between the routers so that, in the event of congestion in a router, it is possible to send its passengers' traffic through a communication link to a backup router.
- **REQ-4:** It must be possible to add or remove access routers in the train network, without compromising the load balancing mechanism.

3.3 NETWORK ARCHITECTURE OVERVIEW

The diagram of figure 3.1 depicts the overview of the proposed software-defined network architecture for trains, used in the centralized solution. Then, the components of the system are described.

The network architecure is divided in three main layers. First, the client network layer, containing the wireless networks created in every car of the train (one per car). Then, the internal network layer, that allows communication between the nodes that provide Internet access to the passengers and the control nodes. Finally, the remote layer, consisting of the aggregator, which receives all the traffic of the passengers and forwards it to the Internet.

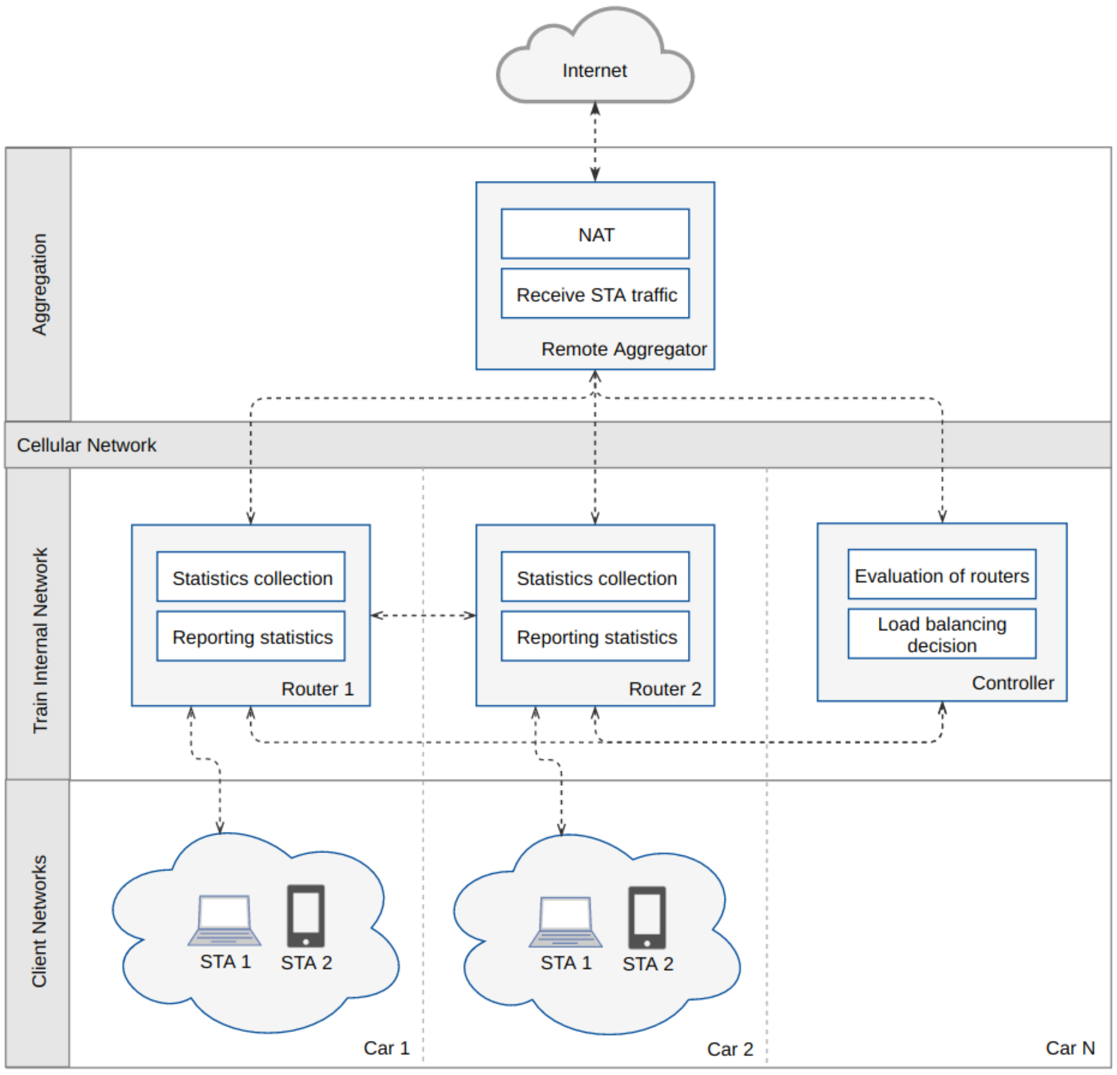

Figure 3.1: Diagram of the network architecture

Each element has its functionality and purpose on the system. Without one of these elements, its normal operation is not possible.

Access Router

The access router is responsible to create the wireless access network inside a car. The passengers will then connect to the Wi-Fi service and start using web services.

As explained in the proposed network architecture in section 3.1, there is one router placed in each car of a train. This router will connect to the cellular network. A train travels through different locations, sometimes distant from cities, which have increased capacity and thus increased available bandwidth. The router may have, sometimes, no connection to the cellular network, as some locations have no coverage of the chosen operator(s).

The traffic will be aggregated in a remote entity before being forwarded to the

Internet.

In summary, passengers are connected to the access router of their car, which forwards the traffic to a gateway to the cellular network, reaching the aggregator entity afterwards.

Station

A Station (STA) is a device that is capable of using the 802.11 protocol. Inside the train, they correspond to the mobile devices used by passengers: smartphones, laptops, tablets, among others. Passengers connect them to the train network to avoid using costly data services provided by their mobile network operator.

SDN Controller

The controller in an SDN environment is the element where the control logic is centralized, and thus has direct control over the data plane elements, through a southbound interface.

The SDN controller located on the train should be incorporated in the load balancer, controlling the forwarding of passengers' traffic according to load balancing decisions. This controller has an interface to an internal network of the train. This internal network allows the traffic to be forwarded between the routers (load balancing event), and the exchange of control messages between the controller and the routers.

Since the controller is also the load balancer of the system, it will evaluate routers to decide on which ones are congested and decide on the backups that should be assigned to them.

Aggregator

This remote network element aggregates the routers' traffic. As explained before, NAT should not be performed on the routers' gateway, because each car has its network and the forwarding of traffic from a congested router to a backup would break the TCP connections to the external network. So, each router sends the aggregated traffic of its car to the aggregator which will then apply NAT on its gateway interface.

The aggregator must know at all times through which tunnel to send responses because, in the event of load balancing, it is crucial to send responses destined to a congested router through the backup router's tunnel. So, if traffic from a car arrives at the aggregator through the tunnel of another car, the forwarding plane of the aggregator must be updated so that the response is sent through the tunnel the request was received from.

This node may also be used by a network administrator for diverse purposes, like managing and controlling the routers and the passengers' traffic. However, its full potential is not explored in this dissertation and should be covered in future work.

3.4 MESSAGE FLOWS

The sequence diagram of figure 3.2 illustrates the messages exchanged between the network elements of the system and processes, providing a better understanding of each nodes' purpose and how they must act in the system. The most relevant interactions of the system are described below the diagram. The processes described here will be detailed in section 3.5, which explains the load balancing mechanism in depth.

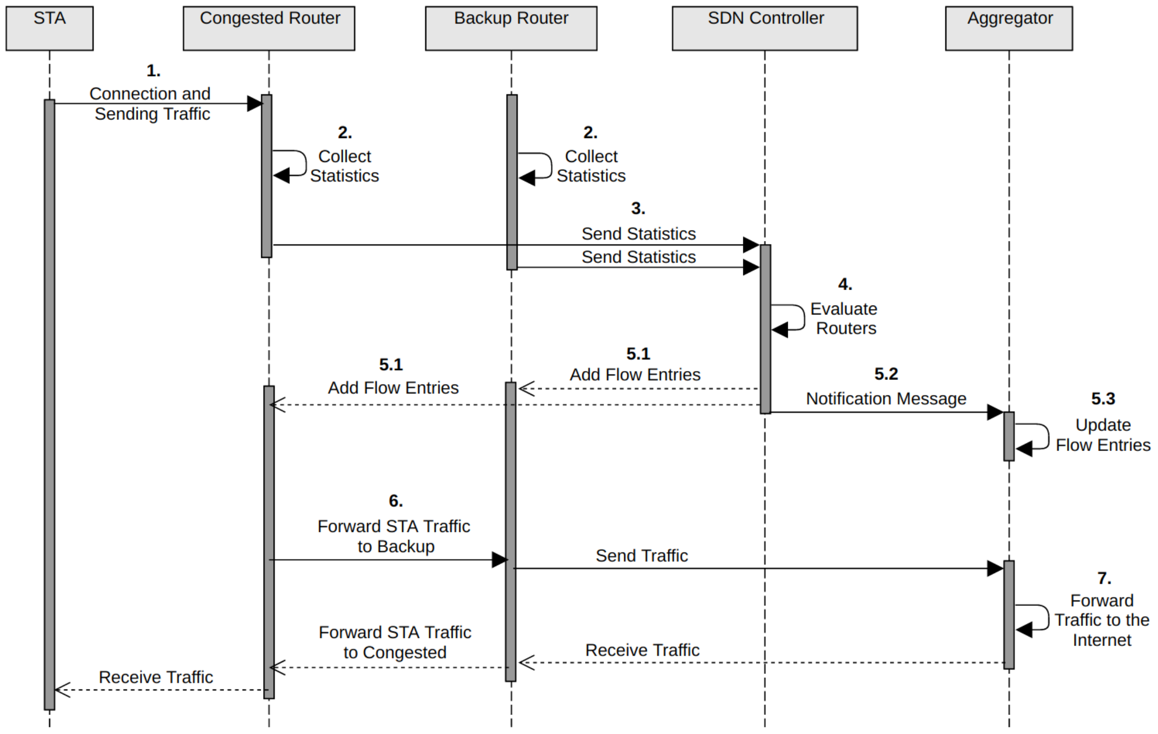

Figure 3.2: Sequence diagram of the message flows in the system

1. Connection to the access routers

Multiple STAs inside a car will connect to the network created by the routers, and start using web services.

2. Collection of statistics

Information should be collected in all interfaces of the router. The gateway statistics will provide the latency value, while the car network interface statistics provide the value of the TCP retransmissions. These are the two statistics to be used in the evaluation process.

3. Report of statistics

The statistics must be sent to the controller/load balancer so that it can evaluate the routers' network status. This communication should occur through a messaging queue system, which must always be operational, else a router may be left out of the load balancing mechanism.

4. Evaluation of routers

Once received the statistics of each router, the load balancer will create the pairs of congested and backup routers, according to the evaluation calculated for each router.

5. Addition of forwarding rules and notification of the aggregator

When a congested-backup pair is processed, the controller should communicate the new forwarding rules to the routers. The SDN controller should be aware of all existing switches (inside each router) that it can manage, so that it can install forwarding rules in them and do other management operations.

The controller will send a notification to the aggregator informing about the new forwarding rules of the routers. Upon receiving the notification, the aggregator will change its own forwading rules to match the train's network topology.

6. Redirection of traffic to backup router

Both congested and backup routers have new forwarding rules commanding the balancing of the traffic between them. From this moment on, the aggregated traffic will be forwarded to the backup router, except traffic to blacklisted destinations in the case of version 2 of the centralized solution.

7. Aggregation of traffic

The traffic from the routers reaches the aggregator, and NAT is applied in its gateway. The aggregator, upon receiving a notification from the load balancer, or when its controller notices changes in the traffic sources of the incoming tunnels, must update its forwarding rules. This is so that responses to the congested router are sent through the backup tunnel. It quickly becomes the most critical process in the system because an outdated rule may send the responses through the wrong tunnel.

3.5 LOAD BALANCING SOLUTION IN A SOFTWARE-DEFINED NETWORK

Diverse load balancing techniques are used in data centers and other cooperating environments in order to utilize the system's resources fully. Some of these techniques were explored in this document's state of the art. The developed distributed and the centralized load balancing mechanisms are similar to a weighted algorithm, in the sense that load is distributed to routers according to their capacity and efficiency to process traffic, which translates into an evaluation value. Every machine may have a different role in the system, depending on the current state of their network interfaces. Machines with a higher evaluation, or in other words, evaluated as more congested than others, will have a chance to assign their load to another machine. The contrary happens to lower evaluation machines, which have a high probability of being chosen as a backup for another router's load.

In this section, the centralized solution's load balancing algorithm is explained in detail. Note that some steps of the process are similar between the distributed and centralized solutions.

The centralized load balancing mechanism is based on the reporting of statistics from the access routers to a controller on the train. In the first version of this algorithm, all traffic is forwarded to the backup router. In the second version, traffic to blacklisted destinations is dropped, while the other portion of the traffic is forwarded to the backup. The foundation of both algorithms is the same, and the difference can only be seen when installing the forwarding rules in the congested router, as will be explained later in this text.

Statistics collection and reporting

Each router collects network statistics of its interfaces periodically, with a high frequency. Then, an asynchronous messaging queue is established between the routers and the controller and the routers send the collected information to the controller, also with a high frequency. This is so that the load balancer has always the most updated information about the routers it manages. Each router will thus, publish the messages containing the statistics information and the controller, the subscriber, will receive those messages.

Evaluation of routers

An evaluation program containing the load balancing logic, located in the controller, will store the statistics values received from the routers, and divide the ones relevant for the load balancing algorithm by their maximum value. In this algorithm, the latency and TCP retransmissions are the statistics used for the evaluation. The maximum value of the TCP retransmissions is its highest value to date in all routers. However, the latency measurement must have a stipulated limit, corresponding to the maximum latency in the connection between the routers and the aggregator. These values are, thus, affected by the network conditions and amount of traffic generated by the passengers. This solution is the most adequate since it is not possible to establish a theoretical limit for the two statistics.

A decimal weight value from 0 to 1 is predefined for each statistic that takes part in the evaluation and may be tweaked by the system administrator to better adapt to the usual traffic of the train. The sum of the weights must be 1.

Then, the values of the evaluation statistics are divided by their maximum value and the obtained results are multiplied by their corresponding weight. At the end of the calculations, the values obtained from this multiplication for all statistics are summed, and the result is multiplied by 100. The evaluation of each router is, thus, a value between 0% and 100%, providing an objective measurement of the congestion status of the router.

A router is labeled as congested if the subtraction of its evaluation with the evaluation of any other router is equal to or greater than a predefined number, the "difference" value. Therefore, the "congested" label does not necessarily mean that there is a bottleneck in the network or that the router is out-of-service. It does mean that there is another router in the train with better network conditions than the congested router, and that could provide a better QoE to its passengers. So, if the subtraction between router A's evaluation and router B's is a positive number equal to or above the difference value, B becomes a backup candidate for the traffic of A. What this means is that B is in a state where it can withstand A's traffic and even though its evaluation may also be elevated, it can assure that A's passengers have access to the Internet by sharing its bandwidth and may even obtain lower response times in the accessed web services.

The difference between the evaluations of each router also determines the frequency to which the system administrator wants the system to trigger the load balance. For instance, choosing a high difference value means that the load balancing is triggered only when a router has both statistics close to their limit, and there is at least one router with a significantly low evaluation. This is likely to happen if the first router has a very weak or no Internet connection, which means the load balancing mechanism would be used as a gateway redundancy technique. It is important to notice, however, that the difference should not assume values that are too low, for instance, less than half of the weight assigned to a statistic, to prevent false positives (refer to the testing scenario in section 5.2.4). The system administrator must find a balance for this value.

Backup assignment and load balancing

The routers and their backup candidates should be sorted in descending order of their evaluation. Their position becomes their priority order. Each congested router is sequentially assigned to a backup candidate. If it has multiple candidates, it must choose the one with the highest evaluation, or the one with the lowest evaluation if no other router has it in their candidates list. For the explanation of this design choice, consider a scenario with routers A, B, C, and D, sorted in descending order. Router A (congested), the one with the highest evaluation, has two backup candidates, C and D, and router B (congested) as only D as its backup candidate. If A does not choose its backup candidate with the highest evaluation (C), B will not get a chance to load balance, even if its evaluation is similar to A's. Finally, only one congested router gets to hold a backup candidate that was chosen by multiple routers.

The following figure 3.3 illustrates the backup assignment process that all routers go through.

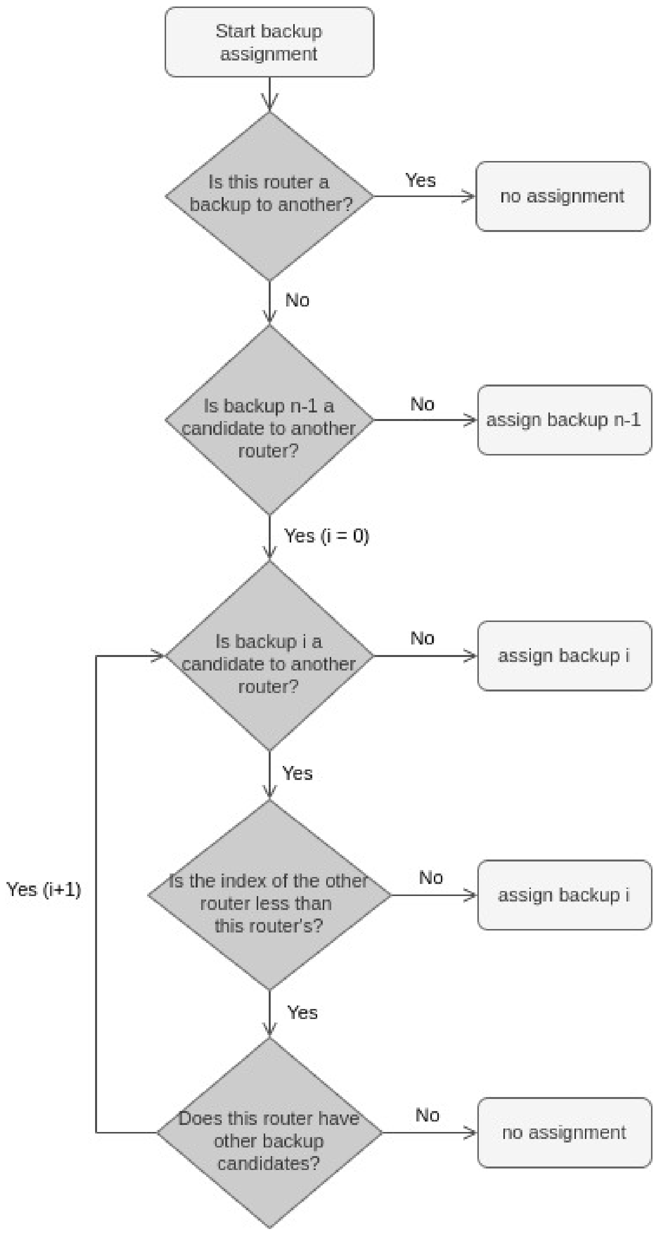
Start backup assignment
Is this router a backup to another?
Yes
no assignment
No
Is backup n-1 a candidate to another router?
No
assign backup n-1
Yes (i = 0)
Is backup i a candidate to another router?
No
assign backup i
Yes
Yes (i+1)
Is the index of the other router less than this router's?
No
assign backup i
Yes
Does this router have other backup candidates?
No
no assignment

Figure 3.3: Flowchart of the backup assignment

With the congested and backup pairs discovered, the load balancer may now send messages to install new forwarding rules in the switches (inside each router). From this point on, the load balancing should be divided into two versions:

- **Version 1:** congested routers will forward all their traffic to the respective backups. After the installation of the forwarding rules (or flow entries), a congested router will send its aggregated traffic to the chosen backup, and the backup must forward the response packets to the congested router.
- **Version 2:** congested routers will forward their traffic to a backup router, except for traffic with blacklisted destinations, or blacklisted web services, which will be dropped. The list must contain destinations that are typically associated with high bandwidth-consuming web services. Therefore, this traffic is discarded in order to improve the QoS of the remainder of the traffic.

Aggregation of traffic

The traffic from the routers reaches the remote entity, or aggregator, through tunnels of a specified protocol. The load balancer will send a message to the aggregator, in a request-reply pattern, informing it to update its forwarding rules, so that response packets to the congested routers are sent through the backups' tunnels instead.

Load balancer operation

The forwarding rules added to the switches inside the routers should not be removed until the router loses its congestion status or is obligated to choose a different backup router. An iteration of the load balancer should give enough time to allow the installation of the load balancing forwarding rules and to update the aggregator. The connection between a backup and the aggregator must always be ensured; however, the backup may not have good connection quality in a particular location of the railway. So, the chosen iteration time should allow the decisions made by the load balancer on the train to be reflected in the aggregator before a new iteration starts.

If, in the following iterations, a previously congested router loses its status, the load balancing forwarding rules are removed from the routers. If, on the other hand, it chooses a new backup, the outdated flow entries are replaced with new ones. If a previously congested router returns to its normal state, it may serve as a backup router to others or keep on routing only its passengers' traffic.

The evaluation of the routers must be performed regularly so that the service quality experienced by the passengers is improved and not compromised.

4 Implementation

The solution design was extensively described in the previous chapter. This chapter proposes an implementation for the distributed and centralized solutions.

The centralized load balancing system aims to balance the passengers' traffic in a train, between the existing routers, with centralized management of the switches inside each router and respective forwarding rules. The distributed solution also load balances traffic between active routers, but the management and load balancing decisions are distributed by the network nodes. A virtualized network was created to test the performance and efficiency of the proposed load balancing mechanisms. The load balancing mechanism in an SDN environment may later be adapted to be used in a physical implementation of the system. The software was built so that the exchange of messages between the network nodes occurs in the least time possible, without overwhelming the CPUs. This is because, with the elevated speed of the train, network conditions may vary frequently.

In the centralized solution, each access router, or in this case, Virtual Machine (VM), has a virtualized switch (Open vSwitch) that can be controlled and managed by an SDN controller, located in another VM. The router VMs send statistics information to the controller through a message queue. Upon receiving it, the controller will evaluate their status of congestion and decide about the congested-backup datapath pairs. When the decision is made, it will add to the flow tables of the switches, new flow entries with higher priority than the ones already installed in them, using the southbound protocol Openflow (version 1.3). The construction of the flow entries depends on information collected about tunnel ports using the OVSDB protocol.

The traffic of the clients reaches the aggregator through GRE tunnels and, during normal operation, it simply forwards the traffic to the gateway. In the event of load balancing, the aggregator must first update its switch's flow entries accordingly, when it receives a notification from the controller on the train or when a "packet in" message is received in its controller (using the Ryu API and Packet library).

In the following sections, the decisions taken during the development of the solution and the implementation design are described in depth. In section 4.1, the technologies used are listed, and their purpose in the system described. Next, the implementation of the distributed load balancing system is explained, on 4.2, and finally, the centralized solution and associated processes, on 4.3.

4.1 OVERVIEW OF USED TECHNOLOGIES

The technologies and protocols used in the developed systems are described next, with an explanation of why they were chosen instead of other similar solutions and what is their purpose in the operation of the load balancing mechanism. First, a diagram of the elements of the system and respective technologies is depicted in figure 4.1.

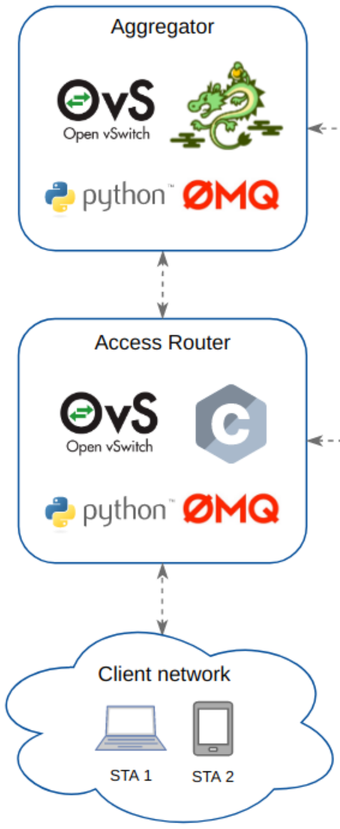

Figure 4.1: Network elements and related technologies

Open vSwitch

Open vSwitch (OVS) is a "multilayer software switch licensed under the open-source Apache 2 license" [69]. It is used in hardware virtualization environments and supports multiple network standards and protocols.

The OVS has diverse components and tools that make it easier to manage and control the switch, like `ovs-vsctl`, to update and retrieve the configuration of the switch daemon and `ovs-ofctl`, a tool for monitoring and controlling the switches, that also allows managing of the flow tables.

OpenFlow v1.3

Openflow is an SDN standard and a southbound interface. It defines the communication protocol in an SDN environment, enabling the controller to manage the network devices of the data plane.

The OpenFlow version used in the solution is the v1.3 [70], which is widely supported and the most used version currently [71, p. 118].

Open vSwitch Database

The Open vSwitch Database (OVSDB) management protocol [72] allows programmatic access to the OVS database holding the configuration of an OVS daemon. Although OpenFlow is used to add or delete flow entries (for instance), OVSDB is needed in order to configure the switch itself.

Ryu

Ryu is a Python-based SDN controller that provides an API for the creation of management and control applications directed to software-defined networks. It supports Openflow v1.3, which is the Openflow version used in the systems.

Ryu was the technology chosen for the SDN controller in the system because the majority of the modules of the solutions was also written in Python. It also provides a well-performing and well-documented API.

OpenWrt

The OpenWrt Project [1] is a Linux OS with a fully writable filesystem and package management, that is targetted to embedded devices, and is especially popular in wireless routers. Using this OS, devices with vendor-specific configurations become fully customizable. The router VMs of the virtualized network use the OpenWrt 18.06.4 release.

This OS uses the Unified Configuration Interface (UCI) system [73], which aims to centralize the configuration of the device and other OpenWrt services. It is an integral

[1] `https://openwrt.org/`

part of the distributed solution but was not used in the SDN environment. Its usage is relevant for the distributed algorithm because it allows controlling the forwarding of traffic in the router VMs.

ZeroMQ

ZeroMQ [2] or ØMQ, is an asynchronous messaging library that provides a message queue service to send and receive messages without the need for a broker.

It supports messaging patterns such as publish-subscribe, request-reply, client-server, among others. For this dissertation, only the first two are relevant. Although it offers diverse types of transport, only TCP was used.

GRE

Generic Routing Encapsulation (GRE) [74] is a tunneling protocol developed by Cisco Systems [3]. It is used in the establishment of a point-to-point or point-to-multipoint secure connection between network nodes, through a public network. It can encapsulate a diverse range of network-layer protocols.

GRE tunnel ports were set up in each switch to forward traffic between the access routers or between them and the aggregator. The tunnels are necessary since the virtual switch does not have control over the physical interfaces that are not added as ports to it, which is the case in this solution. With the tunnels set up, the forwarding of passengers' traffic may be controlled with flow entries.

4.2 DISTRIBUTED LOAD BALANCING

Before developing the centralized solution, it was decided that this work would benefit from an initial load balancing solution to test its reliability in an emulated train network. This is the distributed load balancing mechanism.

A fully functional solution based on routing tables manipulation and default gateway modification was developed, with a load balancing algorithm similar to the one used in the centralized (or SDN) solution and explained in chapter 3, section 3.5. The most significant difference between the two is that this solution is distributed, whereas the SDN-based mechanism is centralized. It is categorized as distributed because routers send statistics information between them and can evaluate each other independently. They each come up with the decision to balance their traffic or not, without the need for a central entity to control the forwarding. Thus, the load balancing decision results from a mutual agreement between the congested and the backup router.

[2] `https://zeromq.org/`

[3] `https://www.cisco.com/`

The positive performance testing results gave the green-light to the use of this type of load balancing algorithm in the centralized solution, which had to be adapted to fit an SDN environment (refer to section 4.3).

The modules and processes associated with the system's operation are explained next.

Discovery

There is an internal network shared by all network elements of the train. Each router must have information on the active peers through that same network, more precisely, the IP addresses of their network interfaces connected to the internal network. This information is of uttermost importance because it is used to set up the message queues for the communication of statistics information between the routers.

The network discovery module executes multiple `ping` commands to the host IP range of the internal network, every 30 seconds (interval can be changed for closer to real-time discovery). The routers that respond to the ping will have their IP addresses stored in a file. At the end of the scan, the file will contain a list of the IP addresses of all active routers.

In the same module, for every router discovered, a static route to its car network is stored in the `/etc/config/network` file, using the UCI. This step is necessary for the forwarding of traffic between routers to be possible.

Statistics Collection

The network statistics of routers are used for the their status evaluation. The information is collected by a main statistics collection module and a module that counts TCP retransmissions.

The main module aggregates information about each interface and stores it in a file, in the form of a list of JSON objects. This format will simplify not only the communication operation but also the loading of the information in each router. This is due to the fact that the evaluation module/load balancer is also written in Python, which has a JSON API that facilitates the extraction of multiple JSON objects from a list.

The following information is collected by the module, for each interface:

- Name of the interface, its MAC, IP address and subnet mask;
- Number of received bytes and packets received but dropped. Collected from the `/sys/class/net/<if>/statistics` [75] directory;
- Number of transmitted bytes and packets dropped during transmission. Also collected from the `/sys/class/net/<if>/statistics` [75] directory;

The number of TCP retransmissions and the round-trip delay time, measured with the `fping` [4] utility, are only collected for the car network's interface and for the gateway, respectively.

All statistics are measured every 550 ms, and the fping command has a timeout of 500 ms. The information stored in the output file corresponds to the difference between the current values and the values of the previous iteration. An iteration of the program lasts 550 ms because if the fping reaches its timeout, the bonus 50 ms will be a comfortable margin for the execution of the other functions of the program. The load balancing mechanism does not use all the statistics for the evaluation of the routers. However, this extra information allows the system administrator to have a summary of the status of the interfaces remotely.

The interfaces surveyed are, thus, the one connecting to the cellular network, the one the clients connect to, and the internal network interface. The information is stored in a local file, and each JSON object has the following generic format:

```
 {
    "name_if"   : <interface name>,
    "IP"        : <IP address>,
    "netmask"   : <subnet mask>,
    "MAC"       : <MAC address>,
    "rx_bytes"  : <# received bytes>,
    "rx_dropped": <# received but dropped packets>,
    "tx_bytes"  : <# transmitted bytes>,
    "tx_dropped": <# packets dropped during transmission>,
    "retrans"   : <# TCP retransmissions>,
    "latency"   : <round-trip delay time>
}
```

The module counting the TCP retransmissions going through an interface captures TCP packets and adds their information, sequentially, to a circular buffer. The information stored is the packet's source IP, destination IP, source port, destination port, sequence and acknowledgment numbers, and flags, which altogether characterize the packets from the STAs. Then, every time a new TCP packet appears, its information is compared against previously stored packets. If all fields match between a pair, a retransmitted packet was found, and the retransmission counter increments its value.

The number of retransmissions since the start of the execution of the program is continuously saved in a file, which will be used by the statistics collection module to calculate the number of retransmissions every 550 milliseconds. The module uses the C/C++ libpcap [5] library, which is used for network traffic capture.

[4]`https://fping.org/`

[5]`https://www.tcpdump.org/`

The developed circular buffer has a maximum size of 50 entries (arbitrary number), but this value can be modified to fit the traffic of a real train. If the traffic volume is elevated, the circular buffer should have a large dimension, in order to increase the probability of finding a matching packet.

Communication of Statistics

For the load balancing to be successful, routers must share updated statistics information between them.

The communication channel chosen uses the ZeroMQ messaging library in order to create a message queueing system through which routers can communicate asynchronously. There is no message broker in the communication, and publish-subscribe was the pattern adopted (refer to figure 4.2).

Every router is a subscriber and a publisher at the same time. A router will subscribe to all the other routers' messages being published and will, itself, publish messages that will be consumed by other routers. The messages are sent from the publisher every 250 ms.

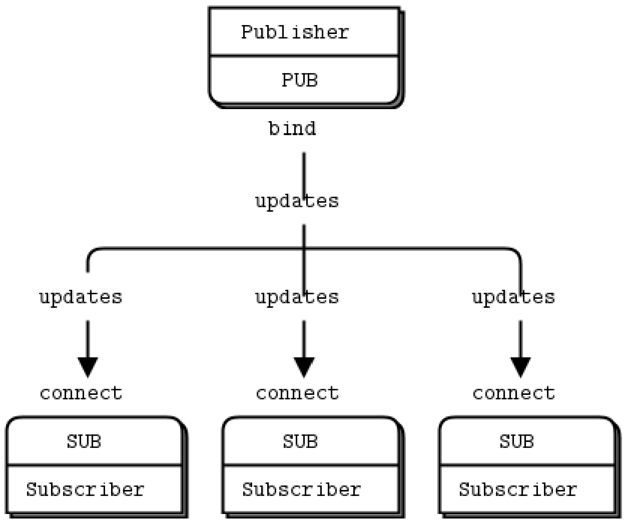

Figure 4.2: Publish-subscribe pattern, retrieved from [76]

In the publisher and subscriber modules, a high water mark was set for each socket. The high water mark value is a limit for the maximum number of pending messages that will be queued in memory for each peer the socket is communicating with. This limit acts as a security measure that prevents the system from running out of memory and crash.

The publisher module will bind to the internal network interface's IP address, read the statistics file that is continuously being updated by the statistics collection module, and publish the information contained in it. Although the statistics are only updated every 550 ms, each message is being sent from the publisher with a periodicity of 250 ms. Since the two modules (statistics collection and their communication) are not synchronized, the frequency chosen means that the statistics of a router A stored in a router B are different from the origin for no longer than 250 ms. This inactivity time in the ZeroMQ module is needed so as not to overload the CPU.

The subscriber module will receive the messages and store the information in a different file per router, in a predefined directory. This module operates in a blocking mode, which means it must receive a message to continue its execution. The files' information will then be loaded to the load balancer module.

Evaluation and Load Balancing

The load balancing module, located in each router, is based on the evaluation of a router's network status and comparison to other routers'. When a router self-diagnoses as congested, the module will control the forwarding of traffic accordingly.

The load balancer in the router will read the statistics information from the continuously updated database and evaluate all routers, including itself, based on that information. The variables used for the evaluation are the round-trip delay time and TCP retransmissions. In a real-world implementation, other statistics may also be featured in the evaluation mechanism.

The maximum value of the TCP retransmissions is its highest value to the date, on every router, and the maximum latency is 500 ms. For the explanation of how the evaluation value of each router is obtained, please refer to chapter 3, section 3.5.

The load balancer in a router will subtract all routers' evaluation to its own, and if the result is equal to or greater than a predefined limit, the router becomes a backup candidate. If the router has no backup candidates, it is not congested and can serve as a backup to other routers. Otherwise, it is labeled as congested and may start the load balancing process.

The backup candidates list is sorted by the routers' evaluation, in descending order. The load balancer processes the list so that the router tries, in a loop, to load balance to the available routers. The backup chosen in an iteration may notify the load balancer that it is already serving as a backup to another router, and so the load balancer processes the next backup candidate, and so on. If, in the end, no router is available for load balancing, no backup assignment is made and the load balancer will restart from the evaluation process, after two seconds.

If there is a backup available, for the forwarding to succeed, a secondary routing table must be established in advance. This routing table will allow the traffic to be routed to a new default gateway, which is the IP address of the internal network interface of the backup router, while also allowing latency measurements in the interface connected to the cellular network.

Before the execution of the load balancing, the program must verify if the router itself is not serving as a backup to another router. If it is, the router is prohibited from continuing with the execution of the load balancing process. Otherwise, an `ip-rule` [6] is added to match the traffic coming from the car network and redirect it to the secondary routing table, which has the new default gateway.

The forwarding rules of the firewall are modified so that the traffic is forwarded between the router and the backup. This is achieved by using the UCI system. For forwarding to happen between two interfaces, two firewall rules are necessary, with interface A as the source and interface B as the destination and vice-versa. In the system's firewall configuration file, however, there must be four forwarding rules (or sections), so that a backup router forwards its own traffic to the gateway and also the congested router's traffic.

If in the following iterations of the algorithm, the router loses its congestion status, the above actions are reverted, and the primary routing table is used instead of the secondary one. The firewall returns to its original configuration, and the traffic is, once again, forwarded between the car network interface and the gateway of the router.

If a backup chosen in an iteration of the load balancer is still a candidate in the following iteration, the load balancer will assign it once again to its router.

The congested router cannot balance its traffic without first being authorized by the backup router, as will be explained next.

Authorization and Forwarding

Routers cannot forward traffic to other routers as they please. One of the main reasons is that, even though a backup may be in the candidates list of a congested router, it might already be serving as a backup to another router. The backup router would most likely become congested if two (or more) routers were redirecting their traffic to it.

So, an authorization mechanism was added to the load balancing module, based on a request-reply pattern (refer to figure 4.3). It requires the congested router to send a forwarding request to the backup which will answer with an "ok" or "not ok".

[6] `http://man7.org/linux/man-pages/man8/ip-rule.8.html`

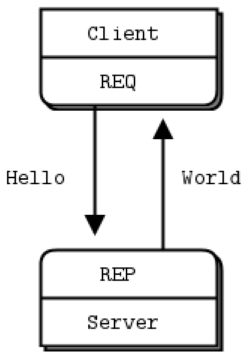

Figure 4.3: Request-reply pattern, retrieved from [76]

First of all, a communication channel is created with ZeroMQ. The congested router will send a message of forwarding request to the backup router and wait for a response, which will determine if the congested router is allowed or not to forward traffic to the backup. If it is, the router can modify its forwarding rules, as explained previously, and start forwarding the traffic.

Upon receiving a request message, the reply module will use the UCI to determine if the router is being used as a backup router, by parsing the firewall forwarding rules.

If it is already serving as a backup, a "not ok" is sent to the congested router (requester), which will have to choose another backup. Else, the backup will change its forwarding rules to accept traffic coming from the internal network interface and forward it to its gateway. An "ok" message is sent back to the congested router, which will also change its forwarding rules.

The load balancing finally begins and will continue until the router is no longer congested, or the backup is no longer suitable. In that case, the router will have to wait for the assign of a new backup.

4.3 CENTRALIZED LOAD BALANCING

As explained before, this work addresses two solutions for load balancing on trains using a distributed and a centralized algorithm. The later consists of a load balancing system in an SDN environment, and its implementation will be explained next. The algorithm's complete description can be found in chapter 3, section 3.5.

This solution provides many advantages, like the provision of centralized management and control, a more scalable network, and a dynamic environment where no network element is isolated. Also, the abstraction of the forwarding plane will provide an increased simplicity in the load balancing control.

In this centralized solution, an SDN controller, which also acts as a load balancer, is the network element that has information on all routers and decides on how the forwarding should be processed. Each router communicates with the controller VM, delivering statistics information and receiving control and management directives, embedded in Openflow messages.

The developed system uses some modules of the distributed solution since there are similar or even equal processes between both solutions. Some were improved to be more robust for a more reliable system. The router VMs' OS was maintained (OpenWrt).

The load balancing mechanism is divided into two versions. The first is version 1, or "forward all", where a congested router will forward all its passengers' traffic to a backup router, according to the controller's decision. The second is version 2, or "blacklist" version. Here, the controller machine has a file with blacklisted destination IP addresses, or in other words, IP addresses of low priority web services. These websites or services may put the network under stress if many users are accessing them. Upon a load balancing event, traffic with these destination IP addresses are dropped, and the remaining traffic is sent to the backup router. The system administrator defines the blacklist, and in a real-world implementation, a DNS server could be set up in the train to simplify the creation of the list so that only domain names were stored and they could be resolved. However, the solution developed only requires destination IP addresses to be stored in the file.

Both versions of the algorithm have the same foundation. They were both developed to ascertain their performance and reliability in a congested network and to compare their strengths and weaknesses.

It is crucial to notice that the latency in the communication between the network elements is close to nonexistent, as this is a virtualized environment. In a real-world implementation, the connection between the network elements must be guaranteed with a high-speed link. This way, statistics are updated in the controller in near real-time and the traffic being forwarded between the routers has the lowest latency.

Finally, a new network element was added to the system: the aggregator, whose purpose on the system and full operation will be explained thoroughly in section 4.3.2.

4.3.1 SETUP OF THE SOLUTION

The overview of the related technologies and the overview of the network architecture were already depicted in section 4.1, and chapter 3, section 3.3, respectively. Figure 4.4 depicts an in-depth diagram of the system, with the communication channels, messages exchanged in the system, and also the major system's components inside each node.

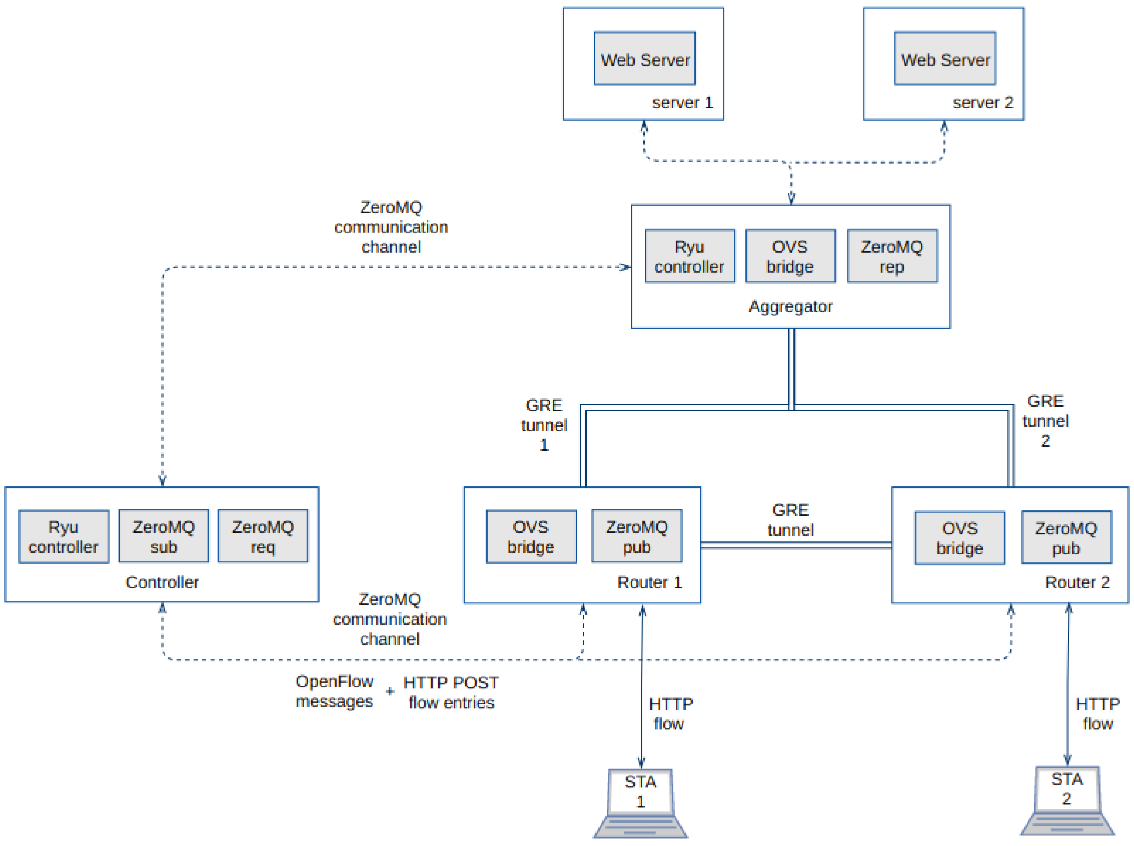

Figure 4.4: Setup of the solution and communication between the network elements

Like in the distributed solution, the communication of statistics occurs through a ZeroMQ message queue. However, in this centralized solution, the subscriber module is only located in the controller machine. So, each router is still a publisher, but no longer sends statistics information to other routers.

OVS is installed in all routers, and one isolated bridge interface is created in each switch. When STAs' traffic arrives at the router, it reaches the OVS bridge, and its forwarding is subject to the controller's decisions. What this means is that traffic may be forwarded between the bridge and the gateway or the backup router, depending on which flow entries are added to the flow table by the controller. Also, all bridges have an IP address on the same network.

The controller and the routers exchange Openflow messages through the internal network, and with the execution of HTTP requests using Ryu's REST APIs (`ryu.app.ofctl_rest` [7]) and usage of the OVSDB management protocol, the forwarding plane is controlled, and the switches queried. The control of the forwarding plane is done by adding or removing flow entries in the switches' flow tables. Flow entries are rules or instructions that tell the switch what to do with an incoming stream of packets.

[7]`https://ryu.readthedocs.io/en/latest/app/ofctl_rest.html`

Since the switch's bridge is isolated, or in other words, has no ports that correspond to physical interfaces of the router, the redirection of traffic must be done with the use of tunnel ports. Therefore, the forwarding between the physical interfaces can be managed by the controller, indirectly.

A GRE tunnel port is created in each switch for each router peer in the train, with a remote IP address of the peer interface connected to the internal network of the train. A GRE tunnel is also set up between each router and the aggregator.

The aggregator has OVS installed, with a single bridge, that receives all traffic coming from the routers. A Ryu controller is also installed, in order to manage the forwarding of that same traffic (in only one of the modules, refer to section 4.3.2). The STAs' traffic of each car travels through different tunnels (one tunnel per router), which simplifies the management of the traffic flows. So, for instance, when a packet from router A is received in the tunnel of router B, which happens due to load balancing, the aggregator will be notified about this change in the topology by the controller on the train (or will detect the change by itself) and update the switch's forwarding rules accordingly. The number of tunnels connecting to the aggregator is equal to the number of routers in the train. Although only one bridge was added to the switch of the aggregator, in a real-world implementation, one bridge should be created for every train the railway company wants to manage.

A bridge added to the switch of a router and the bridge of the aggregator's switch both have an IP address in the same network. Thus, the default gateway of each router is the aggregator's bridge IP address. This topology facilitates the sending and receiving of passengers' traffic between the two entities. However, since switching loops can occur, due to the communication between the routers, the ARP protocol is disabled in the interfaces that allow communication between the routers and the aggregator.

NAT no longer occurs in the gateway of each router but in the aggregator's gateway. This configuration allows for the traffic to be redirected back and forth between the routers without the TCP connections of the traffic flows being broken, since the traffic is aggregated in the aggregator's switch and only then sent to the external network, or the Internet.

In summary, in a normal operation of the network, IP traffic from the STAs is sent to a router, reaches the switch, which will check the OpenFlow flow table and forward the traffic according to its flow entries. The traffic is sent through a GRE tunnel to the aggregator, reaching its switch. The flow entries will command the forwarding of the traffic to the gateway, where NAT rules are applied so that it can finally be sent to the Internet. The response follows the same path as the request.

4.3.2 LOAD BALANCING SYSTEM PROCESSES

Similar to section 4.2, the development of the centralized solution will be explained next, along with the associated modules, processes, and operations that compose the whole system.

Statistics Collection

In this module, reused from the distributed system, the statistics are collected every 550 ms and the format in which they are stored is the same as explained in section 4.2. The origin of the statistics is also the same: the `/sys/class/net/<if>/statistics` directory, `fping` and the module counting the TCP retransmissions.

All the interfaces of the router are monitored. They are the gateway interface, the OVS bridge interface, the car network interface, and the internal network interface. They provide the controller with statistics and configuration information.

Reporting of Statistics

In this centralized solution, the statistics collected in each router are sent only to the controller. The ZeroMQ publisher and subscriber modules are similar to the ones of the distributed solution.

There is a continuous communication of messages from the routers (publisher) to the controller (subscriber), containing all interfaces' information, in a JSON objects list format. The messages publishing periodicity was kept (250 ms) because a high frequency of the communication of the statistics guarantees a regular and closer to real-time update on the controller.

Switch and Tunnel Information

Before starting the load balancer, the controller VM must start the execution of a module that collects information about switches' ports. This module uses Ryu's OVSDB library, that enables it to speak the OVSDB protocol [72]. This library initiates connections from the controller side, so the routers must be listening on a TCP port beforehand. In the module, operations are executed using commands with a syntax similar to `ovs-vsctl` [8] commands.

During an iteration of the program, the discovery module is executed once, and its output (internal network IP addresses) is read into the module so that the `VSCtl` class of the library can describe new OVS instances. Only then, the `VSCtlCommand` class is used to build commands that, once executed, will retrieve from each router: the name of the bridge of the switch, its MAC address, and information about existing tunnels. This information is stored in a file, along with the respective datapath ID. The file

[8] `http://www.openvswitch.org/support/dist-docs/ovs-vsctl.8.txt`

will be used by the load balancer to construct new flow entries that will modify the forwarding in the devices.

Load Balancing Process

The load balancing algorithm used in the system is described in chapter 3, section 3.5. The decisions made during the development of the solution will be explained in detail in the following sections.

The load balancer was based on the solution developed for the distributed approach. This component is now located in the controller VM and suffered several modifications to comply with the new SDN environment.

The logic behind the maximum values of the statistics used in the evaluation was kept from the distributed mechanism. Latency is collected from the gateway interface, and TCP retransmissions from the car network interface, which is the first interface the passengers' traffic goes through. Although the Ryu controller has tools to obtain network statistics of the bridges remotely, the statistics provided are not sufficient for the evaluation of the routers, and that is why they collect the information locally and send it to the controller.

The explanation of the logic behind the assignment of backups to congested routers can be found in chapter 3, section 3.5. The assignment is processed so that if a router has multiple backups, only one is chosen, and if two or more choose the same backup, that backup is assigned to only one of the routers. If a router is serving as a backup to another router, it cannot balance its own traffic. Also, the load balance to a backup is exclusive, or in other words, a backup cannot be shared.

In every iteration of the load balancer, after the backup candidates list is formed, each router is subject to a preprocessing that will determine in which state it is and how it should be processed. Figure 4.5 illustrates this preprocessing, described next.

- The router was not congested in the previous iteration, but now it is, and one backup was assigned to it. In this case, the router is inserted into a list in order to be processed by the load balancing algorithm, so that new flow entries are added to its switch.
- The router was congested in the previous iteration, is still congested, and the same backup was assigned to it. In this case, the router does not need further processing, and the load balancing flow entries remain in the switches' flow tables.
- The router was congested in the previous iteration, is still congested, but now a new backup is assigned to it. The router is inserted into two processing lists so that new flow entries are added and old ones removed by the load balancing mechanism.

- The router was congested in the previous iteration, but not anymore or there is no backup available. The router is inserted into a processing list for removal of old flow entries.

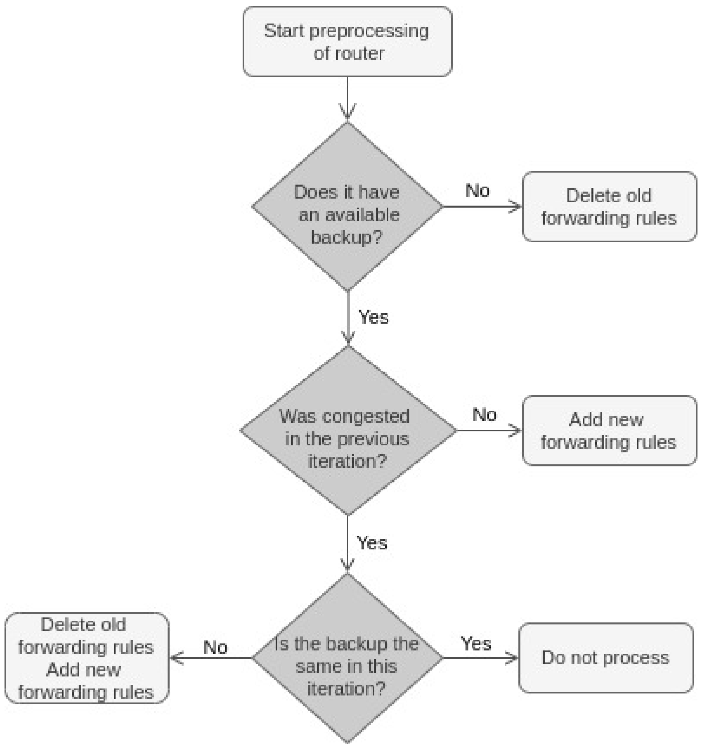
Start preprocessing of router
Does it have an available backup?
No
Delete old forwarding rules
Yes
Was congested in the previous iteration?
No
Add new forwarding rules
Yes
Delete old forwarding rules Add new forwarding rules
No
Is the backup the same in this iteration?
Yes
Do not process

Figure 4.5: Flowchart of the preprocessing of the routers

With the processing lists created (addition and removal), flow entries may finally be added to the congested and backup switches, to modify the forwarding of traffic, and outdated flow entries are removed. This step of the load balancing process uses Python's multiprocessing package [9], so that the load balance of each congested router is processed concurrently. One process is spawned per congested router, and the function invoked executes two steps. The first step consists of retrieving the `ofport` value (port number in the OVS database) of the two tunnel endpoint ports between the congested and the backup switch and of the two tunnel endpoint ports connecting each switch to the aggregator. The second adds flow entries to both switches so that requests from the congested router are forwarded to the backup and the backup delivers its responses, or otherwise, old forwarding rules are removed.

Adding Flow Entries

Ryu provides REST APIs that allow for the retrieval of the OVS statistics, control of the forwarding rules, among others.

[9]`https://docs.python.org/3.7/library/multiprocessing.html`

The load balancing mechanism will add flow entries to the congested and backup switches' flow tables. The flow entries must match traffic from and to the car network in order to apply the desired actions.

In the congested router's switch, a flow entry commands the forwarding of the aggregated traffic, received in the bridge, to the GRE tunnel connecting to the backup router. Another flow entry is also added for the delivery of the response coming from the tunnel. In the backup router's switch, two flow entries are added in order to forward traffic from the peer tunnel to the tunnel connecting to the aggregator, and vice-versa. The flow entries are added to the backup first and only then to the congested router. This will avoid a situation where the congested switch's flow entries were already added, but the backup is not ready for its traffic.

The structure of a generic flow entry inside an HTTP request is seen below.

```
data = {
          "dpid" : <datapath id>,
          "priority" : <priority>,
          "match" : {
                "in_port" : <incoming ofport>,
                "dl_type" : <Ethernet frame type>,
          },
          "actions": [ {
                "type": "OUTPUT",
                "port": <outgoing ofport>
          } ]
        }
```

The "priority" value must be greater than the priority of the flow entries permanently stored in the congested and backup switches' flow tables, which allow the normal forwarding of the traffic to the routers' gateway. The "in_port" is the port in which the traffic is received, the "dl_type" is the Ethernet protocol type, and the "port" inside "actions" is the port where the traffic should be sent. The input and output ports can be the LOCAL [70, p. 10] port or a tunnel port, depending on the flow entry.

The load balancing starts as soon as the requests reach the switches, and the flow tables are updated. The first part of the load balancing mechanism is now complete.

Traffic Aggregation

The traffic reaches the aggregator through a GRE tunnel. There is one tunnel endpoint per car in the OVS bridge of the aggregator. If a router is not balancing its traffic, the aggregator will simply forward it to the default gateway, apply NAT, and deliver the response through the appropriate tunnel. For this to be possible, flow entries are predefined in the switch's flow table.

However, since this is a load balancing environment, traffic from a car network may be forwarded to a router in another car, which means that the traffic will flow through its tunnel to the aggregator.

Since the switch in the aggregator knows beforehand through which tunnel the response of the traffic should go through, there must be a dynamic component that updates the flow entries in its flow table when load balancing is taking place, since the response to the congested router's traffic should go through the backup tunnel.

Update of the Flow Entries in the Aggregator

Two Python modules were developed for the update of the flow entries in the switch of the aggregator.

The first module uses Ryu's API for the creation of a controller application. When traffic from the congested router arrives through the backup's tunnel, an Openflow "packet_in" message sent from the switch is received in the controller (in this case, the message is exchanged within the same machine) with a captured unmatched packet. The message results from a miss in the match table of the switch. The control of the packet is, therefore, transferred to the Ryu controller.

The execution of a "packet in" handler is triggered in the module, which will update the flow table according to the new topology. So, the traffic from the congested network is now supposed to appear through the backup tunnel, and its response, coming from the Internet, must be sent through that same tunnel. This process occurs every time a packet from a car network appears in the tunnel of a router in another car. When the congested router returns to its normal operation, the process is once again executed to update the flow entries to their original configuration.

As long as the unmatched packet is delivered to the controller and the flow entries are updated immediately by the handler, the aggregator will be synchronized with the load balancing decisions taken by the controller on the train.

However, the frequency of the sending of requests may be lower than the frequency of the reception of responses in a TCP connection. What this means is, while the STAs are receiving multiple response packets, a request packet may not be sent, and thus a "packet in" is not received in the previous module. So, the load balancing flow entries may already be installed in the flow tables of the switches, but the aggregator is not updated immediately. For this reason, only the second module was used in the tests to the system.

The second module consists of the controller on the train notifying the aggregator of the new network topology, as soon as the load balancing flow entries are installed in the switches.

First, a message queue is setup with ZeroMQ, with the pattern request-reply. The controller on the train is the requester, and it sends a notification message to the aggregator, with two car networks' CIDR (of the congested and the backup routers). When the message arrives at the replier (aggregator), it will collect its switch's flow entries and the tunnel ports' information associated with the received car networks' CIDR and update the flow table according to the new topology. Every time there is a load balancing event, the controller sends the notification to the aggregator.

The two developed modules cannot be executed in parallel has they may delete or add flow entries in a way that the flow table is populated in an undefined manner.

Finally, in every two seconds of the execution of the load balancer, the routers' network state is evaluated and they are categorized as congested or not congested. The forwarding rules in the flow tables of the switches are updated accordingly.

5 Evaluation and Results

This chapter intends to validate the previously described implementation of the system. Multiple testing scenarios were used to evaluate the performance of both the distributed and centralized solutions. The tests carried out measure how the train network would behave if one or more routers started having a slow connection to the Internet or the connection was lost. Therefore, the tests evaluate the distributed and centralized solutions in an emulated environment that closely matches real network scenarios, in order to deliberate about its potential value in a real-world implementation. We first present the results of the distributed solution, and then, the results of the centralized solution.

All the scenarios proposed were tested with and without the load balancing algorithm. The scenarios are based on the download of a website on the clients' side, while the statistics about every download are continuously saved. The statistics provide the two measurements for the performance evaluation of the load balancing algorithm: number of downloads and load time. The load time, also known as real-time, or total wall clock time, is the human perception of the passage of the time of the download. It is, thus, a valuable statistic for the evaluation of the QoE of passengers. The results were also evaluated by the percentage of the passengers in a car that successfully downloaded websites in the duration of the tests. It is assumed that each car contains 70 passengers, accessing the Internet.

The network degradation occurs by applying a delay to the egress traffic in the gateway of each router, using Linux traffic control [1] and NetEm [2].

The setup for all testing scenarios has three main components: four router VMs, each one with one client VM, corresponding to the aggregated traffic in a car, and

[1] https://man7.org/linux/man-pages/man8/tc.8.html

[2] https://man7.org/linux/man-pages/man8/tc-netem.8.html

finally, one or two web servers that host a website that clients download, using `wget` [3]. Each one of the routers corresponds to a car of a train. All these network nodes and their connections are virtualized. Every VM runs a Linux distribution.

The wget command had a timeout of three seconds and a maximum number of tries equal to one. Each client continuously downloads a website of size 242 kB, without an interval between the end of a download and the beginning of the following. Each wget is made recursively, which means that every file that composes the website is downloaded.

The bandwidth in the gateway of all routers (upload and download) was fixed to 4 Mbps, which means that a client VM can download one website in less than one second. This value is, of course, only the sum of the time it takes to download each item that composes the website and does not include, for instance, the time to connect to the web server. The bandwidth was fixed in the testing scenarios so that it was not part of the degradation of the network. The delay added to the gateways already decreases the effective bandwidth.

The number of tests performed for each scenario was 15 with and 15 without load balancing, and each had a duration of 2 minutes. While the tests were running, the number of downloads made and load times of the website were continuously collected in each client.

5.1 Distributed solution's testing scenarios

The distributed solution was tested with two scenarios, A and B, with the setup of figure 5.1. The routers connect directly to the web server.

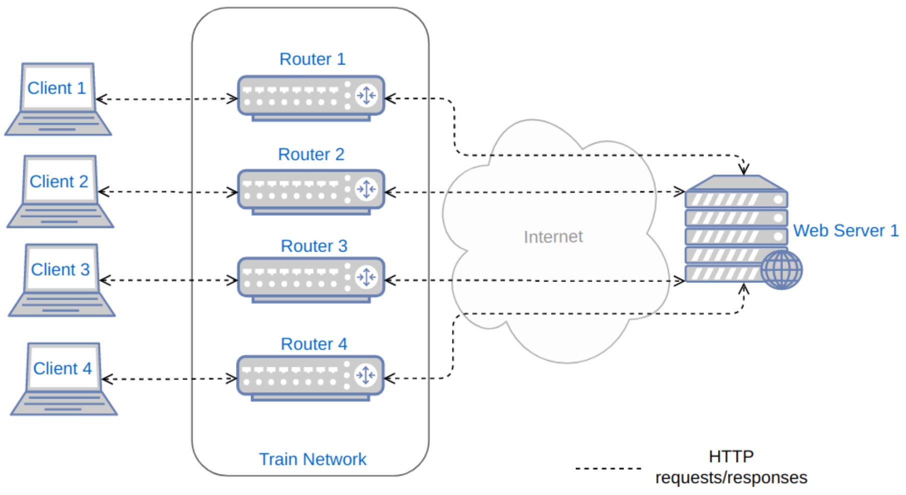

Figure 5.1: Distributed solution network setup

[3] https://www.gnu.org/software/wget/

5.1.1 SCENARIO A - CONSTANT DELAY

In scenario A, a fixed network delay of 10 ms was added to routers 1 and 2, and of 450 ms to routers 3 and 4. These values were chosen to test the impact of low and high latency in the load balancing environment and to see how the last two routers would compete with each other for the existent backups. Thus, the gateways have different link quality to the external network, during an extended period.

The value for the difference between the evaluations of the routers was half of the weight assigned to a statistic so that the load balancing is triggered frequently. The two statistics used in the evaluation have the same weight (50%), so the difference value chosen was 25.

Tables 5.1 and 5.2 show the results for the number of website downloads and respective load times with and without load balancing, in the performed the tests.

Client VM	without load balancing	with load balancing
1	186	102
2	186	99
3	21	94
4	21	93

Table 5.1: Average of website downloads per client - scenario A

Client VM	without load balancing	with load balancing
1	$0,640 \pm 0,005$ s	$1,202 \pm 1,432$ s
2	$0,600 \pm 0,004$ s	$1,192 \pm 1,441$ s
3	$5,999 \pm 0.005$ s	$1,180 \pm 0,859$ s
4	6.000 ± 0.000 s	$1,207 \pm 0,951$ s

Table 5.2: Average of website load times per client - scenario A

During the tests, routers 3 and 4 balanced their traffic most of the time to both routers 1 and 2. These two VMs had to share their bandwidth with their congested peers, which resulted in a significant decrease in their number of downloads and an increase in load times. There were also rare situations where the evaluation of VMs 1 and 2 increased, due to an increase in the latency values or the number of TCP retransmissions. In those moments, VMs 3 and 4 lost their right to use them as backups. Both VMs 3 and 4 had up to eight load balancing events in each test.

The number of downloads in the clients 3 and 4 increased by approximately four and a half times with the load balancing mechanism, and the load times decreased by close to five seconds. Therefore, there was a fairer distribution of the total bandwidth throughout the routers, and the QoE increased significantly for clients 3 and 4.

Finally, in this scenario, with the load balancing mechanism, 100% of the passengers in each car downloaded one website each, and 33% of the passengers in cars 3 and 4 downloaded two websites. Without load balancing, in the cars 3 and 4, only 30% of the passengers downloaded one website (refer to figure 5.2).

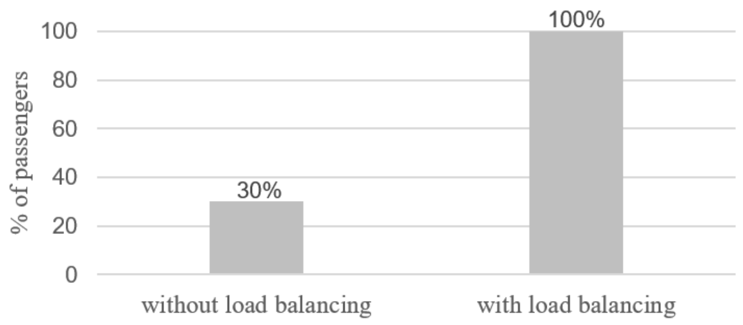

Figure 5.2: Percentage of passengers in cars 3 and 4 that downloaded one website

5.1.2 Scenario B - Constant Delay

This scenario has the same network and load balancer configuration as scenario A, however, router 4 has no Internet connection during the tests. The scenario tests how the load balancing performs when a gateway is not operational and has no connection to the cellular network.

The tables 5.3 and 5.4 show the results for the number of website downloads and their load times, respectively, with and without the load balancing mechanism.

Client VM	without load balancing	with load balancing
1	187	98
2	187	95
3	21	106
4	-	109

Table 5.3: Average of website downloads per client - scenario B

Client VM	**without load balancing**	**with load balancing**
1	$0,640 \pm 0,004$ s	$1,297 \pm 1,316$ s
2	$0,600 \pm 0,007$ s	$1,241 \pm 1,372$ s
3	$5,999 \pm 0,007$ s	$1,039 \pm 0,687$ s
4	-	$1,029 \pm 0,717$ s

Table 5.4: Average of website load times per client - scenario B

Client 3 had an increase in the number of downloads of five times the value without load balancing, similar to scenario A. Router 4, which had no Internet connection, was able to balance its traffic with other routers so that its client could communicate with the web server. Both VMs 3 and 4 had up to eight load balancing events in each test, like in scenario A.

Without load balancing, only 30% of the passengers inside car 3 downloaded one website. On the other hand, using the load balancing mechanism, all 70 passengers of cars 3 and 4 downloaded one website each, and on average 53% of the passengers in cars 3 and 4 downloaded two websites (refer to figure 5.3).

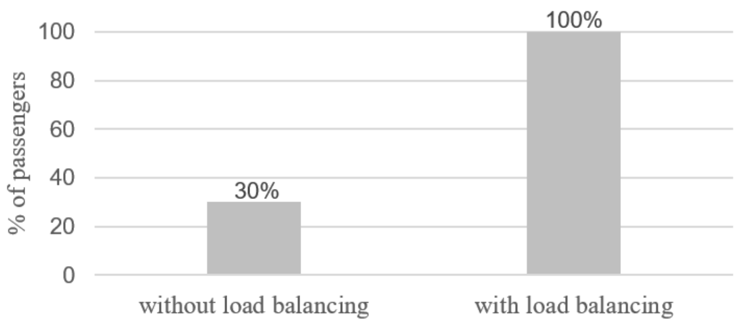

Figure 5.3: Percentage of passengers in cars 3 and 4 that downloaded one website

In this scenario, the QoE increased significantly for clients 3 and 4 and, more importantly, a router with no Internet access could use a backup's capacity to provide it to its passengers.

5.1.3 LIMITATIONS OF THE DISTRIBUTED SOLUTION

The distributed solution performs as expected, and the traffic balancing operates successfully. However, this implementation has limitations that had to be tackled in the centralized solution.

First of all, each router does its own NAT in the gateway interface, which means that TCP sessions will be broken in every load balancing event. If a router is labeled as congested for a small period (for instance, 2 seconds, which is the iteration time of the load balancer module), by the end of the reestablishment of the connection, the router may no longer be congested. So, the load balancing did not have a positive effect.

To overcome this issue in the distributed solution and to obtain improved testing results, a congested router was assigned the same backup router until it was no longer congested, or the backup was no longer suitable. Thus, if the backup increased its evaluation or the router was no longer labeled as congested, the pair was broken. This was the only moment where the TCP sessions were broken.

However, this technique turns the load balancing system into an unfair mechanism. If there is only one backup available on the train, the congested router which got hold of it first will load balance to it until one of the above conditions is met, even if there is another congested router in the train network. Thus, the backup is never shared.

Although multiple load balancing events may increase the total latency associated with the mechanism, this technique was removed in the centralized solution for a fairer load balancing system.

Since this is a distributed solution, there is no central entity with the information of the routers' network status. Thus, the system administrator does not have a centralized tool to monitor the train network. Management of the network would greatly benefit from a centralized solution.

Furthermore, the solution was developed using the router's OS capabilities. So, if the configuration files change their structure on future versions of the OS, the modules will have to suffer changes to their code. The modules are not easily portable to another OS, requiring significant modifications to the source code.

Finally, a solution that is not flexible, scalable, and adaptable, resorting only to the OS's networking tools, is prone to become obsolete in the future due to the rise in the usage of virtualized network functions. Nevertheless, because the performance tests showed positive results, the load balancing technique served as the basis for the algorithm used in the centralized solution.

5.2 CENTRALIZED SOLUTION'S TESTING SCENARIOS

In the tests of the centralized load balancing system, four router VMs have one client VM each. The routers connect to an onboard controller and to the aggregator (default gateway). The aggregator will then connect to one or two web server VMs. In this system, routers do not have NAT enabled in their gateway, to reduce the breaking of TCP sessions. The setup of the centralized solution's testing scenarios is depicted in figure 5.4, for scenarios A and D, and in the figure 5.6 for scenarios B and C.

This solution was submitted to four testing scenarios to evaluate its performance in different network conditions.

5.2.1 SCENARIO A - CONSTANT DELAY

This scenario is equal to scenario A of the distributed solution, where the gateways of each car have different link quality to the external network, during an extended period. In this scenario, each client continuously executes a wget process and the difference between the routers' evaluations must be greater than or equal to 25 for them to be considered a congested-backup pair. The two statistics used in the evaluation, latency and TCP retransmissions, have the same weight of 50%, equal to the distributed solution. The delay added to the routers 1 and 2 is 10 ms and to the routers 3 and 4, 450 ms. So, VMs 3 and 4 should have one backup each for most of the duration of the tests performed. Figure 5.4 depicts the setup used in the tests.

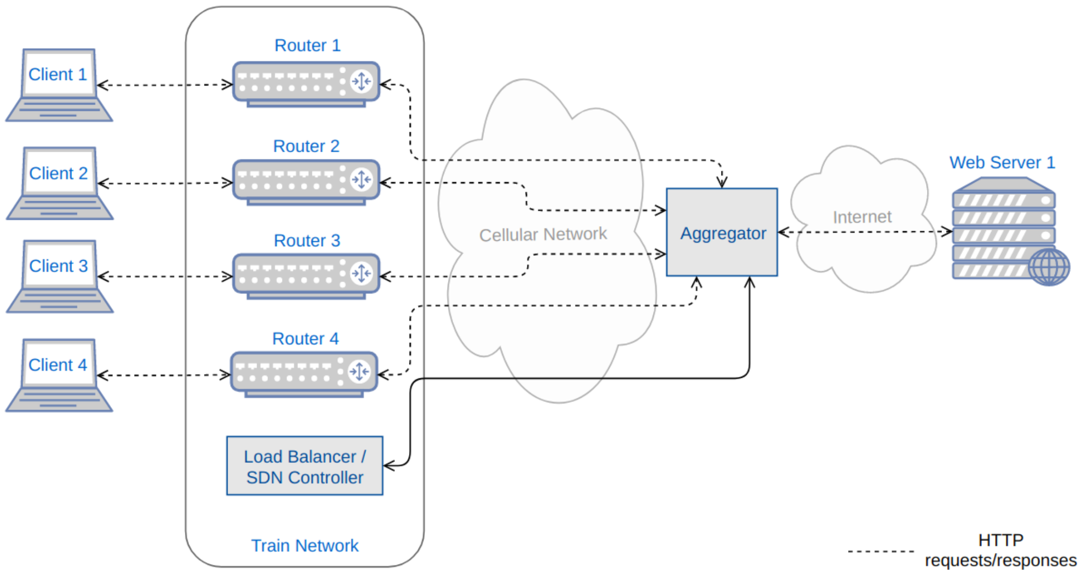

Figure 5.4: Network setup with one web server

Client VM	without load balancing	with load balancing
1	183	123
2	183	124
3	20	73
4	20	70

Table 5.5: Average of website downloads per client - scenario A

Client VM	without load balancing	with load balancing
1	$0,646 \pm 0,024$ s	$0,985 \pm 0,311$ s
2	$0,607 \pm 0,025$ s	$0,898 \pm 0,290$ s
3	$6,105 \pm 0,339$ s	$1,273 \pm 0,580$ s
4	$6,085 \pm 0,281$ s	$1,281 \pm 0,652$ s

Table 5.6: Average of website load times per client - scenario A

Referring to table 5.5, the number of downloads of the two congested VMs increased by approximately three and a half times with the load balancing mechanism. As expected, VMs 1 and 2 decreased their number of downloads since their bandwidth had to be shared with their congested peers.

The column of the results with load balancing of table 5.6 shows that although the download times of VMs 1 and 2 increased by 300 ms (approximately), the load times in the VMs 3 and 4 decreased by almost one fifth. Thus, passengers connected to the two congested car networks will have a significantly improved QoE if the network conditions are approximately constant during a specific period, and the backup routers have enough bandwidth for the new traffic.

In this scenario, with the load balancing mechanism, 100% of the passengers in cars 3 and 4 downloaded one website each. Without load balancing, this value decreases to 28% (refer to figure 5.5).

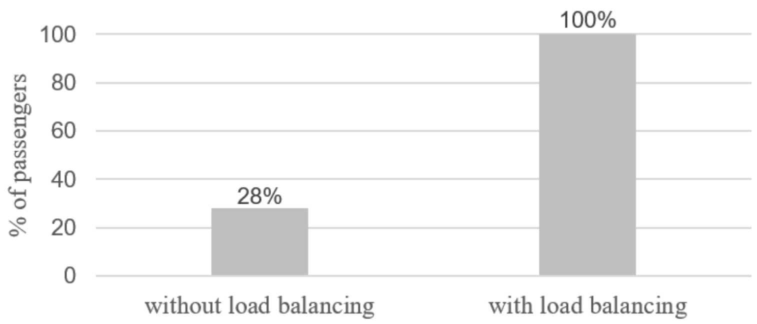

Figure 5.5: Percentage of passengers in cars 3 and 4 that downloaded one website

The load balancer had an average of 32 iterations with load balancing events in the tests. Although the balancing of the traffic of routers 3 and 4 should be favored, the event is triggered multiple times due to different factors:

- The congested VMs did not choose the same backup in the course of the tests, because the backups' latency values and the number of TCP retransmissions varied.
- The evaluations of routers 3 and 4 are similar for most of the test's duration since the same delay is added to them, just like with routers 1 and 2. However, their statistics are still variable. Due to the backup assignment from the highest to the lowest congested router, and the sorting of the candidates list in descending order, explained in chapter 3, section 3.5, the more their evaluation changes, the more load balancing events occur.
- If VMs 3 and 4 have an equal evaluation, the first to be assigned to a backup is the one whose statistics information was read first (top-down traversal of the statistics directory). This router will be assigned to the backup with the highest evaluation, which may vary between iterations.
- For the setup configuration in this scenario, the percentage of iterations of the load balancer in which VMs 3 and 4 were load balancing simultaneously is on average 67%, and not the expected 100%. The number of TCP retransmissions and the latency of one or more routers may increase spontaneously in a single iteration of the load balancer, and their evaluation rises to the point where it may lead to the problematic situations where VMs 1 and 2 load balance between them, when there is no need for that, or VMs 3 and 4 load balance between them, even though both their latencies are elevated. These occurrences led to an increase in the number of load balancing events.

The above situations and behaviors happened in all the centralized solutions' testing scenarios.

In the tests, a delay of up to 120 ms occurred in each topology modification in the aggregator. This delay was a factor in the increase of the download times in the clients in congested networks, due to packet loss caused by a temporary mismatch between the forwarding rules in the routers and in the aggregator (refer to section 5.2.5). In some moments of the tests, this packet loss led to read errors during wget downloads and the download had to be restarted.

After analyzing the obtained results, it is possible to conclude that, with the load balancer, the performance of the network can significantly improve, as well as the QoE of passengers. Thus, the main goal of the solution is attained, which is to increase the QoE of passengers in congested networks by increasing the number of websites downloaded and decrease the time it takes to download them.

5.2.2 Scenario B - Constant delay

This scenario, like the previous, evaluates the version 1 of the centralized algorithm of load balancing. Gateways 1 and 2 have a constant delay of 10 ms and gateways 3 and 4 a delay of 450 ms. However, in this scenario, clients 3 and 4 download simultaneously the same website from two web servers, while clients 1 and 2 download only one website (from server 1). Figure 5.6 depicts the setup used in this testing scenario.

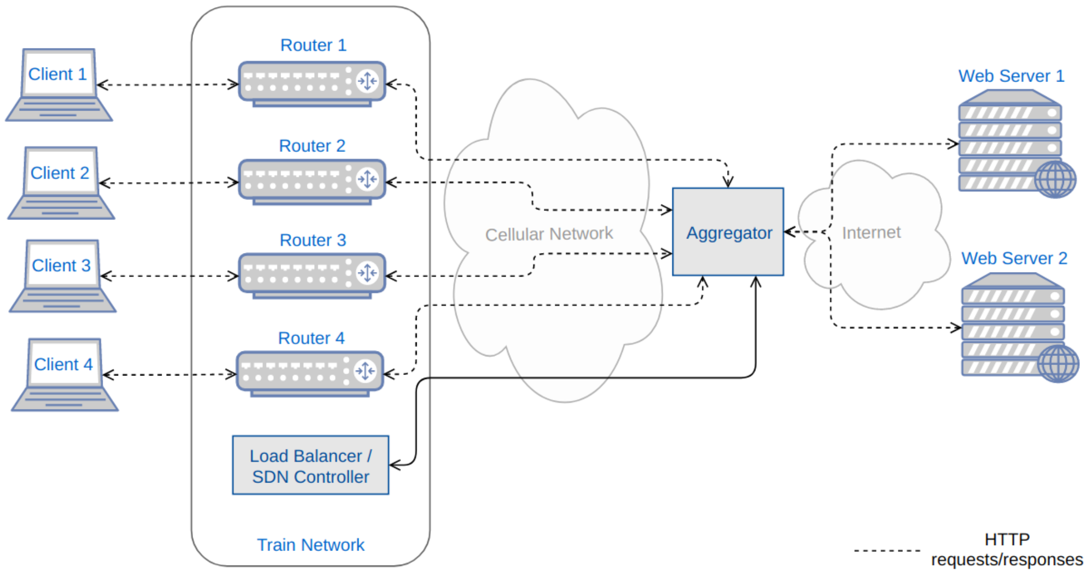

Figure 5.6: Network setup with two web servers

Client VM	without load balancing	with load balancing
1	181	101
2	181	103
3	39	102
4	39	100

Table 5.7: Average of website downloads per client - scenario B

Client VM	without load balancing	with load balancing
1	$0,651 \pm 0,031$ s	$1,212 \pm 0,487$ s
2	$0,609 \pm 0,029$ s	$1,117 \pm 0,485$ s
3	$6,299 \pm 0,463$ s	$1,932 \pm 0,867$ s
4	$6,304 \pm 0,453$ s	$1,937 \pm 0,901$ s

Table 5.8: Average of website load times per client - scenario B

The results of the tests, compiled in tables 5.7 and 5.8, show that the number of downloads in the two congested networks was more than the double with load balancing, and their load times decreased by approximately a third (for both servers).

Without load balancing, only 27% of the passengers in cars 3 and 4 downloaded both websites. However, using the load balancing mechanism, 72% of the passengers in cars 3 and 4 downloaded the two websites, from both servers (refer to figure 5.7). This would translate into a considerable increase in the satisfaction of the passengers with the Internet service.

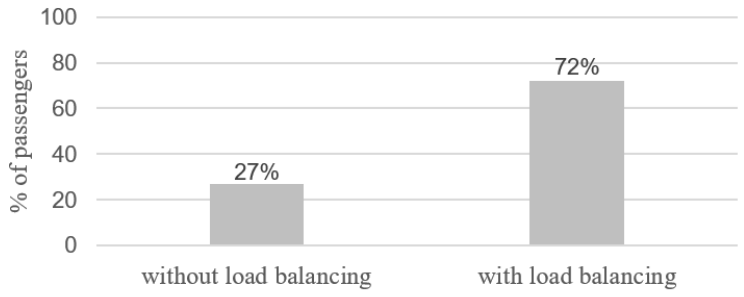

Figure 5.7: Percentage of passengers in cars 3 and 4 that downloaded the two websites

5.2.3 SCENARIO C - CONSTANT DELAY

Scenario C evaluates the version 2 of the centralized algorithm. With the same setup and delay scheme as scenario B (refer to figure 5.6), it evaluates the need for a load balancing algorithm with traffic prioritization. Clients 1 and 2 download a website hosted in web server 1 and clients 3 and 4 download two equal websites simultaneously from web servers 1 and 2.

The web server 2 is a blacklisted destination in this scenario, which means all IP communications from and to it are dropped by the congested switch.

Client VM	without load balancing	with load balancing
1	182	129
2	182	133
3	39	65
4	39	64

Table 5.9: Average of website downloads per client - scenario C

Client VM	without load balancing	with load balancing
1	$0,647 \pm 0,025$ s	$0,930 \pm 0,291$ s
2	$0,607 \pm 0,028$ s	$0,856 \pm 0,271$ s
3	$6,294 \pm 0,479$ s	$1,353 \pm 0,760$ s
4	$6,278 \pm 0,369$ s	$1,364 \pm 0,805$ s

Table 5.10: Average of website load times per client - scenario C

The results of scenario C, tables 5.9 and 5.10, show that, by giving priority to the traffic destined to server 1 and dropping traffic from and to server 2, the number of downloads of the high priority traffic will increase, comparing to the previous scenario B. Thus, the QoS was improved for the web service. In a real-world scenario, high priority destinations may be low-bandwidth consuming websites, while blacklisted destinations may be streaming services, for instance.

In this scenario, 92% of the passengers in cars 3 and 4 downloaded one website from the prioritized destination (refer to figure 5.8).

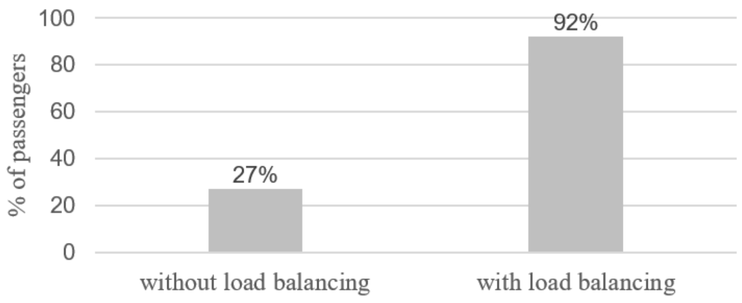

Figure 5.8: Percentage of passengers in cars 3 and 4 that downloaded the two websites, without load balancing, and the prioritized website, with load balancing

In conclusion, version 2 of the algorithm can provide an improved QoS for high priority web services. However, comparing the results on tables 5.7 and 5.9, if there is sufficient bandwidth in the backup router for all the congested router's traffic (to web servers 1 and 2) and its own, the prioritization of the traffic may be unnecessary. Even though the load times in the congested networks decreased in this scenario, the number of downloads from server 1 increased only by an average of 14, compared to scenario B.

Dropping the traffic of popular web services, like video streaming, should be a practice used only if the bandwidth is limited in the backup router. The train network should avoid forbidding passengers to access the web services of their choice.

5.2.4 SCENARIO D - SEQUENTIAL DELAY

In this scenario, a sequential network degradation was added to the routers, unlike the previous scenarios. This scenario arises from the assumption that, taking into account the single direction in which the train travels, a router in car N will have the same network conditions and connection quality to the cellular network as a router in car N-1 after a specific time interval.

Each client runs a wget command, and the difference between the evaluations must be greater than or equal to 15. This value, lower than the previous 25, was chosen to trigger the load balancing more frequently, considering the delay values of table 5.11.

During a high-speed train trip, we used an Android app in a smartphone to monitor the cellular network continuously. The monitoring results captured how the signal strength varies during the trip.

A small sample of the measurement of one of the collected statistics, the Reference Signal Received Power (RSRP) (in dBm), was extracted from the obtained results. Then, we mapped a range of values, spanning from the highest to the lowest RSRP value of the sample, to delay values from 18 ms to 522 ms. In the collected sample, the RSRP value associated with the highest signal strength is -77 dBm, and the value associated with the lowest signal strength (or total disconnection) is -105 dBm. There is an interval of 18 ms between each consecutive RSRP value.

During each test, the delays of table 5.11 were sequentially added to the gateways of the routers. Each value was fixed for 5 seconds, and the time-lag between the routers was also 5 seconds. Thus, router N has the same delay as router N-1 after 5 seconds.

The following graph 5.9 and table 5.11 show the variation of the delays added to the routers. The graph shows the delay applied to the first router during a test. Afterwards, the results of the testing scenario are compiled in tables 5.12 and 5.13.

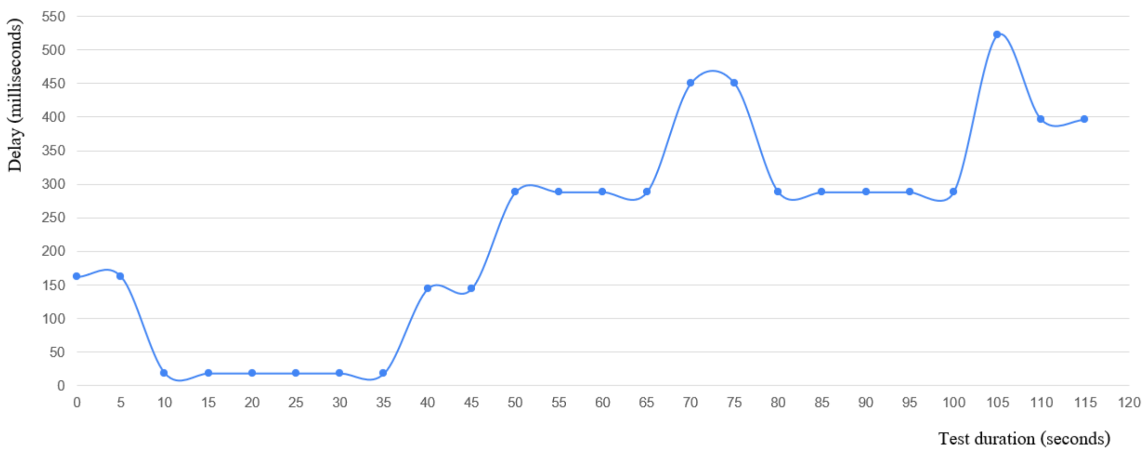

Figure 5.9: Delay added to the router in the first car during each test

Time	Router 1	Router 2	Router 3	Router 4
0	162	18	18	18
5	162	162	18	18
10	18	162	162	18
15	18	18	162	162
20	18	18	18	162
25	18	18	18	18
30	18	18	18	18
35	18	18	18	18
40	144	18	18	18
45	144	144	18	18
50	288	144	144	18
55	288	288	144	144
60	288	288	288	144
65	288	288	288	288
70	450	288	288	288
75	450	450	288	288
80	288	450	450	288
85	288	288	450	450
90	288	288	288	450
95	288	288	288	288
100	288	288	288	288
105	522	288	288	288
110	396	522	288	288
115	396	396	522	288

Table 5.11: Delays added to the routers (milliseconds) every five seconds

Client VM	without load balancing	with load balancing
1	62	54
2	68	62
3	73	66
4	79	65

Table 5.12: Average of website downloads per client - scenario D

Client VM	without load balancing	with load balancing
1	$1,971 \pm 1,758$ s	$1,837 \pm 1,795$ s
2	$1,680 \pm 1,664$ s	$1,541 \pm 1,607$ s
3	$1,576 \pm 1,576$ s	$1,542 \pm 1,536$ s
4	$1,463 \pm 1,476$ s	$1,539 \pm 1,527$ s

Table 5.13: Average of website load times per client - scenario D

This scenario with sequential degradation did not benefit from the use of the load balancing mechanism. The negative results are due to the fact that the network degradation is elevated in all machines during the tests, most specifically in the second half of table 5.11, and with a low evaluation difference, the load balancing is triggered between machines who have similar network degradation. For instance, a machine with a delay of 450 ms may load balance to a machine with a delay of 288 ms, although this backup also has an elevated latency. Furthermore, the fact that the delay is changed every five seconds may bring some issues, since an iteration of the load balancer lasts two seconds and the statistics information collected by the program may be outdated for more than one second after the application of the delay on the routers.

Adding the described issues to the latency that results from the execution of the multiple steps associated with the load balancer (refer to section 5.2.5), if router N has the same network conditions of router N-1 after a short time (which may be the case on a train), the network will not benefit from this type of load balancing.

In conclusion, the developed system only provides positive results when the network conditions are approximately constant for a longer period, and the difference between the evaluations is increased.

5.2.5 LIMITATIONS OF THE CENTRALIZED SOLUTION

Although the load balancing system provided positive results in scenarios A, B, and C, there are some issues associated with the solution that emerged when performing the multiple tests. Some were already described in the discussion of the results of scenario A, in subsection 5.2.1, and in the discussion of scenario D, in subsection 5.2.4. Below, two additional issues are described.

TCP retransmissions

The counting of the TCP retransmissions did not increase or decrease in the same proportion as the latency values. What this means is, specially on the congested networks, the TCP retransmissions count did not have a regular value with each statistics collection. Thus, this value did not “coincide” with the delay values applied. Furthermore, the routers with the lowest delay applied also retransmitted packets occasionally.

Since the TCP retransmissions have a weight of 50% in the evaluation of the routers, this statistic may elevate a router’s evaluation to the point where it chooses a backup with the same delay as itself. If the maximum latency stipulated in the connection between the routers and the aggregator increased, for instance, greater than 1 second, the frequency of the TCP retransmissions counted by the module would increase. However, the consequent rise in the RTT timeout means that the statistics are not collected as frequently, which may have consequences upon load balancing decisions.

In conclusion, although an increase in the number of TCP retransmissions may indicate congestion in a network, since it was not a regular value in the tests, the routers’ network status was incorrectly evaluated in some moments.

Overall, this statistic does not provide a correct evaluation of the state of the link to the aggregator and should not be used in the system. In future work, the system would benefit from the addition of other evaluation statistics that may improve the evaluation of the network state, like signal strength or the number of dropped packets, for instance.

Flow entries delay

The aggregator and the controller on the train should be synchronized so that modifications in the forwarding plane of the train are reflected immediately in the aggregator’s switch (in a load balancing event). However, if there is latency in the communication from the controller to the aggregator, there may be a time window where the aggregator sends packets destined to routers through the wrong tunnel (for instance, old backup router). In this case, some packets may be lost and retransmissions appear.

In the tests, although there was no delay applied to the egress traffic of the controller, there was a delay between the moment a notification from the load balancer arrived at the aggregator to the end of the topology modification, that could be up to 120 ms. This delay caused a temporary inconsistency between the forwarding rules installed in the routers and the aggregator. In a real-world implementation, adding to this latency, there is an increased latency in the communication between the remote node and the controller and routers, which will aggravate the issue.

One solution to this problem would be to place the aggregator’s functionalities inside the train. However, that would remove the possibility of remote control, management, and monitoring of the access routers and clients’ traffic. Another solution, the one we believe is the most adequate to the usage of this system, is to only load balance when the congested router is close to disconnection to the cellular network (or is disconnected), in order to avoid “unnecessary” load balancing events. Thus, the developed system would behave like a gateway redundancy mechanism.

Also, the HTTP requests executed during the addition and deletion of flow entries may take a different time to get a response and in a real-world implementation, the delay may be aggravated depending on the type of connection and technology used for the communication between the controller and the routers. If the latency is increased in the communication, the execution time of the load balancing process will increase.

6 Conclusions

The train network topology that is considered in this work does not allow an efficient usage of the existent resources as each network node is isolated in each car, and there is no communication of network state. The main goal of this dissertation was to develop two load balancing systems, one distributed and the other centralized, that would provide an environment where routers could communicate their network statistics to other nodes in a train and balance their traffic according to their network state.

The state of the art of this document covered different concepts and topics related to Internet access on trains, load balancing, SDN, gateway redundancy, among others. The described solutions for Internet access on trains found in the literature helped to conceptualize a solution for the later developed systems.

A distributed solution was developed for an initial approach to a load balancing mechanism. In this solution, routers communicate statistics information with each other and each router has the capacity to decide on load balancing events. Then, a more robust and flexible centralized solution was developed. This solution is based on an SDN environment where routers send information to an onboard controller, which then uses the collected information to decide on load balancing events and informs a remote node about changes in the forwarding plane. This node aggregates the passengers' traffic before sending it to the Internet.

All functional and nonfunctional requirements were met, except for nonfunctional requirement number four (refer to chapter 3, section 3.2.2). If a router is removed from the network after the scanning of the internal network to retrieve IP addresses of active routers, the module that collects OVS ports information (refer to chapter 4, section 4.3.2) hangs for more than 2 seconds (one iteration of the load balancer). This issue will compromise the load balancing mechanism as it depends on the information collected from that module. Thus, it should be solved in future work.

The two testing scenarios of the distributed solution provided positive results, with a uniform distribution of the total capacity of the train. In the centralized solution, the scenarios proposed which had a constant network degradation produced positive results, and it was proved that QoS and QoE could be significantly improved when using the load balancing mechanism. However, the results of the last testing scenario of this solution (refer to chapter 5, section 5.2.4) showed that if the latency in the communication to the outside changes frequently, within a small time window, and the difference between the evaluations is lower than half of the weight of a statistic, the load balancing mechanism is not beneficial to the train network. Considering the possibility that, on a real train, router N has the same network degradation as router N-1 after some time (depending on the speed of the train), the developed load balancing system can be beneficial to the train network if used as a gateway redundancy mechanism.

Finally, this work lays the foundation of a solution for load balancing in the Internet services of moving vehicles, especially trains. The results obtained prove that the solutions can provide an increased QoE to passengers since they allow a fairer sharing of the aggregated network capacity of the train. Indeed, a future implementation in trains would increase the satisfaction of the passengers with the Wi-Fi service, as they would have a more productive and entertaining trip.

6.1 FUTURE WORK

As described previously in this document, especially in chapter 5, there are some issues with the centralized solution that need to be tackled in the future to improve the reliability of the solution and allow a possible implementation in a real train network. Some guidelines for future work are mentioned below:

- **Add and remove routers**: it should be possible to add or remove access routers in the train network without compromising the load balancing mechanism.
- **Adding new statistics**: of the two statistics used for the evaluation of the routers, latency was the one that provided a reliable evaluation of the congestion status. New statistics should be added for the evaluation to be closer to the real network status. Two possible statistics are the number of dropped packets and the signal strength (RSRP or RSRQ).
- **Aggregation of traffic in a 5G network**: use the aggregator as a MEC entity, when the technology becomes more developed in 5G networks. In this environment, latency would decrease between the train and the remote node.
- **Changes in the backup allocation**: instead of just establishing a minimum difference value between the evaluations, the choice of a backup should be more

flexible. One idea would be to develop a machine learning algorithm that explores the traffic pattern of the passengers and uses past statistics of the routers' interfaces to infer not only the behavior of the passengers in the access to the Internet but also the moments on the trip and locations where that access was degraded. This information would be used to make improved load balancing decisions.

- **Potential of the remote node**: considering that the aggregator is located outside the train, like in the developed system, add more functionalities to it so that not only it aggregates traffic from the train but also remotely controls, manages, and monitors the onboard routers and clients' traffic.
- **Security**: add security mechanisms to the various elements of the system.

References

[1] E. Haleplidis et al., *Software-Defined Networking (SDN): Layers and Architecture Terminology*, RFC 7426 (Informational), Internet Research Task Force, Jan. 2015. [Online]. Available: https://www.ietf.org/rfc/rfc7426.txt.

[2] J. Moreno, J. M. Riera, L. Haro and C. Rodriguez, “A Survey on Future Railway Radio Communications Services: Challenges and Opportunities”, *IEEE Communications Magazine*, vol. 53, no. 10, pp. 62–68, Oct. 2015. DOI: 10.1109/MCOM.2015.7295465.

[3] A. Parichehreh, S. Savazzi, L. Goratti and U. Spagnolini, “Seamless LTE connectivity in High Speed Trains”, in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, 2014, pp. 2067–2072. DOI: 10.1109/WCNC.2014.6952608.

[4] P. Fraga-Lamas, T. Fernández-Caramés and L. Castedo, “Towards the Internet of Smart Trains: a Review on Industrial IoT-Connected Railways”, *Sensors*, vol. 17, Jun. 2017. DOI: 10.3390/s17061457.

[5] É. Masson, M. Berbineau and S. Lefebvre, “Broadband Internet Access On Board High Speed Trains, a Technological Survey”, May 2015, pp. 165–176. DOI: 10.1007/978-3-319-17765-6_15.

[6] D. T. Fokum and V. S. Frost, “A Survey on Methods for Broadband Internet Access on Trains”, *IEEE Communications Surveys & Tutorials*, vol. 12, no. 2, pp. 171–185, Apr. 2010. DOI: 10.1109/SURV.2010.021110.00060.

[7] T. Han and N. Ansari, “RADIATE: Radio Over Fiber as an Antenna Extender for High-Speed Train Communications”, *IEEE Wireless Communications*, vol. 22, no. 1, pp. 130–137, Mar. 2015. DOI: 10.1109/MWC.2015.7054728.

[8] R. Singh, M. Ahlawat, and D. Sharma, “A Review on Radio Over Fiber Communication System”, vol. 6, no. 4, pp. 2319–7471, Apr. 2017.

[9] B. Lannoo, D. Colle, M. Pickavet and P. Demeester, “Radio-Over-Fiber-Based Solution to Provide Broadband Internet Access to Train Passengers [Topics in Optical Communications]”, *IEEE Communications Magazine*, vol. 45, no. 2, pp. 56–62, Feb. 2007. DOI: 10.1109/MCOM.2007.313395.

[10] H. W. Chang, M. C. Tseng, S. Y. Chen, M. H. Cheng and S. K. Wen, “Field Trial Results for Integrated WiMAX and Radio-Over-Fiber Systems on High Speed Rail”, in *2011 IEEE 22nd International Symposium on Personal, Indoor and Mobile Radio Communications*, Sep. 2011, pp. 2111–2115. DOI: 10.1109/PIMRC.2011.6139887.

[11] S. Hassan and K. Saeed, "Li-fi technology: Data transmission through visible light", *IJECE*, vol. 11, Aug. 2017. DOI: 10.5281/zenodo.1132246.

[12] Y. Kaymak, R. Rojas-Cessa, J. Feng, N. Ansari and M. Zhou, "On Divergence-Angle Efficiency of a Laser Beam in Free-Space Optical Communications for High-Speed Trains", *IEEE Transactions on Vehicular Technology*, vol. 66, no. 9, pp. 7677–7687, Mar. 2017. DOI: 10.1109/TVT.2017.2686818.

[13] R. Paudel, Z. Ghassemlooy, H. Le-Minh and S. Rajbhandari, "Modelling of Free Space Optical Link for Ground-to-Train Communications Using a Gaussian Source", *IET Optoelectronics*, vol. 7, no. 1, pp. 1–8, Apr. 2013. DOI: 10.1049/iet-opt.2012.0047.

[14] K. Ishizu, M. Kuroda and H. Harada, "Bullet-Train Network Architecture for Broadband and Real-Time Access", in *2007 12th IEEE Symposium on Computers and Communications*, Jul. 2007, pp. 241–248. DOI: 10.1109/ISCC.2007.4381563.

[15] G. Bianchi, N. Blefari-Melazzi, E. Grazioni, S. Salsano and V. Sangregorio, "Internet Access on Fast Trains: 802.11-Based On-Board Wireless Distribution Network Alternatives", in *12th IST Mobile & Wireless Communications Summit*, Jun. 2003, pp. 15–18.

[16] L. Goratti, S. Savazzi, A. Parichehreh and U. Spagnolini, "Distributed Load Balancing for Future 5G Systems On-Board High-Speed Trains", in *Proceedings of the 2014 1st International Conference on 5G for Ubiquitous Connectivity, 5GU 2014*, Feb. 2015, pp. 140–145. DOI: 10.4108/icst.5gu.2014.258110.

[17] K. R. Kumar, P. Angolkar, D. Das and R. Ramalingam, "SWiFT: a Novel Architecture for Seamless Wireless Internet for Fast Trains", in *VTC Spring 2008 - IEEE Vehicular Technology Conference*, May 2008, pp. 3011–3015. DOI: 10.1109/VETECS.2008.322.

[18] L. Verstrepen et al., "Making a Well-Founded Choice of the Wireless Technology for Train-to-Wayside Data Services", in *2010 9th Conference of Telecommunication, Media and Internet*, Jun. 2010, pp. 1–7. DOI: 10.1109/CTTE.2010.5557701.

[19] M. Terada and F. Teraoka, "Providing a High-Speed Train With a Broadband and Fault Tolerant IPv4/6 NEMO Environment", in *2012 IEEE Globecom Workshops*, Dec. 2012, pp. 1052–1056. DOI: 10.1109/GLOCOMW.2012.6477723.

[20] O. B. Karimi, J. Liu and C. Wang, "Seamless Wireless Connectivity for Multimedia Services in High Speed Trains", *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 4, pp. 729–739, 2012. DOI: 10.1109/JSAC.2012.120507.

[21] J. Garcia, S. Alfredsson and A. Brunstrom, "Examining Cellular Access Systems on Trains: Measurements and Change Detection", in *2017 Network Traffic Measurement and Analysis Conference (TMA)*, Jun. 2017, pp. 1–6. DOI: 10.23919/TMA.2017.8002916.

[22] National Rail Enquiries, *WiFi Facilities*. [Online]. Available: https://www.nationalrail.co.uk/stations_destinations/44866.aspx, (Accessed on: Apr., 14, 2020).

[23] B. Ai et al., "Future Railway Services-Oriented Mobile Communications Network", *IEEE Communications Magazine*, vol. 53, no. 10, pp. 78–85, 2015. DOI: 10.1109/MCOM.2015.7295467.

[24] S. Li, L. D. Xu, and S. Zhao, "5G Internet of Things: a Survey", *Journal of Industrial Information Integration*, vol. 10, pp. 1–9, Jun. 2018. DOI: 10.1016/j.jii.2018.01.005.

[25] GSMA, *5G Spectrum - GSMA Public Policy Position*, Mar. 2020. [Online]. Available: https://www.gsma.com/spectrum/wp-content/uploads/2020/03/5G-Spectrum-Positions.pdf.

[26] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher and V. Young, "ETSI White Paper No. 11: Mobile Edge Computing - a Key Technology Towards 5G", ETSI, White Paper, Sep. 2015.

[27] A. Sarkar, S. Agarwal and A. Nath, "Li-Fi Technology: Data Transmission Through Visible Light", *IJARCSMS*, vol. 3, no. 6, pp. 1–10, Jun. 2015.

[28] X. Bao, G. Yu, J. Dai, and X. Zhu, "Li-Fi: Light Fidelity - a Survey", *Wireless Networks*, vol. 21, Jan. 2015. DOI: 10.1007/s11276-015-0889-0.

[29] B. Naudts et al., "Internet on Trains: a Multi-Criteria Analysis of On-Board Deployment Options for On-Train Cellular Connectivity", in *2014 16th International Telecommunications Network Strategy and Planning Symposium (Networks)*, 2014, pp. 1–7. DOI: 10.1109/NETWKS.2014.6959256.

[30] Netflix Help Center, *Internet Connection Speed Recommendations.* [Online]. Available: https://help.netflix.com/en/node/306, (Accessed on: Apr., 10, 2020).

[31] G. M. Su et al., "QoE in Video Streaming Over Wireless Networks: Perspectives and Research Challenges", *Wireless Networks*, vol. 22, no. 5, Aug. 2015. DOI: 10.1007/s11276-015-1028-7.

[32] 3GPP, *LTE.* [Online]. Available: https://www.3gpp.org/technologies/keywords-acronyms/98-lte, (Accessed on: Apr., 10, 2020).

[33] c2c, *How Does WiFi Work Onboard?* [Online]. Available: https://www.c2c-online.co.uk/help_centre/onboard/how-does-wifi-work-onboard/, (Accessed on: Apr., 17, 2020).

[34] Deutsche Bahn, *WiFi On Board Trains and at Stations.* [Online]. Available: https://www.bahn.com/en/view/trains/on-board-service/wifi.shtml, (Accessed on: Apr., 17, 2020).

[35] Eurostar, *Eurostar Terms and Conditions.* [Online]. Available: https://www.eurostar.com/rw-en/website-terms-and-conditions, (Accessed on: Apr., 17, 2020).

[36] Queensland Rail, *Wi-FiFAQ.* [Online]. Available: https://www.queenslandrail.com.au/Wi-Fi/Pages/Wi-FiFAQ.aspx/#limitations, (Accessed on: Apr., 17, 2020).

[37] Railway Technology, *WiFi Woes: the Difficulties of Improving Onboard Internet Services*, Oct. 2017. [Online]. Available: https://www.railway-technology.com/features/featurewifi-woes-the-difficulties-of-improving-onboard-internet-services-5943140/, (Accessed on: May, 25, 2020).

[38] H. Kim and N. Feamster, "Improving Network Management With Software Defined Networking", *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, Feb. 2013. DOI: 10.1109/MCOM.2013.6461195.

[39] D. Kreutz et al., "Software-Defined Networking: a Comprehensive Survey", vol. 103, no. 1, pp. 14–76, Jan. 2015. DOI: 10.1109/JPROC.2014.2371999.

[40] S. K. Tayyaba et al., "Software-Defined Networks (SDNs) and Internet of Things (IoTs): a Qualitative Prediction for 2020", *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 11, pp. 385–404, 2016. DOI: 10.14569/IJACSA.2016.071151.

[41] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks", *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617–1634, Feb. 2014. DOI: 10.1109/SURV.2014.012214.00180.

[42] A. Lara, A. Kolasani and B. Ramamurthy, "Network Innovation Using OpenFlow: a Survey", *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 493–512, Aug. 2013. DOI: 10.1109/SURV.2013.081313.00105.

[43] I. Irawati, S. Hadiyoso and Y. Hariyani, "Link Aggregation Control Protocol on Software Defined Network", *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 5, pp. 2706–2712, Oct. 2017. DOI: 10.11591/ijece.v7i5.pp2706-2712.

[44] R. Junior, M. Vieira and A. Loureiro, “Dynamic Link Aggregation in Software Defined Networking”, in *2018 IEEE Symposium on Computers and Communications (ISCC)*, Jun. 2018, pp. 615–620. DOI: 10.1109/ISCC.2018.8538685.

[45] P. Sköldström and B. C. Sanchez, “Virtual Aggregation Using SDN”, in *2013 Second European Workshop on Software Defined Networks*, Oct. 2013, pp. 56–61. DOI: 10.1109/EWSDN.2013.16.

[46] A. Mimidis, C. Caba and J. Soler, “Dynamic Aggregation of Traffic Flows in SDN: Applied to Backhaul Networks”, in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, Jun. 2016, pp. 136–140. DOI: 10.1109/NETSOFT.2016.7502459.

[47] T. Kosugiyama, K. Tanabe, H. Nakayama, T. Hayashi and K. Yamaoka, “A Flow Aggregation Method Based on End-to-End Delay in SDN”, in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6. DOI: 10.1109/ICC.2017.7996341.

[48] J. Pavlik, A. Komarek, V. Sobeslav and J. Horalek, “Gateway Redundancy Protocols”, in *2014 IEEE 15th International Symposium on Computational Intelligence and Informatics (CINTI)*, Nov. 2014, pp. 459–464. DOI: 10.1109/CINTI.2014.7028719.

[49] T. Li, B. Cole, P. Morton, and D. Li, *Cisco Hot Standby Router Protocol (HSRP)*, RFC 2281 (Informational), Internet Engineering Task Force, Mar. 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2281.txt.

[50] R. Hinden, *Virtual Router Redundancy Protocol (VRRP)*, RFC 3768 (Draft Standard), Obsoleted by RFC 5798, Internet Engineering Task Force, Apr. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3768.txt.

[51] Cisco, *GLBP - Gateway Load Balancing Protocol.* [Online]. Available: https://www.cisco.com/en/US/docs/ios/12_2t/12_2t15/feature/guide/ft_glbp.html, (Accessed on: June, 19, 2020).

[52] M. Randles, D. Lamb and A. Taleb-Bendiab, “A Comparative Study Into Distributed Load Balancing Algorithms for Cloud Computing”, in *2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops*, Apr. 2010, pp. 551–556. DOI: 10.1109/WAINA.2010.85.

[53] N. Jain and I. Chana, “Cloud Load Balancing Techniques: a Step Towards Green Computing”, *International Journal of Computer Science Issues*, vol. 9, no. 1, pp. 238–246, Jan. 2012.

[54] H. K. Mehta, P. Kanungo and M. Chandwani, “Decentralized Content Aware Load Balancing Algorithm for Distributed Computing Environments”, in *Proceedings of the ICWET ’11 International Conference and Workshop on Emerging Trends in Technology*, Jan. 2011, pp. 370–375. DOI: 10.1145/1980022.1980102.

[55] Y. Bejerano and S. J. Han, “Cell Breathing Techniques for Load Balancing in Wireless LANs”, *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 735–749, Feb. 2009. DOI: 10.1109/TMC.2009.50.

[56] A. Bhadani and S. Chaudhary, “Performance Evaluation of Web Servers Using Central Load Balancing Policy Over Virtual Machines on Cloud”, in *COMPUTE ’10: Proceedings of the Third Annual ACM Bangalore Conference*, Jan. 2010, pp. 1–4. DOI: 10.1145/1754288.1754304.

[57] N. Mishra and N. Mishra, “Load Balancing Techniques: Need, Objectives and Major Challenges in Cloud Computing - a Systematic Review”, *International Journal of Computer Applications*, vol. 131, pp. 11–19, Dec. 2015. DOI: 10.5120/ijca2015907523.

[58] K. A. Nuaimi, N. Mohamed, M. A. Nuaimi and J. Al-Jaroodi, “A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms”, in *2012 Second Symposium on Network Cloud Computing and Applications*, Dec. 2012, pp. 137–142. DOI: 10.1109/NCCA.2012.29.

[59] V. Sreenivas, M. Prathap and M. Kemal, “Load Balancing Techniques: Major Challenge in Cloud Computing - a Systematic Review”, in *2014 International Conference on Electronics and Communication Systems (ICECS)*, Feb. 2014, pp. 1–6. DOI: `10.1109/ECS.2014.6892523`.

[60] M. Rahman, S. Iqbal and J. Gao, “Load Balancer as a Service in Cloud Computing”, in *2014 IEEE 8th International Symposium on Service Oriented System Engineering*, Apr. 2014, pp. 204–211. DOI: `10.1109/SOSE.2014.31`.

[61] S. Sathyanarayana and M. Moh, “Joint Route-Server Load Balancing in Software Defined Networks Using Ant Colony Optimization”, in *2016 International Conference on High Performance Computing & Simulation (HPCS)*, Jul. 2016, pp. 156–163. DOI: `10.1109/HPCSim.2016.7568330`.

[62] S. Kaur, K. Kumar, J. Singh and N. S. Ghumman, “Round-Robin Based Load Balancing in Software Defined Networking”, in *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, Mar. 2015, pp. 2136–2139.

[63] U. Zakia and H. B. Yedder, “Dynamic Load Balancing in SDN-Based Data Center Networks”, in *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Oct. 2017, pp. 242–247. DOI: `10.1109/IEMCON.2017.8117206`.

[64] A. Ikram, S. Arif, N. Ayub and W. Arif, “Load Balancing in Software Defined Networking (SDN)”, *MAGNT Research Report*, vol. 5(1), pp. 298–305, 2018. DOI: `1444-8939.2018/5-1/MRR.33`.

[65] S. Attarha, K. H. Hosseiny, G. Mirjalily and K. Mizanian, “A Load Balanced Congestion Aware Routing Mechanism for Software Defined Networks”, in *2017 Iranian Conference on Electrical Engineering (ICEE)*, May 2017, pp. 2206–2210. DOI: `10.1109/IranianCEE.2017.7985428`.

[66] S. Kaur and J. Singh, “Implementation of Server Load Balancing in Software Defined Networking”, in *Proceedings of Third International Conference INDIA 2016*, vol. 2, Feb. 2016, pp. 147–157. DOI: `10.1007/978-81-322-2752-6_14`.

[67] A. A. Neghabi, N. J. Navimipour, M. Hosseinzadeh and A. Rezaee, “Load Balancing Mechanisms in the Software Defined Networks: a Systematic and Comprehensive Review of the Literature”, *IEEE Access*, vol. 6, pp. 14 159–14 178, Mar. 2018. DOI: `10.1109/ACCESS.2018.2805842`.

[68] P. Kaur, J. K. Chahal and A. Bhandari, “Load Balancing in Software Defined Networking: a Review”, *Asian Journal of Computer Science and Technology*, vol. 7, no. 2, pp. 1–5, Apr. 2019.

[69] Open vSwitch, *What Is Open vSwitch?* [Online]. Available: `http://docs.openvswitch.org/en/latest/intro/what-is-ovs/`, (Accessed on: May, 20, 2020).

[70] The Open Network Foundation, “OpenFlow Switch Specification - Version 1.3.0 (Wire Protocol 0x04)”, ONF TS-006, Jun. 2012. [Online]. Available: `https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf`.

[71] O. Galinina, S. Andreev, S. Balandin and Y. Koucheryavy, *Internet of Things, Smart Spaces, and Next Generation Networks and Systems: 17th International Conference, NEW2AN 2017, and 10th Conference, ruSMART 2017*. St. Petersburg, Russia: Springer, Aug. 2017, ISBN: 978-3-319-67379-0.

[72] B. Pfaff and B. Davie, *The Open vSwitch Database Management Protocol*, RFC 7047 (Informational), Internet Engineering Task Force, Dec. 2013. [Online]. Available: `http://www.ietf.org/rfc/rfc7047.txt`.

[73] OpenWrt, *The UCI System*, Sep. 2019. [Online]. Available: `https://openwrt.org/docs/guide-user/base-system/uci`, (Accessed on: Apr., 10, 2020).

[74] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina, *Generic Routing Encapsulation (GRE)*, RFC 2784 (Proposed Standard), Updated by RFC 2890, Internet Engineering Task Force, Mar. 2000. [Online]. Available: `http://www.ietf.org/rfc/rfc2784.txt`.

[75] The Linux Kernel Archives, *sysfs-class-net-statistics*. [Online]. Available: `https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-class-net-statistics`, (Accessed on: Apr., 10, 2020).

[76] ZeroMQ, *ØMQ - The Guide*. [Online]. Available: `http://zguide.zeromq.org/page:all`, (Accessed on: Apr., 10, 2020).