

# Optimizing Telematics Network Performance through Resource Virtualization in a Disruptive Environment: The Case of the IP/MPLS Core Network

Patrick Dany Bavoua Kenfack<sup>1</sup>, Alphonse Binele Abana<sup>1</sup>, Emmanuel Tonye<sup>1</sup>,  
Paul Salomon Ngohe Ekam<sup>1</sup> & Gilles Herve J. Ngotty Mbang<sup>1</sup>

<sup>1</sup> Department of Electrical and Telecommunications Engineering, National Advanced School of Engineering of Yaounde, University of Yaounde I, Yaounde, Cameroon

Correspondence: Patrick Dany Bavoua Kenfack, National Advanced School of Engineering of Yaounde, University of Yaounde I, Yaounde, Cameroon. Tel: +237-69-966-7134. E-mail: danybavoua@gmail.com

Received: May 31, 2023 Accepted: June 30, 2023 Online Published: July 10, 2023

doi:10.5539/nct.v8n2p1

URL: <https://doi.org/10.5539/nct.v8n2p1>

## Abstract

We offer a security solution to considerably reduce latency in an IP network by virtualizing the IP/MPLS core network. It consists of adapting a virtualization method to a complex IP network, presenting the simulation of the implementation of this virtualization and the modifications to be made to certain aspects of the code of the solution. These modifications would take into account the key performance indicators of the network in order to guarantee its security and the transmission through very wide bands of data. To do this, we use Software Defined Network (SDN) technology. It allows us to have an emergent, scalable, dynamic, secure, laudable and adaptable network architecture, making it suitable for today's high bandwidth applications and IT services. This architecture decouples network control and digital data transfer functions, making network control directly programmable and the underlying infrastructure abstracted to network applications and services. After describing the soft failover to a virtualized network, we present the new architecture that describes the separation of the control and data planes of the core of the IP/MPLS network of the Autonomous Port of Kribi (PAK) in Cameroon, as part of our research work. We will then present the aspects in which modifying the code would contribute to improving one of the key qualities of service, namely latency in the heart of the network. We go from latencies above 100 ms to latencies below 1 ms; finally we recommend the approach for a continuous modification of the code with a view to optimizing the performance of the network in a continuous process for the reduction of its latency.

**Keywords:** Virtualization, Software Defined Network, IP/MPLS, Core network, Latency

## 1. Introduction

Seaports, defined as strategic nodes that facilitate the transit of various goods, play a major role in international trade. They ensure the transition from a mode of land transport to maritime transport and vice versa and establish exchanges of information between geographically distant points (de Barcelona, 2012).

The development and modernization of seaports have become one of the primary concerns of port authorities and political leaders in most countries of the world, because the economic development of landlocked and coastal regions depends mainly on port services. The 4th generation ports go even further and are characterized by telematic networks (communication networks based on new technologies) that connect different port areas and allow collaboration with other ports, with the objectives of internationalization and diversification. of their activities (de Barcelona, 2012).

Thus, the port has become a vast communication platform between the port authority and the actors of the port community. Faced with these global changes, the Cameroonian public authorities are obliged to upgrade port companies. In Cameroon, these seaports are administered by entities designated "Autonomous Ports".

In Cameroon, these seaports are administered by entities designated "Autonomous Ports". Port performance is one of the essential elements to guarantee uninterrupted global trade flows. One element making it possible to guarantee port performance would be the establishment of a vast, reliable, secure telematics network with high transmission capacities. This service is carried out through an IP/MPLS network cloud, managed by the structure in charge of the Autonomous Port's information systems. This therefore requires the guarantee of a good quality

of service (QoS) for the routing of data from one end of the cloud of their network to the other. However, the current so-called traditional networks no longer meet the constraints imposed by new uses and innovative applications driven by the rise of Cloud Computing, the development of M2M (Machine to Machine) and more generally by the Internet of Things (IoT) with the number of connections between objects doubling between 2019 and 2025 (gSM Association, 2020). The need for virtualization is therefore felt to solve these problems of security and latency in these telematic networks.

Technologies such as the Software Defined Network (SDN) and standard interfaces such as OpenFlow are now solutions that facilitate programmability (ability to be programmed) and customization of network functions. The Network Function Virtualization (NFV) for its part aims to provide the execution of network functions on environments of different suppliers, while guaranteeing good portability and interoperability. The integration of these two technologies seems imminent in the establishment of virtual, centralized, intelligent, agile, high-performance networks and network services of the future.

To achieve the operation of a virtualized network, it is a question of carrying out a progressive deployment of the virtualization solution of the network, without however radically cutting the operation of the previous so-called traditional network, to allow the IT teams to have a practical vision of the stages of the virtualization process and to appropriate new concepts. Also, it will be a question of understanding the functions of an IP/MPLS network core, of virtualizing them in order to meet the ever-increasing need of users in terms of ever-lower latency and other QoS criteria.

Issues related to the acquisition and maintenance costs of hardware and software are very difficult for companies to manage. The constant evolution of computer hardware and telecommunications leads IT departments to create increasingly heterogeneous infrastructures.

This proliferation of routers, switches, servers and physical infrastructure of a core network has become a source of financial problems and above all a problem in terms of the effective management of the entire telematics infrastructure. The core of the company's network will have to serve internally (local staff network) and externally (lessees and concessionaires) in the port area, to provide Internet and other services. This system, which presents itself as neuralgic, requires a high level of security. A significant flow of confidential data for the country as well as for the companies that make up all the players in the port area will pass through this network.

Likewise, the fact of dedicating a server and/or a router each time for an application leads to underutilization of system resources. We are quickly confronted with problems: of traffic management; bandwidth and latency management; setting up general configurations in several pieces of equipment simultaneously; the cost of purchasing equipment given the ever-increasing scalability of networks; the cost of maintaining several pieces of equipment subject to breakdowns and malfunctions of all kinds; securing this complex environment.

In view of the above, the need to migrate to virtualized networks is undoubtedly felt, our study will answer the following major questions:

- How to adapt virtualization methods in a traditional network?
- How to improve the calculation of traffic within the core IP/MPLS network so that networks meet the current constraints imposed by the increase in network load, cloud computing and the Internet of Things?
- What modification should be made in the source code of the virtualization solution in order to have a more efficient IP core network while maintaining a very good level of traffic security?

## 2. Network Virtualization

### 2.1 Why Virtualize Networks ?

There are many good reasons why a business or organization might invest in virtualization. The latter can increase IT agility, scalability and flexibility while creating significant cost savings. Greater workload mobility, increased performance and availability of resources, automated operations. In our context, some key benefits are illustrated below:

- Resource optimization

Today's enterprise-level computing resources tend to be so sophisticated that they often have excess capacity. By virtualizing hardware and allocating parts of it based on actual user and application needs, available computing power, storage space, and network bandwidth can be used much more efficiently.

- Expense cost optimization

It is common to dedicate individual computers to a single application. If multiple applications only use a small

amount of processing power, the administrator can consolidate multiple computers into a single server running multiple virtual environments. Virtualization requires fewer servers and extends the life of existing hardware, which clearly reduces maintenance costs.

➤ Optimization of service availability

Agility is being able to respond to changing requirements as quickly and flexibly as possible. Virtualization offers new opportunities for data center administration, allowing users to take advantage of:

- The guaranteed availability of servers and applications; rapid disaster recovery in the event of large-scale outages.
- Instant deployment of new VMs or even aggregate pools of VMs via template images.
- Elasticity, i.e. provisioning resources when and where needed instead of keeping the entire data center in a permanent state.
- Configuration of running IT environments with no impact on users.

➤ Automatic fault tolerance

Virtualization of servers and network equipment functions makes it possible to implement redundancy without purchasing additional hardware. Redundancy, in the sense of running the same application on multiple servers, is a security measure: if for any reason one server fails, another server running the same application takes over, minimizing service interruption. This type of redundancy works in two ways when applied to virtual machines:

If a virtual system fails, another virtual system takes over.

By running redundant virtual machines on separate physical hardware, one can also provide better protection against physical hardware failures.

➤ Flexible load migration

Migration refers to moving a server environment from one location to another. With most virtualization solutions, it is possible to move a virtual machine from one physical machine in the environment to another. With physical servers, this was originally only possible if both physical machines were running on the same hardware, operating system, and processor. In the virtual world, a server can be migrated between physical hosts with entirely different hardware configurations. Migration is typically used to improve reliability and availability.

➤ Protection of investments in existing systems

The physical part of the server will eventually become obsolete and moving from one system to another can be difficult. In order to continue offering the services provided by these legacy systems, one can run it as a virtual machine on new modern hardware, while the legacy system itself still behaves as if it were running on the same legacy hardware. From an application point of view, nothing will have changed. In fact, its performance may well benefit from the new underlying hardware. This gives the organization time to move on to new processes without worrying about hardware issues, especially in situations where the manufacturer of the legacy hardware no longer exists or cannot fix the faulty equipment.

Additional benefits of virtualization include:

Reduction of CapEx, OpEx and time to market; Reduced or eliminated system downtime; Increased IT productivity, efficiency, agility and responsiveness; Faster delivery of applications and resources; Better business continuity and disaster recovery; Simplified data center management; a true SDDC.

## 2.2 Types of virtualizations

- Network virtualization : a technology that allows the creation of logically isolated network partitions on shared physical networks so that a heterogeneous collection of multiple virtual networks can coexist simultaneously on the shared networks.
- Virtualization of servers or Data Center: a technique that allows multiple operating systems to run on a single server as highly efficient virtual machines.
- Virtualization of computers: a method of deploying simulated desktop environments on hundreds of physical machines at once, where, unlike traditional desktop environments, mass configurations, updates, and security checks can be performed at high scale.
- Virtualization of data and applications: Data virtualization refers to any process, involving a data management approach that allows aggregation data from different information sources to develop a single,

logical, virtual view of information.

- Virtualization of operating systems: Operating system (OS) virtualization is a server virtualization technology that involves customizing a standard operating system so that it can run different applications run by multiple users on a single computer at a time.

### 2.3 Network virtualization technologies

#### 2.3.1. Software Defined Networking

SDN technology makes it possible to make the operation of network devices programmable and to ensure their control, via a central element called a controller. SDN is now the subject of a standard defined by the Open Network Foundation (ONF). SDN thinking grew out of a significant problem faced by the networking industry. It was becoming increasingly difficult to build and administer large-scale IP/Ethernet networks.

In logical topology, SDN aims to separate the network control plane from the data plane. Thus, in a traditional network, when a packet arrives at the level of a switch (switch) of the network, the rules integrated in the firmware of this one govern the way of forwarding this packet. Each packet is thus sent to the same destination.

In an SDN scenario, rules governing packet management come to the switch from a controller, an application that runs on a server. The switch then queries the controller for the packet handling rules.

Thus, thanks to the SDN, the network administrator can, at will, modify the rules of a network device (switch, router, etc.) by enacting priorities, or even by blocking certain specific types of packets. The level of control is therefore very granular and can also be programmed. SDN allows the administrator to use less expensive equipment and increase control over network traffic flows. In the SDN architecture we have 03 main layers: application, control and infrastructure as shown in the following figure 1:

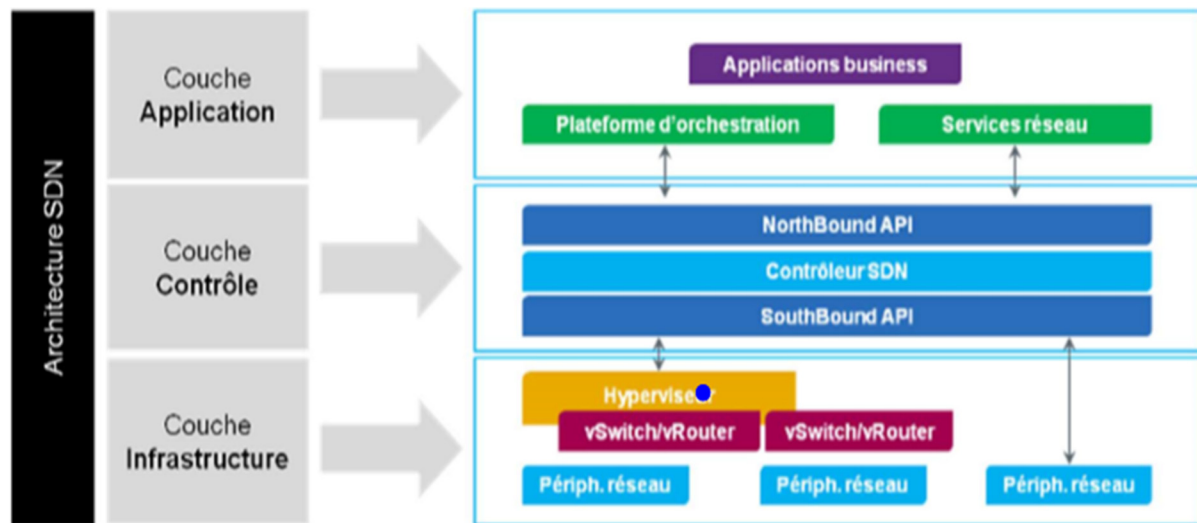


Figure 1. Architecture of SDN technology (Oloumane, 2019)

- Advantages of SDN technologies

We can identify as the main characteristics and advantages of SDN technology the following points:

SDN solutions make it possible to add new components or configure the existing one through programmable interfaces or the use of scripts;

- Administrators can now dynamically and automatically adjust network-wide traffic flows to meet fluctuating needs;
- Network intelligence is centralized through SDN controllers implemented in the control software. This control software offers a global and coherent view of the telematics network;
- The protocols and interfaces used are standard and not tied to a supplier;
- Applications that need to access geographically dispersed data and services on the public and private

cloud require flexible traffic management;

- SDN technology allows this flexibility by offering the bandwidth to evolve according to traffic variations;
- .By applying scheduling and automation to network control, network operators can dynamically define and manage their networks.

➤ Architecture SDN

SDN technology moves control of network resources from network equipment to a dedicated network element, called an SDN controller, which provides a means to schedule, orchestrate, control, and manage network functions through software. The SDN architecture is composed of the following three layers:

- The Application Layer;
- The Control Layer, which has the control plane;
- The Network Infrastructure Layer, which contains the data and forwarding plane (SDN Resources).

SDN has the following key aspects (Sanner, 2019):

- The separation of the control plane (SDN controller) and the data transfer plane (network resources). The first fundamental characteristic of SDN is the separation of the transmission and control planes. The forwarding functionality, including the logic and tables for choosing how to handle incoming packets, based on characteristics such as MAC address, IP address, and VLAN ID, resides in the forwarding plane. The basic actions performed by the forwarding plane can be described by how incoming packets are handled. It can forward, drop, consume or replicate an incoming packet. He can also transform the packet in a certain way before taking further action. A packet can be dropped due to buffer overflow conditions or due to specific filtering resulting from a QoS rate limiting function, for example. Special case packets that require processing by the control or management planes are consumed and forwarded to the appropriate plane. Finally, a special case of forwarding concerns multicast, where the incoming packet must be replicated before forwarding the different copies to different output ports. The logic and algorithms used to program the transfer plane reside in the control plane. Many of these protocols and algorithms require global knowledge of the network. The control plane determines how forwarding tables and logic in the data plane should be programmed or configured. Since in a traditional network, each device has its own control plane, the main task of this control plane is to execute routing or switching protocols so that all distributed forwarding tables across network devices remain synchronized. The most fundamental result of this synchronization is the prevention of loops.

Ultimately, the data transfer plane transmits the packets while the control plane tells the transfer plane how and where to process the packets.

- An abstraction across the higher layers of the network, which can aid in programming and open the door to automation. The protocols in SDN technology allow central reconfiguration of the network through transactions. A transaction is an elementary operation that acts on one or more devices. Typically, transactions are implemented as APIs for use by third-party programs and on separate software components from controllers for modularity. As a result, the control plane of current switches/router is moving to a central location, leaving behind a simple box with a data-only plane that can simply forward traffic as directed by a centralized SDN controller, as shown in figure 2.

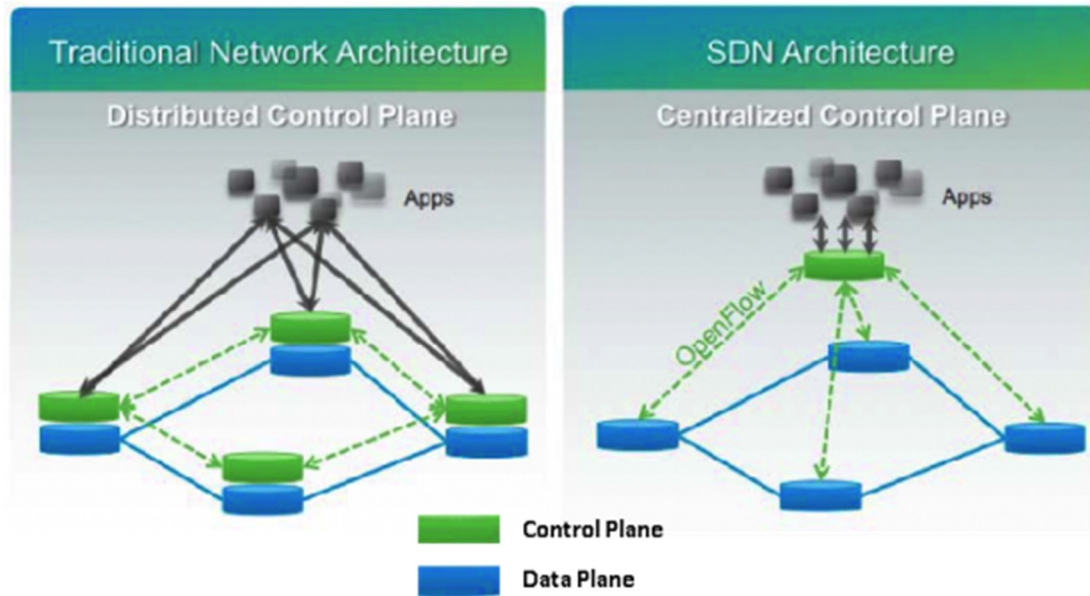


Figure 2. SDN — Separation of control and data planes (<https://mahieufraancois.wordpress.com/veille-technologique/>)

#### ➤ How SDN Technology Works

Conceptually, the behavior and operation of an SDN is simple. In Figure 3, we provide a graphical representation of how the basic components of SDN work: devices SDN, the controller and the applications. The easiest way to understand how it works is to look at it from the bottom up, starting with the SDN device. As can be seen in Figure 4, SDN devices contain functionality to decide what to do with each incoming packet. Devices also contain the data that drives these transfer decisions. The data itself is represented by the streams defined by the controller, as shown in the upper left of each device. A flow describes a set of packets transferred from one network endpoint (or set of endpoints) to another endpoint (or set of endpoints).

Endpoints can be defined as TCP/UDP IP address-port pairs, VLAN endpoints, layer three tunnel endpoints, and ingress port, among others. A set of rules then describes the actions the device should take for all packets belonging to that flow. A flow is unidirectional in the sense that packets that travel between the same two ends in the opposite direction could each constitute a separate flow. Flows are represented on a device as a flow entry.

A flow table resides on the network device and consists of a series of flow entries and the actions to take when a packet matching that flow arrives at the device. When the SDN device receives a packet, it consults its flow tables looking for a match. These flow tables had been built previously when the controller downloaded the appropriate flow rules to the device. If the SDN device finds a match, it takes the appropriate configured action, which usually involves forwarding the packet. If it doesn't find a match, the switch can either drop the packet or pass it to the controller, depending on the OpenFlow version and switch configuration. The definition of a flow is a relatively simple programming expression of what can be a very complex control plane calculation previously performed by the controller. For the reader less familiar with traditional switching hardware architecture, it is important to understand that this complexity is such that it simply cannot be run at line rates and must instead be digested by the control plane and reduced to simple rules that can be dealt with at this rate. The SDN controller is responsible for abstracting the network of SDN devices it controls and presenting an abstraction of those network resources to SDN applications running above. The controller allows the SDN application to define flows on devices and helps the application respond to packets that are transmitted to the controller by SDN devices. In Figure 3 we see the entire network that the controller maintains. This allows it to calculate optimal transmission solutions for the network in a deterministic and predictable way. Since a controller can control a large number of network devices, these calculations are normally performed on a high-performance machine, with an order-of-magnitude performance advantage over CPU and memory capacity over to what is usually granted to network devices themselves.

For example, a controller may be implemented on a two (2) GHz eight-core processor versus the more typical one-gigahertz single-core processor on a switch. SDN applications are built on top of the controller. These applications should not be confused with the application layer defined in the seven-layer OSI model of computer networks. Since SDN applications are actually part of network layers two and three, this concept is orthogonal to that of applications in the narrow hierarchy of OSI protocol layers. The SDN application interfaces with the controller, using it to define proactive flows on devices and to receive packets that have been transmitted to the controller. Proactive flows are established by the application; Typically, the application sets these streams at application start-up, and the streams persist until a configuration change is made. This type of proactive flow is called static flow. Another type of proactive flow is where the controller decides to modify a flow based on the traffic load currently being routed through a network device. In addition to flows pro actively defined by the application, some flows are defined in response to a packet transmitted to the controller. Upon receiving incoming packets that have been transmitted to the controller, the SDN application will instruct the controller how to respond to the packet and, if necessary, establish new flows on the device to allow that device to respond locally the next time it is received. 'it will see a packet belonging to this stream. These flows are called reactive flows. In this way, it is now possible to write software applications that implement forwarding, routing, overlay, multipath and access control functions, among others.

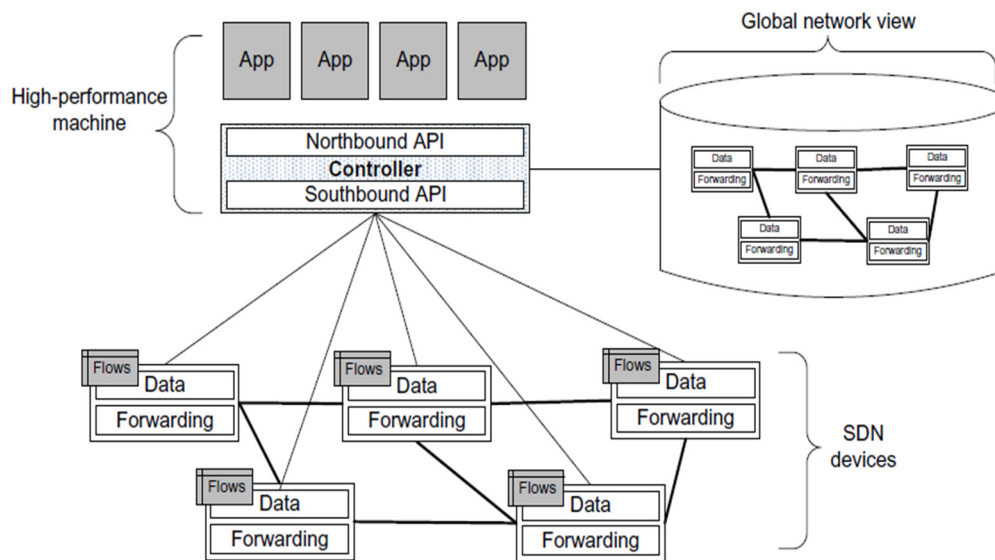


Figure 3. SDN Architecture and functioning (Oloumane, 2019)

### 2.3.2. Network Functions Virtualization

NFV has been standardized by the Industry Specification Group (ISG) of ETSI (European Telecommunications Standards Institute). It was born from the desire to accelerate the deployment of new network services. A will that was limited until then, in a way, by the hardware.

The NFV according to Figure 4 is presented as a means of reducing costs and accelerating the deployment of services for network operators, by decoupling functions (firewall, routing, data encryption, etc.) from dedicated hardware and transferring them to virtual servers.

Network functions based on physical devices are virtualized through a hypervisor, allowing the network to grow without having to add dedicated devices. All virtual machines are orchestrated by a single hypervisor. When the hypervisor controls network functions, services can be run on standard servers.

One can have as main characteristics of NFV:

- Reduction of material investment expenditure;
- Reducing the need for space, power and simplifying the management of network functions;

- Reducing the time needed to deploy new services;
- The facilitation of tests in the network during a migration or an update for example;
- Providing agility and flexibility by easily scaling services to meet new needs and allowing services to be delivered via software on standard servers.

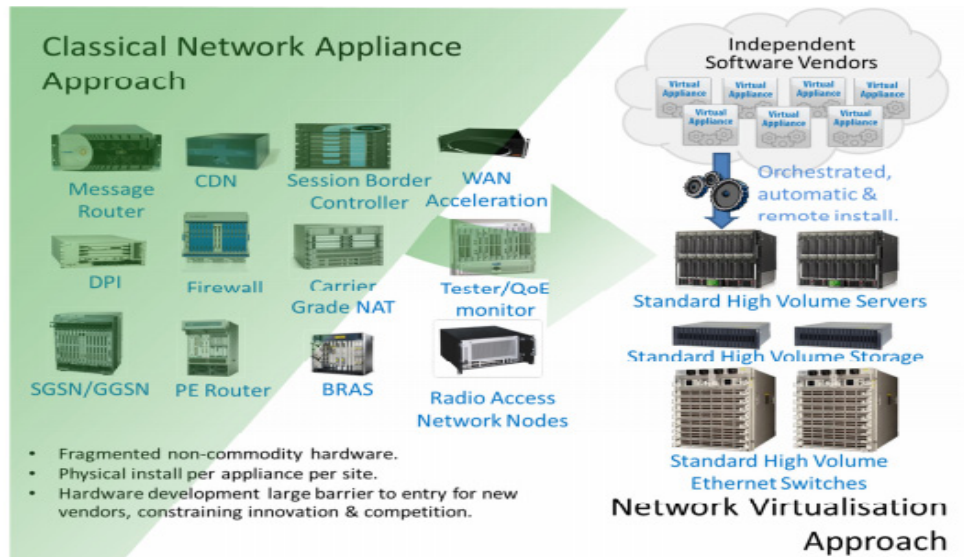


Figure 4. Architecture of the NFV approach (<https://mahieufraancois.wordpress.com/veille-technologique/>)

#### 2.4 IP/MPLS Networks

In order to fully understand the principles of MPLS, we must begin with its terminology. From an ISP's perspective, the client device, usually a router, used to connect to the ISP is called the client device (CE). On the other side, the ISP device used as the connection to the customer is called the provider (PE). Other routers in the ISP's network, which are only responsible for core operations and are not connected to external devices, are simply called provider (P) routers. From the MPLS perspective, all routers that run the MPLS protocol are called Label Switched Routers (LSRs). This terminology comes from the fact that the MPLS architecture uses labels to forward packets through the network instead of IP addresses, which is the traditional forwarding method. We can say that the P routers are in fact also LSR routers. Additionally, PE routers, from the MPLS perspective, can also be referred to as Label Edge Routers (LERs). The path taken by MPLS packets from their entry point in the ISP network to the exit LER is called the Label Switched Path (LSP) (Ottou, 2020). It is important to note that LSPs are unidirectional. The MPLS architecture can be seen in Figure 5.

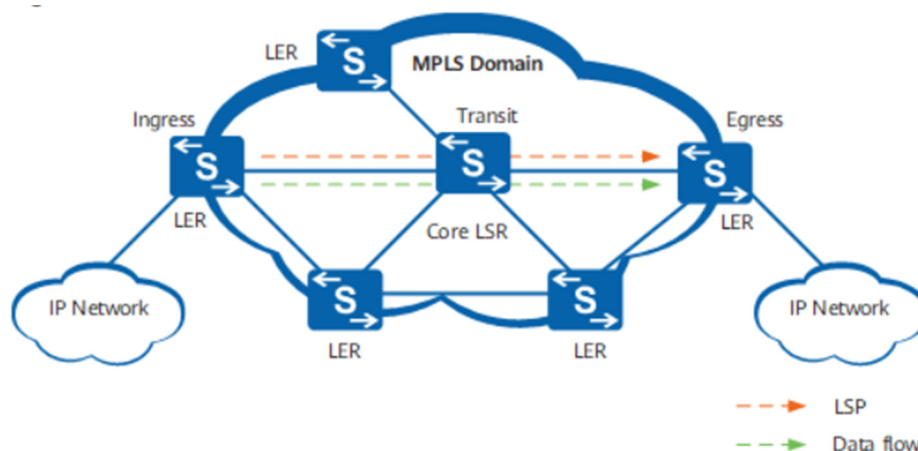


Figure 5. Sample MPLS Architecture (huawei.mpls\_overview)

MPLS constructs FECs according to many criteria: destination address, source address, application, QoS, etc.

When an IP packet arrives at an ingress LER, it will be associated with a FEC as shown in Figure 6. Then, just as with traditional IP routing, a routing protocol will be implemented to discover a path to at the LER egress. But unlike classic IP routing, this operation is only performed once. Then all packets belonging to the same FEC will be routed along this path which will be called Label Switched Path (LSP). An LSP is the path established through one or more LSRs to join multiple LERs within an MPLS network, configured solely via the labeling mechanism, for a particular FEC. It can be established statically or dynamically.

Thus we had the separation between the routing function and the switching function: the routing is done only at the first stage. Then all packets belonging to the same FEC will be single-switched through this discovered path.

In order for LSRs to switch packets correctly, the Ingress LER assigns a label called Label to these packets (label imposition or label pushing). Thus, if we take the example of Figure 6, LSR1 will know by consulting its switching table that any incoming packet with the label  $L=18$  belongs to the FEC such and therefore must be switched to an output such by assigning it a new label  $L=21$  (label swapping). This switch operation will be performed by all LSRs in the LSP until it reaches the Egress LER which will remove the label (label popping or label disposition) and route the packet back into the IP world in the traditional way, but like the operations routing are complex and expensive, it is recommended to perform the pop operation on the last LSR (Penultimate node) of the LSP (penultimate node of the LSP before the LER) to avoid overloading the LER unnecessarily.

A Penultimate node is the immediate router preceding the egress LER router for a given LSP within an MPLS network. It is the penultimate hop on an LSP. It plays a special role for optimization.

The routing of packets in the MPLS domain is therefore not based on IP address, but on label (switching label or labels).

It is clear that after the discovery of path (by the routing protocol), it is necessary to implement a protocol which makes it possible to distribute the labels between the LSR so that these last ones can constitute their tables of switching and thus carry out the switching of appropriate label for each incoming packet. This task is performed by “a label distribution protocol” such as LDP (Label Distribution Protocol) or RSVP-TE (ReSerVation Protocol-Traffic Engineering).

The three basic label operations (pushing, swapping, and popping) are all that is needed for MPLS. Label pushing/popping can be the result of FEC classification as complex as you like. Thus we will have placed all the complexity at the ends of the MPLS network while the heart of the network will only perform the simple function of label swapping by consulting the switching table.

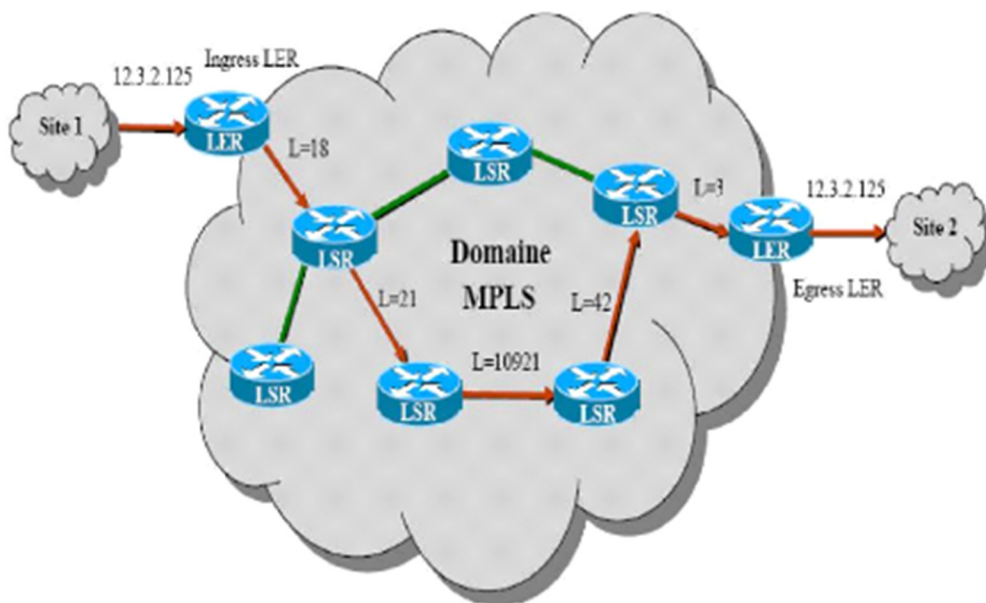


Figure 6. Label Switching in MPLS (Amine, 2010)

### ➤ Virtual Private Network (VPN)

MPLS/VPN provides a method of connecting sites belonging to one or more VPNs as shown in the example in Figure 7, with the possibility of overlapping IP addressing plans for different VPNs. Indeed, private IP addressing is widely used today, and nothing prevents several companies from using the same address ranges. MPLS/VPN makes it possible to isolate traffic between sites that do not belong to the same VPN, and by being totally transparent for these sites between them. In the MPLS/VPN perspective, a VPN is a set of sites placed under the same administrative authority, or grouped according to a particular interest. This part discusses the concepts of MPLS/VPN, in particular with the notions of virtual routers (VRF) and the MP-BGP protocol (Multi-Protocol Border Gateway Protocol), dedicated to the exchange of VPN routes.

Propagating the client's route through an MPLS/VPN network is done using the following process:

- ✓ Step 1: CE router sends IPv4 routing update to PE router
- ✓ Step 2: the PE router adds a 64-bit RD to the IPv4 routing update, resulting in a globally unique 96-bit VPNv4 prefix.
- ✓ Step 3: the VPNv4 prefix is propagated through an internal Multi-Protocol Internal Border Gateway Protocol (MPiBGP) session to other PE routers.
- ✓ Step 4: PE receiving routers strip the RD of the VPNv4 prefix, resulting in an IPv4 prefix.
- ✓ Step 5: the IPv4 prefix is forwarded to other CE routers in an IPv4 routing update.

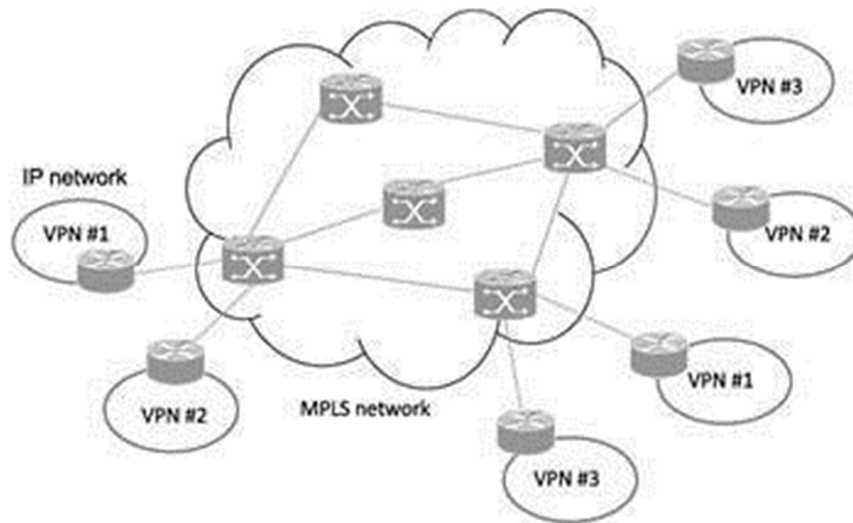


Figure 7. MPLS VPN Scenario (Oki, Rojas-Cessa, Tatipamula, & Vogt, 2012)

### ➤ Security requirements

#### Threats in IP/MPLS networks

We generally observe three main types of security threats or attacks against MPLS networks:

- ✓ Intrusions, where the underlying objective is to gain unauthorized access to resources;
- ✓ Denial of Service (DoS) attacks, or Distributed Denial of Service (DDoS) attacks where resources are no longer available to authorized users;
- ✓ Unauthorized access to equipment, which may be the cause of certain malfunctions.

The table below shows the types of threats, their descriptions and the countermeasures that can be taken for this purpose:

Table 1. Description of threats in MPLS networks

Threats	Description	Counter measure
Intrusion	Intrusions can come from any outdoor location that has connectivity to your network. These attacks can come from other VPNs, the Internet, or the service provider itself.	Protection against attack types comes from the ability to filter unwanted traffic from unwanted sources at network entry points.
	The PE is, by default, “visible” from the VPN user's network on layer 3. This does not pose a direct threat to a VPN, but to the kernel.	The PE must be specially secured with authentication, encryption and even IPS protocols.
	Most MPLS providers use private IP space for PLD, so you would have to enter that network either by intrusion or physical attack access with LDP.	It is possible to authenticate the PLD with MD5, if the provider uses this, then LDP would not be an attack vector.
Denied service	DOS attacks against service provider devices can also cause denial of service to parts of your VPN.	Although it can sometimes be difficult to protect your network against DOS attacks, the main protection against them lies in the proper design of the MPLS VPN network.
	These attacks differ in that for DOS it would be a question, for example, of sending 10 Gb/s of data from the same IP address to a target server, in order to saturate its connection which would only be 1 Gb/s, whereas, in the DDOS, it is about sending 1 Gb/s from ten (10) different servers, totaling 10 Gb/s and blocking a remote server with a 1 Gb/s connection. The result is the same, but this attack is harder to block.	
Unauthorized physical access	The MPLS packets themselves are not visible from the IP address, so you would need physical access to the network carrying the MPLS packets.	Security here is guaranteed by the restriction of access to equipment on the supplier side and on the customer side. An access control system must be installed at both the supplier and the customer.

#### ➤ SDN approach in IP/MPS network security

The SDN (Software Defined Network) architecture offers new opportunities for implementing security mechanisms in terms of detecting unauthorized activities. At the same time, there are certain risks associated with this technology. The approach presented covers a design of the measurement method, the virtual test bed and the classification mechanism of the SDN. The collected data set can be used in machine learning methods to detect unauthorized activities.

After having studied the operation of an SDN controller in an SDN architecture, we will see how it manages attacks coming from vectors and how it detects them thanks to traffic flow control algorithms.

#### ➤ SDN attack vectors

The SDN architecture has a big impact on the class of attack that can be performed, as shown in Figure 8. The most dangerous situation occurs when the SDN controller is compromised. This can be done by exploiting vulnerabilities in processes and services running on the controller. Therefore, the entire SDN domain is compromised and the attacker has the ability to take control of all network devices. The degree of vulnerability of such attacks mainly depends on the hardware implementation of the SDN controller, the programming languages and the libraries used. Threat prevention can use IDS and IPS techniques, as well as methods of replication and recovery of the state of SDN servers before the attack. Due to the nature of SDN technologies, traditional IDSs may be insufficient. A potential security vulnerability may also exist on the administrative station, which is used to manage the network operating system (SDN controller). These terminals are used to develop applications for

network equipment control logic and to monitor network activities.

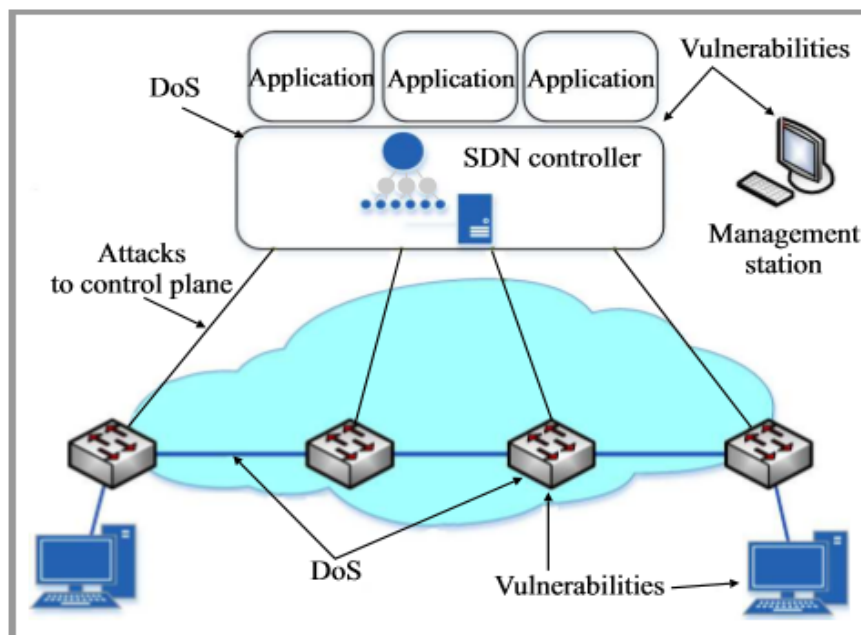


Figure 8. Potential attack vectors in SDN technology (Jankowski & Amanowicz, 2015)

The traffic that crosses the IP/MPLS network comes from an end user, the attack vector can be a computer or a management station of the SDN controller itself. Thus, the traffic that will be routed through the network will already be corrupted. Other threats are attacks on the communication flow between the data and the control plane. For example, the security protocol between controllers and network devices can be TLS. A potential attack vector would exploit vulnerabilities in its implementation.

#### ➤ SDN Intrusion Detection

It is important to detect intrusions in a network in order to mitigate them. Attacks in a network through interconnected equipment create much more widespread damage compared to those that cross a local network. The table below shows the different network intrusion detection methods for SDN:

Table 2. Intrusion Detection Method Table (Jankowski & Amanowicz, 2015)

Method detection	of	Principle	Extraction of attacks	Logic algorithm	Attacks detected	Network traffic
<b>Method 1</b>		Evaluation of the first packet transmitted to the SDN controller	Maximum entropy detector, TRW-CB, Rate-limiting, NETAD	N/A	DoS, intrusion	Favorable/benign, real SDN network, SDN attacks, artificial traffic.
<b>Method 2</b>		Threat level assessment	TRW-CB, rate-limiting	Fuzzy logic	DoS	Artificial traffic
<b>Method 3</b>		Creation of profiles using sFlow, Openflow	TRW-CB, entropy level	N/A	DDoS, intrusion propagation	Favorable, real SDN network, SDN attacks, artificial traffic.

<b>Method 4</b>	Collection of flow statistics	Based on stream statistics	Organized maps, artificial neural network	DDoS	Artificial traffic
-----------------	-------------------------------	----------------------------	---	------	--------------------

### 3. Materials and Method

#### 3.1 Flowchart of Methodological Steps

The flowchart in Figure 9 summarizes the Method used

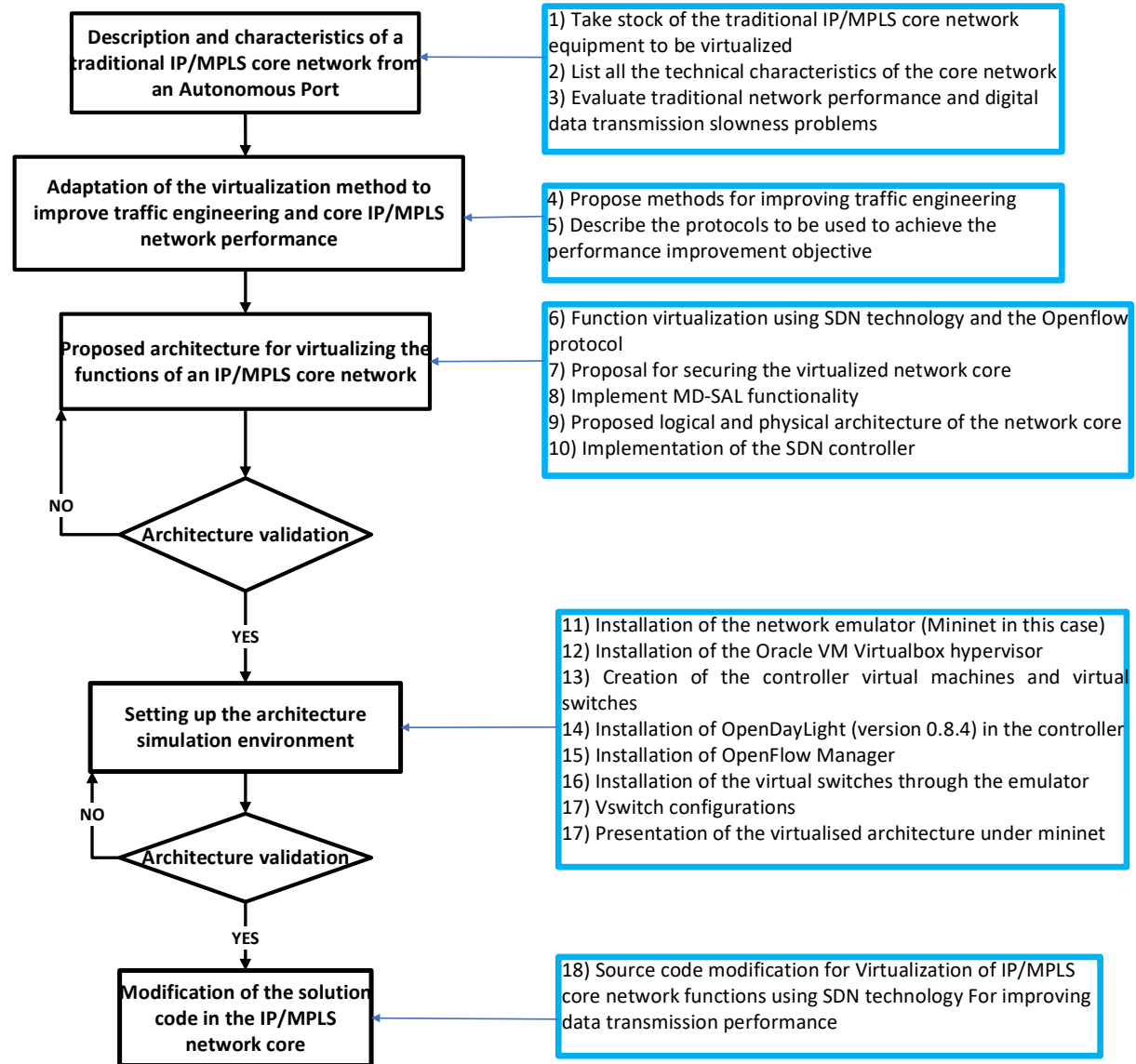


Figure 9. Methodological diagram of the virtualization of the functions of an IP/MPLS core network of an Autonomous Port

#### 3.2 Core Characteristics of the IP/MPLS Network of an Autonomous Port

The Autonomous Port's IP/MPLS network in our work consists of routers in all its sites, which interconnect them. The company network traffic is routed through the transparent cloud of the operator whose routers (fig. 18) constitute elements of this IP/MPLS cloud. In addition, the Port d'Autonome, as a port authority, is responsible for all electronic communications infrastructure and services in the port area. As a result, companies setting up in the port area have the Autonomous Port as their Internet access provider and various IT services. An optical fiber mesh

was made to allow partners to have access to services. Our network is made up of P and PE routers, with CE client routers at the partners. The virtualization of network functions will allow this entire heart of the network to accelerate the flow of information passing through the cloud.

Figure 10 presents the current architecture of the core of the IP/MPLS cloud of an Autonomous Port including the various partners. We have the router specifications summarized in Table 3:

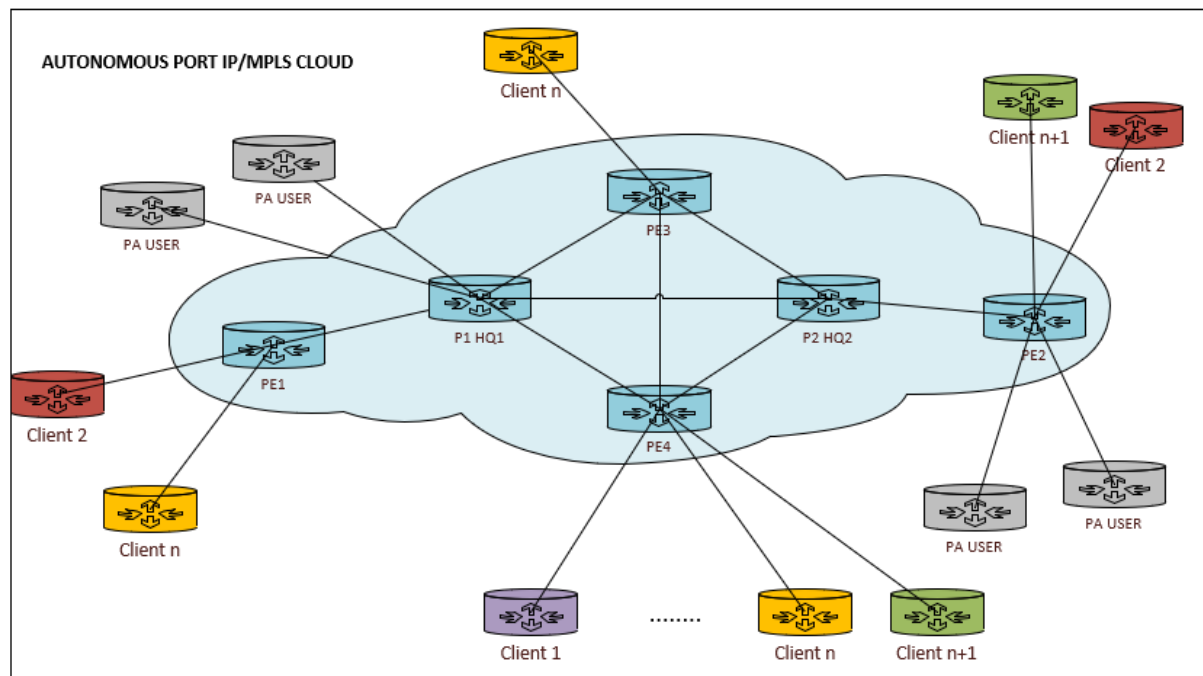


Figure 10. Autonomous Port IP/MPLS network core with partners

Table 3. Core IP/MPLS Network Router Specifications

<b>Characteristics</b>		
<b>Physical Specifications</b>		
GE RJ45 Ports		16
GE SFP Slots		16
<b>System performance</b>		
IPS Throughput		7.8 Gbps
NGFW Throughput		6 Gbps
Threat Protection Throughput		5 Gbps
Firewall Throughput (1518/512/64 bytes UDP packets)		32 / 32 / 24 Gbps
Firewall Throughput (Packet Per second)		36 Mpps
Concurrent Sessions (TCP)		4 Million
New sessions/second (TCP)		450,000
IPsec VPN Throughput (512 bytes)		20 Gbps
SSL Inspection Concurrent Session (IPS,avg.HTTP)		300,000
Application Control Throughput		12 Gbps
Maximum Number of Forti switches supported		48
Maximum Number of FortiAPs (Total/Tunnel Mode)		512 / 256
<b>Environment</b>		
Power Consumption (Average /Maximum)	109 W / 214 W -115 W / 221 W; 6A,	100–240V 60–50Hz
Heat Dissipation	730 BTU/h	754 BTU/h

#### ➤ Virtualization for Traffic Engineering Improvement

Multiprotocol Label Switching (MPLS) was created to improve packet forwarding performance at the core of networks, and it is widely used for this purpose. It has also been adapted for other use cases, and one of the most important is traffic engineering.

MPLS allows network engineers to optimize network resources by sending traffic over a less congested physical path, rather than the default shortest path designated by the routing protocol. This is achieved by appending a short label to each packet with specific routing instructions that direct packets from the router to the next router, rather than allowing routers to forward them based on next hop searches. New paths can be created manually or through signaling protocols, and they help speed up traffic.

##### ✓ Configuring the MPLS tunnel

Traffic engineering can significantly improve the performance of MPLS networks. If you have already deployed MPLS in your network — perhaps for a VPN — MPLS traffic engineering can be very beneficial. I explain how specific traffic paths are defined and calculated using routing attributes and protocols, design criteria, and other design-centric issues to consider.

##### ✓ MPLS Path Selection for Bandwidth Optimization

MPLS traffic engineering has three main uses. These include optimizing bandwidth by selecting an alternate path, supporting a service level agreement (SLA), or enabling fast rerouting. We will describe the use of label-switched paths for bandwidth optimization.

##### ✓ Meeting SLAs with MPLS Traffic Routing

MPLS traffic engineering can be used to meet an SLA. Not all traffic is the same and not all customers or sites can benefit from the same service. Voice and video traffic has traditionally been carried over circuit-based TDM links. These applications are very sensitive to delays and losses, so we need to ensure that they are properly supported on the packet-switched network.

##### ✓ MPLS Fast Rerouting

The most common use of MPLS traffic engineering is fast rerouting. The basics of fast rerouting and how to use it to create backup and preconfigured paths in MPLS traffic engineering are important in setting up a traffic engineering solution.

However, we can find alternatives to traffic engineering that will further improve the routing and switching processes in a network with regard to the desired performance.

Integrating many new control plane features into our network can be too complex if our network is not centrally managed and virtualized. Troubleshooting, management, user training, control plane health, and data plane health can all be concerns. Fortunately, the equipment described supports the Open Shortest Path First (OSPF), Intermediate System to Intermediate System (IS-IS) and Resource Reservation Protocol (RSVP) protocols; therefore, we are considering the IP Fast Redirection (FRR) feature.

Two types of FRR IP include loop-free alternate (LFA) and remote LFA. Depending on the topology, LFA might provide fairly fast forwarding capability for your primary path, and in that case we wouldn't need MPLS. Some topologies may require a remote LFA to cover outages. The difference is that the traffic will be sent to a node that will not send it back, resulting in a micro loop.

Service providers, such as Autonomous Port, may not need MPLS traffic engineering where it will only be used to protect backbone paths. For example, they may have protected dense wavelength division multiplexing (DWDM) to enable automatic protection switching in their transport systems. This provides them with sub-50ms backup paths in the event of a failure.

Another scenario occurs when Enhanced Interior Gateway Routing Protocol (EIGRP) is built using a mechanism similar to fast forwarding, called EIGRP "Feasible Successor" (FS). In this system, all loop-free alternate paths may be kept in the EIGRP topology database or may be used unequally. (EIGRP is like MPLS traffic engineering in supporting unequal cost multipath.) If unequal cost multipath is not used, but another loop-free path exists in the database topological, when a primary path fails, EIGRP will not send a request. It will only run a Broadcasting Update Algorithm (DUAL) to select a successor among possible successors and install the best path in the routing and forwarding information databases.

EIGRP FS is different from MPLS traffic engineering and other fast forwarding mechanisms. Alternate topology information still needs to be calculated because the prefixes are not found in the control or data planes. You don't need to use the FRR IP feature for EIGRP, but EIGRP FS still seems slower than FRR.

Finally, Software Defined Networking (SDN) and using OpenFlow can give us the ability to calculate topology information because the controller has a centralized view and can upload forwarding information directly to the transfer table. MPLS traffic engineering would be difficult to enable, due to the lack of complete topology visibility, but with SDN traffic redirection it can be easier and even faster.




### 3.3 Conception

#### ➤ In terms of network security

The centralized network security solutions proposed with a single controller, or several, but which are redundant and not distributed, do not ensure security in its entirety, because a denial of service, or a remote control of the controller by a hacker, can render the network inoperative. This can be avoided by using multiple distributed controllers in the MPLS network. In practice, the MD-SAL (Model-Driven Service Abstraction Layer) functionality provided by the OpenDaylight controller makes it possible to implement redundancy and high availability with several SDN controllers. Also, threats from users through the use of USB keys, Internet access modems or connected objects must be detected and isolated. This is why we propose a new architecture for securing exchanges in a network, extended to the IoT.

As shown in Figure 11, this solution is to create multiple clusters. Each cluster is made up of one or more network devices that are responsible for the interconnection. Within each cluster, an SDN controller manages the OpenFlow network. Each SDN controller is paired with an IDS. The IDS is responsible for detecting intrusions within the network perimeter of each cluster. In other words, a cluster constitutes an SDN domain in which we use an OpenFlow network with an SDN controller and an IDS to manage the security domain, which we call a trust zone. To form a trusted zone, all the equipment in this zone must be completely secure. This safety work is carried out by the controller and IDS couple. The SDN controller serves as a smart firewall for the trusted zone and has security rules specific to the cluster's security needs.

#### Legend:

-  link between Controllers
-  OpenFlow Link
-  Physical link

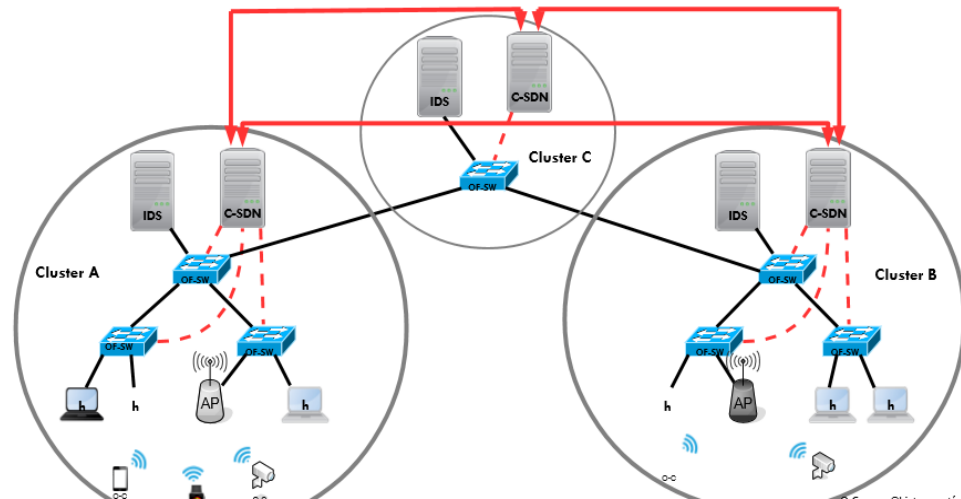


Figure 11. Distributed and collaborative architecture for securing a network with SDN (Mahamat Charfadiene, 2019)

In the case of a virtualized network, it is a question of setting in the SDN controller. The SDN controller is the brain of the whole network and its security is particularly important. The SDN controller is more susceptible to attack because it is logically centralized. South-facing APIs and protocols are also vulnerable to security threats and attacks. The security of the SDN controller is directly related to the reliability and availability of the entire network. For example, if someone manages to completely attack the SDN controller by launching a denial of service (DoS), attack and steal the permissions of the SDN controller, it will affect the entire network, leading to serious security problems. Considering its importance, we need to implement dedicated security hardening policies for the SDN controller.

➤ In terms of the virtualization of network functions

In a traditional network, the control and data planes sit within the network element, which makes network management quite difficult since applications are directly implemented on the network elements as shown in Figure 12.

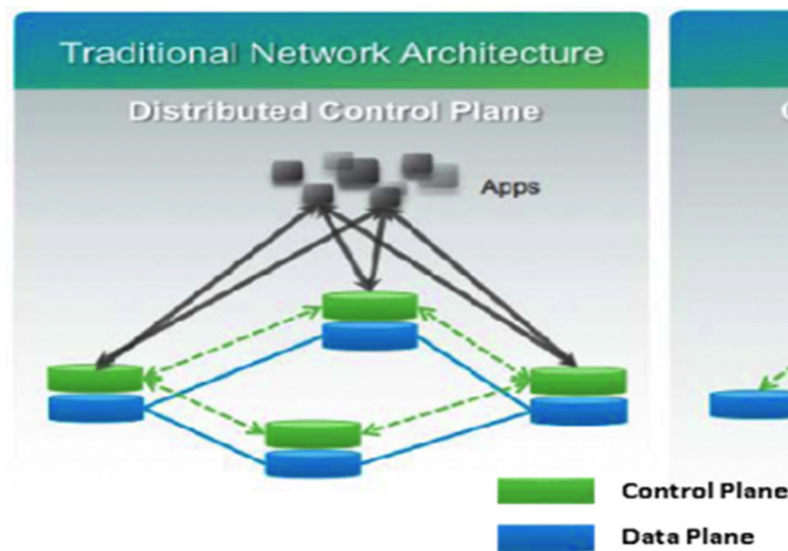


Figure 12. Traditional application management architecture in the network  
(<https://mahieufraancois.wordpress.com/veille-technologique/>)

The objective of our work is to separate the control and data planes as shown in Figure 13.

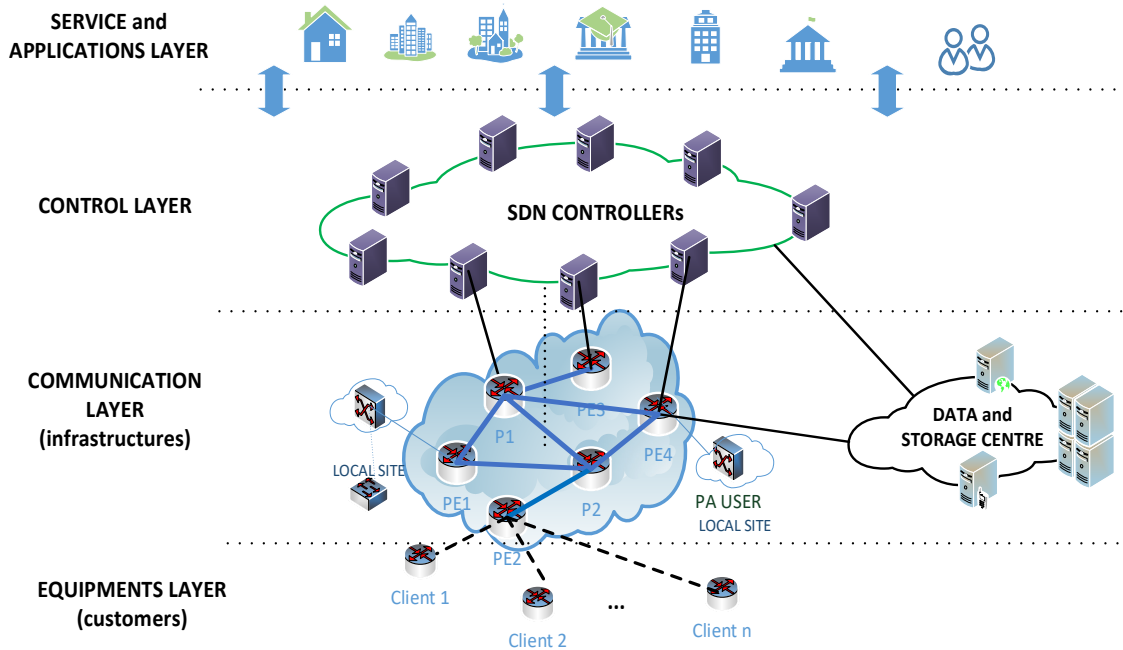


Figure 13. Target architecture of network virtualization through SDN technology

### 3.4 Simulation Scenario

For our simulation, we will implement a core network formed by 4 P routers, 2 PE routers managed by an SDN controller. In order to virtualize the routing function, we will replace the routers with virtual switches each installed in a separate virtual machine.

#### ➤ Choice of solutions

The choices of our solutions for virtualization are:

- ✓ MININET (Network Emulation Network): free software allowing network emulation;
- ✓ ORACLE VM VirtualBox: Free machine virtualization software published by Oracle. This software makes it possible to run several machines at the same time.
- ✓ OPEN DAYLIGHT: It is a programmable SDN controller, the source code of which will be modified to improve the performance of network virtualization and the key performance indicators of said network;
- ✓ OPEN VSWITCH: Application allowing to create several virtual switches;
- ✓ OPENFLOW MANAGER: For the supervision of flows in the virtualized network.

After installing the tools that will allow us to virtualize our core IP/MPLS network of an Autonomous Port. However, we have not yet simulated the network. The creation of the infrastructure to be simulated will be done on the Mininet tool by writing a script in Python. All the results will be presented in the results and simulations section.

#### ➤ Virtual Switch Network Configurations

Assign the host private network card so that all VMs communicate with each other and also communicate with the host machine. Our virtual switches will be created directly from the emulator, these switches will be Open vswitch already integrating the OpenFlow protocol as shown in Figure 14. The basic elements of the OpenFlow or Vswitch switch are the OF tables.

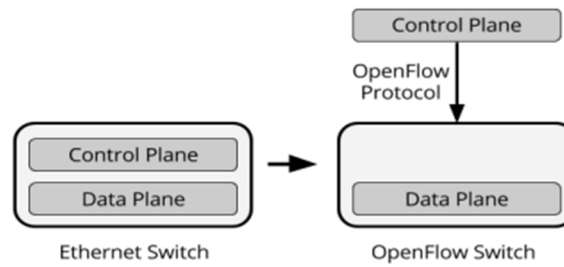


Figure 14. Architecture of the OpenFlow switch

➤ Architecture under MININET

To launch the emulator in which the architecture of our core IP/MPLS network to be virtualized will be edited, we will type the command: **/mininet/examples/miniedit.py** in the Mininet terminal.

This command allows us to display the representation interface of our architecture, as shown in Figures 15 and 16.

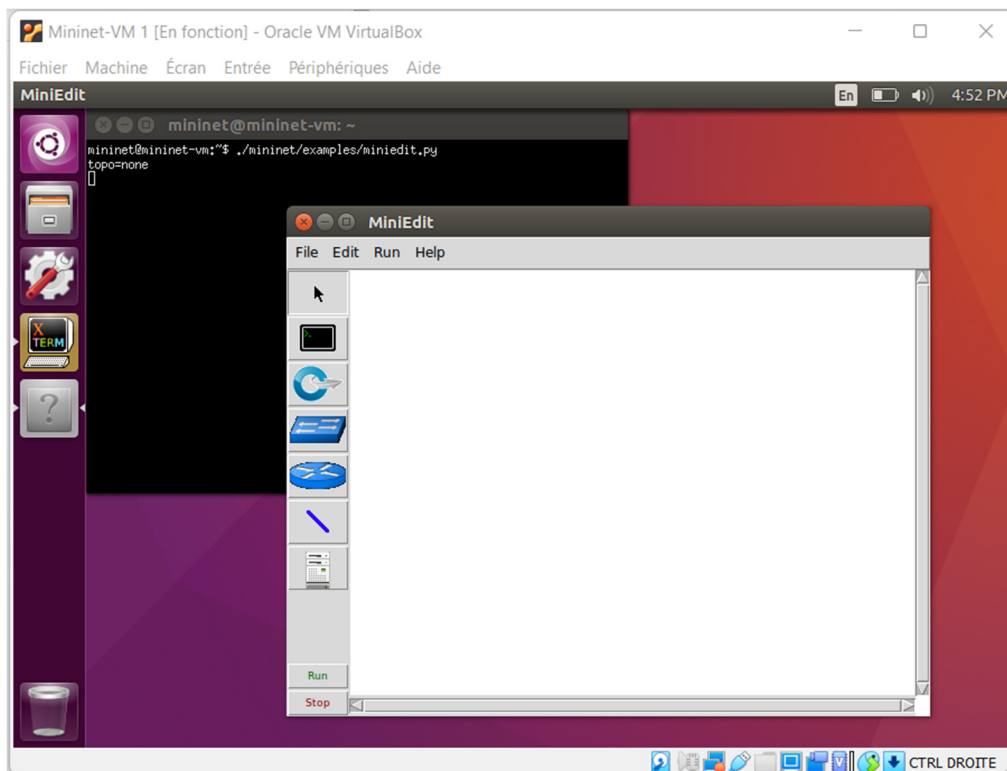


Figure 15. Miniedit interface for drawing the architecture

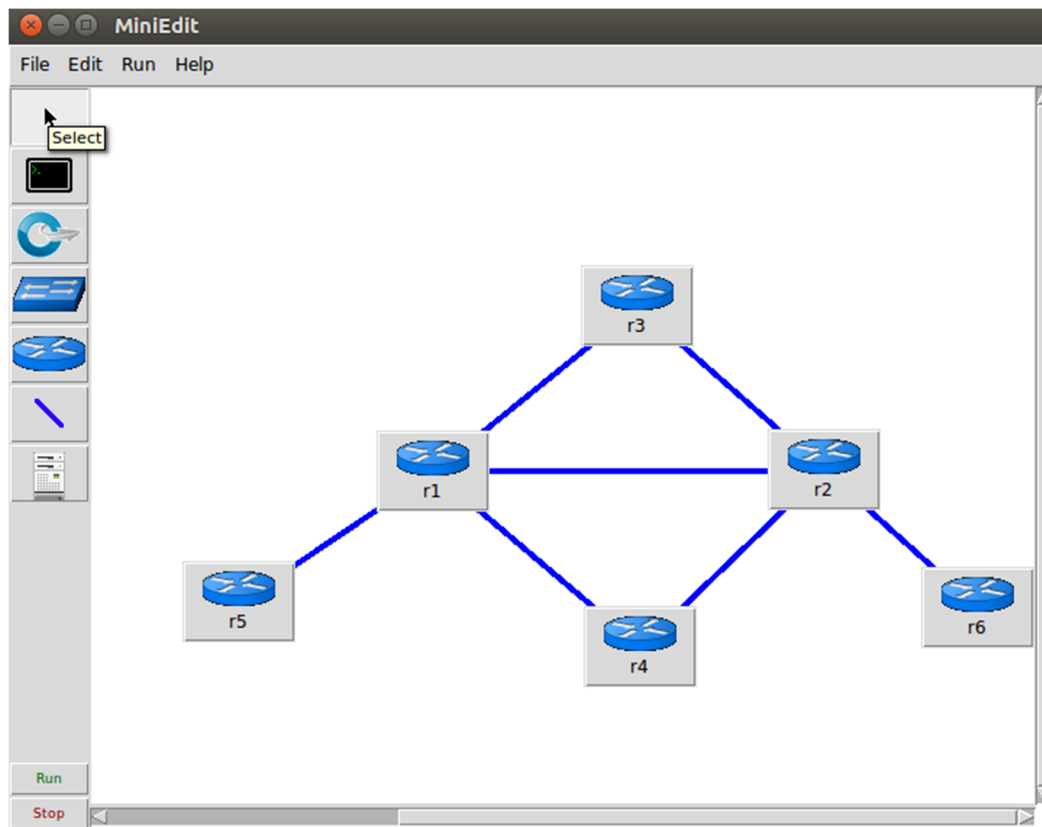


Figure 16. Representation of the heart of the traditional IP/MPLS network of an Autonomous Port

#### ➤ Configurations in OPEN VSWITCH

In order for the topology to be observed as it is seen under MININET, it is important that the configurations of the virtual switches are well done. Let's recap the procedure:

- Emulate the network in Mininet through Miniedit;
- Run virtual machines with virtualbox;
- Use OPENDAYLIGHT as an SDN controller;
- Use OPEN VSWITCH for the transfer plan;
- Use the Openflow manager module to manage the different flows.

## 4. Results and Comments

### 4.1 Migration to a Virtualized Network Core

#### 4.1.1 Architecture of a traditional Autonomous Port IP/MPLS cloud network

Figure 17 shows us the architecture of the core network of an Autonomous Port to be virtualized in view of the security issues, fluidity and availability of the information system.

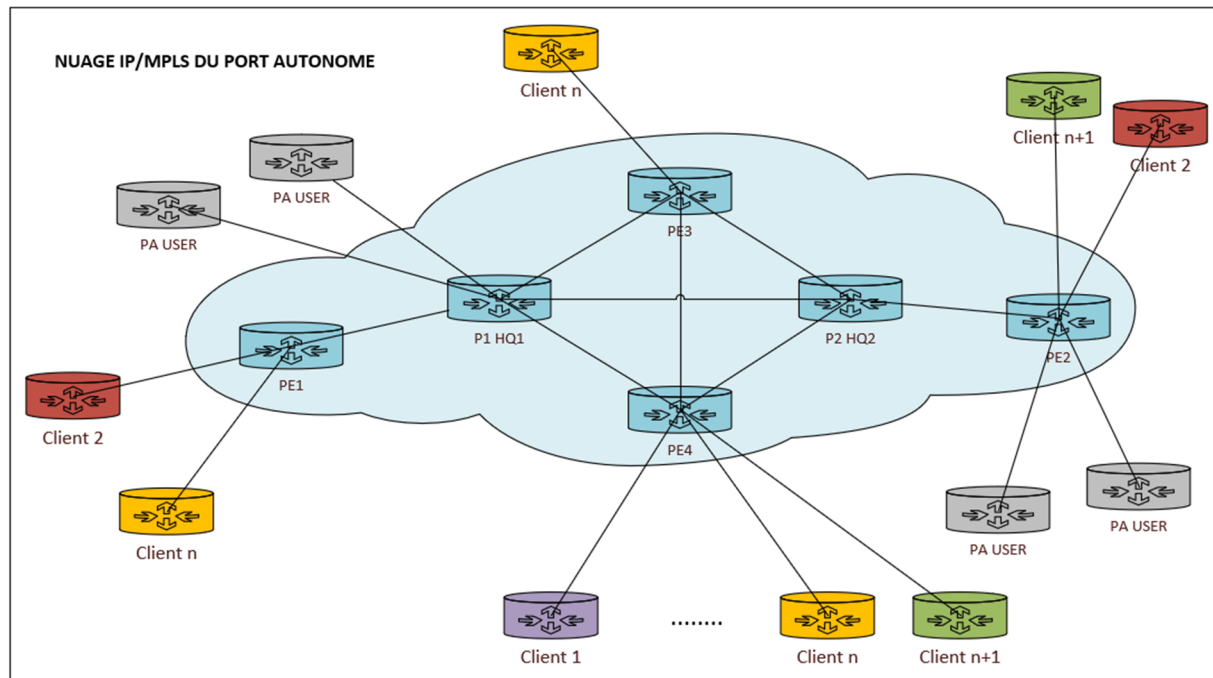


Figure 17. IP/MPLS network core cloud of an Autonomous Port to be virtualized

It should be recalled that this architecture is that of the core IP/MPLS network of an Autonomous Port which is multi-site in the country, serving the sites where the personnel are deployed, and the various industries and companies installed around its area of operation.

In a fourth-generation PA, fluidity in communications is highly demanded. The flow generated by communications between more than a thousand (1000) connected devices would increase latency in a traditional network. In addition, communications are intended to be secure between the sites of each “client”.

We propose to simulate the virtualization of the functions of this core network. This virtualization would further reduce latency in the network by centralizing it, because all the control processes managed in each device will now be managed by a controller: SDN controller on the one hand. On the other hand, the data streams will be analyzed through the OFM software where we will see the improvement in latency.

#### 4.1.2 Adaptation of the IP network core virtualization method

Once implemented, SDN technology would relieve the traditional network of the control plane to leave only the data transmission part. Control of data transmissions and distribution of loads in the network would be done automatically according to congested areas. The desired architectures are shown in Figures 18 and 19 below:

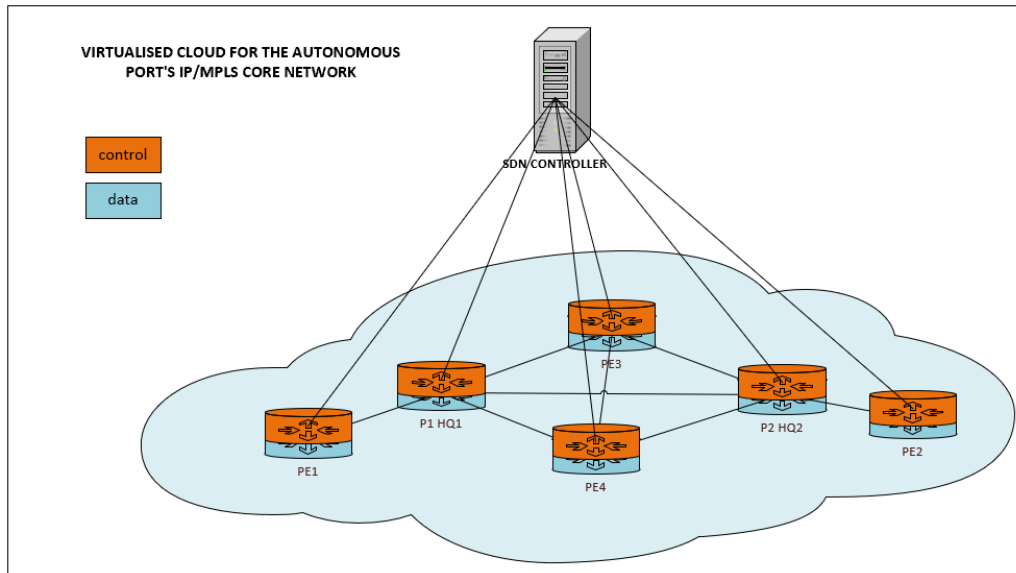


Figure 18. Virtualized architecture of the IP/MPLS core network of an Autonomous Port, with separation of the control and data planes

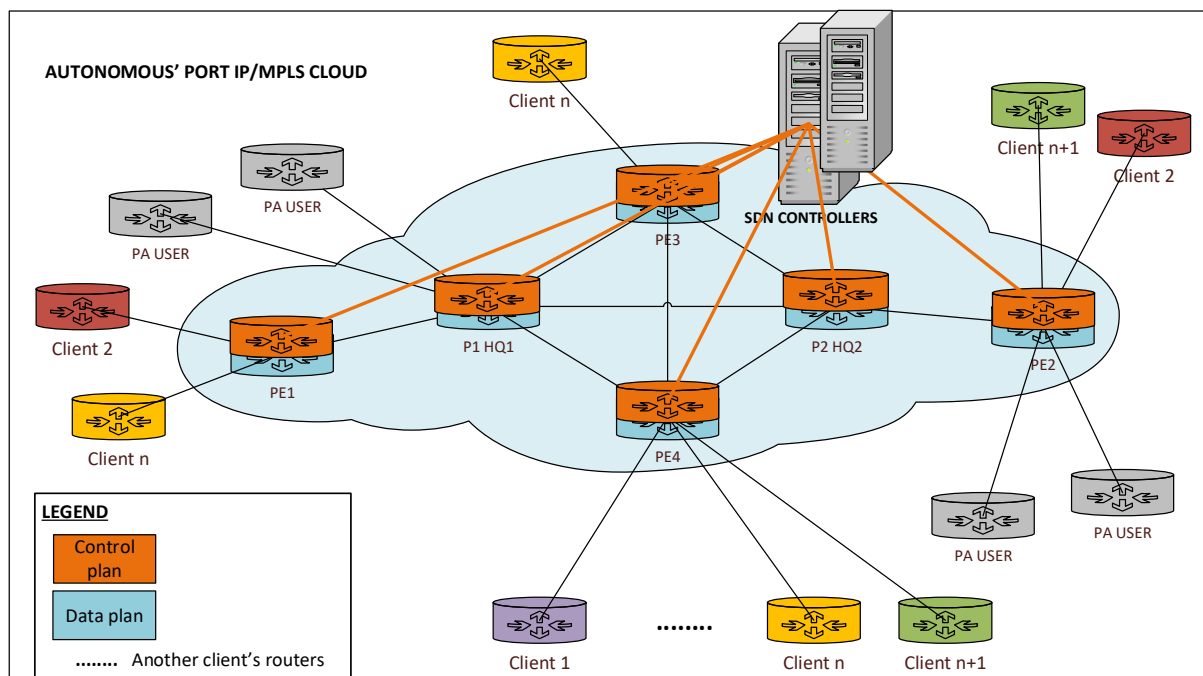


Figure 19. Virtualized architecture of an IP/MPLS core network of an Autonomous Port with the sites

### 4.2 Implementing IP/MPLS Core Network Function Virtualization

#### 4.2.1 Separation of control and data planes

It's worth remembering the fundamentals of SDN plane separation: a simplified appliance and central control,

automation, and network virtualization. The levels of abstraction will be represented in the simulation of our network. The data or forwarding plane is responsible for forwarding packets based on the forwarding tables. The control plane consists of the protocols used to manipulate the data transfer mechanisms, the management plane relies on the software tools that are used to influence the protocols in the control plane. This separation is the foundation of SDN.

Figure 20 represents the virtualized architecture in our simulation environment.

Legend :

 SDN Controller

 OpenFlow Link

 MPLS Router

 End User

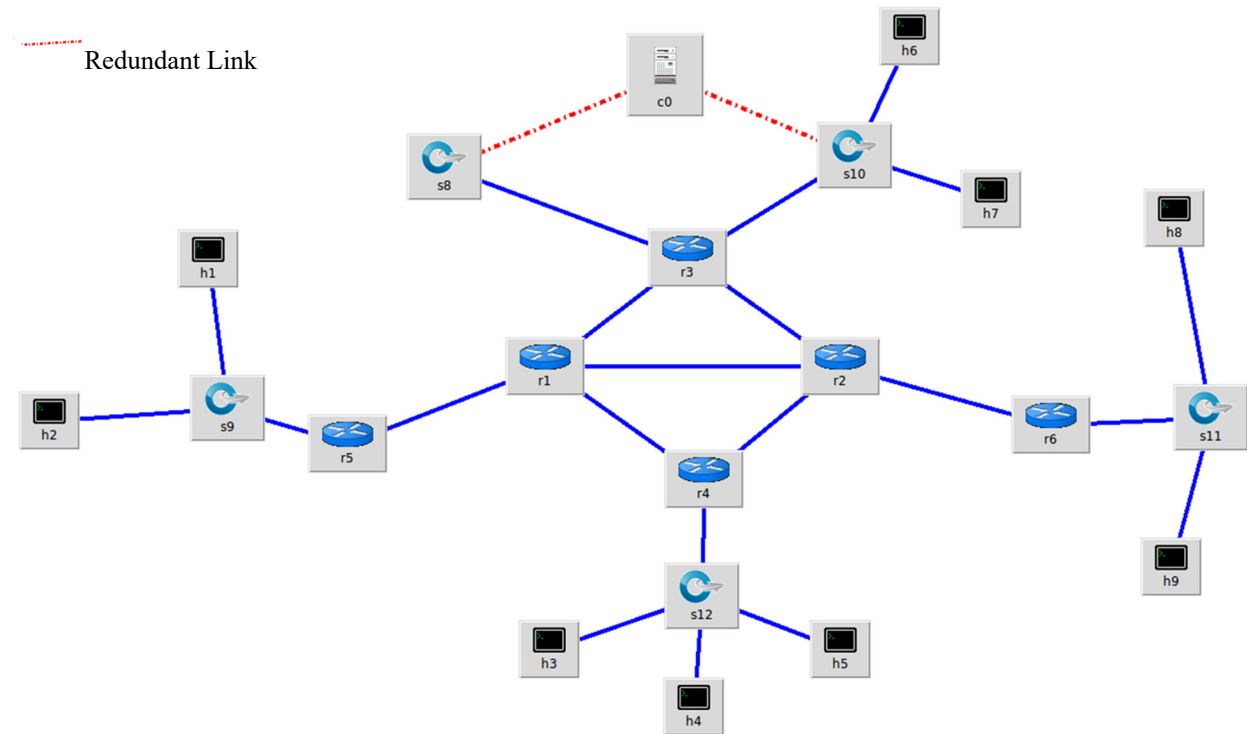
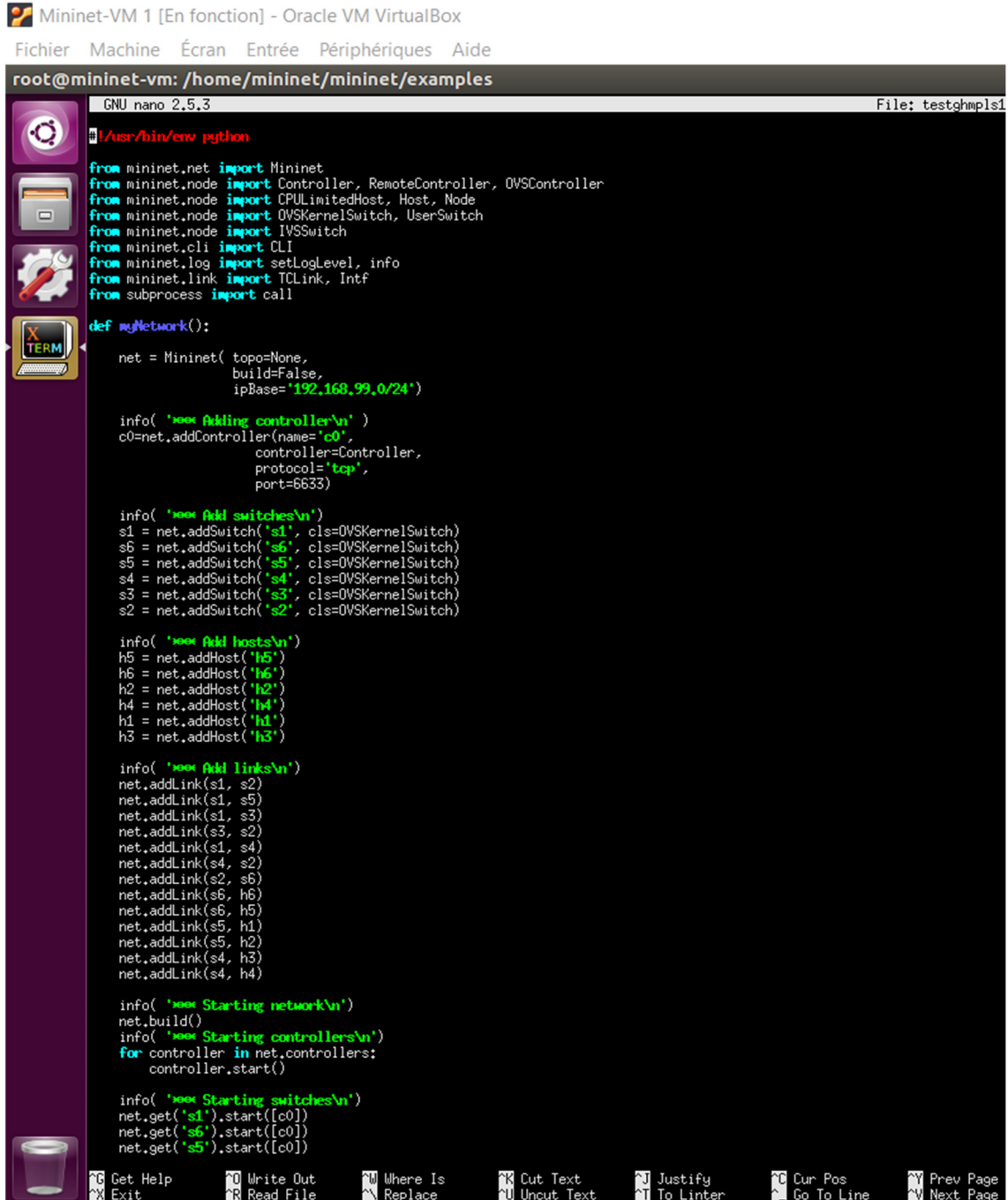


Figure 20. Virtualized architecture in mininet with an SDN controller

Then we make sure that the generated python script of this architecture which will be executed in mininet to interface with OpenDayLight is correct.

We put ourselves in the Custom directory of mininet, and type the command `nano testghmpls1.py` (The name of the file of our architecture within the framework of our simulation). We get the entire script as depicted in Figure 21 below.



The screenshot shows a terminal window titled "Mininet-VM 1 [En fonction] - Oracle VM VirtualBox". The terminal prompt is "root@mininet-vm: /home/mininet/mininet/examples". The user has opened a file named "testghmpls1" in the GNU nano 2.5.3 editor. The script defines a network topology using Mininet, including a controller, switches, hosts, and links.

```
#!/usr/bin/env python

from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.node import IVSSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
from subprocess import call

def myNetwork():

    net = Mininet( topo=None,
                  build=False,
                  ipBase='192.168.99.0/24')

    info( '*** Adding controller\n' )
    c0=net.addController(name='c0',
                        controller=Controller,
                        protocol='tcp',
                        port=6633)

    info( '*** Add switches\n' )
    s1 = net.addSwitch( 's1', cls=OVSKernelSwitch)
    s6 = net.addSwitch( 's6', cls=OVSKernelSwitch)
    s5 = net.addSwitch( 's5', cls=OVSKernelSwitch)
    s4 = net.addSwitch( 's4', cls=OVSKernelSwitch)
    s3 = net.addSwitch( 's3', cls=OVSKernelSwitch)
    s2 = net.addSwitch( 's2', cls=OVSKernelSwitch)

    info( '*** Add hosts\n' )
    h5 = net.addHost( 'h5')
    h6 = net.addHost( 'h6')
    h2 = net.addHost( 'h2')
    h4 = net.addHost( 'h4')
    h1 = net.addHost( 'h1')
    h3 = net.addHost( 'h3')

    info( '*** Add links\n' )
    net.addLink(s1, s2)
    net.addLink(s1, s5)
    net.addLink(s1, s3)
    net.addLink(s3, s2)
    net.addLink(s1, s4)
    net.addLink(s4, s2)
    net.addLink(s2, s6)
    net.addLink(s6, h6)
    net.addLink(s6, h5)
    net.addLink(s5, h1)
    net.addLink(s5, h2)
    net.addLink(s4, h3)
    net.addLink(s4, h4)

    info( '*** Starting network\n' )
    net.build()
    info( '*** Starting controllers\n' )
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches\n' )
    net.get('s1').start([c0])
    net.get('s6').start([c0])
    net.get('s5').start([c0])
```

Figure 21. Editable python script of virtualized architecture on mininet

We start the script to view it in the OpenDayLight controller and observe the flows via OpenFlow Manager. Figure 22 presents the results obtained by simulating just the heart of the IP/MPLS network of an AP.

```

root@mininet-vm:/home/mininet/mininet/custom# ifconfig eth0 192.168.99.59
root@mininet-vm:/home/mininet/mininet/custom# ping 192.168.99.57
PING 192.168.99.57 (192.168.99.57) 56(84) bytes of data:
64 bytes from 192.168.99.57: icmp_seq=1 ttl=64 time=0.778 ms
64 bytes from 192.168.99.57: icmp_seq=2 ttl=64 time=0.583 ms
^C
--- 192.168.99.57 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt: min/avg/max/mdev = 0.583/0.680/0.778/0.100 ms
root@mininet-vm:/home/mininet/mininet/custom# sudo mn --custom test2601a.py --to
o mytopo --mac --controller=remote,ip=192.168.99.57,port=6633
** Creating network
** Adding controller
** Adding hosts:
1 h2 h3 h4 h5 h6
** Adding switches:
1 s2 s3 s4 s5 s6
** Adding links:
s1, h3) (s1, h4) (s1, s3) (s2, s3) (s3, s6) (s4, s1) (s4, s2) (s4, s5) (s5, h1)
(s5, h2) (s6, h5) (s6, h6)
** Configuring hosts
1 h2 h3 h4 h5 h6
** Starting controller
0
** Starting 6 switches
s1 s2 s3 s4 s5 s6 ...
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6
h2 -> h1 h3 h4 h5 h6
h3 -> X h2 h4 h5 h6
h4 -> X h2 h3 h5 h6
h5 -> h1 h2 h3 h4 h6
h6 -> h1 h2 h3 h4 h5

```

Figure 22. Command to start the simulation of our architecture

Figure 23 shows the core network architecture seen from the OPENDAYLIGHT controller

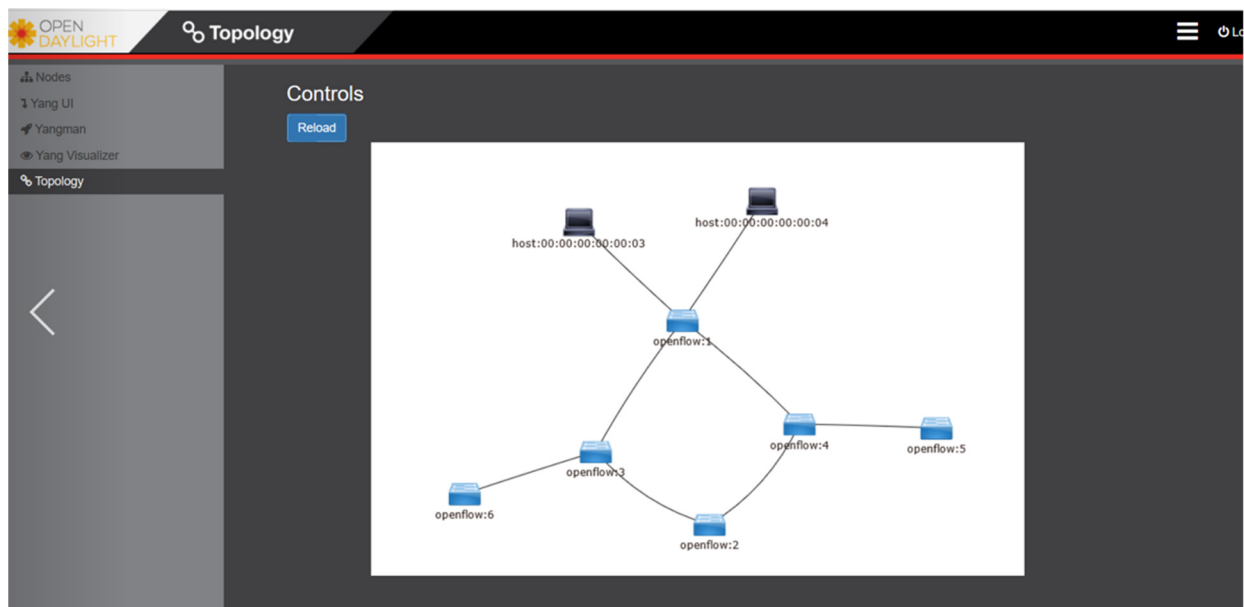


Figure 23. Core network architecture seen from the OPENDAYLIGHT controller

Figures 24, 25 and 26 respectively present the topology of the network on the YANG visualizer on the ODL controller, the Visualization of the nodes of the network through the ODL controller and the List of the different labels that can be obtained through the architecture of the network on ODL.

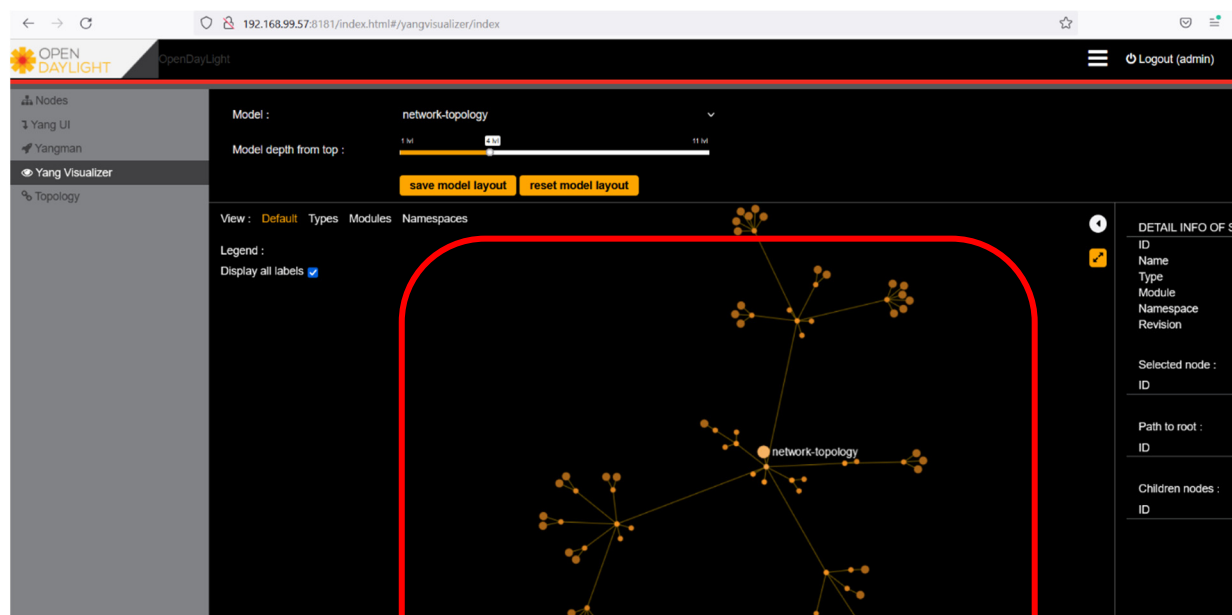


Figure 24. Network topology overview on YANG visualizer on ODL controller

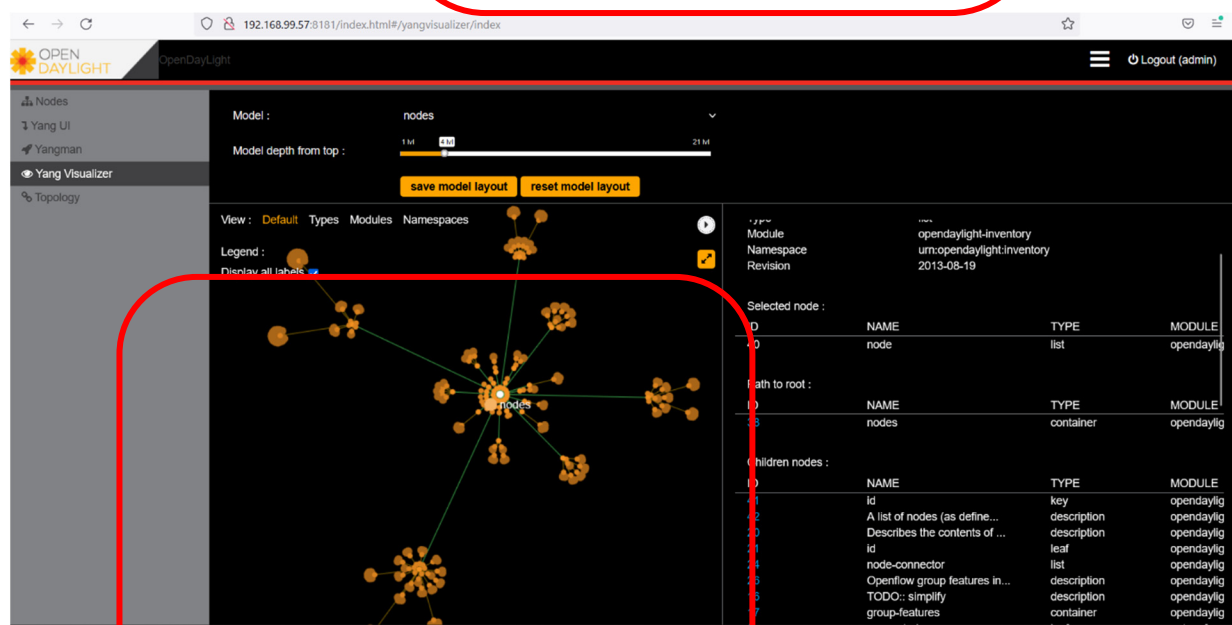


Figure 25. Visualization of network nodes through the ODL controller

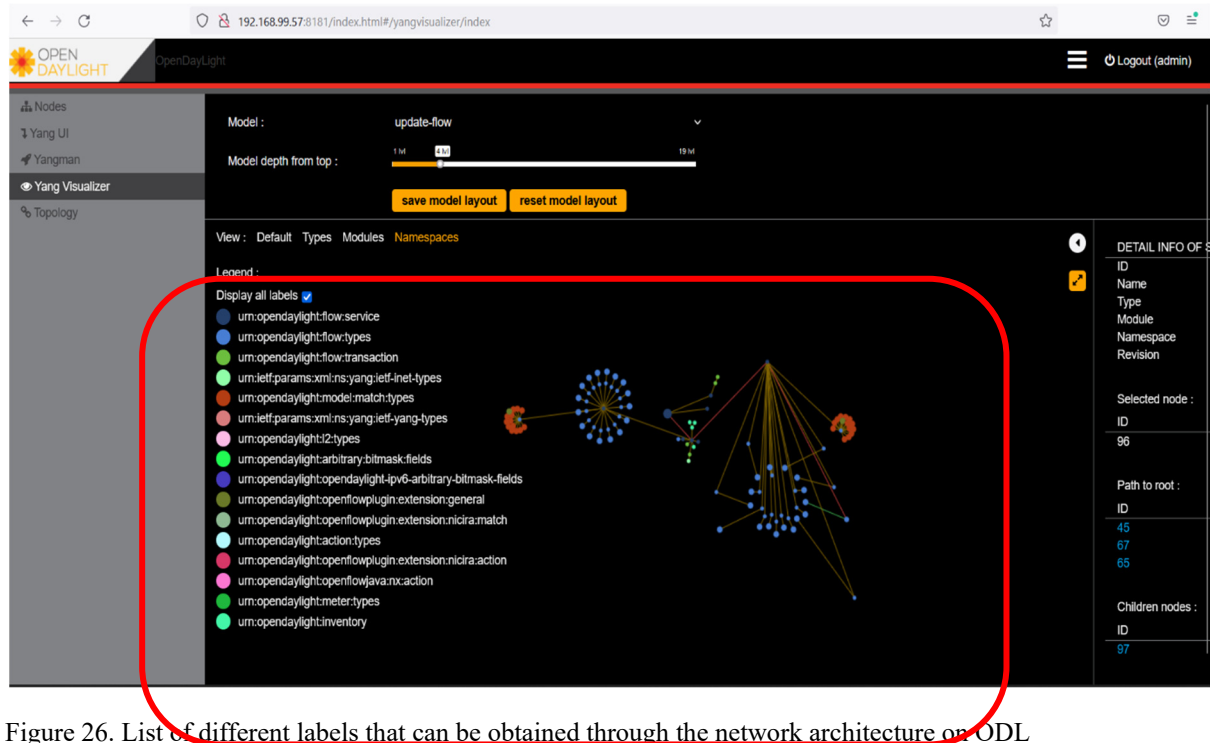


Figure 26. List of different labels that can be obtained through the network architecture on ODL

On the ODL controller, it is possible to program the different modules and to modify the source code of each of the modules to improve their functionality. This is the development component that allows you to customize or create the modules that will be used to further control flows in the network. As part of our master research work, we have modified the call script of the architecture deployed in an emulator in an SDN: ODL controller.

Thanks to the ODL controller, it is then easy to manage all the flows passing through the IP/MPLS cloud of our Autonomous Port network. We will see through Figure 27 an example of an interface on which a script can be written to create or customize the module.

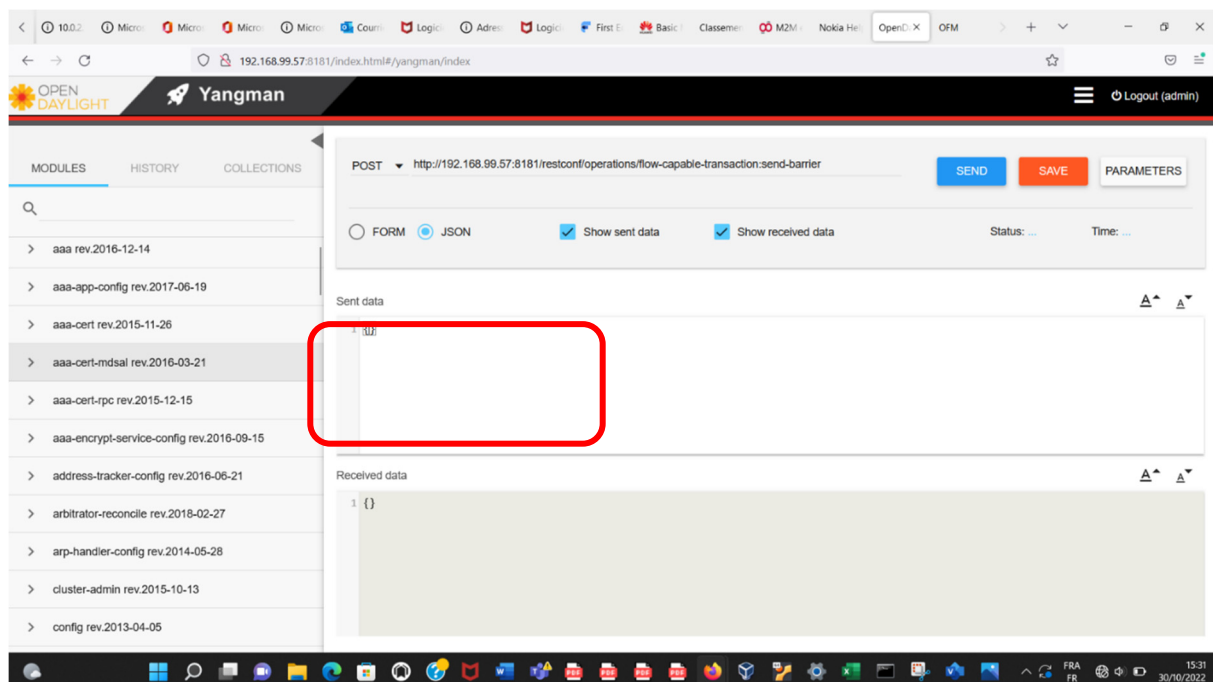
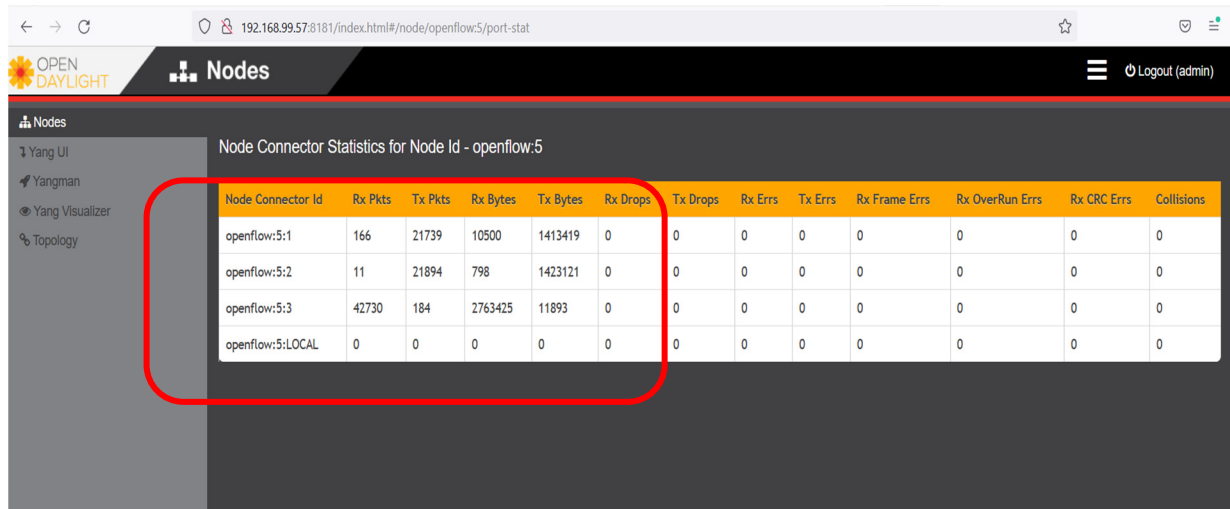


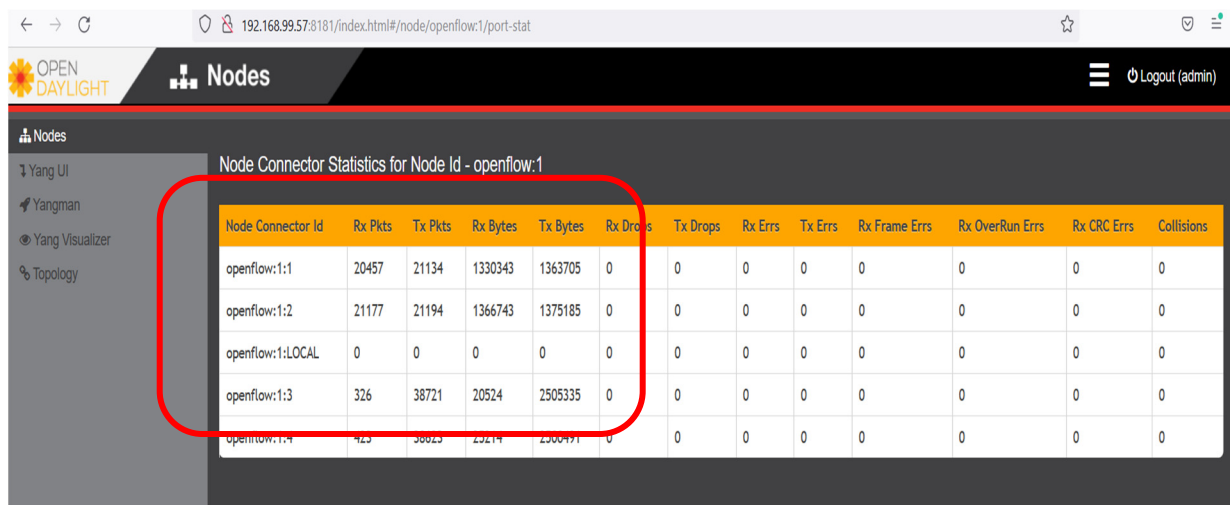
Figure 27. Programming interface of a module in the SDN ODL controller

In figure 28 we have the statistical table of the flow of OVS5; we note that the bytes received are of the order of 10,500 whereas in figure 29 the statistical table of the flow of the OVS1 presents us the bytes received of the order of 1,330,343. This difference would come from the function occupied by each of these OVS in the heart of the network. On the one hand, the OVS5 plays the role of PE router, a router that does not receive enough traffic compared to the OVS1 which plays the role of P router, through which all the core network flows pass. It therefore appears this great difference in the statistical results of these two OVS.



Node Connector Id	Rx Pkts	Tx Pkts	Rx Bytes	Tx Bytes	Rx Drops	Tx Drops	Rx Errs	Tx Errs	Rx Frame Errs	Rx OverRun Errs	Rx CRC Errs	Collisions
openflow:5:1	166	21739	10500	1413419	0	0	0	0	0	0	0	0
openflow:5:2	11	21894	798	1423121	0	0	0	0	0	0	0	0
openflow:5:3	42730	184	2763425	11893	0	0	0	0	0	0	0	0
openflow:5:LOCAL	0	0	0	0	0	0	0	0	0	0	0	0

Figure 28. OVS 5 flow statistics table obtained from the OF protocol



Node Connector Id	Rx Pkts	Tx Pkts	Rx Bytes	Tx Bytes	Rx Drops	Tx Drops	Rx Errs	Tx Errs	Rx Frame Errs	Rx OverRun Errs	Rx CRC Errs	Collisions
openflow:1:1	20457	21134	1330343	1363705	0	0	0	0	0	0	0	0
openflow:1:2	21177	21194	1366743	1375185	0	0	0	0	0	0	0	0
openflow:1:LOCAL	0	0	0	0	0	0	0	0	0	0	0	0
openflow:1:3	326	38721	20524	2505335	0	0	0	0	0	0	0	0
openflow:1:4	423	36923	23214	2300491	0	0	0	0	0	0	0	0

Figure 29. OVS1 flow statistics table obtained from OF protocol

Figures 30, 31 and 32 respectively present the Virtualized Network Core with an SDN controller in OpenFlow Manager, the OVS table of the IP/MPLS network core simulation architecture of an Autonomous Port and the OFM Interface presenting the OVS4 flows between the different core elements of the simulated IP/MPLS network

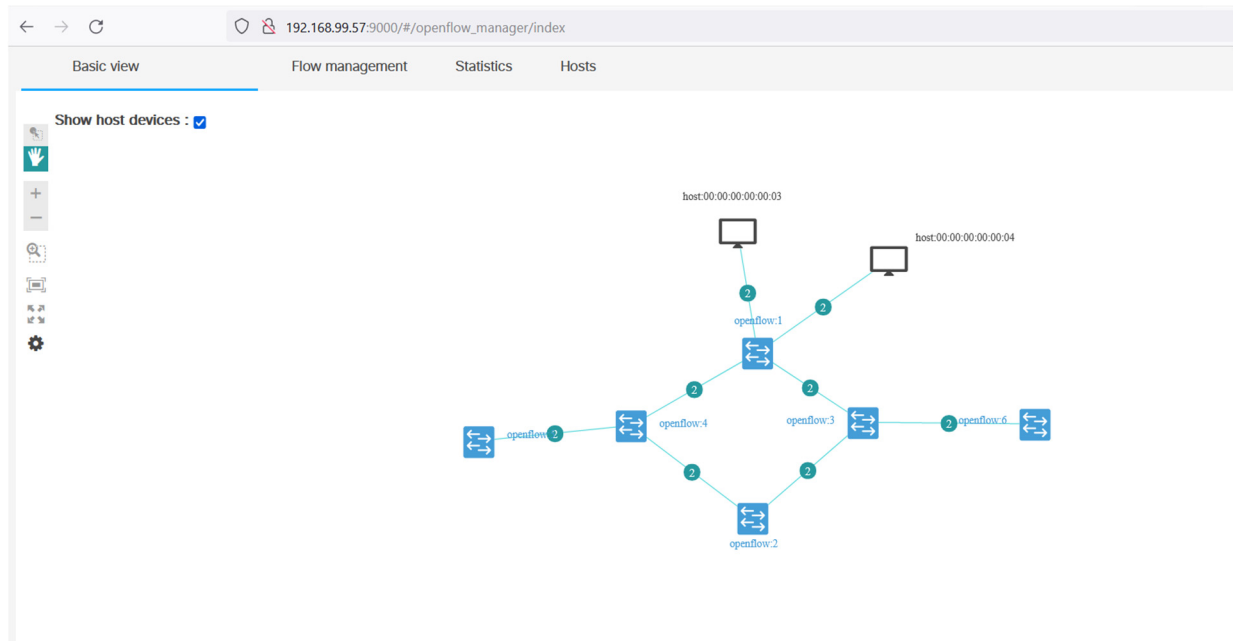


Figure 30. Virtualized network core with an SDN controller in OpenFlow Manager

Device	Device type	Device name	OF protocol version	Deployment mode	Pending flows	Configured flows
openflow:4	Open vSwitch	s4	of13	Not available	0	11
openflow:5	Open vSwitch	s5	of13	Not available	0	5
openflow:6	Open vSwitch	s6	of13	Not available	0	5
openflow:1	Open vSwitch	s1	of13	Not available	0	12
openflow:2	Open vSwitch	s2	of13	Not available	0	4
openflow:3	Open vSwitch	s3	of13	Not available	0	5

Figure 31. Table of OVS simulation architecture of the core IP/MPLS network of an Autonomous Port

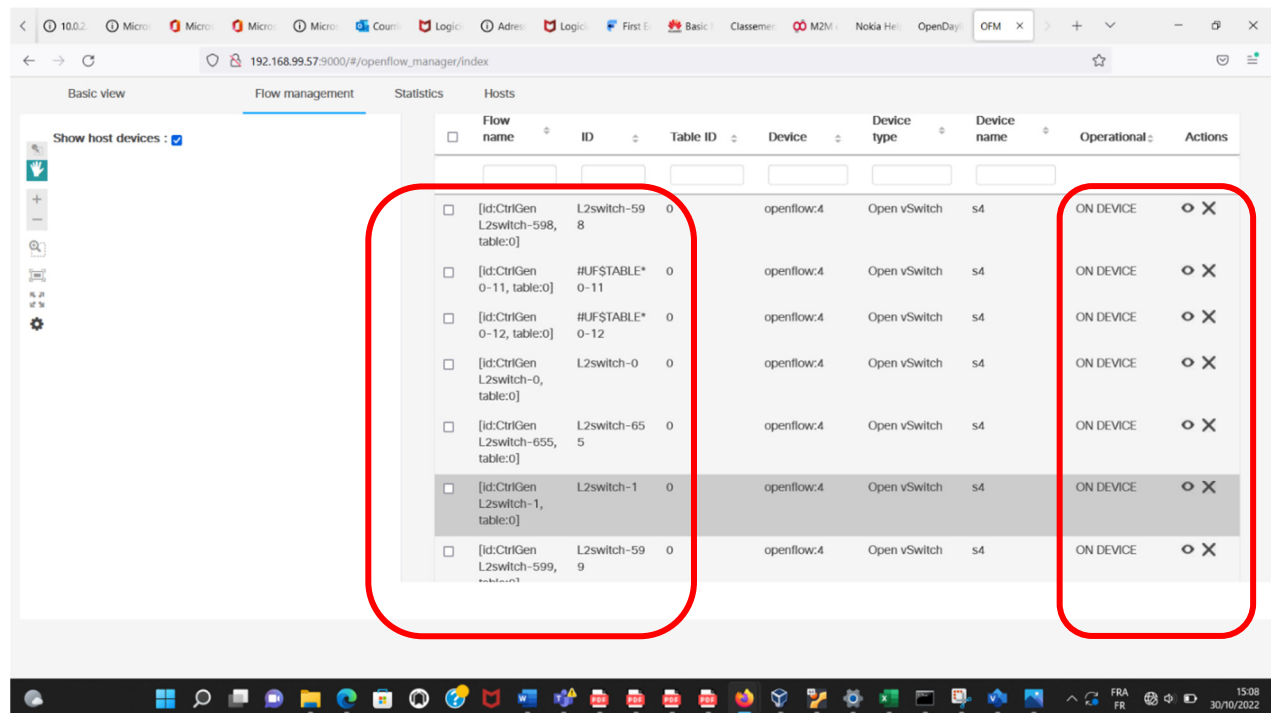


Figure 32. OFM interface showing the OVS4 flows between the different core elements of the simulated IP/MPLS network

Figure 33 presents the analysis of the average latency in our virtualized network

```

Host: h1"
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.059 ms
--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.059/0.195/0.639/0.222 ms
root@mininet-vm:/home/mininet# ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.297 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.044 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.049 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.047 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.047 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.055 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.060 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.072 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.045 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.046 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.057 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.061 ms
--- 10.0.0.2 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 10999ms
rtt min/avg/max/mdev = 0.044/0.073/0.297/0.068 ms
root@mininet-vm:/home/mininet#

```

Figure 33. Analysis of average latency in a virtualized network

In Figure 33 we observe at the level of the **time** parameter of the ping result that it is of the average order of **0.0732 ms < 0.08 ms**.

In a traditional network, latency results are around 10ms. By obtaining a result of less than 0.08 ms we guarantee

excellent fluidity through the virtualized network. The difference is around one hundred (100) times faster in a virtualized network than in a traditional network.

Figures 34 and 35 respectively present the OFM Interface presenting the OVS4 flow transmission statistics in **nanoseconds** between the different core elements of the simulated IP/MPLS network and the different statistics obtained through OFM

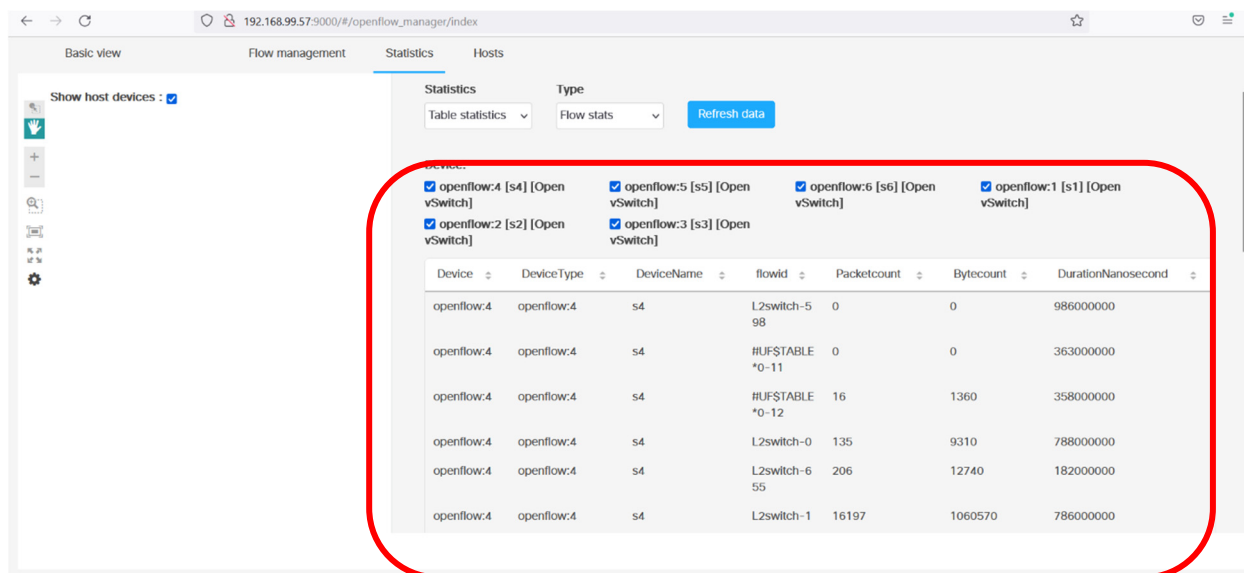


Figure 34. OFM interface showing OVS4 flow transmission statistics in nanoseconds between the different core elements of the simulated IP/MPLS network

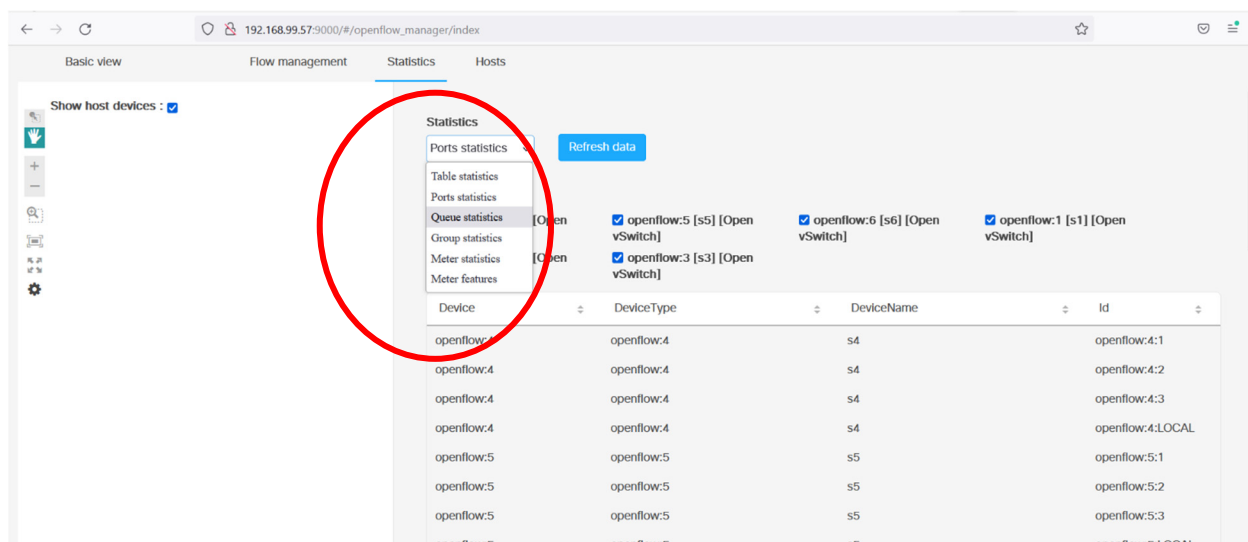


Figure 35. Different statistics obtained through OFM

For a particular stream, you can configure the MPLS parameters. Figure 36 shows the MPLS parameter configuration interface.

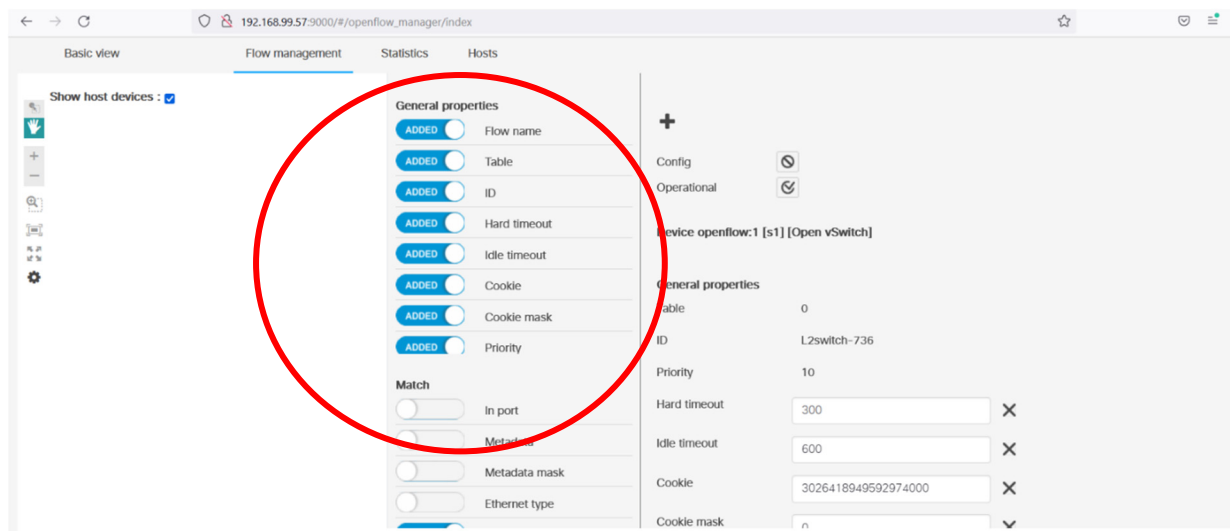


Figure 36. Presentation of the various general properties of a flow traversing the network

Figure 37 shows Configuration of stream MPLS parameters to optimize latency in the IP/MPLS core of the network

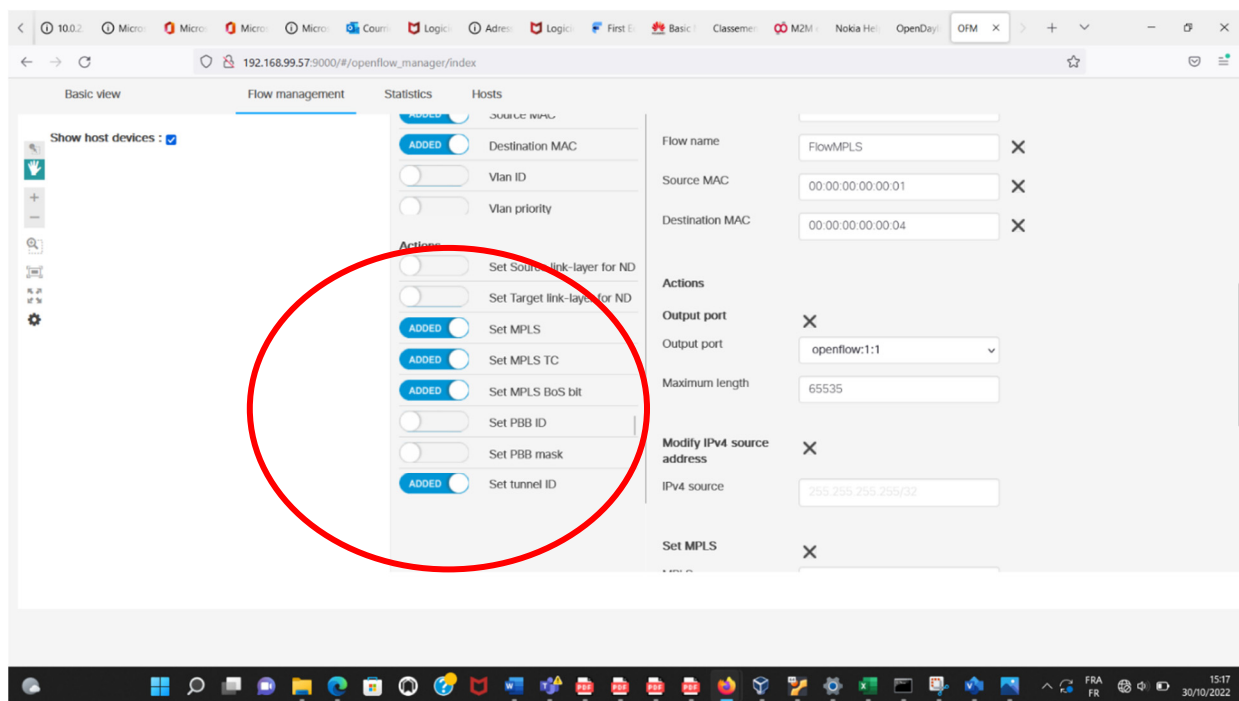


Figure 37. Configuring stream MPLS parameters to optimize latency in the IP/MPLS core of the network

#### 4.2.2. Securing data transmissions through the virtualized IP/MPLS cloud

Security in the transmission of data is very important in the life of the information system of an Autonomous Port. The data that transits in this network is very sensitive for the life of the company on the one hand and even for the State which is in charge of it. Figure 38 presents the five areas for setting up a virtualized infrastructure.

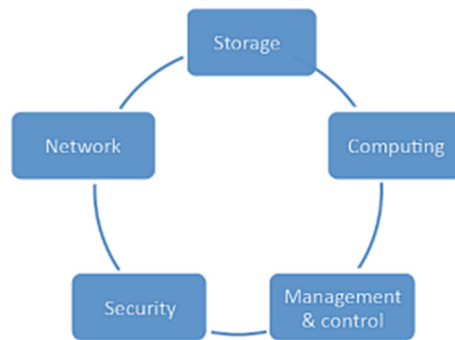


Figure 38. The five areas necessary for the life of virtualization

We will present the results of securing the heart of the IP/MPLS network of an Autonomous Port against DDOS attacks, and the redundancy of the SDN controller which would avoid having a "Single Point of Failure".

➤ Securing against DDOS attacks

We will focus our results on securing against network DDOS attacks. This attack is different from the application attack in that its main objective is to saturate the network connection of a server in order to no longer respond to requests, whereas the application targets a site whose pages take a long time to load. generate.

In the case of a virtualized network, it is a question of setting in the SDN controller. The SDN controller is the brain of the whole network and its security is particularly important. Flow analysis will be done through OFM for flood requests, and in addition the installation of an IDS would be important.

➤ SDN controller redundancy

The SDN controller redundancy architecture is shown in Figure 39.

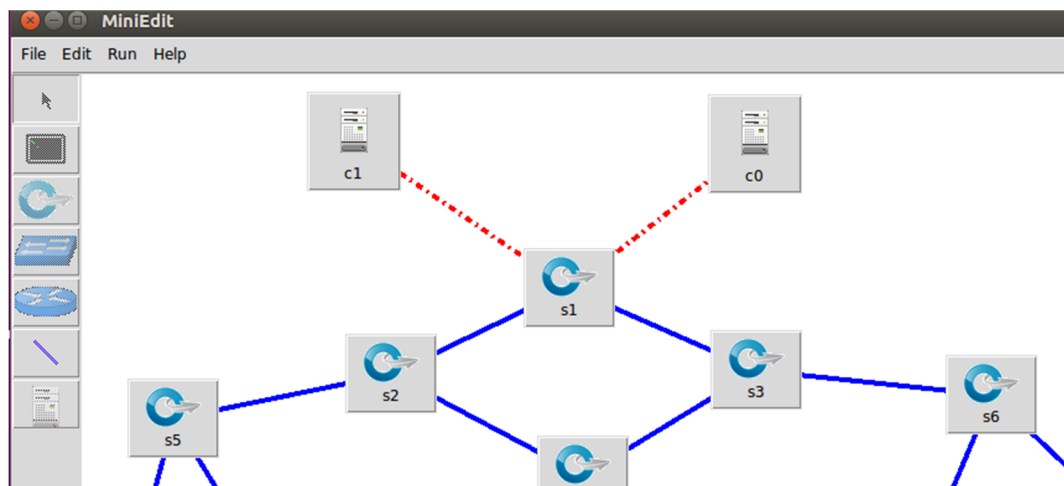


Figure 39. SDN controller redundancy to ensure IP/MPLS core network control security

This redundancy ensures flow control security in the network core. If the C0 Controller fails, the C1 Controller which is in redundancy, takes over automatically, while waiting for C0 to be repaired.

## 5. Conclusion and Perspectives

To perform their traffic flow routing functions, routers mobilize a signaling effort and a significant calculation, in particular for the discovery of neighboring equipment, the calculation of the optimal route, the inspection of the flow, the filtering, the sharing of load, etc. The large networks of 3rd and 4th generation Autonomous Ports should achieve optimal fluidity in the transmission of data and the management of the security of their information systems. The objective of this work was to demonstrate that it would be important to deal with the problem of QoS, thanks to virtualization, at the level of the heart of the network so that the latency in the propagation of data in the network

is more and more weak. The arrival of networks programmed by SDN software, which make it possible to virtualize the functions of the heart of the IP/MPLS network, modifies the landscape of traditional networks and advocates the separation of the control and data transfer planes. These programmable networks make it possible to relax or even overcome traditional constraints, which are less agile.

This programming by software is possible thanks to the dissociation of the data flow transmission/processing layer and the control/signaling layer which is in the same physical equipment. SDN makes it possible to manage the transfer plan from a logically centralized controller which is ODL in the context of our study. This technology can be implemented gradually, independently of specific equipment manufacturers. We go from very high latencies to sub-1ms latencies. The intelligence of the traditional network equipment being managed by the controller, there is the flexibility to distribute the loads according to the increase in load in the network, to avoid congestion.

The data transits in the network in a secure way, this makes it possible to deploy new services in a much faster way and with significantly reduced costs. This security is twofold, that of the controller who plays the role of orchestrator in the network and that of the communication between the agencies of the customers connected to the IP/MPLS cloud of the Autonomous Port, and even of the local IP network of the distributed Autonomous Port at multiple remote sites.

## References

- Amine, A. (2010). *Mise en œuvre d'un cœur de réseau IP/MPLS*. Université de bechar.
- Chachoua, F. (2018). The performance of Algerian ports: comparative study by the method of data envelopment analysis ((DEA), Thesis in obtaining a Doctorate in Management of Science, University of Abdelhamid Ibn Badis).
- de Barcelona, A. (2012). *The evolution of sea transport: 4th generation ports*, Barcelona Treb., Barcelona. Spain, Tech. Rep. Retrieved January 10, 2022, from <https://treball.barcelonactiva.cat>
- gSM Association. (2020). The mobile economy 2020. *GSMA HEAD OFFICE*. Retrieved January 10, 2022, from [https://www.gsma.com/mobileeconomy/wp-content/uploads/2020/03/gsma\\_mobileeconomy2020\\_global.pdf](https://www.gsma.com/mobileeconomy/wp-content/uploads/2020/03/gsma_mobileeconomy2020_global.pdf)
- <https://mahieufrancois.wordpress.com/veille-technologique/> (consulted on 01 10 2022)
- huawei.mpls\_overview. Retrieved from <https://support.huawei.com/enterprise/fr/doc/edoc1100116685/7c5ca4fb/basic-mpls-architecture>
- Jankowski, D., & Amanowicz, M. (2015). Intrusion detection in software defined networks with self-organized maps. *Journal of telecommunications and information technology*, (4), 3-9.
- Mahamat Charfadine, S., Flauzac, O., Nolot, F., Rabat, C., & Gonzalez, C. (2019). Secure exchanges activity in function of event detection with the SDN. In *e-Infrastructure and e-Services for Developing Countries: 10th EAI International Conference, AFRICOMM 2018, Dakar, Senegal, November 29-30, 2019, Proceedings 10* (pp. 315-324). Springer International Publishing.
- Oki, E., Rojas-Cessa, R., Tatipamula, M., & Vogt, C. (2012). *Advanced internet protocols, services, and applications*. John Wiley & Sons.
- Oloumane, E. (2019). *Optimization of the quality of service by the implementation of a programmable IP/MPLS network core and the virtualization of its functions: SDN and NFV technologies* (Dissertation in view of obtaining the diploma of Design Engineer in Telecommunications, National Advanced School of Engineering).
- Ottou, N. L. (2020). *Design and deployment of an SD-WAN solution: application to the management of CAMTEL company networks* (Dissertation in view of obtaining the diploma of Design Engineer in Telecommunications, National Advanced School of Engineering).
- Sanner, J. M. (2019). *Architecture du plan de contrôle SDN et placement de services réseaux dans les infrastructures des opérateurs* (Doctoral dissertation, Université de Rennes 1 [UR1]).

## Copyrights

Copyright for this article is retained by the author(s), with first publication rights granted to the journal.

This is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/4.0/>).