S3N - Smart Solution for Substation Networks, an architecture for the management of communication networks in power substations

ALEXÁNDER LEAL PIEDRAHITA

UNIVERSIDAD DE ANTIOQUIA
1803

UNIVERSIDAD
DE ANTIOQUIA
1 8 0 3

Faculty of Engineering

Doctoral Thesis

S3N - Smart Solution for Substation Networks, an architecture for the management of communication networks in power substations

Author:
Alexánder Leal

Supervisor:
Prof. Dr. Juan Felipe Botero

Thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electronic Engineering

November 2018

Supervisor
Prof. Dr. Juan Felipe Botero (University of Antioquia)

Possible Members of the Examination Committee
Prof. Dr. Adrian Lara (University of Costa Rica)
Prof. Dr. Oscar Mauricio Caicedo (University of Cauca)
Prof. Dr. Sandra Julieta Rueda (University of the Andes)
Prof. Dr. Javier Rubio Loyola (Center for Research and Advanced Studies of the National Polytechnic Institute of Mexico, CINVESTAV)
Prof. Dr. Xavier Hesselbach-Serra (Polytechnic University of Catalonia)

alexánder leal piedrahita : *S3N - Smart Solution for Substation Networks*, an architecture for the management of communication networks in power substations. © November 2018
Medellín - Colombia.

Family ...
an special group of people who love, respect and support each other.

— Valentina Leal

Dedicated to all my family, you know what I mean. Those people that, with or without blood-bond, are caring, supporting and loving you all the time, no matter what.

ABSTRACT

Today, the communications network has become an essential element to the operation of any type of organization or infrastructure, such is the case of the electrical power substations. Such networks in particular, demand high levels of availability and reliability, as the substation is a key element in the chain of energy generation and distribution. However, although recent network modernization introduced new features that allow optimizing the operation of the substation, the variety of devices present in such environment (Intelligent Electronic Devices (IEDs), Merging Units (MUs), Network Switches, IEEE 1588 Master Clock) and the huge set of application-level protocols (Sampled Measured Values (SV), Generic Object Oriented Substation Event (GOOSE), Manufacturing Message Specification protocol (MMS), Precision Time Protocol (PTP), among others), increase the management complexity.

Nevertheless, in recent years, data networks have been permeated by two major trends aiming to facilitate the administration of complex networks: Software Defined Networking (SDN) and virtualization technologies, which make the network management more flexible and enable the rapid development and deployment of network services.

This thesis proposes a set of contributions to solve the research challenges around of the current operation of a power substation communication network that have not been tackled by the research community. To do that, it performs a comprehensive review of the appropriation of SDN as an enabler in the management and operation of the power substations communication networks. The first research challenge we identified in this work is that, to the best of our knowledge, there are not research works proposing a complete architecture for the management of the communications networks of the power substation; also existing works do not introduce the virtualization technologies as an enabler in this environment. They only present how the application of SDN concepts may improve the performance of different communication tasks in power substations. This thesis introduces a novel architecture called Smart Solution for Substation Networks (S_3N), which presents a different way to represent the interaction among all elements involved in the operation of the power substation, taking the communications network as the central point and the SDN paradigm as a key element of its formulation.

The second challenge found in this work is that there is no unique criterion to define the structure of the network topology since, in every power substation, the end user implements their own topologies or the topology suggested by a vendor. In this context, this thesis presents a methodology to specify and characterize a reliable topology that

guarantees fault-tolerance, according to the guidelines described in the architecture S3N. In addition, this thesis presents alternative SDN solutions for loops-based topologies in the proposed network topology, which would be technically unfeasible using common network protocols. These solutions include algorithms to solve problems related to the broadcast and multicast traffic management.

Also, we discovered that, although the communication networks of modern electrical substations provide major benefits, various research articles have evidenced several vulnerabilities related to the operation protocols in this critical infrastructure. This thesis, in order to improve the security, presents two strategies to detect intrusions and one SDN approach to mitigate attacks in the reconnaissance phase.

Finally, all these contributions would not be enough to guarantee a reliable operation without mechanisms to bring traffic differentiation and provisioning. This thesis makes the best out of the architecture proposed to deploy Quality of Service (QoS) inside power substation communication networks, under the SDN paradigm.

PUBLICATIONS

Brevity in writing is the best insurance for its perusal

– *Rudolf* Virchow

Some ideas and figures presented in this document are based on the following own publications:

- Erwin Alexander Leal and Juan Felipe Botero. "S3N-Smart Solution for Substation Networks, an architecture for the management of communication networks in power substations." In: *Lecture Notes in Computer Science* 9701 (2016), pp. 52–56. ISSN: 0302-9743. DOI: 10.1007/978-3-319-39814-3_5
  Journal Quartile in Scopus: Q2.

- Erwin Alexander Leal and Juan Felipe Botero. "Transforming communication networks in power substations through SDN." In: *IEEE Latin America Transactions* 14.10 (2016), pp. 4409–4415. ISSN: 1548-0992. DOI: 10.1109/TLA.2016.7786323
  Journal Quartile in Scopus: Q2.

- Erwin Alexander Leal and Juan Felipe Botero. "Software defined power substations: An architecture for network communications and its control plane." In: *Communications (LATINCOM), 2016 8th IEEE Latin-American Conference on*. IEEE. 2016, pp. 1–6. ISBN: 978-1-5090-5137-3. DOI: 10.1109/LATINCOM.2016.7811573

- Carlos Mario Duran, Erwin Alexander Leal, and Juan Felipe Botero. "Improving fault tolerance in critical networks through OpenFlow." In: *Communications and Computing (COLCOM), 2017 IEEE Colombian Conference on*. IEEE. 2017, pp. 1–6. ISBN: 978-1-5386-1060-2. DOI: 10.1109/ColComCon.2017.8088195

- Erwin Alexander Leal, Juan Felipe Botero, and Eduardo Jacob. "Improving early attack detection in networks with sFlow and SDN." In: *Communications in Computer and Information Science* 916 (2018), pp. 323–335. ISSN: 1865-0929. DOI: 10.1007/978-3-030-00353-1_29
  Journal Quartile in Scopus: Q3.

- Erwin Alexander Leal, Juan Felipe Botero, and Eduardo Jacob. "QoS Proposal for IEC 61850 traffic under an SDN environment." In: *Communications (LATINCOM), 2018 10th IEEE Latin-American Conference on*. IEEE. 2018, pp. 1–6. ISBN: 978-1-5386-6754-5. DOI: 10.1109/LATINCOM.2018.8613240

- Erwin Alexander Leal. "Hierarchical clustering for anomalous traffic conditions detection in power substations." In: *IEEE Latin America Transactions* (2018). ISSN: 1548-0992. Submitted
  Journal Quartile in Scopus: Q2.

- Erwin Alexander Leal and Juan Felipe Botero. "Defining a reliable network topology in Software-defined power substations." In: *IEEE Access* 7.1 (2019), pp. 1–17. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2893114
  Journal Quartile in Scopus: Q1.

- Erwin Alexander Leal and Juan Felipe Botero. "A novel architecture for power substations communications networks based on SDN and virtualization paradigms." In: *IEEE Communications Magazine* (2019). ISSN: 0163-6804. Submitted
  Journal Quartile in Scopus: Q1.

We are witnessing huge changes in the field of communications networks. This dissertation is my way of becoming part of the transformation.

— Alexánder Leal

ACKNOWLEDGMENTS

This work would not have been possible without the support of several people and organizations.

First and foremost, I would like to thank my supervisor, Dr. Juan Felipe Botero for supporting me throughout the entire Ph.D. process –even before the beginning– and giving me the freedom to work on the topic that I wanted to explore. Without his encouragement, advice, patience and knowledge this challenging work would not have been possible.

A thank you also goes to Prof. Luis Alejandro Fletscher for his motivation to start this process and his tips during this time.

I would not like to miss the opportunity to thank Prof. Eduardo Jacob for enabling my six-month internship at the I2T Research Group of the University of the Basque Country, Euskal Herriko Unibertsitatea, in 2018.

A Ph.D. requires not just academic stuff, hard work, longer and more stressful days or writing a lot. Family is an important part of this journey process. I was lucky to have the support from my parents, brother, daughter and wife. Thanks for being part of this experience, for understanding my position, and being there when I needed you. I know that there were many sacrifices. One special thanks to my wife for providing a warm home and comfortable environment ;).

And last but certainly not least, a significant thank you to the University of Antioquia fo giving me the opportunity of continuing with my professional formation through its Studies Commission and the Administrative Department of Science, Technology and Innovation (Colciencias) for the scholarship that supported the development of this thesis.

Finally, thank you to all of the members of my jury for being part of this process. Your interest and support are much valued.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

LIST OF ALGORITHMS

ACRONYMS

ACSI Abstract Communication Service Interface

ADL Architecture Description Language

API Application Programming Interface

ARP Adress Resolution Protocol

ASDU Application Service Data Unit

BFS Breadth First Search

CoS Class of Service

CT Current Transformer

DFR Digital Fault Recorder

DHCP Dynamic Host Configuration Protocol

DNS Domain Name System

DoS Denial of Service

DDoS Distributed Denial of Service

ETH Ethernet

ETSI European Telecommunications Standards Institute

GOOSE Generic Object Oriented Substation Event

FTP File Transfer Protocol

HFSC Hierarchical Fair Service Curve

HMI Human Machine Interface

HSR High-availability Seamless Redundancy

HTB Hierarchical Token Bucket

HTTP HyperText Transfer Protocol

ICMP Internet Control Message Protocol

IED Intelligent Electronic Device

IETF Internet Engineering Task Force

IGMP Internet Group Management Protocol

ILP Integer Linear Programming

ITU International Telecomunication Union

IP Internet Protocol

JVM Java Virtual Machine

LAN Local Area Network

LLDP Link Layer Discovery Protocol

MAC Media Access Control

MMS Manufacturing Message Specification protocol

MU Merging Unit

NIC Network Interface Card

NIDS Network Intrusion Detection System

NFV Network Function Virtualization

ODL OpenDaylight

OFDP OpenFlow Discovery Protocol

OS Operating System

OVSDB Open Virtual Switch Data Base

PC Personal Computer

PRP Parallel Redundancy Protocol

PTOC Protection Time Overcurrent

PTP Precision Time Protocol

QoS Quality of Service

REST Representational State Transfer

RMON Remote Network MONitoring

RSTP Rapid Spanning Tree Protocol

S3N Smart Solution for Substation Networks

SAN Storage Area Network

SAS Substation Automation System

SCADA Supervisory Control And Data Acquisition

SCD Substation Configuration Description

SCL System Configuration description Language

SDN Software Defined Networking

SGAM Smart Grid Architecture Model

SMB Server Message Block

SNMP Simple Network Management Protocol

STP Spanning Tree Protocol

SV Sampled Measured Values

TCP Transmission Control Protocol

TRILL Transparent Interconnection of Lots of Links

TS Time Synchronization

UML Unified Modeling Language

UDP User Datagram Protocol

VLAN Virtual Local Area Network

VoIP Voice Over Internet Protocol

VPN Virtual Private Network

VM Virtual Machine

VT Voltage Transformer

WAN Wide Area Network

XCBR Circuit Breaker

Part I

PROBLEM STATEMENT AND STATE OF THE ART

Currently, the functions of processing, data network, and storage inside a power substation are performed by physically independent devices that are managed in a decentralized manner, which complicates their configuration/-maintenance similar to the existing conditions in the data centers of the mid-90s. In addition, this situation generates both inefficiencies and security vulnerabilities due to the bad use of computing and network resources in such critical infrastructure. Therefore, to simplify the system and eliminate existing inefficiencies, this proposal presents a system able to orchestrate computation and connectivity resources using the concepts proposed by Software Defined Networking (SDN) and virtualization technologies. Its aim is to improve the network management through the automatization of the tasks provisioning, which reduces the risks associated with cyber-attacks and human errors that can generate blackouts or brownouts.

1 INTRODUCTION

An investment in knowledge pays the best interest.
— Benjamin Franklin

Smart Grid is a concept that aims to provide mechanisms for the generation and consumption of energy in a more efficient and intelligent manner. This concept proposes the appropriation of data networks advantages to the grid operation in the area of control, communications and monitoring [Far10].

Smart Grid proposes the appropriation of data networks advantages to the grid operation.

To reach this purpose, the modernization of the infrastructure that supports the generation, transmission, distribution and consumption of power has caused the emergence, within the communications network, of a variety of IP compliant devices that are interconnected through a network based in Ethernet technology [KK13]. The automation process substation is oriented by the IEC 61850 standard [Tc5], which covers almost all aspects of Substation Automation System (SAS), providing new communication specifications to provide interoperability and flexibility in the operation of communication networks. However, this process of modernization has caused that the communication network management, in power substations, has become complex due to the large number of elements that comprise it (Intelligent Electronic Devices - IED: such as Breakers, switches, Protection and Control; Merging Units - MU, Network Switches, IEEE 1588 Master Clock), *where each device executes functions with different requirements of connectivity, delay, bandwidth provisioning, synchronization and security* [Hua+15].

Nevertheless, in recent years, data networks have been permeated by two major trends aiming to facilitate the administration of complex networks: SDN and virtualization technologies, which make the network management more flexible and enable the rapid development and deployment of network services [JP13]. SDN allows providing programmability to the network infrastructure, facilitating its management, while virtualization technologies allow, through software, to create a virtual version of a technological resource as an operating system, a storage device, a hardware component or a network resource, where and when it is needed [Kre+15; CB09; Mij+16]. But, introducing these technological enablers to the current power substation is not a straight-forward task; there are many research questions to answer. Are they adequate to manage a critical infrastructure?, Can they meet the current standards and requirements set by the sector (delay, reliability), provide scalability and allow provisioning?, Will they reduce investment and operation costs?, Are they safe solutions?.

Without mentioning that *a power substation is a critical infrastructure whose operation can significantly impact a highly energy-dependent society*.

Although solutions reached through SDN and virtualization technologies are transforming the way large datacenters, corporate networks and campuses have operated their networks; it is also seen how gradually these concepts can positively impact the performance and management of the increasing complexity of communications networks in power substations [Cah+13; Mol+15b; LFM14]. However, as proposed in [Bob+14], communication networks involved in the operation of the electric grid, must meet requirements that are not the same as the ones of a corporate network. Hence, several studies have evaluated the advantages and disadvantages of incorporating SDN and virtualization technologies in the current operation of a power substation communication network [PD14; Dor+14b; Don+15; KFK15; Dor+14a; Luw14].

The aim of this thesis is to study and evaluate current contributions and propose new strategies to improve the current operation of a power substation communication network in terms of management, reliability, service provisioning and security. In this context, we propose the development of Smart Solution for Substation Networks (S3N); a novel architecture to manage the communication networks in power substations, which use the principles proposed by SDN and virtualization technologies for its conception.

1.1 SCIENTIFIC CONTRIBUTION

Figure 1.1 gives an overview of the contributions generated out of this thesis. There are three main covered research topics around of the S3N proposal: the background, the modeling and the developing. Also, Figure 1.1 comprises annotations of the form $[x]_y$, denoting that scientific publication $[x]$ contributes to Chapter y.

The first part of this monograph, the *background*, is devoted to the presentation of the main concepts needed to understand both the problem this thesis faces and the set of existing proposals around this topic. This is a contribution of this thesis to the state of the art around the appropriation of SDN as an enabler in the management and operation of the power substations communication networks. The main result of this contribution is published in [LB16c]. This publication was matured using elements of the previous publication [LB16a].

The second part of this thesis, the *modeling*, details the contributions made in the conception of new elements that allow the improvement of the management and the reliability in power substations communications network. In [LB16b] a novel architecture, S3N, is proposed. This architecture presents a different way to represent the interaction between all elements involved in the operation of the power sub-

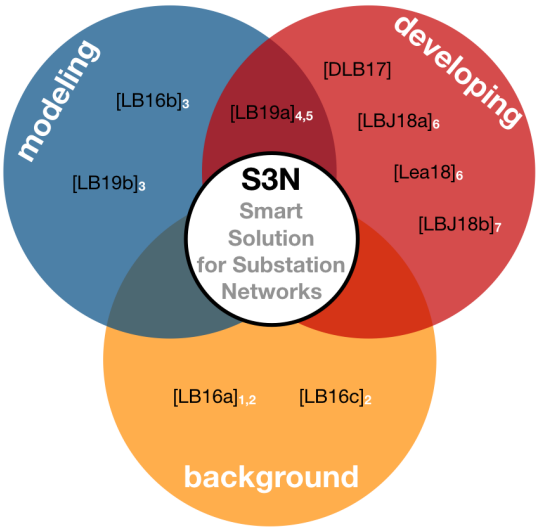


Figure 1.1: Contribution of this work according to the research studies conducted by the author

station, taking as the focal point the communications network and the SDN paradigm as key element of its formulation. Furthermore, a methodology to specify and characterize a reliable topology that guarantees fault-tolerance, according to the guidelines described in the architecture S3N, is suggested in [LB19b]. The outcome of this work gives a significant instrument to compare network topologies according to different criteria: terminal reliability, graph metrics, and end-to-end time-delay. In addition, [LB19a] complements the work presented in [LB16b] explaining in depth the elements that integrate the architecture S3N, as well as their interactions, using the Kruchten's 4 + 1 Model View.

The third part of this thesis presents the *implementation* of alternative solutions to shortcomings found in the existing mechanisms to manage power substations communications network. For example, we develop algorithms to solve complex issues related to loops-based topologies such as broadcast traffic control, path redundancy, packet redundancy, and multicast traffic management. Also, we evaluate two strategies to detect attacks in the reconnaissance phase: one using hierarchical clustering algorithms and statistical type descriptors (averages) and another one making use of the sampling and monitoring network tool, sFlow, to overcome scalability issues. Lastly, we investigate the challenges of service provisioning in power substations communications network, by proposing a complete SDN architecture for QoS provisioning. These important contributions are supported by these works [LBJ18a; LBJ18b; Lea18; LB19b].

Finally, the collaboration made in [DLB17] is a significant contribution to the understanding of fault-tolerant systems, employing the OpenFlow Select and Fast Failover groups. However, it was not included in the present monograph.

1.2 ORGANIZATION

The organization of this thesis is shown in Figure 1.2. There, the individual chapters, their main topics and interdependences are shown.

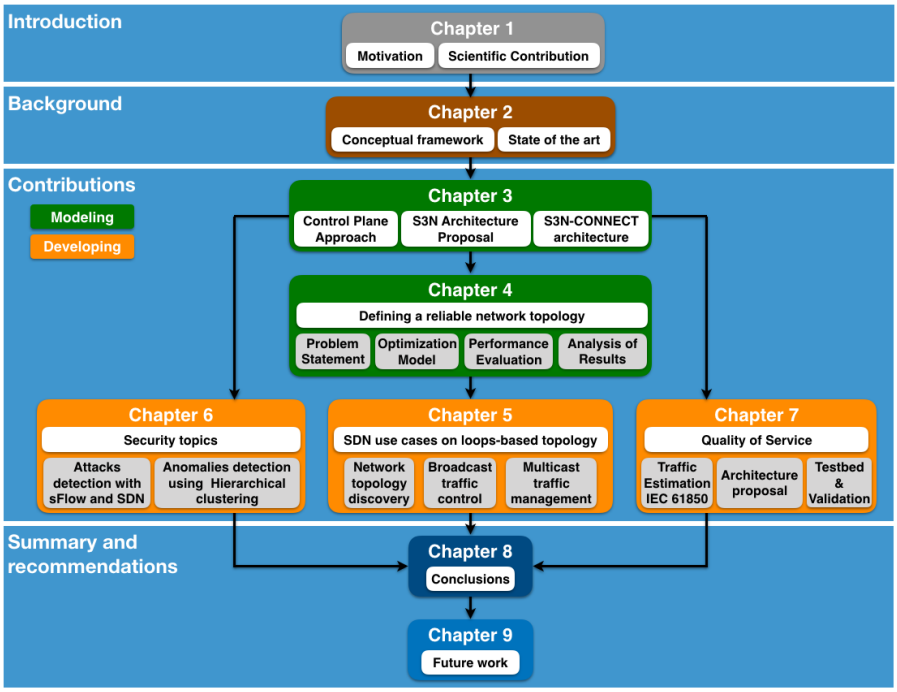


Figure 1.2: Thesis organization

Following this introductory part is Chapter 2, which presents the conceptual framework required to address this thesis and the literature review around the appropriation of the Software Defined Networking (SDN) as an enabler in the management and operation of the power substations communication networks.

The second part of this monograph presents a set of contributions oriented to determine the feasibility of improving network management in power substations. On the one hand, Chapter 3 and Chapter 4 propose a novel network architecture and the conception of a reliable topology, to reach the aforementioned proposal; while Chapter 5, Chapter 6 and Chapter 7 provide implementations to validate our approach.

Chapter 3 introduces a novel architecture, S3N, to model the power substations communications network using the concepts proposed by SDN and virtualization technologies for its formulation.

Chapter 4 proposes a methodology to specify and characterize a reliable topology that guarantees fault-tolerance, according to the guidelines described in the architecture S3N.

Chapter 5 exposes several SDN use cases about how to solve complex issues in loops-based topologies, such as broadcast traffic control, path redundancy, packet redundancy, and multicast traffic management.

Chapter 6 presents two approaches to detect attacks in the reconnaissance phase. The first one uses hierarchical clustering algorithms and statistical type descriptors (averages) for this purpose. While the second one makes use of the sampling and monitoring tool sFlow in an SDN environment. The latter approach delved thanks to the collaboration between Alexánder Leal and Prof. Dr. Juan Felipe Botero from the Telecommunications Engineering Department of the University of Antioquia with Eduardo Jacob from the Telecommunications Engineering Department of the University of the Basque Country.

Chapter 7 describes a proposal to provide QoS to critical infrastructure networks such as power substation communication networks. This contribution is a result of a collaboration between Alexánder Leal and Prof. Dr. Juan Felipe Botero from the Telecommunications Engineering Department of the University of Antioquia with Eduardo Jacob from the Telecommunications Engineering Department of the University of the Basque Country.

Chapter 8 presents the conclusions and main results of this thesis, while the Chapter 9 suggests several ideas, or improvements of the presented contributions, that can be subject of future work.

Finally, following these concluding chapters is the Appendix A with miscellaneous information to complement the S3N Architecture Views.

2 BACKGROUND

To know what you know and what you do not know, that is true knowledge.

— Confucius

This chapter presents the basic knowledge required to develop this thesis. The concepts of power substation communications networks, Software Defined Networking (SDN), Virtualization Technologies and Architecture Descriptions are explored here. In particular, the main contribution presented in this chapter is the state of the art around the appropriation of SDN as an enabler in the management and operation of the power substations communication networks (Section 2.2). This contribution was published in [LB16c].

2.1 CONCEPTUAL FRAMEWORK

This section summarizes the main notions around this thesis to give a better understanding of its reach and contributions.

2.1.1 *Automated Power Substation*

A power substation is a facility that belongs to the chain of generation and supply of electric power, whose tasks are the transformation and distribution of this type of energy. The concept of automated substation implies that the control, protection and monitoring processes in this infrastructure are automated. It means, they are governed by a set of intelligent devices with IP support (IED, MU, actuators, etc), interconnected through a communications network based on Ethernet technology (see Figure 2.1) [KK13].

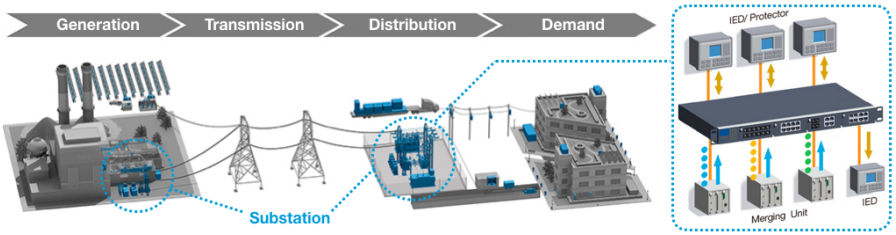


Figure 2.1: Location of power substations in the chain of generation and supply of electric power

2.1.2 *Communication networks in power substations*

In general, we can define a communication network of an automated power substation, as a set of devices with IP support, exchanging information via an Ethernet network that uses switches as interconnecting elements. This network is set up in order to ensure a communication platform supporting management, monitoring, synchronization, protection, control and sensing operations, within power substations.

Currently, the substation automation process is guided by the IEC 61850 standard [Tc5], which covers almost all aspects of a Substation Automation System (SAS). This standard provides recommendations to guarantee interoperability of devices from different manufacturers. Also, the standard defines how management, control and protection, and measurement devices intercommunicate inside a substation. As it can be seen in Figure 2.2, the model proposed by the standard IEC 61850 is hierarchical, where three levels are identified: station, bay and process; interconnected via the process bus and the station bus.

IEC 61850 is a global standard to guarantee that devices and tools from different vendors work together!

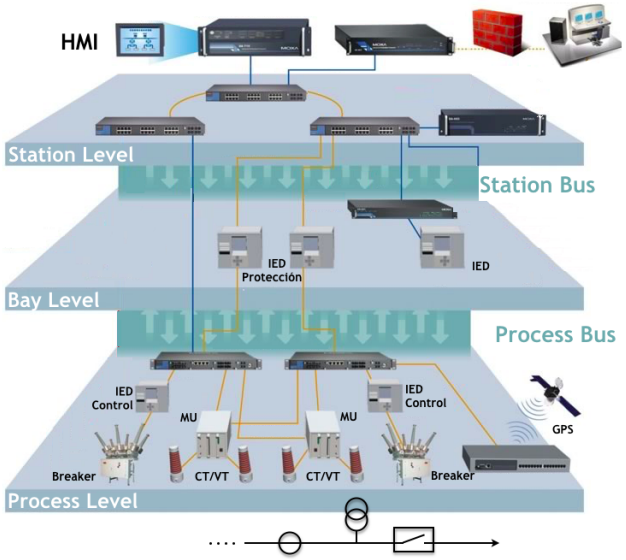


Figure 2.2: Communication model for the IEC 61850 standard

The process level is composed of actuators, measuring devices called MUs, Ethernet switches and yard equipments such as: Current Transformers (CTs), Voltage Transformers (VTs) and breakers. In the bay level, we find protection and control IEDs, while at the station level Ethernet switches and communication network management devices are located.

The IEC 61850 standard also defines four types of communication services, in order to ensure the correct operation of the network (see Table 2.1) [KK13].

TYPE OF SERVICE	DESCRIPTION
Abstract Communication Service Interface (ACSI)	Defined in IEC 61850-7-2, addresses the basic requirements for the process of exchanging information. With this aim, the MMS is used to transport operational information for the management of the substation between the user interface system and the IEDs.
GOOSE	Defined in IEC 61850-8-1 for the purpose of distributing event data (commands, alarms, indications, trip messages), between IEDs across entire substation network.
SV	Specified in IEC 61850-9-2, is used to transmit analog values (current and voltage) from the MUs to the IEDs.
Time Synchronization (TS)	Uses the PTP for ensuring clock synchronization among devices of a distributed system.

Table 2.1: Communication services defined in the IEC 61850 standard

2.1.3 *Software Defined Networking*

SDN corresponds to a "novel" network architecture, whose beginnings date back to 2008 at the University of Stanford, place where the first experiments that exposed the concepts and ideas proposed by this new architecture were performed. However, SDN is only the current stage of a path that began with the vision of programmable networks.

The idea of *programmable networks* appears as a proposal to facilitate network evolution, avoiding the complex tasks related to network management. Network operators always have had to transform high-level administration policies into low-level configuration commands to guarantee the operation of the communications network, handling the troubles associated with the user interfaces related with different vendors and providers. Before SDN existed other proposals in the field of *programmable networks*: OPENSIG, SOFTNET, Active Networks (AN), 4D Project, NETCONF and Ethane [Nun+14].

SDN makes network management easier and enables the development and deployment of new network services thanks to the separation of the control and data planes.

SDN proposed, to the outdated and complex management scheme bound to the traditional networks, a new management paradigm. Essentially, SDN poses the separation of the control plane (abstraction responsible for defining how to handle traffic) and data plane (layer responsible of implementing the decisions taken by the control plane). See Figure 2.3.

In traditional networks, the functions of the control plane and data plane are integrated in the same device [Kre+15]. For example, on a router, the control plane corresponds to the routing algorithm used to define the best route for a packet on a network, while the data plane corresponds to the routing table that allows the router to know through which network interface it should forward a packet. Note that the entries in the routing table are defined by the routing algorithm, which in the traditional case is a distributed control mechanism that defines how network traffic is handled. However, the separation be-

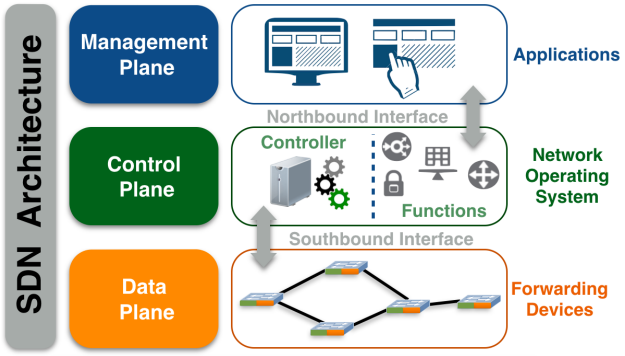


Figure 2.3: SDN Architecture

tween data plane and control plane implies that the control should be shifted to an external entity so-called controller (see Figure 2.4), which essentially provides the resources needed to program the forwarding devices. Basically, it is responsible for defining the rules that would act in forwarding devices, so that they know what to do whenever a packet arrives. Here, it should be mentioned that a traffic flow is a new SDN concept and it is defined as a sequence of packets transmitted through the network, sharing some criteria (e.g. MAC address, IP address, transport layer port, protocol type, etc.).

As a programmer can develop an application to be executed on a particular hardware (PC, Smartphone), SDN provides an architecture that enables the programmability of the entire communications network.

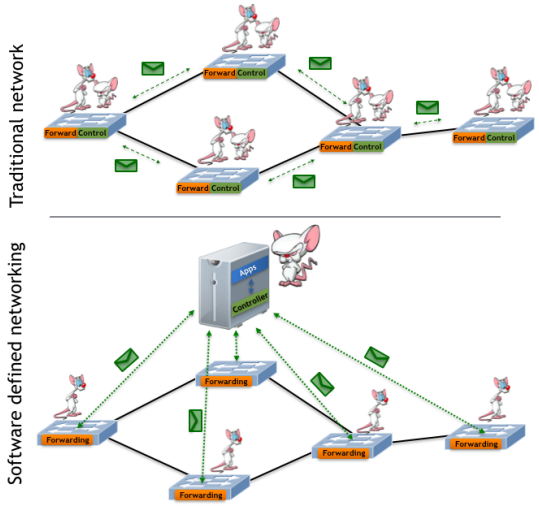


Figure 2.4: Comparison between a traditional network and an SDN

As shown in Figure 2.4, the SDN architecture presents a logically-centralized control of the network through the controller, an entity that simplifies the network administration, since it integrates into a single element: management functions, maintenance and network monitoring. However, the SDN architecture also requires a well-defined Application Programming Interface (API), so that the SDN controller (control plane) can exert a direct control over the forwarding equipment (data plane). This interface is so-called southbound interface. Currently, the most widely API has been OpenFlow [McK+08], al-

though others have also been defined such as ForcES [Dor+10]. In Figure 2.3, we can also observe the northbound interface, an interface that offers a common programming abstraction to the upper layers, mainly to the network applications such as load balancer, security systems and traffic engineering.

Although SDN started as an academic project, today it is one of the trends of greater impact on the industry. Currently, most manufacturers of network equipment such as switches, include at least one family of SDN devices. Even such, important organizations as Google, Facebook, Yahoo, Microsoft, Verizon and Deutsche Telekom, founded the Open Networking Foundation [ONF11], an organization dedicated to the promotion and adoption of SDN through the development of open standards, and which today comprises more than 500 companies worldwide.

2.1.4 *Virtualization Technologies*

Virtualization Technologies allow to create a virtual version of a technological resource as an operating system, a storage device, a hardware component or a network resource, on demand [CB09]. At present, the virtualization concept is applied to a wide range of infrastructures supporting information technologies. Table 2.2 shows virtualization techniques in different environments [Bar13].

I don't doubt at all that virtualization is useful in some areas. What I doubt rather strongly is that it will ever have the kind of impact that the people involved in virtualization want it to have.

– *Linus* Torvalds

HARDWARE	SOFTWARE	NETWORK	STORAGE	LANGUAGE	DESKTOP
Full Partial Para	OS level Application	VLAN VPN NFV	Block File	Java Virtual Machine	Virtual Hosted

Table 2.2: Types of Virtualization Techniques

- *Hardware virtualization* is a virtualization technique that simulates the underlying hardware using a hypervisor to provide each Virtual Machine (VM) with all the services of the physical system.
- *Software virtualization* allows a software to run in an environment for which it was not designed natively in isolation. For example, running a windows application on a Linux operating system through an emulator. Besides, the virtualization allows to execute several instances of the same Operating System (OS) in parallel.
- *Network virtualization* can be approached from different fields of application. For example: the creation of logical sub-interfaces within a physical interface to implement tunnels such as Virtual Private Network (VPN), the creation of multiple virtual networks supported on the same physical network infrastructure as Virtual

Local Area Networks (VLANs) or the deployment of network functions on commodity hardware instead of expensive dedicated hardware devices, Network Function Virtualization (NFV).

- *Storage virtualization* is a common component of Storage Area Network (SAN) system where the physical storage is completely abstracted of the logical storage, allowing distributed file systems. Although we also find examples around us as the virtual units that we observe in the file explorer when viewing the contents of a partitioned hard disk.

- *Programming Language virtualization.* In this case, a VM is placed in a layer above the operating system, so that it can execute high-level programs written and compiled under its operating conditions. In this way a program written in a high level language can be executed in any operating system that has the appropriate VM.

2.1.5 *Architecture Description*

According to the International Standard *ISO/IEC/IEEE 42010 System and software engineering - Architecture description* [ISO11], the conceptualization of a system's architecture, as expressed in an architecture description, assists the understanding of the system's essence and key properties pertaining to its behaviour, composition and evolution, which in turn affect concerns such as the feasibility, utility and maintainability of the system.

Architecture descriptions are used by the parties that create, utilize and manage modern systems to improve communication and co-operation, enabling them to work in an integrated, coherent fashion. Architecture frameworks and architecture description languages are being created as assets that codify the conventions and common practices of architecting and the description of architectures within different communities and domains of application. Figure 2.5 depicts concepts pertaining to the practice of architecture description when applying this International Standard to produce one architecture description expressing one architecture for one system-of-interest.

This International Standard provides a core ontology for the description of architectures, providing a common terminology and a conceptual basis that facilitates the specification of requirements, the definition, communication and revision of architectures from the use of an Architecture Description Language (ADL).

2.1.5.1 *Terms and definitions*

- *Architecting:* process of conceiving, defining, expressing, documenting, communicating, certifying proper implementation

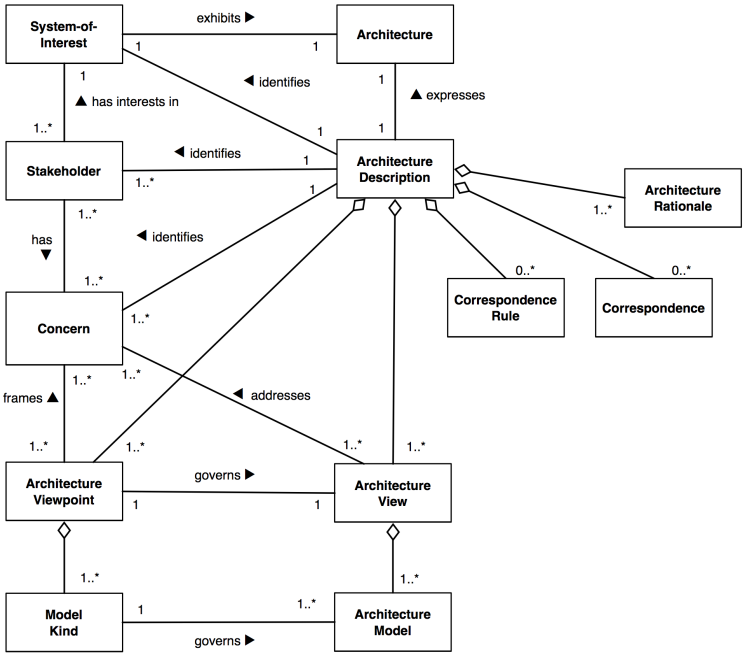


Figure 2.5: Conceptual model of an architecture description

of, maintaining and improving an architecture throughout a system's life cycle.

- *System of interest:* or simply system, refers to the system whose architecture is under consideration in the preparation of an architecture description.
- *Environment:* context determining the setting and circumstances of all influences upon a system.
- *Architecture:* fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution.
- *Stakeholder:* individual, team, organization, or classes thereof, having an interest in a system.
- *Concern:* interest in a system relevant to one or more of its stakeholders.
- *Architecture Description:* work product used to express an architecture.
- *Architecture Framework:* conventions, principles and practices for the description of architectures established within a specific domain of application and/or community of stakeholders.
- *Architecture View:* work product expressing the architecture of a system from the perspective of specific system concerns.

a viewpoint is a way of looking at a system (such as conventions, notations) ... a view is what you see!

- *Architecture Viewpoint:* work product establishing the conventions for the construction, interpretation and use of architecture views to frame specific system concerns.
- *Model kind:* conventions for a type of modelling.
- *Architecture Model:* uses modelling conventions appropriate to the concerns to be addressed.

2.2 STATE OF THE ART

This section highlights the research proposals addressed to the study of the application of SDN in the management and operation of communication networks, within the power substations.

2.2.1 *General Categorization*

In this part, a general categorization is proposed according to the approach of the presented works: proof of concept approaches, evaluation of technological requirements and solutions offered by the industry.

2.2.1.1 *Proof of concept approaches*

Studies addressing proof of concept proposals to evaluate the viability of applying SDN concepts in communication networks of power substations have appeared recently.

The first reference dates from 2013 at the University of Colorado [Cah+13]. In this paper, the authors argue that Energy Communications Networks (ECN) require an underlying technology which allows: (i) rapid innovation that enables the evolution of both the specialization within each infrastructure as well as integration between different infrastructures; and (ii) reduction of management complexity and verification of the correct network operation. As a proposal of such technology, the authors implemented SDECN, an experimental SDN-based system that introduced novel features such as network discovery, self configuration and network monitoring in the control plane.

In 2014, at the University of the Basque Country, a group of researchers presented [Mol+15b]. In this paper, the authors describe an architecture based on the principles SDN, but complying with the IEC 61850 standard. Among the modules proposed for the control plane, it is interesting to highlight: routing, traffic filtering, monitoring, quality of service, load balancing and security. To validate the capabilities of the proposed architecture, three use cases were presented: (i) implementation quality of service policies (QoS pushing), (ii) detection of denial of service (DoS) attacks and (iii) load balancing.

Also in 2014, researchers at the Federal Fluminense University developed the work entitled "SMARTFlow" [LFM14]. This proposal, just as [Cah+13] and [Mol+15b], uses the concepts related to SDN to pose a self-configuring system in order to manage the communications networks based on the IEC 61850 standard. The main features of the control plane SMARTFlow include: (i) automatic discovery of network topology (interconnection between switches), and identification of edge devices (IEDs); (ii) automatic configuration of groups by calculating Layer 2 multicast trees, for GOOSE and SV services; and (iii) reconfiguration of all flow entries in case of network failures.

In 2016, at the University of Shanghai Jiao Tong, [Li+16] present a new autonomous scheme for the dynamic allocation of bandwidth, oriented to the IEC 61850 communication standard and supported in SDN. The main contribution of this work is to model the SDN controller as a new IED. With this approach in mind, for an adequate formulation of bandwidth allocation policies, the new IED will monitor the network at the station and process levels and will receive the bandwidth requirements of the other protection and control IEDs. The results of this new approach significantly improve bandwidth utilization for MMS, GOOSE and SV communication services.

2.2.1.2 *Evaluation of technological requirements*

On the other hand, recent works have also been found that analyze in detail the advantages and disadvantages of incorporating SDN concepts into communication networks in substations, also proposing evidence of use cases to validate their assumptions.

For example, in 2014, researchers from Salzburg Research in Austria [PD14] evaluated the impact of using SDN in the domain of energy communication networks implementing two use cases: (i) failure of a link and (ii) presence of traffic overload. In their research, the authors expose risks such as: vulnerability of the network when there is a failure in the single controller, poor control over the detection of conflicts between flow rules, and the possibility of exploiting disclosure attacks when the network operates in reactive mode (when a switch sends a packet to the controller, since it does not know the action to execute on the packet). However, they also provide references on how these risks were mitigated in other types of environments. In the same way, the article mentions advantages of incorporating SDN as: the simplification of the configuration and management of VLANs, the increase of the security inside the network, and the possibility of applying traffic engineering dynamically. The conclusion of the study, as a consequence of the achieved results, was not to recommend the use of SDN with OpenFlow in production environments.

In that same year (2014), researchers from the Institute of Communication Networks of the TU Dortmund University in Germany [Dor+14b], also tested the effect of using SDN to meet the specific

communication needs in power substations. For this purpose, they propose a testbed to evaluate different failure scenarios such as: (i) the disturbance of a link and (ii) the congestion management through reservation of bandwidth and implementation of dedicated links. The obtained results show the advantages of using SDN in comparison with traditional routing mechanisms and quality of service implementation techniques, providing a more reliable communication network, capable of handling complex failure scenarios.

In 2015, authors from the University of the Basque Country [Mol+15a], evaluated how to take advantage of the redundant links present in critical communication networks, according to the guidelines of the IEC61850 standard. These redundant links are usually used as a backup solution but remain under-utilized, that is, they are not used in normal network traffic conditions. In this context, the researchers applied load-balancing and multiple-trajectory techniques using SDN to improve communication performance in networks with partial mesh topologies. The results show lower latencies that met the defined transmission requirements in the IEC61850 standard.

To complement, in this opinion article [Don+15], the authors analyze the opportunities that SDN can bring to communications networks in substations to improve the capacity of response to failures in the field of security and their corresponding challenges. The discussion revolves around three questions: (i) how SDN can increase the ability to face malicious attacks in the power substation communication networks?, (ii) what are the risks that introduce SDN, in terms of security, and how can they be managed? and (iii) how can the proposed solutions be validated and evaluated?

2.2.1.3 *Solutions offered by the industry*

Likewise, some efforts from the governmental and private sphere have developed solutions to improve the communications network management in power substations. It underlines the Department of Energy of the United States (DoE) interest in alliance with the well-known manufacturer Schweitzer Engineering Laboratories (SEL), who developed a network switch for power substations with SDN support (SEL2470S) [Sch18b], and an SDN controller for power substations (SEL5056 Flow Controller) [Sch18a].
To conclude this review, Table 2.3 summarizes the contributions, limitations and testing platforms of each of the works addressed, and its corresponding categorization.

2.2.2 *Findings*

This section highlights the main findings around the presented works in the preceding section.

2.2.2.1 *Trend to emulate network traffic*

Of the articles reviewed, only [Cah+13] evaluated the performance of the experiment from traffic generated by real IEDs. All others used software tools such as: rapid61850 [Bla+13], Scapy [Bio11], CMT II [PF06], libIEC61850 [Zil16] and Iperf [Tir99] to emulate this traffic partially. This fact has the advantage that researchers have a wide variety of tools to emulate traffic in this type of networks, without resorting to the use of expensive IEDs. On the other hand, the fact of not having real devices can limit the scope of the study.

	Title	Contribution	Testbed	Limitations
Proof of concept approaches	Software-defined energy communication networks (SDECN): From substation automation to future smart grids [Cah+13] (US)	SDECN. System to facilitate the management of power substation communication networks through a novel system of discovery and autoconfiguration of the network with SDN. Modules: self-discovery and monitoring.	**Network** (Emulated switches in mininet [Tea12]), **IEDs** (SEL2411 and SEL351), **Controller** (Ryu [Ryu]), **Southbound interface** (Openflow [McK+08])	Emulated evaluation network. Controller based on Python, interpreted language whose performance is not optimal compared to other languages. Multicast traffic was treated as broadcast in its implementation.
	Using Software Defined Networking to manage and control IEC 61850 based systems [Mol+15b] (ES)	Network architecture for power substations based on the IEC 61850 standard, supported by SDN operating principles. Modules: ShortestPath routing, traffic filtering, monitoring, quality of service, load balancing and security (firewall, anomaly detection and spoofing control).	**Network** (Emulated switches in mininet [Tea12]), **IEDs** (emulated with rapid61850 [Bla+13]), **Controller** (Floodlight [Big12]), **Southbound interface** (Openflow [McK+08])	Both the IEDs and the evaluation network are emulated. Only GOOSE and SV traffic are mentioned. Connections between switches and IEDs must be known in advance, there is no self-discovery.
	SMARTFlow: Uma Proposta para a Autoconfiguração de Redes de Subestação IEC 61850 Baseada em OpenFlow [LFM14] (BR)	Self-configuring system for network management based on the IEC 61850 standard that reduces network load by up to 44 %, compared to traditional solutions. Modules: routing based on multicast trees, traffic prioritization, failure detection and loops (STP), and self-discovery.	**Network** (Emulated switches in mininet [Tea12]), **IEDs** (emulated with Scapy [Bio11]), **Controller** (POX [Pox]), **Southbound interface** (Openflow [McK+08])	Both the IEDs and the evaluation network are emulated. Only GOOSE messages are transmitted (no SV, MMS traffic). Controller based on Python, interpreted language whose performance is not optimal compared to other languages.

Table 2.3 – *Continued on next page*

Table 2.3 – *Continued from previous page*

	Title	Contribution	Testbed	Limitations
	SDN based dynamic and autonomous bandwidth allocation as ACSI services of IEC61850 communications in smart grid [Li+16] (CN)	Novel scheme for bandwidth allocation based on SDN and IEC 61850 standar. It guarantees an adequate management of the bandwidth of MMS, GOOSE and SV traffic. Modules: Mapping of network services to ACSI services, traffic classifier, quality of service.	**Network** (Emulated switches in mininet [Tea12]), **IEDs** (emulated like hosts in mininet [Tea12]), **Controller** (POX [Pox]), **Southbound interface** (Openflow [McK+08])	Both the IEDs and the evaluation network are emulated. GOOSE and SV traffic were simulated like ICMP packets. Controller based on Python, interpreted language whose performance is not optimal compared to other languages.
Evaluation of technological requirements	Evaluation of software defined networking for power systems [PD14] (AT)	SDN and OpenFlow offer important advantages. However, its use is not recommended in production environments according to the found results. It is expected that in the future SDN may be incorporated into the operation of the energy communication networks. Use cases: Link failure and traffic overload.	**Network** (Switches HP2920), **IEDs** (emulated with CMT II [PF06]), **Controller** (Opendaylight [Odl] and HP controller [HP13]), **Southbound interface** (Openflow [McK+08])	Only GOOSE messages and FTP traffic were emulated (no SV, MMS traffic). The cases of use presented used different controllers, which does not allow directly compare the results.
	Software-defined networking for Smart Grid communications: Applications, challenges and advantages [Dor+14b] (DE).	It demonstrates the benefits of incorporating SDN to the Smart-Grid communication networks, compared to the traditional mechanisms, providing a reliable communication network capable of handling complex failure scenarios. Use cases: link disruption and congestion management through bandwidth reservation and dedicated links.	**Network** (Emulated switches on PCs with 4 ports Ethernet NICs executing OpenVSwitch [Nic09]), **IEDs** (emulated with libIEC61850 [Zil16]), **Controller** (Beacon [Eri13]), **Southbound interface** (Openflow [McK+08])	Only MMS and messages are emulated (no GOOSE traffic). The Beacon controller is deprecated (its advanced version is Floodlight).
	Managing path diversity in Layer 2 critical networks by using OpenFlow [Mol+15a] (ES).	SDN solution to use several routes simultaneously, increasing the performance of the network, through the combination of load balancing and multiple trajectory techniques. As a validation mechanism, robust network topologies were implemented that meet the operation and recovery requirements defined by the IEC 61850 standard.	**Network** (Emulated switches on mininet [Tea12]), **Traffic** (emulated with Iperf [Tir99]), **Controller** (Opendaylight [Odl]), **Southbound interface** (Openflow [McK+08])	The SV, GOOSE and MMS traffic flows are treated like ICMP, UDP and TCP flows traffic; according to the definitions suggested by IEC 61850 standard for SV/GOOSE/MMS.

Table 2.3 – *Continued on next page*

Table 2.3 – *Continued from previous page*

	Title	Contribution	Testbed	Limitations
Industry solutions	Watchdog Project [Sch18b] and Software Defined Networking (SDN) Project [Sch18a] (US).	Alliance between a governmental entity (US Department of Energy, DoE) and a private entity (Schweitzer Engineering Laboratories, SEL), for developing solutions that improve the management of power substation communication networks with SDN.	**Software** (SDN Controller), **Hardware** (Switch SEL 2740S)	Proprietary Solution

Table 2.3: Summarized overview of SDN solutions in power substation communication networks

2.2.2.2 *Emulated network topologies*

Except for [PD14] and [Dor+14b] most of the studies used mininet [Tea12] as a platform to recreate the topology of the network. This phenomenon is a consequence of that, at the time of carrying out the studies, the researchers did not have the wide variety of solutions offered today in the SDN field. However, this also reflects how mininet [Tea12] consolidates itself as a platform for experimentation in the fields of teaching and research of the computer networks.

2.2.2.3 *Wide variety of SDN controllers*

All the papers presented in this survey evaluated their work using a variety of controllers: Ryu [Ryu], Floodlight [Big12], POX [Pox], Opendaylight [Odl], HP controller [HP13] and Beacon [Eri13]. This means, in the period that the studies were carried out, there is no evidence of a tendency towards the use of only one type of controller. This fact confirms the continuous interest in the development of elements that contribute to the deployment of SDN, particularly the field of controllers.

2.2.2.4 *Openflow, southbound interface by default*

All the studies in this review used OpenFlow [McK+08] as an interface to the data plane (Southbound Interface). This event is the result of that OpenFlow was the first interface used in the implementation of the SDN architecture and continues in constant update.

2.2.3 *Discussion*

The review reveals the wide interest that exists, both in the academic community and in the industry sector, in evaluating the benefits that SDN can offer to the field of power substation communication networks.

In addition, this chapter exposes the advantages offered by SDN to conceive, develop and implement solutions quick and inexpensive (through emulation instruments).

The results found were encouraging in most cases, and suggest that SDN can boost the management of this type of network satisfying their demanding operation requirements, and offering the possibility of developing solutions. However, it is necessary to implement testbeds that improve the limitations found in this review. In this way, it will be possible to estimate whether the requirements defined by the IEC 61850 standard are satisfied. So then, the experimentation on real network topologies, with traffic conditions that match the great variety of protocols that coexist in this type of environments (MMS, GOOSE, PTP and SV), and the use of more robust SDN controllers in production environments are required.

2.2.4 *Conclusion*

The wide variety of studies regarding the application of SDN around the power substation communication networks, and the obtained results show that SDN in the short term could become in a technological enabler for the operation and management of this type of networks. The review suggests that is feasible meeting the delay time requirements for the control and monitoring messages transmitted on the network, enabling scenarios with fast fault-tolerance and improving the security of the systems involved.

Part II

CONTRIBUTIONS TO THE ARCHITECTURE PROPOSAL

Generically, an architecture is the description of the set of components and the relationships between them. Simple enough. The trouble starts when you tack on an adjective: There are software architectures, hardware architectures, network architectures, system architectures, and enterprise architectures. People have their own preconceived notions and experiences about "architecture." A software architecture describes the layout of the software modules and the connections and relationships among them. A hardware architecture can describe how the hardware components are organized. However, both these definitions can apply to a single computer, a single information system, or a family of information systems. *Thus "architecture" can have a range of meanings, goals, and abstraction levels, depending on who's speaking.* [AKL99]

3 ARCHITECTURE PROPOSAL

Perfection (in design) is achieved not when there is nothing more to add, but rather when there is nothing more to take away.

— Antoine De Saint-Exupery

This chapter presents a proposal for the control plane of a power substation communications network enabled by SDN. This proposal takes as reference the functionalities presented in Section 2.2. Also, this chapter introduces a novel architecture, Smart Solution for Substation Networks (S3N), to model the power substations communications network of today and tomorrow using the concepts proposed by SDN and virtualization technologies for its conception. In addition, the chapter introduces a comprehensible description of the S3N-CONNECT module of the S3N architecture, module in charge of providing communication (physical or virtual) among all devices related to the operation of the power substation communications network. This contribution was published in [LB16b] and submitted to [LB19a].

3.1 CONTROL PLANE APPROACH, UNDER POWER SUBSTATION COMMUNICATION NETWORK CONTEXT

According to the aforementioned review presented in Section 2.2, several works have proposed the possibility of incorporating technology enablers such as SDN, in order to automate processes and improve the management and operation of the communications network in power substations. This section presents a proposal to the control plane of the SDN architecture, in terms of power substations' communications network requirements.

3.1.1 *Network functionalities categorization*

Eight network functionalities on the improvement of operation network in a power substation were identified in the proofs of concept presented in Section 2.2.1.1. These functions are categorized in Table 3.1 according to its frequency of occurrence.

- *Network topology discovery.* Corresponds to a functionality of high preference, where most of the reviewed research proposals use Link Layer Discovery Protocol (LLDP) to discover the network topology. In contrast, [Mol+15b] assumes a static configuration, based on prior knowledge of the network. Here, it is important to note that the network discovery mechanism runs periodically

PREFERENCE	CRITERIA	FUNCTIONALITY
HIGH	Functionalities were implemented in 3 or 4 research proposals	Topology discovery, Loops prevention, Management multicast traffic and Monitoring.
MEDIUM	Functionalities are only present in two studies	Failure detection and Load balancing.
LOW	Functionalities just appear in one work	QoS and Security.

Table 3.1: Network functionalities founded in the Control Plane

(about every 5 seconds). Therefore, considering that within a substation network, the topology does not change dynamically; the modification of this frequency parameter may increase controller's performance. In addition, according to [Pak+14], minor modifications to the standard discovery algorithm, can improve the controller load conditions up to 45%.

- *Multicast traffic management.* It is a module of high preference. To handle this kind of traffic, [Cah+13] and [Mol+15b] used VLANs, while [Bot+15] used slices. On the other hand, [LFM14] proposed automatic configuration groups by calculating multicast routing trees, reducing network load up to 44%. Another interesting alternative proposes an efficient handling of multicast traffic, by manipulating the fields of multicast addresses [ISC11].

- *Monitoring.* It is also a highly implemented module, that demonstrates the importance of collecting traffic statistics. According to [Mol+15b], sFlow (RFC 3176), presents major advantages over monitoring strategies as OpenFlow Counters, NetFlow and SNMP/RMON. sFlow was also used in [Bot+15], whereas in [Cah+13] a module of the controller was used.

- *Preventing switching loops.* For this purpose, [LFM14] and [Bot+15] used Spanning Tree Protocol (STP). Whereas [Mol+15b] used shortest path algorithms. STP limits network scalability, because only allows one active path between two nodes, and sometimes, it does not choose the shortest or fastest path. However, there are alternatives such as Rapid Spanning Tree Protocol (RSTP), Transparent Interconnection of Lots of Links (TRILL), or shortest path algorithms [CKM12]. Nevertheless, we believe that the introduction of SDN could allow the use of optimization techniques to find the best group of loop-free paths, since the SDN controller is fully aware of network topology.

- *Failure detection.* Despite the importance of this feature, it was only addressed in [LFM14] and [Bot+15], where notification strategies to show link failures were implemented.

- *Load balancing.* Only two papers, [Mol+15b] and [LFM14], incorporated this feature, adapting modules provided by the controller. In this area, [Mol+15a] proposed an interesting alternative where network efficiency is increased using the links that STP leaves inactive, to send redundant information on alternative paths.

- *Quality of service.* Only a project posed this functionality using a controller module [Mol+15b]. However, taking into account the wide variety of traffic transiting through the network (SV, GOOSE, MMS, PTP, among others), we consider that the implementation of discrimination strategies for critical and noncritical traffic is necessary.

- *Security.* Only in [Mol+15b] a firewall and an intrusion detection system were implemented. However, in critical infrastructures, the security of the communication network should be an essential functionality. Nevertheless, authors have proposed contributions in this field. In [YS15], a system to detect traffic anomalies was implemented using computational intelligence, and [HSL09] posed mechanisms to provide authentication and integrity to GOOSE and SV messages.

3.1.2 *Control Plane Proposal*

According to the network functionalities identified in the aforementioned review, three domains will be defined for the control plane: supervision domain, redundancy and availability domain, and traffic control domain. In addition, each domain is composed of a set of elements that are defined below (see Figure 3.1).

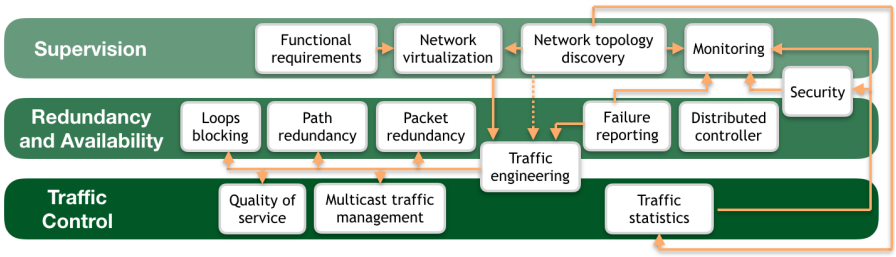


Figure 3.1: Approach to the control plane of the power substation communication network

- *Functional requirements.* This module collects the network requests, generated by the network substation operator. For example, what devices should be interconnected? with or without redundancy?, what quality services policies will be required?, among others.

- *Network topology discovery.* This module takes information, provided by network devices, to discover the physical topology of the network. When there are not network virtualization functionalities in place, this module provides information to the traffic engineering module directly.
- *Network virtualization.* This module consumes information from the network topology discovery and functional requirements modules. With this information, it defines the network resources and necessary information schemes, to build virtual networks in accordance with the defined requirements.
- *Monitoring.* This module collects information from four modules: network topology discovery, traffic statistics, failure reporting and security; in order to bring the network status to the substation operator.
- *Failure reporting.* This module is in charge of reporting, classifying and analyzing failures.
- *Distributed controller.* This module is responsible to guarantee the availability of the SDN controller in the event of failure.
- *Security.* This module consumes information from the network topology discovery module, in order to implement access control lists, through the registration of devices' MAC addresses. It also uses information from the statistics traffic module, in order to detect traffic anomalies.
- *Traffic engineering.* This module is in charge of the modules: loops blocking, path redundancy, packet redundancy, quality of service and multicast traffic management. It receives requests from the network virtualization module and defines how to handle them.
- *Loops blocking.* This module detects and prevents loops by means of programmable techniques, since the SDN controller is fully aware of network topology.
- *Path redundancy.* This module allows to discovery alternative paths, between two nodes, to send redundant information.
- *Packet redundancy.* This module implements reliability mechanisms to avoid packet loss.
- *Quality of service.* This module discriminates critical and non-critical traffic within a substation network to give preferential treatment for the services that need it.
- *Multicast traffic management.* This module provides mechanisms to guarantee the correct operation of GOOSE and SV services, reducing the flooding related to multicast traffic.

- *Traffic statistics.* This module collects and process traffic to bring useful information to the monitoring and security modules.

3.2 NOVEL ARCHITECTURE PROPOSAL FOR POWER SUBSTATIONS, S3N

In this section, we propose a reconceptualization of the power substations' communications network architecture using the concepts proposed by SDN and virtualization technologies. For the development of this stage, the suggested definitions in the International Standard *ISO/IEC/IEEE 42010 System and software engineering - Architecture description* [ISO11], (see Figure 2.5 in Section 2.1.5) are explained and contextualised on the following sub-sections.

3.2.1 *System of interest*

It refers to the system whose architecture is under consideration.

Power substations are a key element in the chain of energy generation and distribution and their communication networks demand high levels of availability and reliability, as well as a functional, secure, scalable and easy to administrate management platform. We are focused on the power substation communication networks, since they are an essential element in the operation of this type of critical infrastructures.

3.2.2 *Stakeholders*

They are those who have an interest in a system (individual, team, organization).

Below is a list of stakeholders interested in this architecture.

- Power substation operators.
- Power substation' Automation and Control engineers.
- Computer communications support engineers.
- Hardware and software infrastructure engineers.
- International standardization and regulation bodies.

3.2.3 *Concerns*

It is about the interest in a system relevant to one or more of its stakeholders.

Currently, the functions of processing, data network, and storage inside a substation are performed by physically independent devices that are managed in a decentralized manner, which complicates their configuration/maintenance similar to the existing conditions in the data centers of the mid-90s. In addition, this situation generates both inefficiencies and security vulnerabilities due to the bad use of computing and network resources in such critical infrastructure. Therefore, to simplify the system and eliminate existing inefficiencies, this proposal

presents a system able to orchestrate computation and connectivity resources using the concepts proposed by SDN and virtualization technologies. Its aim is to improve the network management through the automatization of the tasks provisioning, which reduces the risks associated with cyber-attacks and human errors that can generate blackouts or brownouts.

3.2.4 *Architecture ViewPoint*

It refers to the conventions for the construction, interpretation and use of architecture views within of specific domain.

Given the domain of the system of interest, we will take two models as reference. The first, our own conceptual model named Smart Solution for Substation Networks (S3N) and the second, the Smart Grid Architecture Model (SGAM), defined by the CEN-CENELEC-ETSI Smart Grid Coordination Group [CEN12].

3.2.4.1 *S3N - Architecture*

As shown in Figure 2.2, the communication model for IEC 61850 proposes a hierarchical structure, based on 3 levels interconnected by two data buses, where all devices that integrate the communications network can be grouped into five broad categories: peripherical communications, management, protection and control, measuring and interconnect devices. However, we believe that, the interaction between these categories of devices, could also be represented by a concentric model, or rings model, named Smart Solution for Substation Networks (S3N) as shown Figure 3.2.

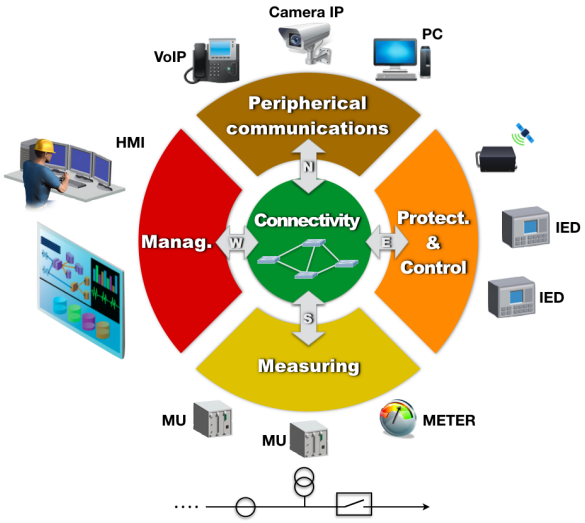


Figure 3.2: S3N architecture

In the S3N architecture, the central axis corresponds to all interconnection devices, mainly Ethernet switches and wiring. This component is called connectivity or S3N-CONNECT. S3N-CONNECT component provides communication among all the components that are directly

related with the operation of the communications network. That is, 1) protection and control devices (IEDs, actuators), grouped in the component named Protection and Control or S3N-PROTECT; 2) management devices (Events Recorder, Monitoring) which are grouped in the component named management or S3N-MANAGE; 3) measurement devices (MUs and meters), grouped in the component named measuring or S3N-MEASURE; and 4) devices for the support of other communications (PCs, VoIP, security cameras, thermal cameras, etc), grouped in the component named peripherical communications or S3N-PERCOM.

Another important element in this model refers to the communications buses: *North (N), South (S), East (E)* and *West (W)*. As illustrated in Figure 3.2, these communication buses will identify different types of traffic (communications services), according to their location:

- *North communications bus (N):* Refers to traffic *to/from* the peripherical communications component (S3N-PERCOM). In this domain we identify the traffic associated with communication services such as VoIP, video-call, video-surveillance (security cameras) or video-protection (infrared thermographic cameras for monitoring the temperature level of devices - overheating).
- *South communications bus (S):* Refers to traffic *to/from* the measurement component (S3N-MEASURE). In this domain we will identify the traffic associated with communication services such as SV, GOOSE and PTP.
- *East communications bus (E):* Refers to traffic *to/from* the protection and control component (S3N-PROTECT). In this domain we will identify the traffic associated with communication services such as SV, GOOSE and PTP.
- *West communications bus (W):* Refers to traffic *to/from* the management component (S3N-MANAGE). In this domain we will identify the traffic associated with communication services such as MMS.

Vertical traffic in the IEC61850 model corresponds to EW (East-West) *traffic, in the S3N model.*

The significant thing about this type of notation for communications is that the traffic that is exchanged between the S3N-MEASURE sector and the S3N-PROTECT can simply be called *SE (South-East)* traffic. Or for example, the traffic that goes from the S3N-PROTECT sector to S3N-MANAGE will be identified as traffic or communications *EW (East-West)*.

The model presented in Figure 3.2 is able to represent the interaction of the communications network in its current condition. Nevertheless, this model can be expanded through abstraction layers, with the aim of providing more flexibility to the proposed structure and conceiving the possible changes that may experience this communications

network in the future. Figure 3.3 illustrates an extended S3N capacity model, by incorporating an abstraction of three layers: infrastructure, virtualization and functionality.

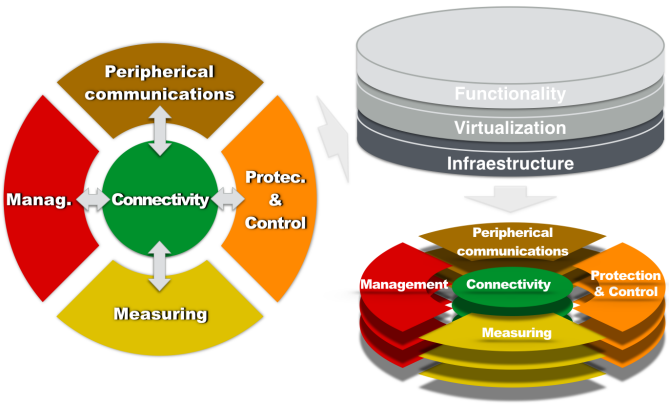


Figure 3.3: S3N architecture - Layer structure

- *Infrastructure layer.* It supplies physical computing resources, instrumentation, network and storage resources through the use of open hardware (with virtualization support).
- *Virtualization layer.* It provides a runtime environment for virtualized resources, reducing provisioning times and maintenance, and increasing system flexibility.
- *Functionality layer.* It corresponds to the domain of virtual applications (functions) that can be consumed by virtualized resources (virtual servers, virtual IEDs or a virtual network), located in the virtualization layer.

3.2.4.2 *SGAM Model*

The SGAM Framework aims at offering support to the design of smart grids use cases with an architectural approach allowing for a representation of interoperability viewpoints in a technology neutral manner [CEN12]. As illustrated in Figure 3.4, SGAM is a three dimensional model that is merging the dimension of five interoperability layers (Business, Function, Information, Communication and Component) with the two dimensions of the Smart Grid Plane: electrical domains and information management zones.

In the Smart Grid Plane the zones represent the hierarchical levels of power system management: Process, Field, Station, Operation, Enterprise and Market, while the domains cover the complete electrical energy conversion chain: Bulk Generation, Transmission, Distribution, DER and Customers Premises. Then, the five interoperability layers are described below.

- *Business Layer.* It allows to represent regulatory and economic (market) structures and policies, business models, and business portfolios (products & services) of market parties involved.

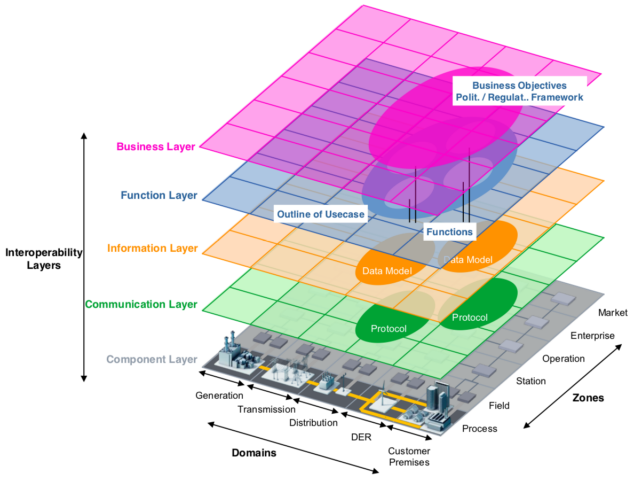


Figure 3.4: Smart Grid Architecture Model (SGAM)

- *Function Layer.* It describes functions and services including their relationships. The functions are represented independent from actors and physical implementations in applications, systems and components.
- *Information Layer.* It describes the information that is being used and exchanged between functions, services and components. It contains information objects and the underlying canonical data models.
- *Communication Layer.* It exposes the protocols and mechanisms used for the interoperable exchange of information between components in the context of the underlying use case, function or service and related information objects or data models.
- *Component Layer.* It allows to represent the physical distribution of all participating components in the smart grid context. This includes system actors, applications, power system equipment, protection and tele-control devices, network infrastructure (wired/wireless communication connections, routers, switches, servers) and any kind of computers.

3.2.5 *Architecture Views*

A view is a projection of a model, omitting entities that are irrelevant from a certain perspective or vantage point.

A *view* is a representation of a partial aspect of the *system of interest* through the use of the conventions established by the Architecture Viewpoint. Basically, the *views* allow observing a system from different perspectives for providing a complete description of it. In our case, the S3N model will provide a first conceptual view of a use case of the architecture proposal, while the SGAM model will help to delimit the scope of the S3N architecture and illustrate system components, their interconnections and the mechanisms used for the exchange of information.

3.2.5.1 *S3N - Use Case*

Figure 3.5 illustrates a use case where the rings model structured in layers represent a future power substations' communications network architecture.

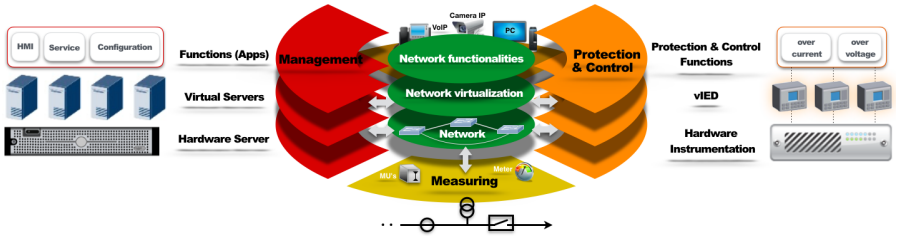


Figure 3.5: S3N - Use case

- *Peripherical Communications (S3N-PERCOM):* In this particular use case, this segment only has participation in infrastructure layer, just as it is today.
- *Protection and Control (S3N-PROTECT):* In this component, it is possible identify three layers. At the lower layer, corresponding to the infrastructure level, we could find a hardware instrumentation. In turn, on this instrumentation hardware is possible to instantiate virtual IEDs in the virtualization layer, which could consume protection and control functions stored in the functionality layer (over-current protection, over-voltage protection, among others).
- *Management (S3N-MANAGE):* Here, it is also possible to identify three layers. At the lower layer, we could find a hardware server that allows to instantiate virtual servers in the virtualization layer. Besides, these virtual servers could execute applications (recording, monitoring), using a software as a service model.
- *Measuring (S3N-MEASURE):* In this particular use case, this segment only has participation in the infrastructure layer, just as it is today.
- *Connectivity (S3N-CONNECT):* Similarly, in the core of the architecture, we could find, in the lower layer (infrastructure), all hardware necessary to ensure the interconnection of all devices (hardware switches and wiring). Then, on the layer infrastructure, virtual networks could be instantiated in virtualization layer, which can use functions from functionality layer for its operation (traffic filtering, loops preventing, among others). These network functions were precisely discussed in our approach to control plane Section 3.1.2.

3.2.5.2 *SGAM Views*

Figure 3.6 exposes the scope of the proposed architecture under SGAM context. Thus, S3N is defined in the zones of field, station and operation, of the distribution domain, through the layers of components, communication, function and information.

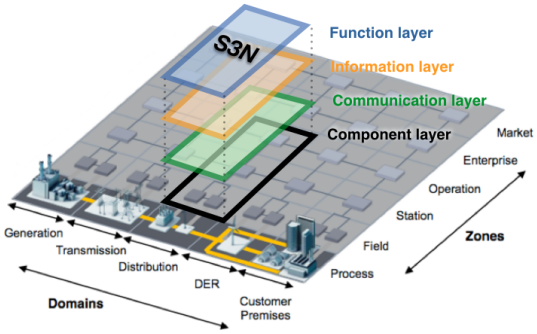


Figure 3.6: S3N scope under SGAM context

The *views* related with components, communication, information and function layers can be found in Section A.1.

3.3 CORE ARCHITECTURE DESCRIPTION, S3N-CONNECT

This section presents a comprehensible vision of the S3N-CONNECT module, the core of the S3N architecture. Its intention is to identify the elements that integrate it and explain their interactions using the Kruchten's 4 + 1 Model View [Kru95].

3.3.1 *System of interest*

S3N-CONNECT corresponds to the central axis of the S3N architecture. This module provides communication (physical or virtual), among all the sectors linked to the operation of the communication network: protection and control devices (S3N-PROTECT), management devices (S3N-MANAGE), measurement devices (S3N-MEASURE), and devices for the support of other communications and security (S3N-PERCOM). This module, as shown in Figure 3.7, will be governed by an SDN architecture.

3.3.2 *Stakeholders*

Below is a list of stakeholders interested in this module of the S3N architecture.

- Power substation operators.
- Power substation' Automation and Control engineers.

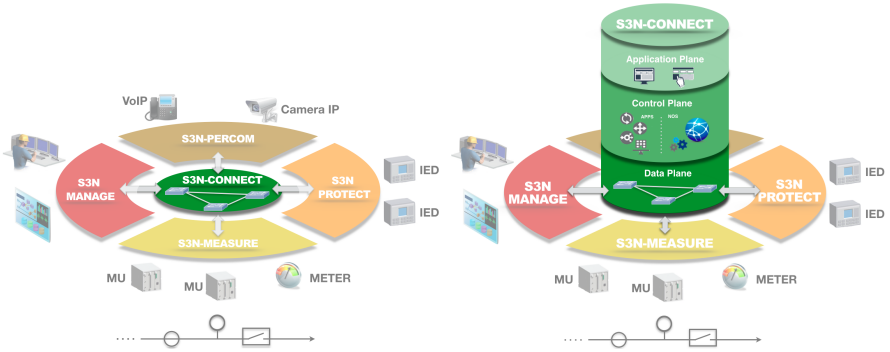


Figure 3.7: S3N-CONNECT module governed by an SDN architecture

- Computer communications support engineers.

3.3.3 *Concerns*

The communications network has become an essential element to the operation of any type of organization or infrastructure, such is the case of the electrical power substations. Such networks demand high levels of availability and reliability, as the substation is a key element in the chain of energy generation and distribution. However, although recent network modernization introduced new features that allow optimizing the operation of the substation, the variety of devices that integrate it (IEDs, MUs, Network Switches, IEEE 1588 Master Clock) and the huge set of application-level protocols (SV, GOOSE, MMS, PTP, among others), increase the management complexity. A power substation contains hundreds of devices generating and consuming critical information in real time, with different requirements of connectivity, delay, bandwidth provisioning, synchronization and security, according to their purpose or field of application. Without mentioning that the IEDs require adequate maintenance and configuration if you do not wish to compromise the efficiency, reliability, availability and safety of the substation's networks.

3.3.4 *Architecture ViewPoint*

Given the domain of the system of interest, we will take as a reference the Kruchten's 4 + 1 Model View [Kru95], a view model that conforms to the *ISO/IEC/IEEE 42010 System and software engineering - Architecture description* [ISO11].

3.3.4.1 *Kruchten's 4 + 1 Model View*

It is a model of multiple and concurrent *views* that is closely related to all stakeholders according to their role in the development of the project: end users, developers, systems and/or telecommunications engineers, integrators and project managers. The model allows the

description of a software architecture using 4 + 1 views, which allow describing in an isolated way the particular behavior of the system. As illustrated in Figure 3.8, the model proposes the following perspectives or views:

If it is not written down, it does not exist.

– Philippe KRUCHTEN

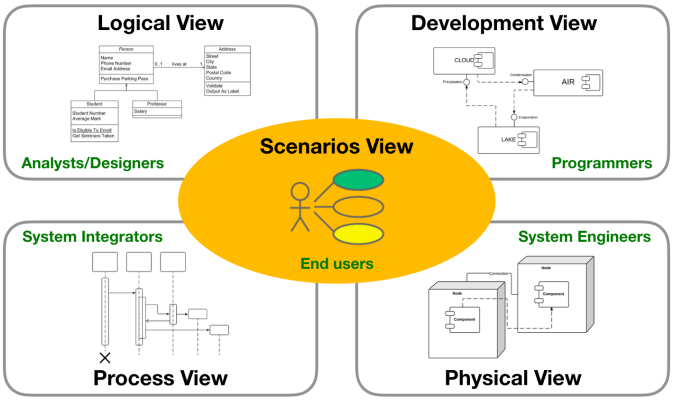


Figure 3.8: Kruchten's 4 + 1 Model View

- *Logical View:* This view represents the functionality that the system will provide to end users, it means, what the system must do and the functions and services that it offers. These functionalities are a response to the needs identified in the domain of the problem. The elements that support the functionality are modeled in this view.
  Notation: UML class, communication or sequence diagrams.

- *Process View:* In this view is represented the processes that exist in the system and the way how they communicate each other from the perspective of a systems integrator. It focuses on the behavior of the system at runtime.
  Notation: UML activity diagram.

- *Development View:* Also named implementation view. It shows the system from the perspective of a programmer and deals with software management. It will present how the software system is divided into components, the dependencies that exist between them and the organization of the software modules.
  Notation: UML component and packet diagrams.

- *Physical View:* Also known as deployment view. In this view, all the physical components of the system and the physical connections between those components (including the services), are shown from the perspective of a systems/telecommunications engineer. This view must identify the mechanisms and protocols used for the exchange of information between components.
  Notation: UML deployment diagram.

- *Scenarios View:* Also called as use cases view. This view consolidates the previous views from the perspective of the project

manager, where the scenarios become an abstraction of the most important requirements represented by the use cases.
Notation: UML use cases diagram.

It is important to mention that this model defines what is necessary to document in each *view*, and not the way to do it. For example, a logical *view* can be documented graphically with a UML class diagram, but it does not mean that this *view* has to be documented with that diagram, simply this diagram (by its characteristics) can document this *view*. Kruchten does not define the *viewpoints* to make *views*, it only defines the information that each *view* must contain. So, when the software architects use UML, they save the step of having to document the *viewpoints* because UML already has its representations widely documented. A summary of the UML diagrams used for the representation of the Kruchten 4 + 1 model is shown in Table 3.2.

UML is a standardized modeling language enabling developers to specify, visualize, construct and document artifacts of a software system.

4 + 1 MODEL VIEW	UML DIAGRAM
Scenarios	Use cases. It can be complemented with other diagrams such as: Activity, Interaction or State.
Logic	Class, State, Collaboration.
Development	Component, Packet.
Physical	Deployment.
Process	Activity, States, Sequence.

Table 3.2: UML diagrams used in the Kruchten 4 + 1 model

3.3.5 *Architecture Views*

In this case, the Kruchten's 4 + 1 Model View [Kru95] will provide the reference model to describe the way how the S3N architecture could be appropriated as a solution to the aforementioned concern (Section 3.3.3).

3.3.5.1 *Logical View*

The logical view includes the functional requirements that the system must provide in terms of service to its users (communications network operators within power substation). Given the complexity of the system, the logical view will be represented by three diagrams:

- Functional block diagram of the control plane proposed under SDN architecture.
- Functional block diagram around the frameworks used for development.
- Class diagram for the OpenDayLight controller framework.

FUNCTIONAL BLOCK DIAGRAM OF THE CONTROL PLANE PROPOSED In the S3N architecture, the S3N-CONNECT component is governed by a SDN environment. Next, in Figure 3.9, the three domains defined for the control plane (Section 3.1.2), along with the modules implemented for demonstrating the scope of our architecture proposal S3N, are illustrated.

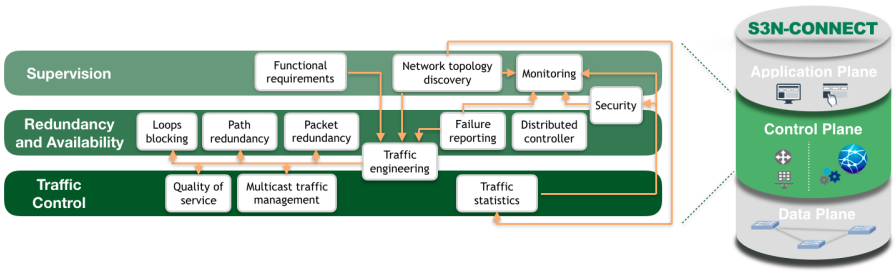


Figure 3.9: Functional block diagram of the S3N-CONNECT control plane

FUNCTIONAL BLOCK DIAGRAM AROUND THE FRAMEWORKS USED FOR DEVELOPMENT In our S3N architecture adoption, the S3N-CONNECT component makes use of different frameworks for the implementation of the functionalities described in Figure 3.9. Next, the Figure 3.10 presents the domains of the frameworks used: OpenDaylight (ODL), sFlow and NeXt UI. Note that some functionalities are articulated through the integration of two frameworks: *monitoring* (NeXt & sFlow) and *security* (ODL and sFlow).

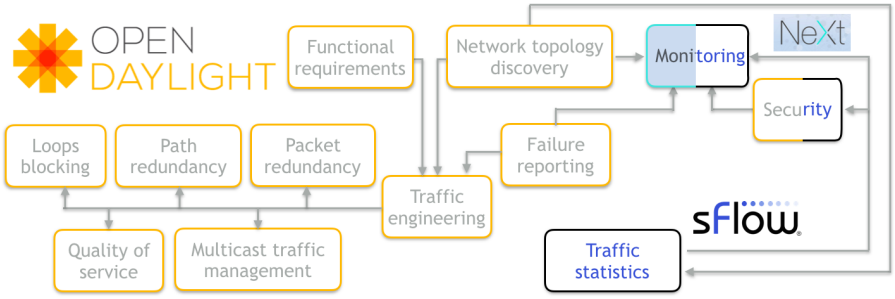


Figure 3.10: Frameworks used within S3N-CONNECT component

- *OpenDaylight*. ODL is a modular open platform for networks management under the SDN paradigm, focused on the programmability of the network. The ODL platform is designed to allow any user (researcher or supplier) to develop a network controller according to their needs. The ODL controller is implemented in Java and runs within its own Java Virtual Machine (JVM). This means it can be implemented on any hardware or platform whose operating system is compatible with Java [Odl].

- *sFlow.* sFlow is a packet sampling technology that can be implemented in a wide range of devices (switches, routers, servers or Personal Computers (PCs)), with the purpose of analyzing and monitoring traffic statistics or computing resources such as CPU and memory [PPM01]. sFlow provides REST & JavaScript APIs which allow parameterizing metrics and defining levels of alarm and/or notification, according to the monitoring requirements predefined by the network operator. The elements that integrate the sFlow architecture, as well as its operation scheme, are described in RFC 3176 [PMP01].

- *NeXt Toolkit.* NeXt UI is an HTML5/Javascript/CSS framework that can be used by user interface developers to integrate representations of interactive network topologies into their web applications. NeXt provides high performance and allows user interaction with the network topology through an event listener API [CIS18].

CLASS DIAGRAM FOR THE OPENDAYLIGHT CONTROLLER FRAMEWORK Figure 3.11 illustrate the packages developed under the OpenDaylight platform to satisfy the functionalities implemented for the system (see Figure 3.10). Likewise, Figure 3.12 shows how each package is broken down into a series of key abstractions, such as the classes and their corresponding relationships (See detailed information in Section A.2.2).

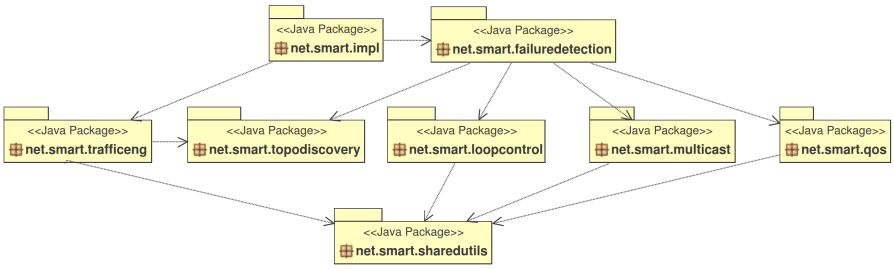


Figure 3.11: UML packet diagram of S3N-CONNECT

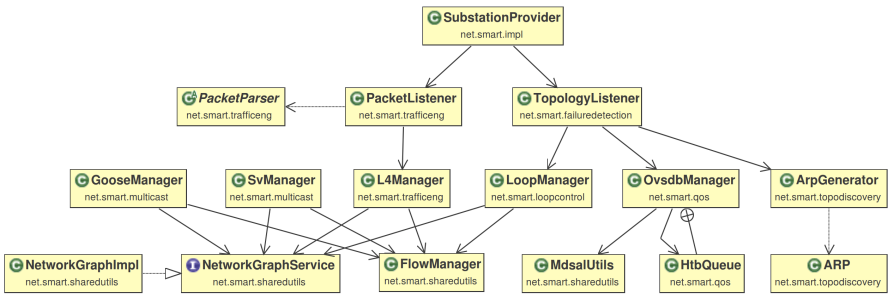


Figure 3.12: UML classes diagram of S3N-CONNECT

3.3.5.2 *Development View*

The development view, or implementation view, represents the static organization of the software modules built for the system. Figure 3.13 illustrates the software components of the system and their dependencies. Detailed information about the components and dependencies of the software components under the OpenDayLight domain can be found in Section A.2.3.

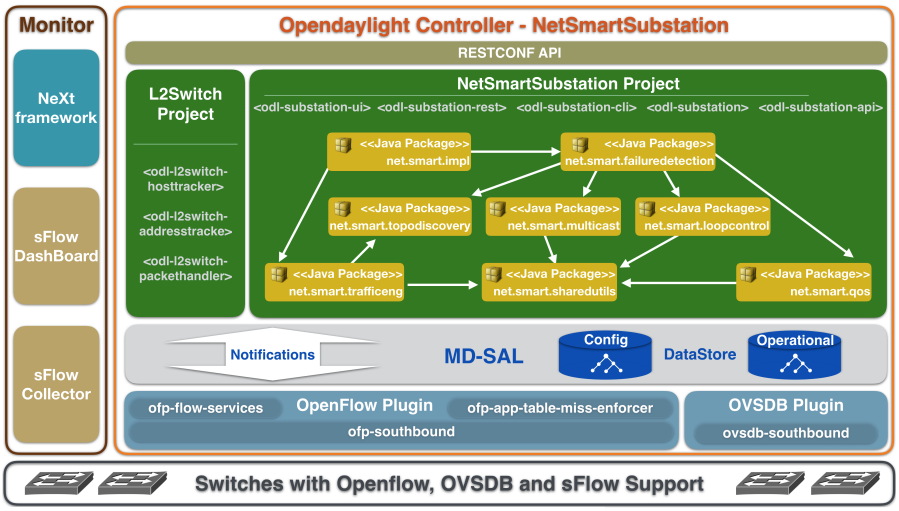


Figure 3.13: Components diagram for the S3N-CONNECT module.

3.3.5.3 *Process View*

The process view shows the interaction of the objects inside the system and/or with elements from the outside. Next, the operation schemes associated with the topology discovery process for switches and hosts; the Breadth First Search (BFS) tree calculation to mitigate the problems associated with the broadcast traffic in topologies with loops; and the multicast tree calculation for the SV traffic management are explained for the purpose of providing a better understanding of the sequence diagrams used to represent this view.

NETWORK TOPOLOGY DISCOVERY AND CONTROL LOOPS The network topology discovery functionality of the NetSmartSubstation project aims to gather information about the links and nodes (switches or hosts) that make up the topology of the network in the Operational DataStore. This function comprises two phases: core devices discovery (*switch-nodes*) and their corresponding links, and edge devices discovery (*host-nodes*) and their corresponding links. These tasks are carried out by executing two java plugins related to two native Opendaylight projects (L2Switch and OpenFlowPlugin), along with functionalities developed by us under the Opendaylight platform. It is important to

mention that the process of network topology discovery incorporates in its implementation the loops control process, hence the reason for the title of this section. Figure 3.14 shows the sequence diagram associated with the topology discovery and loop control operations and detailed information about the development of this functionality can be found in Section 5.2.

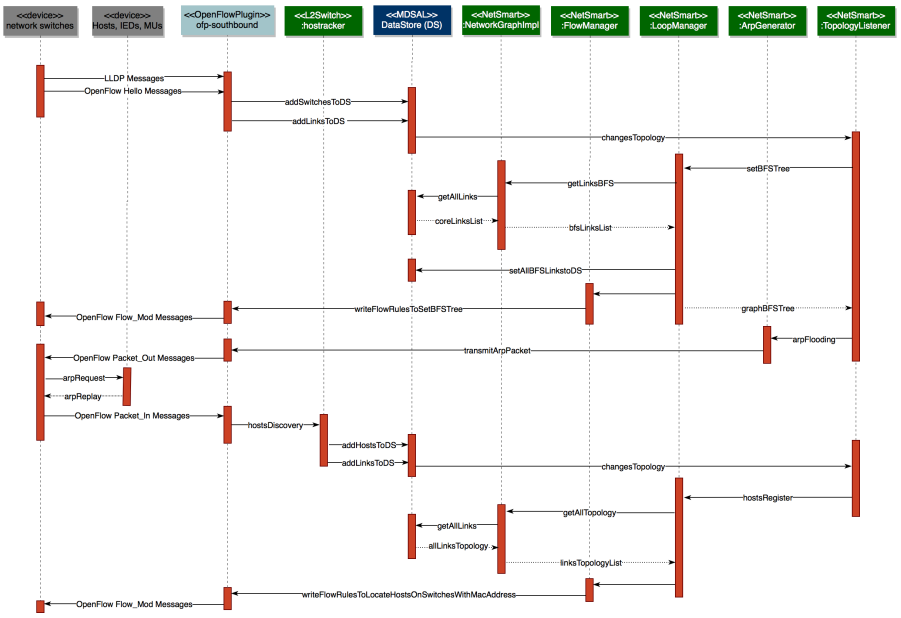


Figure 3.14: Sequence diagram associated with the topology discovery and loop control.

SV MULTICAST TRAFFIC MANAGEMENT The SV multicast traffic management module of the NetSmartSubstation project (see Figure 3.13), has the purpose of calculating trees for the adequate propagation of the SV multicast traffic. In this way, it is guaranteed that this traffic does not flood the network. A *SvTree* tree will include: a root node that corresponds to an SV traffic generator device or *Publisher*, a MU; sheets that correspond to IEDs devices or *Subscribers*; intermediate nodes that correspond to switches and branches that are only the links through which this traffic will be transmitted. Here it is important to mention that each Publisher will have its own propagation tree (*SvTree*).

Next, Figure 3.15 illustrates the sequence diagram associated with the operations carried out for the SV multicast traffic management. The orange box in the sequence diagram shows the process to execute when occurring a link failure in the network topology. Detailed information about the development of this functionality can be found in Section 5.4.

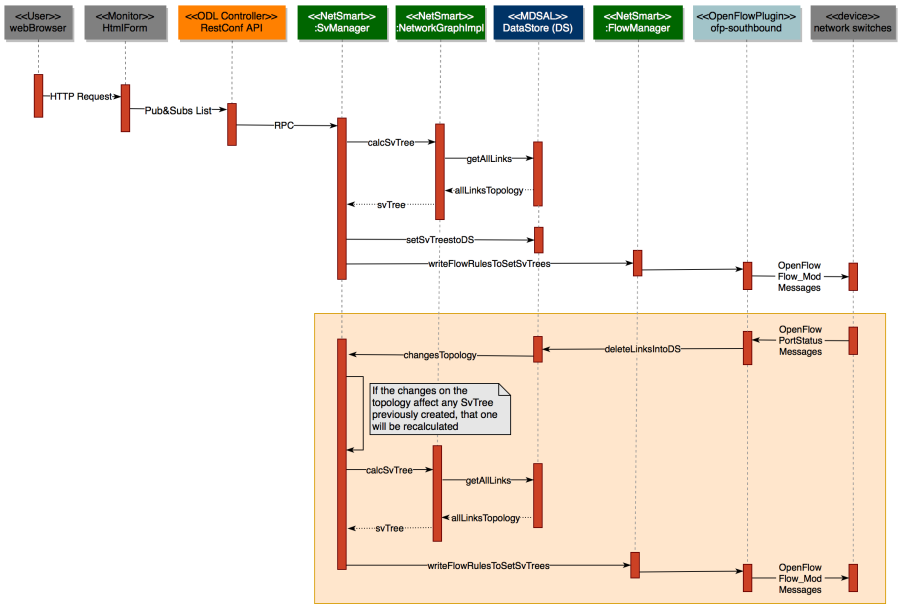


Figure 3.15: Sequence diagram associated with the SV multicast traffic management.

GOOSE MULTICAST TRAFFIC MANAGEMENT The GOOSE multicast traffic management module of the NetSmartSubstation project (see Figure 3.13), has the purpose of calculating a directed graph with two link-disjoint paths between any source and destination switch of the topology, so that if one link in the main path fails, communication is still possible through a backup path. This feature guarantees zero recovery time with no packet loss. In addition, this directed graph allows an adequate propagation of the GOOSE multicast traffic, preventing the network flooding. This module emulates the functionalities of zero recovery time protocols in a single network topology, providing path redundancy and packet redundancy.

Next, Figure 3.16 illustrates the sequence diagram associated with the operations carried out for the GOOSE multicast traffic management. The orange box in the sequence diagram shows the process to execute in case of a link failure in the network topology. Detailed information about the development of this functionality can be found in Section 5.5.

3.3.5.4 *Physical View*

The physical view, or deployment view, represents the software components that make up the solution of the system and how they are distributed on the hardware that will support the application. This view provides guidance at the moment of deploying the solution inside the infrastructure, illustrating the devices that comprise the solution and their corresponding physical and logical connections. Next, Figure 3.17 shows the layout of the software components involved in

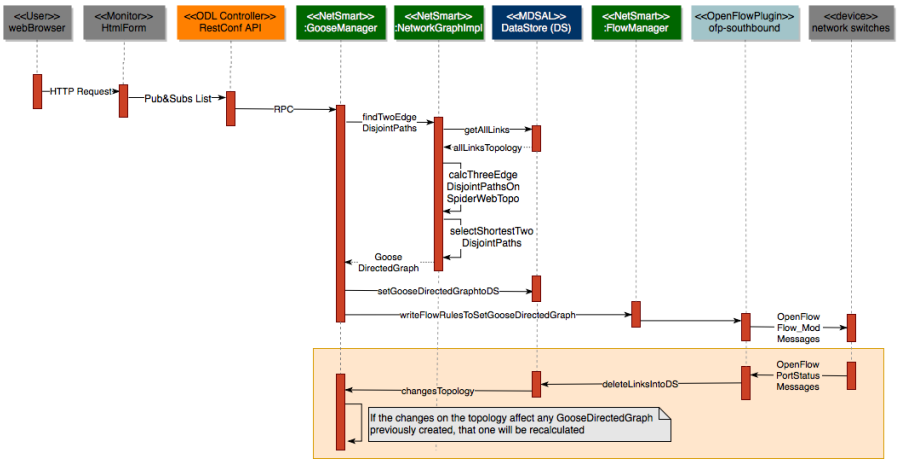


Figure 3.16: Sequence diagram associated with the GOOSE multicast traffic management.

the solution directly (« *monitor* » and « *controller* »), and indirectly (« *webBrowser* », « *sFlowAgent* », « *OpenFlow13* »).

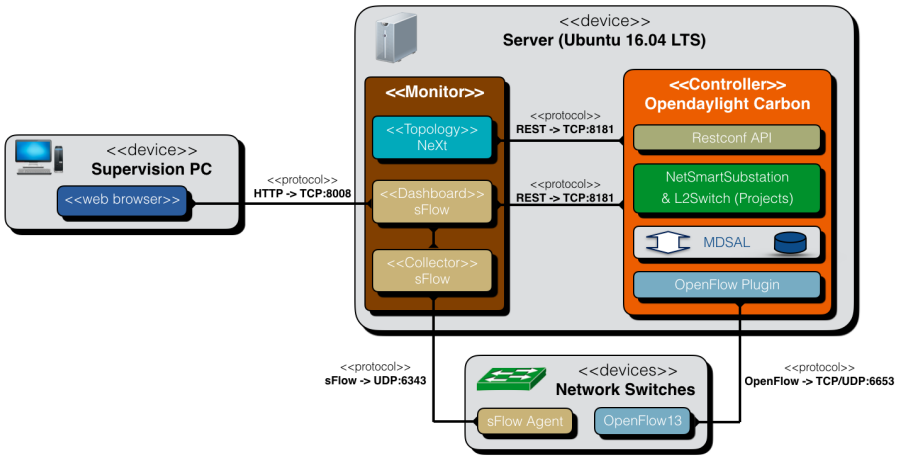


Figure 3.17: Software components distribution that integrate the system on physical equipment.

- *Server.* Equipment where the controller and the monitoring component will be executed. It is recommended that this equipment has at least 8G RAM and a CoreI7 or similar processor. In addition, this server must have enabled the ports illustrated in Figure 3.17 for its correct operation (TCP: 8008, 8181, 6653 and UDP: 6343,6653).
- *Supervision PC.* Workstation of the power substation operator. This device requires a web-browser with JavaScript support to display the network topology, and traffic metrics predefined in the system. This station communicates with the Server through the web interface that is exposed on the TCP port:8008 via HTTP.

- *Switches.* Layer 2 devices in charge of ensuring connectivity within of the power substation communications network. These devices must offer support to the OpenFlow v1.3 protocol and sFlow standard. The sFlow agent must be configured, with the following sampling and polling parameters [sampling = 500 & polling = 20], to send the traffic statistics to the IP address assigned to the server where the « *Collector* » component resides.

Figure 3.18 illustrates the connection scheme used for the aforementioned devices. The proposed network topology is highly reliable to be implemented in power substations environments, according to the network redundancy considerations proposed by the IEC 62439 standard, as well as the operation time requirements suggested by the on IEC 61850 standard. Its conception and evaluation are explained in Chapter 4

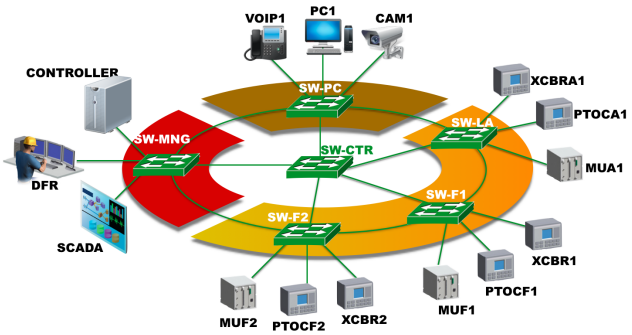


Figure 3.18: Use case diagram, network switch with SDN support

3.3.5.5 *Scenarios View*

The scenarios or use cases view help to model the behavior of the system by illustrating the user's interaction with the main system or other external systems. In this way the stakeholders understand how the system has been conceived to be used. To describe this view, four use cases diagrams are illustrated in Figure 3.22, Figure 3.21, Figure 3.20 and Figure 3.19.

3.4 CONCLUSION

This chapter presents two significant contributions in the field of communications network of power substations. First, the introduction to an architecture known such as S3N, which allows to represent, alternatively, the operation scheme of current, and future, automated communication networks. This new approach, based on technology enablers such as SDN and virtualization technologies, provides benefits to the communications network management, in line with the SmartGrid concept, and opens the door for possible areas of research

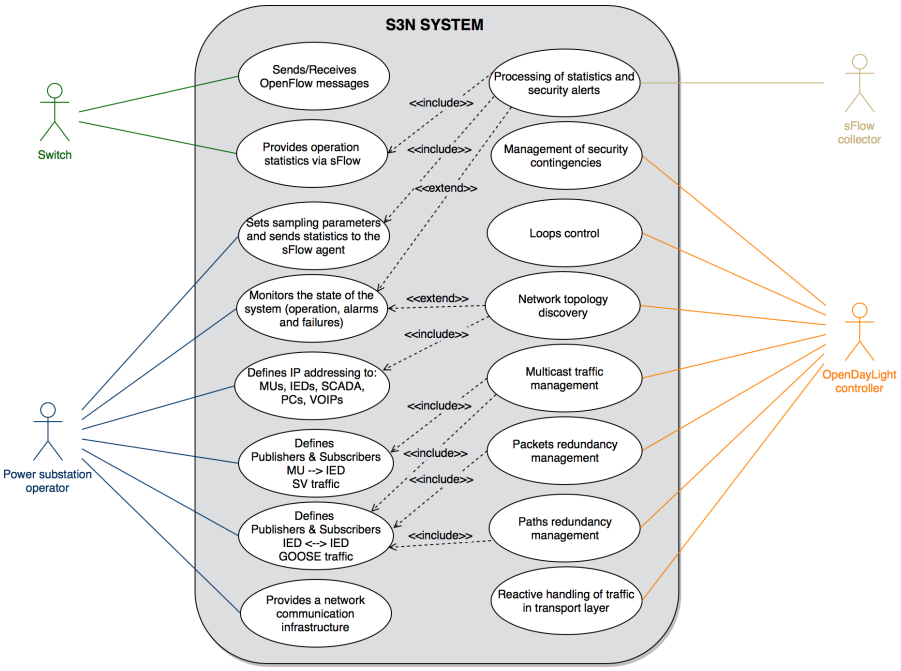


Figure 3.19: Use case diagram, S3N system

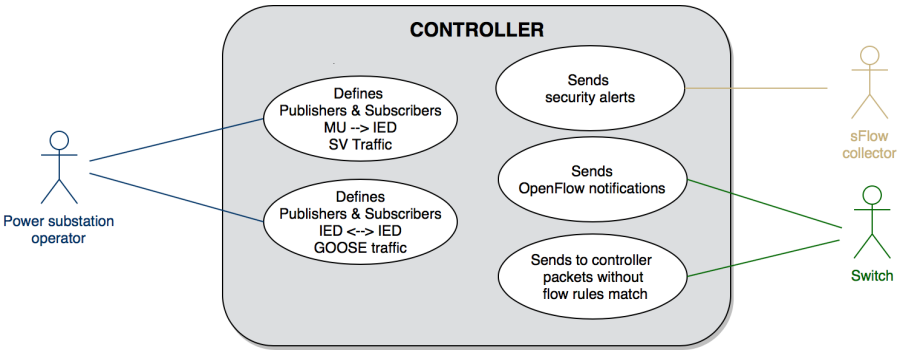


Figure 3.20: Use case diagram, S3N controller

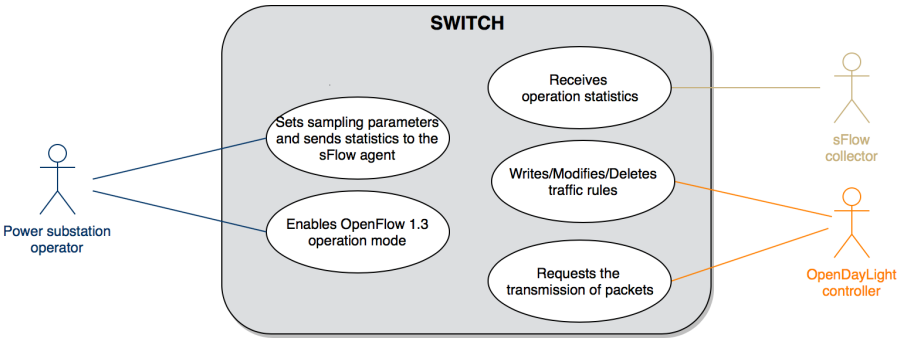


Figure 3.21: Use case diagram, network switch with SDN support

and development. For example, in the field of provisioning and management of S3N-PROTECT and S3N-MANAGE components. Secondly,

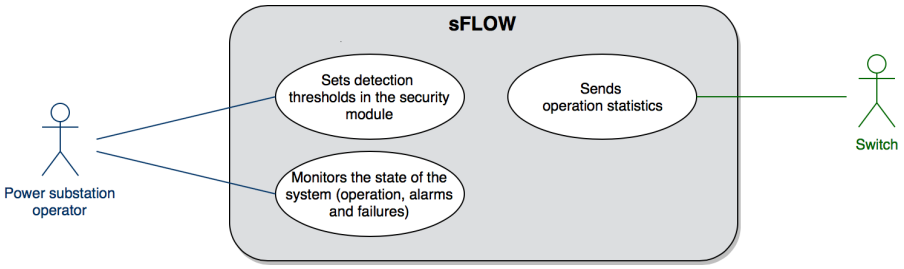


Figure 3.22: Use case diagram, sFlow framework

a novel approach for the future network control plane in power substations is proposed. This novel network control plane architecture is not an adaption of well known SDN-based control planes in different data networks (data centers, carrier networks), but on the contrary, it is defined from power substations' particularities.

4

DEFINING A RELIABLE NETWORK TOPOLOGY IN A SOFTWARE-DEFINED POWER SUBSTATIONS CONTEXT

A network topology is an entity with many critical connections (links). In the same way, a relationship is an entity with many critical connections (feelings). So, if you want to get the best of both worlds ensure that those connections be reliable.

— Alexánder Leal

This chapter proposes a methodology to specify and characterize a reliable topology, using an ILP approach, according to the guidelines described in the architecture Smart Solution for Substation Networks (S3N) where station and process buses are not separated (see Section 3.2.4.1). However, this procedure can be applied in any other environment, not just in the power substation communications network. This contribution was submitted to [LB19b].

4.1 INTRODUCTION

A substation communications network is a mission-critical network and needs to be designed with redundancy principles to guarantee fault-tolerance. In these networks, the interconnection of devices is a key element in system's operation because they (IEDs, Bay Controllers, MUs, DFR and HMI) require connectivity with a high level of reliability.

In this scenario, the topology is the main component to provide reliability to the communications network. Currently, different network topologies can be implemented for substation networks based on the IEC 61850 standard, for example: Star, Ring, Multiple Ring, Mesh or combinations of these [Int13a]. However, there is no single network topology that reaches better performance for all substation automation applications. Each topology has its strengths and weaknesses depending on the use, but it must always ensure redundancy and low latency. This means, if one connection element fails, communication should still be possible through a backup connection ensuring an appropriate delay. Although there are other important factors associated with the choice of the topology (relative cost, administration issues and application suitability), our approach is oriented to the reliability and latency aspects in a Software-defined power substations context.

In Section 3.2.4.1, we presented a reconceptualization of the power substations communications network architecture called S3N, aligned to a Software-defined environment. This proposal represents all interactions between the elements incorporated by the network model

provided in the standard IEC 61850 (a hierarchical architecture integrated by three levels identified as station, bay and process; and interconnected via the process bus and the station bus)[Tc5].

In the S3N architecture, the central axis corresponds to all interconnection devices, mainly Ethernet switches and wiring. This component is called connectivity or S3N-CONNECT and provides communication among all the components that are directly related to the operation of the communications network. S3N-CONNECT provides connectivity to all devices in a single physical network, in contrast to traditional approaches where two separated physical networks are used: one called *station bus* (governed by a ring of partial mesh topologies) and the other one called *process bus* (defined for the resiliency introduced by PRP and HSR protocols) [Hir12], see Figure 2.2.

Therefore, it is paramount to define a convenient topology, able to ensure network resiliency and zero recovery time, for the right system operation in a power substation communications network. This chapter proposes the selection of a reliable network topology using an ILP approach. The criteria used to achieve this objective are: SDN environment, edge-disjoint paths (redundancy) and IEC 61850-90-5 recommendations (backward compatibility). The reliability of the proposed solution is evaluated using the following metrics: terminal reliability, graph metrics and end-to-end time-delay.

4.2 PROBLEM STATEMENT

The concept of automated substation, guided by the IEC 61850 standard [Tc5], allowed the transition from traditional wired connections to an Ethernet-based network with IP support, introducing the definition of new communication concepts in order to provide: reliability, interoperability and flexibility. For instance, new network protocols have appeared with the aim of offering redundancy (PRP and HSR), bring protection and control (GOOSE), get measurements (SV) or allow management (MMS), among others. Also, regarding network topology, the convention of using two communications buses (station and process) to interconnect all devices within the power substation is nowadays commonly used. However, all these new features increase the network management complexity, as well as the CAPEX (CAPital EXpenditures) and OPEX (OPerating EXpense). Particularly, this chapter is focused on proposing a reliable network topology in a Software-defined power substations context, because we consider that technological enablers such as SDN facilitate the administration of complex networks, guaranteeing that network topology becomes agnostic to the operation scheme of any protocol.

In previous works [Kan11; YK10; Ali+15; Liu+14], different authors have discussed about how the network topology in a power substation

environment should look like. However, each approach proposes very different solutions (see Figure 4.1).

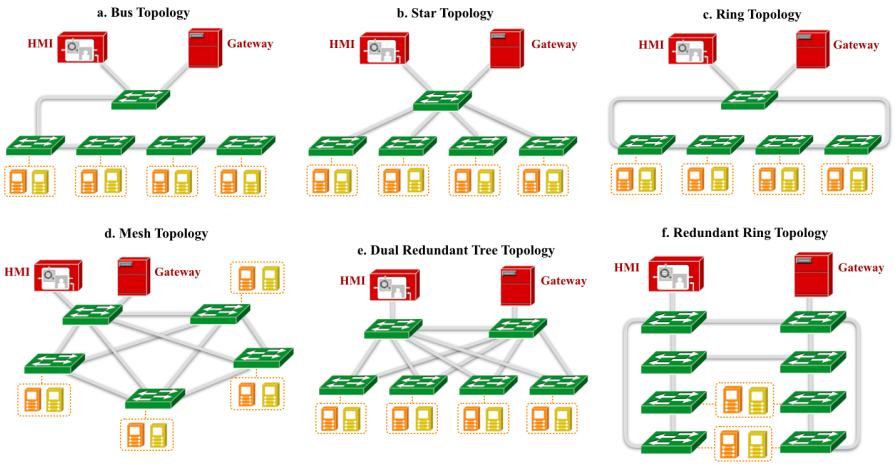


Figure 4.1: Network Topologies in power substations

For example, the IEEE members of the Substation Protection Subcommittee inside the Power System Relaying Committee in [IEE15] agree with the analysis done in [Kan11] where the redundant-ring (see Figure 4.1f) provides the highest reliability as compared to other three studied topologies (Cascade, Ring, Star-ring). On the other hand, in [YK10], authors suggest that dual redundant tree-based topology is the best option based on Cisco's experience. In fact, all solutions are appropriated as long as they fulfill the required operation conditions.

The following subsections explain which conditions should be taken into account to define a reliable topology in a Software-defined power substations context.

4.2.1 *S3N Architecture Legacy*

In the S3N architecture, the connectivity component (S3N-CONNECT) is governed by a SDN environment, integrating in a single network the station and process buses defined in the IEC 61850 standard [LB16a]. For this reason, in the topology design problem, we are going to consider a single network topology. In the SDN environment, it is necessary to establish a communication channel between the SDN controller and the set of switches of the network topology, with the purpose of exchanging control messages. There are two ways to implement this control channel: *out-of-band* and *in-band* control. *Out-of-band* control uses separate Ethernet ports and links to connect the switches to the controller and exchanges control traffic in a dedicated way, acting as a separate physical network. On the other hand, *in-band* control uses available network links to send both types of traffic, data and control.

The S3N architecture is based on a concentric model (see Figure 3.2). Therefore, at the beginning, to provide basic connectivity among all the edge components of the architecture (HMI, IEDs, MUs, Bay Controllers, Cameras, among others), the switches attached to those components will be interconnected by a star topology to the main switch (located at the center of the star). Hence, the communication between SDN controller and the others switches is carried out through the main switch, using *in-band* SDN management.

4.2.2 *Network engineering guidelines (IEC 61850-90-4)*

Ideally, all devices belonging to a single bay (IEDs, MUs and/or bay Controller) should be connected to a single switch [Int13a].

4.2.3 *Redundancy*

Redundancy refers to the careful use of redundant connections between network devices or, the use of multiple network devices itself, to provide reliability on the network. This ensures that, if a link or node component fails, several communication connections will not be affected. Particularly, in our approach, the network topology is oriented to guarantee two link-disjoint paths between any source and destination switch of the topology, so that if one link in the main path fails, communication is still possible through a backup path. This feature is the most important issue in our design because, in the S3N architecture, there are two modules in charge of guaranteeing the simultaneous transmission of multicast GOOSE messages through two independent paths: *Path redundancy* and *Packet redundancy* (see Section 3.1.2). In this way, the S3N architecture will provide similar behavior to the one provided by protocols such as PRP or HSR, without the need to use additional devices (redbox or HSR interface) or duplicated networks. Figure 4.2 shows, on the same graph, the concept related to the expressions two node-disjoint paths and two link-disjoint paths (edge-disjoint paths), for a pair of vertices (u,v).

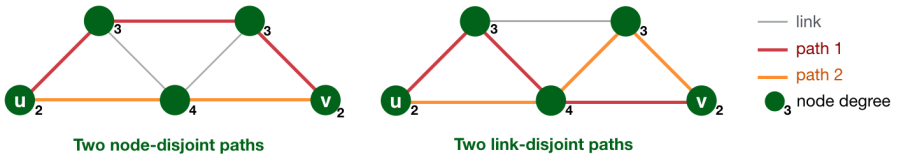


Figure 4.2: Node-disjoint paths and Link-disjoint paths concepts

In this context, to guarantee the aforementioned operation, a *two-edge-connected* graph is required. First, a connected graph is a graph where there always exists a path between any pair of vertices. Therefore, a *two-edge-connected* graph is a graph where there are two paths

to connect any pair of vertices of the topology; it means that the graph will always be connected, even if any edge of the graph is removed (see Figure 4.2).

To determine what are the requirements in a *two-edge-connected* graph, we are going to use one of the most important facts about connectivity in graph theory; the Menger's theorem [Men27], which characterizes the connectivity and edge-connectivity of a graph in terms of the number of independent paths between vertices.

> *Menger's Theorem (edge version): Let G be a graph and u, v ∈ V(G). The maximum number of edge-disjoint u,v paths in G is equal to the minimum number of edges needed to be removed from G to disconnect u from v.*

Then, as a consequence of Menger's Theorem, the maximum edge connectivity of a given graph is the smallest degree of any node, since deleting these edges disconnects the graph [Ski90]. In this way, according to the Menger's Theorem, Figure 4.2, corresponds to *two-edge-connected* graph because the smallest degree of the nodes is two. In conclusion, to guarantee a *two-edge-connected* graph we have to ensure that each node in our topology is connected, at least, through two links.

4.3 OPTIMIZATION MODEL

4.3.1 *Formulation*

To solve the problem of generating a reliable network topology in a Software-defined power substations context, we propose an ILP model. Before introducing the model, Table 4.1 details the notation. Also, the terms "switch" and "node" are used interchangeably throughout this chapter.

Parameter	n	Refers to the number of switches (Nodes), in the S3N-Connect component.
Indexes	i	index associated to the source node of the link.
	j	index associated to the destination node of the link.
Variable	$x_{i,j}$	Binary variable to indicate the presence of a link between the i node and the j node.

Table 4.1: Model Elements

4.3.2 *Objective function*

Suppose the network topology is represented by an undirected graph $G(N, L)$ composed of a set of nodes N and a set of links L. The links are assumed to be bidirectional, with the same characteristics (cost, bandwidth). The goal of the model is to formulate a network topology

that guarantees the reliability considerations defined in the previous section, given the n parameter, such that the number of required links is minimum.

$$minimize \quad \sum_{i}\sum_{j} x_{i,j} \tag{4.1}$$

4.3.3 *Constraints*

- *No loops:* A node cannot be source and destination of one link.

$$x_{i,j} = 0 \;\Rightarrow\; i = j \tag{4.2}$$

- *SDN Criteria:* All switches (nodes) are connected to the main switch to facilitate their connection to the SDN controller, since the main switch is directly attached to the SDN controller. For the sake of simplicity in the formulation, the main switch corresponds to the first node ($i = 1$).

$$\sum_{j=2}^{n} x_{i,j} = n - 1 \quad \Rightarrow \quad i = 1 \tag{4.3}$$

- *Link-disjoint paths:* Each node (switch) should be connected, at least, through two links. In other words, the minimal degree for each node (switch) should be two.

$$\sum_{j=1}^{n} x_{i,j} \geq 2 \;\; \forall\, i \tag{4.4}$$

To summarize, Figure 4.3 illustrates the meaning of the constrains aforementioned on the network topology modeling.

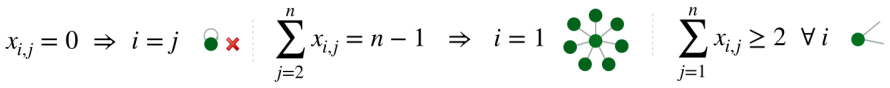


Figure 4.3: Constrains graph representation

4.3.4 *Solution*

The solver used to solve the ILP aforementioned problem was the GNU Linear Programming Kit (GLPK) [Makoo]. The results obtained for

different n values are shown in Figure 4.4. As noted, the restrictions were accomplished and the reader can check by direct observation that the obtained graph is *two-edge-connected*, even though one link fails, the connectivity in the network remains because there is a feasible path between any pair of nodes.

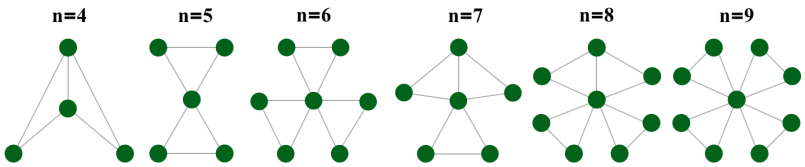


Figure 4.4: Topologies obtained from the proposed ILP

4.3.5 *Improved Solution*

The resulting topologies show a critical central failure point. Accordingly, we propose to increment the degree of the nodes to 3 in the constraint defined by the Equation 4.4, obtaining the results illustrated in Figure 4.5 where it is easy to note that changing the constraint raised in the Equation 4.4, we achieve a 100% redundant solution, even if the there is a failure in the central node. This topology is known as an artificial spider web or orb web, and could be regarded as the mixture of two topologies: star and ring. A spider web topology gives a highly optimized structure, efficient to rapidly transmit information between nodes that are located further away from the center of the topology and 100% tolerant to link failures. [AO10]. Here, it is important to mention that in [Liu+14], this topology was analyzed as an architecture for substation process-level network with promising results in comparison with traditional topologies (bus, ring and star).

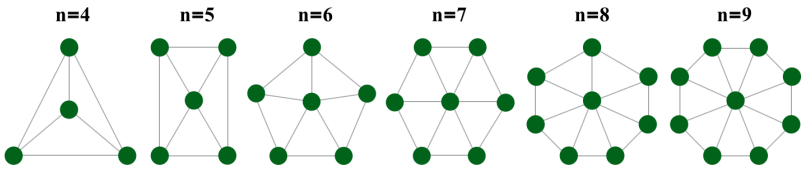


Figure 4.5: Topologies obtained for the improved optimization model

4.4 PERFORMANCE EVALUATION

To validate the reliability of the obtained solution, we choose to compare the performance of different network topologies (included the spider web topology), over a use-case. Particularly, we choose a small substation of 220/132 kV, with single bus, which is classified as T1-1 by IEC 61850-5 [Int13b]. Figure 4.6 shows the single line diagram for such substation, along with a possible structure for the devices of control, protection and measuring, in each line bay.

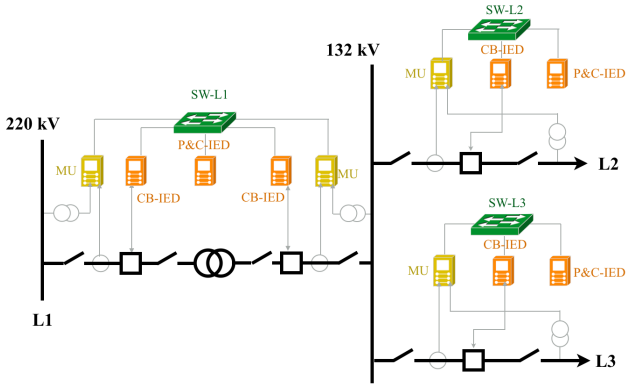


Figure 4.6: Substation T1-1 type

According to [Kan11], in this type of substation, there are 5 bays, three line bays, one transformer bay and one bus bay, where each bay has a specific scheme of measurement, protection and control. Furthermore, with regard to the S3N architecture we will have to interconnect at least 7 switches: SW_{MNG}, SW_{COM}, SW_{L1}, SW_{L2}, SW_{L3}, SW_{XFRM} and SW_{BUS} (see Figure 4.7).

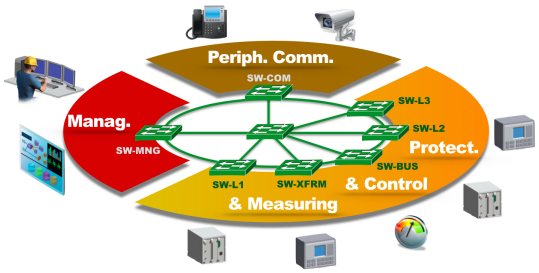


Figure 4.7: Spider web topology under T1-1 use-case

The evaluation process involves three different types of analysis: k-Terminal Reliability, Graph Metrics and ETE delay analysis; over seven different network topologies, according to the aforementioned use-case (see Figure 4.8).

However, it is important to mention that, although the suggested methodology will be applied to one particular type of substation, this method can be applied to any power substation.

4.4.1 *k-terminal Reliability*

Terminal reliability (connectedness probability), is defined as the probability of maintaining nodes connected considering all possible paths between origin-destination pairs. In other words, this metric reflects the redundancy of a network in which alternative routes could be used when a link fails. That is the reason why we are going to use this metric to characterize the probability of network operation. Specifically, k-terminal reliability gives the probability that k specified nodes in a network are connected [BF12].

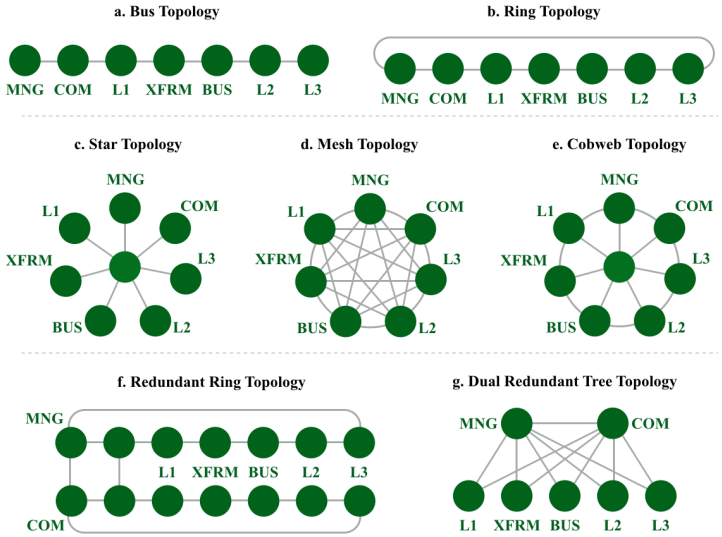


Figure 4.8: Different network topologies for the proposed use-case

For the computation of the two-terminal reliability metric, we choose the algorithm SYREL [HR87], which incorporates the best features of both path and cutset methods, is accurate and has a relatively low computational complexity. At SYREL, the network topology is represented by a probabilistic graph $G(N, L)$, where N and L are the set of nodes (switches) and links. Also, each simple path between a given pair of nodes is denoted as P_i (with $i = 1, 2, 3...$). Finally, the algorithm assumes that the failures of elements are statistically independent and they have a probability of being available p_l, or being in failed state q_l, with $p_l = 1 - q_l$ and $l \in L$. The terminal reliability between a pair of nodes is given by Equation 4.5.

$$R_{terminal} = Pr(\bigcup_{i=1}^{m} E_i) \tag{4.5}$$

where E_i denotes the event in which path P_i is up and m represents the number of paths between those two nodes. This expression for the terminal reliability can be computed by decomposing the set of paths in the graph into another set of mutually exclusive paths between two nodes. For example, if we have three possible paths between two nodes called P_1, P_2 and P_3 (see Figure 4.9); the terminal reliability expression consists of three terms which correspond to three mutually exclusive events: the first event occurs when P_1 is up, the second event occurs when P_2 is up and P_1 is down, and the third event occurs when P_3 is up and both P_1 and P_2 are down. In other words, if $\bar{E}_i$ denotes that path i is not operational, $R_{terminal}$ can be decomposed into mutually exclusive events as $Pr(E_1) + Pr(E_2 \wedge \bar{E}_1) + Pr(E_3 \wedge \bar{E}_2 \wedge \bar{E}_1)$.

Now, for applying this mechanism it is necessary to establish what is the probability that link $l \in L$ is available, p_l. For this purpose, we use the MTBF concept [TA04]. MTBF, or mean time between failures, which

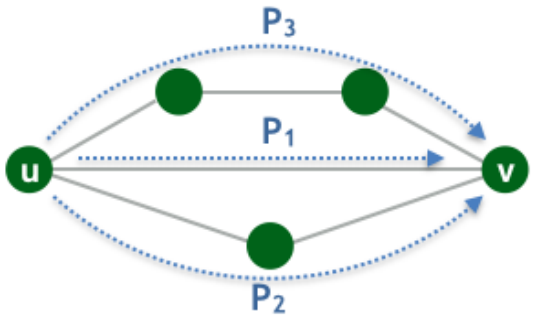


Figure 4.9: Possible paths between two nodes in a K-terminal context

is a basic measure of a system reliability, typically represented in units of time, where the higher MTBF number indicates the higher reliability of the product. Now, once the MTBF of an element is calculated, the probability that element will be operational for a given time could be expressed with the Equation 4.6, assuming a failure-rate constant.

$$R_{system} = e^{-(time/MTBF)} \tag{4.6}$$

Thus, taking into account that the MTBF for passive copper cables is around 50 million hours (the order of magnitude higher in the industry) [Sha15], and the MTBF for an Ethernet network interface is 1 million hours [Cis15]; we can determine the probability that these elements operate without failure during a time using Equation 4.6. For example, for a period of 10 years, the probabilities that a cooper cable and Ethernet network interface remain available are 0.99825 and 0.91613 respectively.

Now, we can define p_l, the probability that link $l \in L$ is available with Equation 4.7, (see Figure 4.9).

$$p_l = (R_{nic-src} \cap R_{cable} \cap R_{nic-dst}) = 0.83783_{(10\ years)} \tag{4.7}$$

4.4.2 *Connectivity Measures*

Graph theory gives several measures to analyze and compare network efficiency over different topologies [KD89; Bar16]. In this methodology, we use the diameter as delay metric, beta index as a complexity indicator, the number of edge-disjoint paths as resilience/reliability measure and the average path length as a robustness attribute.

- *Diameter (d_{max}):* Indicates the largest distance between any two nodes in the network, through the shortest path. Diameter (d_{max}) could be defined by Equation 4.8 where $d(i,j)$ denotes the length of the shortest path between node i and j. This feature is important in our comparison because it gives a perspective about the

behavior of the latency, since the latency is directly proportional to the distance $d(i,j)$. High values of d_{max} imply an important factor to take into account in the latency operation.

$$d_{max} = max \;\; d(i,j) \tag{4.8}$$

- *Beta Index(β):* It measures the level of connectivity in a graph and is expressed by the relationship between the number of links (l) over the number of nodes (n). Beta index can take values between 0 and 3. Simple network topologies like tree or bus have a β value of less than one. A connected network with one cycle has a value of 1. And more complex networks have a value greater than 1.

$$\beta = l/n \tag{4.9}$$

- *Average shortest path length ($\langle d \rangle$):* It is defined as the average of the shortest paths between all nodes in the network. For a directed network of N nodes, $\langle d \rangle$ gives the expected distance between two connected nodes. A short average path length indicates a lower probability of system failure since the transmission path will have fewer elements.

$$\langle d \rangle = \frac{1}{n(n-1)} \sum_{i \neq j} d(i,j) \;\; \forall\, i,j \in V \tag{4.10}$$

- *edge-disjoint paths ($P_{edge-disj}$):* Two paths are edge-disjoint (edge independent) if they don't share any edges.

4.4.3 *End-To-End Delay*

Latency is a critical factor in the operation of communications networks of power substations and its performance is directly related to the number of hops (switches) per path. That is the reason why we consider its analysis in this paper. IEC 61850-5 [Int13b], defines the maximum transmission times required for different services. For example, SV data values and GOOSE trip commands are services with the highest latency requirements (Transfer Time class TT6), which demand 3ms as transmission limit or End-To-End delay (ETE). The end-to-end delay is the elapsed time from the time a data packet is sent out from the source application layer, until it is completely received by the application layer in the destination node. However, it is very important to take into account that, according to IEC 61850-10 [Int12]

and IEC 61850-90-4 [Int13a], the ETE of 3ms is distributed as follows: 80 percent to the processing times in the IED stacks (2.4ms) and the remaining 20 percent (0.6ms) for the communication network.

In [Int13a; Sie14; XN13], authors analyze quantitatively the latency caused by a single hop in a path evaluating the wireline propagation delay and the processing time of a packet at a switch (store and forward, switch fabric processing, and frame queuing processes).

- *Wireline Propagation Delay (T_{wl}):* Refers to the elapsed time taken by the packet to traverse the physical medium. It depends on the physical link length and propagation speed. Now, assuming that we will use Ethernet cable CAT-5E / CAT-6, which gives a propagation delay of 0.64C in a distance of 100mt, where C is the speed of light, we can define the T_{wl} as follows Equation 4.11:

$$T_{wl} = \frac{distance}{prop.factor * C} = \frac{100mts}{0.64C} = 520ns \tag{4.11}$$

- *Store and Forward Delay (T_{sf}):* Refers to the elapsed time in the switch while the first packet is fully received and stored in memory, the packet is read back from memory, and the packet is forwarded to the output queue. This delay is proportional to the size of the frame forwarding and is inversely proportional to the bit rate. Now, assuming that we are working on a FastEthernet network (100Mbps) with compact GOOSE frames of 300bytes, the T_{sf} can be defined as follows Equation 4.12:

$$T_{sf} = \frac{framesize}{bitrate} = \frac{300bytes}{100Mbps} = 24\mu s \tag{4.12}$$

- *Switch processing delay (T_{sw}):* Corresponds to a fixed value given by the vendor, which depends on the processing speed of switch chip. Generally, industrial Ethernet switches have a value around $8\mu s$.

- *Queuing delay (T_q):* It is the most difficult metric to determine because depends on the knowledge of all traffic patterns on a network (network load), at any moment. For this reason, some assumptions have to be made. Thus, on a loaded network, it is possible to assume that the likelihood of a frame already in the queue is proportional to the network load. Under this condition, the average queuing latency can then be estimated in Equation 4.13, where $T_{sf(max)}$ is the store and forward latency of a full-size frame (1500 byte). Now, a network with 50% load would have an average queuing latency of $60\mu s$

$$T_q = (\%_{load})T_{sf(max)} = (0.5)\frac{1500bytes}{100Mbps} = 60\mu s \tag{4.13}$$

Based on the analysis above, we propose the scenario of Figure 4.10. to calculate the latency caused by links and interconnection devices (switches) on the communication network path. Particularly, the scenario is validated for network topologies with $d_{max} = 2$ in normal operation conditions (e.g. star, spider-web and dual-tree). However, the proposed methodology could be applied to the calculation of other distances or topologies.

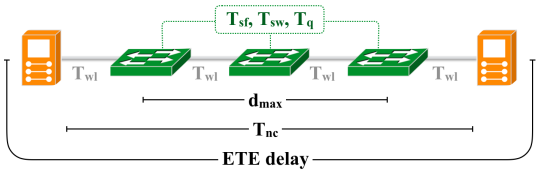


Figure 4.10: Elements involve in the estimate latency for End-To-End Delay

According to Figure 4.10, the latency related to the network communications T_{nc} will be given by Equation 4.14,

$$T_{nc} = 4\ T_{wl} + 3\ (T_{sf} + T_{sw} + T_q) = 278\mu s \tag{4.14}$$

4.5 ANALYSIS OF RESULTS

The reliability of the obtained solution was evaluated using three different approaches: k-terminal reliability, graph metrics and ETE time-delay performance.

Table 4.2 summarizes the results obtained for each of the aforementioned metrics, according to the network topologies defined for our use case (see Figure 4.8).

TOPOLOGY	$R_{(MNG,L3)} \rightarrow P_i$	$R_{(L1,L3)} \rightarrow P_i$	$R_{(L2,L3)} \rightarrow P_i$	β	d_{max}	$\langle d \rangle$	$P_{edge-disj}$	$ETE\ (d_{max})$
Bus	0.346 → 1	0.493 → 1	0.837 → 1	0.86	6	2.66	1	$556\mu s$
Ring	0.838 → 2	0.791 → 2	0.838 → 2	1.00	3	2.00	2	$370\mu s$
Star	0.792 → 1	0.792 → 1	0.792 → 1	0.88	2	1.75	1	$278\mu s$
Mesh	0.999 → 326	0.999 → 326	0.999 → 326	3.00	1	1.00	6	$186\mu s$
Redun. Ring	0.900 → 4	0.827 → 4	0.900 → 4	1.14	7	3.04	2	$648\mu s$
Dual Tree	0.973 → 6	0.947 → 10	0.947 → 10	1.56	2	1.47	2	$278\mu s$
Spider Web	0.926 → 29	0.953 → 27	0.990 → 23	1.75	2	1.50	3	$278\mu s$

Table 4.2: Two Terminal Reliability, Connectivity and ETE analysis for different topologies

4.5.1 *Two terminal Reliability*

On each network topology (see Figure 4.8), the two terminal reliability calculations were conducted for three different pairs of nodes $R_{(MNG,L3)}, R_{(L1,L3)}$, and $R_{(L2,L3)}$ to verify the behavior when nodes are located both, close or far away. The terminal reliability analysis showed that topologies with highest values of probability of maintaining nodes connected were: mesh (with a value of 0.999 on the

three different pairs of nodes: $R_{(MNG,L3)}$, $R_{(L1,L3)}$, $R_{(L2,L3)}$); spider web (with values of 0.926, 0.953 and 0.990 for each pair of nodes) and dual redundant tree (with values of 0.973, 0.947 and 0.947 for each pair of nodes). However, although the mesh is the topology with greater redundancy in our study, its complexity, as shown by the beta index, is the worst in terms of network management; that is the reason why this topology is not implemented on real networks and it will be disregarded in our analysis. Hence, in terms of reliability, the spider web and the dual redundant tree are the network topologies offering the best performance.

Also, next to the two-terminal reliability value, the number of paths used to calculate P_i are indicated (see Table 4.2). There, if we compare the P_i values obtained by the spider web and the dual redundant tree network topologies, one can observe that the spider web topology gives two or even three times more paths to interconnect a pair of nodes than dual redundant tree topology (29 instead of 6, 27 instead of 10 and 23 instead of 10), according to the pair of nodes evaluated.

In this context, the network topology with best two terminal reliability is the spider web.

4.5.2 *Graph metrics*

Let us begin this part of the analysis with the diameter d_{max} and the average shortest path length $\langle d \rangle$ metrics. These parameters give an idea about the proximity level of the network. Small values of these parameters indicate the best network performance because if, in average nodes are closer, the transmission path will have fewer elements, and that means a lower probability of system failure and shorter delays. According to Table 4.2, network topologies with smaller values for d_{max} and $\langle d \rangle$ measurements, excluding the mesh topology for the reasons aforementioned, are dual redundant tree ($d_{max} = 2$, $\langle d \rangle = 1.47$) and spider web ($d_{max} = 2$, $\langle d \rangle = 1.50$).

Another important parameter used in this analysis is the beta index (β). This index is a good instrument to address the complexity of the networks. Values between 1 and 2 are appropriate, while that 3 it is the worst scenario (e.g. mesh topology). Topologies such as a ring ($\beta = 1$), dual ring ($\beta = 1.14$), dual redundant tree ($\beta = 1.56$) and spider web ($\beta = 1.75$) offer an appropriate complexity level. Nevertheless, ring and dual ring topologies did not get a good qualification under the first analysis criteria (terminal reliability), even when they offer two edge-disjoint paths. Consequently, dual redundant tree and spider web topologies continue displaying the best metrics.

Last graph metric to present in this discussion is the number of edge-disjoint paths between a pair of nodes ($P_{edge-disj}$). This graph parameter brings a determining factor in this evaluation in terms of reliability requirements. Table 4.2 shows that spider web topology with

$P_{edge-disj} = 3$, offers the best option above other topologies, without taking into account the mesh topology.

In this terms, although dual redundant tree and spider web topologies give similar values to graph meters d_{max}, $\langle d \rangle$ and β, the parameter $P_{edge-disj}$ is paramount to select the spider web topology.

4.5.3 *ETE time-delay*

As a third analysis criteria, the latency is an important indicator. The ETE time-delay analysis showed that dual redundant tree and spider web topologies obtained a value of $ETE(d_{max}) = 278\mu s$, assuming that all switches of the path had a network load of a 50% and worked with FastEthernet technology. This measure satisfies the communication network time defined for a IEC 61850-5 standard of $600\mu s$, equivalent to the 20% of the transmission time required for critical services (3ms). However, it is important to mention that: 1) the delay link contribution is negligible in comparison with the delay caused by the switch operation and, 2) the latency values calculated could be reduced implementing GigaEthernet technology, and with this, the margin of reliability of the network would be increased.

In summary, the results confirm that the network topology proposed, spider web, is highly reliable. However, dual redundant tree showed being a good option too.

4.6 CONCLUSION

The selection of the requirements to define the right network topology changes according to the purpose for which a network is built. However, a right balance between the requirements is not easy to get. Reliability, costs, real-time performance and management are features that can not be satisfied simultaneously.

In this chapter, we propose an ILP model to solve the problem of reliability for a network topology in a Software-defined power substations context, taking into account requirements such as SDN legacy, digital substation recommendations and paths redundancy. Our studio corroborated that the proposed solution, the spider web topology, is a reliable network topology suitable for improving the performance of the operation network, by using three different analytical approximations: terminal reliability, graph metrics and ETE time-delay performance; and its comparison against several practical Ethernet architectures.

5 USE CASES IMPLEMENTATION WITH SDN

Make everything as simple as possible, but not simpler.
— Albert Einstein

This chapter exposes the benefits of working in an SDN context exposing how SDN can be applied to get the network functionalities defined in Section 3.1.2, for the S3N architecture's control plane. This contribution was submitted to [LB19b].

5.1 INTRODUCTION

As we mentioned in Chapter 4, a substation communications network has to be designed with redundancy principles to guarantee fault-tolerance. However, managing the proposed topology in a traditional way, using common network protocols like STP or VLAN, would be technically unfeasible. On the one hand, to offer fault-tolerance the traffic must have the possibility to traverse different paths; however, on the other hand, STP will disable all redundant paths in the topology to avoid loops.

This fact becomes a motivation to show the benefits of working in an SDN context. With SDN we can develop applications to define the traffic behavior in the chosen topology, according to the operational requirements of the power substation communications network. This chapter shows the operating principles of the developed modules to solve particular problematics within the communication network for power substations such as network topology discovery, broadcast traffic control in loop-based topologies, path redundancy, packet redundancy, and multicast traffic management.

All proofs of concept presented here were designed and implemented for the T1-1 power substation (see Figure 4.6), using the spiderweb network topology (see Figure 4.7) in accordance with the analysis exposed in Chapter 4. The proposed algorithms were implemented under the Opendaylight SDN controller [Odl], while the topology was built using Mininet [Tea12], a network emulator widely used in the SDN research networking field. Finally, the measuring MUs, protection and control IEDs and supervising devices were emulated using *hosts-containers* executing applications from the libIEC61850 project [Zil16]. libIEC61850 is an open-source framework with code written in C that provides an API for implementing MMS server and client, GOOSE publisher and subscribers and SV publisher and subscribers.

5.2 NETWORK TOPOLOGY DISCOVERY

To get a single end-to-end view of your network infrastructure always has been a key aspect for network monitoring, identifying failures, and improving the network efficiency. So, it is not a coincidence the presence of several vendor protocols to reach this functionality: Cisco Discovery Protocol by Cisco, Foundry Discovery Protocol by Brocade, Bay Network Management Protocol and Nortel Discovery Protocol by Nortel, Extreme Discovery Protocol by Extreme Networks, Link Layer Topology Discovery by Microsoft, and others. But, the constraints related to the use of proprietary protocols are already known. However, there are also neutral network protocols that, working together, enable network topology discovery. For example, LLDP allows identifying the port-to-port connectivity and the capabilities from switches, while ARP and ICMP can be used to discover hosts. In addition, Simple Network Management Protocol (SNMP) is another option to discover the network topology of SNMP-enabled network devices.

In our approach, the network topology discovery is a paramount aspect to develop other functionalities such as: the appliance of graph theory to handle multicast traffic, determine the shortest path between hosts, recovery the network operational state in case of failures, monitor the network, among others.

In particular, we present an SDN solution to get an entire view of the communication network using the ARP and LLDP protocols along with two features related to the native Opendaylight projects: L2Switch and OpenFlowPlugin. This solution comprises two phases: core devices discovery (*switch-nodes*) and their corresponding links, and edge devices discovery (*host-nodes*) and their corresponding links.

- *Core devices discovery (switches and links between switches).* The OpenFlowPlugin uses a procedure called OFDP, which leverages the packet format of LLDP to perform network topology discovery and register it in the SDN controller. However, OFDP operates completely different to LLDP [GEN10]. In short, the SDN controller sends to each switch a message with an LLDP packet and the instructions of what to do with it, encapsulated in a (*Packet-Out primitive*). Then, the switches flood LLDP packets to all ports, according to the instructions received. Subsequently, other switches receive the packets and redirect the LLDP traffic to the SDN controller. Finally, the SDN controller learns the topology network from information about incoming ports (see Figure 5.1). Here, when the OFDP process is completed, we build a network graph $G(N, L)$ where N are switches and L links based on the information available from the *core-network* in the SDN controller.

  Here, it is important to mention that the OFDP mechanism is the current de-facto standard of topology discovery in SDN controllers (POX, Floodlight, Beacon, Ryu, Cisco Open SDN Con-

troller, OpenDaylight and ONOS), derived from the topology discovery mechanism used in NOX (the original OpenFlow controller).

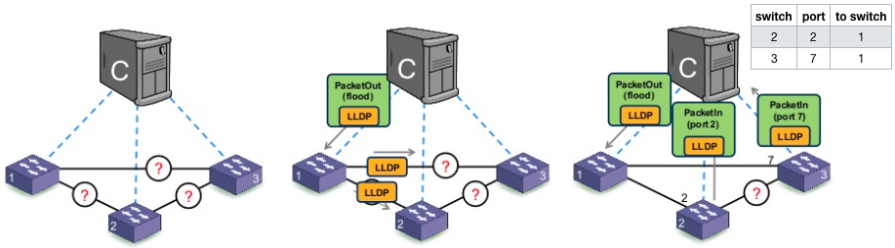


Figure 5.1: Network topology discovery by using OFDP.

However, the described procedure only allows discovering the *switch-nodes* and their links. It means, there is only a partial discovery of the network topology (*core-network*), because the *host-nodes* and their corresponding links are missing.

- *Edge devices discovery (hosts, IEDs, MUs and their links to the switches).* The feature *odl-l2switch-hosttracker* of the L2Switch project is in charge of detecting and registering each *edge-node*, and its corresponding link to the *core-network*, by detecting ARP Replay messages in the SDN controller. So then, it will be necessary to generate ARP requests from the SDN controller to each edge device IP addresses linked to edge devices, which were registered in the SDN controller by the power substation operator. However, before executing this operation, the SDN controller will execute the Algorithm 5.2, to install rules on the *switch-nodes* that guarantee a proper handling of the ARP broadcast traffic within a loop-based network topology.

In summary, with the two aforementioned procedures (*Core and Edge devices discovery*), we can get the complete network graph $G(N, L)$ associated with network topology discovery. This network graph brings a detailed vision of the network topology and shall be used as input in several modules to provide solutions to other complex issues.

The structure of the proposed algorithm is shown in Algorithm 5.1. In addition, the sequence diagram related with the topology discovery and loop control operations was illustrated in paragraph 3.3.5.3.

Initially, line 2 assumes that each OpenFlow-enabled switch has preconfigured the IP address and TCP port number of the SDN controller. So then, on startup, the switches will establish a session with the SDN controller notifying information about MAC addresses, active switch ports, among others [Ope15].

On lines 3 to 5 the OpenFlowPlugin, through the *odl-openflowplugin-app-table-miss-enforcer* feature, registers a flow by default on each

Algorithm 5.1 Network topology discovery in an SDN context (SpiderWeb Topology)

Input: Switches, Links, *ip_addresses_end_devices*
Output: Network Graph $G(N, L)$ where N are switches, MUs or IEDs, and L links
OpenFlow rules set to install in the switches

1: Initialize variables:
switchNodes⟨*List of N*⟩, *discoveredLinks*⟨*List of N*⟩, *networkGraph*⟨$G(N, L)$⟩,
srcNode⟨N⟩, *dstNode*⟨N⟩
2: *switchNodes* ←(all switches with OpenFlow session established)
3: **for each** *switch* ∈ *switchNodes* **do**
4: Add flow priority=0, dl_type=*lldp*, actions=*controller*
5: **end for**
6: **for each** *switch* ∈ *switchNodes* **do**
7: Send *Packet-Out* with packet=LLDP, actions=*flood*
8: **end for**
9: *discoveredLinks* ← (all links learned with OFDP)
10: updateNetworkGraph(*discoveredLinks*)
11: Call Algorithm 5.2 ▷ ARP Manager in an SDN context (SpiderWeb Topology)
12: **for each** *ip* ∈ *ip_addresses_end_devices* **do**
13: sendArpRequest(*ip*)
14: **end for**
15: *discoveredLinks* ← (all links learned through *odl-l2switch-hosttracker*)
16: updateNetworkGraph(*discoveredLinks*)

17: **function** updateNetworkGraph(*linksList*)
18: **for each** *link* ∈ *linksList* **do**
19: **if** *link* is not already added **then**
20: *srcNode* ← *getSourceNode*(*link*)
21: *dstNode* ← *getDestinationNode*(*link*)
22: *networkGraph* ← *addVertex*(*srcNode*)
23: *networkGraph* ← *addVertex*(*dstNode*)
24: *networkGraph* ← *addEdge*(*srcNode*, *dstNode*, *link*)
25: **end if**
26: **end for**
27: **end function**

switch-node of the topology to redirect all LLDP traffic to the SDN controller. Then, on lines 6 to 9, the OFDP procedure begins the explained process. Thus, with the information collected by the OFDP procedure, a *networkGraph* is built on line 10, which contains the switches and the links between switches from the network topology.

Subsequently on line 11, when the *core-network* has been discovered, the SDN controller will proceed to calculate a BFS tree on the network topology to avoid loops and install rules on the *switch-nodes* to propagate the ARP broadcast traffic on the calculated BFS tree. This procedure is outlined on Section 5.3.

Later, lines 12 to 15, we send ARP requests from the SDN controller to all edge devices registered by the power substation operator. This action will allow the feature *odl-l2switch-hosttracker* of the L2Switch project detecting and registering each *edge-node*, and its corresponding link to the *core-network*.

Finally, the *networkGraph* structure is updated with the new information about the *edge-nodes* to get a complete view of the network topology state (line 16). Lines 17 to 27 illustrate the building pro-

cess of the network graph *networkGraph* by using the Java Universal Network/Graph (JUNG) Framework [O'M10].

5.3 BROADCAST TRAFFIC CONTROL IN LOOP-BASED TOPOLOGIES

To mitigate the broadcast storm problem related to the broadcast protocols in loop-based topologies, networking engineers traditionally implement solutions like STP which introduces associated difficulties: redundant links will be disabled to all types traffic even knowing that it is only necessary to block the broadcast traffic. Instead, since the SDN controller is fully aware of network topology, we can take advantage of this knowledge to detect loops and use programmable techniques to handle the broadcast traffic.

In particular, we present an SDN solution to handle the ARP broadcast traffic, without blocking other traffic types, building an exclusive tree graph to manage the ARP traffic since the SDN controller has a centralized view of the communication network. However, this case use can be extended to handle other IPv4 broadcast protocols such as: DHCP, SMB, etc.

This application uses the Breadth First Search (BFS) Algorithm [Zus48], taking the node located in the center of a spiderweb topology as root. BFS is an algorithm for traversing or searching elements in a graph, often used on trees. It starts selecting a node as tree root and exploring all the neighbors of this node. Then for each neighbor discovered explores their respective adjacent neighbors, and so on until the whole tree is traveled. The proposed structure for this application is showed in Algorithm 5.2.

At the beginning, we build a network graph with the topology information contained in the SDN controller. Then, on lines 2 to 6, we determine the root node checking the connectivity degree of each switch (node) in the topology graph. Our interest is to start at the center of the topology, and that node is the only one inside the topology with a connectivity degree greater than three. Here, it is important to mention that, according to the S3N architecture, we will always have a switch for the Management sector, another one for the Peripherical communications sector and the remaining ones for each bay. In this context, the root node will always have a connectivity degree greater than three.

Lines 7 to 20 implement the BFS algorithm to obtain the links belonging to the BFS tree, *linksBFSTree*. With this information, on line 21, we obtain the links that do not belong to the BFS tree, *linksToBlockArp*, subtracting them from all links of spiderweb stored in *linksBFSTree*.

Finally, lines 22 to 29 are in charge of applying flows to the switches through some SDN southbound (OpenFlow, OVSDB). In short, when an ARP frame of type REQUEST arrives to any switch, the switch sends the frame out all interfaces, not including the incoming interface

Algorithm 5.2 ARP Manager in an SDN context (SpiderWeb Topology)

Input: Network Graph $G(N, L)$ where N are switches and L links
Output: OpenFlow rules set to install in the switches

```
1: Initialize variables:
   rootNode⟨N⟩, vertex⟨N⟩, neighbor⟨N⟩, queueNodes⟨Queue of N⟩, foundNodes⟨List of N⟩,
   visitedNodes⟨List of N⟩, linksBFSTree⟨List of L⟩, linksToBlockArp⟨List of L⟩
2: for each switch ∈ G(N, L) do
3:     if switch degree is greater than 3 then
4:         rootTree ← switch
5:     end if
6: end for
7: queueNodes ← enqueue rootNode
8: foundNodes ← rootNode
9: while queueNodes is not empty do
10:    vertex ← front of queueNodes
11:    for each (neighbor of vertex) ∈ G(N, L) do
12:        if neighbor is not in foundNodes AND neighbor is not in visitedNodes then
13:            linksBFSTree ← link(vertex, neighbor)
14:            queueNodes ← enqueue neighbor
15:            foundNodes ← neighbor
16:        end if
17:    end for
18:    visitedNodes ← vertex
19:    queueNodes ← dequeue
20: end while
21: linksToBlockArp ← (all links L ∈ G − linksBFSTree)
22: for each switch in G do
23:    Add flow priority=5, dl_type=arp_request, actions=flood
24:    for each switchPort in switch do
25:        if switchPort belong to linksToBlockArp then
26:            Add flow priority=10, dl_type=arp, in_port=switchPort, actions=drop
27:        end if
28:    end for
29: end for
```

(FLOOD action). In contrast, when an ARP frame of any type arrives to a switch interface belonging to the source or destination link of *linksToBlockArp*, that frame will be dropped independently of the ARP type (REQUEST, REPLY, etc). Figure 5.2 shows, step by step, the operation scheme of the presented algorithm.

In addition, the algorithm behavior was evaluated in two failure conditions: 1) turning off one link belonging to the BFS tree and 2) disabling the central switch of the topology. In both scenarios the network topology connectivity was guaranteed, and the flows were installed properly according to the conditions defined in the algorithm. The results for these failure conditions are shown in the Figure 5.3.

5.4 SV MULTICAST TRAFFIC MANAGEMENT

SV communication service is a protocol used to transmit analog values (current and voltage) from the MUs to any IEDs in the power substation. This service uses a publisher/subscriber mechanism where the publisher (MU) sends information to one, or more subscribers simultaneously (IEDs), taking advantage of the multicast functionality

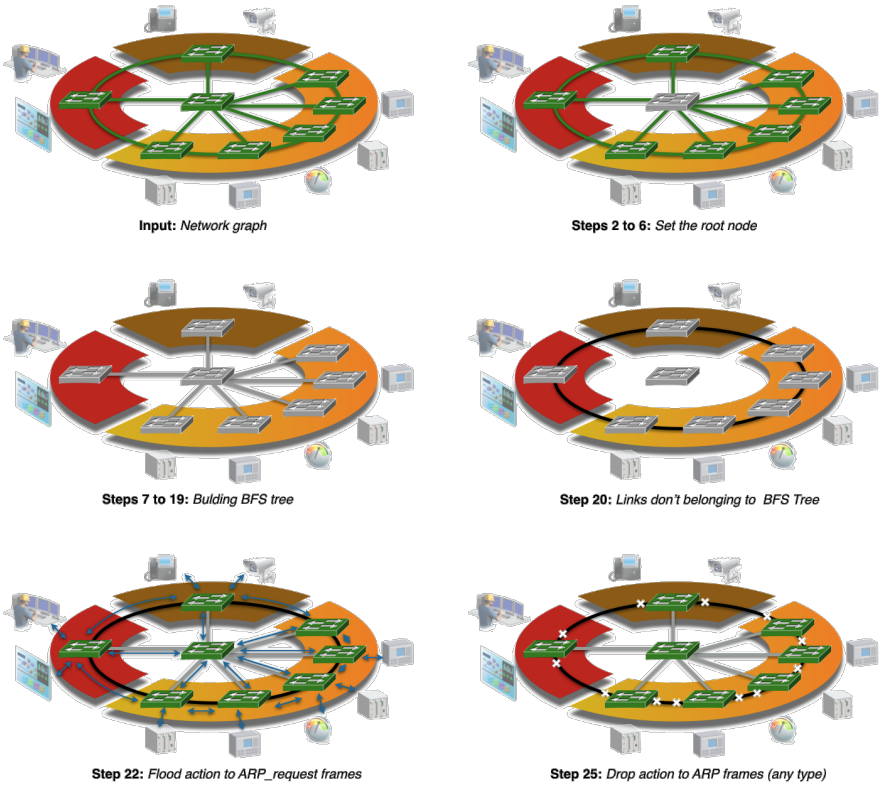


Figure 5.2: Operation scheme to manage ARP traffic

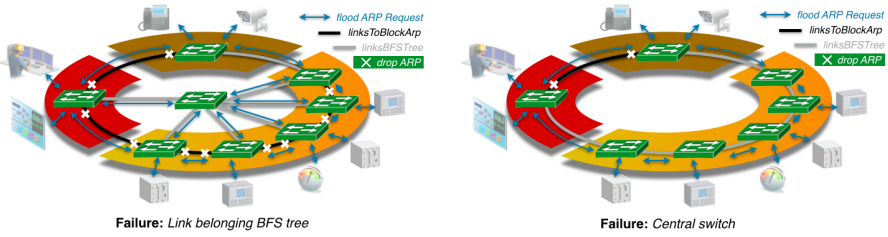


Figure 5.3: Algorithm behavior with failure conditions

provided by Ethernet. A publisher (MU) always is sending SV messages, which are available to any device that requires them (IEDs) by subscribing the corresponding publisher. Also, in the SV communication service there is not retransmission protocol, a lost sample is overwritten by the next successful one.

In traditional networks, a switch handles the multicast traffic in the same way that it deals with broadcast traffic. That means, the multicast frames flood all switch interfaces, except for the interface where the frame arrived. But this behavior is unacceptable in a critical network. To mitigate the issue described, network administrators implement solutions based on VLAN or IGMP technologies to handle any multicast traffic efficiently. However, the right set-up of multicast traffic is hard since the administrator has to configure the network

devices one by one and a misconfiguration can cause erratic operations. In this context, there is a new opportunity to illustrate the benefits of operating under SDN environment developing customized solutions. Here, we get the most out of graph theory to develop an SV multicast traffic management module. This module takes advantage of the fact that the SDN controller has an entire view from the communication network and the knowledge of the publishers and subscribers along with their relationships (which IEDs are subscribed to a specific MU).

In particular, we present an SDN solution to calculate SV graph trees for the adequate propagation of the SV multicast traffic. In this way, it is guaranteed that this traffic does not flood the network and we can have as many SV trees as SV publisher/subscriber groups since each Publisher will have its own propagation tree (svTree). A svTree tree will include: a root node that corresponds to an SV traffic generator device or Publisher (MU); sheets that correspond to IEDs devices or Subscribers; intermediate nodes that correspond to switches and branches that are only the links through which this traffic will be transmitted.

This module uses the Breadth First Search (BFS) Algorithm [Zus48] to explore the svTree calculated, taking the publisher node as root. Then for each intermediate node discovered in the tree (switches), flows rules are applied to guarantee the adequate forwarding of SV traffic. The proposed structure for this application is showed in Algorithm 5.3. This algorithm only illustrates the building of a single SV multicast group, it means one publisher and their corresponding subscribers. Also, the sequence diagram associated with the SV multicast traffic management can be consulted in paragraph 3.3.5.3.

Initially, the module takes as input the network graph $G(N, L)$ built for the execution of the Algorithm 5.2 along with information related to the SV multicast group (publisher, subscribers and SV multicast address).

Then, on lines 2 to 6, we build a tree graph called *svTree* sequentially. First, for each subscriber of the SV multicast group, we find the shortest *path* between the publisher and the respective subscriber. This *path* will be used to get a single tree branch. Later, each branch is added to the *svTree* structure. Here, it is important to mention that each *svPublisher* has associated a single *svTree*. This feature becomes important in the future in the presence of link failures in the topology because, when the failed link is identified, we can easily determine using graphs theory which SV trees were affected and re-calculate only these specific trees. Thus, the system response is optimized. In traditional networks, with the aforementioned behavior, this type of characteristic is unfeasible.

Lines 7 to 28 implement once again the BFS algorithm. But, in this case, we use the algorithm to explore the svTree calculated and not to build a specific tree (see lines 7 to 20 in Algorithm 5.2). In the tree

Algorithm 5.3 SV Manager in an SDN context (SpiderWeb Topology)

Input: Network Graph $G(N, L)$ where N are switches, MUs or IEDs, and L links
svPublisher and his corresponding *svSubscribers* list
sv_multicast_address
Output: Tree Graph $svTree(N, L)$ where N are switches, MUs or IEDs, and L links
OpenFlow rules set to install in the switches

1: Initialize variables:
$svPublisher\langle N\rangle, svSubscribers\langle List\ of\ N\rangle, subscriber\langle N\rangle, path\langle List\ of\ L\rangle, branch\langle G(N,L)\rangle,$
$svTree\langle G(N,L)\rangle, rootNode\langle N\rangle, vertex\langle N\rangle, neighbor\langle N\rangle, queueNodes\langle Queue\ of\ N\rangle,$
$foundNodes\langle List\ of\ N\rangle, visitedNodes\langle List\ of\ N\rangle, link\langle L\rangle, switchPorts\langle List\ of\ ports\rangle$
2: **for each** *subscriber* in *svSubscribers* **do**
3: $path \leftarrow findPath(svPublisher, subscriber)$
4: $branch \leftarrow buildBranch(path)$
5: $svTree \leftarrow addBranch(branch)$
6: **end for**
7: $rootNode \leftarrow findRootTree(svTree)$
8: $queueNodes \leftarrow$ enqueue $rootNode$
9: $foundNodes \leftarrow rootNode$
10: **while** *queueNodes* is not empty **do**
11: $vertex \leftarrow$ front of $queueNodes$
12: **for each** ($neighbor$ of $vertex$) $\in svTree$ **do**
13: **if** $neighbor$ is not in $foundNodes$ AND $neighbor$ is not in $visitedNodes$ **then**
14: **if** $vertex \in$ switches **then**
15: $link \leftarrow getLink(vertex, neighbor)$
16: $switchPorts \leftarrow addSwitchPort(vertex, link)$
17: **end if**
18: $queueNodes \leftarrow$ enqueue $neighbor$
19: $foundNodes \leftarrow neighbor$
20: **end if**
21: **end for**
22: $visitedNodes \leftarrow vertex$
23: $queueNodes \leftarrow$ dequeue
24: **if** $vertex \in$ switches **then**
25: Add flow dl_src=*svPublisher* dl_dst=*sv_mcast_address*, actions=output:*switchPorts*
26: $switchPorts \leftarrow$ clear
27: **end if**
28: **end while**

discovering process, we identify the intermediate nodes (switches) on line 14 and we take note of the switch ports related to the *svTree* branches (lines 15 and 16). This information is important because it will be used on line 25 to apply flows to the switches through some SDN southbound (OpenFlow, OVSDB). On this occasion, we uniquely identify the SV flows using only the Ethernet Media Access Control (MAC) addresses, although it is possible to add another field such as EtherType or the SV message identifier svID, we prefer to make the match as simple as possible. Also, note that the flow action output allows sending the same SV message through different ports at the same time.

Figure 5.4 shows, step by step, the operation scheme of the presented algorithm.

Last, although the Algorithm 5.3 only shows the procedure to build a single *svTree*, the SV multicast traffic management module reuses this algorithm as many *svTree* as required. All SV tree graphs calculated are stored to guarantee a fast recovery process as we mentioned before.

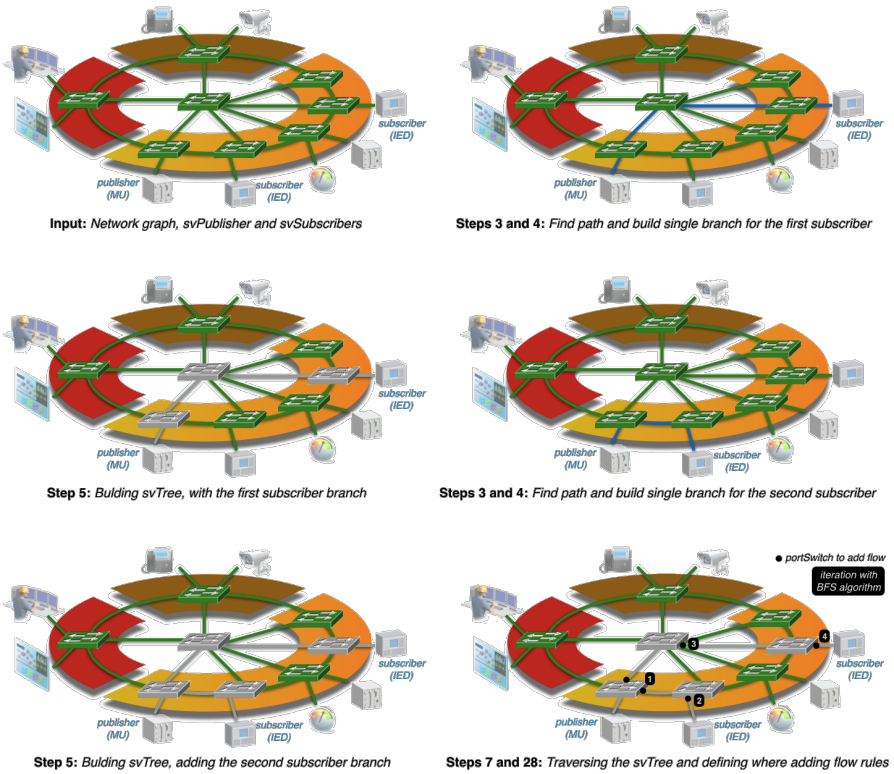


Figure 5.4: Operation scheme to build the *svTree* and handle the SV traffic

5.5 GOOSE MULTICAST TRAFFIC MANAGEMENT

GOOSE communication service is a protocol used with the purpose of distributing event data (commands, alarms, indications, trip messages), between all IEDs across the entire power substation communication network. Particularly, the fast and reliable transmission of GOOSE trip messages is an important feature within the power substation because these messages carry out critical information (for example, the instruction to trigger a breaker and protect the power substation in case of a transient in a high voltage power line). In this context, to guarantee a fast event-driven transmission, GOOSE messages are exchanged at layer 2 (link layer), optimizing the processing times in the sender (encoding) and the receiver (decoding). Like SV service, in GOOSE the information exchange is based on a publisher/subscriber mechanism using the multicast functionality provided by Ethernet. However, since GOOSE messages are multicast, they are not acknowledged by the destination. To overcome this multicast feature, a GOOSE trip message is retransmitted several times in a row to rectify possible frame losses.

As we mentioned before, GOOSE is a critical communication service with high requirements of reliability. To get this feature, traditional networks in power substation use PRP or HSR, two recovery protocols defined by the IEC 62439-3 standard that provide zero recovery time with no packet loss.

The basic concept behind PRP is that a device is connected to two independent networks. Any message this device publishes is mirrored to both networks. Subscribing devices, also connected to both networks, will accept the first version of the message received, and discard the second version. If one network link fails, the mirrored message will still go through on the second network. The two networks don't need to be identical, but they must not be connected to each other (see Figure 5.5) [HP15].

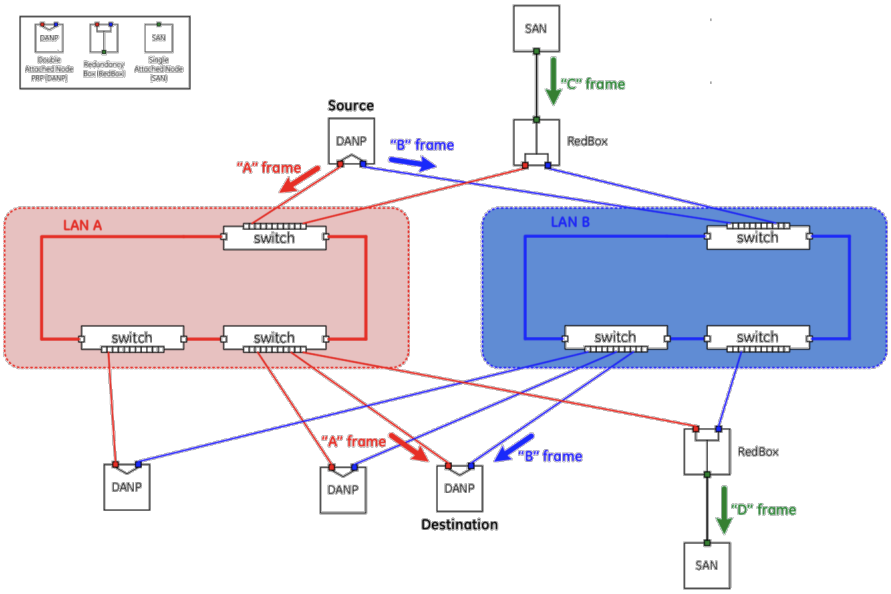


Figure 5.5: PRP concept

The basic concept behind HSR is that all devices are connected in a ring topology, without switches. Any message from the publishing device is duplicated, and sent both directions around the ring. A subscribing device accepts the first version of the message received, and discards the second version. If a network link fails, the version of the message traveling the other direction around the ring will be received and used (see Figure 5.6) [HP15].

Although PRP and HSR provide a high-availability network, they also have weaknesses. PRP requires two completely isolated networks for its operation. It means, a double capital investment, with the addition of the cost of the new devices (DANP and Redbox). This, without mentioning the technical aspects related to the configuration. For example, PRP operates over standard Ethernet switches, therefore also it is necessary to define how to handle the multicast traffic within of each network: VLAN or Internet Group Management Protocol (IGMP). On the other hand, HSR requires specialized LAN nodes for its operation and HSR Ethernet frames are not compatible with standard Ethernet frames. So, there is no compatibility with other Ethernet networks. In addition, the number of nodes connected to an HSR ring is lim-

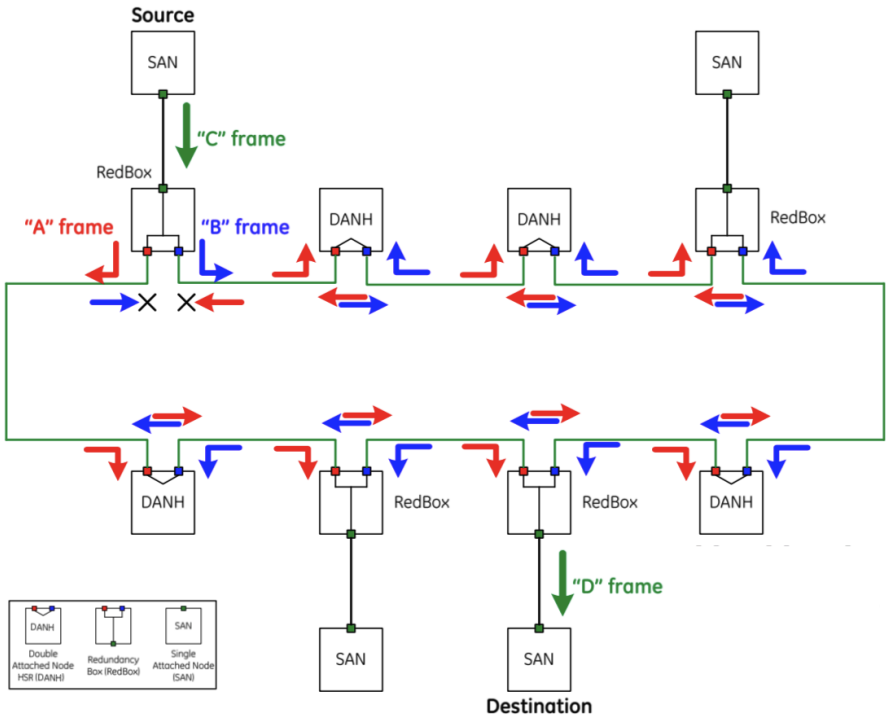


Figure 5.6: HSR concept

ited by the node port that has the least bandwidth. This represents a scalability issue.

In this context, there is a challenge about how to handle GOOSE traffic under an SDN environment with high requirements of reliability (zero recovery time with no packet loss). However, we already advanced in overcoming this matter. In Chapter 4, we define a reliable network topology (the spiderweb), considering criteria such as SDN environment and edge-disjoint paths. Here, we will explain the operating principles of the GOOSE multicast traffic management module under SDN perspective. This module emulates the functionalities of zero recovery time protocols in a single network topology, without the need to use additional devices (Redbox or HSR interface) and with Ethernet networks compatibility. In short, this module is in charge of guaranteeing path redundancy and packet redundancy.

- *Path redundancy.* This functionality allows the discovering of redundant paths between two nodes. In addition, when a path is defined to forward the GOOSE traffic, this implies that there is no indication of flooding process related to multicast traffic. The spiderweb network topology provides three edge-disjoint paths between two end-nodes located in different bays: one path passing through the central switch (path 3) and two paths traversing the ring in opposite directions (paths 1 and 2), see Figure 5.7.

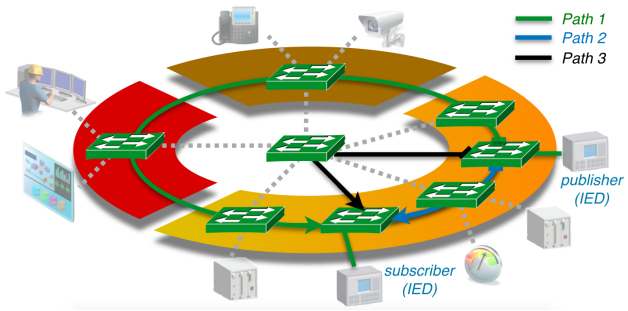


Figure 5.7: Edge-disjoint paths on spiderweb network topology

- *Packet redundancy.* This feature allows sending duplicate information through two edge-disjoint shortest paths.

In particular, we present an SDN solution to calculate GOOSE directed graphs for the adequate propagation of the GOOSE multicast traffic. Our approach computes two edge-disjoint shortest paths in the spiderweb topology and sends duplicate information through them for each publisher-subscriber pair. This is possible since the SDN controller has an entire view from the communication network and it also has information about the publishers and subscribers relationships (which IEDs are subscribed to a specific IED publisher).

This module uses a modified version of Breadth First Search (BFS) algorithm to explore the *directedGraph* calculated, taking the publisher node as the first node to start the procedure. Then for each intermediate node discovered in the directed graph (switches), flows rules are applied to guarantee the adequate forwarding of GOOSE traffic. The structure proposed for this application is shown in Algorithm 5.4. This algorithm only illustrates the building of a single GOOSE multicast group, it means one publisher and their corresponding subscribers. Also, the sequence diagram associated with the GOOSE multicast traffic management can be consulted in paragraph 3.3.5.3.

At the beginning, like SV multicast traffic management module, the GOOSE module takes the network graph $G(N, L)$ built for the execution of the Algorithm 5.2 along with information related to the GOOSE multicast group (publisher, subscribers and GOOSE multicast address).

Then, on lines 2 to 16, we build a directed graph called *directedGraph* merging tree directed graphs according to several criteria. 1) If the publisher and the subscriber are connected to the same node (switch), the tree corresponds with the simple path between the publisher and the subscriber. 2) If the publisher and the subscriber are located on different bays we merge two tree graphs to the *directedGraph*, one tree passing through the central switch and another tree traversing the shortest path through the ring. In a similar way, just as the SV multicast traffic management, each *goosePublisher* has associated a directed graph. This feature contributes to optimize the system response since,

Algorithm 5.4 GOOSE Manager in an SDN context (SpiderWeb Topology)

Input: Network Graph $G(N, L)$ where N are switches, MUs or IEDs, and L links
goosePublisher and his corresponding *gooseSubscribers* list
goose_mcast_address
Output: Directed Graph $directedGraph(N, L)$ where N are switches, MUs or IEDs, and L links
OpenFlow rules set to install in the switches

1: Initialize variables:
$publisher\langle N\rangle, gooseSubscribers\langle List\ of\ N\rangle, subscriber\langle N\rangle, path\langle List\ of\ L\rangle, centerPath\langle List\ of\ L\rangle,$
$ringPath\langle List\ of\ L\rangle, branch\langle G(N,L)\rangle, centerBranch\langle G(N,L)\rangle, ringBranch\langle G(N,L)\rangle,$
$directedGraph\langle G(N,L)\rangle, predecessorNode\langle N\rangle, successorNode\langle N\rangle, queueNodes\langle Queue\ of\ N\rangle,$
$foundNodes\langle List\ of\ N\rangle, visitedNodes\langle List\ of\ N\rangle, link\langle L\rangle, switchPorts\langle List\ of\ ports\rangle$
2: $publisher \leftarrow goosePublisher$
3: **for each** *subscriber* in *gooseSubscribers* **do**
4: **if** (*publisher* AND *subscriber*) are connected to the same switch **then**
5: $path \leftarrow findPath(publisher, subscriber)$
6: $branch \leftarrow buildBranch(path)$
7: $directedGraph \leftarrow mergeBranch(branch)$
8: **else**
9: $centerPath \leftarrow findPathThroughCentralNode(publisher, subscriber)$
10: $centerBranch \leftarrow buildBranch(centerPath)$
11: $directedGraph \leftarrow mergeBranch(centerBranch)$
12: $ringPath \leftarrow findShortestPathThroughRing(publisher, subscriber)$
13: $ringBranch \leftarrow buildBranch(ringPath)$
14: $directedGraph \leftarrow mergeBranch(ringBranch)$
15: **end if**
16: **end for**
17: $queueNodes \leftarrow$ enqueue *publisher*
18: $foundNodes \leftarrow publisher$
19: **while** *queueNodes* is not empty **do**
20: $predecessorNode \leftarrow$ front of *queueNodes*
21: **for each** (*successorNode* of *predecessorNode*) $\in$ *directedGraph* **do**
22: **if** *predecessorNode* $\in$ switches **then**
23: $link \leftarrow getDirectedLink(predecessorNode, successorNode)$
24: $switchPorts \leftarrow addSwitchPort(predecessorNode, link)$
25: **end if**
26: **if** *successorNode* is not in ($foundNodes$ AND *visitedNodes*) **then**
27: $queueNodes \leftarrow$ enqueue *successorNode*
28: $foundNodes \leftarrow successorNode$
29: **end if**
30: **end for**
31: $visitedNodes \leftarrow predecessorNode$
32: $queueNodes \leftarrow$ dequeue
33: **if** *predecessorNode* $\in$ switches **then**
34: Add flow dl_src=*publisher* dl_dst=*goose_mcast_address*, actions=output:*switchPorts*
35: $switchPorts \leftarrow$ clear
36: **end if**
37: **end while**

in case of link failures, the broken link could be identified and, using graph theory, we can easily determine which GOOSE directed graphs were affected and re-calculate only these specific structures. Figure 5.8 shows snapshots in the building of the GOOSE *directedGraph*.

Lines 17 to 37 implement a modified version on the BFS algorithm to traverse the *directedGraph* related with a GOOSE group (publisher and their corresponding subscribers). In this directed graph, *successorNode* is a reachable node from *predecessorNode* while *predecessorNode* is the node that discovers a *successorNode*. Like in Algorithm 5.3, we identify

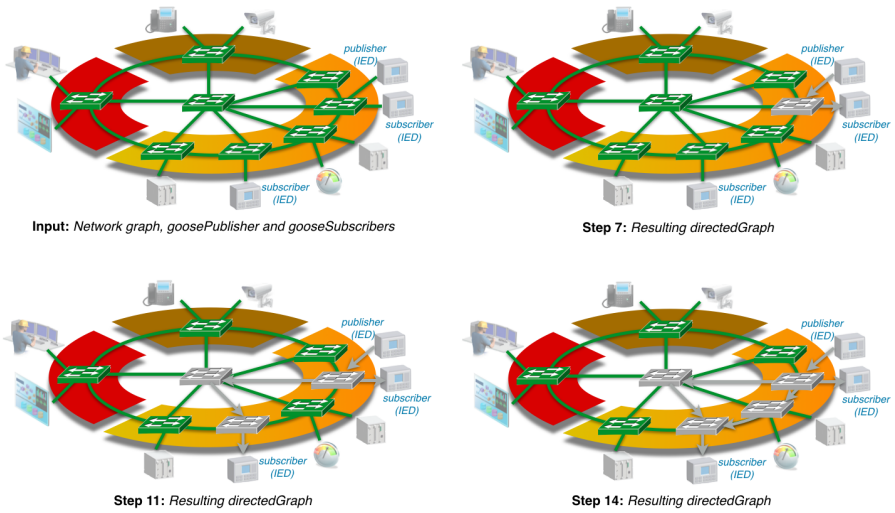


Figure 5.8: Snapshots in the building of the GOOSE *directedGraph*

the intermediate nodes (switches) on line 22 and store in *switchPorts* the ports used to reach the successor nodes (lines 23 and 24). This information will be used on line 34 to set rule flows to the switches through some SDN southbound interface (OpenFlow, OVSDB). Once again, we uniquely identify the GOOSE flows using only the Ethernet MAC addresses, though it is possible to add another field such as EtherType or the application identifier appID. Also, note that the flow action output allows sending the same GOOSE message through different ports at the same time.

Finally, it is important to mention that we are sending duplicate information through different paths with the purpose of guaranteeing zero recovery time. This means that the GOOSE subscribers are receiving GOOSE messages duplicates all the time. Ideally, the switch that connects the GOOSE subscriber should eliminate the duplicate GOOSE messages. However, the SDN environment does not allow interacting with the kernel-switch to set a discard function. Therefore, it will be a task of the GOOSE subscriber to detect and delete the duplicated messages. So, to reach this functionality, we modify the code associated with the GOOSE subscriber from the libIEC61850 library [Zil16].

5.6 CONCLUSION

This chapter shows how SDN applications can solve complex issues, in loop-based topologies, such as broadcast traffic control, path redundancy, packet redundancy, and multicast traffic management; which would be technically unfeasible using common network protocols. In particular, we develop applications to define the traffic behavior in the network topology proposed in Chapter 4, according to the operational requirements of the power substation communications network. The

achieved results, when the algorithms were applied over our testbed, were consequent with the expected operation scheme of the communication network. This demonstrates that in an SDN context there is not only one way to solve the problems. Now, any network engineer is able to develop new solutions (named protocols or applications), according to the specific needs of his/her organization. In addition, we verify the behaviour of the Algorithms 5.1, 5.2, and 5.3 related with the tasks of network topology discovery, ARP broadcast traffic control and SV multicast traffic management, getting identical results over a Dual rendundant tree network topology (see Figure 4.1).

6 SECURITY TOPICS

Security is always excessive until it's not enough.
— Robbie Sinclair

Network security is in charge of preventing information theft as well as to guarantee private communications and service availability. Therefore, it has become an inherent network feature as important as the connectivity itself. In this chapter we consider the problem of detecting intrusions associated with the reconnaissance phase of an attack, along with mitigation techniques, in a communication network running under the SDN environment. Also, this chapter presents a new approach for the recognition of anomalous traffic conditions in communications networks of power substations, through the use of hierarchical clustering algorithms and statistical type descriptors (averages). This contribution was submitted to [Lea18] and published in [LBJ18a] as a collaboration of Alexánder Leal and Prof. Dr. Juan Felipe Botero from the Telecommunications Engineering Department of the University of Antioquia with Eduardo Jacob from the Telecommunications Engineering Department of the University of the Basque Country.

6.1 IMPROVING EARLY ATTACK DETECTION IN NETWORKS WITH SFLOW AND SDN

Network monitoring is a paramount aspect for the detection of abnormal and malicious activity. However, this feature must go hand by hand with mitigation techniques. We take advantage of this to improve the network security using the sFlow monitoring tool along with an SDN controller. sFlow will be in charge of detecting network anomalies defined by user rules, while the SDN technology is responsible to mitigate the intrusion. This technology provides a network architecture in which the network control plane is decoupled from the data plane. This fact enables the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services [Ope18]. On SDN environments, control techniques may be easily developed as a result of its network programmability.

The SDN environment provides several alternatives to implement network anomalies detection: 1) Put the entire responsibility on the SDN controller, providing it with mechanisms for traffic pattern analysis. To achieve this purpose, it will be necessary for each switch to install flow rules to redirect all ingress traffic from edge devices to the

controller. However, this behavior overloads the centralized control plane, forcing the SDN controller to deteriorate its performance over other tasks [Gio+14], and overloading the link to the controller; 2) Incorporate a Network Intrusion Detection System (NIDS). This option implies the implementation of port mirroring mechanisms in the switches to forward the traffic from edge devices to the NIDS, even incorporating an overlay network. Even so, mirroring the whole network traffic to another device may result in significant resource consumption, especially in high traffic environments [GAM14]; 3) Use of OpenFlow flow statistics counters. In this scenario OpenFlow is responsible of communicating control messages to the switches and monitoring traffic as well. This scheme does not come without major consequences. The number of flow-entries in the flow table of the edge switch may grow extensively (counted in tenths of thousands of flows per time-window), thus affecting the switching performance [Gio+14].

In this context, although alternatives mentioned above offer good results, they register scalability issues when traffic increases. In order to provide an alternative solution to the detection and mitigation of network intrusions, we take advantage of packet sampling capability of sFlow [PL04] along with the SDN technology. In our approach, sFlow will be in charge of detecting network anomalies defined by user rules, while the SDN controller will be responsible of mitigating the intrusion.

6.1.1 *Related Work*

In this section, we examine several approaches that propose solutions around network security field with sFlow under an SDN approach.

In [Gio+14], authors implement a system that can detect Distributed Denial of Service (DDoS), Worm propagation and Portscan attacks using an Entropy-based algorithm. Moreover, they present experimental results that demonstrate the effectiveness of the sFlow mechanism compared with the use of OpenFlow flow statistics counters, in terms of overhead imposed on the usage of system resources. The evaluation of this proposal is carried out in a NOX Controller [Gud+08] and, according to the traffic condition analyzed, they use a hardware-based switch NEC-IP8800/S3640 or a software-based OpenvSwitch [Nic09] hosted on Dual-Core 3 GHz with 8GB RAM server.

Mitigation of SYN Flooding Attack under a DDoS scenario is proposed in [Nug+14]. The methodology used to detect the attack relies on monitoring a threshold value of the sum cumulative of TCP-SYN packets reported by each sFlow switch agent. The evaluation of this technique was implemented on Mininet [Tea12] using four OpenvSwitchs [Nic09] and controlled by a Floodlight controller [Big12].

An information security defense mechanism (ISDM) was deployed to perform anomaly detection, mitigation and reduce the loss caused by the DDoS attack [HCP16]. ISDM uses attack signatures to provide

security on Internet of Things environments (smart living appliances). The experimental platform employs OpenvSwitches hosted on a Raspberry Pi 2 using a Ryu Controller [Ryu].

In [ZKM14], OrchSec, an architecture oriented to detect and mitigate ARP Spoofing / Cache Poising, DoS / DDoS was developed. OrchSec functionalities were validated utilizing Mininet [Tea12] and Floodlight [Big12] / POX [Pox] controllers.

In [GAM14], authors propose a modular and scalable architecture to enhance the Remote Triggered Black-Hole functionality, a routing approach towards DDoS attack mitigation. To test their solution, they implement a POX Controller [Pox], a software-switch (OpenvSwitch) [Nic09] and a software-router (Vyatta Core), each of them hosted on individual Dual-Core 3 GHz with 8 GB RAM server.

An SDN controller can be a victim of a DDoS attack too. In [Dha+15], authors study how to improve the controller security in this scenario. The method not only considers the malicious packet to detect a DDoS attack, but it also takes into account the time properties of DDoS attack such as duration and time to detection, in order to prevent the future attack. To demonstrate the operation of their method, they use the Opendaylight controller [Odl] and Mininet [Tea12].

FlowTrApp [BM16] provides an architecture for DDoS attack detection and mitigation in Data Centers. The proposed mechanism first matches an incoming flow with a legitimate sample of traffic and then installs mitigation actions (e.g. a flow is found not lying in the bounds of legitimate traffic pattern i.e., flow rate and flow duration). FlowTrApp was tested using Mininet emulator [Tea12] and the Floodlight controller [Big12].

Unlike the approaches mentioned in this section, which are mostly oriented to detect and mitigate DDoS attacks, our proposed work aims to detect and control an attack in its reconnaissance phase.

6.1.2 *sFlow*

sFlow is the short way for expressing "sampled flow" ... where sampling in networking implies to achieve scalability.

sFlow is a sampling technology for monitoring and collecting data from a wide range of equipment: physical and virtual network devices (switches and routers), physical and virtual hosts (FreeBSD, Linux, Windows; Containers; Hyper-V, KVM, and Xen hypervisors) and software applications (Apache/PHP, JAVA) [PL04]. A couple of major aspects of sFlow take advantage of the statistical properties of packet sampling to produce statistically quantifiable measurements, and having its own RFC 3176 specification [PMP01]. Also, sFlow provides REST and JavaScript API making easy to configure customized measurements, retrieve metrics, set thresholds, and receive notifications. These features transform sFlow in an open-source network tool with a huge capacity to interact with other components: SDN controllers, Orchestrators or DevOps (see Figure 6.1).

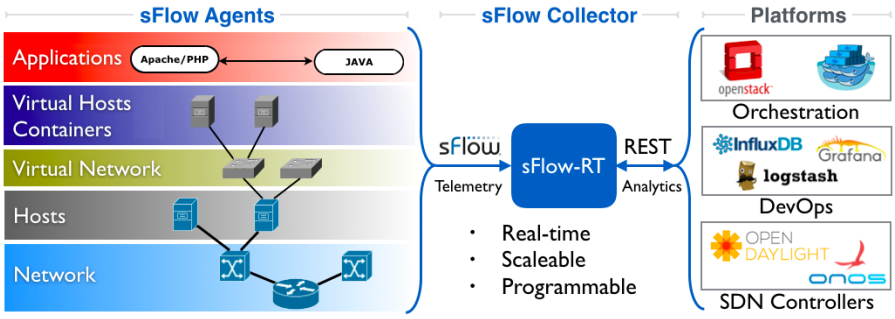


Figure 6.1: sFlow-RT Environment

6.1.2.1 *sFlow components & Operation Scheme*

A sFlow based monitoring system has three elements (see Figure 6.2):

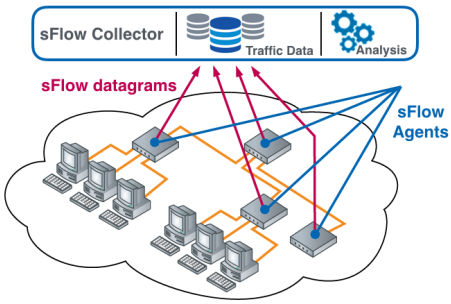


Figure 6.2: sFlow Components

- *sFlow Collector:* A software application, running on a workstation or server, that collects and analyses traffic data sent to it by the sFlow agents in the network. Through sFlow collector, the information collected can be analyzed and presented to network administrators in a variety of ways, such as traffic rates charts, dashboards, and thresholds events [All13].

- *sFlow Agent:* On the network devices context, an agent corresponds to a switch or router that gathers information about traffic on its interfaces and sends it to the sFlow collector. sFlow Agents operate by regularly polling interface statistics registers (counters) such as: [In/Out]UcastPkts, [In/Out]MulticastPkts, [In/Out]BroadcastPkts, [In/Out]Errors; and packet sampling information if the packet sampling is also configured on the agent. Currently, a wide network equipment support sFlow Agents: Alcatel, Cisco, Dell, HP, Huawei, Juniper, OpenvSwitch, Pica8, among others [PL04; All13].

- *sFlow datagrams:* datagrams sent by the sFlow agent to the host where the sFlow Collector through UDP port 6343 is running. They include statistics and packet sampling information [PL04].

6.1.2.2 *sFlow Applications*

SFlow applications can be developed externally using the sFlow-RT Representational State Transfer (REST) API or embedded using sFlow-RT's internal JavaScript API. Our proposal is focused on the last option. Detailed information about how to write a sFlow-RT application can be found on [Sfl].

Typically, applications implement a number of the following steps. Nevertheless, there are other functions to improve the applications.

- *Define flows:* Flows have to be specified in order to detect and mark accordingly the traffic of interest. A flow is defined using a name, keys (usually Ethernet, IP, TCP or UDP headers), value (bytes, frames) and optionally filter attributes. When the flow definition is done, it is possible to identify a set of packets that share common attributes within a period of time.

- *Handle flow records:* It allows the user to register a flow handler to be notified of each new matched flow, according to the flows definition. Flows will only be logged if log:true is specified in the flow specification. In this work, we use that option to generate alarm messages in the system.

- *Define Threshold:* Thresholds are applied to the flows defined above and they can be used to generate a notification when the rate value associated with a flow exceeds the threshold. We use this option to identify ARP network discovery, TCP-SYN service discovery and Deny of Service intrusions.

- *Handle threshold events:* Register an event handler function to be notified of each new event.

We list below a code snippet to identify a DoS attack using sFlow-RT's internal JavaScript API (see Listing. 6.1).

```
// Flow Definition
setFlow('dos_attack',{keys:'ipdestination,stack',value:'frames',filter:'link:outputifindex=null'});
// Threshold setting
setThreshold('dos_threshold',{'metric':'dos_attack',value:10000,'byFlow':true,'timeout':1});
// Event Handler
setEventHandler(function(evt) {
  switch(evt.thresholdID) {
    case 'dos_threshold':
      let ddosKeys = {};
      ddosKeys[evt.flowKey] = evt.value;
      logWarning("DDoS Alert: ", JSON.stringify(ddosKeys));
      break; } },['dos_threshold']);
```

Listing 6.1: Code snippet to identify a DoS attack using sFlow

The *setFlow()* function creates a flow called *dos_attack* that captures the destination IP addresses and calculates the *frames/second* rate for each flow, specifically in the switches interfaces where there are hosts attached *(filter:'link:outputifindex=null')*. In this definition, a special flow key *stack* was used. This key captures the layers decoded from the packet (For example, *eth.ip.udp, eth.arp* among others).

The *setThreshold()* function defines a threshold to trigger an event when the flows related to the flow definition *dos_attack* exceed 10,000 *frames/second*. The *byFlow* flag indicates that a threshold event should be generated for each individual flow crossing the threshold.

The *setEventHandler()* function is triggered when a threshold is reached. In this function, we validate first that the generate event *evt* corresponds with our threshold of interest *dos_threshold*, and if so, we extract the destination IP address and the stack information from the key fields related to the flow definition *dos_attack*.

6.1.3 *Architecture proposal*

This work is oriented to detect and control the reconnaissance process -the initial step of any intrusion operation. For this purpose, we implemented an architecture composed of two domains called Monitoring and Countermeasure (see Figure 6.3).

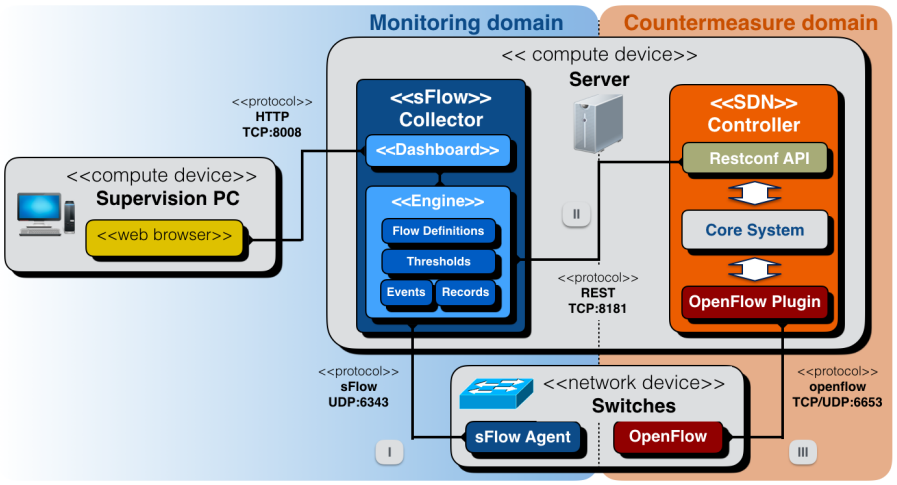


Figure 6.3: Architecture proposal

- *Monitoring domain*, in charge of detecting the initial steps of an attack. This domain contains three elements: 1) the sFlow agent who resides inside the network device with sFlow support, 2) the sFlow Collector located in the server and 3) a supervision element to visualize the behavior of the network and the alert messages (supervision PC).

- *Countermeasure domain*, responsible of mitigating the intrusion. This area is governed by an SDN environment with its respective control plane (controller) and data plane (switches). These planes interact using a southbound protocol (e.g., Open Flow [McK+08]).

6.1.3.1 *Operation*

When the sFlow agent is enabled in the OpenvSwitch, sample packets' header is encapsulated along with metadata to be sent to the sFlow Collector (Step I in Figure 6.3). The metadata includes sampling ratio, a timestamp at the time of capture, switch ID, and forwarding information such as the input and output port numbers. Here, it is important to mention that the rate of samples sFlow produces is not constant; this depends on the rate of packet arrivals.

Then, the real-time analytics engine of the sFlow Collector processes the gathered information. If any sampled packet's header matches with some flow definition in the monitoring script, this sample is processed to determine if an event should be triggered. In that case, a REST request is sent to the other component of our architecture, the SDN controller, responsible for the mitigation of anomalous operations (Step II in Figure 6.3). That request contains the information needed to neutralize the attack (switch ID, port, and MAC address). So, at this point, the controller is able of making adjustments to the network employing a southbound protocol. It means, to add a DROP flow action in the forwarding device table to block the intruder (Step III in Figure 6.3). This flow-entry has higher priority than any other flow in the flow table and its life-time is 30 seconds.

6.1.4 *Performance evaluation*

6.1.4.1 *Environment*

The machine used for this testbed runs Ubuntu 16.04 LTS, with an Intel Core i7 @3.40GHz x 8 CPU, and 16GiB of RAM. To generate network topologies we use Mininet [Tea12], a network emulator used in the SDN research field which includes OpenvSwitch [Nic09] (a virtual switch that supports OpenFlow and the sFlow agent). The control plane of the testbed is governed by the Opendaylight Controller [Odl] and the collecting process is done by the sFlow-RT collector [Sfl].

The testbed topology consists of a tree topology where each virtual switch is connected to the Opendaylight Controller and the sFlow-RT collector, which are hosted on the same server. At the beginning, the testbed implements a scenario with a tree topology of depth 2 and fanout 2 (Figure 6.4a). Later, to verify if the approach is scalable, we modify the parameters of the topology in this manner: depth 3 and fanout 4 (Figure 6.4b).

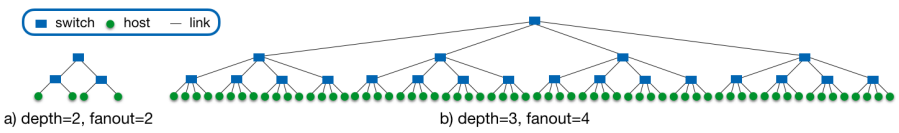


Figure 6.4: Testbed topologies

6.1.4.2 *Setup*

- *Mininet:* For generating network topologies we run the command *sudo mn –custom sflowtestbed.py –topo tree,depth=3,fanout=4 –link tc,bw=100 –controller remote,ip=127.0.0.1:6633*. The script file *sflowtestbed.py* includes several functions to execute over mininet topology, such as:

  - To enable sFlow agent on each OpenvSwitch that belongs to the topology and set the sampling and polling parameters. In this testbed the sampling rate was defined: 1:100. A sampling rate of 1:100 means that among every 100 packets captured by sFlow agent, only the header information related to one will be sent to the collector.
  - To post a JSON representation of the Mininet topology to sFlow collector using REST API.
  - To generate pseudo-randomly ICMP , TCP and UDP background traffic. ICMP traffic is generated at random intervals between all hosts for the first scenario (Figure 6.4a), and between 8 hosts groups in the other case (Figure 6.4b). TCP and UDP flows are established among all hosts, in pairs, using Iperf [Ipe]. We take advantage of the Iperf command to set randomly the port [1024-65535], the duration [50-100 secs] and the bandwidth [1-10 Mbps] parameters within the aforementioned ranges. We define whether the flow will be TCP or UDP by the port value (odd or even).
  - To control the execution of the intrusion process.

  Finally, in all scenarios the link bandwidth was set to 100Mbps *(tc,bw=100)*, to guarantee a reasonable workload of the local system resources when all hosts are transmitting background traffic.

- *OpenDaylight Controller:* the controller is configured to discover the topology and to set rules via OpenFlow with the purpose of guaranteeing basic connectivity between all stations. It means, managing broadcast (ARP) and unicast (IP, TCP, UDP) traffic. We assign a low priority to the flows in charge of forwarding tasks, thus the controller could use high priorities to install the flows to mitigate the intrusion and guarantee the primacy of these rules over the others.

- *sFlow Collector:* sFlow-RT manages several script files in order to process and visualize information. In this testbed, we define JavaScript functions to indicate to the sFlow-RT real-time analytics engine how to detect reconnaissance attacks discussed in the following section. Moreover, we built a dashboard interface

to visualize the metrics of interest, set thresholds, and receive notifications. Also, a file with all IP addresses was created to implement a control access list.

6.1.4.3 *The reconnaissance phase in an attack*

Intrusion in networks takes many forms including denial of service, man-in-the-middle, viruses propagation, etc. Typically, in an intrusion situation, the intruder attempts to gain access to a particular resource (data or host). However, any kind of intrusion is preceded by a reconnaissance process. Reconnaissance is the unauthorized discovery and mapping of systems, services, or vulnerabilities [UP13].

In the initial stage of an intrusion, the intruder eavesdrops the network for the auto-assigning of an IP address in the compromised network segment. For this step, we build an access control list with all IP addresses allowed in our system. If the IP source address of any packet sampled does not match with that list, an event will be triggered. This access control list can be built by the network administrator or in automated way with the information collected from the topology by the controller. Also, an eavesdropping process could be detected polling periodically hosts to detect Network Interface Card (NIC) in promiscuous mode [San01]. This task could be programmed in the controller. But, since our testbed performs over a emulation platform, this functionality was not implemented.

Next, to determine which hosts are available, the intruder will start a network discovery process. In our work, we implement a countermeasure for hosts discovery based on an ARP Request sweep. To detect this process we define a rule to count the ARP Request frames by second. If the frames' number exceeds a predefined threshold an event is generated.

Later, the intruder scans TCP and/or UDP ports in the discovered hosts to determine what network services are available. At this point, a TCP-SYN scan technique is used for this purpose. For this type of intrusion we define two rules. For TCP scan, we count the number of TCP-SYN segments sent in one second from the same IP address. If the number of segments exceeds a predefined threshold, an event is generated. For UDP scan, we detect any ICMP unreachable messages in the network. This type of messages is produced when the intruder tries to access UDP ports that do not have any service configured.

Subsequently, the intruder queries the ports to determine the application type and version, even the type and version of the operating system. Based on this information, the intruder can determine whether a possible vulnerability exists that can be exploited. And with this, the reconnaissance process ends.

In order to evaluate the effectiveness of our approach we replicate each reconnaissance step as follows:

- To emulate the moment when the attacker allocated an IP address inside the network segment, we just set the IP address of a random machine outside of the range defined in the access control list.
- Network discovery process could be done through the NMAP tool [Lyo97b] with the modifiers *"-sP -PR [target IP address]"*. This instruction causes an ARP Request Scan on the network segment defined by the user.
- TCP and UDP service discovery processes are achieved using the NMAP tool too. For the TCP case, we use the modifier *"-sS [IP address target]"*. This command triggers a port-scan process using TCP-SYN segments. For the UDP case we use the modifier *"-p [ports] -sU [IP address target]"*.
- In order to emulate a DoS attack to conclude the intrusion process, we used the tool NPING (a NMAP complement) with the following sintaxis *"nping –udp –source-port 53 –data-length 1400 –rate 2000 –count 200000 –no-capture –quiet [IP address target]"*.

6.1.5 *Validation*

To visualize and analyze the behavior of our system, a dashboard was designed on sFlow-RT (*http://IP_sflow_collector:8008*). There, see Figure 6.5, we present three charts in order to identify the reconnaissance process suggested in Section 5.3: hosts discovery with ARP scan, service discovery with TCP-SYN scan and a DoS attack; on a scenario with a tree topology of depth 2 and fanout 2 (Figure 6.4a).

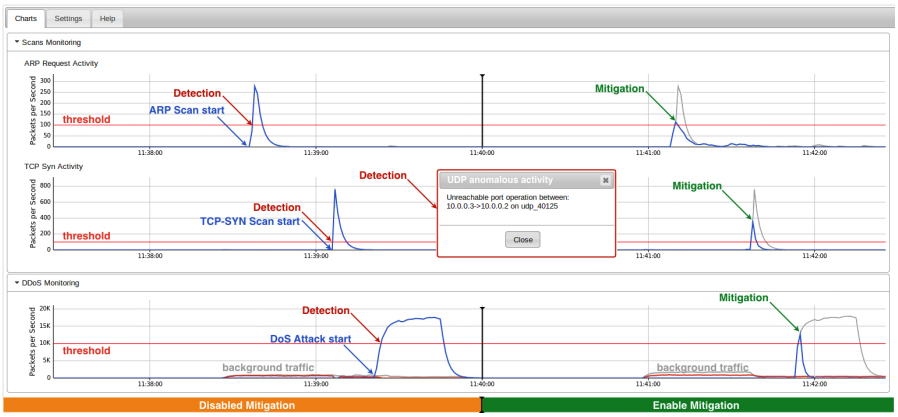


Figure 6.5: Reconnaissance Stage

ARP activity chart, TCP-SYN activity chart, and DoS monitoring chart represent the number of ARP Request frames, TCP-SYN segments, and IP packets gathered for all sFlow agents on every second. Moreover,

threshold values for each chart can be changed by the network manager through Settings tab.

On the left side of the Figure 6.5 (disabled control), we can observe how the network discovery process (timeline 11:38:00 to 11:39:00), the exploration of possible services (timeline 11:39:00 to 11:39:20) and a denial of service attack (timeline 11:39:20 to 11:40:00) were performed. The following list details the log console of the sFlow collector for this part of the procedure (see Listing. 6.2).

```
11:38:00 INFO: Listening, sFlow port 6343
11:38:01 INFO: Listening, HTTP port 8008
11:38:01 INFO: Scripts s3n_stats.js and s3n_alarms.js started
11:38:37 WARNING: ARP_SCAN Alert {MAC:Frames} ->  {"DA5C3216AE0B":103.26693339459928} , on device ->  {"port"
      :"1","dpid":"0000000000000003"}
11:39:05 WARNING: TCPSYN_SCAN Alert {srcIP,dstIP:Frames} ->  {"10.0.0.1,10.0.0.4":100.4977565214356} , on
      device ->  {"port":"1","dpid":"0000000000000002"}
11:39:23 WARNING: DoS Alert {srcIP,dstIP,protocol:Frames} ->  {"10.0.0.2,10.0.0.3,eth.ip.udp.dns"
      :10001.445090767767} , on device ->  {"port":"2","dpid":"0000000000000002"}
11:39:36 WARNING: Unreachable port operation between: 10.0.0.3->10.0.0.2 on udp_40125
```

Listing 6.2: Alert messages on sFlow console log

On the right part of the graph, the same reconnaissance process is executed with the mitigation condition enabled. To improve the evidence about the mitigation process achieved, traffic shapes in gray (which are replicas of the traffic behavior on the left), were added. Finally, the dialog box shown in the figure, it is a consequence of the DoS attack performed since in the testbed there is no service running on UDP port 53 (DNS service), hence ICMP unreachable messages are generated as evidence of an anomalous network condition.

To evidence the scalability of sampling technology such as sFlow, the same detection procedure described above was implemented on a scenario with a tree topology of depth 3 and fanout 4 (Figure 6.4b). The results are presented in the Figure 6.6 using a sampling rate of 1:100 (left side) and 1:10 (right side), in order to verify the behavior of the intrusion process under these conditions. The reader can verify that all parts of the reconnaissance process were detected effectively, regardless of the network size or the sampling rate defined.

Figure 6.6: Sampling Comparison

6.1.6 *Discussion*

The tests carried out show that sFlow is a scalable sampling technology for measuring, collecting, and analyzing network traffic data. This fact enables the technology for different purposes, in our case the capacity to detect a reconnaissance process in order to improve the network security. From the obtained outcomes, we can conclude that:

- The process of host discovering, through the use of the ARP Request scan technique, is a process that is carried out in a highly efficient way. The required time for this procedure depends directly on the size of the network to be scanned, but generally, hundreds of hosts can be scanned in few seconds. As it is a brief type of intrusion, it does not generate any type of stress within the network, hence its detection becomes less important. However, as we mentioned before, this procedure precedes attacks of another magnitude.

- The exploration of available services is a procedure that will depend on the number of hosts to be scanned and the number of ports of interest. In the tests performed, the identification of services was oriented to a single host in the network, where the TCP-SYN process was focused on only 2000 specific ports (NMAP default operation mode). Hence, the action registered in the chart was very short. However, if this task is executed simultaneously for all discovered stations, the outcome can significantly disturb the behavior of network traffic, even emulating the presence of a DoS attack.

- Regarding the DoS attack, the mitigation technique acquires greater prominence when compared with the aforementioned discovery processes, which have a shorter action in the timeline. However, it is important to mention that when a control is implemented in a discovery processes, we do not only prevent the network flooding with ARP or TCP-SYN packets —on the contrary, we have identified the location of the intruder (switch and associated port) and his network identifier (MAC or IP address) and, therefore, all traffic originating from that device has been blocked, which will cancel the advance of the other steps in the intrusion process—.

- With respect to the values of sampling rate, 1:10 and 1:100, they were defined around of the recommendations offered by sFlow for the large flow detection (1:10 for links speed of 10Mbps, 1:100 for 100Mbps and 1:1000 for 1Gbps) [Pha]. The sampling rate values affected the resolution of the traffic shape, but it did not influence the detection process in the testbed.

- Finally, for the two scenarios proposed: the first one integrated by 3 switches, 4 hosts and 8 ports-witch, and the second one composed of 21 switches, 64 hosts and 104 ports-switch; the achieved results were identical.

6.1.7 *Conclusion*

In this section, we evaluated the applicability of using sampling technology, under SDN environment, to detect and mitigate reconnaissance anomalous activity, which is the initial step of any kind of attack. For this purpose, we proposed an architecture and built a testbed to demonstrate the effectiveness of our approach even in large scenarios. Results show that our prototype can effectively detect and control the anomalous activities described in Section 6.1.4.3, and can be extended to identify other ones. Also, we illustrated a system that can be easily replicated in other scenarios with similar needs.

6.2 HIERARCHICAL CLUSTERING FOR ANOMALOUS TRAFFIC CONDITIONS DETECTION IN POWER SUBSTATIONS

Although the communication networks of modern electrical substations, based on the IEC61850 standard [Tc5], provide major benefits than the communication networks of traditional substations, companies are being cautious with their implementation due to the vulnerabilities evidenced through various research articles [Ras+14]. For example, in [Cho+12] and [Pre+10] Denial of Service (DoS) attacks were implemented. Also in [Pre+10] network traffic was intercepted (sniffing). The interception and modification of critical traffic (tampering) was reached in [HDB12] and [HLG14], while a spoofing attack was achieved in [Kus+14].

In this context, the anomalies or intrusion detections within power substation communications networks has become an important research topic, as a consequence of the serious damage that a failure may cause in this critical infrastructure. The majority of intrusion detection systems are focused on the detection of signatures (characteristic pattern associated with a particular intrusion or attack). However, for obvious reasons, this type of detection does not allow the detection of new types of attacks [CMA05]. Hence, the study of unsupervised classification techniques that can allow, through the wide recognition of the normal traffic of the network, the identification of possible abnormal states of operation is of special interest.

This section focuses on exploring the applicability of hierarchical clustering algorithms in the identification of anomalous operation scenarios, specifically, in power substations communication networks based on the IEC 61850 standard.

6.2.1 *Hierarchical clustering algorithms*

A clustering algorithm is a multivariate statistical procedure aiming to group, or classify, the elements of a data space into a compact, separate and homogeneous groups called clusters or classes. In particular, the unsupervised clustering algorithms aim to discover the composition of the classes or groupings to which the elements can belong without having apriori information about the structure of the data. This clustering must guarantee that the degree of natural association is high among members of the same group and low among members of different groups [Gal15]. The unsupervised clustering algorithms are divided into two major categories: hierarchical and partitional. The partitional algorithms divide the data space into a specified number of groups, following an optimization criterion. While the hierarchical algorithms generate a structured organization of nested groups, which is represented by a classification tree known as a dendrogram (see Figure 6.7). The dendrogram illustrates how the algorithm groups the elements step by step and, observing the structure of their branches and the distance among them, the diagram shows the degree of similarity between the different clusters. In addition, depending on where the cut level of the dendrogram is established, the number of classes for the classification algorithm is defined [Gal15].

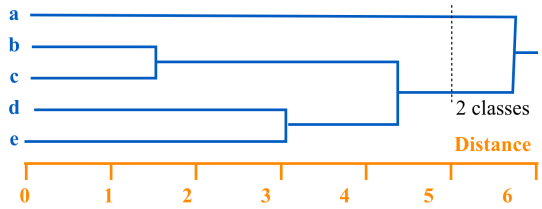


Figure 6.7: Dendogram for a space of 5 elements (a, b, c, d, e)

The techniques of hierarchical clustering are classified into two categories: based on agglomeration and based on division (see Figure 6.8). Agglomerative algorithms, or *bottom up* approach, start the analysis with as many groups as there are elements in the data space. From these initial units, groups are formed ascending, until at the end of the process all treated cases are within in a single set. With an opposite approach, the division-based algorithms, also called *top down*, begin with a set that encompasses all observations, and from this initial cluster, through successive divisions, smaller and smaller groups are formed. At the end of the process, there are as many groupings as cases have been treated.

The operation scheme of the agglomerative hierarchical algorithms, classification mechanism used in our approach, is simple (See algorithm 6.1). However, for its execution, it is necessary to define apriori: 1) what are the measures of association that will allow measuring

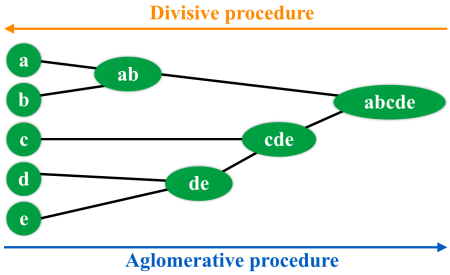


Figure 6.8: Process types in the hierarchical clustering

the proximity of individuals (distance/similarity) and 2) how it can determine when two clusters or classes can be grouped [Gal15].

Algorithm 6.1 Agglomerative hierarchical algorithm

1: Calculate the *distance matrix*
2: Each element is defined as a class
3: **repeat**
4: Group the two closest classes
5: Update *distance matrix*
6: **until** Get a single cluster

There are different metrics to determine the proximity of the individuals to be classified, considering their qualities. For example, if the characteristics of individuals are quantitative, a measure of distance will be used as an indicator of proximity. On the contrary, if the attributes of the individuals are qualitative, a similarity index will be used as proximity metric. Among the most used distances are Euclidean distance, Manhattan distance, Minkowski distance, Pearson correlation, cosine vector, among others. The distance used in this approach is the Euclidean distance (Equation 6.1), as a consequence of the achieved results (Section 6.2.2.5).

$$D^2(x_i, x_j) = (x_{1i} - x_{1j})^2 + (x_{2i} - x_{2j})^2 + ... + (x_{ki} - x_{kj})^2 \qquad (6.1)$$

Having defined the proximity measure (Euclidean distance), it is necessary to define the criteria for identifying which are the closest classes to its corresponding grouping. In the agglomerative hierarchical clustering, different mechanisms are distinguished to achieve this objective. These include Minimum or single-linkage clustering, Maximum or complete-linkage clustering, Mean or average-linkage clustering and Centroid linkage clustering. Figure 6.9 shows the clustering criteria used in these techniques. For example, in single-linkage, the clusters are joined considering the smallest of the distances between the closest members of different groups; while, in complete-linkage, the clusters come together considering the smallest of the distances between the more distant members of different groups. In the technique of the average-linkage, the clusters are united considering the lowest average distance between all the pairs of elements of both sets [Gal15].

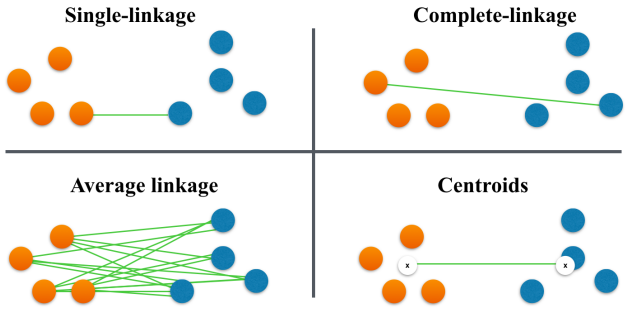


Figure 6.9: Criteria for grouping in the hierarchical clustering

6.2.2 *Methodology*

To illustrate the effectiveness of the hierarchical algorithms as a mechanism for the classification of the operation scenarios in power substation communications, a test scenario was designed and implemented in an isolated and controlled environment. The description of the implemented testbed, the defined operating scenarios, the descriptors used and the classification process carried out are discussed below.

6.2.2.1 *Data capture*

To capture the network traffic, a prototype of a test communications network (testbed) was implemented in an isolated and controlled environment (see Figure 6.10). This network topology was composed of a generic interconnection device (Ethernet switch); two IEDs of reference ABB REM630 and ABB REG620, operating in the modes described in Figure 6.10; a PC for the registration and monitoring of the events transmitted through the IEC 61850 standard and the capture of network traffic through the WIRESHARK application [Com+07]; and a PC (attacker) to execute intrusions such as: the network topology discovering through the NMAP [Lyo97a], the execution of a DoS attack using HPING3 [Sano5] and the fabrication of an spoofing attack for Goose messages sent by the publisher through the use of the OSTINATO application [Sri10]. The behavior of the high voltage line was emulated by the ISA DRTS66 test equipment.

On the communication network described in Figure 6.10, six different operating scenarios were defined: 1) Normal traffic, 2) IED disconnection, 3) Network discovery attack, 4) DoS attack, 5) IED spoofing attack and 6) Failure on the high voltage line. The operating conditions of the described scenarios were generated sequentially and captured in the equipment called Registration and monitoring; to get a capture of 20 minutes of traffic with 226.000 frames (PCAP file).

6.2.2.2 *Identification of Descriptors*

In this stage, a specific set of characteristics or attributes must be defined, so that each element of the data space is represented by a

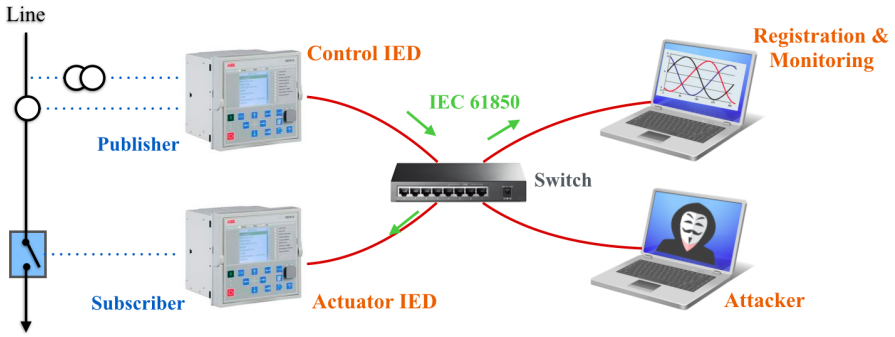


Figure 6.10: Test communications network (testbed)

collection of descriptors. These descriptors will allow identifying the features that affect the classification problem. This initial choice reflects the researcher's opinion about the purpose of their classification [Gal15].

In the field of detection of network traffic patterns through unsupervised classification mechanisms, the studies use two types of attributes as illustrated in Table 6.1. 1) Attributes based on the network flow, it means, on the use of the value of a communication protocol field (IP, UDP, TCP) or MAC address. 2) Statistical attributes, such as the average of a particular type of packets, the distribution function that the parameters of the packet follow, among others.

Our study identified that four descriptors are enough for an adequate classification of the proposed operation scenarios, since we get identical results using different classification algorithms (Section 6.2.2.7). Table 6.2 shows the descriptors used, three of statistical type (*n_frames, n_goose, n_arp*) and one based on the network flow (*goose_seqnum*).

6.2.2.3 *Data pre-processing*

Once the descriptors that will characterize the elements of our data space were identified, the traffic capture file (PCAP) was processed in order to obtain this set of elements. For this purpose, we developed a script in the LUA programming language [IDFF96] to be executed into the TSHARK application [Com12]. In this way, it was possible to get a set of 110 elements. Each element, with four descriptors, shows the behavior of the network traffic in a time window of 10 seconds (see Figure 6.11).

6.2.2.4 *Exploratory Data Analysis*

Figure 6.12 shows the behavior of the normalized descriptors along the data space obtained, for each of the described operation scenarios.

TITLE	ALGORITHM	DESCRIPTORS
Learning rules and clusters for anomaly detection in network traffic [CMA05]	Outliers detection with k-NN	Probabilistic. P (W | U) where U = SrcIp=128.1.2.3, DestIp=128.4.5.6 and W = DestPort=80
Traffic anomaly detection using k-means clustering [MLC07]	K-means	protocol type, source IP address, destination IP address, source port, destination port
P2P traffic identification and optimization using fuzzy c-means clustering [LL11]	Fuzzy c-means	NumberofPacketsSentfoaFlow, NumberofPacketsReceived-foraFlow, DurationoftheFlow, Protocol, SourcePort, DestinationPort TotalNumberofPackets,MeanPacketLength, MeanPayloadLlength, MeanPacketInterarrivalTime, AverageSentPacketSize, AverageReceivedPacketSize, Variances, ByteRatio's
CoCoSpot: Clustering and Recognizing Botnet Command and Control Channels using Traffic Analysis [DRP13]	Hierarchical clustering	transport layer protocol l4p (TCP or UDP), source IP address sip, destination IP address dip, port destination dp
PeerShark: flow-clustering and conversation generation for malicious peer-to-peer traffic identification [NHV14]	X-means (K-means that does not require knowing apriori the number of classes)	Src. IP, Dest. IP, Src. port, Dest. port, Proto (TCP or UDP), Protocol, Packets per second (f/w), Packets per second (b/w), Avg. Payload size (f/w), and Avg. Payload size (b/w), with 'f/w' and 'b/w' signifying the forward and the backward direction of the flow, respectively

Table 6.1: Descriptors used in the recognition of traffic patterns through the use of non-supervised algorithms

DESCRIPTOR	IDENTIFIED SITUATION
n_frames, *average of the total number of frames captured in the time window (10 seconds)*	DoS attack. This attack, independent of the service to attack, generates a huge amount of traffic on the network in a very short period of time.
n_goose, *average of GOOSE packets captured in the time window (10 seconds)*	IED Publisher disconnection or failure in the high voltage line. When an IED is disconnected, the average of GOOSE packets in the time window goes to zero. Similarly, when there is a fault in the high voltage line, the average of GOOSE messages increases as a consequence of the event.
n_arp, *average of ARP packets captured in the time window (10 seconds)*	Execution of a network discovery by an intruder. Most network discovery attacks use the ARP protocol operation scheme as a strategy to discover the stations connected to the network.
goose_seqnum, *SeqNum field of the GOOSE packet header*	Spoofing attack of an IED Publisher. One of the evidences of this attack is the anomalous change of the SeqNum field values in the GOOSE header. These values are registered in sequence, therefore a value out of order implies an intrusion. This descriptor will take the value of one if there is an anomalous change in this field. Otherwise, its value will be zero.

Table 6.2: Descriptors used in this study

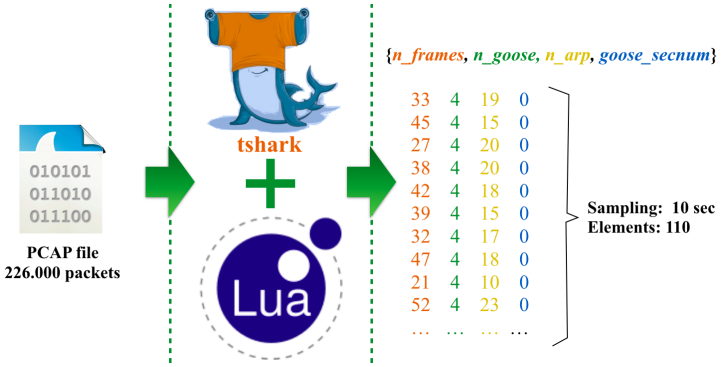
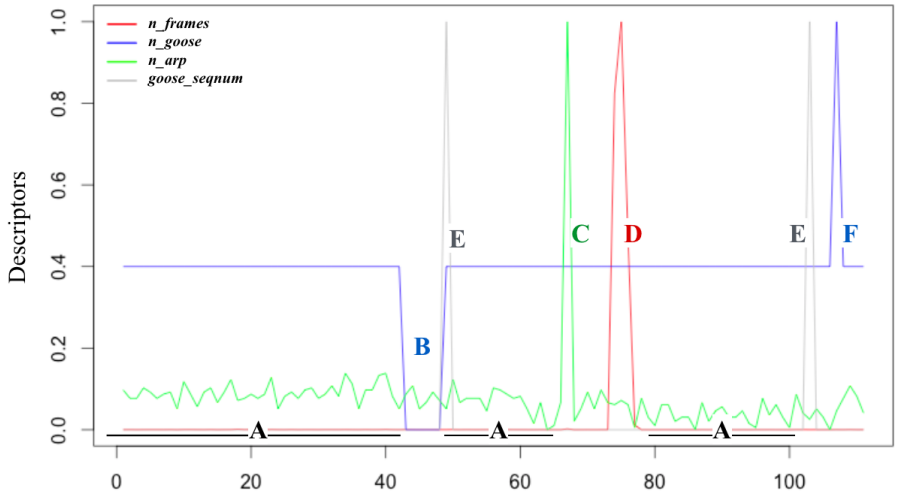


Figure 6.11: Preprocessing data scheme



A. Normal traffic
B. IED disconnection
C. Network discovery attack
E. DoS attack
F. IED spoofing attack
G. Failure on the high voltage line

Figure 6.12: Behavior of the normalized descriptors

6.2.2.5 *Classification process*

In this stage, the clustering algorithm is responsible of assigning to each element of the data space, a category or class (set of elements that share certain characteristics, which also allow differentiating them from the rest). The classification process of this study was carried out using of the *hclust* function of the software for statistical analysis *R* [Tea04]. Although there is no single criterion to determine which measure of association is the most appropriate to measure the proximity of individuals (distances/similarity), and which is the most convenient mechanism for grouping classes, it is recommended to test and compare the results with different methods. Here, we opted to experiment with Euclidean distance and Gower distance as proximity metrics [Gow66], along with single-linkage, complete-linkage, and average-linkage techniques as strategies for grouping classes. From the tests carried out, the best classification scheme was achieved using

single-linkage and Euclidean distance. This combination allowed identifying the six operation scenarios described through six classes while the other schemes required at least 7 classes to correctly identify the six scenarios. Next, Figure 6.13 and Figure 6.14 illustrate the structure of the dendrogram and the classification process obtained according to the behavior of the descriptors.

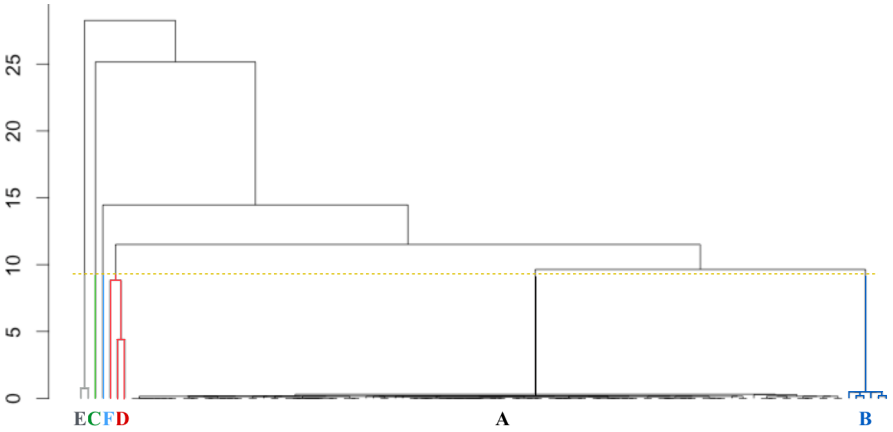


Figure 6.13: Dendogram with a cut of 6 classes

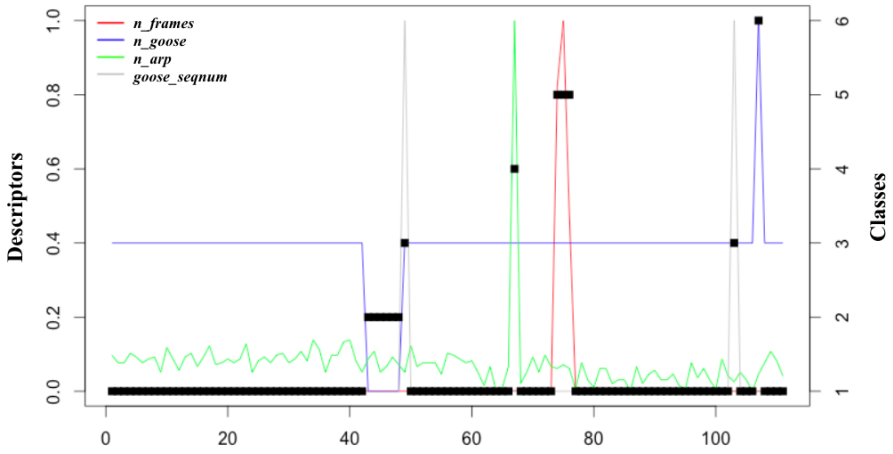


Figure 6.14: Assignment of classes with agglomerative hierarchical algorithm

6.2.2.6 *Analysis of results*

The tests carried out show that the Gower distance and the Euclidean distance presented similar base structures, but different clustering structure. In the same way, despite that the class clustering scheme was changed (simple, complete and average), at the base of the dendrogram it was always possible to identify each one of the proposed scenarios, what changes in the structure is the way they clustered. What becomes clear is that by fixing a cut-off point of the dendrogram to six or seven

classes, it was possible to identify all the defined operating scenarios. However, the best result was reached using Euclidean distance with single-linkage.

Analyzing the structure of the dendrogram (see Figure 6.13), it shows that all operating scenarios can be clearly recognized. However, we expected that all the clusters related to anomalous operation scenarios were grouped in a dominant single class of failure (failure root class), it means to get the normal traffic class *(A)* completely separated of failure classes *(B, C, D, E, F)*.

6.2.2.7 *Validation*

The results reached through an agglomerative hierarchical algorithm motivate a very qualitative interpretation, which can be subjective from the researcher perspective. Hence, it is recommended to compare the achieved results with other types of solutions, where similar results will indicate the presence of a structure in the data. Thus, we explored other solution strategies using partitional and diffuse unsupervised algorithms: K-means [HW79] and LAMDA (Learning Algorithm Multivariable and Data Analysis) [AMDM82]. The results obtained using K-means (see Figure 6.15), defining in advance the *k parameter* equal six (number of operation scenarios), shows how this algorithm identifies all the proposed scenarios.

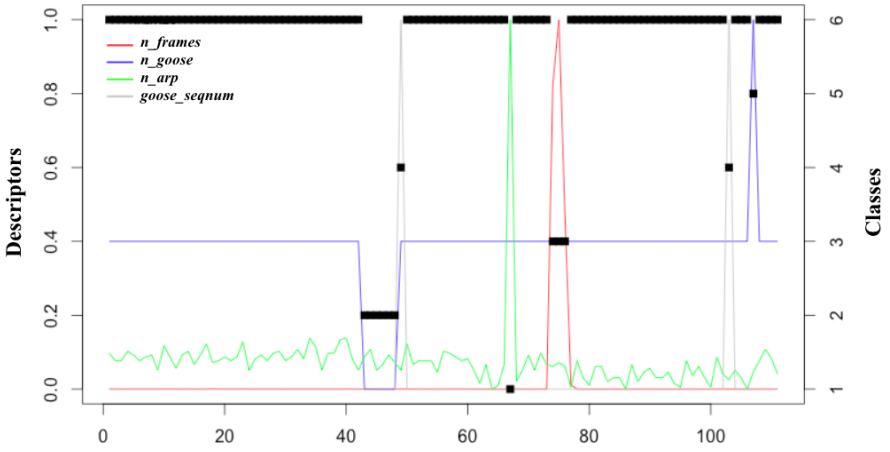


Figure 6.15: Assignment of classes with K-means partitional algorithm

Figure 6.16 illustrates the classification achieved results using the LAMDA algorithm with a Gaussian adaptation function, fuzzy logic connectors Min-Max and a requirement level of 0,6. LAMDA algorithm is incorporated in the P3S application (DISCO Group, LAAS-CNRS). This application also allows extracting the membership graphic associated with each of the classes, Global Adequacy Degree (GAD), see Figure 6.17. The results obtained were in line with the K-means classification.

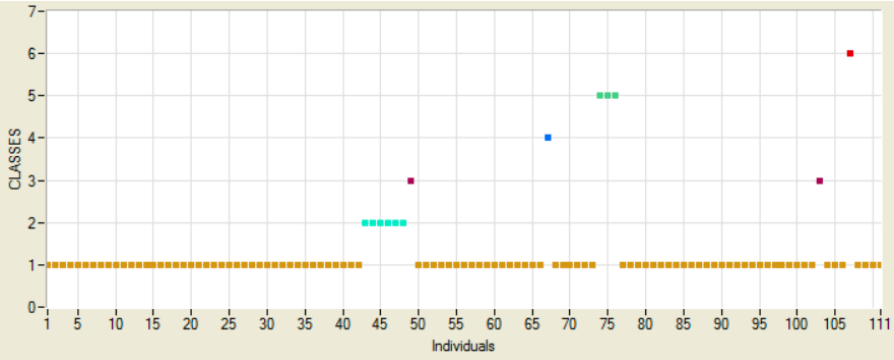


Figure 6.16: Assignment of classes with diffuse LAMDA algorithm

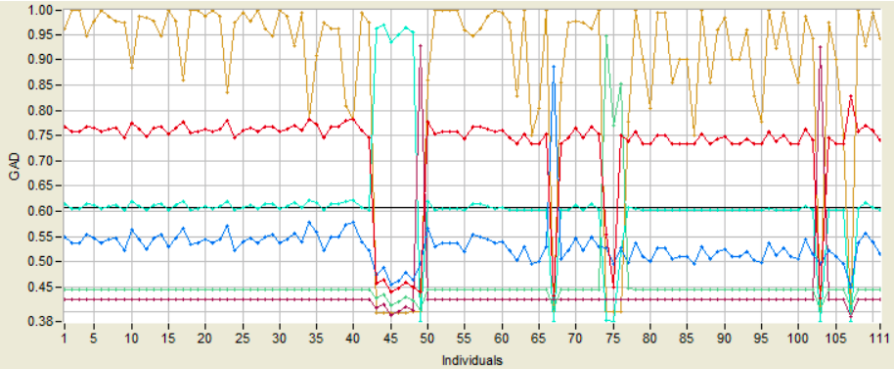


Figure 6.17: Membership graph for classes in the data space

6.2.3 *Discussion*

The selection of descriptors is key in a classification process, therefore, a preliminary analysis to determine their level of contribution is necessary. Likewise, a transformation or pre-treatment of the data may be required, as occurred in this work, where three statistical descriptors (averages) and one network flow descriptor (SeqNum field in GOOSE packet header) were used.

The hierarchical clustering strategy allows, through the dendrogram, to make a preliminary exploration of the possible grouping structures present in the data space, when the number of descriptors is greater than 3. In this way, the techniques of hierarchical clustering are an excellent tool to deal with completely unknown data.

The results evidence the strength in the mechanisms of unsupervised classification to identify all the proposed operating scenarios by using different techniques (partitional, hierarchical and diffuse). Also, the results demonstrate that these algorithms can be useful in several scenarios, for example, the traffic classification in power substation communication networks.

Finally, the fact of getting identical results through different classification algorithms demonstrates the strength of the selected descriptors for the identification of patterns in this particular case of application.

6.2.4 *Conclusion*

In this section, we have presented a practical case of how unsupervised clustering algorithms can be used as an effective tool for the identification of operation scenarios, in power substations communication networks based on the IEC 61850 standard. However, there are still numerous application fields to explore in this area. Particularly the detection of new anomalies, or unknown operation scenarios, is a difficult task for a classification algorithm. In our approach, the selection of the descriptors was successful, given the knowledge of the operating scenarios in advance. They proved to be robust obtaining identical results with other unsupervised clustering techniques such as K-means (partitional-type clustering), or LAMDA (diffuse-type clustering). The challenge then is to ensure that the clustering algorithm is able to classify the normal traffic scenario in a robust manner, in that way the other scenarios will be used to notify anomalous processes in the communications network.

7 QUALITY OF SERVICE

You do not treat all people in the same way because each person is different, with the communication services happen exactly the same thing.

— Alexánder Leal

Even though the QoS concept is widely known, in most cases the network traffic continues being treated in a classic way, with the Best Effort paradigm. One reason for this point is that QoS implementation is a complex task. In this chapter we consider the problem of providing QoS to critical infrastructure networks such as power substation communication networks, which require a careful management of the traffic due to their reliability requirements. The main contributions on this chapter are: 1) the proposal of an architecture to deploy QoS under the SDN paradigm and 2) its validation through the behavior of the protocol with the most demanding operational requirements in terms of delay, GOOSE. These contributions were published in [LBJ18b] as a collaboration of Alexánder Leal and Prof. Dr. Juan Felipe Botero from the Telecommunications Engineering Department of the University of Antioquia with Eduardo Jacob from the Telecommunications Engineering Department of the University of the Basque Country.

7.1 INTRODUCTION

As a result of the modernization of the power substations, the technologies used in traditional data networks are already an integral component of the communication networks of these infrastructures. This upgrade process is known as SAS and it is defined by the IEC 61850 standard [Tc5]. However, although this transformation improves the operation of the substation, new challenges emerged. For example, the arrival of a new set of protocols (SV, GOOSE, MMS, PTP), which demands different transmission times, bandwidth provisioning, reliability and security for its proper operation.

Currently, to implement traffic differentiation and provisioning, the substation engineer uses Class of Service (CoS) to assign priorities to the VLANs in charge of carrying the traffic, according to the network engineering guidelines provided by IEC 61850-90-4 [Int13a]. This strategy usually works in an over-provisioned network, but it does not always guarantee the constraints imposed by the substation traffic. An incorrect configuration of network management protocol, a disturbance recording or a maintenance task can influence the performance of the network. Nevertheless, there are other mechanisms to guarantee

the operational requirements of different protocols, for example, the implementation of QoS policies.

The term QoS presents several approaches according to the context, or even the organization IETF, ITU, and ETSI [GJS03]. In the field of computer networking, QoS refers to traffic prioritization and resource reservation control mechanisms rather than the achieved service quality. QoS is the ability to provide different priority to different applications, users, or data flows to guarantee a certain level of performance [Hua09]. However, to enforce QoS policies is a complex task. The configuration of queuing disciplines requires qualified staff with a wide knowledge about queue parameters, traffic behaviour, device configuration, etc; combined with the fact that if the QoS policy is not configured properly can cause adverse behaviors. For this reason, most networking engineers prefer to over-provision the network capacity as an alternative to deal with the complex QoS control mechanisms. However, this option has scalability problems, especially when the complexity of the systems is increasing, not reducing.

Several works highlight that SDN can improve the management and operation of the power substation communication networks [LB16c]. Therefore, it is relevant to explore how QoS can be deployed under an SDN environment. SDN is a technology that decouples the network control from the forwarding functions, enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services [Ope18]. Thus, SDN can provide an effective solution to ease the deployment of QoS policies because it enables the management and programmability of the entire network from one single point. This section aims at exploiting the benefits of SDN in the implementation of QoS policies inside power substation communication networks.

7.2 RELATED WORK

There are few works around the QoS area under SAS environment, which demonstrates the aforementioned fact: substation engineers usually only use CoS as a mechanism to implement QoS policies. According to this, in [Oli+16] an evaluation about Ethernet switches requirements over IEC 61850 networks was conducted using real case scenarios. Three performance tests were carried out over an Ethernet Switch with 24 Gigabit ports and a switching capacity of 68 Gbps, for each GOOSE, PTP and Sampled Valued protocol. The study concluded that the use of Priority Code Point (PCP) bits in 802.1Q Ethernet frame, CoS mechanism, is enough to ensure a proper operation for these messages even in the worst case situations. There is another work in [Fen+14] where the researchers study the requirements of the traffic present in the digital substation network. That paper proposes a priority classification and a queue scheduling scheme to use with

a CoS strategy, in the same way as is indicated by the IEC 61850-90-4 [Int13a] network engineering guidelines.

However, we can also identify further efforts in this field. For example, in [BHR16] authors discuss how the performance requirements of IEC 61850-based energy applications can be met in Wide Area Network (WAN) context. For this purpose, they propose a QoS extension to the System Configuration description Language (SCL), which has been designed with extensibility features in mind, to enable the direct modeling of communication requirements. This extension includes the well-known TSpec of Integrated Services defined in RFC 2215 and provides QoS assessment to a substation engineer at design time. As a proof of concept, they implemented a prototype that implements the aforementioned extension to guarantee the requirements traffic of a synchrophasor WAN transmission according to TR IEC 61850-90-5.

In [Man17], a researcher proposed a QoS framework for a microgrid communication network, guided by the IEC 61850 standard. The framework executes classification and scheduling tasks. Classification mechanism allows the nodes and agents to identify packets and tag them according to their QoS requirements. Scheduling mechanism -Class Based Weighted Fair Queueing (CBWFQ) with Weighted Random Early Detection (WRED)- determines the next packet to be transmitted. The study was tested on the OMNET simulator.

Also, an SDN approach is presented in [Mol+15b]. There, the authors used port ingress rate limiting strategies to implement QoS policies. They create and configure traffic shaping on Openvswitches [Nic09] thorugh the Open Virtual Switch Data Base (OVSDB) protocol. Then, via the Floodlight SDN Controller, this framework pushes flow rules to redirect specific traffic to different queues. As a result, rates limiting are established taking into consideration the requirements of GOOSE or SV traffic over other services.

Unlike the approaches mentioned in this section, our proposal is not oriented to use CoS techniques. And, even though our architecture implements SDN principles and OVSDB management just like [Mol+15b], the QoS strategy developed and the experimental validation is different. For example, the SDN controllers and the GOOSE/SV traffic emulators implemented are different. In addition, we validate our approach using the most critical communication service in a power substation communication network, GOOSE trip messages, whereas [Mol+15b] only implements a traffic shaping use case with SV, TCP, and UDP traffic.

7.3 CONCEPTUAL FRAMEWORK

7.3.1 *Communications Services on IEC61850 standard*

In general, the IEC 61850 standard [Tc5] defines four types of communication services: MMS, GOOSE, SV and TS; in order to guarantee the correct operation of the power substation network (See Table 2.1). However, the power substation network also supports non-IEC 61850 traffic such as: Supervisory Control And Data Acquisition (SCADA)/Engineering Access, Video surveillance, Thermal monitoring system, Voice Over Internet Protocol (VoIP), among others.

According to the IEC 61850-5 and IEC 61850-8 recommendations, the communication services are mapped into different communication stacks according of their performance requirements (see Figure 7.1).

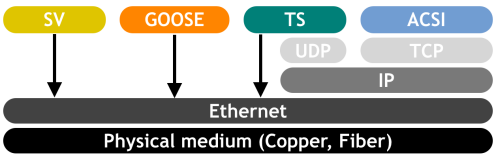


Figure 7.1: Communication stack under the IEC 61850 standard

7.3.2 *Traffic Estimation of the communications services on IEC 61850 standard*

IEC 61850-90-4 is a technical Report focused on engineering a local area network limited to the requirements of IEC 61850-based substation automation.

To implement QoS policies, it is imperative understanding the behavior of the IEC 61850 communication services. Table 7.1 illustrates the main parameters to take into account according to the IEC 61850-90-4 [Int13a] network engineering guidelines, while an additional column is used as reference throughput to be considered in the implementation of QoS policies.

7.3.2.1 *GOOSE*

It transmits messages periodically and generates trip messages when a protection detects a failure. This trip message is transmitted four times (burst) to overcome possible frame losses. Typically, the size of GOOSE messages is within 92 octets to 250 octets and it depends on the configured parameters in the GOOSE Control Block. For example, a trip GOOSE message to handle just one digital status information in the data set (one Boolean value and the related Quality bit string) has an approximate size of 124 octets. To determine the throughput for the GOOSE messages transmitted periodically is enough to multiply the maximum size of the message (250 bytes) and its corresponding rate (for example, 10 frames per second). However, as a GOOSE trip message is not periodic, we use its maximum transmission times (TT6 - 3ms) to estimate the approximate throughput. This time is distributed

Protocol	Comm. Stack	Priority	Function	Transfer Time Class	Packet size	Packets/sec	Throughput
GOOSE	ETH (multicast)	High	Trips, Blocking	TT6 3ms	100-200 bytes	random	3 Mbps
			Releases, Status Changes	TT5 10ms	100-250 bytes	10	20 Kbps
			Fast automatic interations	TT4 20ms	100-250 bytes	10	20 Kbps
SV			General measuring	TT5 10ms	140 bytes	4800	6 Mbps
			Quality and metering accuracy	TT5 10ms	—	15360	12 Mbps
PTP	ETH or UDP/IP/ETH	Medium	TimeSync Events & Commands	—	—	—	3 Mbps
MMS	TCP/IP/ETH		Slow automatic interactions	TT3 100ms	200 bytes	10	16 Kbps
			Operator commands	TT2 500ms	200 bytes	1	1.6 Kbps
			Events, alarms	TT1 1000ms	100-200 bytes	20	32 Kbps
		Low	Files, Log contents	TT0 > 1000ms	50000 bytes >	random	1 Mbps

Table 7.1: QoS Requirements for the communication services defined in the IEC 61850 standard and throughput suggested

as follows: 80 percent to the processing times in the IED stacks (2.4ms) and the remaining 20 percent (0.6ms) for the communication network. According to this, to transmit a trip GOOSE message of 200 bytes with a maximum delay of 0.6ms a throughput of 2.7Mbps is required.

7.3.2.2 *SV*

SV transmits samples values periodically with a rate of 80 samples per cycle transmitted. That means 4000 samples per second (for a 50Hz grid) or 4800 samples per second (for a 60Hz grid). An SV frame has an approximate size of 140 bytes according to the IEC 61850-9-2LE guideline, although this size depends on the number of Application Service Data Units (ASDUs) encapsulated in the SV frame. Assuming an SV frame size of 140 bytes, the throughput related with an SV flow is around 4.5 Mbps (50Hz) and 5.4 Mbps (60Hz). For measurement, the sampling rate is 256 samples per cycle, but 8 points are grouped and sent in a single packet, resulting in a bandwidth of up to 10 Mbps (50Hz) or 12 Mbps (60Hz).

7.3.2.3 *PTP*

It relies on a master-slave scheme, where master broadcasts periodically a Sync message with the reference time, with no knowledge of the slaves. PTP packet size is difficult to determine because it depends on the operation mode (peer-to-peer P2P or end-to-end E2E), or whether the switches support it. Also, as the PTP protocol calculates its network delay, there is not a maximum delay specified in the standard for it. Likewise, since [Int13a; Oli+16] suggest that GOOSE and PTP have similar requirements, we define a rate of 3Mbps.

7.3.2.4 *MMS*

The MMS traffic generated by IEDs comprises a polling part from the SCADA and an event-driven part that depends on reports. An IED sends reports by data change, quality change or data update, with an interval between 60s to 300s [Int13a]. The MMS throughput suggested in the Table 7.1 was calculated following the same principles explained in the GOOSE section for random and periodic packets.

7.3.3 *HTB Queuing Discipline*

A queuing discipline determins how the packets are buffered while waiting to be transmitted

Hierarchical Token Bucket is a classful queueing discipline (algorithm to control the packet scheduling in a specific network interface), that allows a granular control over the outbound bandwidth on a link (traffic shaping) [Bro06; Dev02]. In general, the main HTB properties to set in this queueing discipline are: 1) *rate*, minimum guaranteed rate for this class and their children; 2) *ceil*, maximum rate at which this class is allowed to transmit; and 3) *priority*, defines the priority of the class to request idle bandwidth (see Figure 7.2). To conclude, the operating principles of this algorithm are described as follows:

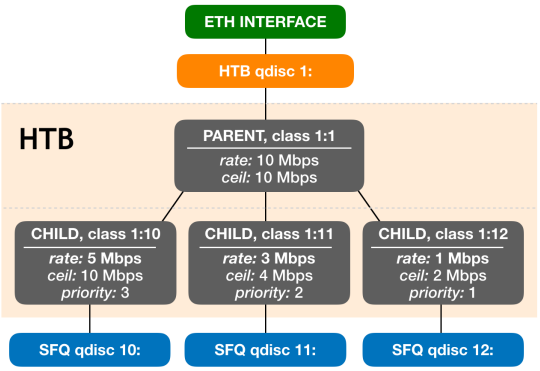


Figure 7.2: HTB queue discipline description

- The sum of the *rates* of the children classes should not exceed the *ceil* of a parent class.
- A child class always send packets with the minimum guaranteed rate.

- When a class uses less bandwidth than the amount assigned (*rate* property), the idle bandwidth is available for any other classes to use.
- When a child class' rate is exceeded, it is allowed to ask for idle bandwidth to its parent class until it reaches *ceil*.
- Classes with higher priority are offered idle bandwidth first.

7.4 ARCHITECTURE PROPOSAL

This work takes advantage of the SDN's benefits to ease the implementation of QoS policies inside power substation networks thanks to its ability for programming the network. Figure 7.3 illustrates the architecture proposal.

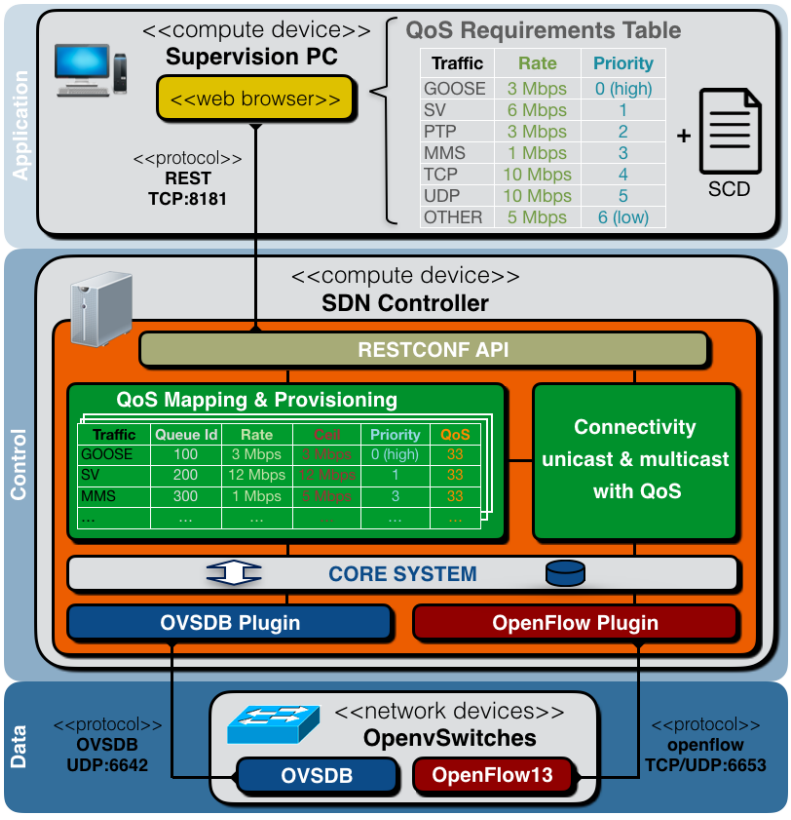


Figure 7.3: Architecture proposal

7.4.1 *Application domain*

In this domain the following requirements are entered by the substation operator and then, they are transmitted to the SDN controller via REST:

- *Substation Configuration Description (SCD).* This file describes the entire power substation, including all its devices and the way that they are linked together in the system.
- *QoS Requirements Table.* This table shows a prioritized arrangement of the main communications services (GOOSE, SV, PTP and MMS), along with the minimum rate to guarantee. Notice that the throughput suggested, in the GOOSE and MMS cases, corresponds to the most challenging condition evaluated. Furthermore, the table presents reference values for any other traffic existing in the network: TCP, UDP or others (ARP, ICMP, etc). However, all values of this table can be modified according to the operator criterion.

7.4.2 *Control domain*

This domain is responsible of: 1) receiving the QoS requirements as well as the origin and destination for the communication services inside the power substation network (Substation Configuration Description (SCD) file), 2) building and mapping QoS policies, 3) determining the needed flows to guarantee the unicast and multicast connectivity between all devices and 4) transmitting flow rules to the switches under the QoS defined environment.

- *QoS Mapping & Provisioning.* This module is in charge of building QoS policies with the HTB strategy and the assistance of the OVSDB plugin [PD13]. To achieve this objective, it is necessary to map all the estimated flows of the system to each switch port (outgoing traffic), from the communication requirements of the SCD file. With this input, the *rate* and *ceil* HTB parameters are estimated, since the *priority* setting was already defined in the QoS Requirements Table. For example, an IED can be subscribed to: two MU (two SV flows), one IED (one GOOSE flow), and the requests from the station level (MMS flow). In that case, the HTB parameters will be defined as shown in the QoS Mapping & Provisioning table of Figure 7.3. Later, all this information passes to the OVSDB plugin for executing the next process: 1) For each switch port, seven queues associated with outgoing traffic are created using the HTB mechanism (one queue for each communication service); 2) These seven queues are linked to just one QoS policy and finally 3) this QoS policy is attached to just one switch port. All this information is stored in the Queue, QoS and Port tables of each OpenvSwitch (OVS). In summary, each switch port has one QoS policy associated, containinig seven queues.
- *Connectivity unicast & multicast with QoS.* This module is responsible for: controlling the broadcast traffic such as ARP in loop-

based topologies, guaranteeing the unicast communication (TCP, UDP), giving multicast connectivity (GOOSE, SV) and providing redundancy to GOOSE messages, using QoS policies. This block takes the information given in the SCD file to establish flows between devices. Once the route for the flow is determined, this unit makes use of the queues IDs linked to the ports (QoS Mapping & Provisioning module) to build the corresponding flow rules. Flow rules are transmitted to the switches through the OpenFlow plugin.

7.4.3 *Data domain*

This area is integrated to the forwarding devices, in this case OpenvSwitches. In the proposed architecture, these devices are managed by the SDN controller through of two southbound protocols: OVSDB [PD13] in charge to create and operate the QoS policies, and OpenFlow [McK+08] responsible of managing the flow rules and linking them with the QoS policies already created. OpenvSwitch supports two classful queuing disciplines: HTB and Hierarchical Fair Service Curve (HFSC). However, as OpenvSwitch lacks support for all the HFSC settings, we use the HTB mechanism.

7.5 TESTBED

7.5.1 *Environment*

This testbed runs on Ubuntu 16.04 LTS machine, with an Intel Core i7 @3.40GHz x 8 CPU, and 16GiB of RAM. To generate network topologies we use Mininet [Tea12], a network emulator used in the SDN research field which includes OpenvSwitch [Nic09] (a virtual switch that supports OpenFlow and OVSDB southbound protocols). The control domain of the testbed is governed by the Opendaylight Controller [Odl]. The testbed topology consists of a spiderweb topology (Figure 7.4b), a mixture of two topologies: star and ring, that provides a highly optimized structure with tolerance to link failures, which was described in Chapter 4. In this case, the network topology represents an approximation of a small substation of 220/132 kV, with single bus, which is classified as T1-1 by IEC 61850-5. Figure 7.4a shows the diagram line for such substation, along with a possible structure for the control, protection and measuring devices in each line bay: Circuit Breaker (XCBR), Protection Time Overcurrent (PTOC) and Merging Unit (MU).

Finally, the measuring MUs, protection and control IEDs and supervising devices were emulated using mininet-hosts executing applications from the libIEC61850 project [Zil16]. This framework provides an API for implementing MMS server and client, GOOSE publisher and

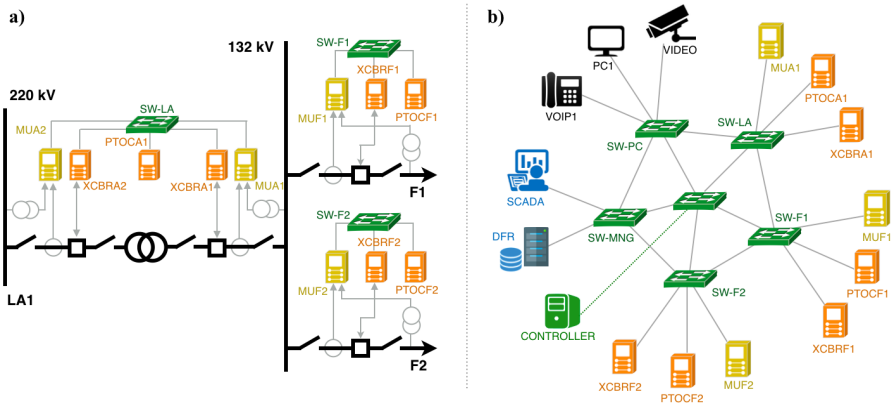


Figure 7.4: a) Diagram line for a T1-1 substation b) Network topology suggested

subscribers and SV publisher and subscribers. Besides, the link bandwidth was set to 100Mbps, to guarantee a reasonable workload of the machine resources under the operation conditions of the system.

7.5.2 *System operation conditions*

In order to evaluate the effectiveness of our approach we propose the next operation conditions (see Figure 7.5).

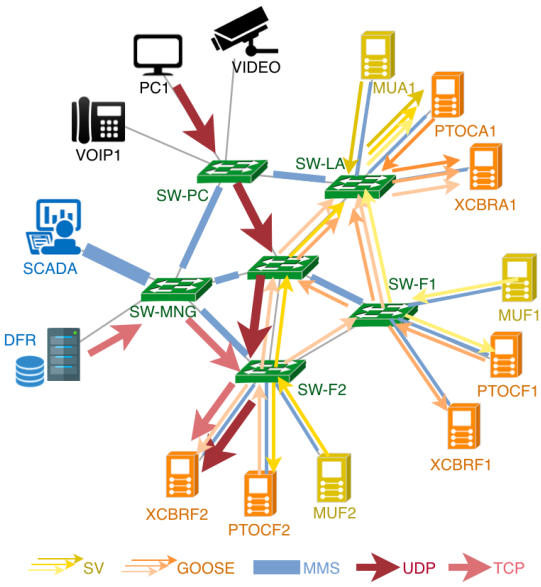


Figure 7.5: System operation conditions

- *SV Flows:* All MUs send SV traffic to each of their bay protections (PTOC). In addition, all the MUs shall also send SV traffic to the PTOCA1, since it will simulate the implementation of a differential protection too.

- *GOOSE Flows:* All protections (PTOCs) send GOOSE traffic periodically to their corresponding bay breakers (XCBRs). Furthermore, in case of detecting an overcurrent, they will trigger the breaker of its bay and the breaker of the main line. It means that all PTOCs will send GOOSE traffic to XCBRA1. It is important to mention that the GOOSE flows coming out of each bay will take the shorter two disjoint paths to get the XCBRA1. This behavior has the purpose of guaranteeing redundancy and fault-tolerance.
- *MMS Flows:* All measurement, protection and control devices send periodically information via MMS to the operation center. The thickness of the line is proportional to the number of MMS flows. A flow for each end device, 3 flows within the spider-web topology and 9 flows in the link that connects with the management center (SCADA).

7.5.3 *QoS considerations*

The queue discipline implemented in this testbed is HTB. HTB allows classifying different types of traffic and, according to this classification, assign them levels of throughput and priority. In this testbed, HTB can be visualized as a tree, where the parent node sets the maximum capacity of the link (100 Mbps, a FastEthernet link), while that the children coincide with the communications services of the power substations and their corresponding parameters: *rate*, *ceil* and *priority* (see Figure 7.6). With the purpose to simplify the validation in our testbed, we set the same QoS policy in all switches ports.

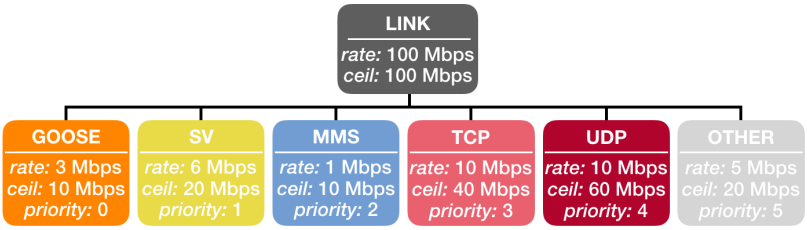


Figure 7.6: HTB testbed queue discipline

7.6 VALIDATION

The most critical information in a power substation involves the transmission of GOOSE trip commands from the protection IEDs to the circuit breaker IEDs. A GOOSE trip demands a maximum transmission time of 3ms, where 2.4 ms are used for processing times in the IED stacks and the remaining 0.6ms are used for the transmission over the communication network. In order to validate the effectiveness of our approach, we propose to generate an overcurrent event under normal and stress operation conditions, with and without QoS policies. The overcurrent

event will cause the transmission of GOOSE trip commands. This way, we can take delay measurements of these messages to determine the performance of the transmission time over the communication network. The graphics presented in this section are the result of 60 tests performed in presence and absence of QoS policies.

7.6.1 *Experiment description*

Figure 7.7 illustrates the traffic patterns in the link that connects the breaker XCBRF2, under the aforementioned conditions. PTOCF2 sends periodically GOOSE messages to the devices subscribed (XCBRF2 and XCBRA1), and XCBRF2 interchanges MMS information with the substation level. Also, two overcurrent events will be generated in the MUF2 (timeline 20s and 40s). This overcurrent will be detected by PTOCF2, who will send a GOOSE trip message to the XCBRF2 and XCBRA1 breakers. Later, to create a stress condition in the link (30s after simulation started), an UDP flow (70 Mbps) will be generated from PC1 as well as a TCP flow (40 Mbps) from DFR. Both flows have as destination the XCBRF2 control IED, trying to reach the available bandwidth of the link to XCBRF2 (100 Mbps).

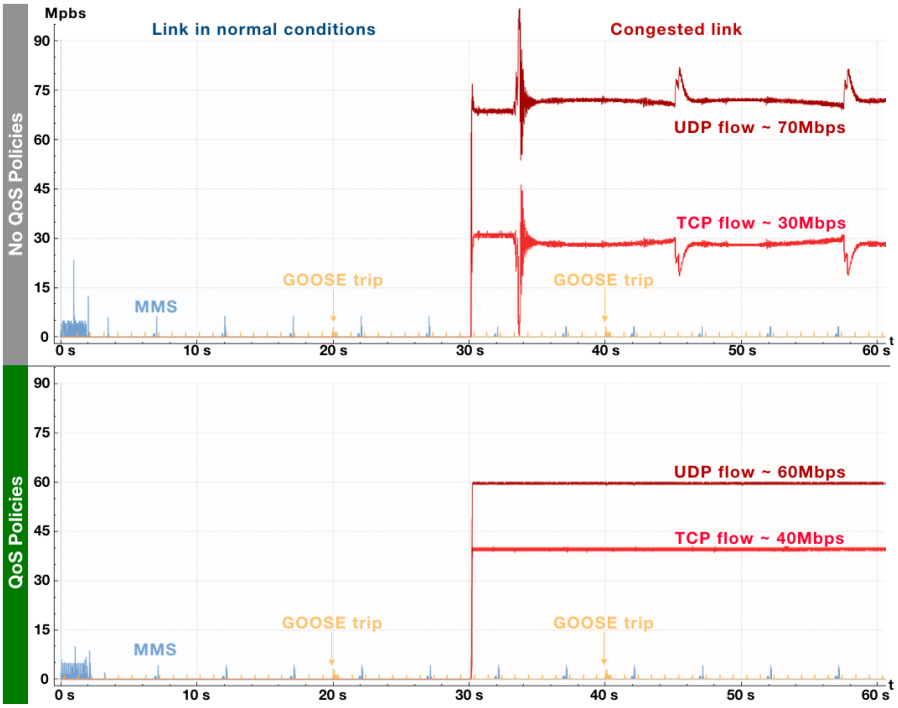


Figure 7.7: Traffic patterns with and without QoS policies

7.6.2 *GOOSE delay evaluation*

Figure 7.8 shows the delay performance for the GOOSE messages that arrived at XCBRF2 from PTOCF2. This delay can be calculated since all devices are running in the same machine, therefore they have the same machine time.

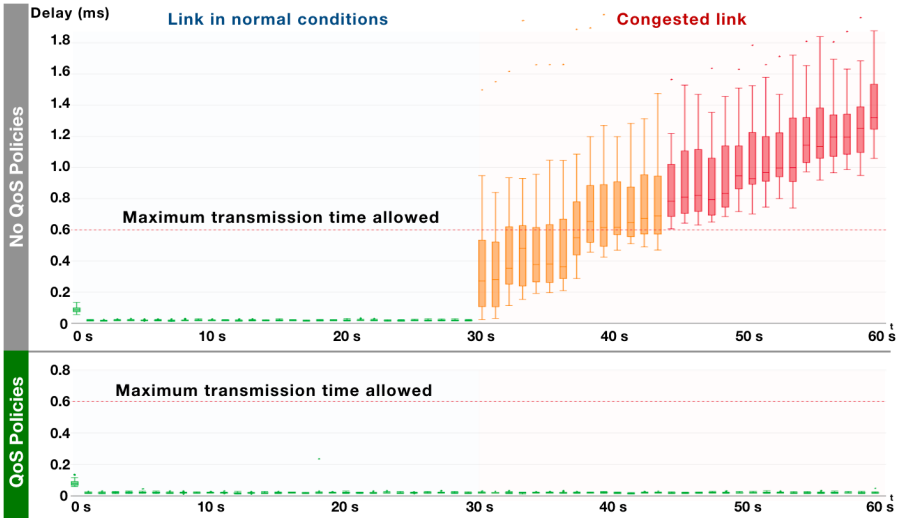


Figure 7.8: Delay performance for GOOSE messages with and without QoS policies

7.6.3 *Analysis of the results*

The tests carried out show that the implementation of QoS policies, under an SDN environment, satisfy the operational requirements of power substation protocols such as GOOSE, even under conditions of heavy traffic loads. This fact demonstrates that traffic shaping strategies are paramount in the right performance of power substation networks and that SDN is a technological enabler that facilitates this task. From the obtained outcomes, it is important to mention that:

- In normal conditions, Figure 7.7 and Figure 7.8 evince that GOOSE protocol operates satisfactorily, regardless of whether or not there are QoS policies implemented.
- In overload conditions, the reader can verify in Figure 7.7 how TCP and UDP traffic are shaped according to the aforementioned guidelines (TCP *ceil* - 40 Mbps, UDP *ceil* - 60Mbps), when the QoS policies are implemented. In contrast, when they are absent, the TCP and UDP flows are contending all the time for the available bandwidth.
- At first sight, in Figure 7.7, GOOSE protocol operation looks suitable even in stress conditions. However, this apparent normal

behavior is due to the big scale of the figure. Figure 7.8 exposes the real performance of the delay on GOOSE messages in overload conditions. It shows how when the TCP and UDP traffic saturate the link (red background), the delay grows gradually if there are no QoS policies applied. At the beginning (timeline between 30s and 45s), sometimes GOOSE messages comply with the maximum transmission time for the communication network (0.6 ms), but sometimes they do not. This issue influences periodic GOOSE messages and GOOSE trips. Finally, after the 45s, under no circumstances the transmission time requirements are fulfilled. On the contrary, in the scenario with QoS policies, the delay performance never exceeded the 0.2 ms.

From the presented findings we can conclude that although GOOSE only requires 3 Mbps of provisioning per flow, this condition has to be defined in advance. On the contrary, at a critical moment, GOOSE will have to compete with the other flows no matter the link capacity, and that could be harmful.

7.7 CONCLUSION

In this chapter we evaluated the applicability of implementing QoS policies to guarantee the time-critical requirements of power substation protocols such as GOOSE, under an SDN environment. For this purpose, we suggested a traffic prioritization scheme, proposed an architecture and built a testbed to demonstrate the effectiveness of our approach. Results show that our prototype can effectively provide: traffic guarantees rates, transmission times according to the IEC 61850-90-4 guidelines and the highest bandwidth utilization through the use of HTB algorithm, by allowing idle bandwidth to be used by other QoS flows.

Part III

CONCLUSION AND FUTURE WORK

8 CONCLUSIONS

A Ph.D. means ...
enjoy the topic that you choose to discover, work hard in your tests, longer and more stressful days, write a lot, be patient with the peer reviewing process and expose, argue and defend your approach convincingly.

— Alexánder Leal

This thesis has exhaustively proposed new approaches to improve the operation of a power substation communication network in terms of management, reliability, service provisioning and security; by the use of the SDN principles. Thanks to this exploration, several contributions to the existing state of the art were introduced in this thesis. In particular, a categorization was proposed according to the focus of existing approaches. Also, a novel network architecture (S_3N) was introduced to model the interaction between the elements involved in the operation of the power substation. Also a reliable network topology outlined with the features of this architecture was proposed. In addition, this thesis included other contributions under an SDN environment in a power substation context such as: architectures to detect and mitigate security attacks in the reconnaissance phase or manage QoS provisioning, as well as algorithms to solve complex issues related to loops-based topologies (broadcast traffic control, path redundancy, packet redundancy, and multicast traffic management).

This chapter sums up the main contributions and results of the thesis as a consequence of the identified research opportunities. It will be split into three sections. Section 8.1 details the main results obtained throughout the development of the thesis. Section 8.2 presents the publications achieved in this thesis and their quality assessment. Finally, Section 8.3 shows the participation of the author of this thesis in research projects.

8.1 MAIN CONTRIBUTIONS AND RESULTS

Chapter 2 revealed the wide interest in evaluating the benefits that SDN can offer to the field of power substation communication networks. The review introduced a categorization according to the approach of the existing works: proofs of concept, evaluation of technical requirements and solutions offered by the industry. Besides, this survey highlighted in the research field that there is a trend to: use network emulators as mininet [Tea12], emulate network traffic, employ a wide

variety of SDN controllers, and use OpenFlow as a southbound interface by default [McK+08]. The results found were encouraging in most cases, and suggest that SDN can boost the management of power substation communications networks satisfying their demanding operation requirements, and offering the possibility of developing novel solutions.

Chapter 3 proposed a novel architecture called Smart Solution for Substation Networks (S3N), which allows representing, alternatively to the network model provided in the standard IEC 61850, the interaction between all elements involved in the operation of the power substation, taking the communications network as the focal point. This new approach uses the concepts proposed by SDN and virtualization technologies for its conception and takes as reference the network functionalities presented in the Chapter 2 to define an original SDN control plane. It means, this control plane is not an adaption of well known SDN-based control planes in different data networks (data centers, carrier networks), but on the contrary, it is defined from power substations' particularities. In addition, to explain in depth the elements that integrate the architecture S3N as well as their interactions, representation models like SGAM [CEN12] and Kruchten's 4 + 1 Model View [Kru95] were used according to the guidelines of the ISO/IEC/IEEE 42010 System and software engineering - Architecture description [ISO11]. We consider that this reconceptualization allows modeling the behavior of current and future power substation communication networks, and brought the opportunity to illustrate the benefits of applying SDN in this environment as shown in the Chapter 5, Chapter 6 and Chapter 7.

Chapter 4 introduced a novel methodology to specify, characterize and evaluate a reliable network topology that guarantees fault-tolerance, according to the guidelines described in the S3N architecture. After defining the S3N architecture in Chapter 3, it was necessary to specify reliable network topology (SDN data plane), to develop the functionalities conceived in its corresponding SDN control plane. In this chapter, we propose an ILP model to solve this research challenge considering requirements such as SDN legacy, power substation recommendations and paths redundancy. However, this procedure can be applied in any other environment, not just in the power substation communications network. The network topology found is called artificial spider web, a combination between a star and a ring topology. This topology gives an optimized structure to transmit information between nodes located further away from the center of the topology and 100% fault-tolerant. In addition, another important contribution that gives this chapter is a method to compare network topologies according to different criteria: terminal reliability, graph metrics, and end-to-end time-delay; which can be applied in any environment. This instrument confirmed that the spider web and the dual redundant

tree topologies are the best options to support the power substation communications network.

Chapter 5 exposed alternative SDN solutions to overcome shortcomings found in the existing mechanisms to manage power substations communications network. These approaches, or functionalities defined in the S3N architecture's control plane, include algorithms to solve complex issues in loops-based topologies like the network topology proposed in Chapter 4. In particular, we develop applications to network topology discovery, broadcast traffic control, SV multicast traffic management and GOOSE multicast traffic management, some of which would be technically unfeasible using common network protocols. It is not possible to bring at the same time fault-tolerance with redundant links if traditional protocols like STP disable all redundant paths in the topology to avoid loops. The achieved results, when the algorithms were applied over our testbed, were consequent with the expected operation scheme of the communication network. One important aspect here is that all algorithms, except for the GOOSE algorithm which takes advantage of the particularities of the spider web topology, proved to work satisfactorily in a dual redundant tree topology. This fact demonstrates the effectiveness of our approach.

Chapter 6 presented two strategies to detect intrusions in power substations communication networks based on the IEC 61850 standard, considering the serious damage that can cause a failure in this critical infrastructure over a highly energy-dependent society. The first makes use of the sampling and monitoring network tool, sFlow. While the second uses unsupervised clustering algorithms for this purpose. In this chapter, we can enumerate several contributions. 1) the adequate definition of statistical descriptors (averages) was a key aspect of the achieved results in both strategies. In the hierarchical clustering approach, the descriptors proved to be robust getting identical results with other unsupervised clustering techniques such as K-means (partitional-type clustering), or LAMDA (diffuse-type clustering). Likewise, in the strategy of sampling packets to detect attacks in the reconnaissance phase, the descriptors shown to be effective. 2) the exploration of how unsupervised clustering algorithms can be used as an effective tool to identify operation scenarios, in power substations communication networks. And 3) an architecture composed of two domains: Monitoring and Countermeasure, where the first domain is governed by sFlow (in charge of detecting network anomalies defined by user rules), while the second domain is managed by an SDN controller (responsible of mitigating the intrusion).

Chapter 7 described the challenges around of QoS provisioning in power substations communication networks, which require a careful management of the traffic due to their reliability requirements. To overcome these shortcomings, the conceptualization of an architecture under the SDN paradigm was carried out and a traffic prioritization

scheme was suggested based on a deep analysis of the IEC 61850 communication services requirements. Results show that our approach can effectively provide: traffic guarantees rates, transmission times according to the IEC 61850-90-4 guidelines and the highest bandwidth utilization through the use of the HTB algorithm, by allowing idle bandwidth to be used by other QoS flows.

Finally, with the development of this thesis, we expected to answer the following research question: *Can the proposed architecture bring solutions that improve the management of power substation communications network, ensuring existing levels of reliability and availability in line with the standards and requirements set by the sector?* And, based on the work and the contributions presented in this thesis, the author argues that the S3N architecture can facilitate the operation and management of power substation communications network thanks to incorporating SDN as a technological enabler.

SDN brings with it the features of the software to the networks. Now, any network engineer is able to develop new solutions (named protocols or applications), according to the specific needs of his organization. However, we can not forget that the software not only involves benefits, it brings responsability too.

8.2 PUBLICATIONS

Part of the research work presented in this Ph.D. Thesis has been internationally validated in different networking peer-reviewed journals and conferences. Several experts have provided their comments in the peer reviews allowing us to improve our investigations and to guide the direction of our research. However, at the moment to finish this dissertation, some papers are still in the reviewing process. The articles carried out during the development of this thesis are detailed in Table 8.1 and Table 8.2.

8.3 RESEARCH PROJECTS

The research contribution made by this thesis to improve the current operation of a power substation communication network in terms of management, reliability, service provisioning and security; was made in the context of two research projects summarized in Table 8.3.

PAPER TITLE	JOURNAL	QUARTILE	STATE
S3N-Smart Solution for Substation Networks, an architecture for the management of communication networks in power substations [LB16a]	Lecture Notes in Computer Science	Q2	Published
Transforming communication networks in power substations through SDN [LB16c]	IEEE Latin America Transactions	Q2	Published
Hierarchical clustering for anomalous traffic conditions detection in power substations [LBJ18a]	IEEE Latin America Transactions	Q2	Submitted
Improving early attack detection in networks with sFlow and SDN [LBJ18a]	Communications in Computer and Information Science	Q3	Published
Defining a reliable network topology in a Software-defined power substations context [LB19b]	IEEE Access	Q1	Published
A novel architecture for power substations communications networks based on SDN and virtualization paradigms [LB19a]	IEEE Communications Magazine	Q1	Submitted

Table 8.1: Publications in journals

PAPER TITLE	CONFERENCE	TYPE
Software defined power substations: An architecture for network communications and its control plane [LB16b]	IEEE Latin-American Conference on Communications	International, Indexed in DBLP and IEEE Xplore
Improving fault tolerance in critical networks through OpenFlow [DLB17]	IIEEE Colombian Conference on Communications and Computing	National, Indexed in IEEE Xplore
QoS Proposal for IEC 61850 traffic under an SDN environment [LBJ18b]	IEEE Latin-American Conference on Communications	International, Indexed in DBLP and IEEE Xplore

Table 8.2: Publications in conferences

PROJECT NAME	DURATION	STATE	PARTNERS INVOLVED
Platform for management of electrical substations supported in Software Defined Networks	24-06-2015 31-03-2016	Finished	University of Antioquia Kinnesis Solutions S.A.S
STORM, Intelligent Management of Electric Services in the Cloud	01-08-2016 01-08-2018	Finished	University of Antioquia

Table 8.3: Research projects where the author of this thesis was involved

9 FUTURE WORK

The future depends on what we do in the present.

— Mahatma Gandhi

The contributions presented in this thesis can still be improved and, even better, being used in several projects and testbeds focused on power substation communication networks or even outside this topic of interest. This chapter is divided into three sections: Section 9.1 presents possible scenarios where the contributions introduced in this thesis can be applied. Section 9.2 details the possible enhancements that can be made to evolve and improve the current state of the presented contributions. Finally, in Section 9.3, new future lines of research are highlighted.

9.1 THESIS CONTRIBUTIONS, WHERE TO APPLY THEM?

One challenge for the energy sector is the constant maintenance and improvement of their communications networks. They have to deal with a variety of equipment maintenance from different vendors, which is not an easy task. In this context, this thesis brings contributions to improve the operation and management of power substation communication networks. A simple search about how SDN is being used as a key technological enabler on utilities shows the relevance of our contributions.

Industry 4.0 is a concept that corresponds with the next step in the industrial development where the devices are connected and communicate with one another to ultimately make decisions. Here, once again we can see how the communications networks is a key element in this emergent concept since the only way to guarantee interaction between smart devices is through a communications network. All contributions presented in this thesis: the S3N architecture, the reliable network topology implemented, the algorithms and the proposals to improve the security and the QoS provisioning; can contribute to fit the challenges of Industry 4.0.

9.2 FURTHER IMPROVEMENTS OF PROPOSED CONTRIBUTIONS

In Chapter 5 we proposed several algorithms to solve complex issues in loops-based topologies. However, the Algorithm 5.4 related to the tasks of GOOSE multicast traffic management was developed to take advantage of the particularities of the spider web topology. This algo-

rithm is susceptible of being improved so it becomes agnostic to the network topology. In this context, there is another important challenge here, since in the current solution the deletion of duplicate GOOSE messages is in charge of the end devices IEDs. It will be interesting to explore the Data Plane Development Kit (DPDK) framework to determine if it is possible to translate the task of deleting duplicate GOOSE messages to the switch.

One of the contributions presented in Chapter 6 is an architecture to detect attacks in the reconnaissance phase using the sampling and monitoring network tool, sFlow. With sFlow, the network administrator can define rules to detect network anomalies. This is an opportunity to incorporate new flow definitions to increase the robustness of the system. In the same way, it is proposed to use the statistical information provided by sFlow to implement entropy analysis to identify network anomalies that have not been defined.

In Chapter 7 we suggested the conceptualization of an architecture under the SDN paradigm to provide QoS provisioning. In this approach, we use a classful queueing discipline HTB created and managed by OVSDB. However, this proposal can be complemented through the use of *meters* in OpenFlow. A *meter* is an element that measures and controls the ingress rate of packets before the output. Unlike queues, *meters* are created and managed by OpenFlow protocol in meter table on the switch and attached directly to the flows instead to the switch ports.

Another pending task corresponds to the study of implementing Distributed SDN Controllers. It means, guaranteeing a logically centralized view through multiple distributed SDN controller instances to increase the robustness of the solution proposed.

9.3 FUTURE RESEARCH IN THE POWER SUBSTATION COMMUNICATIONS NETWORK

Security becomes one of the most critical factors in a communications network based on the IEC 61850 standard. Several works have evidenced vulnerabilities over GOOSE protocol since this is not safe. Here, there is an opportunity of research to determine what could be the best strategies to guarantee authentication and integrity on GOOSE messages, considering the demanding operational requirements of this protocol and the computing resources related to the cryptographic logic.

Another important area to research consists in providing the SDN controller with computational intelligence mechanisms, to evaluate the feasibility of automating management processes and/or detecting anomalies. For examples, the Time Series Data Repository (TSDR) project in OpenDaylight creates a framework for collecting, storing, querying, and maintaining time series data. In this context, all this

information could be used in the training of machines learning to satisfy specific needs.

An IED corresponds to a *hardware/software* solution used in power substations in charge of receiving information from sensors and power equipment, processing it and taking actions according to the needs of the energy operator. However, is an IED susceptible of being virtualized, like any other device? In this thesis the IEDs were emulated using *hosts-containers* executing applications from the libIEC61850 project [Zil16]. It would be interesting to evaluate how to develop an virtual-IED using *Unikernel* technology. An *Unikernel* is a single-address-space machine image constructed using library operating systems. In this way, we could deploy IEDs in a customized virtual instrumentation hardware, overcoming many shortcomings related to the commisioning of these devices.

Part IV

APPENDIX

A

S3N ARCHITECURE VIEWS

A.1 SGAM VIEWS

Next, it is illustrated how the SGAM layer model can be applied to get *views* of our system of interest.

A.1.1 *Components and Function Layers*

Figure A.1 shows, integrated in the same graph, the components layer as well as the function layer, according to the color scheme used to identify the different sectors of the S3N architecture.

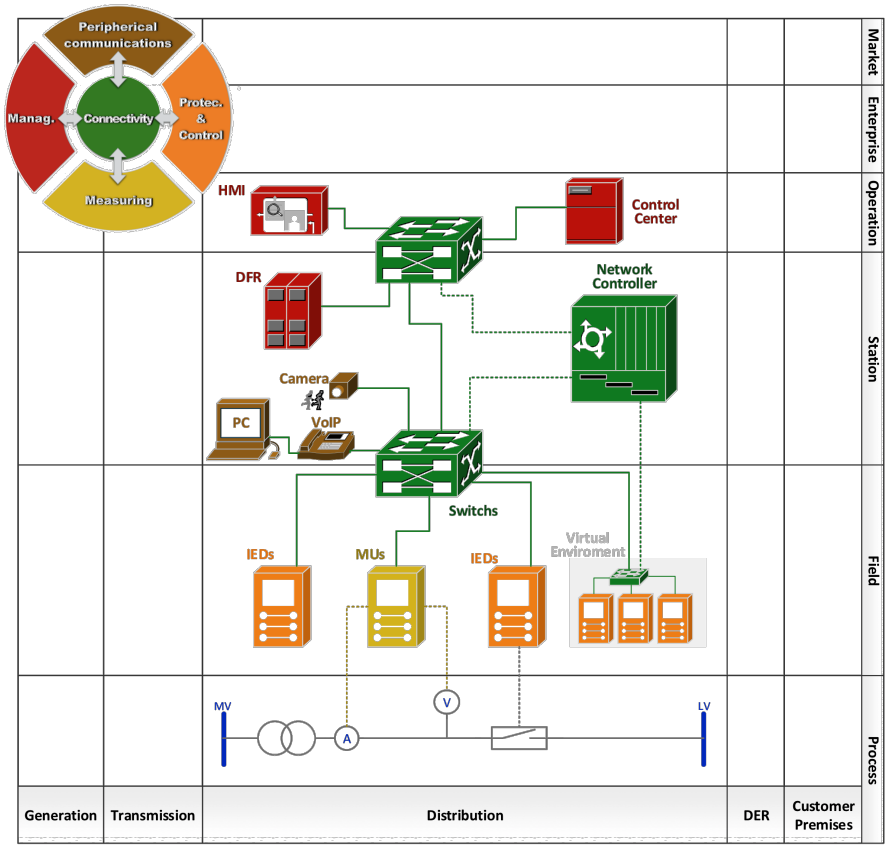


Figure A.1: SGAM Components and Function Layers

A.1.2 *Communication Layer*

Figure A.2 presents the communication protocols for the data exchange of the necessary information between the components.

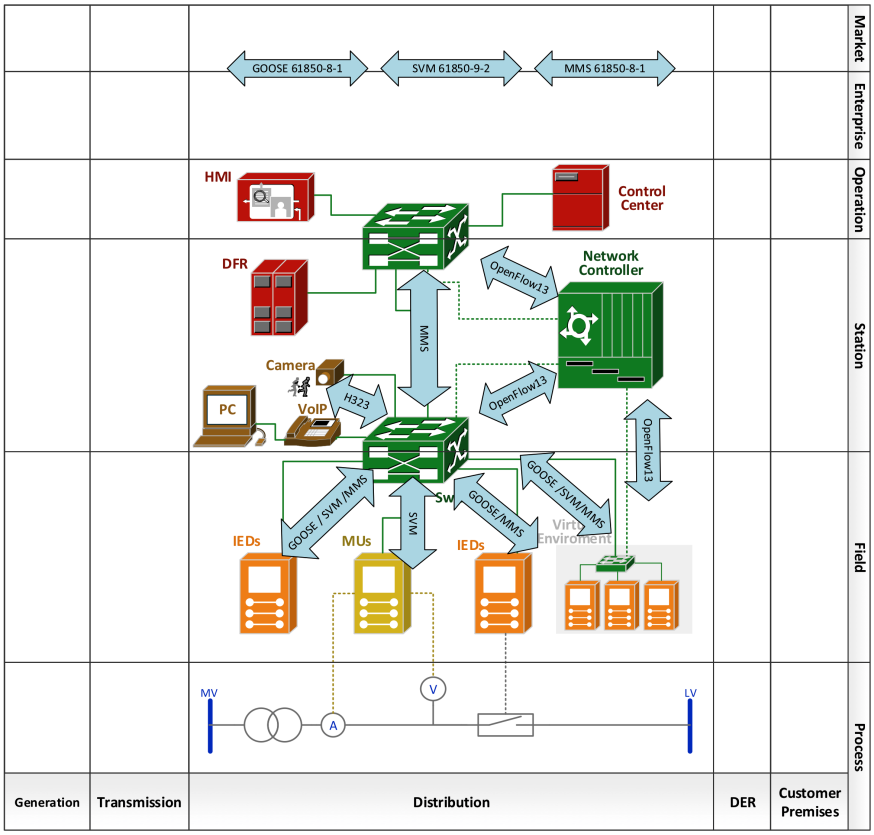


Figure A.2: SGAM Communication Layer

A.2 KRUCHTEN VIEWS

Next, it is shown some *views* around Kruchten 4+1 model view.

A.2.1 *Logical View, Extended class diagram with operations*

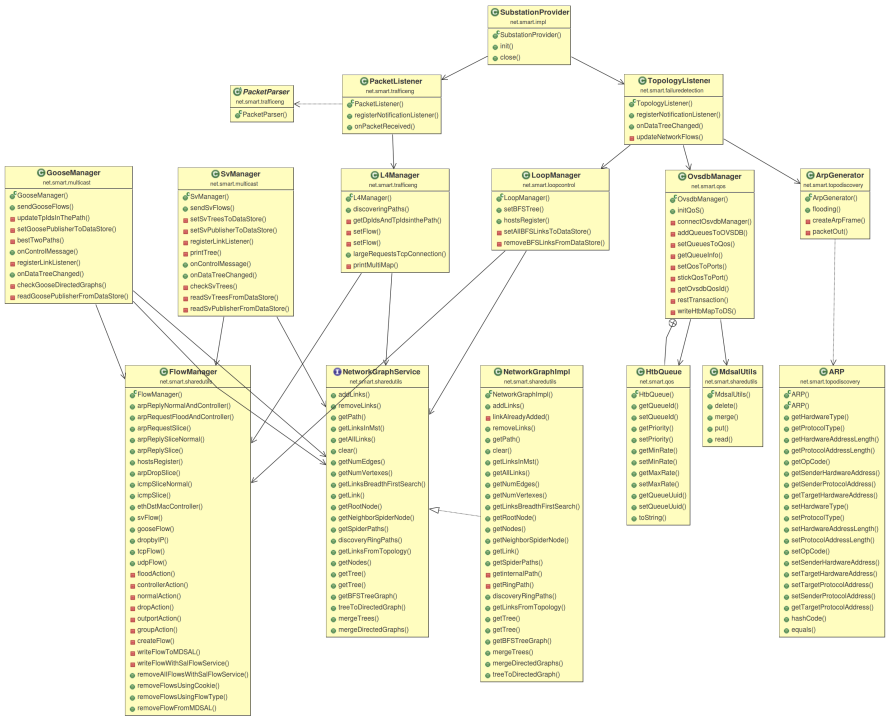


Figure A.3: Class diagram and their corresponding operations under the OpenDaylight framework domain

A.2.2 *Logical View, Extended class diagram with attributes*

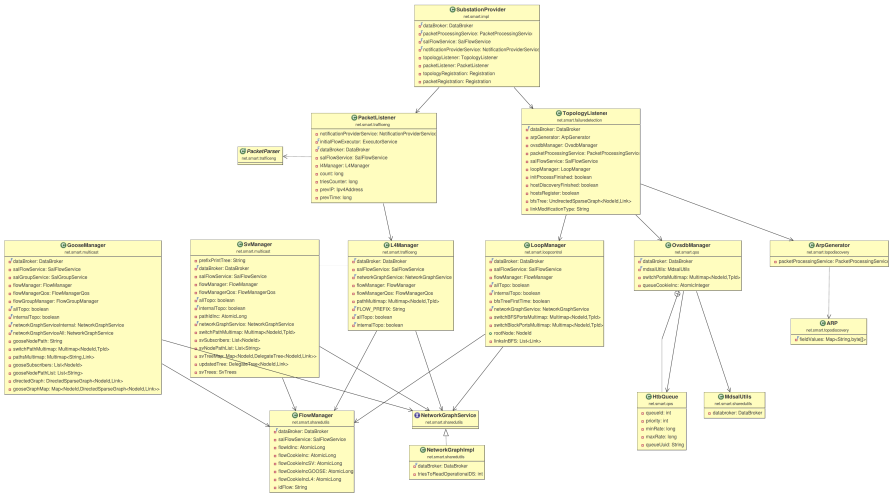


Figure A.4: Class diagram and their corresponding attributes under the OpenDaylight framework domain

A.2.3 *Development View, Components under the OpenDayLight domain*

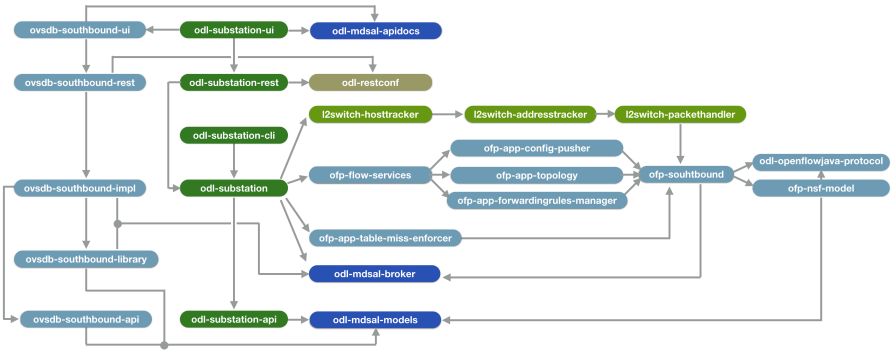


Figure A.5: Components and dependencies diagram under the OpenDay-Light domain

BIBLIOGRAPHY

[LB16a] Erwin Alexander Leal and Juan Felipe Botero. "S3N-Smart Solution for Substation Networks, an architecture for the management of communication networks in power substations." In: *Lecture Notes in Computer Science* 9701 (2016), pp. 52–56. ISSN: 0302-9743. DOI: 10.1007/978-3-319-39814-3_5.

[LB16c] Erwin Alexander Leal and Juan Felipe Botero. "Transforming communication networks in power substations through SDN." In: *IEEE Latin America Transactions* 14.10 (2016), pp. 4409–4415. ISSN: 1548-0992. DOI: 10.1109/TLA.2016.7786323.

[LB16b] Erwin Alexander Leal and Juan Felipe Botero. "Software defined power substations: An architecture for network communications and its control plane." In: *Communications (LATINCOM), 2016 8th IEEE Latin-American Conference on*. IEEE. 2016, pp. 1–6. ISBN: 978-1-5090-5137-3. DOI: 10.1109/LATINCOM.2016.7811573.

[DLB17] Carlos Mario Duran, Erwin Alexander Leal, and Juan Felipe Botero. "Improving fault tolerance in critical networks through OpenFlow." In: *Communications and Computing (COLCOM), 2017 IEEE Colombian Conference on*. IEEE. 2017, pp. 1–6. ISBN: 978-1-5386-1060-2. DOI: 10.1109/ColComCon.2017.8088195.

[LBJ18a] Erwin Alexander Leal, Juan Felipe Botero, and Eduardo Jacob. "Improving early attack detection in networks with sFlow and SDN." In: *Communications in Computer and Information Science* 916 (2018), pp. 323–335. ISSN: 1865-0929. DOI: 10.1007/978-3-030-00353-1_29.

[LBJ18b] Erwin Alexander Leal, Juan Felipe Botero, and Eduardo Jacob. "QoS Proposal for IEC 61850 traffic under an SDN environment." In: *Communications (LATINCOM), 2018 10th IEEE Latin-American Conference on*. IEEE. 2018, pp. 1–6. ISBN: 978-1-5386-6754-5. DOI: 10.1109/LATINCOM.2018.8613240.

[Lea18] Erwin Alexander Leal. "Hierarchical clustering for anomalous traffic conditions detection in power substations." In: *IEEE Latin America Transactions* (2018). ISSN: 1548-0992. Submitted.

[LB19b] Erwin Alexander Leal and Juan Felipe Botero. "Defining a reliable network topology in Software-defined power substations." In: *IEEE Access* 7.1 (2019), pp. 1–17. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2893114.

[LB19a] Erwin Alexander Leal and Juan Felipe Botero. "A novel architecture for power substations communications networks based on SDN and virtualization paradigms." In: *IEEE Communications Magazine* (2019). ISSN: 0163-6804. Submitted.

[Far10] Hassan Farhangi. "The path of the smart grid." In: *IEEE power and energy magazine* 8.1 (2010).

[KK13] Reduan H Khan and Jamil Y Khan. "A comprehensive review of the application characteristics and traffic requirements of a smart grid communications network." In: *Computer Networks* 57.3 (2013), pp. 825–845.

[Tc5] "IEC 61850: Communication networks and systems for power utility automation." In: *International Electro technical Commission Std* (2010).

[Hua+15] Qi Huang, Shi Jing, Jianbo Yi, and Wei Zhen. *Innovative testing and measurement solutions for smart grid*. John Wiley & Sons, 2015.

[JP13] Raj Jain and Subharthi Paul. "Network virtualization and software defined networking for cloud computing: a survey." In: *IEEE Communications Magazine* 51.11 (2013), pp. 24–31.

[Kre+15] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. "Software-defined networking: A comprehensive survey." In: *Proceedings of the IEEE* 103.1 (2015), pp. 14–76.

[CB09] NM Mosharaf Kabir Chowdhury and Raouf Boutaba. "Network virtualization: state of the art and research challenges." In: *IEEE Communications magazine* 47.7 (2009), pp. 20–26.

[Mij+16] Rashid Mijumbi, Joan Serrat, Juan-Luis Gorricho, Niels Bouten, Filip De Turck, and Raouf Boutaba. "Network function virtualization: State-of-the-art and research challenges." In: *IEEE Communications Surveys & Tutorials* 18.1 (2016), pp. 236–262.

[Cah+13] Adam Cahn, Juan Hoyos, Matthew Hulse, and Eric Keller. "Software-defined energy communication networks: From substation automation to future smart grids." In: *Smart Grid Communications (SmartGridComm),*

2013 IEEE International Conference on. IEEE. 2013, pp. 558–563.

[Mol+15b] Elias Molina, Eduardo Jacob, Jon Matias, Naiara Moreira, and Armando Astarloa. "Using software defined networking to manage and control IEC 61850-based systems." In: *Computers & Electrical Engineering* 43 (2015), pp. 142–154.

[LFM14] Yona Lopes, Natalia Castro Fernandes, and Carlos Alberto Malcher. "SMARTFlow: Uma Proposta para a Autoconfiguração de Redes de Subestação IEC 61850 Baseada em OpenFlow." In: *XIX Workshop de Gerência e Operação de Redes e Serviços, WGRS 2014, Proceedings*, 2014, pp. 31–44.

[Bob+14] Rakesh Bobba, Donald R Borries, Rod Hilburn, Joyce Sanders, Mark Hadley, and Rhett Smith. *Software-Defined Networking Addresses Control System Requirements*. Published in Sensible Cybersecurity for Power Systems: A Collection of Technical Papers Representing Modern Solutions, 2018. 2014. URL: https://www.selinc.com/WorkArea/DownloadAsset.aspx?id=104399.

[PD14] Thomas Pfeiffenberger and Jia Lei Du. "Evaluation of software-defined networking for power systems." In: *Intelligent Energy and Power Systems (IEPS), 2014 IEEE International Conference on*. IEEE. 2014, pp. 181–185.

[Dor+14b] Nils Dorsch, Fabian Kurtz, Hanno Georg, Christian Hägerling, and Christian Wietfeld. "Software-defined networking for smart grid communications: Applications, challenges and advantages." In: *Smart Grid Communications, 2014 IEEE International Conference on*. IEEE. 2014, pp. 422–427.

[Don+15] Xinshu Dong, Hui Lin, Rui Tan, Ravishankar K Iyer, and Zbigniew Kalbarczyk. "Software-defined networking for smart grid resilience: Opportunities and challenges." In: *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*. ACM. 2015, pp. 61–68.

[KFK15] Jaebeom Kim, Fethi Filali, and Young-Bae Ko. "Trends and potentials of the smart grid infrastructure: from ICT sub-system to SDN-enabled smart grid architecture." In: *Applied Sciences* 5.4 (2015), pp. 706–727.

[Dor+14a] Nils Dorsch, Boguslaw Jablkowski, Hanno Georg, Olaf Spinczyk, and Christian Wietfeld. "Analysis of communication networks for smart substations using a virtualized execution platform." In: *2014 IEEE International*

Conference on Communications (ICC). IEEE. 2014, pp. 4239–4245.

[Luw14] Emmanuel Luwaca. "Virtualization of a sensor node to enable the simulation of IEC 61850-based sampled value messages." PhD thesis. Cape Peninsula University of Technology, 2014.

[Nun+14] Bruno Astuto A Nunes, Marc Mendonca, Xuan-Nam Nguyen, Katia Obraczka, and Thierry Turletti. "A survey of software-defined networking: Past, present, and future of programmable networks." In: *IEEE Communications Surveys & Tutorials* 16.3 (2014), pp. 1617–1634.

[McK+08] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. "OpenFlow: enabling innovation in campus networks." In: *ACM SIGCOMM Computer Communication Review* 38.2 (2008), pp. 69–74.

[Dor+10] Avri Doria, J Hadi Salim, Robert Haas, Horzmud Khosravi, Weiming Wang, Ligang Dong, Ram Gopal, and Joel Halpern. *Forwarding and control element separation (ForCES) protocol specification*. Tech. rep. 2010.

[ONF11] ONF. *Open Networking Foundation*. 2011. URL: https://www.opennetworking.org/ (visited on 08/30/2016).

[Bar13] Diane Barrett. "Security Architecture and Forensic Awareness in Virtualized Environments." In: *Cybercrime and Cloud Forensics: Applications for Investigation Processes*. IGI Global, 2013, pp. 133–135.

[ISO11] May ISO. *Systems and software engineering–architecture description*. Tech. rep. ISO/IEC/IEEE 42010, 2011.

[Li+16] Gaolei Li, Jun Wu, Longhua Guo, Jianhua Li, and Hongkai Wang. "SDN based dynamic and autonomous bandwidth allocation as ACSI services of IEC61850 communications in smart grid." In: *Smart Energy Grid Engineering (SEGE), 2016 IEEE*. IEEE. 2016, pp. 342–346.

[Mol+15a] Elias Molina, Jon Matias, Armando Astarloa, and Eduardo Jacob. "Managing path diversity in layer 2 critical networks by using OpenFlow." In: *Network and Service Management (CNSM), 2015 11th International Conference on*. IEEE. 2015, pp. 394–397.

[Sch18b] Schweitzer Engineering Laboratories (SEL). *Watchdog Project*. 2018. URL: https://www.selinc.com/workarea/downloadasset.aspx?id=104832 (visited on 03/14/2018).

[Sch18a] Schweitzer Engineering Laboratories (SEL). *Software Defined Networking (SDN) Project*. 2018. URL: https://www.selinc.com/WorkArea/DownloadAsset.aspx?id=109179 (visited on 03/14/2018).

[Bla+13] Steven M Blair, Federico Coffele, Campbell D Booth, and Graeme M Burt. "An open platform for rapid-prototyping protection and control schemes with IEC 61850." In: *IEEE Transactions on Power Delivery* 28.2 (2013), pp. 1103–1110.

[Bio11] Biondi, Philippe. *Scapy packet manipulator*. 2011. URL: http://www.secdev.org/projects/scapy (visited on 03/14/2018).

[PF06] Thomas Pfeiffenberger and Thomas Fichtel. "CMT II: An agent based framework for comprehensive IP measurements." In: *TRIDENTCOM*. 2006.

[Zil16] Michael Zillgith. *libIEC61850 open source library for IEC 61850*. 2016. URL: http://www.libiec61850.com/libiec61850/.

[Tir99] Ajay Tirumala. *Iperf: The TCP/UDP bandwidth measurement tool*. 1999. URL: https://iperf.fr/.

[Tea12] Mininet Team. *Mininet. An Instant Virtual Network on your Laptop*. 2012. URL: http://mininet.org.

[Ryu] *Ryu controller*. 2012. URL: https://osrg.github.io/ryu/ (visited on 03/14/2018).

[Big12] Big Switch Networks. *Floodlight Project*. 2012. URL: http://www.projectfloodlight.org.

[Pox] *POX controller*. 2012. URL: https://github.com/noxrepo/pox/ (visited on 03/14/2018).

[Odl] *OpenDaylight controller*. 2013. URL: https://www.opendaylight.org (visited on 03/14/2018).

[HP13] HP. *HP VAN SDN Controller Administrator Guide*. 2013. URL: https://support.hpe.com/.

[Nic09] Nicira Networks. *Open vSwitch*. 2009. URL: http://openvswitch.org.

[Eri13] David Erickson. "The beacon openflow controller." In: *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM. 2013, pp. 13–18.

[AKL99] Frank Armour, Stephen Kaisler, and S.Y. Liu. "A Big Picture Look at Enterprise Architectures." In: 1 (Feb. 1999), pp. 35 –42.

[Pak+14] Farzaneh Pakzad, Marius Portmann, Wee Lum Tan, and Jadwiga Indulska. "Efficient topology discovery in software defined networks." In: *Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on*. IEEE. 2014, pp. 1–8.

[Bot+15] Juan Felipe Botero, Juan Esteban Hoyos, Andres Castano, Willmar Arcila, Mario Duran, and Joan Rodriguez. "Plataforma para la gestión de subestaciones eléctricas soportado en Redes Definidas por Software." Proyecto de investigación, Universidad de Antioquia - Ruta N. 2015.

[ISC11] David ME Ingram, Pascal Schaub, and Duncan A Campbell. "Multicast traffic filtering for sampled value process bus networks." In: *Proceedings of the 37th Annual Conference of the IEEE Industrial Electronics Society (IECON 2011)*. IEEE (Institute of Electrical and Electronics Engineers, Inc.) 2011, pp. 1–6.

[CKM12] L Cepa, Z Kocur, and Z Muller. "Migration of the IT Technologies to the Smart Grids." In: *Elektronika ir Elektrotechnika* 123.7 (2012), pp. 123–128.

[YS15] Hyunguk Yoo and Taeshik Shon. "Novel approach for detecting network anomalies for substation automation based on IEC 61850." In: *Multimedia Tools and Applications* 74.1 (2015), pp. 303–318.

[HSL09] Sugwon Hong, Dae-Yong Shin, and Seung-Jae Lee. "Experimenting Security Algorithms for the IEC 61850-based Substation Communication." In: (2009).

[CEN12] CEN-CENELEC-ETSI, Smart Grid Coordination. *Smart Grid Reference Architecture*. 2012. URL: https://ec.europa.eu/energy/sites/ener/files/documents/xpert_group1_reference_architecture.pdf (visited on 09/14/2018).

[Kru95] Philippe B Kruchten. "The 4+ 1 view model of architecture." In: *IEEE software* 12.6 (1995), pp. 42–50.

[PPM01] Peter Phaal, Sonia Panchen, and Neil McKee. *InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks*. Tech. rep. 2001.

[PMP01] Sonia Panchen, Neil McKee, and Peter Phaal. *InMon Corporations sFlow: A Method for Monitoring Traffic in Switched and Routed Networks*. RFC 3176. Sept. 2001. DOI: 10.17487/rfc3176. URL: https://rfc-editor.org/rfc/rfc3176.txt.

[CIS18] CISCO. *NeXt UI Toolkit - Devnet*. 2018. URL: https://developer.cisco.com/site/neXt/ (visited on 09/14/2018).

[Int13a] International Electrotechnical Commission - TC57. *Communication networks and systems for power utility automation – Part 90-4: Network engineering guidelines*. IEC/TR 61850-90-4. IEC, 2013.

[Hir12] Hirschmann Company. *Hirschmann White Paper, Data Communication in Substation Automation System (SAS). WP 1004HE - Part 3*. 2012. URL: http://www.beldensolutions.com/de/Service/Downloadcenter/index.phtml (visited on 12/06/2017).

[Kan11] Mitalkumar Kanabar. "Investigating performance and reliability of process bus networks for digital protective relaying." PhD thesis. The University of Western Ontario, 2011.

[YK10] Navindra Yadav and Eruch Kapadia. "IP and Ethernet Communication Technologies and Topologies for IED networks." In: *Grid InterOp, October* (2010).

[Ali+15] Ikbal Ali, Mini S Thomas, Sunil Gupta, and SM Suhail Hussain. "IEC 61850 substation communication network architecture for efficient energy system automation." In: *Energy Technology & Policy* 2.1 (2015), pp. 82–91.

[Liu+14] Xiaosheng Liu, Jiwei Pang, Liang Zhang, and Dianguo Xu. "A high-reliability and determinacy architecture for smart substation process-level network based on cobweb topology." In: *IEEE Transactions on Power Delivery* 29.2 (2014), pp. 842–850.

[IEE15] IEEE PES - Power System Relaying Committee. *Centralized Substation Protection and Control*. Report of Working Group K15. IEEE, 2015.

[Men27] Karl Menger. "Zur allgemeinen kurventheorie." In: *Fundamenta Mathematicae* 10.1 (1927), pp. 96–115.

[Ski90] Steven Skiena. *Implementing discrete mathematics: combinatorics and graph theory with Mathematica*. pag. 177. Addison-Wesley, 1990, p. 177. ISBN: 0201509431.

[Mak00] Andrew Makhorin. *GLPK (GNU Linear Programming Kit)*. 2000. URL: http://www.gnu.org/software/glpk/.

[AO10] Yuko Aoyanagi and Ko Okumura. "Simple model for the mechanics of spider webs." In: *Physical Review Letters* 104.3 (2010), p. 038102.

[Int13b] International Electrotechnical Commission. *Communication networks and systems in substations -Part 5: Communication requirements for functions and device models*. IEC 61850-5. IEC - International Standard, 2013.

[BF12] Frank Beichelt and P Franken. *Reliability and maintenance: Networks and Systems*. CRC Press, Boca Raton, London, New York, 2012.

[HR87] Salim Hariri and Cauligi S Raghavendra. "SYREL: A symbolic reliability algorithm based on path and cut-set methods." In: *IEEE transactions on Computers* 36.10 (1987), pp. 1224–1232.

[TA04] Wendy Torell and Victor Avelar. *Mean time between failure: Explanation and standards*. White Paper 78, American Power Conversion. 2004. URL: https://pdfs.semanticscholar.org/de97/955063b165d594e0aa0b55e6e550b51aa51a.pdf.

[Sha15] Gilad Shainer. *Copper Vs. Fiber: What's Best In The Next-Gen Data Center?* 2015. URL: https://www.networkcomputing.com/data-centers/copper-vs-fiber-whats-best-next-gen-data-center/1410243954 (visited on 05/19/2017).

[Cis15] Cisco Systems. *Data Sheet, CISCO 1000BASE-T SFP*. 2015. URL: https://www.future-x.de/images/datasheet/s101552.pdf (visited on 05/19/2017).

[KD89] KANSKY Karl and Pascal Danscoine. "Measures of network structure." In: *Flux* 5.1 (1989), pp. 89–121.

[Bar16] Albert-László Barabási. *Network science*. Cambridge University Press, 2016.

[Int12] International Electrotechnical Commission. *Communication networks and systems for power utility automation – Part 10: Conformance testing*. IEC 61850-10. IEC - International Standard, 2012.

[Sie14] Siemens. *Latency on a Switched Ethernet Network - Application Note 8*. 2014. URL: https://support.industry.siemens.com/cs/attachments/94772587/RUGGEDCOM.pdf (visited on 05/19/2017).

[XN13] Xin Xu and Yimin Ni. "Analysis of networking mode caused by GOOSE delay of smart substation." In: *Software Engineering and Service Science (ICSESS), 2013 4th IEEE International Conference on*. IEEE. 2013, pp. 503–506.

[GEN10] GENI (Global Environment for Network Innovations). *OpenFlow Discovery Protocol and Link Layer Discovery Protocol*. 2010. URL: http://groups.geni.net/geni/wiki/OpenFlowDiscoveryProtocol (visited on 03/14/2018).

[Ope15] Open Networking Foundation. *OpenFlow Switch Specification*. 2015. URL: https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.3.5.pdf (visited on 03/14/2018).

[O'M10] O'Madadhain, Joshua and Fisher, Danyel and White, Scott and Boey, Y. *JUNG - Java Universal Network/Graph Framework*. 2010. URL: http://jung.sourceforge.net (visited on 12/06/2017).

[Zus48] Konrad Zuse. "Der Plankalkül - Programming language." (Konrad Zuse Internet Archive. See pp. 96–105 of the linked pdf file). PhD dissertation. 1948. URL: http://zuse.zib.de/item/gHI1cNsUuQweHB6.

[HP15] Rich Hunt and Bogdan Popescu. "Comparison of PRP and HSR Networks for Protection and Control Applications." In: *Western Protective Relay Conference, Spokane, WA*. 2015.

[Ope18] Open Networking Foundation. *Software-Defined Networking (SDN) Definition*. 2018. URL: https://www.opennetworking.org/sdn-resources/sdn-definition (visited on 03/14/2018).

[Gio+14] Kostas Giotis, Christos Argyropoulos, Georgios Androulidakis, Dimitrios Kalogeras, and Vasilis Maglaris. "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments." In: *Computer Networks* 62 (2014), pp. 122–136.

[GAM14] Kostas Giotis, Georgios Androulidakis, and Vasilis Maglaris. "Leveraging SDN for efficient anomaly detection and mitigation on legacy networks." In: *Software Defined Networks (EWSDN), 2014 Third European Workshop on*. IEEE. 2014, pp. 85–90.

[PL04] P Phaal and M Lavine. *sflow protocol specification version 5*. 2004.

[Gud+08] Natasha Gude, Teemu Koponen, Justin Pettit, Ben Pfaff, Martín Casado, Nick McKeown, and Scott Shenker. "NOX: towards an operating system for networks." In: *ACM SIGCOMM Computer Communication Review* 38.3 (2008), pp. 105–110.

[Nug+14] Muhammad Nugraha, Isyana Paramita, Ardiansyah Musa, Deokjai Choi, and Buseung Cho. "Utilizing OpenFlow and sFlow to detect and mitigate SYN flooding attack." In: *Journal of Korea Multimedia society* 17.8 (2014), pp. 988–994.

[HCP16] LIN Hsiao-Chung and WANG Ping. "Implementation of an SDN-based Security Defense Mechanism Against DDoS Attacks." In: *DEStech Transactions on Economics, Business and Management* iceme-ebm (2016).

[ZKM14] Adel Zaalouk, Rahamatullah Khondoker, and Ronald Marx. "An orchestrator-based architecture for enhancing network-security using network monitoring and sdn control functions." In: *Network Operations and Management Symposium (NOMS)*. IEEE. 2014, pp. 1–9.

[Dha+15] NI Gde Dharma, M Fiqri Muthohar, JD Alvin Prayuda, K Priagung, and Deokjai Choi. "Time-based DDoS detection and mitigation for SDN controller." In: *Network Operations and Management Symposium (APNOMS)*. IEEE. 2015, pp. 550–553.

[BM16] Chaitanya Buragohain and Nabajyoti Medhi. "FlowTrApp: An SDN based architecture for DDoS attack detection and mitigation in data centers." In: *Signal Processing and Integrated Networks (SPIN), 2016 3rd International Conference on*. IEEE. 2016, pp. 519–524.

[All13] Allied Telesis. *How To | Use sFlow® in a Network*. 2013. URL: https://www.alliedtelesis.com/sites/default/files/aw-_use_sflow_in_a_network_revb1.pdf (visited on 03/14/2018).

[Sfl] *Writing sFlow Applications*. 2015. URL: https://sflow-rt.com/writing_applications.php (visited on 03/14/2018).

[Ipe] *What is iPerf / iPerf3 ?* 2003. URL: https://iperf.fr/ (visited on 03/14/2018).

[UP13] M Uma and Ganapathi Padmavathi. "A Survey on Various Cyber Attacks and their Classification." In: *IJ Network Security* 15.5 (2013), pp. 390–396.

[Sano1] Daiji Sanai. *Detection of promiscuous nodes using ARP packets*. 2001. URL: http://www.securityfriday.com/promiscuous_detection_01.pdf (visited on 03/14/2018).

[Lyo97b] Gordon Lyon. *Nmap: the Network Mapper*. 1997. URL: https://nmap.org/ (visited on 03/14/2018).

[Pha] *Large flow*. 2013. URL: http://blog.sflow.com/2013/06/large-flow-detection-script.html (visited on 03/14/2018).

[Ras+14] Muhammad Talha Abdul Rashid, Salman Yussof, Yunus Yusoff, and Roslan Ismail. "A review of security attacks on IEC61850 substation automation system network." In: *Information Technology and Multimedia (ICIMU), 2014 International Conference on*. IEEE. 2014, pp. 5–10.

[Cho+12] Kyung Choi, Xinyi Chen, Shi Li, Mihui Kim, Kijoon Chae, and JungChan Na. "Intrusion detection of NSM based DoS attacks using data mining in smart grid." In: *Energies* 5.10 (2012), pp. 4091–4109.

[Pre+10] Upeka Kanchana Premaratne, Jagath Samarabandu, Tarlochan S Sidhu, Robert Beresh, and Jian-Cheng Tan. "An intrusion detection system for IEC61850 automated substations." In: *IEEE Transactions on Power Delivery* 25.4 (2010), pp. 2376–2383.

[HDB12] Juan Hoyos, Mark Dehus, and Timthy X Brown. "Exploiting the GOOSE protocol: A practical attack on cyber-infrastructure." In: *Globecom Workshops (GC Wkshps), 2012 IEEE*. IEEE. 2012, pp. 1508–1513.

[HLG14] Junho Hong, Chen-Ching Liu, and Manimaran Govindarasu. "Detection of cyber intrusions using network-based multicast messages for substation automation." In: *Innovative Smart Grid Technologies Conference (ISGT), 2014 IEEE PES*. IEEE. 2014, pp. 1–5.

[Kus+14] Nishchal Kush, Ejaz Ahmed, Mark Branagan, and Ernest Foo. "Poisoned GOOSE: exploiting the GOOSE protocol." In: *Proceedings of the Twelfth Australasian Information Security Conference-Volume 149*. Australian Computer Society, Inc. 2014, pp. 17–22.

[CMA05] Philip K Chan, Matthew V Mahoney, and Muhammad H Arshad. "Learning rules and clusters for anomaly detection in network traffic." In: *Managing Cyber Threats*. Springer, 2005, pp. 81–99.

[Gal15] Jose Angel Gallardo. *Analisis de Datos Multivariantes. Universidad de Granada*. 2015. URL: http://www.ugr.es/~gallardo/ (visited on 03/14/2018).

[Com+07] Gerald Combs et al. *Wireshark, network protocol analyzer*. 2007. URL: http://www.wireshark.org.

[Lyo97a] Gordon Lyon. *Nmap security scanner*. 1997. URL: http://nmap.org.

[San05] Salvatore Sanfilippo. *hping3 - Linux main page*. 2005. URL: http://www.hping.org.

[Sri10] P Srivats. *Ostinato, Packet traffic generator*. 2010. URL: http://ostinato.org.

[MLC07] Gerhard Münz, Sa Li, and Georg Carle. "Traffic anomaly detection using k-means clustering." In: *GI/ITG Workshop MMBnet*. 2007, pp. 13–14.

[LL11] Duo Liu and Chung-Horng Lung. "P2P traffic identification and optimization using fuzzy c-means clustering." In: *Fuzzy Systems (FUZZ), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2245–2252.

[DRP13] Christian J Dietrich, Christian Rossow, and Norbert Pohlmann. "CoCoSpot: Clustering and recognizing botnet command and control channels using traffic analysis." In: *Computer Networks* 57.2 (2013), pp. 475–486.

[NHV14] Pratik Narang, Chittaranjan Hota, and VN Venkatakrishnan. "PeerShark: flow-clustering and conversation-generation for malicious peer-to-peer traffic identification." In: *EURASIP Journal on Information security* 2014.1 (2014), p. 15.

[IDFF96] Roberto Ierusalimschy, Luiz Henrique De Figueiredo, and Waldemar Celes Filho. "Lua—an extensible extension language." In: *Software: Practice and Experience* 26.6 (1996), pp. 635–652.

[Com12] Geral Combs. *TShark—Dump and Analyze Network Traffic*. 2012. URL: https://www.wireshark.org/docs/man-pages/tshark.html.

[Tea04] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. 2004. URL: https://www.r-project.org.

[Gow66] John C Gower. "Some distance properties of latent root and vector methods used in multivariate analysis." In: *Biometrika* 53.3-4 (1966), pp. 325–338.

[HW79] J. A. Hartigan and M. A. Wong. "A k-means clustering algorithm." In: *JSTOR: Applied Statistics* 28.1 (1979), pp. 100–108.

[AMDM82] J Aguilar-Martin and R Lopez De Mantaras. "The process of classification and learning the meaning of linguistic descriptors of concepts." In: *Approximate reasoning in decision analysis* 1982 (1982), pp. 165–175.

[GJS03] Janusz Gozdecki, Andrzej Jajszczyk, and Rafal Stankiewicz. "Quality of service terminology in IP networks." In: *IEEE communications magazine* 41.3 (2003), pp. 153–159.

[Hua09] Fang Huang. "Implementation and QoS for High-performance GIServices in Special Information Grid." In: *Quantitative Quality of Service for Grid Computing: Applications for Heterogeneity, Large-Scale Distribution, and Dynamic Environments*. IGI Global, 2009, pp. 181–203.

[Oli+16] Lucas B Oliviera, GE Grid Solutions Brazil, Marcelo Zapella, GE Grid Solution Brazil, Wilian Zanatta, and GANI Arshad. "ETHERNET SWITCHES REQUIREMENT OVER IEC 61850 NETWORKS." In: *CONFERENCE OF ELECTRIC POWER SUPPLY INDUSTRY (CEPSI)*. Vol. 21. 2016, pp. 1–9.

[Fen+14] Shanqiang Feng, Chunchao Hu, Kai Ma, Xiaoyue Zhang, Wenqing Lan, and Hu Chen. "Implementation and Application of QoS in Power Ethernet Switch." In: *Advanced Materials Research* (2014).

[BHR16] Christof Brandauer, Matthias Herlich, and Jürgen Resch. *A proposal for explicit communication quality management in IEC 61850*. 2016. URL: https://www.salzburgresearch.at/wp-content/uploads/2016/09/PACWorld-Brandauer-et-al-A-proposal-for-explicit-communication-quality-management-in-IEC-61850.pdf (visited on 03/14/2018).

[Man17] Mavis Mangwana. "Design of a Quality of Service Framework for Micro Grid Communication Network based on IEC 61850 Standard." MA thesis. Harare, Zimbabwe: University of Zimbabwe, 2017.

[Bro06] Martin Brown. *Traffic Control HOWTO*. 2006. URL: http://linux-ip.net/articles/Traffic-Control-HOWTO/ (visited on 07/20/2018).

[Dev02] Martin Devera. *HTB Linux queuing discipline manual*. 2002. URL: http://luxik.cdi.cz/~devik/qos/htb/manual/userg.htm (visited on 07/20/2018).

[PD13] Ben Pfaff and Bruce Davie. *The open vSwitch database management protocol*. Tech. rep. 2013.

DECLARATION

I declare that the contents of this dissertation "S3N - SMART SOLUTION FOR SUBSTATION NETWORKS" are original and result of my own work. I confirm that:

- This thesis has not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university.
- Any material in this thesis, that has been already published for other authors, is always attributed and referenced.
- In the works presented by myself with others, I have made clear was done by others and what I have contributed myself.

Signed:

Medellín - Colombia, November 2018

Author,
Alexánder LEAL

Approval,
Prof. Dr. Juan Felipe BOTERO

COLOPHON

This document was typeset using LaTeX . The document layout is based on the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. Some adaptations were made by Alexánder Leal .

https://bitbucket.org/amiede/classicthesis/

Final Version as of March 21, 2019 (classicthesis v4.6).

UNIVERSIDAD
DE ANTIOQUIA

1 8 0 3