Wide-Area Situation Awareness based on a Secure Interconnection between Cyber-Physical Control Systems

por

Lorena Cazorla Suárez

Memoria presentada para optar al título de
Doctor por la Universidad de Málaga

Directores:

María Cristina Alcaraz Tello

Profesor Ayudante Doctor

Javier López Muñoz

Catedrático de Universidad

Septiembre de 2017

AUTOR: Lorena Cazorla Suárez

http://orcid.org/0000-0002-5286-0032

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

Contents

1 Introduction **1**
- 1.1 Evolution of the Control and the Interoperability . . . 1
  - 1.1.1 Control Infrastructures and Generations . . . 3
  - 1.1.2 Cyber-Physical Control Systems: Technologies and Protocols . . . 5
  - 1.1.3 A Review on Interoperability . . . 7
- 1.2 Security Issues: Lessons Learned and Risks . . . 8
  - 1.2.1 Cyber Stealth Attacks in Critical Infrastructures . . . 10
  - 1.2.2 Cyber Attacks in Interconnected CPCSs . . . 13
- 1.3 Goals of the Thesis . . . 14
  - 1.3.1 Direct Contributions . . . 15
  - 1.3.2 Outline of this Thesis . . . 16
- 1.4 Publications and funding . . . 17

2 Advanced Threats to the CPCS **19**
- 2.1 Stages of a Stealth Attack . . . 20
- 2.2 AICAn Taxonomy . . . 21
- 2.3 Classification of Cyber Stealth Attacks . . . 23
  - 2.3.1 Disconnection and Goodput Reduction . . . 24
  - 2.3.2 Active Eavesdropping . . . 25
  - 2.3.3 Scanning and Probing . . . 25
  - 2.3.4 Covert and Side Channel Exploitation . . . 27
  - 2.3.5 Code Injection . . . 29
  - 2.3.6 Assessment of Stealthiness . . . 31
- 2.4 Countermeasures and Prevention Mechanisms Against Stealth Attacks . . . 34
- 2.5 Summary . . . 40

3 Analysis of Requirements of Secure Interoperability in CPCS **43**
- 3.1 The Special Requirements and Constraints of Critical Control Systems . . . 45
  - 3.1.1 CCS Requirements . . . 45
  - 3.1.2 CCS Constraints . . . 47
- 3.2 NFR Model Framework . . . 48
- 3.3 NFR Requirements for Interoperability . . . 48
- 3.4 NFR Satisficing Techniques for Interoperability . . . 56
- 3.5 Metrics for Interoperability . . . 65

3.6 Requirements for Protection Solutions Deployed within Critical Systems 77
3.6.1 NFR Requirements for Protection Systems 79
3.6.2 NFR Satisficing Techniques for Protection Systems 81
3.6.3 Metrics for Protection Systems 84
3.7 Discussion 89

4 Services for the Secure Interoperability in CPCS 93
4.1 Basic Security Services 94
4.2 Prevention and Detection 95
4.2.1 Monitoring and Detection 96
4.2.2 Intelligence and Learning Techniques 98
4.2.3 Automation 104
4.2.4 Review of the State of the Art: Approaches, Techniques and Automation for Detection solutions 107
4.3 Awareness and Reaction 108
4.3.1 Module 1: Detection 110
4.3.2 Module 2: Automation 111
4.3.3 Module 3: Response System 113
4.3.4 Review of the State of the Art: Approaches, Techniques and Tools for IDPRS solutions 121
4.3.5 Analysis of Solutions and Countermeasures 127
4.3.6 Discussion 131
4.4 Restoration 135
4.5 Context-Awareness and Situational Awareness in CPCS 137
4.5.1 General Architecture and Technologies 137
4.5.2 Context and Situation Awareness Approaches 139
4.5.3 Suitability of Detection Approaches for Smart Grid Domains 141
4.5.4 Adequacy of Awareness Mechanisms of Prevention, Response and Restoration in the CPCSs of the SG 145
4.6 Summary 148

5 Design of interoperable and Secure CPC Systems 151
5.1 Definition of the Application Scenario and Objectives of the Design 151
5.2 Background and Technologies for the Design 155
5.3 Design of the Secure Interoperability Architecture Model 159
5.3.1 Access Control Mechanisms 161
5.3.2 Cloud Infrastructure 162
5.3.3 Fog Infrastructure 164
5.3.4 Software Defined Networking 166
5.3.5 Defense in Depth 167
5.3.6 Diagnosis System 168
5.3.7 Delegation Mechanisms 169
5.4 Design of the Services Modules 173
5.4.1 Authentication and Access Manager 173
5.4.2 Policies Manager 176
5.4.3 Monitoring Manager 179
5.4.4 Accountability and Backup 181

5.4.5 Maintenance Manager . . . 182
5.5 Analysis of the Operation of the Secure Interoperability Architecture . . . 182
5.5.1 Local Communication . . . 183
5.5.2 Global Communication . . . 184
5.5.3 Cases of Example of the Operation of the Platform . . . 185
5.6 Summary . . . 187

6 Experimentation and Validation of the Architectural Model 191
6.1 Scope of the Validation . . . 191
6.2 Design of the experiment: Scenario and Validation Cases . . . 194
6.2.1 Validation Case 1: Protocol Converter . . . 196
6.2.2 Validation Case 2: Monitoring Manager . . . 197
6.2.3 Validation Case 3: Authorization and Access Manager . . . 200
6.3 Definition of the Testbed: Modules and Libraries . . . 201
6.3.1 Mininet . . . 202
6.3.2 POX OpenFlow Controller . . . 202
6.3.3 Modbus/TCP Protocol . . . 203
6.3.4 IEC 60870-5-104 Transmission Protocol . . . 203
6.3.5 IEC/TS 62351-8 - RBAC Standard . . . 205
6.3.6 Other Tools and Resources . . . 205
6.4 Experimentation Results . . . 206
6.4.1 Protocol Converter . . . 207
6.4.2 Monitoring and Accountability Managers . . . 212
6.4.3 Authorization and Access Manager . . . 219
6.4.4 Performance Analysis . . . 222
6.5 Discussion . . . 228

7 Conclusions 233
7.1 Motivation and Contextual Framework . . . 233
7.2 Contributions . . . 234
7.3 Open Research Issues and Future Work . . . 235

A Resumen 237
A.1 Marco de la tesis, objetivos y contribuciones . . . 238
A.2 Ataques avanzados a los sistemas de control ciber-físicos . . . 240
A.3 Análisis de requisitos para la interoperabilidad segura de sistemas de control ciber-físicos . . . 244
A.4 Servicios para la interoperabilidad segura de sistemas de control ciber-físicos . . . 245
A.5 Diseño de una plataforma de interoperabilidad segura para sistemas de control ciber-físicos . . . 247
A.6 Experimentación y validación del modelo arquitectural propuesto . . . 251
A.7 Conclusiones y trabajo futuro . . . 252

Bibliography 255

List of Figures

1.1 Evolution of the industry and control systems generations, adapted from [1] and [2] 2

2.1 Stages of a stealth attack 20
2.2 AICAn taxonomy 23

3.1 CCS requirements 46
3.2 NFR requirements for interoperability, a global view 50
3.3 Softgoals for the interoperability architecture and critical systems 51
3.4 Softgoals for interoperability 52
3.5 Satisficing techniques for the interoperability architecture 57
3.6 Satisficing techniques for the subjacent infrastructure 59
3.7 Satisficing techniques for the communication infrastructure 61
3.8 Satisficing techniques for interoperability 62
3.9 Satisficing techniques for the industrial IoT 63
3.10 Satisficing techniques for virtualization 64
3.11 Interoperability architecture for the CPCSs of the SG 65
3.12 CCS and IDS softgoals 79
3.13 Survivability, softgoals and satisficing techniques 83
3.14 Real-time performance, softgoals and satisficing techniques 84
3.15 Safety critical, softgoals and satisficing techniques 85
3.16 Sustainability, softgoals and satisficing techniques 86
3.17 Robustness, softgoals and satisficing techniques 86

4.1 Security services 94
4.2 The needs of online automation of a critical IDS system 106
4.3 Situational awareness for critical contexts, adapted from [3] 110
4.4 Methodological framework, containing a taxonomy of IDPRS, adapted from [4] and [5] 111

5.1 Secure interoperability conceptual infrastructure for the smart grid 154
5.2 Conventional networking vs. SDN 158
5.3 Secure interoperability infrastructure for the smart grid 160
5.4 Services offered by the controller modules 161
5.5 Services offered by the cloud 162
5.6 Main components of the secure interoperability infrastructure 165
5.7 Defense in depth and diagnosis system 168

5.8 Primary and secondary roles . . . 170
5.9 The IoT communications scenario . . . 186
5.10 The cyberattack communications scenario . . . 188

6.1 Simplified controller . . . 192
6.2 The NIST smart grid model [6] . . . 195
6.3 Testbed scenario for the secure interoperability infrastructure . . . 195
6.4 Converted traffic between nodes that use different communication protocols . . . 196
6.5 Disconnection attack . . . 197
6.6 Active eavesdropping attack . . . 199
6.7 Covert channel attack . . . 200
6.8 Authorization and access manager in the testbed . . . 201
6.9 Modbus packet . . . 203
6.10 IEC104 packet [7] . . . 204
6.11 Mininet testbed scenario illustrating the communications in the network . . . 206
6.12 Industrial protocols communications within the testbed . . . 207
6.13 Transparent interoperability proxy . . . 208
6.14 Proxy protocol conversion . . . 209
6.15 Proxy utilization in the testbed . . . 210
6.16 Notification of alarms to the HMI in the disconnection attack to the testbed . . . 213
6.17 IEC104 ASDU [7] . . . 216
6.18 IEC104 ASDU containing covert channel . . . 217
6.19 Authorization Manager utilization in the testbed . . . 219
6.20 Role-based connection rejection in the testbed . . . 220
6.21 Authorization and access manager . . . 221
6.22 Testbed traffic graph . . . 222
6.23 Proxy activation traffic graph . . . 223
6.24 Flow diagram for the Interoperability proxy . . . 224
6.25 Throughput graph of the transparent proxy utilization between the IEC104 client and the Modbus/TCP server . . . 225
6.26 Throughput graph of the regular connection established between the IEC104 client and the IEC104 server . . . 226
6.27 Roundtrip time graph of the traffic in the testbed networks . . . 228
6.28 RBAC authorization traffic . . . 229

A.1 Evolución de la industria y de sus sistemas de control a través de las generaciones, adaptado de [1] y [2] . . . 238
A.2 Taxonomía AICAn . . . 241
A.3 Servicios de seguridad . . . 246
A.4 Infraestructura para la interoperabilidad segura del smart grid . . . 248
A.5 Servicios ofrecidos por los módulos controlador . . . 249
A.6 Servicios ofrecidos por el cloud . . . 250
A.7 Defensa en profundidad y sistema de diagnóstico . . . 251

List of Tables

2.1 Cyber stealth attacks and their relation with AICAn 33
2.2 Cyber stealth attacks summary table 37

3.1 Architecture softgoals definitions table 54
3.2 Satisficing techniques for interoperability 58
3.3 Maintainability metrics 68
3.4 Reliability and availability metrics 69
3.5 Performance metrics 70
3.6 Safety and dependability metrics 70
3.7 Responsiveness metrics 71
3.8 Auditing metrics 72
3.9 Secure interoperability metrics 72
3.10 Virtualization metrics 73
3.11 Monitoring and restoration metrics 74
3.12 Authorization and delegation metrics 76
3.13 Requirements for protection systems definitions table 80
3.14 Satisficing techniques 81
3.15 Metrics for protection systems 85

4.1 Review of several systems according to the automation and knowledge dimensions. 107
4.2 Classification of the existing IDPRS solutions according to our taxonomy 128
4.3 Taxonomy of intrusion response actions 129
4.4 Classification and characteristics of the restoration techniques 136
4.5 Protection approaches applied in smart grid environments 140
4.6 Detection approaches in relation with the SG characteristics and requirements . . 143
4.7 Classification and characteristics of the response mechanisms 147
4.8 Adapting WASA techniques to cyber-physical environments 148

5.1 Cloud vs. fog features, extracted from [8] 157

A.1 Tabla resumen sobre los ciberataques camuflados 243

Chapter 1

Introduction

1.1 Evolution of the Control and the Interoperability

The evolution of the industry through the years have made industrial processes intimately linked to *Information Technologies*, from this union, it is of particularly interest the emergence of *Control Systems* (CSs) as methods to manage and supervise the industrial processes. CSs present many challenges from the point of view of cybersecurity and interoperability of the systems, which are current hot topics of research. This thesis focuses on the analysis of the cybersecurity and interoperability processes for the control systems of the industry. Of particular interest to us is the analysis of the control systems of *Critical Infrastructures* (CIs). CIs are industrial infrastructures which provide the most necessary services to society, so their continuous correct operation is of paramount importance.

To contextualize our research, it is important to describe the current industrial frame and trends, which shape the design and characteristics of the control systems and therefore our investigation. The work of this thesis started in the frame of the *third industrial revolution*, and has evolved through the appearance of a new paradigm in the recent years, adapting its contents and characteristics to the newly evolving scenario. This industrial frame has been introduced as the *Industry 4.0*, also called the *fourth industrial revolution*, which refers to an industrial movement currently taking place, and which is guiding the industry towards adopting certain new technologies and paradigms for their modernization. As it is possible to observe in Figure 1.1, this industrial revolution has been preceded by three other industrial revolutions in the history of mankind. The first one coincided with the introduction of water- and steam-powered mechanical manufacturing facilities at the end of 18th century, which use was intensified throughout the 19th century [9].

The second industrial revolution followed the introduction of electrically-powered mass production based on the division of labour, around the 1870s. While the third industrial revolution, or the "*digital revolution*" took place around the 1970s until recent years, when the industry started using electronics and information technology to achieve further automation of manufacturing [10]. Finally, the fourth industrial revolution became publicly known in 2011 under the name of "*Industrie 4.0*", in order to improve the industrial processes and competitiveness of the German industry as part of the 2020 strategy for Germany [10]. This new industrial framework

envisioned a paradigm shift from the traditionally centralized processes to a decentralized architecture. The first approach to tackle this objective is centered around three key components and technologies, i.e., the *Internet of Things* (IoT), *Cyber-Physical Systems* (CPSs), and *Smart Factories* [10].

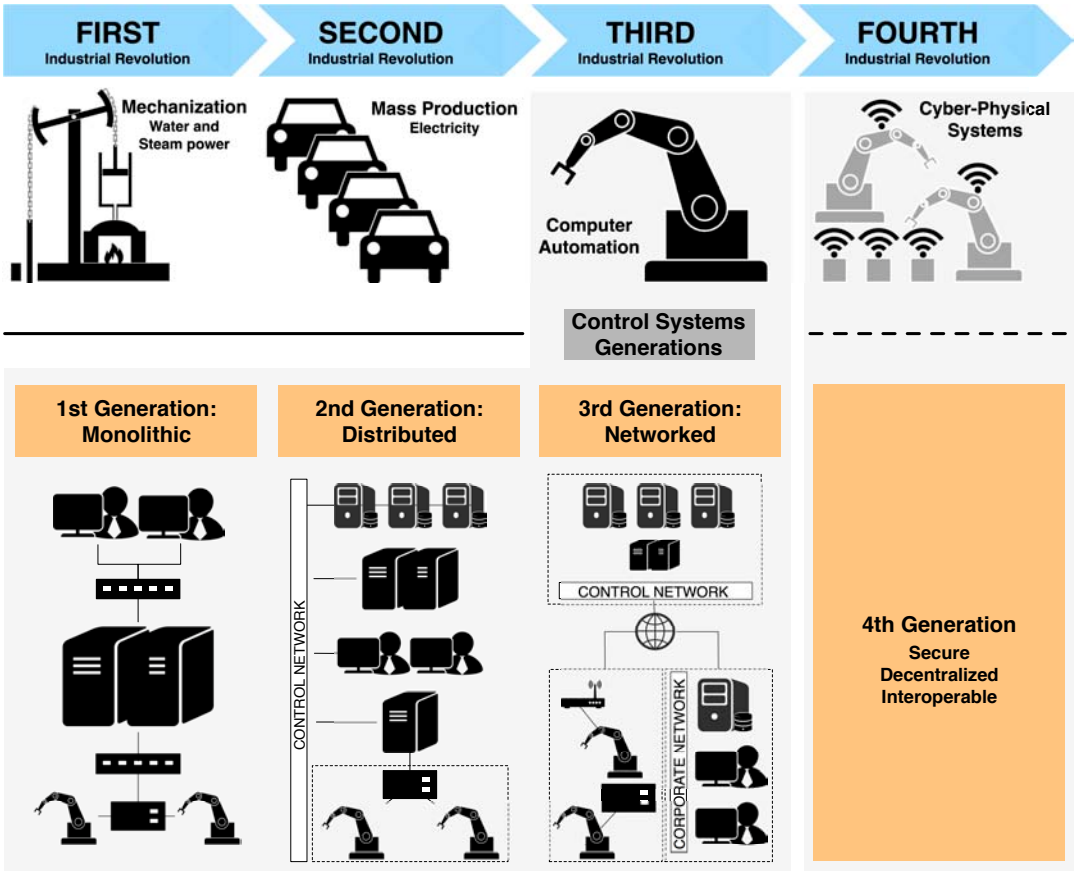


Figure 1.1: Evolution of the industry and control systems generations, adapted from [1] and [2]

A more elaborated concept included the following design principles: *interconnection* and interoperability, *information transparency*, *decentralized decisions*, and *technical assistance* [10]:

- In an environment where different machines, devices, sensors, and IoT "*things*" coexist and need to interact, new challenges arise with respect to their interoperability and cooperation capabilities. Wireless communication technologies, standardized procedures and protocols, compatible interfaces and processes, enabling technologies such as cloud computing, are examples of technologies and guidelines suitable for achieving interoperability and successful interconnection and cooperation in this new scenario.

- Since a key component of this new paradigm is the introduction of CPSs, it is possible for the virtual world to access to information of the physical world. This characteristic generates a generation of context-aware systems, capable of analyzing their environment and making decisions based on different sources of data. Therefore typical control information can be aggregated to context information for interpretation in a transparent way.

- Decentralization is key to this new industry paradigm, where the affluence of different types of devices and systems (CPSs, IoT things) to the different networks of the industry build a complex scenario for the interoperability and decision making, therefore intelligent and autonomous decentralized decision-making procedures are of great importance in order to accelerate and improve the problem solving in this environment, where local virtual and physical-world information can provide enough data to make well informed decisions.
- There is a current migration of the functions of humans in the industry, they evolve from machine operators towards strategic decision makers, system administrators and problem solvers. With the increase on automation, and the inclusion of sophisticated automation intelligent processes which can tackle the complex processes of the industry, human workers are now in charge of the maintenance and management of the industrial systems and processes. Current control systems and monitoring infrastructures now can gather relevant information about the behavior and dynamics of the industrial infrastructures, complementing them with contextual information about the physical world provided by the CPSs. This, therefore, results in efficient industrial procedures which are well structured in order to provide important information for rapid problem solving and maintenance tasks.

We find, therefore, that control systems and monitoring processes are also key elements for the correct operation of the industry. Together with the discussed technologies and principles, we understand that one of the main challenges of the Industry 4.0 is adapting the control processes traditionally used during the third industrial revolution, to this new scenario and paradigm. As we can see in Figure 1.1, control systems have also evolved over the years, to accommodate to the new arising needs of the industry. It is the focus of our interest to better understand the characteristics and capabilities of these control systems, and the new requirements they need to comply with in order to tackle the challenges the Industry 4.0 presents.

Thus, we position our research in this transition stage which represents the need to adapt the existing systems (control systems in particular) to the newly appeared technologies and solutions. Although in this tesis we discuss Industry 4.0, we do not claim an strict adherence to the actual Industry 4.0 paradigm, since this framework emerged during the course of our work. However we adapt our solutions and objectives to observe some of the new principles and characteristics brought by this new industrial revolution and, for simplicity, hereafter we refer to our context framework as Industry 4.0. To better understand and characterize our research, we will not focus on the control systems of a generic industry, rather we will contextualize our research on the control systems in charge of the *Smart Grid* (SG) [11], which can be viewed as the intelligent version of the traditional electric grid, where the energy industry meets the Industry 4.0 paradigm. Therefore, in the remainder of this section, we analyze the control systems' evolution through their generations, and identify the main challenges and characteristics they need to provide in order to successfully operate in an Industry 4.0 context.

1.1.1 Control Infrastructures and Generations

To better understand control systems, we can define a CS as a system that performs the management and the regulation of behavior of other devices or systems. In particular, *Industrial Control Systems* (ICSs) are a type of control system deployed to aid the operation of industrial infrastructures such as electrical, water, oil, gas and data. Examples of such ICSs are *Super-*

visory Control And Data Acquisition (SCADA) systems and the *Distributed Control Systems* (DCSs). ICSs are deployed in multiple types of industries, but when they serve as assistance to infrastructures essential for the well being of the society, i.e., CIs, they are considered critical infrastructures in themselves, and they are denominated *Critical Control Systems* (CCSs).

Since first introduced in the 1960s, CSs have evolved through three main generations, i.e., *Monolithic*, *Distributed* and *Networked*, until the present day. Figure 1.1 depicts in orange the course of these generations, which start with the third industrial revolution. The main characteristics of each generation can be outlined as follows [1, 12]:

- *First Generation - Monolithic*: were the first CS's architectures designed, based on a centralized control mainframe system. In this design, the control mainframe was usually composed of two systems: one being the primary control unit, and the secondary one was in standby in case the primary failed. They were in charge of performing the supervisory and data acquisition tasks for the whole system, i.e., registering critical data and managing the monitoring processes in the system. The field CS managing the industrial processes in the CI was built using *Remote Terminal Units* (RTUs) connected to the central mainframe. These RTUs and field equipment had constrained capabilities (e.g., 8-bit microprocessor with 4-16 KB RAM), and implemented low data transmission rates for their communication with the mainframe (e.g., telephone, radio, automation protocols such as for instance Modbus serial or IEC-101), which provided limited functionalities to the CI.

- *Second Generation - Distributed*: the second generation of CSs incorporated a set of new technologies based on IP addresses which allowed the control processes to be distributed among different systems, e.g., *Master Terminal Units* (MTUs), control workstations, *Human-Machine Interfaces* (HMIs) or dedicated servers (configuration server, database server, applications server, historian). The implementation of distribution of control processes improved the control tasks robustness, pushing the primary/standby configuration of the monolithic mainframe to the background, where this configuration remained for backup purposes only. In terms of communications with the field CS, the networks evolved to the use of large *Local-Area Networks* (LANs), controlled by MTUs. Additionally, RTUs technologies advanced to include faster microprocessors and bigger memories, which enabled the remote devices to perform more advanced and autonomous processes.

- *Third Generation - Networked*: the latest advances to CSs are presented in the third generation, which is present in most of today's CIs. The networked generation breaks with the isolation concept traditionally implemented in the CIs in the previous generations, introducing network designs which use TCP/IP protocol (Transmission Control Protocol/Internet Protocol) to establish connections to the different segments or networks of the CIs via the Internet. These connections made possible the implementation of real-time monitoring processes and communications, multiple sessions, and interconnectivity. Technical advances in the field CSs provided new capabilities for the remote devices, especially regarding communication among field devices, enabled by the TCP/IP networks. New control protocols (e.g., Modbus/TCP [13], DNP3 [14], IEC104 [7]) and wireless technologies emerged for their use within critical systems, facilitating fast control and maintenance tasks in the field networks and enabling remote operations within the CI. However, most of these network protocols and systems lack basic security mechanisms, e.g., authentication

and encryption mechanisms, which, together with the discontinuation of use of the isolation paradigm, introduces new (and previously unknown in this context) vulnerabilities within the CIs.

CCSs, as previously discussed, are critical systems which correct operation is of paramount importance. The need of protecting and securing these systems are well recognized and covered in multiple guidelines and recommendations issued by many countries around the globe, where the different nations establish varied protection measures for their CIs. In Europe, the European Commission provides guides with the COM(2011) 163 [15], the COM(2009) 149 [16] and the guidelines towards Europe 2020 [17]. It is of particular importance the presence of the guides of *Situational Awareness* (SA), and the guides for preparedness and response [6, 15].

Taking into account, on the one hand the aforementioned security and protection recommendations for CSs and the discussed security problems inherent to the networked generation of CSs, and on the other hand the new challenges and the change of paradigm in the industrial sector brought by the Industry 4.0, there arises the need to create a new architecture design for the CSs capable of tackling this new scenario. This new architecture would constitute the *Fourth Generation* of CSs, as we can see in Figure 1.1.

The main characteristics we envision for this new fourth generation of CSs in a setting of similar characteristics to Industry 4.0 are the (i) *security* intrinsic to the design, (ii) *decentralized* control processes, and (iii) ensured *interoperability* among the heterogeneous components coexisting in the scenario (see Figure 1.1). As we discussed in the previous section, the tendencies of the new industrial revolution are to introduce advanced technologies and paradigms that need to be accommodated to this new situation, and tackled from the control point of view. Therefore, the inclusion of technologies such as the industrial IoT and CPSs within CCSs needs of an infrastructure flexible enough to take advantage of these decentralized infrastructures, while technologies such as the cloud can give support to the interoperability processes, which always contemplates the security of the system from the point of view of design. It is therefore, a hot research challenge to provide architectural designs which take all these aforementioned considerations into account for building this new fourth generation of CSs, something we try to address throughout this thesis.

1.1.2 Cyber-Physical Control Systems: Technologies and Protocols

Cyber-physical systems, as reviewed in Section 1.1, are collaborative systems composed of autonomous and intelligent devices capable of managing data flows and operations, and monitoring physical entities integrated as part of CIs. The interaction between cyber-physical devices is carried out through large, heterogeneous and interconnected communication infrastructures. Through these infrastructures, CSs can process and manage measurements and evidence produced in remote locations, as well as distribute and visualize control transactions. The interfaces that lead these activities are generally control devices with the capacity to ensure the intermediation tasks between the acquisition world and the control world, such as gateways, servers or RTUs. An RTU, typically working at 22MHz-200MHz with 256 bytes-64MB RAM, 8KB-32MB flash memory and 16KB- 256KB EEPROM, serves as a data collector or a front-end to reach remote substations equipped with sensors or actuators responsible for executing a specific action in the field.

Cyber-physical devices can be categorized according to their functional capacities: *weak*, *heavy-duty* and *powerful-duty* [18]. We can describe these clases in the context of the SG, where within the class weak, we identify extremely constrained devices but with sufficient competences to run simple operations, such as 4MHz, 1KB RAM and 4KB-16KB ROM (e.g., home-appliances, sensors). Devices belonging to the heavy-duty category are relatively expensive devices (e.g., handled-devices, smartphones) that are able to execute any simple or complex critical action. Their microprocessors are quite potent, working at around 13MHz-180MHz, 256KB-512KB RAM and 4MB-32MB ROM, and within this category, we highlight the role of the RTUs, smart meters (8-50MHz, 4KB-32KB RAM and 32-512KB flash memory) or industrial *Wireless Sensor Networks* (WSNs). Industrial sensors usually present slightly greater capacities than conventional ones, equipped with a 4MHz- 32MHz, 8KB-128KB RAM, 128KB-192KB ROM, and with the ability to protect the observed infrastructure through their sensorial modules and manage data streams. Finally, the powerful-duty class contains all those devices with significant capacities to address any action or application, such as servers, proxies or gateways.

A particularization of CPSs are the *Cyber-Physical Control Systems* (CPCSs), which we can define as those CPSs devoted to perform control tasks. It is possible to find a correspondence of the CPCSs in Figure 1.1 with the CSs in their fourth generation. Given this consideration, these control systems need to provide the three characteristics previously mentioned, i.e., *security*, *decentralization* and *interoperability*. Regarding the functional capabilities of the CPCSs in themselves, it is considered that they can be classified as the aforementioned CPSs (into weak, heavy-duty and powerful-duty), however, since they need to enable the decentralization capabilities necessary for an Industry 4.0 context (see Section 1.1), we consider that the CPCSs need to be medium to powerful-duty devices, capable of running advanced software for control tasks, interoperability and local decision making. Other CPS devices belonging to the IoT make an important part of the Industry 4.0, but their control functions are reduced or assigned to the more powerful (CPCS) devices.

In terms of security, the technological competences of current CPCSs consist of minimal protection services that traditional situational awareness systems demand, such as perception of the environment observed, the comprehension of their meaning and their projection in the near future, initially introduced by Endsley in [19]. However, SA solutions for critical control applications deployed on large distributions can become insufficient since local protection through dynamic services should also be required to ensure one of the eight priority areas defined by the *National Institute of Standards and Technology* (NIST) in [20].

This priority area, known as *Wide-Area Situational Awareness* (WASA), not only focuses on monitoring critical components and their performance at all times, but also on automatically anticipating, detecting and responding to unplanned faults, and if necessary to restore states before major disruptions can arise. This also means that WASA strategies should be consolidated in a methodological process that helps the underlying system extract, interpret and respond to threatening situations, as proposed by Alcaraz and Lopez in [3]. But even so, this work neglects the relevance of studying preventive and corrective measures taking into account the properties of the context, the features of the underlying system and its technological capacities. Therefore, additional work on the security of CPCSs is necessary in order to address the aforementioned challenges for cybersecurity arising from the implementation of the characteristics of the Industry 4.0 paradigm [9]. In the next sections we further analyze the problem of security in this industrial scenario, in particular applied to CSs due to the criticality of these systems.

The last feature we need to review for CPCSs according to the characteristics we have elicited for control systems in a similar framework to the Industry 4.0 (see Figure 1.1) is the interoperability. Interoperability for CCSs is an open issue for research, which is currently the focus of intense investigation. We devote next section to review the approaches and efforts for interoperability in CSs that are present in the literature.

1.1.3 A Review on Interoperability

Interoperability is defined as the characteristic of a product or system to work and cooperate with other products or systems without restrictions. This desirable feature enables large heterogeneous infrastructures to be able to operate correctly in a cooperative way. Hence, this is a vital property for the operation of CCSs in CIs. Especially important is the interoperability for the new generation of CSs as reviewed in Section 1.1, where the new industrial scenario evolving towards the Industry 4.0 challenges the established procedures by introducing new technologies within the CI, such as CPSs and industrial IoT devices, where they all need to work in a cooperative efficient way, paying special attention to fulfilling the requirements and constraints of operation of the critical systems [21].

Achieving interoperability in the context of critical systems is currently a trending problem object of intense research by the scientific community and the institutions worldwide. From the intensification of efforts to provide a more interoperable smart grid [6], and the establishment a more interconnected health care system [22], to the need of achieving more interconnected and interoperable first communications systems to protect CIs by first response teams [23], the need to achieve interoperability of CSs is rapidly rising. Governments are also urging the industry to take measures to protect their CIs through a strong investment in cybersecurity, and coordination strategies, where interoperable systems are essential tools [24].

Since interoperability is a major concern for the Industry 4.0, where devices and technologies such as the Industrial IoT need to achieve interoperability between devices and machines that use different protocols and have different architectures, there have been created several working groups gathering the efforts of the industry to tackle this problem [25]. Organizations such as the *Industrial Internet Consortium* (IIC) or the *Open Connectivity Foundation* (OCF) among others are joining efforts to create an adequate industrial scenario where security and interoperability are considered from the design point of view, and allows these new technologies to be correctly integrated and exploited. Examples of these are *Industrial Internet Reference Architecture* (IIRA) [25] and the *Industrial Internet Security Framework* (IISF) [26] which explore interoperability and security platforms from the point of view of the industrial internet and the industrial IoT. The IIC proposes a three-tier architecture pattern (edge, platform and enterprise) which combines the presence of a layered databus with a gateway-mediated connectivity for the management of the interoperability and interconnection of the different actors in the platform.

However, despite the increased efforts of the industry, governments and academia, still few other solutions and approaches have been proposed, most of them focusing on the simulation and modeling of interdependences, rather than tackling interoperability itself. In [27], C. Alcaraz et al. propose a system that enables interoperability among the distributed systems belonging to the SG. This approach is based on a context-aware policy enforcement system which uses

the role-based access control model defined by the IEC/TS 62351-8 standard [28]. This interoperability infrastructure enables the connectivity of different technologies belonging to different owners and manufacturers in three stages: authentication, authorization, interoperability. The systems participating in the architecture are supposed to implement varied access and security policies. Thus, the connectivity among these heterogeneous systems is achieved through a network architecture based on a supernode containing a policy decision point. This system is capable of taking interoperability and access decisions by means of an expert system capable of denying or establishing connections using contextual information and authorization and access information.

Another approach to interoperability is provided by the development of the European project DIESIS [29], which analyze interoperability from a simulation point of view, through the connection of three different CI simulators. The interoperability approach of this work is provided through the use of ontologies, and a communication middleware for distributed simulations. The system implements ontologies at three levels, i.e., federation, infrastructure domain, and simulator, which formalize the conceptualization of CI domains. Additionally, the middleware acts by allowing the propagation of effects from domain to domain, a capability implemented by rules that specify the way each two interconnected objects interact.

Further efforts on the DIESIS simulation are presented in [30], where the authors focus on the implementation of a demonstrator for the approach of CI simulators interconnection. In this paper, through a proof of concept, the authors aim to analyze the behavior of the previously developed communication middleware when coupling heterogeneous simulation systems, using various time and data models. The system also integrates the ontology-based knowledge based system, and the simulations for the interoperability are performed arbitrarily using pair-wise or group-wise coupling.

Other works related to the interoperability of critical systems come from the field of interdependencies modeling. Works such as [31, 32, 33] focus their study on the identification, modeling and simulation of interdependencies existing in critical infrastructures. This area of knowledge aims to understand the infrastructures existing between CIs and their implications in terms of security. Also, outside critical systems, there are multiple active research efforts to tackle interoperability between information technology systems, where different platforms and services can be used for this purpose, e.g., OSGi [34]. They are, however, difficult to apply to CIs due to the special characteristics and constrains of these systems. Therefore, we find that interoperability for CIs is a multidisciplinar area which presents high complexity due to vastness of the systems considered, the complicated nature of their interdependences, and the varied nature of their interconnections (physical, cybernetic, geographic, logical interdependencies).

1.2 Security Issues: Lessons Learned and Risks

Security issues, especially cyber-security ones, as we reviewed in Section 1.1.2, are a critical point to address in current CSs, as well as an important characteristic to address in the design of the new generations of CSs. *Information and Communication Technologies* (ICTs) have now become essential elements in our society since they offer significant improvements in efficiency, cost reduction and enhancing quality of life. Mobile computing technologies, embedded systems,

smart devices, wireless communication and the growth of the Internet are becoming the major driving forces. These enable management of information from anywhere, at any time and anyway, allowing an easier implementation and quicker operation of the great majority of today's competitors' infrastructures and their services [35]. In fact, most of these physical facilities are highly interconnected to other national (and international) systems through communication systems, and managed through software-based systems, where the atomic data are not only the integral elements of the infrastructure itself but are also needed between infrastructures in order for them to function properly [1, 21].

Critical infrastructures are interconnections of a set of systems and assets, whether physical or virtual [1], which are integral to the social, political, and economic life of a nation and its citizens. Examples of these infrastructures can be water treatment systems, energy generation and distribution systems, finance, borders, transportation, etc. In policy terms, the European Union (EU) considers critical infrastructures to be "*those physical and information technology facilities, networks, services, and assets. the disruption or destruction of which has which, if disrupted or destroyed, would have a serious impact on the health, safety, security or economic wellbeing of citizens or the effective functioning of governments in the Member States*" [36]. Similarly, the United States (US) government considers critical infrastructures as those "*systems and assets, whether physical or virtual, so vital to the United States that the incapacity or destruction of such systems and assets would have a debilitating impact on security, national economic security, national public health or safety, or any combination of those matters*" - extract from Law 107-56, Section 1016, entitled critical infrastructure protection act of 2001 [37].

Any protection put into place to safeguard CIs should focus on preserving not only the physical elements of the infrastructure but also and most importantly its virtual (cyber) elements, as a disruption of these assets may trigger the same damage as the disruption of physical components, putting the security and safety of these interconnected systems at risk. In order to guarantee that CIs operate continuously, they are monitored by CSs to ensure the correct performance of processes and operations. In the current industry, these control and monitoring operations are performed by SCADA systems, which are composed of hybrid integral systems where a set of control processes is widely distributed over large geographic locations, but any information has to be centralized at a single point, the SCADA center. Remote substations comprise smart collectors (field devices) capable of interpreting ingoing/outgoing traffic, of sending information to the SCADA center or executing control actions in the field. These devices, widely known as *Programmable Logic Controllers* (PLCs) or RTUs, are connected to sensors in charge of perceiving measurement values (e.g., pressure) or actuators to carry out an action.

Given these characteristics and the intrinsic criticality of the CSs, any physical or virtual disruption related to communication or control may have devastating consequences for the continuity of services and business. Government and industry entities are already announcing the importance of addressing aspects of cyber-defense in their respective critical sectors, where control systems are in the sights of potential attackers [38, 39].

1.2.1 Cyber Stealth Attacks in Critical Infrastructures

One of the most dangerous threats that CIs face are cyber-attacks, where adversaries can remotely perform malicious acts that may have a disastrous impact on the infrastructures. This, together with an increasing number of threats, faults and errors registered, have alerted institutions worldwide [38, 39, 40]. There are annual reports published by the different governments through specific organizations such as the European Union Network and Information Security Agency (ENISA) [41] and the Industrial Control System Cyber Emergency Response Team (ICS-CERT) [42, 43, 44], reflecting the current situation and the severity of potential threats. The number of specific incidents apparently continues to grow, requiring a major effort to establish security and protection measures immediately.

ENISA's work on managing incidents [41] in conjunction with the National Regulatory Authorities (NRAs) of the 28 EU member states was established in 2012 thanks to Article 13a of the framework Directive (2009/140/EC) [45]. According to the two latest reports, the number of incidents caused by natural disasters, human error, malicious actions, system faults and third party faults, and registered in the different sectors has already reached significant numbers. The majority of them targeted communication networks (51 in 2011 and 79 in 2012) based on fixed telephony (e.g., VoIP over DSL, cable, etc.), fixed Internet (e.g., dial up, DSL, cable, etc.), mobile telephony (e.g., UMTS, GSM), mobile Internet (e.g., UMTS, GSM). With very similar goals, ICS-CERT via the Critical Infrastructure Information Act (the CII Act) of 2002 manages incidents from owner organizations of CIs.

According to ICS-CERT, the number of incidents became more noticeable in 2010, the year in which information technologies started to be well-known, in which active remote accesses (e.g., Internet connections, connection to sub-networks, use of wireless technologies) also started to be exploited. The power grid industry is leader in the number of detected incidents (18 in total), followed by nuclear, chemical and water management, which received between 8 and 15% of the threats. The majority of the incidents reported were related to SSH (Secure Shell), brute-force attacks, scanning and spear-phishing (2 out of 3 attacks) in the power grid with the aim being to acquire credentials or personal information. As we can appreciate, one of the most dangerous threats that CIs face are cyber-attacks, where adversaries can remotely perform malicious acts that may have a disastrous impact on the infrastructures [46]. This is especially true when these cyber attacks target CIs and the adversaries' objective is to remain unnoticed while pursuing their goals, and so we face *stealth attacks*, a sophisticated and potentially very dangerous type of cyber attack. Usually these attacks are launched by powerful adversaries with the objective of extracting sensitive or reconnaissance information without being noticed, to sometimes, afterwards, use this information to launch malicious attacks to cause disruptions to CIs. Some examples of these attacks, perpetrated in 2010 are:

- CIKR Mariposa [47]. Mariposa was a botnet, performing operations of denial of service attacks, e-mailing spam, personal information theft, modifications in the web-browser's searches, and other similar cyber-attacks.
- Stuxnet worm [48]. The first malware code designed specifically for engineering controllers (i.e., PLCs/RTUs). The worm, with the ability to infect numerous network devices without leaving evidence of the attack, was primarily focused on reaching and manipulating critical sections of a particular PLC of Siemens. The origin of the infection was traced back to

the unsuitable use of personal media devices (USB drivers).

In 2011, 197 reports of incidents were received; the water sector, topping the list with 81 incidents. Many of the reported incidents were related to spear-phishing for illicitly obtaining security credentials or unauthorized access to restricted systems, as well as other relevant attacks such as:

- Night Dragon attack [49]. Attack reported by McAfee, which was based on a combination of a set of potential threats (e.g., social engineering) and malware (e.g., Trojans) to breach the security of corporate networks in charge of managing control systems.
- Nitro Attacks [50]. Sophisticated attack that involved several companies in the chemical sector, primarily private companies involved in research, development, and manufacture of chemicals and advanced materials. The attack aimed to collect confidential data, and infected machines in the order of 27% in the USA, 20% in Bangladesh, 14% in United Kingdom, 6% in Argentina, 4% in Singapore, 4% in China, Taiwan, Germany and Czech Republic; 2% in Hong Kong, India, Netherlands and Finland; 1% in South Korea, France, Russia, Japan, Sweden, Norway, and Canada.
- Duqu [51]. Virus considered to be a mutation of Stuxnet but without the ability to self-replicate. Despite this feature, Duqu is able to reveal private information, configurations and accesses and has a similar behavior to Flame, described below.

The number of incidents remained equally high in 2012 with 198 registered [43]. 41% of the threats targeted the energy sector and its control systems, and the water sector witnessed the second highest number of incidents with 15% of the threats. The report of 2012 also noted two important aspects. On the one hand, systems connected to the Internet and protected through weak or default credentials were those that most received the most common attacks on the Internet; and on the other hand, more and more the water sector was becoming a specific target for attackers. This report presented some specific examples, such as the case of the water utility located in Springfield (Curran-Gardner public water district), which was attacked from an IP address located in Russia without leaving any evidence of this intrusion in the SCADA system. Other examples of cyber-attack are:

- Flame [52]. Worm originally designed to open back doors, infect and modify functions, in addition to stealing confidential data, destroying information or recording conversations.
- Gauss [53]. Cyber-spionage toolkit designed to steal data from individuals in the Middle East. It gathers passwords (banking credentials, browser cookies, etc.) working in a similar way to Flame.

In 2013, ICS-CERT received roughly 200 incidents [44]. The highest percentage of incidents was found to be in the energy sector (53%) followed by critical manufacturing (17%). The majority of these incidents were related to cyber-attacks such as watering hole attacks (with the intention of attacking those strategic points (e.g., servers, websites) that are frequently visited by targets), SQL (Structured Query Language) injection, and spear-phishing attacks. In the first quarter of 2014, the ICS-CERT reported attacks mainly on the energy and water sectors, followed by the transportation sector, where the main vulnerabilities targeted were weaknesses and flaws in the design of the systems [54].

In the following years, the number of cyber attacks have slowly increased in number and sophis-

tication, becoming a widespread threat to the world's CIs. Of particular importance is the fact that in December 2015, the ICS-CERT registered the first known cyber attack to have a physical impact to the power grid, becoming the first documented cyber-physical attack against a CI [55]. This Ukraine power grid cyber attack made use of spear-phishing emails with the BlackEnergy malware in order to control the SCADA system and switch off substations and different control devices (modems, RTUs, etc.), as well as to destroy information using the KillDisk malware [56].

According to the ICS-CERT, in the next year, 2016, 290 incidents were reported, where the critical manufacturing sector accounted for 63 incidents, the communications sector had 62 incidents and the energy sector had 59 incidents [55]. Spear phishing represented 26% of these incidents and network scanning and probing accounted for 12%. According to ENISA [57], the top threats for the last couple of years include malware and web based attacks. As we can see this phenomenon is escalating, in mid 2017 the WannaCry ransomware attack was launched, infecting more than 230,000 computers in 150 countries. This cyber attack used the WanaCrypt0r ransomware which takes advantage of a Microsoft vulnerability (specifically the CVE-2017-0143 (MS17-010) vulnerability in components of the SMBv1 service - port TCP 445) [58] in order encrypt the user's files, demanding payments in the cryptocurrency bitcoin to decrypt them [59]. This attack affected multiple types of infrastructures and companies such as manufacturing companies, oil refineries, city infrastructure objects and electrical distribution network facilities, being Telefonica (the biggest Spanish telecommunications company) and the *National Health Service* (NHS) of England examples of affected CIs.

Through this review of recent attacks, we can readily identify the real danger behind stealthy adversaries, and the need to understand them better in order to prevent attacks and counteract them, especially in critical contexts. The concept of stealth attacks was introduced for conventional networks by M. Jakobsson et al. in 2003 [60]. They were described in the literature as those attacks in which the cost and visibility of the attacker have to be minimized. Cyber stealth attacks "*allow a skilled but not very powerful attacker to target communication networks in a way that makes it unlikely that he gets traced and caught*" [60]. This type of adversary has proliferated in recent years targeting critical systems, since the first known high-scale stealthy attacks on CIs (Mariposa, Stuxnet).

These incidents showed the characteristics and sophisticated capabilities of these types of attacks, and proved that it is possible to adapt stealthy techniques used for conventional networks to threaten critical scenarios. However, besides these highly complex attacks, we understand that it is also possible to take this same knowledge on stealth attacks from general-purpose networks to implement stealthy cyber attacks on CIs in a less complex manner, but with potential, equally harmful results. CIs, especially ICSs, have, over the years, added ICTs to their infrastructures, but they have not incorporated sufficient security mechanisms to protect them [1], so they have inherited many threats and weaknesses from traditional networks. This lack of strong security mechanisms opens the door to multiple types of cyber attacks against CIs, one of the most powerful being stealthy attacks. Our work is, to the best of our knowledge, the first attempt to undertake the analysis of this kind of stealth attack in CIs.

1.2.2 Cyber Attacks in Interconnected CPCSs

Both cyber attacks and cyber stealth attacks threatening CSs and CIs around the globe can be directly applicable to CPCSs and in general to the infrastructures transitioning into the new industrial paradigms. However, there are slight variations that can change the scenario of the cyber attacks and cyber stealth attacks to the CPC systems. This is due to the changes in the architectural design introduced by the new industrial models, where numerous and heterogeneous new technologies and types of devices are included within control networks, changing the focus of the CSs from a centralized perspective to a distributed one (see Section 1.1.1).

These changes introduced by the fourth generation of CSs (see Figure 1.1), have direct consequences on the security of the CIs and on how the CCSs can handle threatening events such as system faults and anomalies, and cyber attacks. On the one hand, this new paradigm brings positive changes which highly improve the security and defense mechanisms of the system, i.e., the use of a *decentralized* paradigm, and the incorporation of *security* by design within the system. On the other hand, characteristics such as the need for *interoperability* and the inclusion of new *technologies* (e.g., CPCSs, industrial IoT, cloud computing) result in additional security challenges that might inject new unknown security impacts and vulnerabilities within the critical systems.

In a more elaborated way, we can discuss that the introduction of decentralization for CCSs have a positive impact on the security and robustness of the system, given that distributed autonomous decision-making processes help the system take control decisions and actions even in the presence of threatening effects affecting a section of the network, the CSs can make use of their decentralization to re-route control tasks and responsibilities to other actors in the system. This characteristic can help palliate and rapidly mitigate the effects of cyber attacks and cyber stealth attacks (see Section 1.2.1).

Additionally, the inclusion of security within the CPCSs from a design point of view positively reinforces the strength of the CS in the face of threats such as cyber (stealth) attacks. As we previously discussed, traditional control systems up to the third generation did not usually introduce security mechanisms by default. CSs lacked even the most basic security processes such as authentication and encryption procedures. This is the case of widespread control protocols such as Modbus [13], where its version Modbus/TCP does not provide the measures to protect the confidentiality and integrity of the messages between master and slave devices [61]. In this case, lack of authentication mechanisms and integrity checks introduce numerous vulnerabilities within any system using this protocol.

In the third generation of CSs (see Figure 1.1), some providers add basic security services to some segments of their CSs, however they still largely coexist with legacy equipment and protocols unable to introduce security mechanisms. Therefore, the change of perspective introduced in this fourth generation of CSs, where the (cyber) security takes a relevant position in the design of the infrastructure, helps improve the security of the whole infrastructure and cope with cyber (stealth) attacks. In terms of the used protocols, the fourth generation will prioritize and encourage the usage of different and more secure control protocols, in order to align this CS layer with the requirements and practices of the Industry 4.0 for cybersecurity.

However, as aforementioned, when proposing a new design model for the architecture of the CSs in their fourth generation, it is necessary to consider that the introduction of new actors and

technologies within critical systems might cause unforeseen impacts to the CI. To take advantage of the new technologies offered by the Industry 4.0 without introducing new vulnerabilities into the critical setting, it is important to consider security by default. This way it is possible to provide a robust architectural design which provides interoperability with these new solutions and that can leverage all the positive characteristics provided by this new paradigm.

1.3 Goals of the Thesis

In the previous sections, we have given the initial motivation of this thesis, where we have presented the new research challenges introduced by the new generation of control systems and their architectural design. These newly arising problems are the driving factors that have shaped the research agenda of this thesis. As discussed in Section 1.2, given the extensive presence of multiple cyber threats targeting CIs, and specially their control systems, interoperability and cybersecurity must be included among the key characteristics that shape the design of future CSs. The achievement of these essential qualities can be accomplished by the preparation of a propitious control scenario which facilitates the inclusion of the technologies brought by the new industrial revolution within CIs. CPCSs, the cloud, IoT, and others, bring new operational methods into scene, thus to build secure and efficient CCSs for this new generation, it is necessary to take new methodologies and procedures into account.

We, therefore, aim to approach the construction of a secure interoperability platform for the control of the CIs, shaped by the characteristics of the fourth generation of control systems, and aligned with the Industry 4.0 by the consideration of its tendencies and the inclusion of several technologies belonging to this paradigm. The inclusion of these technologies, while fulfilling the purposes of security, decentralization and interoperability desired for the new generations of CSs, introduce new challenges in this scenario. Our ultimate objective with this work is therefore to contribute towards the resolution of these challenges, with the purpose of proposing a new solution approach for the architectural design of a control system architecture which contemplates all the characteristics and needs that the newly identified scenario brings to light.

It is possible to particularize these research objectives, by dividing them into different research targets. Since we intend to provide an architectural design for the fourth generation of CSs, it is necessary to take into account the environment of application of our proposal. We address this design goal in a setting dominated by the tendencies introduced by the new industrial revolution. Within this context, we focus on the needs of the CSs of the smart grid. We therefore have to clearly understand the scenario, analyzing the inherent characteristics, requirements and needs of this new setting, as well as the threats that can arise within this system from a point of view of cybersecurity.

Based on this preliminary research, it is therefore possible to determine the security mechanisms and services that our architecture needs to incorporate from a point of view of design. It is our aim to establish wide-area situational awareness within our design, therefore our efforts on the security research are directed towards this principle. Given the special focus on security brought by the fourth generation of control systems, and the criticality of the SG for society, carefully considering the security services implemented by the architectural model is one of the key pillars of our research.

Once established the security services for our model, we aim to design an architecture for the fourth generation of CSs. This model has a special focus on security and interoperability, also contemplating and including some of the technologies, protocols and services newly arriving to the industrial context, where they coexist and cooperate within the CI in order to provide the necessary services for the control of the SG. This focus on security and interoperability brings to scene an environment of interconnectivity and cooperation which highlights the interdependencies between critical infrastructures, to contemplate this characteristic in our design, we address the scenario of interoperability and cooperation among a federation of smart grid providers' infrastructures, which shape the outcomes of our research in this area.

1.3.1 Direct Contributions

In light of the research objectives described in the previous section, we provide a list of the main direct contributions of this thesis, which are closely related to the challenges and goals previously identified:

- We review the advanced threats that menace the operation of critical systems, in particular we focus on the cyber attacks and cyber stealth attacks that threaten the critical control systems of critical infrastructures, such as the smart grid. Since anomalies and system faults have been largely studied in the context of CIs, we focus our research on the analysis and understanding of these newly arising attacks to critical systems, and the possible mitigation and countermeasuring techniques that can help palliating their effects.
- We examine and identify the special requirements that constraint the design and operation of a secure interoperability architecture for the CSs (particularly CPCSs) of the SG. We focus on modeling the non-functional requirements that shape such infrastructure, following the NFR methodology to leverage essential requirements, satisficing techniques and metrics for our architectural model. We perform this process twice, firstly to extract the model for the secure interoperability, and later, the model for the protection system's architecture.
- We study the services needed for the secure interoperability of the CSs of the SG. This thorough review covers the security mechanisms from basic services to advanced procedures capable of tackling sophisticated cyber threats menacing the control systems at any level. Our analysis is divided into different areas, i.e., prevention, awareness and reaction, and restoration, which contemplate a robust security model for the protection of critical systems.
- We provide a designed architectural model for the secure interoperability and interconnection of the CPCSs of the SG. This scenario contemplates the interconnectivity scenario where a federation of SG providers interact through the secure interoperability platform in order to manage and control their infrastructures in a cooperative way. This platform takes into account some of the inherent characteristics and new technologies and services present in the new industrial scenario, which transits to the Industry 4.0 paradigm.
- We present a proof of concept of our architectural model, which helps validate the proposed design for the secure interoperability platform through experimentations. We devise a set of validation cases that test some of the main functionalities offered by the designed secure

interoperability system, providing insight about its performance, capabilities, and pending issues for the future.

1.3.2 Outline of this Thesis

In this chapter, we have introduced the principal motivation and scenario of this thesis, we present the new challenges arising from the newly arrived Industry 4.0 paradigm, which shapes a new generation of control systems, capable of including modern technologies and services such as the CPCSs, industrial IoT devices, cloud computing, etc. In order to better understand such an scenario, we review the principal components introduced by this paradigm, i.e., interoperability, security and distribution, and contrast them with the existing infrastructure designs' capabilities and issues. We determine that the goal of this thesis is to provide a new architectural model for the fourth generation of control systems, capable of contemplating all the key characteristics and components of the new paradigm.

Before proceeding with the design, we review the main issues and threats that menace current infrastructure's CSs and CPCSs, in Chapter 2 we review these advanced cyber threats in the form of cyber stealth attacks, identifying, providing a taxonomy and classification of these attacks, as well as describing the main countermeasures and preventive actions that can be taken to mitigate their effects within critical settings.

To address the design of our architectural model, we initiate it by performing a requirements analysis. In Chapter 3, we identify the main requirements and constraints for CCSs. The focus of this chapter is on non-functional requirements, thus we follow the NFR model framework methodology and the goal question metric approach in order to identify the requirements, satisficing techniques and metrics for the secure interoperability and protection solutions for CCSs of the SG.

Chapter 4 is devoted to the analysis and identification of the main security services and characteristics that are needed in order to establish comprehensive security mechanisms in our architectural model design. This chapter focuses on the security for CSs, one of the three main pillar characteristics of the fourth generation of CSs. We study security at different levels, from the basic security services to the advanced and specific security services for the infrastructure. A special consideration is given to the prevention, awareness and reaction mechanisms that can be implemented in our design, and their key characteristics and applicability to the critical scenario.

In Chapter 5, we provide our design approach to tackle the problem of secure interoperability for the CPCSs of a federation of SG providers. In this section we review the key technologies that make part of our design and the chief interoperability and security components needed to implement this new type of CS for the Industry 4.0 paradigm.

In order to better understand the capabilities and characteristics of our architectural model, we devote Chapter 6 to the validation and analysis of the behavior and performance of a prototype version of our design. This prototype version is presented through a testbed which contains some of the main functionalities devised for the architectural model in the previous section. To perform the validation of the proposed system, we design a series of validation cases through simulations and experimentations which aim to determine the performance and behavior of

the prototype model, and provide insight about the capabilities of a large-scale, real-life secure interoperability system based on our original design.

Finally, Chapter 7 summarizes the contributions of this thesis and describes lines of future work and open research problems.

1.4 Publications and funding

The work in this thesis has led to several publications in journals and international conferences. Next, we provide a list of these contributions, organized by the type of publication:

Articles in ISI-JCR Journals

- L. Cazorla, C. Alcaraz and J. Lopez, "Cyber Stealth Attacks in Critical Information Infrastructures", In *IEEE Systems Journal*, issue 99, pp. 1-15, 2016. ISI JCR Impact Factor 2016: 3.882.
- L. Cazorla, C. Alcaraz, and J. Lopez, "A Three-Stage Analysis of IDS for Critical Infrastructures", In Computers & Security, vol. 55, issue November, Elsevier, pp. 235-250, 2015. ISI JCR Impact Factor 2015: 1.64.
- L. Cazorla, C. Alcaraz, and J. Lopez, "Awareness and Reaction Strategies for Critical Infrastructure Protection", In Computers and Electrical Engineering, vol. 47, issue October, Elsevier, pp. 299-317, 2015. ISI JCR Impact Factor 2015: 1.084.

International conference papers

- L. Cazorla, C. Alcaraz, and J. Lopez, "Towards Automatic Critical Infrastructure Protection through Machine Learning", In 8th International Conference on Critical Information Infrastructures Security, vol. 8328, Springer International Publishing, pp. 197-203, 2013.
- C. Alcaraz, L. Cazorla, and G. Fernandez, "Context-Awareness using Anomaly-based Detectors for Smart Grid Domains", In 9th International Conference on Risks and Security of Internet and Systems , vol. 8924, Springer International Publishing, pp. 17-34, 2015.

Book chapter

- C. Alcaraz, L. Cazorla, and J. Lopez, "Cyber-Physical Systems for Wide-Area Situational Awareness", In Cyber-Physical Systems: Foundations, Principles and Applications, Elsevier, In Press.

The work in this thesis has been funded by a FPI fellowship from the Junta de Andalucía through the project FISICCO (P11-TIC-07223). In addition, parts of this work have received support from the projects PISCIS (P10-TIC-06334), and FACIES (HOME/2011/CIPS/AG/4000002115).

Chapter 2

Advanced Threats to the CPCS

Critical Infrastructures, especially their control systems, have increasingly become the target of many and sophisticated cyber attacks over the years. As mentioned in Section 1.2, critical systems are specially vulnerable to cyber attacks, and *Computer Emergency Response Teams* (CERTs) around the globe confirm the intensification of the number of malicious cyber attacks launched in order to cause disruptions to CIs (see Section 1.2.1 for more information). Especially harmful and dangerous for the CIs are the cyber stealth attacks, introduced in Section 1.2.1, which are capable of causing great damage to critical systems or sensitive data leaks without the system administrators notice.

As discussed in the previous chapter (see Section 1.2), there are many types of cyber-attacks targeting CIs with different objectives at mind. They vary in levels of sophistication, ranging from simple physical system's disruption to advanced large-scale cyber attacks such as the Stuxnet worm [48] or the WannaCry ransomware attack [58] (see Section 1.2.1). There is much literature reviewing cyber attacks, and many reports and discussions on large-scale cyber stealth attacks. However, taking into account the new scenario for the CCSs we depict in Chapter 1 where the new industrial paradigm of Industry 4.0 emerges, and brings with it a new generation of CSs including many new concepts and technologies, we identify a different type of cyber stealthy attack that has not been previously analyzed and discussed in the context of critical systems.

These attacks are the cyber stealth attacks which employ methods and practices applied to general-purpose networks, and which in the new generation of CSs, where CPCSs abound, they represent a growing threat. These attacks target networks and devices in the control systems which were previously nonexistent in critical networks, and which now (if unprotected) introduce new vulnerabilities within critical systems. We, therefore, devote this chapter to analyze in depth and understand the characteristics, scope and mitigation processes of the main cyber stealth attacks belonging to this last discussed category. And, since these attacks can be launched against multiple systems of the *Critical Information Infrastructures* (CIIs) e.g., CPCSs, the communications networks, etc. we use the term CI in general for the remainder of the chapter, referring to all the critical systems and devices (CPCSs, industrial sensors, communication networks, etc.) which are objective of these attacks.

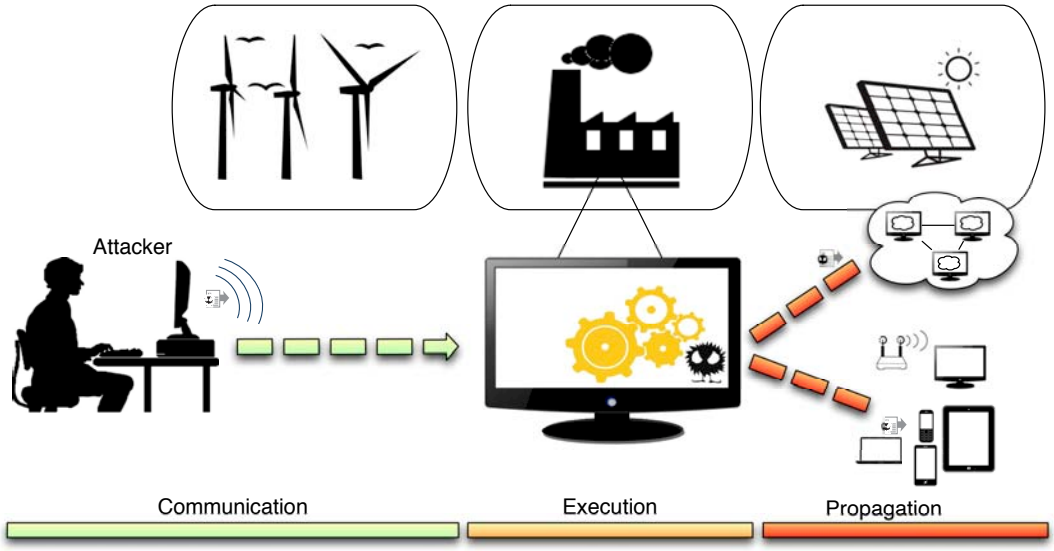


Figure 2.1: Stages of a stealth attack

2.1 Stages of a Stealth Attack

To better understand stealthy attacks, we begin our analysis by determining the process of the attack itself. Stealth attacks, as in any kind of (cyber) attack, are composed of three main stages or phases that have to be fulfilled so as to achieve the adversary's objectives, namely: (i) *stealthiness of the communication*, (ii) *stealthiness of the execution*, (iii) *stealthiness of the propagation*. Figure 2.1 illustrates these stages, where each phase is based on the preceding one. Every single attack is different in nature, and can comprise one or more of the three stages mentioned, always following the established order: first the communication phase, then the execution of the attack and lastly its propagation.

In the specific case of stealthy attacks, they follow these three phases, but the adversary remains undetected while pursuing his objective. However, it is important to note that the success of a stealth attack depends on the intention of the adversary, since his objective might be to achieve only one or two of the stages; e.g., the attacker aims to scan the ports of a system unnoticed, to determine which ones are open, and he does not care about being detected afterwards. In this case, therefore, by succeeding in the first stage of development of the stealth attack, the adversary fulfills his tasks.

Figure 2.1 represents an external cyber attacker, that transmits the attack to the CI, mainly targeting the communication networks and the system's critical nodes. This first phase of the attack is the least intrusive stage of the attack, since sometimes the only aim of the adversary is to achieve this phase undetected. In a second step, the adversary achieves the execution of the attack within the CI itself, this execution could result in vast damage or compromised information, since the adversary remains unnoticed while extracting information or damaging the equipment. The last stage of the attack represented in the figure, is the propagation of the attack to other nodes or to other connected infrastructures. The successful achievement of this step reveals a highly sophisticated attack, launched by skilled adversaries, with good knowledge

of the victim system.

However, the criticality of the attack depends on the intention of the adversary, i.e., it is not the same to subtract information as to cause irreparable damage to the CIs. Additionally, as we have mentioned, each attack achieves one or several of the aforementioned stages according to the objectives of the adversary, i.e., in the case of industrial spies, they may only want to extract information without being discovered, and without causing any harm to the CIs. In Section 2.3, we provide a review of the stealth attacks against CIs, indicating the scope of each attack and the intentions of the adversaries.

2.2 AICAn Taxonomy

In the current literature, there is a wide variety of attack taxonomies and studies on cybersecurity for both conventional and critical systems [3, 62, 63, 64]. However, it is important to stress that the majority of these studies do not consider new ways to address recent security problems. For example, Lipson showed in [65] a chronological study of threats carried out since 1980, and most of these threats are still present in modern information systems. This means that the area of security remains open, where more attention needs to be paid by the scientific community, and more specifically, when ICTs are being adopted in critical contexts.

To complement these studies on stealth attacks in critical scenarios, we extend the taxonomy proposed in [3], based on the security properties *Availability* (A), *Integrity* (I) and *Confidentiality* (C), AIC. To this end, we consider the attack taxonomies given by the ENISA in [66], F. Skopik et al. in [67] and the security framework for ROLL (Routing Over Low Power and Lossy Networks) specified by IETF (Internet Engineering Task Force) in [68].

The motivation behind the extension of the taxonomy based on AIC is the fact that besides being attacked, there are multiple types of anomalies appearing all the time within a critical infrastructure, therefore it is necessary to include certain indicators of anomalies to study the effect they alone have, and when (stealth) attacks are present. In the critical infrastructures field, it is, for example, necessary to discern between infrastructural anomalies and control anomalies:

- *Infrastructural Anomalies* (InfAn), related to physical events (e.g., pressure, flow, radiation) relative to the critical infrastructure itself and its components.
- *Control Anomalies* (CAn), corresponding to any unexpected alteration in the control of critical systems caused by Hardware (HW) and Software (SW) faults, errors or intrusion.
- *Intrusion anomalies* (IntrAn), associated with those malicious actions within the physical infrastructure or its control systems that cause unforeseen incidents.
- Combinations of the above. For example, an IntrAn can trigger a CAn, or vice-versa; or an IntrAn can produce abnormal changes in the readings values causing an InfAn (e.g., a stealth attack).

Given the importance of taking into account the anomalies when detecting intrusion or security gaps, we therefore propose to include a new class within the taxonomy given in [3], denoted

here as AICAn and depicted in Figure 2.2. This new taxonomy comprises the following threat classes:

Most of the stealthy attacks base their strategies on conventional threats against the availability (A), integrity (I) and confidentiality (C) of critical data, its hardware/software resources and user's information (credentials and roles) [3]. However, as mentioned above, adversaries can also take advantage of existing vulnerabilities or anomalies to attack the critical system's AIC. For this reason, we propose a new taxonomy based on AIC plus anomalies, denominated here as AICAn, where, for each category, we identify a subset of threats according to the their nature and type:

- Availability: these threats aim to reduce, as much as possible, the accessibility and disposition of resources and information of the system, infringing upon some of the aforementioned SCADA security requirements. These threats can be carried out through a set of actions related to *Denial of Service/Distributed Denial of Service* (DoS/DDoS), or physical attacks. Depending on the intentions of the attacker (exhaustion of assets, operational disruption or reduction of functionalities), we identify two subcategories within the availability property: *Resource Availability* (RA) and *Information Availability* (IA).
- Integrity: correspond to those vulnerabilities exploited to distort critical sections of a node/object or its messages, such as an overflow or implementation attack. Availability attacks may also have a repercussion on the integrity of a node and its assets, thereby violating one of the essential security requirements of a SCADA system. We consider two sub-types of integrity threats: *Resource Integrity* (RI), and *Information Integrity* (II). Additionally, if an adversary is capable of manipulating security credentials and roles so as to impersonate the users or the administrator of the system identities, a threat to the *User Integrity* (UI) and *Host-User Integrity* (HUI) can arise.
- Confidentiality: concerns the adversary's ability to eavesdrop or deliberately expose sensitive information belonging to configurations or critical data, i.e., information on operational control (commands, alarms or measurements) or information associated with connectivity, routing tables, nodes location, existing vulnerabilities, etc. This allows the adversary to carry out subsequent attacks [69], and thus we have to differentiate between *Resource Confidentiality* (RC) and *Information Confidentiality* (IC) in our analysis.
- Anomalies: an anomaly is defined as something that deviates from the standard or common. If the system presents a specific set of rules/patterns of behavior, an anomaly would therefore be the introduction of new unknown patterns, or the breach of such rules/patterns. As we have stated, it iso possible to identify three anomaly categories: *Infrastructural Anomaly* (InfAn), *Control Anomaly* (CAn), *Intrusion Anomaly* (IntrAn), and any combination of them.

All of these threats, especially those related to availability, integrity, confidentiality and intrusion anomalies, can be the origin of the distortion or corruption of assets, destruction of assets, denial of service, information disclosure and eavesdropping [62]. To form the AICAn taxonomy, however, we have to consider the possibility that unforeseen events (anomalies) can also become potential threats, which may open up new security gaps that can be exploited through stealth attacks; or that these events may stem from these attacks as well.

Stealth attacks, as described above, happen in a scenario where the objective of the adversary is

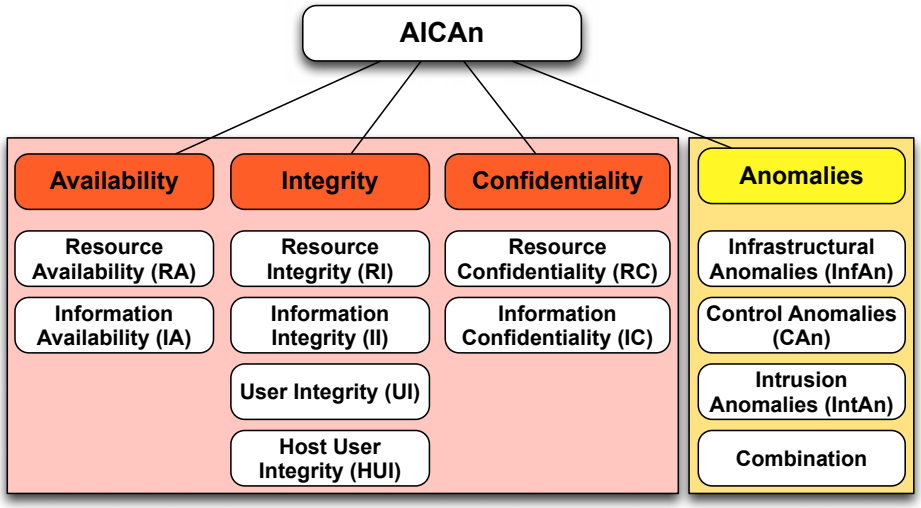


Figure 2.2: AICAn taxonomy

not only to successfully perform the attack, but also to do so with a minimal effort, and in a way that hides his existence and activities to the largest possible extent. It is therefore important to identify the methods or weapons employed by the adversaries, which are closely related to the AICAn taxonomy [60]. Firstly, *impersonation*, which attacks the integrity (I) of the system, and consists in introducing packets with stated originators different from the real originators, which can be performed by spoofing IP addresses or by using communication frequencies that have been assigned to others. This is always supposing that the originator of the impersonation is an honest party.

Secondly, the *lies* weapon threatens the integrity (I) of the system, where the attacker propagates incorrect information, such as incorrect routing tables. Lastly, *overloading*, which threatens the availability (A) of the system, is a technique that has been proposed as a possible technique to mount DoS attacks, where the attacker injects invalid messages (message with violated integrity, replayed message or junk message). Technically, overloading is difficult to implement as a stealth attack, nevertheless, it can be quite effective in controlling operations such as route discovery or routing table update.

2.3 Classification of Cyber Stealth Attacks

Stealth attacks can be categorized according to several parameters. In our review of the literature, we find there are five types of stealth attacks depending on the objective of the adversary: (i) *disconnection and goodput reduction* [60], (ii) *active eavesdropping* [60], (iii) *scanning and probing* [70], (iv) *covert and side channel exploitation* [71, 72] [73], and (v) *code injection* [74, 73].

2.3.1 Disconnection and Goodput Reduction

In this first type of attack, the adversary wishes to disconnect the communications network (a partition of the network or isolate particular nodes) of the CI, or degrade its operation (its *goodput*). Here, the adversary does not need to control the nodes, but only needs to make them inadvertently get involved in the attack by tricking them into modifying their behavior (e.g., modifying their routing tables incorrectly) to cause disruption. This attack implies a threat to the availability and sometimes the integrity of the victim system, constituting a risk to the IA, RA and RI according to the AICAn taxonomy; also, these threats indicate possible anomalies in the infrastructure regarding confidentiality (CAn) and due to the intrusion itself (IntrAn). This attack is especially applicable to CPSs deployed in the field and remote networks, as present in the fourth industrial revolution.

An attacker may disconnect a victim in several ways, e.g, M. Jakobsson et al. [60] provide different variations of the disconnection attack in wireless mobile networks, where the power consumption of the devices is critical to their operation:

- *Disconnection due to the unreachability of the nodes*: the adversary disconnects the victim nodes making the other nodes believe they are unreachable (attack against the IA, RA). This attack has several variations, implementing different degrees of stealthiness by using these methods:
  - The adversary routes considerable amounts of traffic through the victim until it runs out of power. This attack is based on the cost that sending messages has in terms of the battery power consumed.
  - The adversary attacks all the known neighbors of the victim node making their batteries run out of energy. This causes disconnection as well, but it can be overcome by moving into another neighborhood.
  - The adversary routes traffic to the victim node and its neighbors, causing a portion of the messages to be dropped due to insufficient bandwidth. This version of the attack takes into account the response of a router trying to reach a node several times, and then concluding that the node is disconnected.
- *Removal of an entry in the routing table*: here, the adversary disconnects a node removing its entry in the routing tables of the network, making the victim node "disappear" (attack against the IA, RA, RI). It is also possible that the attacker forges the route discovery messages to convince the source node and other legitimate nodes that no route to the victim can be found.
- *Goodput reduction*: the disconnection of one or more nodes usually implies a reduction of the goodput of a network. The adversary can disconnect a large number of nodes, corrupt a large enough number of routing tables to increase the de facto traffic through each node, or degrade the power supplies of a large enough portion of the routers, virtually disabling them. This constitutes attacks against the IA, RA and RI of the AICAn taxonomy.

Stealthy implementation of these procedures allows a low exposure of the adversary during the attack. What we have previously discussed are stealth versions of the common DDoS attack [60]. Regarding stealth DoS, there are several ways of performing this type of attack, for example,

M. Jakobsson et al. provided an overview on how it can be carried out against different types of wireless networks in [60, 75].

2.3.2 Active Eavesdropping

This second type of stealth attack comprises the modification of the routing information of a section of the communications network of the CI, in order to hijack traffic from and to selected victim nodes [60]. Here the attacker can perform traffic analysis and selective filtering of packets without the knowledge of the victim, to actively eavesdrop on him and modify his behavior, e.g., making nodes of the network "disappear" and detouring the network traffic through compromised nodes. This attack usually threatens the confidentiality of the system (IC, RC), thus we usually see the activation of the indicator CAn in the presence of eavesdropping attacks. Sometimes it also introduces risks to the availability or integrity (IA, RI).

The simplest way to achieve this attack is to corrupt the routing tables of nodes on the path between a victim and the sender/receiver. The attacker can remove correct routing table entries and add incorrect ones in order to force rerouting [60]:

- For *incoming traffic*, i.e., packets going into the victim, the attacker forces all incoming traffic to be sent through a node he has previously corrupted. To receive traffic only from certain sources, the attacker can selectively tamper with the routing tables, allowing only those entries that are useful to the attacker to remain correct.
- For *outgoing traffic*, i.e., packets sent from the victim to another node in the network, the attacker modifies the routing tables of the victim and/or the routing tables of the nodes close to the victim forcing traffic to be rerouted through a corrupted node.

To corrupt the routing tables of the network, the adversary can use the very tools of the routing protocols. The attacker can propagate routing tables where the entries are modified; another option is to make use of the route discovery process of the network to include new routes or report route error, in order to tamper with the routing tables. This attack was especially introduced in the context of *Critical Infrastructure Protection* (CIP) in the networked generation of CSs, and it is becoming more and more dangerous in the 4th generation due to the massive presence of remote and distributed networks which use wireless technologies and establish communications through the Internet.

2.3.3 Scanning and Probing

Scanning is a method for discovering exploitable communication channels. It implies a previous reconnaissance of the network or a particular host [70] of the CI. The objective of port scanning is to determine which ports of the system are open, and through them obtain valuable information; e.g., which services are running on the system that are available to the attacker, what services of the *Operating System* (OS) are being used, parameters such as IP and MAC addresses, topological information, etc. The idea is to probe as many listeners as possible, and keep track of the ones that are receptive or useful to your particular need [76].

These types of attacks are the least dangerous in terms of threats to the AICAn of the system, therefore threatening the correct operation of the system, but they present a threat to the confidentiality of the resources (RC) and they can serve as a precursor to more powerful and disruptive attacks, thus they need to be always considered and monitored. C. Yin et al. [77] state that the port-scan is at the beginning of the process of intrusion, and there are varied techniques to scan the system, e.g., stealth scan, fragmentation scan, changes of scan order, slow scan, randomizing inter-probe timing, scan with forged address or distributed scan. G. Lyon states in [76] that several techniques have been developed over time for surveying the protocols and ports on which a target machine is listening.

This threat is present in the networked generation of CSs and would increase its presence in the 4th generation since the CSs tend to be distributed and therefore expose their interfaces to the Internet. While there are many industrial protocols and systems that can be probed, the necessary inclusion of remote operations for control tasks have made them especially vulnerable to scanning and probing attacks coming from the Internet, being the most common attacks the TCP-based ones.

During a normal TCP connection, the source initiates the connection by sending a SYN (synchronize) packet to a port on the destination system. If a service is listening on that port, the service responds with a SYN/ACK (synchronize/ acknowledgment) packet. The client initiating the connection then responds with an ACK packet, and the connection is established. If the destination host is not waiting for a connection on the specified port, it responds with an RST (reset) packet. Most system logs do not log completed connections until the final ACK packet is received from the source [78].

To scan the system, this standard behavior is modified in different ways. Here, we describe some of these variations, in order of degree of stealthiness:

- *TCP connect() scanning*: the most basic form of TCP scanning, where the connect() system call of the OS is used to open a connection to every interesting port on a machine. If the port is listening, the connect() call will succeed; otherwise the port is unreachable. This technique is fast and does not need any super user permissions, however, it is easily detectable and filterable, since the target node will log the connection and error messages when the adversary initiates the connection to the port service and immediately shuts it down.

- *TCP SYN scanning*: sometimes referred to as half-open scanning, since the TCP connection is not fully opened. The attacker sends a SYN packet, as it would happen to open a real connection, and waits for a response. The response can be a SYN/ACK packet if the port is listening, or a RST packet if the port is not listening. When the adversary receives a SYN/ACK packet, he sends a RST packet to tear down the connection. This attack needs super user permissions to build the SYN packets. The advantage of this attack is that systems do not usually log these kinds of attempts at communication; however, it is easily detectable if the firewalls are configured to detect SYN packets targeting restricted ports.

- *TCP FIN scanning*: increasing the level of stealthiness, the FIN (finalize) scanning technique [79] is based on the idea that closed ports respond to FIN packets with RST packets, while open ports ignore them. The FIN scan's stealth packets are unusual because they are

sent to a device without first going through the normal TCP handshaking. Nevertheless, there are some systems that are not vulnerable to this type of scan, because they respond to a FIN packet with an RST packet regardless of the current state of the port.

- *Christmas scan*: this type of scanning technique sends a TCP packet to a remote device with the SYN, FIN, ACK flags set. This is colloquially called a Christmas tree scan because of the alternating bits turned on and off in the flags byte (00101001), like the lights of a Christmas tree. Similar to the FIN scan, a closed port responds to this packet with an RST packet, and an open port ignores it.
- *Null scan*: the adversary creates a TCP packet with all the TCP flags off. This is a type of packet that never occurs in the real world. As in the previous two situations, an open port receiving this kind of packet ignores it, and a closed port responds with an RST packet.

These last three attacks are denominated stealth scan attacks [78], because they do not usually generate a log entry on the scanned host, and they allow an attacker to determine which ports are open on a target node, without being detected by the host OS. Many attacks in the literature use stealth scans and probes as a first stage in reconnaissance to gain insight into the characteristics of the system, to later trigger a more sophisticated and informed attack.

I. Dainotti et al. [80] provide a study on stealth scans carried out by botnets, in a coordinated and distributed infrastructure, targeting critical voice communications infrastructures. This scan attack is called sipscan and probes each target IP address with two packets: (1) an UDP packet sent to the port 5060 carrying a session initiation protocol (SIP) header, and (2) a TCP SYN packet that attempts to open a connection on port 80. This attack is usually the first step in a more sophisticated attack, where the attacker sends malware that infects the nodes of the network to make them act to profit the adversary.

2.3.4 Covert and Side Channel Exploitation

A side channel attack is very powerful in practice [81]. Here the adversary measures side channel information and is able to recover very sensitive information about the functional behavior of a system, without utilizing its dedicated interface [71]. Side channel attacks exploit the external manifestations of the system, like processing time, power consumption and electromagnetic emission to identify the internal computations [72]. This type of attack represents a threat to the confidentiality of the resource (RC) and in the particular case of side channel attacks that induce faults in the system, the anomalies indicators that are activated are InfAn, CAn and the IntrAn.

The aim of side channel attacks is usually to identify a "leakage" or source of secret data (side-channel analysis), where the attacker can use the results of this information to identify weaknesses in the system. The different types of side channel attacks are: timing attacks, power analysis attacks, electromagnetic analysis attacks, fault induction attacks, optical side channel attacks, and traffic analysis [71]:

- *Timing attack*: the adversary analyzes the running time of the system in order to extract knowledge about the type of computations and the parameters used. The main targets of timing attacks are cryptographic systems.

- *Power analysis attack*: here, the adversary measures the power consumption of the system to extract knowledge about it. There are several types of power analysis attacks, mainly targeting cryptosystems, which employ different methodologies and levels of sophistication to obtain the information; e.g., simple power analysis, differential power analysis or correlation power analysis.

- *Electromagnetic analysis attack*: this kind of attack implies the analysis of the electromagnetic variations of a system by the adversary. There are several types of electromagnetic attack which target very different kinds of systems; however, this kind of attack is most often designed for constrained cryptosystems.

- *Fault induction attacks*: the induction of faults in the system can result in erroneous operations that can shed some potentially valuable information about its operation.

- *Optical side channel attacks*: here the adversary is capable of retrieving information via the light emission from the monitors and LEDs (light-emitting diode) of a system. There are different kinds of displays and LEDs, and the information that can be extracted from them is varied.

- *Traffic analysis attacks*: this kind of attack provides the adversary with information about the topology of the network, through the analysis of the traffic flows.

A variation of a side channel attack is the use of *covert channels* [73], where there is a hidden connection between the transmitter and the receiver, thus there is a chance to extract or send valuable information through the channel without the system noticing. There are two types of covert channels: (i) communicating extra information to a host, and (ii) hiding the fact that the communication to a host exists [74]. Covert channels usually take advantage of places where random data is naturally transmitted, thus the encrypted information can be transmitted replacing this data. This technique is sometimes referred to as piggybacking [82], where the messages are hidden within the regular messages of the network. There are many varied ways of implementing covert channels, and the targets are multiple. However the commonality behind this type of attack is its dangerousness and its potential to induce multiple threats within the victim systems, targeting most AICAn variables. According to N. Tomar et al. in [73] the following vulnerabilities that can favor covert channels:

- *Virus and other malware*: software such as viruses and Trojan horses can be introduced inside the victim's system, to perform activities such as capturing packets and injecting scripts into the victim's programs.

- *Important resources*: resources such as system files, disks, RAM, etc. are valuable to attackers, and vulnerable due to their criticality in the normal operation of the system.

- *Data sensitivity*: within the system coexist data with different degrees of sensitivity. The most sensitive data is the most interesting information to attackers, and thus the target of covert channel exploitation.

- *Vulnerable protocols*: several protocols implemented by CIs that are not properly secured, or they do not implement security mechanisms such as authentication (e.g., Modbus [13]). To protect the systems against covert channels attacks, it is important to strengthen their security.

- *Design robustness*: covert channels take advantage of principally two vulnerabilities of the system: design oversight, and weaknesses due to the system's design. Design oversight-derived vulnerabilities are unintentional and unforeseen, however weaknesses inherent in the system's characteristics are strong obstacles to the security of the system and provides a way of access for covert channels.
- *Packet headers*: as seen in [74], covert channels can be embedded in TCP and IP header fields, with very different objectives and functionalities.
- *Super user permissions*: an attacker can take advantage of an unintentional, careless or default assignment of super user permissions to processes, to create a covert channel.
- *Handshake trials*: communication protocols usually have handshake procedures to start the transmission of information. Some attackers use handshake trials to transfer information in an unnoticed way.
- *Public resources*: resources that are shared in the network, such as printers or hard drive disks, are vulnerable to attacks if they are not protected by security mechanisms.
- *Authentication*: as we have previously seen, some protocols and systems lack adequate authentication mechanisms, such as Modbus, DNP3 or ICCP [13]. This adds multiple vulnerabilities to the unprotected systems, among them, the use of covert channels by an attacker.

Most of these attacks introduce, as we have described, a wide range of AICAn threats, e.g., the attacks that exploit flaws or use malware are capable of threatening the availability (IA, RA) and the integrity of the system (II, RI), as well as compromising the integrity of the user and the host (UI, HUI); the confidentiality of the system can be also compromised (IC, RC), activating the CAn and sometimes the IntrAn indicators.

2.3.5 Code Injection

A *code injection-based attack* consists in introducing or "injecting" a tainted or illegitimate code within a computer program, in order to alter its outputs or change its course of execution [83], and cause different effects, e.g., compromise sensitive data, execute malware, etc. These attacks target varied types of control SW in the CIs, and pose a threat to multiple variables of the AICAn taxonomy, allowing the adversary to interfere with the AIC of the system, and insert CAn and IntrAn anomalies.

Depending on the targeted system's characteristics and the degree of stealthiness intended in the attack, it can be performed using two main channels: *system vulnerabilities*, and *malware infection* (i.e., infecting the system with malware, virus or Trojan horses). Injections exploiting design vulnerabilities appear when system designers and developers make incorrect assumptions about the use of the system's services, e.g., (i) the input characters of a field will always be the regular and required ones (e.g., no colons, numbers or quotation marks are expected); (ii) the input of a field will never exceed a pre-determined size; (iii) the numeric values introduced as inputs in a system will always stay between the upper and lower bounds expected; (iv) the client supplied values cannot be modified by the adversary (e.g., cookies poisoning attack [84]);

(v) it is safe to take pointers or array indexes from the requested input; (vi) the input will never provide false information or fake values (e.g., the size of a file); etc. [85].

On the other hand, malware can also pose successful and potentially harmful threats when implementing injection attacks (e.g., Stuxnet [48], Duqu [51], etc.). There are multiple types of code injections, and several ways of classifying them. We have decided to categorize them according to the target they are designed to inject, thus these attacks can be roughly summarized into the following four categories:

- *Database injection*: are the injections performed by the adversary to corrupt the databases of the system, or retrieve valuable information from it, without having the proper credentials to access the system. Database injections compromise the AIC of the system (IA, RA, II, RI, IC and RC) and activates the CAn and IntrAn indicators of anomalies. The most well-known attacks in this category are the SQL-injection attacks [75].
- *Command injection*: also known as shell injection attacks [86], can occur when the the system allows software to execute a command line. Therefore the attacker can make the system execute commands or functions to carry out unwanted tasks. This type of attack allows the attacker to threaten the AIC of the system (IA, RA, II, RI, IC and RC) and of the user (UI, HUI), in addition to introducing the CAn and IntrAn anomalies.
- *Website injection*: is the set of attacks that take advantage of flaws existing within websites, browsers or web applications that allow the adversary to introduce code and execute unwanted actions in an otherwise trusted environment (threatens AICAn like the previous attack). The most well-known attack within this category is *cross-site scripting* (XSS), which occurs when the adversary exploits a flaw detected on a web server to inject some code in the server, for his own use [87, 88]. Related attacks are the *Cross-Site Request Forgery* (CSRF) [89], where the adversary forces the victim to execute unwanted actions on a web application in which he is currently authenticated; or the *Server-Side Includes* (SSI) Injection [90], where the attacker introduces scripts in HTML pages or executes arbitrary codes remotely.
- *OS injection*: comprise those attacks that target the stack, heap, pointers or internal variables determining the behavior of the system. Code injection at this level can make the OS execute unwanted routines and procedures, inserted in the OS's running processes through the modification of the system variables to point to external code introduced by the attacker [91]. They threaten the AICAn as does the previous attack.

In a critical context, these attacks can target different parts of the infrastructure, namely the *corporate networks*, the *SCADA center* and the *remote substations*. The first are based on local area networks connected to the SCADA to gain access to critical data streams on SCADA servers, and are vulnerable to injections designed for conventional networks. The SCADA center is in charge of constantly monitoring the infrastructures through distributed substations. The remote substations are control sub-networks based on field devices (sensors, actuators) and communication interfaces (PLCs, gateways, etc.) in charge of sending sensorial measurements to the SCADA center. The SCADA center and the remote substations are vulnerable to injections specifically designed to target industrial devices and protocols.

Code injection attacks usually tend to implement some degree of stealthiness, since the adversary usually aims to retrieve valuable information from the system, or to force a desired (malicious)

behavior without the end user being alerted. The actual level of stealthiness depends on the objective of the attacker, and also on the way the injection is tailored to the targeted system. According to Figure 2.1, it is possible to evaluate the degree of stealthiness of a given attack (in the communication, execution and transmission phases) and assess the potential threats and risks it poses.

2.3.6 Assessment of Stealthiness

We can differentiate two main kinds of behaviors in cyber stealth attacks: the *reconnaissance based attacks* and the *attacks with disruptive or tampering objectives*. These two main groups differ in the threats they pose to the correct operation of the CIs in terms of the AICAn taxonomy. Attacks with reconnaissance objectives, e.g., scanning and probing, or side channel attacks, are characterized by an adversary who tries to gather as much information as possible from the victim system, without being discovered in the communication phase (see Figure 2.1). In the case of this type of adversary behavior, the properties of the AICAn that are affected are usually related to the confidentiality, specifically the confidentiality of the resources (RC). In some of the cases, the attack is capable of retrieving certain information from the system, thus the IC property of the AICAn is compromised.

Some of the reconnaissance attacks might cause disruptions in the victim system, when the attacker intentionally induces faults to obtain information; in this case, the availability of the system can be affected, i.e., the IA and RA properties of the AICAn taxonomy; and the indicators of anomalies InfAn, CAn and IntrAn could be activated. Let us take a simple example, the *TCP connect() scanning* attack, where the attacker probes the ports of the system in search of useful open ports. This attack does not cause any disruption to the victim system, however the adversary is able to extract information about it, using just the communication phase of the attack to his own benefit. The information discovered in the reconnaissance attacks can be used by the adversary to launch more sophisticated attacks in a later step, using the knowledge acquired in the reconnaissance. The level of stealthiness achieved by this first group of attacks is determined by the stealthiness of its communication phase; i.e., whenever the adversary implements the attack in such a way that the victim system's warning mechanisms are not triggered by the reconnaissance actions, the attack can be categorized as stealthy.

Our second category of attacks, those with disruptive or tampering objectives, are characterized by an adversary who tries to achieve all the phases of the attack, i.e., communication, execution and sometimes propagation, stealthily. These attacks are much more complex, requiring highly skilled and informed attackers, capable of communicating with the system and executing the attack and if desired, propagating it to infect other components or target systems. Due to the possibilities they offer to the attacker, they are very dangerous to the victim system in terms of AICAn, because they can potentially disrupt all the AIC properties of the system and trigger all the different types of anomalies. The most representative attacks in this category are covert channel attacks and code injections.

To evaluate the level of stealthiness of a given attack it is necessary to evaluate each phase of the attack in order to determine if all of them are stealthy, and if the defensive mechanisms (e.g., *Intrusion Detection System* (IDS)) of the victim are not alerted by the attacker's actions. As an example, we consider a code injection attack where the adversary's objective is to stealthily

achieve the three phases of the attack. Firstly, in the communication phase of the attack, the adversary can exploit vulnerabilities detected in the target system, or can make use of malware (virus, Trojan horses, etc.).

Both methods open the door to performing code injection stealthily if the attacker specifically designs the attack to avoid triggering the defense mechanisms of the victim system. Therefore the injection attack is considered stealthy at the communication stage if the vulnerability exploitation or the malware communication is stealthy. An example of this first phase is the exploitation of the industrial communication protocols used in the CIs, e.g., the Modbus/TCP protocol, commonly used in SCADA and DCS networks for process control, which do not provide authentication of the source of a request. This provides an adversary with a chance to attempt to gather information on the system being controlled and about the PLC [92].

In the second phase, the execution of the injected code (see Figure 2.1), the level of stealthiness achieved in this stage depends on the implementation of the attack and on the defense mechanisms available in the targeted system. If the attack is designed to perform its tasks in a way that avoids triggering any alarm, and the security mechanisms implemented are not finely tuned to detect this kind of attack, the injection can be considered stealthy in its execution stage. To illustrate this assessment in the context of CIP, we analyze the PLCs Modicon M340 from Schneider Electric, which has a disclosed vulnerability to CSRF attacks [93]. These devices incorporate a web server interface that processes requests from clients about the underlying infrastructure. However, the web server does not implement security mechanisms to verify their authenticity, thus an adversary could trick a client into sending an unintentional request to the web server, which would be considered authentic [94].

The injected commands could be sent to the PLC through a specially crafted HTTP request, for example, sending the victim a request embedded in an image <`img src="http://plc-web-server.com/?query_string"/`>, where the `query_string` would request the server to perform some malicious action that would be considered legitimate. The adversary could exploit this vulnerability to remotely reset or alter the PLC's configuration. Lastly, we can assess the stealthiness of the propagation stage of a code injection. Through the exploitation of vulnerabilities, the attack could in some cases be successfully disseminated. However, through the use of malware it is possible to stealthily communicate the injection attack to other victims, as we have seen in the Stuxnet worm [48], or its variation Duqu [51], that were specifically designed to attack a particular PLC manufactured by Siemens, and infect numerous network devices without leaving evidence of the attack.

Therefore, we conclude that cyber attacks with disruptive or tampering objectives can be stealthily carried out through the three phases illustrated in Figure 2.1. We also stress that these types of attacks should be classified as very dangerous to ICSs, since the adversary could launch a potentially harmful attack that executes malicious actions and propagates its effects without being noticed, threatening not only a CI, but spreading the threat to other dependent or interconnected targets.

Table 2.1 summarizes the contents that have been reviewed in this section, providing a tentative analysis of the threats that stealth attacks pose to CIs in relation to the AICAn taxonomy. In this table, divided into targeted areas and threat categories, it is possible to observe that attacks are closely related to one another, since attackers, irrespective of their modus operandi, generally base their goals on the execution of a set of combined threats to the AIC of the

Table 2.1: Cyber stealth attacks and their relation with AICAn

Category	Stealth Attacks	Stealthiness			IA	RA	II	RI	UI	HUI	IC	RC	InfAn	CAn	IntrAn
Disconnection and Goodput Reduction	Unreachability of the nodes	○			✔	✔								✔	
	Removal of entries in routing tables	○	❍		✔	✔		✔							✔
	Goodput reduction	○	❍		L	L		✔						✔	
Active Eavesdropping	Traffic hijacking	○	❍		U			U			✔	U		✔	
	Modification of the routing tables	○	❍		U			U			✔	U		✔	
Scanning and Probing	TCP connect() scanning	○										✔			
	TCP SYN scanning	○										✔			
	TCP FIN scanning	○										✔			
	Christmas scan	○										✔			
	Null scan	○										✔			
Side-Channel Exploitation	Timing attack	○										✔			
	Power analysis attack	○										✔			
	Electromagnetic analysis attack	○										✔			
	Fault induction attack	○										✔	✔	L	✔
	Optical side channel attack	○										✔			
	Traffic analysis attack	○										✔			
Covert Channel Exploitation	Due to virus and malware	○	❍	●	U	U	U		L	L	✔	✔		L	U
	Targeting important resources	○	❍		L	U	L	U			✔	✔		L	L
	Targeting sensitive data	○	❍		L		L				✔	✔		U	
	Using vulnerable protocols	○	❍	●		U	L		✔	✔	✔	✔		✔	
	Using design flaws	○	❍	●	U	U	✔	✔	L	L	✔	✔		✔	
	Using packet headers	○	❍								✔	✔			
	Using super user permissions	○	❍	●	L	U	U	✔			✔	✔		✔	
	Using handshake trials	○	❍								✔	✔			
	Using public resources	○	❍		✔	✔	✔	L			✔	✔		✔	
	Using lack of authentication	○	❍		U	U	L	L	✔	✔	✔	✔		✔	L
Code Injection	Database injection	○	❍		✔	✔	✔	✔			✔	✔		✔	✔
	Command injection	○	❍	●	✔	✔	✔	✔	✔	✔	✔	✔		✔	✔
	Website injection	○	❍	●	✔	✔	✔	✔	✔	✔	✔	✔		✔	✔
	OS injection	○	❍	●	✔	✔	✔	✔	✔	✔	✔	✔		✔	✔

○: stealthy communication of the attack. ❍ : stealthy execution of the attack. ● : stealthy propagation of the attack.
✔: threat violates security property. L: threat is likely to violate security property. U: threat is unlikely to violate security property.

system, as discussed previously. The AICAn analysis is based on the discussion, by a group of experts, of the impact on AICAn by different implementations of each stealthy attack listed. It is important to note that the assignment of likelihood in this table is determined by the different implementations of each of the selected stealth attacks, and may vary if other examples are taken into account. However, we believe this study shows an interesting overview on the impact of stealth attacks on CIs from the point of view of AICAn. From Table 2.1 we can conclude that most of the stealth cyber attacks focus on altering the integrity of the information of the system, possibly inducing threats to the availability of resources and information, and consequently causing control anomalies.

Additionally, some of the more sophisticated attacks expand their scope to also exploit the system's vulnerabilities in order to alter the integrity and confidentiality of the resources and information, and introduce the possibility of impersonation (UI and HUI compromising), producing CAn and IntrAn anomalies. From this table, we conclude that most of these attacks focus on the exploitation of the vulnerabilities associated with control and also those vulnerabilities intentionally produced by intruders. We also note that threats classified as covert channel exploitation and code injection can become potentially harmful threats to CIs, since they can compromise or degrade a wider range of security properties necessary for the good operation of critical systems, endangering the availability, integrity and confidentiality of these systems.

2.4 Countermeasures and Prevention Mechanisms Against Stealth Attacks

Given the restrictive nature of stealth attacks where the adversary wants to carry out his actions unnoticed, they must be very precise and tailored to the target system. Therefore, the defense mechanisms and the countermeasures applied must always take into account the environment of the system that is being protected. In this section we discuss measures that counteract stealth attacks equivalent to those discussed in Section 2.3 in general-purpose networks, which are applicable to critical settings with the adequate adaptations to fit the constrained environment of CIs, e.g., protocol reinforcements, introduction of additional equipment within the network, physical measures, etc. An extensive review of the literature provides two main lines of action for the protection of CIs: *avoidance mechanisms* (passive protection) and *detection and recovery mechanisms* (active protection). We devote this section to providing some ideas about how to protect the systems or minimize the effects of these stealthy attacks.

Avoidance mechanisms are put into place to prevent threats and reduce risks, while detection and recovery provide early detection and warning against attacks, and help restore the system to its original working state, palliating the effect of anomalies or attacks. These protection mechanisms are applied to counteract the weapons used to perform the attacks. The most threatening of the weapons under consideration, i.e., the one with the least visibility and cost, is the use of impersonation. The use of lies is a weapon with an inferior degree of stealthiness than impersonation, however it is also threatening if the attacker uses it to propagate incorrect information to corrupt the targeted system. Overloading has the lowest degree of stealthiness, nevertheless a skilled adversary could make use of it to collapse a subsystem of a CI without drawing the attention of the system administrators.

In general terms, it is possible to employ different methods to counteract these weapons; the main avoidance mechanisms that can be used are: *cryptography*, *standardization* and *reputation mechanisms*. Apart from these, when addressing each different attack, it is possible to apply specific countermeasures, either active or passive protection. The use of *cryptographic authentication methods* improves resistance against stealth attacks, since cryptographic authentication is harder to forge than IP addresses, etc. It is also important to note that in the field of CIP, the most-used protocols (e.g., Modbus [92]) still lack authentication mechanisms, something that is advantageous to the attacker [75]. Additionally, the naturally scarce resources such as bandwidth, storage, computation capabilities or power, provide the adversaries with targets to easily bring down the operation of the network.

Nevertheless, the implementation of cryptography in constrained systems is challenging, thus it is necessary to consider the use of lightweight cryptographic primitives for authentication, e.g., symmetric cryptography or elliptic curve cryptography [81]. However, to only rely on authentication is insufficient to thwart stealth attacks, since the corruption of legitimate nodes' behaviors perverts the correct authentication processes [75]. Thus it is necessary to strengthen the authentication process by *applying recommended and standard procedures*.

There are varied authentication and access control techniques, such as *Discretionary Access Control* (DAC), where the access control is assigned according to pre-defined criteria, *Mandatory Access Control* (MAC), where the access control is evaluated according to a set of policies, *Attribute-Based Access Control* (ABAC), where the access is granted according to the evaluation of a set of attributes combined according to system policies, *Role-Based Access Control* (RBAC), which uses roles and privileges to grant access to users, location-based authentication, where the physical location of the user is combined with identity tokens to authorize access, and several others [95, 96].

From all these methods, we find interesting the separation of the user identity from its privileges within the system, since it provides a more versatile and dynamic management. These characteristics can be find in the RBAC model. What is more, the IEC/TS 62351-8 standard [28] is especially focused on the security of remote control substations (for CIP), and underlines the need to implement access control mechanisms using RBAC, which is directly recommended for this environment. This standard, additionally recommends the usage of the *Principle of the Least Privilege*, or principle of minimal privilege, which states that the sole entities able to gain access to logical devices and modify their objects will be those (virtual and physical) entities with the suitable permissions to operate in the field.

To address this, authentication must be based on the assignation of subjects-to-roles and roles-to-rights, restricting the accesses to particular objects developed in substations (e.g., IEC-61850 objects [97]). This difficulty is increased due to the knowledge uncertainty about the honesty of the different hosts. However, several of the aforementioned problems can be palliated (even solved) when deploying *reputation mechanisms* to protect the networks, so that even if the nodes are compromised by adversaries, the reliability of the system can still be assured. The use of reputation has various advantages, such as the use of collaborative methods, which provides robustness to the design of the network and eliminates the connectivity dependencies between nodes [75].

Cryptography and reputation measures are especially beneficial for *goodput reduction attacks*. Although these two main countermeasures try to minimize and palliate all kinds of stealth attacks

against the networks, they are particularly useful in the case of the disconnection attacks or the *active eavesdropping*, where once detected, the traffic going through the corrupted nodes can be averted or reduced [60].

Scanning and probing attacks are one of the most critical types of stealth attacks, since they open the door to other more sophisticated and more informed attacks. Some countermeasures against these attacks are provided by V. Marinova-Boncheva in the paper [70]. The author proposes the use of stealth probes to detect any attacker that prolongs his procedures for a long period of time, for example, checking for system vulnerabilities and open ports for a period of two months. To this end, the stealth probes collect information from the system, checking for methodical attacks that last an extended period of time, they sample a wide area and discover correlating attacks. Basically this technique implies the use of mixed signature-based and anomaly-based IDSs.

Another way to confront stealth scanning and probing is proposed by C. Yin et al. in [77], where they suggest the use of honeypots to detect the attacks and alert the system's administrators. A honeypot is "*an information system resource whose value lies in unauthorized or illicit use of that resource*"; it reacts like a normal machine, based on the type of OS it simulates, while it is recording and transferring packets to scan detection mechanisms to learn the tactics and tools used by the attackers and alert the administrators of illegal accesses to the network it is protecting.

The countermeasures for *side channel attacks* are highly tailored to the type of exploitation and the actual implementation of the attack. G. Joy Persial et al. provide certain guidelines to counteract side channel attacks in their work in [71]:

- *Timing attacks*: this kind of attack can be prevented by hiding time variations or using blinding techniques [98]. A simple form of hiding variations is to make the computations in constant time. Another possibility is to always add certain computations to the execution of the algorithms to mask the timings. Other variations include hiding the internal state of the systems, so that the attacker is no longer able to simulate internal computations.

- *Power analysis attack*: the power consumption is reduced using masking and elimination techniques. Masking "*randomizes the signal values at the internal circuit nodes while still producing the correct cipher text*" [71]. It can be done at software level, adding random masks to data subsequently encrypted, or at hardware level where the system adds random mask bits to balance the degree of randomness of the resulting message.

- *Electromagnetic analysis attack*: this kind of attack can be prevented by covering the system with a protective casing that hides or attenuates the electromagnetic radiations. This case also prevents the attacker from accessing the individual physical components of the system.

- *Fault induction attack*: can be prevented by checking the computations [98] or verifying the signature of the sent messages to identify the failures. There are IDSs specifically designed to identify core failures and hijacks and the correct operation of the systems [99] [100].

- *Optical side channel attack*: to prevent the adversary from retrieving information from display monitors and leds, once the device is ready to be deployed. These lighting signals

Table 2.2: Cyber stealth attacks summary table

Category	Stealth Attack	Stealthiness			Weapons	Countermeasures
Disconnection and Goodput Reduction	Unreachability of the nodes	○			Impersonation Lies Overloading	Cryptography Reputation mechanisms
	Removal of entries in routing tables	○	❍			
	Goodput reduction	○	❍			
Active Eavesdropping	Traffic hijacking	○	❍			Cryptography Reputation mechanisms
	Modification of the routing tables	○	❍			
Scanning and Probing	TCP connect() scanning	○				Stealth probes Honeypots
	TCP SYN scanning	○				
	TCP FIN scanning	○				
	Christmas scan	○				
	Null scan	○				
Side Channel Exploitation	Timing attack	○				Hiding timing variations Blinding techniques Masking techniques Protective casing IDS and validation of computations Disabling and masking of light signals Encryption and masking of the channel
	Power analysis attack	○				
	Electromagnetic analysis attack	○				
	Fault induction attack	○				
	Optical side channel attack	○				
	Traffic analysis attack	○				
Covert Channel Exploitation	Due to virus and malware	○	❍	●		Anti-malware Resource monitoring Special security mechanisms applied to sensitive data Secure protocols Design assessment and correction Use of IDS Proper policies to assign permissions Handshake restrictions Restricted access to public resources Authentication mechanisms
	Targeting important resources	○	❍			
	Targeting sensitive data	○	❍			
	Using vulnerable protocols	○	❍	●		
	Using design flaws	○	❍	●		
	Using packet headers	○	❍			
	Using super user permissions	○	❍	●		
	Using handshake trials	○	❍			
	Using public resources	○	❍			
	Using lack of authentication	○	❍			
Code Injection	Based on design flaws	○	❍	●		Monitoring tools Prevention and validation mechanisms
	Based on malware propagation	○	❍	●		

○: stealthy communication of the attack. ❍ : stealthy execution of the attack. ● : stealthy propagation of the attack.

used for debugging should be disabled, or masked.

- *Traffic analysis attack*: counteracting this type of attack is very difficult [71], since it is necessary to encrypt the messages transmitted and mask the channel, to prevent the adversary from analyzing the traffic. In their work in [101], J. Deng et al. provide different countermeasures to prevent this attack, based on modifications of the routing schemes used by the nodes of the network.

Existing countermeasures for *covert channels* are varied, and comprise the use of commercial solutions such as antivirus and anti-malware SW, and restricting and strengthening the implementation of the network's protocols and policies. Examples are [73]:

- *Anti-malware*: as we have previously seen, software such as viruses, worms and Trojan horses can be introduced inside the victim's system, to capture packets and inject scripts into the victim's programs. Updated anti-virus and anti-malware SW can generally detect these behaviors.
- *Resource monitoring*: resources such as system files, disks, RAM, sockets, etc. are valuable to attackers, and thus adversaries frequently target them. Monitoring these resources with HIDS can provide insight into the system's status and help detect the presence of covert channels.
- *Data sensitivity*: information can be classified according to its level of sensitivity, thus special security mechanisms can be put into place to differing degrees to protect the data according to its sensitivity.
- *Secure protocols*: to protect the systems against covert channels attacks, it is important to strengthen the security of the network, thus implementing secure protocols, e.g., HTTPS instead of HTTP, helps prevent such attacks and protects the transmission of sensitive information.
- *Design robustness*: covert channels take advantage of design oversight vulnerabilities,and weaknesses due to the system's design. In the first case, these unintentional failures can be corrected once discovered, removing the covert channel. In the second case, they cannot be removed until the system is re-designed to eliminate the vulnerabilities. However, the use of good practices, such as secure programming or process desegmentation, can make the system more resilient against covert channels.
- *Network Intrusion Detection Systems (NIDS)*: such as Snort [102], monitor packet header fields such as ACK, SYN, to detect patterns that can indicate (unmask) the presence of covert channels.
- *Super user permissions*: super user permissions may be needed to execute software, but it is necessary to carefully evaluate the processes granted with these permissions, to avoid harmful routines that are able to damage the system.
- *Handshake restrictions*: handshake trials between systems can be a way used by a malicious actor to fool traffic monitoring systems, thus a limitation on these trials should be put into place.
- *Public resources*: the access to public resources such as printers or shared disks should be restricted and limited to the known users of the network, and reinforced with au-

thentication methods for preventing covert channels. For example, the use of RBAC, *Attribute-Based Access Control* (ABAC), Kerberos or simple Public Key Infrastructure (PKI) could help.

- *Authentication*: methods like passwords, captchas [103] or biometric mechanisms can help protect the system against covert channels, as well as RBAC/ABAC, Kerberos or PKI. Additionally, the IEC/TS 62351-8 [28] standard for security in substations recommends the use of authentication mechanisms, and more particularly RBAC to reduce complexities in the entire SCADA network.

Prevention methods for covert channels are not restricted to just these points. Since the covert channels implemented for a system are highly tailored to its individual characteristics, each of the targeted environments will provide new challenges to the adversary. Thus, new behaviors will appear, and consequently, the targeted systems can be protected in different ways according to each specific situation.

Regarding the countermeasures that can be put into place to prevent and fight *code injection attacks*, in addition to the general measures that can be used (i.e., cryptography, standardization and reputation), it is possible to take two different approaches: *prevention and validation mechanisms* and *monitoring tools* (e.g., IDSs, antivirus, anti-malware SW).

To prevent code injection, it is important to secure the input and output handling, by introducing validation mechanisms, selective inclusion and exclusion procedures, standardized input and text formatting and encoding, parametric variables, dissociation and modularization of the procedures from the kernel of the system, good handling of super user credentials, isolation of some critical procedures, hash validation of executable images, and similar mechanisms [83, 104].

In order to detect the most sophisticated and stealthy injection attacks, it is important to deploy intelligent and finely tuned IDSs, capable of adapting to new dynamics and learning new attacks [105], beyond just relying on attack signatures and known events. These automatic and adaptive capabilities provide the detection systems with tools to detect and prevent highly targeted and complex stealth attacks [106, 107, 108].

Most of the countermeasures and preventive mechanisms discussed in this section can be categorized as avoidance mechanisms (passive protection), however, as cyber attacks against control systems are becoming increasingly aggressive and sophisticated, it is necessary to put into place active protection mechanisms, to address the continuous threats to the CIs [42, 15]. Thus, as discussed and as a complementary measure to avoidance mechanisms, detection and recovery mechanisms are the techniques put in place for early detection, prevention of and counteraction to risks in order to restore the system to its original working state, and palliate the effect of the attacks or anomalies happening within the system.

Given this definition, we classify the active protection mechanisms into two main categories: the methods that require the intervention of an operator, and the automatic methods. Within the first class, we find the early warning systems, the IDS, and all the situational awareness [3] mechanisms deployed to detect and alert the human operators of any attack or anomaly happening within the system under surveillance. To the contrary, the automatic methods are those tools deployed to provide an automatic response to the problems that arise, with little to no supervision from the human operators.

Currently there is little literature on the automatic or semi automatic response mechanisms, since their application to CIs is complex and potentially dangerous, due to the criticality of the environment. However, it is absolutely essential to start to deploy such techniques within CIs, since faster counteractions would help prevent the effect of attacks or anomalies from cascading to other interconnected and interdependent CIs [15]. Solutions that can provide these automatic functionalities are the *Intrusion Prevention Systems* (IPSs), SW that "*has all the capabilities of an intrusion detection system and can also attempt to stop possible incidents*" [109].

The IPS is often integrated as an extension of the IDS, but it usually receives less attention than IDS research due to the intrinsic complexity of developing the mechanisms that offer an automated and correct response against certain events. However, the increased complexity and speed of cyber-attacks in recent years shows the acute necessity for complex intelligent dynamic response mechanisms [105]. These systems can perform a wide variety of actions, from operations on files and re-routing, to automatic revocation of privileges for certain profiles of the infrastructure. Thus, using this module, it is not necessary to alert the system's human operator/administrator to launch countermeasure actions, the system itself could select and execute them in a semi-supervised or unsupervised way.

In Table 2.2, we summarize the analysis of the stealth attacks from the point of view of countermeasures and protection, also reviewing the level of stealthiness of the attacks corresponding to Figure 2.1. This evaluation takes into account their associated AICAn risks (see Table 2.1), always considering the worst scenario possible; i.e., the maximum level of stealthiness that an adversary can achieve using these techniques and approaches. Moreover, we provide an overview of the most suitable countermeasures applicable to prevent or react against the stealth attacks, outlined in the last column of this table. This set of tentative measures is a selection of procedures that come from the context of general-purpose networks (trying to palliate or avoid stealth attacks in these non-critical settings) and which can be applied to CIs with a few adaptations to fit the specific needs of critical environments (industrial protocols, additional equipment, etc.).

2.5 Summary

According to M. Jakobsson et al. [60], stealth attacks are better (i.e., more profitable) than regular attacks, which require a higher amount of energy and leave the attacker more exposed to detection. In the previous sections, we have identified five different types (main categories) of stealth attacks, namely: (i) disconnection and goodput reduction, (ii) active eavesdropping, (iii) scanning and probing, (iv) covert and side-channel exploitation, and (v) code injection attacks. We have described their objectives and scope and using the AICAn taxonomy, we have determined their potential threats to CIs.

This study therefore shows the danger inherent in attacks where the adversary tries to go unnoticed, since the system can be threatened for long periods of time without being protected, the actions against the infrastructure are varied and range from simple probing of the system to extraction of sensitive information, or disruption to the correct operation of the CIs affected. Additionally, the adversaries are able to propagate their threats to other nodes or interdependent CIs, thus creating cascading effects through the interconnected infrastructures.

Besides the vulnerabilities introduced in the scenario associated to the interest of the infrastructure to adversaries (sensitive data, potential of social disruption, etc.), the high complexity of the environment and their interconnected nature increase exposure to potential attackers and unintentional errors. According to NIST [110], a high number of interconnections present increased opportunities for DoS attacks, introduction of malicious code or compromised HW. Moreover, when dealing with a vast amount of nodes in the network, as happens in CIs, the number of entry points and paths exploitable by and adversary increases.

Nevertheless, there are several methods that help prevent and counteract the attacks studied. The main actions we find that currently are indicated to help in the case of stealth attacks are the preventive mechanisms, such as reputation or cryptography. We find therefore that it is essential to incorporate protection tools for control elements, governance, validation and testing of SW and HW components, to prevent any perturbation to the system's security properties. Moreover, protection of communication channels (using for example cryptography, virtual private networks, bump-in-the-wire, etc.) is also needed, since most of the cyber threats rely on attacks against the confidentiality (information or configurations of resources), in order to learn about the environment, conditions and elements of the victim system.

However, in the event of truly sophisticated stealth attacks, it is necessary to include a layer of protection that provides reactive recovery mechanisms capable of launching automatic reactions against an attack that is underway, to restore the normal operation of the system under attack, as soon as possible. Within this category we find the IDS and IPS modules, capable of advanced detection mechanisms, and in some cases, of launching some prevention actions and alerts to the security profiles responsible for the nodes under attack. Currently, there is little research on automatic and semi-automatic reaction systems, due to the inherent complexity of the modules, which is vastly increased in the case of CIs, where any disturbance of their operation is of critical relevance.

Chapter 3

Analysis of Requirements of Secure Interoperability in CPCS

Practically all our critical infrastructures are today under the supervision and are dependent on other additional systems, the control systems, whose underlying infrastructures in turn, rely heavily on the new information and communication technologies for control. Indeed, ICTs have now become essential elements in our society because they offer significant benefits to improve efficiency, cost reduction and enhance quality of life. In their evolution towards the fourth generation of CSs (see Section 1.1.1), they are incorporating technologies such as mobile computing, distributed systems, smart devices, IoT devices, wireless communication or cloud-computing. These new concepts and technologies are becoming the major driving forces behind the management of diverse information, allowing a quicker operation of the great majority of today's competitors' infrastructures and their services.

In fact, most of these physical facilities are highly interconnected to other national (and international) infrastructures through communication systems, and are managed through ICTs [35]. This new way of monitoring makes the present control systems critical in themselves (as discussed in Chapter 1), where the notion of criticality is intertwined with the nature of the system and its sensitivity to adverse events caused by unforeseen faults or intentional threats. This also means that CIs and their minimum services (e.g., water, energy or transport) are also dependent on the effectiveness of the ICTs integrated inside CSs in charge of collecting, distributing and processing the correct functional performance of resources and the provision of services. A particularization of CSs are the CPCSs (see the definition in Section 1.1.2), also considered critical systems in themselves. In this section we will refer to CSs in general, and refer to CPCSs when a particularization is needed.

Examples of CSs are DCSs or SCADA systems, both belonging to the category of ICSs [1] (see Section 1.1.1). SCADA systems, in particular given their centralized nature are commonly sensitive to a number of threatening factors: (deliberate/unintentional) faults and existing vulnerabilities related to access control, communication or control. All of them may imply not only the degradation of the minimum monitoring services but also the neglect of other essential services for society, which could even result in the well-known *cascading effect* between infrastructures [31].

The proof of this is found in annual reports published by different governments through specific organizations such as ENISA [41] and the ICS-CERT [43, 54], respectively (see Section 1.2.1). Both reflect the current situation and the severity of potential threats, where the number of specific incidents apparently continues to grow. This requires an immense effort to design protection measures without infringing the five basic control principles defined in [21]: *real-time operational performance, dependability, survivability, sustainability* and *safety critical*.

These five requirements are basic because they encompass a further set of important conditions, such as availability, integrity, access to component, component lifetime, change management and reliability, among others [111]. Many of these requirements are also included in the guidelines published by the NIST in [112] for ICSs and for smart grids in [20], and even in the guidelines published by the *North American Electric Reliability Corporation* (NERC) through its NERC-CIP cybersecurity series (002-009) [113] (for further information on the guidelines characteristics see [21]).

On the other hand, we have so far identified three chief vulnerabilities in CSs: (i) weaknesses in the configurations of the CIs (network configurations, security policies, defense mechanisms, etc.); (ii) architectural design of the system; and (iii) errors in the development, which are generally related to software and hardware elements. Many of these vulnerabilities come from modernizing the TCP/IP-based technologies, increasing, on the one hand, the complexities of the CS and, on the other hand, adding new vulnerabilities to the control.

Indeed, existing SCADA architectures, their devices and their protocols, have to adjust to the new technological changes brought by the Industry 4.0 and the new generation of CSs to offer network architectures which are distributed, secure and autonomous for data management in real time, interoperability between systems and protocols (e.g., Modbus or DNP3 with ISA100.11a, WirelessHART or ZigBee PRO) and scalability for the cooperation and collaboration with the new control technologies. A clear example of this new change, as discussed in Chapter 1, is found in the new electrical distribution generation, i.e, in the smart grid generation.

In the SG, CSs serve as the central edge of supervision and data acquisition from thousands to millions of smart devices with direct connection to diverse networks: backhaul, *Wide Area*, *Field Area*, *Neighborhood Area*, and *Local Area Networks*. In this context, backhaul and the Internet are the chief sources that connect the different sub-domains with the rest of the networks, including *Advanced Metering Infrastructures* that characterize the bidirectional interfaces between the real world, and the acquisition and control world.

Through these interfaces it is possible to manage and interact with smart meters and utility business systems, substituting the traditional one-way advanced meters. This technological evolution also shows how CSs are becoming more complex at the different levels (functionally, architecturally), and hence also require special care when adapting new technologies. In other words, finding a perfect connection between systems and guaranteeing secure monitoring at all times requires, for each adapted technology, a set of minimum requirements which should not interfere with the basic control principles.

Given these considerations, this chapter aims to identify and analyze the main requirements that cover the security of the control systems, particularly the CPCSs, and the ones that make possible this interconnection and interaction of subsystems in order to provide correct interoperability and interconnection among them, creating a secure interoperability infrastructure among control

systems, in which different critical systems interact and cooperate in a secure and efficient way to help the CIs perform is tasks and provide their critical services to the society.

Through this analysis, we aim to provide a guideline an analysis that contain the main requirements these secure interoperability infrastructure has to comply with in order to ensure its correct operation for the interconnection of critical control systems, as well as different methods to satisfy these requirements (in the form of sets of techniques) and a way to measure and assess the fulfillment of the identified requirements and the performance of the suggested secure interoperability architecture. Throughout this chapter we review the different requirements and characteristics that constrain our scenario, to later analyze the possible techniques to help the architecture comply with the requirements, and after that, we provide an example set of metrics to help evaluate the correct operation of such system. We also make a particularization on IDS systems deployed as components of the infrastructure for its protection.

3.1 The Special Requirements and Constraints of Critical Control Systems

As stated, CSs are systems that manage other critical systems (also known as "systems of systems"). Specifically, ICSs are deployed to aid in the operation of industrial infrastructures, the services of which are also essential to social and economic welfare. This feature means that CSs can also be considered as critical control systems (CCSs, as discussed in Section 1.1.1), where the correct operation of their control services against unforeseen and/or dynamic changes is of paramount importance. In this section, we review the main characteristics and requirements of CSs, as well as the constraints any system has to consider when dealing with these critical systems. This analysis is the first step in our study, which sets the basis to better understand the features needed to build a secure interoperability infrastructure for control systems as we devised in Chapter 1.

3.1.1 CCS Requirements

CCS are complex systems, and this complexity is due to several factors: (i) ICSs are composed of multiple networks with thousands of nodes in them; (ii) the heterogeneity of the network is high, integrating multiple types of nodes and technologies, i.e., legacy equipment (e.g., old RTUs and industrial sensors) using protocols such as Modbus/TCP [13] running alongside modern devices (e.g., cloud servers, wireless sensors, mobile devices, etc.) which use technologies such as Bluetooth [114] or ISA100.11a [115]; (iii) ICS components and sub-systems have many dependencies between them, and CCSs also have interdependencies with other CIs [31].

These factors make CCSs challenging systems to manage, moreover, they also make the ICSs targets vulnerable to attacks [41]. In recent years, cyber attacks targeting CIs have increased exponentially, as shown in the cases of Stuxnet [48], Duqu [51], the Nitro attacks [50], the WannaCry ransomware attack [58] and others [39, 47] (see Section 1.2.1).

Due to their complexity, CCSs need to be analyzed in order to better understand their requirements, to provide them with adequate protection against the multiple threats they face because

of their characteristics. In [21], the authors compile the requirements that a CCS must comply with to achieve the right levels of security and performance. We describe them here, as these requirements form the basis of our study:

- *Real-time performance*: CCSs have hard real-time constraints regarding communications, execution processes and system upgrading, as none of them should cause delays in the system. The communications' response time is heavily constrained, sometimes tightened to a maximum of one millisecond [1]. Additionally, naturally occurring faults in CIs, or malicious activities can introduce delays in the system, something that needs to be palliated and reduced as soon as possible.
- *Dependability*: is "*the ability of a system to properly offer its services on time, avoiding frequent and severe internal faults*" [21], thus a control system must provide its service despite fault occurring. Dependability comprises five attributes that absolutely have to be observed: a*vailability, reliability, maintainability, safety* and *security* [21].
- *Sustainability*: as defined in [21], sustainability is "*the ability of a system to meet the needs of the present without compromising its ability to meet future needs*", i.e., the system must continue to function like the day it was deployed despite any later updates, upgrades or modification of its components (hardware and software).
- *Survivability*: is "*the capability of a system to fulfill its mission and thus to face malicious, deliberate or accidental faults in a timely manner*" [21]. Survivability is composed of three main elements: *unsusceptibility*, *resilience* and *recoverability* (defined below).
- *Safety critical*: this is safety related to critical environments; its implementation makes it possible to prevent unplanned effects that the failure of a critical system could inflict on society. It also relates to the protection against faults cascading from critical infrastructure to another, the so-called "*cascading effect*" [21, 31].

These requirements are common to all critical environments. These scenarios are highly complex and heterogeneous and, as we have mentioned, they combine multiple technologies and are subject to many restrictions and constraints. This makes it difficult to use the same state-of-the-art in ICTs as in regular, non-critical information systems. In Figure 3.1, we represent the aforementioned CCS requirements, graphically. In this figure, not only do we consider the hierarchies of the requirements and how they are composed, we also consider that all these requirements have the same level of importance or criticality.

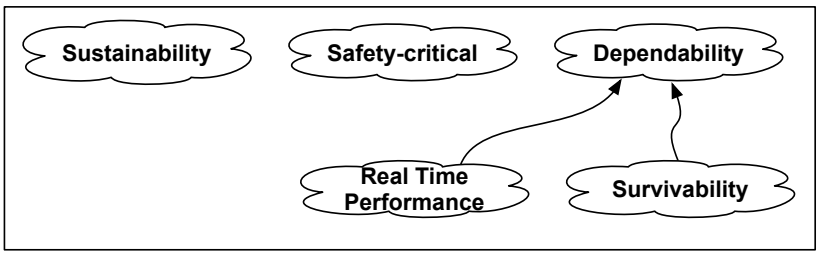


Figure 3.1: CCS requirements

The reason for the hierarchization of these requirements is purely semantic, according to the

definition and concepts of safety, security and survivability engineering [116]. We understand that if one of the requirements is not met, the CCS fails to perform its tasks and is incapable of providing the required service. This failure to meet the CCS requirements causes failures in the affected CI which could also possibly cause a cascading effect, affecting those related and interdependent infrastructures that rely on the service affected.

3.1.2 CCS Constraints

SCADA systems are one of the main types of CCSs used for large, geographically-dispersed distribution operations, such as electrical power grids, petroleum and gas pipelines, water or waste-water systems [117]. The special characteristics of CCSs, particularly SCADA, as seen in [21], result in special constraints that control HW and SW elements deployed within them have to comply with. Fleury et al. [111] identify these constraints and classify them in 5 different categories:

- *Performance and availability*: critical data must be available at all times, without delays or jitter in data delivery, and it must be reliable and have high integrity [111]. In addition to these constraints, any (cyber) security mechanism implemented must be fail-safe so that the failures of such mechanisms do not result in the failure of the CI.
- *Deployment and management*: CCSs need to be highly stable with respect to failures before they can be deployed because they govern physical systems with equipment deployed to last decades. What is more, their operation cannot suffer down-time for system maintenance and upgrades in the way that is common to traditional ICT systems [111]. Thus, practices such as SW patching are not trivial in CSs, since it is not practical (and sometimes impossible) to take down CCSs to apply security patches [118].
- There are strong *computation, space and storage* constraints in CCSs because they were adopted in the 1960s and although their architecture has evolved, legacy equipment, SW and protocols are still working in today's networks and need to be taken into account [1].
- A common constraint found in control systems is the strict *application timing* requirements, some of which require a message delivery time of no more than 2 ms [99].
- The *extra costs* associated with security computations, i.e., the ones performed solely to achieve a device's security goals, do not scale well in critical environments, due to the diversity of its many embedded systems [99].

These above-mentioned constraints define and restrict the capabilities of the CCSs, and hence the security mechanisms applied to those systems, especially in the case of the secure interoperability architecture for the CSs of the SG discussed previously. Apart from the use of different communication protocols and data types, tailored security solutions are particularly critical wherever the resources are constrained and the security measures applied compete with the control software to perform their tasks.

It is vital that the secure interoperability solutions implemented for CSs in the SG observe at any time these identified requirements and constraints for the CCSs. Particularly, they must respect the *responsiveness* aspect of the system, (e.g., a command from the controller to actuator should be executed in real-time by the latter), and the *timeliness* of any related data being delivered

within its designated time period, also meaning *freshness* of data, i.e., the data is only valid for its assigned time period [117].

A particularization is the application of specific security and protection solutions for the CCSs, in a way that is compatible with their requirements. A. Nicholson et al. [118] defend that Anti-Virus, Firewalls, IDSs and *Intrusion Prevention and Response Systems* (IPRSs) solutions found in general information technology networks are equally effective when employed to protect control networks, but they must be tailored to the types of data used in this environment. We analyze the requirements and characteristics of the secure interoperability architecture in the sections that follow, and later we particularize this study with the analysis in depth of the application of detection and protection modules for CCSs.

3.2 NFR Model Framework

Since our work aims to study and assess the operation of a secure interoperability architecture, rather than reflect specific behaviors, we create a model of the system based on non-functional requirements. A *Non-Functional Requirement* (NFR) is defined as "*a SW requirement that describes not what the SW will do, but how the SW will do it*" [119]. Examples of these NFRs are SW performance requirements, SW external interface requirements, and SW quality attributes.

NFRs are difficult to test, therefore, they are usually evaluated subjectively [119]. It is our aim, however, to develop our study further, translating these high-level requirements into quantitative information with respect to the IDS solution and its suitability within the critical environment where it will be deployed. To this end, we base our analysis on different frameworks and guidelines to be able to model our scenario.

In this section, we gather the main NFRs in order to establish the basic requirements for the secure interoperability of CCSs as a first stage of modeling our system. To this end, we use the NFR Framework, as described in [119]. This framework is used to model qualitative process-oriented goals, dividing them into *non-functional requirements*, *satisficing techniques* and *claims*. Since our proposed model implies high-level non-functional requirements, instead of goals, we will model softgoals, where a *softgoal* is defined as a goal with "*no clear-cut definition and or criteria as to whether it is satisfied or not, since NFRs are subjective, relative, and interdependent*" [119].

3.3 NFR Requirements for Interoperability

Following the NFR framework methodology, we address the identification of requirements that allow us to design a CCSs which key components are the ones determined by the fourth generation of CSs, as described in Section 1.1.1, i.e., decentralization, security and interoperability. Additionally, our new design model need to incorporate the key technologies brought into scene by the new Industry 4.0 paradigm, e.g., CPSs, IoT devices, cloud computing, etc. Since we need a framework of study, we center our analysis on the CSs of the smart grid, as discussed in Chapter 1. Therefore, the analysis of these requirements aim to provide a frame of reference

containing the main requirements that CSs, and particularly CPCSs, have to comply with in order to be integrated within a secure interoperability platform where varied types of systems and actors of the SG interact and operate collaboratively.

We picture a very heterogeneous scenario, where different energy providers, control centers, network protocols, legacy and modern equipment have to coexist and cooperate in order to provide their services without interruption and in a manner that observes and adheres to the previously identified requirements and constraints for CCSs (see Section 3.1). We analyze the softgoals for the CSs based on the requirements for the CCS previously identified in Section 3.1.1 (see Figure 3.1).

We provide a complete map of the requirements in Figure 3.2. However, for the sake of clarity, we have divided the identified softgoals into two subgroups: the *softgoals for the interoperability architecture and critical systems*, and the *softgoals for the interoperability*. The first set of requirements (see Figure 3.3) reflects the softgoals we have identified which guide the design and development of an interoperability architecture for critical systems, and the diagram takes into account *CIP softgoals* and *architecture softgoals*.

The CIP softgoals are those that reflects the need of any CS deployed within the critical systems of implementing certain characteristics in order to comply with the CCSs' constraints and requirements (see Section 3.1). The architecture softgoals describe those requirements that need to be addressed in order to build a secure interoperability architecture. To better differentiate the two types of requirements, Figure 3.3 illustrates them with a color code, in yellow we find the CIP softgoals, and in orange the architecture softgoals.

The second set of softgoals, the softgoals for interoperability (see Figure 3.4) is composed of a mixture of requirements, addressing different needs of the secure interoperability architecture for the CPCSs of the SG. Among these requirements we find: *interoperability softgoals*, *virtualization softgoals*, *cloud softgoals* and *industrial IoT softgoals*. Also, in the figure we find *CIP softgoals* and *architecture softgoals*, which are represented there and repeated from Figure 3.3 in order to better understand the relations between the two different sets of requirements and how they interact and depend from each other.

In order to better understand these identified softgoals and their repercussions in the design of a secure interoperability architecture, we devote this section to describe and explain those requirements that are representative of this scenario. We focus only on the requirements that are especially interesting for the scenario of secure interoperability, leaving out the analysis of those NFRs of general nature, that can be easily consulted in the literature, e.g., for security engineering requirements definitions and clarifications, see [116].

When we observe the diagram in Figure 3.2 (also present in the subset Figures 3.3 and 3.4), we can see that the five softgoals depicted at the top of the diagram are the ones described in Section 3.1.1, corresponding to the basic requirements for the correct operation of the CCSs (in color code yellow). If we focus on the first subset of requirements (corresponding to Figure 3.3), we observe that below them, we find the rest of the CIP softgoals and the architecture softgoals we have identified for the secure interoperability architecture, and that extend the study in [21]. The definitions for the most representative requirements in Figure 3.3 are presented next, where we first describe the ones in the CIP softgoals set (code color yellow), i.e., *Safety*, *Accountability*, *(Un)Susceptibility*, *Awareness* and *Context Awareness*.

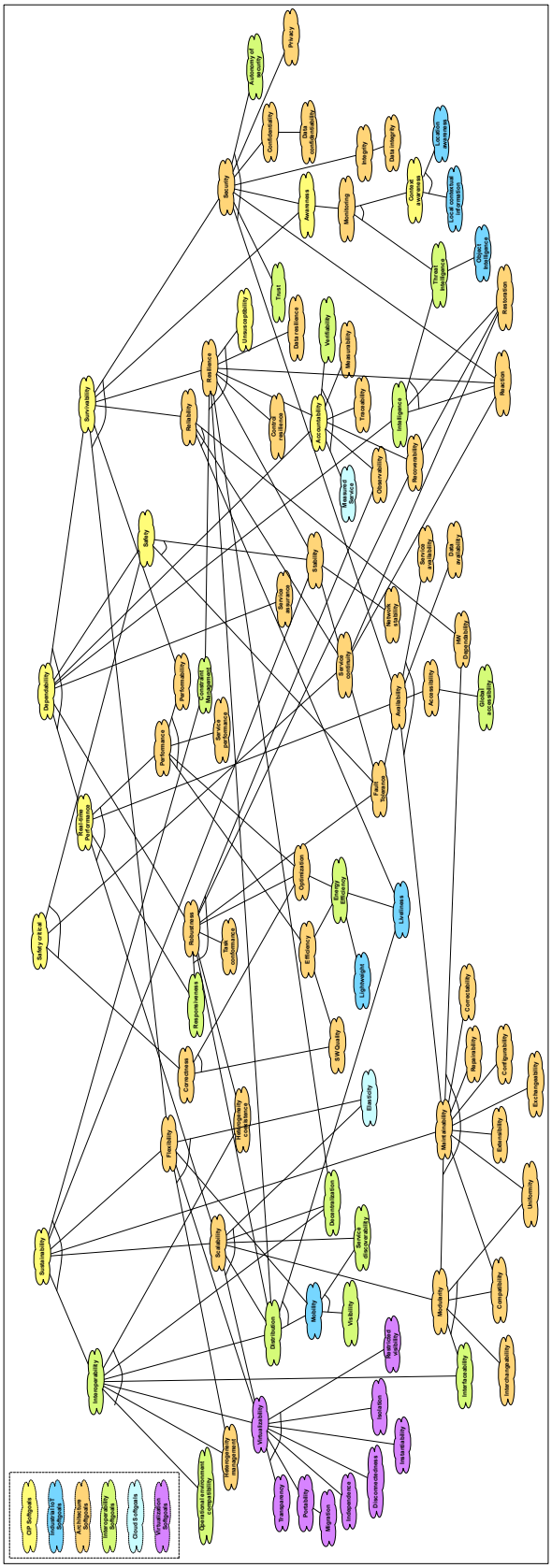


Figure 3.2: NFR requirements for interoperability, a global view

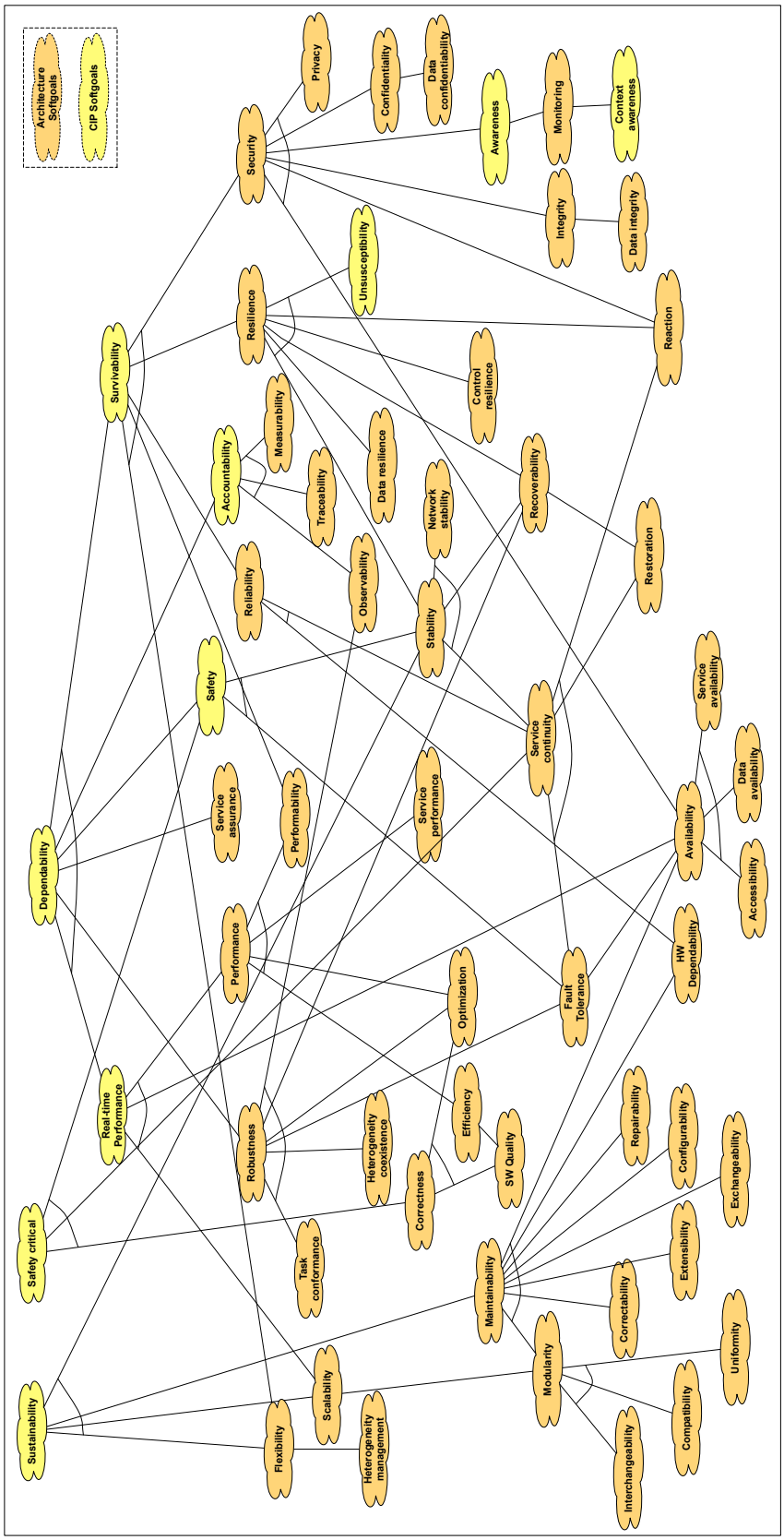


Figure 3.3: Softgoals for the interoperability architecture and critical systems

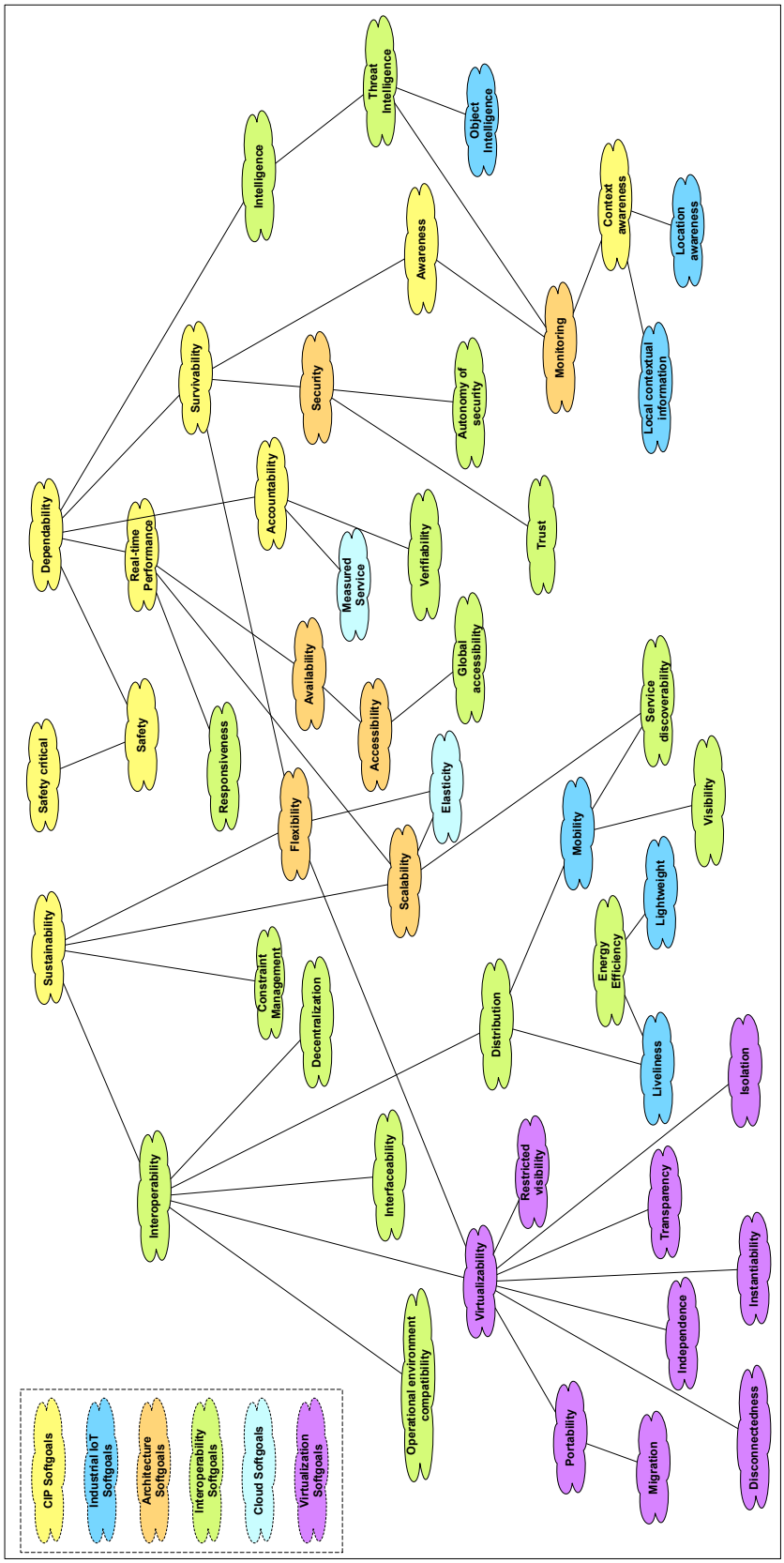


Figure 3.4: Softgoals for interoperability

We define *Safety* from the point of view of reliability engineering as the "*degree to which accidental harm is prevented, reduced and properly reacted to*" [116]. A safety system has the mechanisms to prevent, reduce and react to accidental harm that could affect society in some way. A *safety critical* system is a particularization of a safety system, where the environment or the system itself is critical to social and economic welfare. Safety requirements in CIs must be always be complied with, otherwise malfunctions in the infrastructures operation could affect society and even endanger human life [21]. *Accountability* is defined as the fact or condition of being accountable [120], related to the responsibility to justify actions or decisions. In computer science, this same term applies to the need to preserve records and traces of any actions and decisions taken within the system for later analysis, consult or audit. This requirement is in line with the main security services provided by an interoperability architecture, therefore, it is vital that all CPCSs within this architecture have to comply with this softgoal.

(Un)Susceptibility is the state of being likely or liable to be influenced or harmed by a particular thing [120]. Critical systems, specifically CCSs should not be susceptible to any fault or attack. This is highly complex to achieve, since CSs are such complex and heterogeneous systems, and what is more, in an interoperability scenario, they inherently have multiple vulnerabilities given the multiple systems interconnected and interdependent that participate in the interoperability infrastructure. In order to achieve the highest levels of unsusceptibility to threats, critical systems usually implement multiple security mechanisms for detection, prevention and protection against threatening and malicious events occurring within the system.

Awareness is defined as the "*knowledge or perception of a situation or fact*" [120]. In the field of computer security, awareness refers to these measures taken in order to reinforce the security of the system by analyzing and monitoring its states and behavior. *Context Awareness* is an extension of the previous softgoal, which refers to the ability of a computer system to analyze and monitor its environment and the behavior of their surroundings, i.e., the context, in order to rapidly detect problems that arise, and avoid any malfunctions derived from these events. This requirement has a variation, *Location Awareness* which is applicable to mobile devices, since they have to be aware of their context and also of their location, in order to implement awareness mechanisms. We find that all kinds of awareness are vital for the correct operation of the CPCSs in an interoperability environment. All these requirements contribute to the concept of *Situation Awareness* (see Chapter 1) referring to the perception of environmental elements to understand how they impact system's dynamics and performance.

Additional to these requirements, Figure 3.3 represents a second set of softgoals, the ones related to our scenario's architecture, which are represented with a color code orange. To better understand them, we extract them and provide definitions for the most representative ones in the definitions Table 3.1.

We continue our study by describing the main softgoals in the set of softgoals for interoperability (see Figure 3.4). We therefore devote this part of the analysis to describe the most representative softgoals belonging to the categories distinguished in the figure: interoperability (code color green), virtualization (code color purple), industrial IoT (code color dark blue), and cloud (code color light blue). It is interesting to note that interoperability itself can be considered a goal to achieve in our system, being defined as "*the ability of two or more systems or components to exchange information and to use the information that has been exchanged*" [123]. Thus we consider interoperable, a heterogeneous set of systems that are capable of exchanging useful

Table 3.1: Architecture softgoals definitions table

Softgoal	Definition
Heterogeneity Management	the ability of a system to manage diversity. In computer sciences, a system able to manage heterogeneity is capable of interoperate with diversity in HW, SW, network protocols, etc. without its performance being impacted by the presence of this degree of diversity. This ability of the system to correctly operate in an heterogeneous environment is referred as *heterogeneity coexistence*.
Extensibility	an extensible system "*is able to support new control components such as new technologies, protocols, HW and SW components, and security services*" [21]. Scalability (he ability of a system, network, or process to handle a growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth [121]) and extensibility tend to be mixed up and confused, however they represent two very important challenges for complex heterogeneous systems like CCSs. Extensibility is a maintenance and updating characteristic which is especially desirable in a CI, where the arrival of SW applications and new technologies to CIs such as the IoT or WSNs, greatly increases the need to incorporate new interoperable devices and protocols.
Maintainability, serviceability	the capability of the system to be maintained over time in order to continuously improve it and keep it free from defects and errors. Maintainability is influenced by three factors [116]: (1) *correctability*, the ease with which minor defects can be corrected between major changes, while the system is still in use; (2) *extensibility*, defined above; and (3) *repairability*, the ability of a damaged or failed system to be restored to acceptable operating conditions, within a specified period of time (*repair time*) [120]. Other softgoals directly linked with maintainability are: *exchangeability*, *HW dependability*, and *modularity*, which are described above.
Recoverability	the ability to recover quickly from a system failure or disaster, to the point and (if it is possible) to the state of the system at which the failure occurred [116]. It is closely related to survivability, but it alludes to the capabilities of the system or deployment.
Reliability	"*the degree to which a work product operates without failure under given conditions during a given time period*" [116]. It relates to the costs produced by hazards turning into incidents, and the level of loss of revenue for the company or the customer. It differs from safety, as safety deals with dangerous hazards which could lead to severe accidents with an impact on society.
Performability	the performance of a system, viewed from the point of view of dependability, i.e., the performance of a system in the presence of faults over a specified period of time [122]. An interoperable architecture for CCSs should have high performability in order to ensure the critical operations are performed in a timely and secure manner, avoiding faults and incidents within the system.
Robustness	"*the degree to which an executable work product continues to function properly under abnormal conditions or circumstances*" [116]. *Fault tolerance* is one of its main sub-quality factors. Robustness, from a usability point of view has four related criteria: *recoverability* (defined above), *observability* (consistency and inferability of the internal states of a system from the external outputs), and *task conformance* (support for the tasks established by design) and *responsiveness*, which is described below.
Accessibility	the degree to which the user interface enables users to perform their specified tasks [116]. In an SG interoperability environment, accessibility also refers to non-human users, i.e., the accessibility of services or network nodes by the CPCSs in the network enables the control systems to perform their tasks. This softgoal is related to the *availability* of the system, a vital requirement for critical infrastructures.
Fault Tolerance	the system that is fault tolerant is capable of continuing its operation despite the occurrence of failures; a desirable condition which guarantees the resilience or self-healing of the underlying CCS.
Reaction	"the degree to which the system responds (e.g., recovers) after an accident or attack" [116]. Reaction is related to *recoverability*, and *restoration* (returning a system to a former safe condition after the occurrence of a failure). The reaction approach in a critical environment must ensure that any essential services that may have been lost or degraded are targeted before the recovery of any non-essential services [116].
Resilience	the capability of a system to maintain a proper service in the face of faults, as well as being able to return to normal operation as soon as possible. In terms of survivability, resilient is the antonym of vulnerable. In the context of CPCSs, we refer to the resilience of the information (*data resilience*), and the resilience of the control system (*control resilience*).
Monitoring	the process of providing regular surveillance or systematic review of a system [120]. This softgoal conveys the need of a critical system of continuous analysis and vigilance in search for unwanted and possibly threatening events. Monitoring is directly linked to the *awareness* requirements for CPCSs.
Service Assurance	the application in the system of policies and processes directed to ensure that the services offered meet predefined dependability and quality levels. Also, in combination with *reliability* properties and related to *safety critical* requirements, the softgoal *service continuity* indicates the need for the system to provide a continuous, stable operation.

information with each other. In our work we search for the configuration and characteristics needed in a secure and interoperable infrastructure for CCSs, therefore this can be considered one of the pillar softgoals in our analysis. Interoperability of HW, SW, protocols is of vital importance to build a CS belonging to the new generation of CSs (see Section 1.1.1).

When addressing the analysis of the sets of interoperability NFRs, we are firstly interested in the *Responsiveness* of the system, which is the ability of a functional system to perform an assigned task within the required time interval [124]. For our purpose, we will consider that a system is responsive when it is capable of performing its functions in the required time intervals, with no (or at least no significant) delays. Also, the system must have a good degree of *Operational Environment Compatibility*, defined as "*the degree to which a system functions correctly under specified conditions of the physical environment(s) in which it is intended to operate*" [116]. Any component added to the system must be operationally compatible with the underlying system in order to be deployed within a CCS, where machinery and environmental radiation and noise might interfere with the operation of unprepared systems.

We consider *Decentralization*, the transfer of important functions or services to diverse nodes in a network in order to distribute responsibilities and workload, and *Distribution*, a distributed system is a model where the components belonging to a network communicate and coordinate their actions in order to achieve a common goal, key softgoals in our analysis for a secure and interoperable infrastructure. These properties build high availability [125], resilience and robustness into a system. Interestingly, by achieving these softgoals we achieve a state in which the components of the system work concurrently to achieve a goal, however the faults occurring within one system do not affect the others (they are independent).

Regarding security, se aim to achieve *Autonomy of Security*, which conveys the need that the security system implemented in a critical interoperability environment is independent from the control or influence of any other system in the infrastructure. This allow the security mechanisms to function correctly regardless of any faults occurring in the control system, thus protecting its integrity, confidentiality and availability. Additionally, we consider vital to implement *Intelligence* within the system, referring to the presence of solutions capable of adapting to new circumstances and acting more efficiently in the face of new system dynamics. This requirement is linked to the *threat intelligence* softgoal, which refers to the intelligence in the face of threats, i.e., acting efficiently and adaptively against any anomalous occurrence.

An interesting requirement in our scenario is *Virtualizability*, defined as the ability of a system of being virtualized, or provided in a virtual manner. Virtualization in computer science refers to the creation of a virtual version of something, e.g., computer software, computer hardware, storing devices, or computer network resources [120]. Virtualized network functions, the inclusion of cloud computing services, etc., are new resources and services that can be included in the CPCS interoperability scenario with the help of virtualization. Virtualizability is related to the *Instantiability* softgoal, which is defined as the ability to create an object (an instance) of a specific class [120]. Virtual services can be instantiated on demand to respond to the requests of each client.

Finally, when analyzing mobile systems within a control infrastructure, it is important that they provide *Local Contextual Information* to the control system. This softgoal reflects the need of the interoperability system, especially when dealing with mobile and wireless devices, to provide accurate information about the context of the system in a periodic or frequent manner. This will

allow the system to be aware of the health of the surrounding infrastructure and avoid or timely react to any fault or threat to the system. This softgoal is directly linked to the *awareness* and *monitoring* requirements.

Cloud NFRs, as well as some of the softgoals belonging to the different sets of NFRs described in Figure 3.2 have not been described in this section, as we discussed earlier, this is due to the need to focus our study on the most representative NFRs for the scenario of secure interoperability. Definitions and specifications of these omitted softgoals can be found in the general literature for software, security and reliability engineering, e.g., see [116], [126], [127].

3.4 NFR Satisficing Techniques for Interoperability

In Section 3.3 we identified the requirements that any interoperability platform should comply with, taking into account both the requirements in critical systems and the ones that illustrate the behavior of the communication architecture, the interoperability system, the industrial IoT and the virtualized components within it. We represented them through the NFR Framework softgoals, and now we need to go a step further in order to find specific ways to analyze whether these softgoals can be satisfied, and to find determined techniques or tools to achieve the interoperability.

Since we are working with NFRs, we have to address their characteristics to carry out our study. The problem is that these goals or properties lack a clear definition, as they are usually based on abstract terms which is not very useful from a measurement perspective [128]. It is therefore difficult to assess whether or not the NFRs have been satisfied, because there is no clear-cut criteria for this evaluation. Our approach to tackle this problem is inspired by the *Goal Question Metric* (GQM) approach [129]. The GQM approach is a goal-oriented methodology for the identification of measurements in SW engineering. It is built upon the idea of decomposing the problem into several goals, which are further refined by questions and metrics for answering them. We follow this idea to continue our study on the interoperability softgoals for CIs, in order to bind these abstract characteristics to specific practices.

Instead of refining our analysis in terms of questions at the operational level of the GQM, and in line with the NFR Framework, we reflect on the operations taken for reaching the identified softgoals in terms of satisficing techniques. Thus, in this section, we aim to identify those techniques that can be implemented by the system, capable of satisficing the established NFRs. Table 3.2 presents the simplified matching of the satisficing techniques found for the interoperability softgoals, directly linked to the requirements of the critical systems.

We provide this simplified matching between the NFRs for CIP and the satisficing techniques for the architecture (See Figure 3.5) because we consider the requirements for CIP to be the top requirements for the interoperability of a platform for critical systems, and all the NFRs are below them.

Figure 3.5 (and hence Table 3.2) presents the different types of satisficing techniques identified for the interoperability of CPCSs. In order to best comprehend the image, we have extracted these different types of satisficing techniques into separate figures following the code of color present in the general figure. Those techniques colored in yellow allow the protection of the

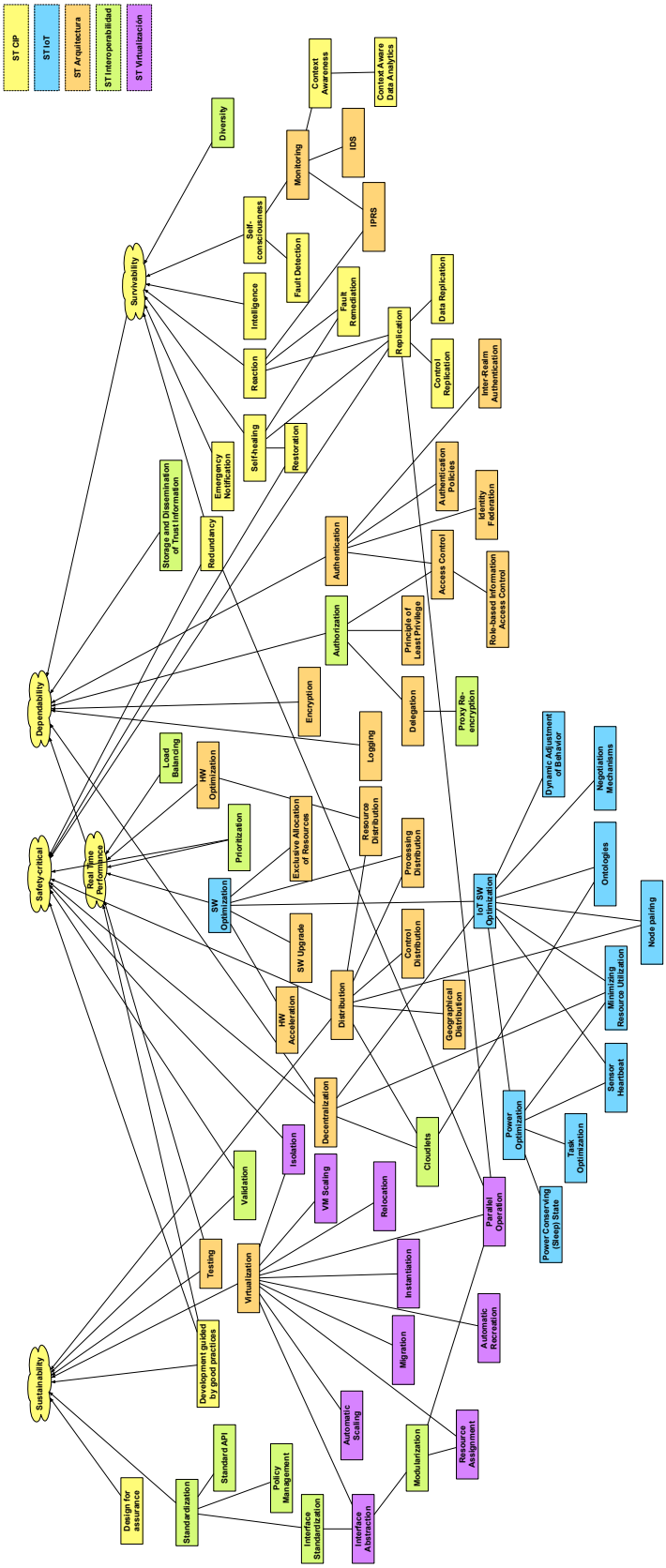


Figure 3.5: Satisficing techniques for the interoperability architecture

Table 3.2: Satisficing techniques for interoperability

CIP Requirements		Techniques		
Dependability	Real-Time Performance	HW Optimization	Resource Distribution	
		SW Optimization	HW Acceleration	SW Upgrade
			Exclusive Allocation of Resources	Task Optimization
			Power Conserving State	Power Optimization
			IoT SW Optimization	Sensor Heartbeat
			Negotiation Mechanisms	Ontologies
			Minimizing Resource Utilization	Node Pairing
			Dynamic Adjustment of Behavior	
		Use of good practices		
		Testing		
		Prioritization		
		Load Balancing		
	Survivability	Redundancy	Parallel Operation	
		Reaction	Fault Remediation	Replication
			Replication	Data Replication
			Control Replication	
		Self-Healing	Restoration	
		Self-Consciousness	Fault Detection	Monitoring
			Context Aware Data Analytics	IDS
			Context Awareness	IPRS
		Intelligence		
		Emergency Notification		
		Diversity		
		Authorization	Access Control	Proxy Re-encryption
			Principle of Least Privilege	Delegation
			Role-based Information Access Control	
		Authentication	Inter-Realm Authentication	Identity Federation
			Authentication Policies	
		Decentralization	Minimizing Resource Utilization	IoT SW Optimization
			Power Optimization	Task Optimization
			Power Conserving State	Sensor Heartbeat
			Negotiation Mechanisms	Ontologies
			Node Pairing	Cloudlets
			Dynamic Adjustment of Behavior	
		Logging		
		Encryption		
		Storage and Dissemination of Trust Information		
Safety Critical		Decentralization	Minimizing Resource Utilization	IoT SW Optimization
			Power Optimization	Task Optimization
			Power Conserving State	Sensor Heartbeat
			Negotiation Mechanisms	Ontologies
			Dynamic Adjustment of Behavior	
		Replication	Control Replication	Data Replication
		Distribution	Cloudlets	Node Pairing
			Geographical Distribution	Resource Distribution
		Redundancy	Parallel Operation	
		Restoration		
		Prioritization		
		Isolation		
		Validation		
		Use of good practices		
Sustainability		Standardization	Standard API	Policy Management
			Modularization	Resource Assignment
			Interface Standardization	Interface Abstraction
			Parallel Operation	
		Virtualization	Automatic Scaling	Migration
			Automatic Recreation	Instantiation
			Relocation	Virtual Machine (VM)
			Isolation	Scaling
		Distribution	Cloudlets	Node Pairing
			Geographical Distribution	Resource Distribution
		Validation		
		Testing		
		Use of good practices		
		Design for assurance		

critical control systems in general terms, it is also depicted in Figure 3.6.

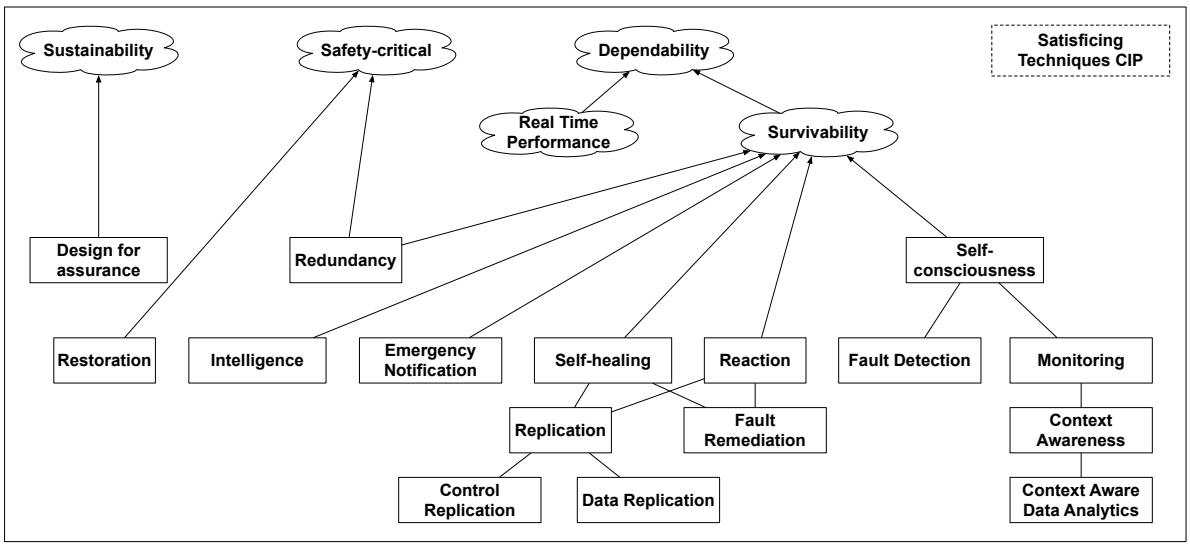


Figure 3.6: Satisficing techniques for the subjacent infrastructure

Satisficing techniques colored orange indicate the ones that provide architectural robustness and integrity to the interoperability platform and the communication infrastructure; we extract these techniques into Figure 3.7. In green, we show the techniques that enable the interoperability itself, described also in Figure 3.8. Colored blue are the satisficing techniques that help the integration of industrial IoT nodes within the infrastructure, which we can also see in Figure 3.9. Purple represents the techniques related to the virtualization of services within the platform, available in Figure 3.10.

To better understand the scope of the satisficing techniques, we provide a brief description of some of them, focusing on the most important or representative ones for our scenario. The first group of satisficing techniques described correspond to the ones in Figure 3.6, where we find techniques interesting for a CIP scenario. Among them we have, for example, *Design for Assurance*, which refers to provisioning evidence for compliance to governing rules, and that the governing rules provide appropriate grounds for trustworthiness [130]. It is based on assurance cases, which make easier the accountability and the evaluation of the compliance to good practices and standards easier. Following good practices of design oriented for assurance helps the resulting system to improve interoperability, given that this compliance to governing rules often implies the application of guidelines and standards which helps the construction of more compatible and reliable systems.

Another interesting technique is related to *Restoration*, referring to mechanisms capable of returning a system to a former or original condition after the occurrence of a failure. Restoration is central to the need for mitigation and recovery techniques. International organizations provide guidelines to improve recovery capabilities in CIs [15, 131]. Also *Intelligence* generates solutions capable of adapting to new circumstances and acting more efficiently against any anomalous occurrence. *Machine learning* (ML) techniques can provide the system with the necessary intelligence to perform automatic actions based on computed decisions taking into account multiple sources of data and a high number of interdependencies and constraints. These tasks

can be very complex and escape the possibilities of a human operator, hence Intelligence solutions are very important for complex interconnected scenarios.

In a CIP environment, implementing *Redundancy* is of paramount importance. It is defined as the inclusion of extra components that are not strictly necessary for the normal operation, in case of failure. When the primary devices stop working due to a fault, the secondary components are activated to maintain the normal operation of the system, while the primary ones are under repair. Redundancy can be implemented by introducing exact copies of primary components in the system, or using components of different natures as redundant ones in order to maximize diversity. Related to this technique, we have *Replication*. Replication implies providing multiple identical instances of the same system (or task), all of them running in parallel. Replication benefits performance and availability (see Section 3.1.2), since replicated components help when there are peaks of activity. The use of replication implies that all the replicated systems are always running to balance their workload, in contrast to redundancy, where the additional components are put in place to ensure the continuation of the operation even if a system is brought down by a failure. Two forms of replication can be implemented: *Control Replication*, where the SCADA processes are replicated, and *Data Replication*, where the important information is stored in different separated systems.

Regarding the protection of the CPCSs, we can conform a group of techniques to ensure the effectiveness of the mechanisms put in place, such as *Emergency Notification*, which refers to the capability of providing timely and descriptive alerts and notifications of the important events occurring within the system architecture. These notifications must contain sufficient information to allow the human operators in charge of managing these alerts to take appropriate and rapid actions to mitigate the threats and avoid cascading failures between failing interconnected systems. Also *Reaction* mechanisms, defined as "*a response to some foregoing action or stimulus*" [120]. In this scenario, we define reaction as the capability of the system to react against anomalous and/or malicious events, through the IPRS modules deployed across the interoperability platform.

Self-Consciousness, which allows the system to continuously monitor itself and its internal states, and *Self-Healing* (self-healing systems are able to detect a malfunction and to react to it, returning to their normal status and operation) techniques also help build a secure interoperability system, protecting the underlying CCS. Self-consciousness allows each component of the system to provide early detection capabilities and build security into the platform, and self-healing complements self-consciousness and restoration, providing the components of the platform to be able to recover from anomalous dynamics, thus preventing cascading failures and supporting the service continuity.

In line with the self-consciousness and self-healing techniques, we have the *Fault Detection* and *Fault Remediation* satisficing techniques. The interoperability system must be aware of its behaviors and internal states, thus being able to detect any fault occurring within it. In order to avoid any damage occurring from these faults, it is important to take rapid action and mitigate the errors in the system before it cascades and threatens the normal operation of the system. Additionally, and related to the *context awareness* softgoal identified in Section 3.3, the interoperability architecture have to implement a *Context Awareness* satisficing technique which allows the system to analyze and monitor its environment and the behavior of their surroundings, i.e., the context, in order to rapidly detect problems that arise, and avoid any malfunctions

derived from these events. This technique is highly related to the previous techniques (i.e., self-consciousness, fault detection), and all together provide security through prevention to the infrastructure.

In order to achieve a context aware environment, it is important to analyze the information behavior of the system and its surroundings. This information comes in the form of raw Big Data, thus, it is important for the system to be able to examine this information and extract conclusions from the data, taking always into account the contextual information of the system. The technique which helps us achieve this understanding of the contextual information is referred to as *Context Aware Data Analytics*.

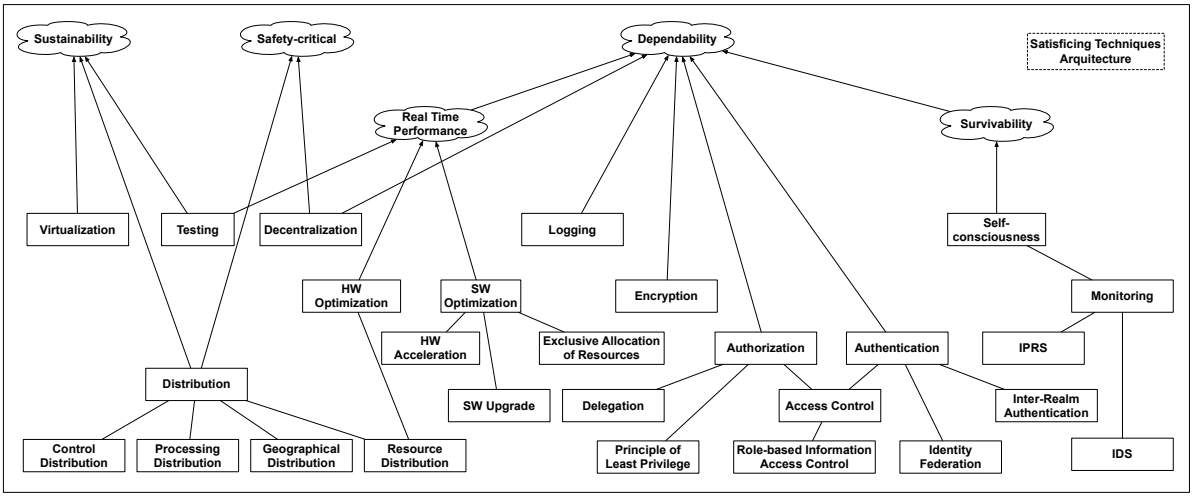


Figure 3.7: Satisficing techniques for the communication infrastructure

Once understood the first set of satisficing techniques, we focus on a second set containing those techniques which have an impact on the communication infrastructure (represented in orange in Figure 3.5). To better understand their dependencies and relations, we have extracted them to Figure 3.7, and we continue with our analysis by providing definitions for some of the more representative ones for our scenario, e.g., the monitoring mechanisms. *Monitoring* provides continuous analysis and vigilance of the surveilled system dynamics in order to detect and understand possible threatening occurrences. This technique was described first as a softgoal in Section 3.3, and as a satisficing technique in this section. There are a wide variety of monitoring tools and solutions which allow to better understand the inner dynamics of the surveilled system, as well as its environment. Examples of these tools are the IDS and IPRS solutions, which provide protective and defensive mechanisms that, put in place in a CPCS network or host, monitor the system and/or its environment in order to detect threatening events (IDS) and provide countermeasure actions to mitigate their effects (IPRS). The implementation of these techniques are of high importance in an interconnected heterogeneous environment such as the SG.

Within this second set of techniques we find two of them which are extremely important for our scenario, i.e., *Decentralization* and *Distribution* mechanisms. Decentralization, previously defined as a softgoal, consists on the implementation of methods that allow the transference and distribution of functions over the different nodes of a network in order to balance the workload

and avoid centralized authorities which are vulnerable to faults and attacks. Distribution refers to the implementation of mechanisms to communicate and coordinate network nodes in order for them to be able to perform concurrent cooperative tasks. Three types of distribution can be implemented [125]: *Resource Distribution* or *modularization*, where the system is composed of multiple distributed processing resources interconnected to form a single system with integrated control and system transparency to users; *Processing Distribution* or *parallelism*, where the systems cooperate to solve a common problem; and *Control Distribution* or *autonomy*, where the system is configured to perform autonomous operations based on distributed symmetrical control and message-passing communication protocols among resources or the system.

Also, we find that *Delegation* mechanisms can result of interest for this scenario. Delegation is defined as the authorization to represent others, or do something as a representative [120]. In the context of computer security, it refers to the process of allowing another user of process to use other user's credentials or permissions to perform a task. This technique is highly useful in critical situations, where a system administrator with the proper credentials might not be present at the time of the emergency, thus the permissions to mitigate the critical situation should be delegated to other available operators. Complementing this technique we have the principle of least privilege (as seen in Section 2.4), which conveys the need that in a computer system, especially in a critical environment, each module, function or piece of information should only be accessed by those actors in the system (users, processes) which have a legitimate purpose for accessing or using it.

To aid delegation, we can implement a *Proxy Re-Encryption* (PRE) mechanism, which is a type of Public-Key Encryption that provides a "re-encryption" functionality allowing a proxy to transform ciphertexts encrypted under a public key, into ciphertexts decryptable by another key. To do so, the proxy must be in possession of a re-encryption key that makes this process possible, while at the same time making it impossible for the proxy to learn any information about the encrypted data [132].

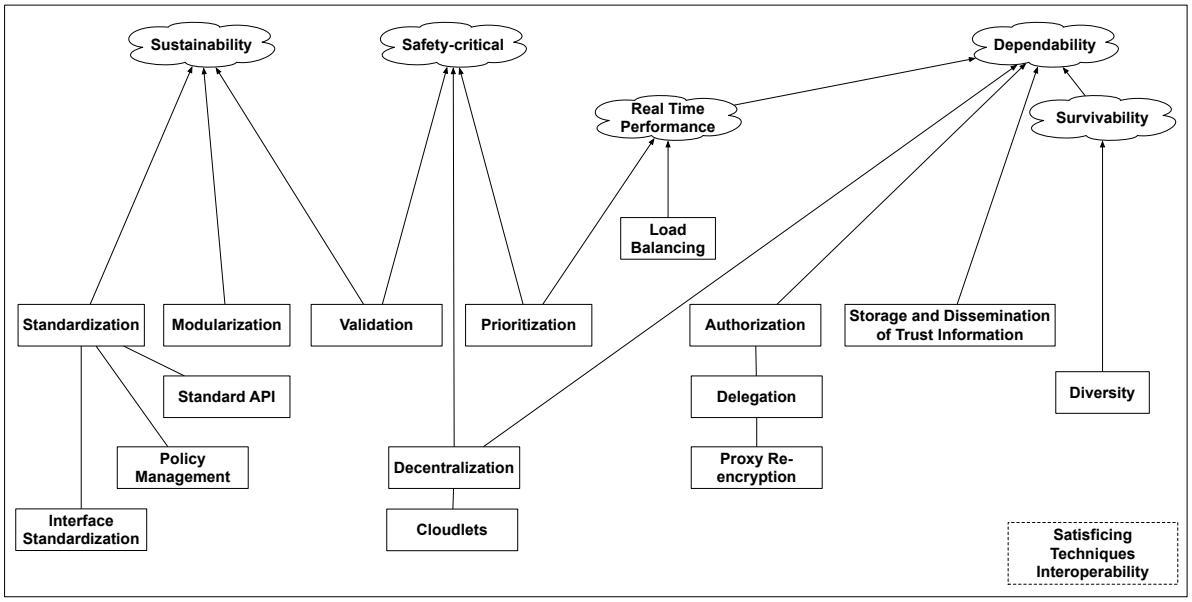


Figure 3.8: Satisficing techniques for interoperability

The third set of satisficing techniques we have identified is related to the interoperability of the systems within the platform. These satisficing techniques are depicted in color green in Figure 3.5, and extracted for clarity and understandability in Figure 3.8. Here we can propose some definitions for the most representative ones for our work, e.g., *Prioritization*, defined as the establishment of priorities among processes in a system, to ensure that the most critical ones always have available the assets they need to operate properly. In CCSs, critical tasks need to be taken care of as soon as possible, this is made possible through the organization of tasks according to their priority.

Another interesting technique in this set for our scenario is *Diversity*, which provides different implementations of the same specification (HW or SW), and uses them as replicated systems to cope with errors in a specific implementation. Diversity complements the techniques of replication and redundancy. An additional mechanism to help the interoperability of our system is *Policy Management*, which refers to the definition, management, update and report on the status of system policies. This capability must be present in all CSs, where the policies are defined by the organization to which the infrastructure belongs. Security policies, especially, must be up to date and be well defined in order for the system to be secure and interoperable with external modules and devices.

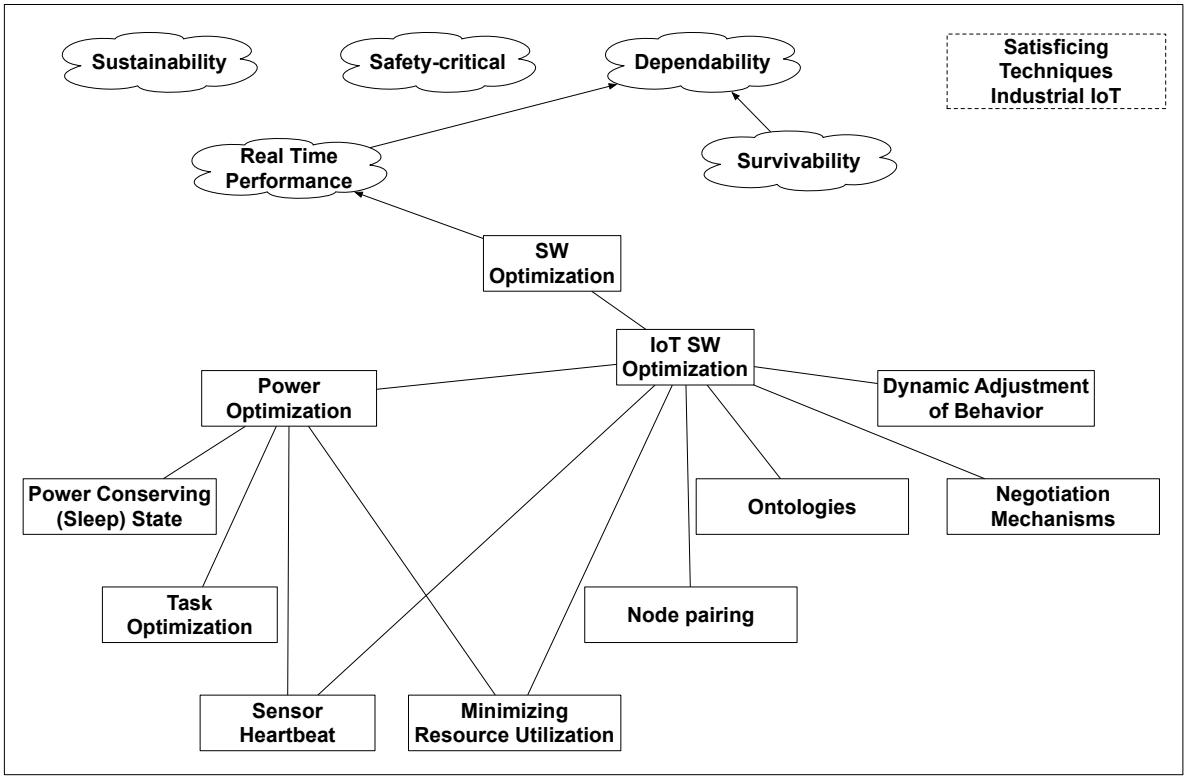


Figure 3.9: Satisficing techniques for the industrial IoT

Following the next set of definitions in Figure 3.5, we now describe several satisficing techniques proposed for the compliance of the Industrial IoT systems within the interoperability platform. They are marked in blue in Figure 3.5, and extracted to Figure 3.9 for better comprehensibility. Among them, we discuss three of them for our scenario, i.e., *Sensor Heartbeat*, *Node Pairing*,

and *Negotiation Mechanisms*.

The good health and correct operation of WSNs, and networks of IoT devices in general, is monitored through the emission of periodic signals of *liveliness* to other nodes in the network, denominated "heartbeats", where the device communicates its presence in the network, and goes back to sleep. We, then, refer to the technique which achieves this monitoring as "sensor heartbeat". Node Pairing is the technique in charge of effectively establishing a communication between two nodes in a network, in order for them to exchange information. This process need to be secured in order to avoid the flooding of false information through the network with malicious purposes. And negotiation mechanisms are those procedures taken, usually by two nodes at a time, in order to establish a communication between them. In this negotiations, the nodes agree the usage of common means for the communication, e.g., the communication protocols, or the security policies and mechanisms. This is usually done automatically, and the systems *adjust their behavior dynamically* according to the parameters of the negotiation. This procedure also needs to be secured, in order to avoid a malicious node to force insecure means of communication to perform attacks to the system.

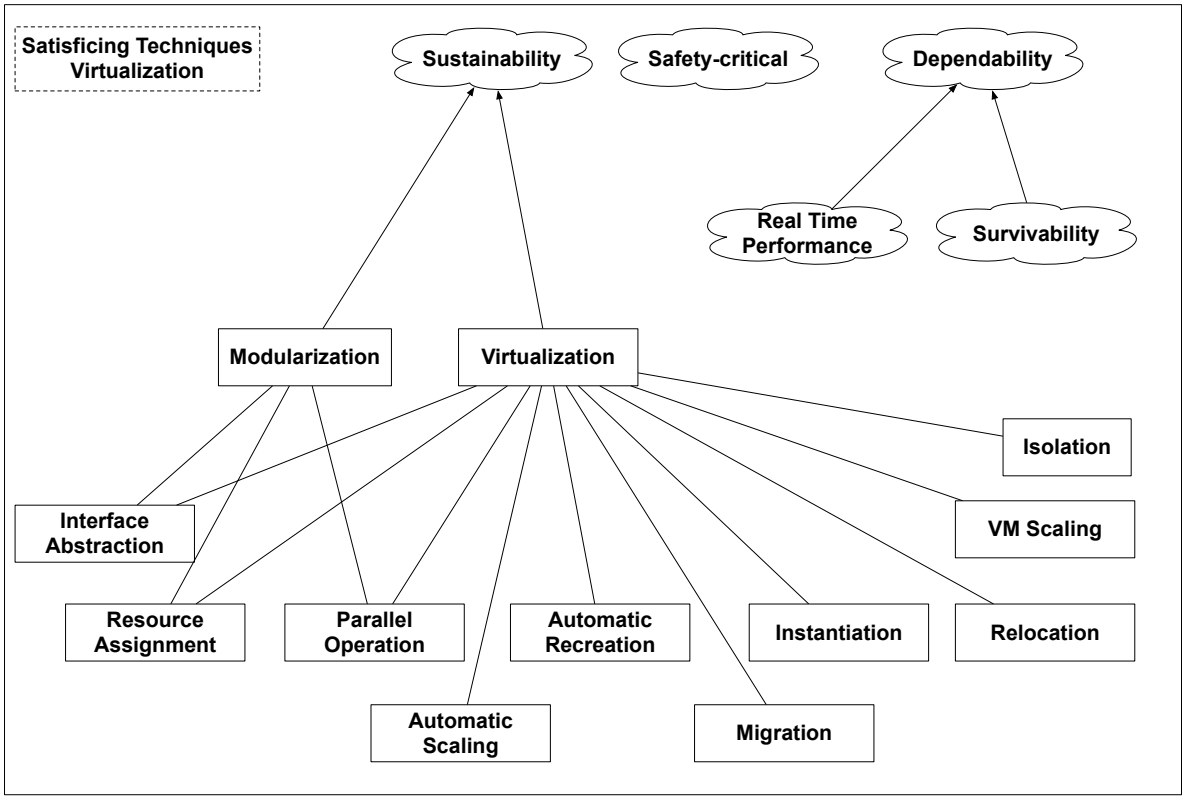


Figure 3.10: Satisficing techniques for virtualization

The last set of satisficing techniques studied is the ones referred to the virtualization of services and functions in the network. They are colored orange in Figure 3.5 and extracted for clarity into Figure 3.10. These techniques are common in virtualization scenarios, we can find their descriptions as well as interesting uses in the literature (see [126]).

3.5 Metrics for Interoperability

In the previous sections, we have focused our efforts on the analysis of the different variables that influence and constrain the construction of a secure decentralized interoperability architecture for the CSs of the smart grid. We especially aim to better understand the requirements that constrain the inclusion of interoperability mechanisms for the CPCSs of different domains of the SG. In Section 3.1, we reviewed the special requirements and constraints that are present in any critical scenario.

Following the NFR Framework [119], which allows us to model the softgoals of the system, in Section 3.3 we provide a detailed analysis of the NFRs for the interoperability of the CPCSs of the SG. Section 3.4 reviews those satisficing techniques that aim to provide means and tools to fulfill the previously identified requirements, inspiring the materialization of the operational level on the GQM approach [129]. Following this methodology, we now address the next level of the GQM, the quantitative level. To this end, we create a theoretical interoperability scenario for the CPCSs of the SG, where we consider the interactions of different key elements belonging to the Industry 4.0 environment. These architectural components appear at different domains in the infrastructure and have different scopes (see Figure 3.11) in a proposed (fourth generation) CS for the SG:

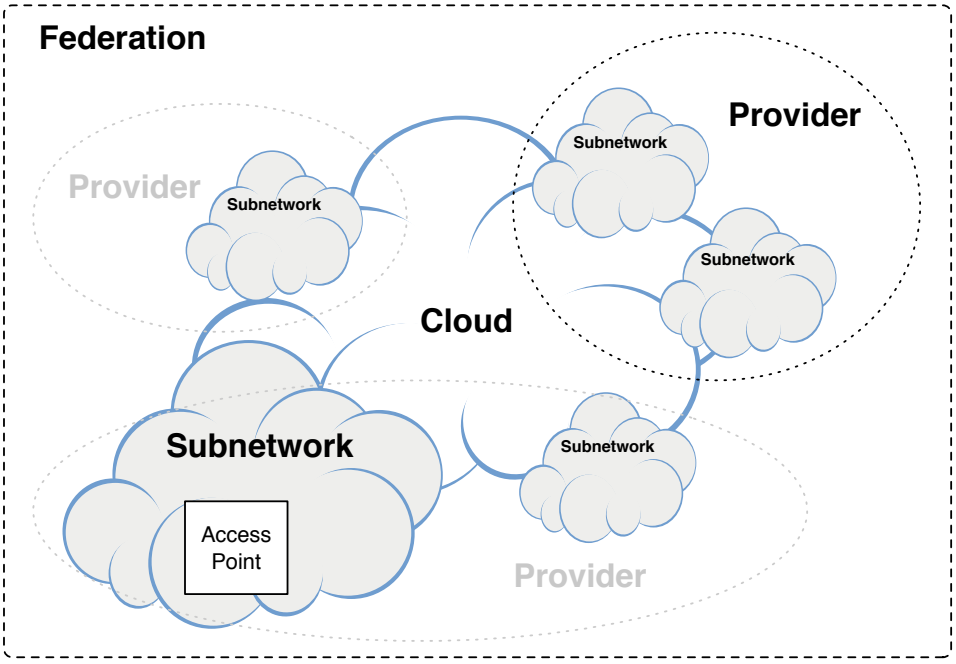


Figure 3.11: Interoperability architecture for the CPCSs of the SG

- *Access Point*: the access points have the most reduced scope in the infrastructure, they perform different interoperability functions at a local level, there are several access points per subnetwork.
- *Subnetwork*: are the building units of the interoperability platform, each subnetwork perform different tasks for the control and interoperability of the SG. Each subnetwork belongs

to one or more energy providers.

- *Provider*: the energy providers are the corporative actors of the infrastructure, the equipment in each of their subnetworks are controlled by a provider or a collaboration of several of them.
- *Federation*: a federation of energy providers is the entity that encompasses different energy providers which establish cooperation mechanisms among them. They may share equipment and resources as part as the cooperation mechanisms.
- *Cloud*: is the global infrastructure that acts as means of cooperation and communication between the shared resources and means of the federation of providers, the cloud can perform multiple interoperability operations while maintaining a global perspective of the interoperability of the infrastructure.

Once this scenario is set, we devote this section to the identification of interesting metrics which help evaluate the operation of this theoretical interoperability architecture for the CPCSs of the SG. Metrics, as denominated in the field of SW engineering, are quantitative measures of the degree to which a system or process has a given property. For our purposes, we consider a metric as an evaluation method for assessing the level of satisfaction of certain non-functional properties in a quantitative or qualitative way, on the basis of evidence and contextual input, like, for example, stakeholder's criteria [128].

In our analysis, we provide examples of sets of metrics for evaluating some of the main NFRs identified in Section 3.3, i.e., the main high-level sets of softgoals, which include different sets of softgoals inheriting their properties (see Figures 3.3, and 3.4). In our analysis of metrics, we connect sets of metrics and sets of softgoals as a variation of the GQM approach, because in GQM a set of metrics is used to evaluate each operational question of the model.

We make, however, this high-level connection of sets of metrics and softgoals, in relation to the satisficing techniques. Otherwise, following the in-depth GQM analysis, the extension of this study would be too extensive to be included in this thesis. Our different sets of metrics are mere suggestions and examples, we understand that we have not been exhaustive, however we do refer to different standards and guides where it is possible to find extensive information and different implementations and formulae expressing detailed metrics within these sets.

We divide our study into 10 different sets of metrics according to the main softgoal categories in Section 3.3: *maintainability*, *reliability and availability*, *performance*, *safety and dependability*, *responsiveness*, *auditing*, *interoperability*, *virtualization*, *monitoring and restoration*, and *authorization and delegation*. In addition to the analysis and definitions of the metrics, in order to provide certain level of validation to the identified sets of metrics, we incorporate to our study a set of controls or informative references where it is possible to find guidelines and useful information in how to apply and construe a given metric.

These controls will help determine the formulation of the metric, its scope in the secure interoperability infrastructure, and support its application and its usefulness in this environment by covering the identified need for this measurement with references to its application in the literature. Most of these controls come from prominent control frameworks, such as the NIST *Framework for Improving Critical Infrastructure Cybersecurity* [131], and different representative standards and guidelines for the security and interoperability of the critical infrastructures,

e.g., NISTIR 7628 [20], NIST SP800-82 [112], the NISCC Guide [133], IEC/TS 62351 series [134], NERC-CIP [113], ISO/IEC 27002 [135], IEEE 2030 [11], NIST SP800-144 [127] or the NFV Guides [126].

The NIST Framework [131] contemplates five high-level functions for cybersecurity analysis: *identify* (ID), *protect* (PR), *detect* (DE), *respond* (RS) and *recover* (RC). Within the analysis provided, this framework provides 6 different standards and guidelines for controls, i.e., CCS CSC [136], COBIT 5 [137], ISA 62443-2-1:2009 [138], ISA 62443-3-3:2013 [139], ISO/IEC 27001:2013 [140] and NIST SP800-53 [141].

Taking into account the aforementioned proposed structure of analysis for the metrics, we provide an overview of a first set of metrics in Table 3.3, which are related to the maintainability requirements of the secure interoperability platform (see Section 3.3). In the table we can review the scope of each metric in our scenario, the requirements to which the metrics are related, and the set of controls which help define and apply them. Here we can highlight some examples of them, providing a brief high-level description of the metrics, which can be adapted and specified on a later refinement step of specification:

- *Restoration Time*: is the estimated time required for a system to be restored to its original operation, in the case of failure. This metric has an important impact on availability, thus the infrastructure should have support mechanisms put in place to reduce its time and impact as much as possible.
- *Repair Time*: also "*Mean Time To Repair*" ($MTTR_1$) is the average time required to repair a system that has failed. $MTTR_1$ has to be kept to a minimum in critical systems, which implement redundancy and replication mechanisms to compensate the failure of a single component.
- *Recovery Time*: also "*Mean Time To Recovery*" ($MTTR_2$), refers to the average time a given system will take to recover from a failure. Similar to $MTTR_1$, it indicates the time lapsed before the system returns to its normal operation. It has to be kept under a given threshold, in order to avoid failures cascading to other dependent systems.

This set of metrics described above are, as we discussed, high-level metrics (we do not provide concrete formulae, since they can be formulated in several ways according to the scenario of application), however we find that these are a representative example of metrics useful for analyzing the maintenance needs of a critical system. To better understand the metrics and their application to this environment, it is possible to refer to the following standards and guidelines: IEC/TS 62351-7 [142], NISTIR 7628 [20], NIST SP800-82 [112], NIST SP800-53 [141], IEC/TS 62351 [134], NERC-CIP [113]. And the following categories of the NIST Framework for Improving Critical Infrastructure Cybersecurity [131]: ID.AM, RC.RP, RC.IM, RC.CO, PR.IP-7, PR.MA; corresponding to the following standards and guidelines: CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], and NIST SP800-53 [141].

The next set of metrics analyzed corresponds to the reliability and availability metrics, represented in Table 3.4. This set contains an example of interesting measurements that can be taken in a secure interoperability platform for the CPCSs of the SG in order to analyze the fulfillment of the reliability and availability requirements. We propose several definitions as examples for these metrics:

Table 3.3: Maintainability metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Installation SW Cost	•	•			•	Maintainability, Modularity, Virtualizability	CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], NIST SP800-53 [141], IEC/TS 62351-7 [142], NISTIR 7628 [20], NIST SP800-82 [112], NIST SP800-53 [141], IEC/TS 62351 [134], NERC-CIP [113]
Installation HW Cost	•	•			•	Scalability, Virtualizability, Operational Environment Compatibility, Heterogeneity Management, Heterogeneity Coexistence	
Obtaining SW Cost	•	•			•	Scalability, Virtualizability, Operational Environment Compatibility, Heterogeneity Management, Heterogeneity Coexistence	
Obtaining HW Cost	•	•			•	Scalability, Virtualizability, Operational Environment Compatibility, Heterogeneity Management, Heterogeneity Coexistence, HW Dependability	
Maintenance HW Cost	•	•			•	Scalability, Virtualizability, Operational Environment Compatibility, Heterogeneity Management, Heterogeneity Coexistence	
Maintenance SW Cost	•	•			•	Scalability, Virtualizability, Operational Environment Compatibility, Heterogeneity Management, Heterogeneity Coexistence	
Planned Maintenance	•	•			•	Maintainability, Service Continuity, Service Performance, Service Assurance	
Restoration Time	•	•			•	Stability, Resilience, Reliability	
Repair Time	•	•			•	Maintainability, Restoration	
Maintenance Man-hours	•	•			•	Maintainability, Restoration	
Failures Over Time	•	•			•	Resilience, Fault Tolerance	
Recovery Time	•	•			•	Recoverability	
Upgrade Events	•	•			•	Heterogeneity Management, Extensibility, Scalability, Service Continuity	
Mean Down Time	•	•			•	Safety, Service Continuity	
Mean Time Between Failures	•	•			•	Safety, Service Continuity	
Number of Maintenance Operations per Month	•	•	•		•	Heterogeneity Management, Extensibility, Scalability, Service Continuity, HW Dependability	

Table 3.4: Reliability and availability metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Diversity	•	•	•	•		Heterogeneity Coexistence, Heterogeneity Management, Operational Environment Comptatibility, Interfaceability	COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140], NIST SP800-53 [141], CCS CSC [136], IEEE 2030 [11], ISA 62443-3-3 [139], NISTIR 7628 [20]
Replication	•	•	•	•	•	Safety Critical, Survivability	
Uptime	•	•			•	Availability, Service Continuity	
Downtime	•	•			•	Availability, Service Continuity	
Availability	•	•			•	Availability, Service Assurance	
Redundancy	•	•	•	•	•	Robustness, Resilience	

- *Diversity*: measures the number of different implementations of the same specification, the more diverse a system is, the more resilient to failures it is.
- *Replication*: measures the number of replicated systems that are present in the system. It can also be applied to a component, in order to identify its level of replication, e.g., RAID systems: level 0 to level 7.
- *Uptime*: is the measure of the time a system is working and available, representing the time it can work non-stop and without maintenance. Examples are: the percentage of time the system is running and number of hours uptime versus number of hours of outage/downtime.
- *Redundancy*: refers to the inclusion of extra components that are not strictly necessary for the normal operation, in case of failure. This metric measures the number of redundant components available in the system.

To better understand these definitions and the application scope of these metrics for the control systems, the following guidelines provide insight about the definitions and applications of these metrics: NISTIR 7628 [20], and the categories: PR.MA, PR.DS-5, PR.IP-8, PR.DS-3, PR.DS-4, PR.IP-4, ID.BE-4 of the NIST Framework[131], corresponding to the standards and guidelines: COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140], NIST SP800-53 [141], CCS CSC [136], and ISA 62443-3-3 [139].

In Table 3.5 we find a set of metrics related to the performance of the control systems in a secure interoperability platform. These metrics are high-level examples of the different formulated metrics that can be applied to measure this variable. Performance metrics are common in the literature, and their formulae can vary depending on the concrete scenario of application, here we propose the following definitions for some of the most interesting metrics in our scenario:

- *Performance per Watt*: is the rate of computation that can be delivered for each watt of energy consumed. A low ratio of performance per watt is an indicator of the suitability of a component for the CPCSs.
- *Relative Efficiency*: the relative efficiency of two procedures, known as the ratio of their efficiencies is frequently calculated as the comparison made between a given procedure and a notional "best possible" procedure.

Table 3.5: Performance metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Speed	•					Performance	IEC/TS 62351-3 [143], IEC/TS 62351-4 [144], IEC/TS 62351-5 [145], IEC/TS 62351-6 [146], COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140], NIST SP800-53 [141], NIST SP800-82 [112], NISTIR 7628 [20], NIST SP800-53 [141]
Completion Time	•	•			•	Performance, Service Assurance, Reliability	
Service Time	•	•			•	Performance	
Speed of Cryptographic Algorithms	•	•			•	Performance, Security	
Performance per Watt	•				•	Performance	
Bandwidth	•	•	•	•	•	Performance	
Mean Delay of General Traffic	•	•	•	•	•	Performance	
Channel Capacity	•	•	•	•	•	Performance	
Relative Efficiency	•	•			•	Performance	

In order to better understand these definitions and the scope of application of the metrics, we refer to the following standards and guidelines: IEC/TS 62351-3 [143], IEC/TS 62351-4 [144], IEC/TS 62351-5 [145], IEC/TS 62351-6 [146], NIST SP800-82 [112], NISTIR 7628 [20], and NIST SP800-53 [141]. And to the ID.AM-5 category of the NIST Framework [131], corresponding to the standards: COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140], and NIST SP800-53 [141].

Table 3.6: Safety and dependability metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Percent System Safety Hazards	•	•	•	•	•	Safety Critical	NISTIR 7628 [20], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], NIST SP800-53 [141]
Mean Time to Unsafe Failure	•	•	•	•	•	Robustness, Safety	
Degree of Redundancy per Access Point	•	•	•		•	Maintainability, Service Continuity	
Number of Network Domains		•	•	•		Decentralization	
Safety Design Stability	•	•	•	•	•	Safety	
Safety Integrity Level	•	•	•	•	•	Safety	
Time to Countermeasure	•	•			•	Robustness, Resilience	
Safety Requirements Traceability	•	•	•	•	•	Accountability	
Number of Countries			•	•	•	Responsiveness, Trust, Accesibility	

Table 3.6 presents a set of metrics related to the safety and dependability requirements of the

CPCSs interoperability infrastructure. These metrics are high-level examples of the different metrics that can be employed to analyze the fulfillment of these requirements in a real environment. We can highlight some of them and provide general definitions for them:

- *Mean Time To Unsafe Failure* (MTTUF): represents the average time that a system will operate safely before the occurrence of a failure that produces an unsafe system state [147]. MTTUF should be as high as possible, indicating that there are few probable unsafe failures for the whole system, and thus a robust dependable system.
- *Safety Integrity Level* (SIL): metrics that are available at IEC 61508 [148] allow developing systems observing the level of safety of the system. Solutions complying with the specifications of this standard will ensure that the SIL of the subsystem is adequate for the CCS.

To better understand this set of metrics, it is possible to consult the NISTIR 7628 [20] guidelines, and refer to the PR.IP-4 and ID.BE-4 categories of the NIST Framework [131], which correspond to the standards: COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140] and NIST SP800-53 [141].

Table 3.7: Responsiveness metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Latency	•	•			•	Performance, Responsiveness	NISTIR 7628 [20], COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140], NIST SP800-53 [141]
Response Time	•	•			•	Performance, Responsiveness, Availability, Virtualizability	
Throughput	•	•			•	Performance, Heterogeneity Management	
Schedulability	•	•			•	Performance, Responsiveness, Availability, Virtualizability	
Jitter	•	•			•	Performance, Responsiveness	

The set of metrics represented in Table 3.7 correspond to the ones related to the responsiveness of the system. Responsiveness is commonly addressed in the literature for network engineering. Within this set, we can highlight a metric very significative for the CIP scenario, *Response time*, which is defined as the time it takes to initially respond to a request for a service or to access resources [116]. It is vital for the availability (data and control) of the system.

The responsiveness metrics are reflected in the NISTIR 7628 [20] guidelines, and can be observed in the ID.AM-5 category of the NIST Framework [131], which refers to the following standards: COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140] and NIST SP800-53 [141].

The next set of metrics, illustrated in Table 3.8, is related to the auditing of the system. Auditing is vital for implementing accountability processes, and helps maintain a set of quality standards highly important for the correct operation of the system. In a secure interoperability platform for the CPCSs of the SG, it is important to implement and quantify these auditing mechanisms. Metrics relevant to this process could include the *Number of Auditors*, *Number of Audits per Year*, and *Number of Logs per Minute*. Controls and informative references for this set of metrics can be found in the NISTIR 7628 [20] guide, and in the PR.PT-1 category of the NIST

Table 3.8: Auditing metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Number of External Auditors			•	•	•	Accountability	NISTIR 7628 [20], CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], NIST SP800-53 [141]
Number of External Audits per Year			•	•		Accountability	
Number of Internal Audits per Year			•	•	•	Accountability	
Number of Logs per Minute	•	•			•	Accountability, Performance	
Number of Internal Auditors			•	•		Accountability	

Framework [131], which is related to the standards and guidelines: CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], and NIST SP800-53 [141].

Table 3.9: Secure interoperability metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Number of Subnetworks		•	•	•		Interoperability, Scalability, Modularity, Maintainability, Flexibility	COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140], NIST SP800-53 [141], ISO/IEC 27002 [135], IEEE 2030 [11], NISTIR 7628 [20], NIST SP800-82 [112], IEC/TS 62351 [134], NERC-CIP [113], NIST SP800-144 [127]
Time to Translation	•					Heterogeneity Management, Responsiveness	
Number of Protocols	•	•	•	•		Heterogeneity Coexistence	
Number of Different Types of Subnetworks			•	•		Trust, Interoperability	
Number of Security Policies	•	•	•		•	Interoperability, Heterogeneity Coexistence	
Number of Nodes	•	•				Distribution, Decentralization, Scalability	
Strength of Cryptographic Mechanisms	•	•			•	Security, Reliability	

In in Table 3.9 we find a set of metrics measuring the secure interoperability of the system. The proposed metrics examples are related to the characteristics of the different networks that coexist within an interoperability scenario for the control systems of the SG. Within this set (which can be expanded according to the application scenario) we can highlight two main metrics:

- *Number of Subnetworks*: measures the different types of subnetworks that coexist within a system. These networks might have different types of communications protocols (measured by the *Number of Protocols* metric), different security policies (measured by the metric *Number of Security Policies*) and have different characteristics, e.g., being a cabled or wireless kind of network (this can be measured by the metric *Number of Different Types of Subnetworks*).

- *Time to Translation*: in case of protocol conversion or message translation, this metric indicates the time lapsed from the emission of the message until it has been translated (or converted) to be sent to the destination using the destination's protocol or language. This metric is useful in an interoperability infrastructure, where multiple nodes communicate among them using different protocols.

In order to better understand these definitions and to find concrete examples of metrics for these categories, it is possible to consult the following standards and guidelines: ISO/IEC 27002 [135], IEEE 2030 [11], NISTIR 7628 [20], NIST SP800-82 [112], IEC/TS 62351 [134], NERC-CIP [113], NIST SP800-144 [127]. And also refer to the categories: ID.GV-1 and PR.IP-5 of the NIST Framework [131], which correspond to the standards and guidelines: COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140] and NIST SP800-53 [141].

Table 3.10: Virtualization metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Number of Configurations Backup per Minute	•	•			•	Resilience, Service Continuity, Accountability, Restoration, Safety Critical	CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], NIST SP800-53 [141], NFV Guides [126], NIST SP800-82 [112], NERC-CIP [113], NIST SP800-144 [127]
Velocity of New Service Deployment	•				•	Reaction, Observability, Awareness	
Number of Virtualized Access Points		•	•	•		Optimization, Virtualizability, Performance, Dependability	
Percentage of Virtualization		•	•	•		Heterogeneity Coexistence, Virtualizability, Dependability	
Number of Services in Public Cloud			•	•		Isolation, Trust, Security	
Number of Services in Proprietary Cloud			•	•		Virtualizability, Dependability, Maintainability, Interoperability, Scalability	

An important set of metrics in the scenario at hand (see Figure 3.11) is the virtualization metrics, which help understand the virtualization characteristics of the critical environment we describe. While virtualization metrics are common for multiple information technologies scenarios, measuring the flexibility and performance of the virtualized services is of upmost importance in a critical environment. The metrics represented in Table 3.10 are examples of the measurements it is possible to assess the virtualization of the system:

- *Number of Configurations Backup per Minute*: measures the number of backups of configuration files that are made per minute in the system. While this number can be reduced in a small system, when referring to a complex network such as the SG CPCSs, this can be a massive number of system backups. This metric indicates the complexity of the system and the backup and maintenance mechanisms implemented for its protection, as well as provides insight about the needs of backup management needed for the platform (e.g., dedicated servers with a broad bandwidth used for backups).

- *Velocity of New Service Deployment*: measures the time lapse needed for a virtualized service to be created and deployed within the system. This metric is related to the flexibility and elasticity of a virtualization system.
- *Number of Virtualized Access Points*: refers to the number of access points which are offered in a virtual rather than physical form. This metric is related to the *Percentage of Virtualization* metric, which indicates the number or percentage of virtualized services offered in a given network.
- *Number of Services in Public Cloud*: indicates the number of services that are deployed in a public cloud, in contrast to the *Number of Services in Proprietary Cloud* metric, which indicates the number of services available only in the private cloud. In a critical system, given to the sensitivity of the data managed, it is usually preferred to employ proprietary server infrastructures. However, less sensitive data can be transferred to the public cloud for storage and global availability.

Controls and representative information regarding these metrics can be found in the following standards and guidelines: the NFV Guides [126], NIST SP800-82 [112], NERC-CIP [113] and NIST SP800-144 [127]. Also it is possible to find information in the ID.AM category of the NIST Framework [131], corresponding to the standards and guidelines: CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140] and NIST SP800-53 [141].

Table 3.11: Monitoring and restoration metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Number of Critical Events per Minute	•	•	•	•	•	Awareness, Accountability	CCS CSC [136], COBIT 5 [137], ISO/IEC 27001 [140], NIST SP800-53 [141], ISA 62443-2-1 [138], ISA 62443-3-3 [139], IEC/TS 62351-7 [142], NISTIR 7628 [20], NIST SP800-82 [112], IEC/TS 62351 [134], NERC-CIP [113], NISCC Guide [133]
Number of Countermeasures	•				•	Reaction, Recoverability	
Number of Restoration Operations	•				•	Restoration, Recoverability	
Number of Packets Inspected per Second	•				•	Monitoring	
Number of Alarms per Minute	•	•	•	•	•	Monitoring, Accountability	

In order to assess the protection mechanisms implemented within a secure interoperability architecture, it is necessary to find metrics to measure the performance of the monitoring and restoration procedures deployed within the system. This set of metrics, presented in Table 3.11, constitutes an example of the different measurements that can be taken in order to evaluate the operation and suitability of these protection mechanisms within the infrastructure of CPCSs for the SG. We provide example definitions for some of them as follows:

- *Number of Critical Events per Minute*: measures the number of critical events occurring in a system, this gives indication of the vulnerability of the system in the face of external threats or system faults. In a critical environment, it is vital to maintain this number as

low as possible, being maintenance and upgrade operations adequate to solve the arising problems.

- *Number of Packets Inspected per Second*: measures the number of network packets that can be processed and inspected by a monitoring system per second. Techniques such as deep packet inspection helps the system to detect any abnormal behavior or event. A high throughput is desirable in a critical network, avoiding delays for this procedure.

To better understand these metrics and to broaden the definitions and scope of the metrics, it is possible to consult the following standards and guidelines: IEC/TS 62351-7 [142], NISTIR 7628 [20], NIST SP800-82 [112], IEC/TS 62351 [134], NERC-CIP [113] and NISCC Guide [133]. Additionally, these metrics are covered in the following categories of the NIST Framework: RC.RP, RC.IM, RC.CO, RS.RP, RS.CO, RS.AN, RS.MI, RS.IM, DE.AE, DE.CM, DE.DP, [131]. Which correspond to the standards and guides: CCS CSC [136], COBIT 5 [137], ISO/IEC 27001 [140], NIST SP800-53 [141], ISA 62443-2-1 [138] and ISA 62443-3-3 [139].

This last set of metrics is dedicated to the evaluation of the mechanisms of authorization and delegation implemented within the secure interoperability infrastructure of our scenario. These mechanisms are critical for the correct operation of the system, given that they provide redundancy and backup procedures for the access and authorization mechanisms of control of the system. Table 3.12 present an example set of metrics to illustrate the main measurements that is possible to make in order to evaluate the presence and performance of the authorization and delegation mechanisms, we specially highlight two of them, which are of paramount importance for a critical scenario:

- *Time to new Authorization in Critical Context*: introduces a variation on the *Time to New Authorization*, measuring the time it takes for the system to produce a new authorization, in the presence of a critical event occurring within the system. In a critical infrastructure, the authorization and permissions must be prioritized, i.e., non-vital procedures must be relegated to a second plane in the presence of a critical event, giving priority to those procedures (e.g., authorization) which will allow the staff to remedy the critical situation. Therefore, all critical authorizations must be provided as fast as possible.

- *Time to new Critical Authorization in Critical Context*: is the variation of the previous metric, *Time to New Authorization in Critical Context*, in the event that the profiles of the system that usually (and are usually authorized to) tend the critical events are not available, however a member of the staff with other type of profile is present and available to act to remedy the situation. In this scenario, the authorization system must issue a critical authorization for the new user, as rapidly as possible, but also, complying with all the security procedures of the infrastructure. This metric assesses the secure authorization mechanisms put in place in the infrastructure for these purposes.

To better understand these authorization and delegation metrics, it is possible to consult the controls in the NIST Framework: PR.AC and PR.AT [131]. Which refer to the standards and guidelines: CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140] and NIST SP800-53 [141]. It is also possible to find further information regarding these metrics and their implementation in the following standards and guides: IEC/TS 62351-8 [28], NISTIR 7628 [20], NIST SP800-82 [112], IEC/TS 62351 [134], NERC-CIP [113] and NIST SP800-144 [127].

Table 3.12: Authorization and delegation metrics

Metric	Scope					Requirements	Controls
	Access Point	Subnetwork	Provider	Federation	Cloud		
Time to New Authentication	•	•	•			Responsiveness, Service Performance, Distribution, Decentralization	CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], NIST SP800-53 [141], IEC/TS 62351-8 [28], NISTIR 7628 [20], NIST SP800-82 [112], IEC/TS 62351 [134], NERC-CIP [113], NIST SP800-144 [127]
Number of Users	•	•	•	•		Security	
Time to New Authorization	•	•	•			Responsiveness, Service Performance, Distribution, Decentralization	
Time to new Authorization in Critical Context	•	•	•			Responsiveness, Service Performance, Distribution, Decentralization, Performance, Safety Critical	
Number of Field Operators		•	•	•		Mobility, Security	
Number of Roles for Delegation		•	•	•		Decentralization	
Time to Delegation	•	•	•	•		Responsiveness, Service Performance, Distribution, Decentralization, Performance, Safety Critical	
Time to Reencryption					•	Security, Performance	
Time to new Critical Authorization in Critical Context	•	•	•	•		Responsiveness, Service Performance, Distribution, Decentralization, Performance, Safety Critical, Mobility	
Number of Role Managers		•	•			Security	
Number of Roles in the System		•	•	•		Security	
Number of Accesses per Minute	•				•	Service Performance, Performance, Resilience, Scalability, Availability	
Number of Security Managers	•	•	•	•		Security	
Number of Operations Authorized per Minute	•	•			•	Security, Accountability, Performance	

Throughout this section, we have detailed the main sets of metrics corresponding to the main requirements for the secure interoperability of CCSs for the SG previously identified in Section 3.3, the above-mentioned metrics are high-level sets of examples of metrics that can be further developed into concrete formulas for measurements within a given specific scenario for evaluation. This study intends to provide an overview of the main measurements that is possible to make in order to assess the fulfillment of the identified softgoals for our secure interoperability infrastructure.

Since they are high-level sets of metrics, we have not performed any validation procedures, instead, we provide a set of literature controls and significative references to important standards and guidelines in which it is possible to find documentation related to the different aspects to evaluate and assess (which can be translated as metrics) and their description and scope at each of the different levels of application (i.e., general cybersecurity, network protocol protection level, smart grid security level, etc.).

Further work can be done in the specification of the metrics to be employed within a determined scenario of application, and validation of these resulting metrics can be possible through a formal process of validation, or through an expert group interview process, such as the Delphi method [149]. However, given the stage of definition and refinement of our scenario, this process remains as future work.

3.6 Requirements for Protection Solutions Deployed within Critical Systems

Since CSs are considered critical assets with a great potential impact for the society, it is vital to protect their correct operation and integrity [21]. As reviewed in Section 1.2, the protection and establishment of security mechanisms for CSs from the design is one of the pillars and key characteristics of the new generation of CSs for the Industry 4.0. In previous sections, we have reviewed the requirements and techniques for the secure interoperability architecture design from a holistic perspective. However, a deeper analysis is in order, since it is necessary to include in our design advanced protection solutions to face the sophisticated menaces that threaten this new environment (see Section 2.4), and where their compatibility and suitability to the new protocols and needs has not been yet established.

Therefore, in this section we aim to analyze in detail these mechanisms that help protect the correct operation of the CSs and CPCSs and help prevent threatening events of diverse nature occurring within the infrastructure, and avoid the generation of cascading effects of threats through interdependent critical infrastructures. In order to do so, we extend the NFR study performed in Section 3.3, analyzing in depth those softgoals specially related to the security and protection of the CSs.

Traditional means of protection for CSs are usually based on the triad *firewalls*, *Demilitarized Zones* (DMZs) and *antivirus* [1, 117]. However, in the new context of Industry 4.0, where security has to be implemented by design, it is necessary to go a step further and deploy sophisticated awareness and protection mechanisms specifically designed and adapted to critical contexts [16, 150]. These mechanisms can be applied to prevention, awareness, reaction and

restoration procedures. In our work, we decide to focus on the area of monitoring and detection for prevention and awareness procedures, hence evaluating the technologies enabling these characteristics for the CCSs of the SG. However, this analysis is easily extrapolated to other modules for the rest of aforementioned procedures.

These awareness mechanisms focus of this work are the IDSs [1], a security layer (HW or SW), designed to detect ongoing intrusive activities in computer systems and networks [151]. In the context of CPCSs, IDSs are designed to monitor network or system activities for suspicious behaviors and produce reports to a management station. The IDS is also used for other purposes, such as identifying problems with security policies, documentation of threats and to dissuade individuals from violating security policies [152].

Therefore, in this section, we look at those requirements that IDSs should consider since they act as the main way of filtering and defense in CSs. We also explore the set of metrics that help determine the compliance level of the requirements of the IDSs in relation to the five control requirements. It is necessary to focus on detection technologies due to the current problems in finding a suitable IDS capable of offering sufficient means to not only understand any SCADA vulnerability and all SCADA traffic (i.e., property protocols), but also to guarantee sustainability, coexistence and scalability when other types of TCP/IP protocols (e.g., 6LowPAN for Internet of Things) have to be integrated within the system.

To the best of our knowledge, the current literature does not contain any extensive theoretical study on identifying those IDS requirements that have to be considered when implementing IDS solutions for ICSs. There are only specific-purpose approaches restricted to a set of factors [117],1805, e.g., the type of observed protocol (e.g., the SCADA IDS defined within the SPARKS European project can identify anomalies associated with IEC-61850 and IEC-60870-5 networks [153]), the acquisition architecture (centralized, distributed), the level of scalability and detection granularity, the type of response (passive, active), or the degree of interoperability.

In an effort to deliver such point of view, we have modeled the requirements that any IDS solution has to fulfill to be deployed within a CI. This model takes into account the need to comply with the CCS requirements identified in [21], and the constraints of CCSs as described in Section 3.1.2. These constraints on the CCS impose powerful restrictions on the IDS solution which is deployed in the critical environment, then our model will help build a more secure control system and satisfy its requirements, through the identification and compliance with the identified requirements for the IDS.

In order to do so, we use the NFR model framework [119], as described in section 3.2, applying it to the analysis of the IDS solution suitability for the critical environment where it will be deployed. Building on Figure 3.1, which represents the requirements or softgoals for the a critical control system, we have Figure 3.12, which illustrates two main areas. The top half contains the CCS softgoals as identified in Figure 3.1, while the bottom half of the figure shows the IDS softgoals, needed in a critical scenario.

In other words, Figure 3.12 shows the application of the NFR Framework to describe the non-functional requirements or softgoals that the IDS needs to satisfy in order to comply with the CCS requirements. All the requirements are organized hierarchically and connected with other softgoals according to the definition and concepts of safety, security and survivability engineering [116]. The identification of the softgoals is a first step towards modeling the requirements of a

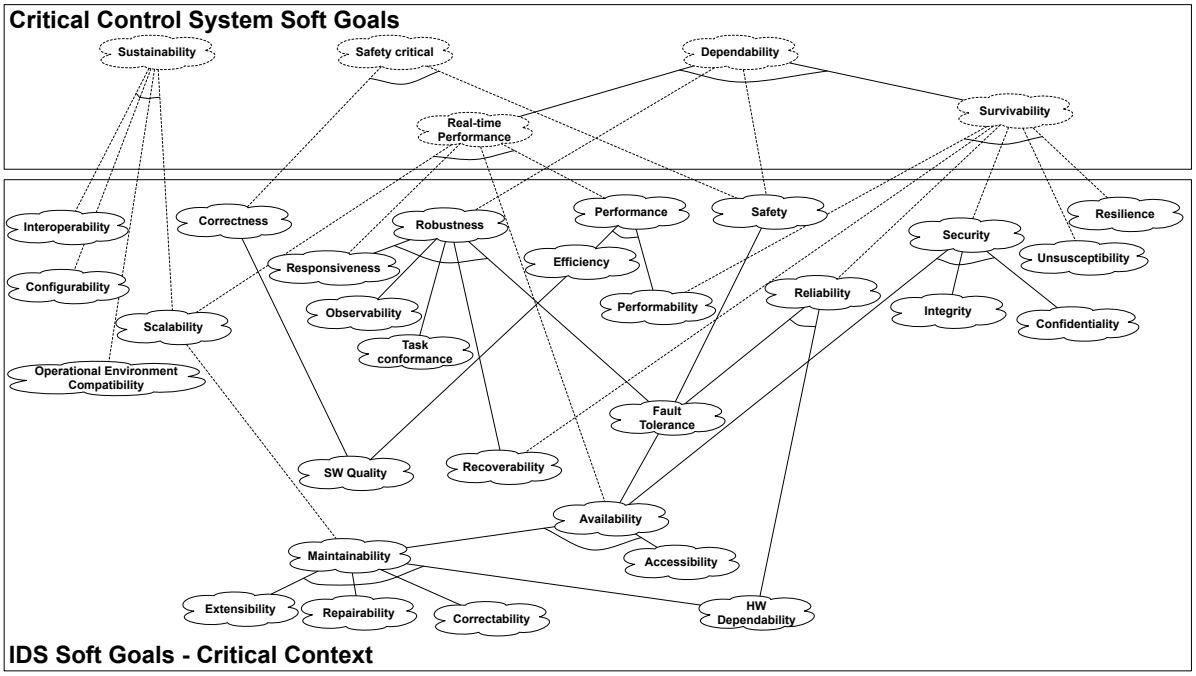


Figure 3.12: CCS and IDS softgoals

critical complex system. The next stage of our study is the definition of the NFR softgoals for those IDS to be configured within a CCS.

3.6.1 NFR Requirements for Protection Systems

In this section, we define the concepts involved in the NFRs identified in the model, in order to provide more information about the non-functional requirements selected for the IDS so as to comply with the requirements of the CCS (see Figure 3.12). Most of the definitions are already present in Section 3.3, however here we apply them to the context of protection systems. These definitions determine the scope of the requirements and will be used as a reference to later identify the techniques capable of satisfying the requirements. As we did in Section 3.3, to better understand the softgoals, we extract them and provide definitions for the most representative ones in the definitions Table 3.13.

Given the definitions presented in Table 3.13 and the representation of the softgoals illustrated in Figure 3.12, we can better understand the scope and definitions of the NFRs identified for an IDS for CPCSs. We have established and defined the characteristics present in a critical environment, and those requirements and constraints that restrict the inclusion of new components within such a scenario. In the next section, we will further develop our study in order to determine how to comply with the NFRs defined in this section.

Table 3.13: Requirements for protection systems definitions table

Softgoal	Definition and Description
Responsiveness	a responsive IDS will be capable of performing its functions in the required time intervals (responsiveness of the IDS), and without introducing further delays or overheads within the surveilled system (responsiveness of the infrastructure).
Safety	the safety and *safety critical* of the infrastructure must be observed at every moment by the security and protection modules deployed within it. No mechanisms implemented by these additional modules should introduce within the infrastructure variables capable of damaging the system or causing malfunctions in it in any way.
Security	this property is a primary requirement for defense and protection of control assets in critical systems, since the securement of CCSs is vital to maintain the normal operation of CIs [16]. Its variables, *Availability*, *Integrity* and *Confidentiality*, have to be continuously monitored in order for the system to be able to perform its tasks correctly. The protection systems such as the IDS are vital to maintain this security and awareness for protection.
Fault Tolerance	the ability of the system, in this case, the IDS, of continuing its operation despite the occurrence of failures or malicious attacks. Thus the infrastructure has to be fault tolerant, as well as its protection system. This softgoal is related to *Resilience* and *Recoverability* NFRs, which additionally imply the ability of the system to return to normal operation as soon as possible after a threatening event.
Scalability, *Extensibility*	softgoals that are mainly referred to the control infrastructure. However, the protection mechanisms put in place should also be scalable and extensible in order to be able to handle any growing amount of work due to the increase of traffic or tasks needed to monitored due to changes in the surveilled infrastructure behavior or configuration.
Maintainability	and its related softgoals (see Section 3.3) are directly applicable to the IDS component, since this module needs to be maintainable in order to be secure, correctly tuned to the surveilled system and to provide up-to-date protection measures. Especially important are the *Configurability* and *Accessibility* requirements for the IDS modules, which must be accessible to the system administrators to receive updates and modifications.
(Un)Susceptibility	unsusceptibility and tolerance against failures make the IDS system stable and avoid that a fault occurring within the protection system influences in any way the surveilled infrastructure.
Efficiency	this requirement, together with the *relative efficiency* softgoal indicate the need for the IDS component of being efficient and not introducing any overhead within the surveilled infrastructure. This is especially important in the case of critical systems, since their availability requirements for control and information are vital for the correct operation of the infrastructure.
Correctness	this requirement, together with its main factors, namely: *accuracy*, *currency* and *precision*, are vital for IDSs in order to guarantee reliability of data and operations. The correctness of the monitoring procedures allow the system to efficiently perform its tasks, issuing a reduced number of accurate alarms that inform the system administrators of the events threatening the correct operation of the surveilled system.
Operational Environment Compatibility	the IDS must be operationally compatible in order to be deployed within a CPCS, where machinery and environmental radiation and noise might interfere with the operation of unprepared systems.
Reliability	vital that the protection systems put in place are reliable, since any failure occurring within them can trigger a critical incident in the CPCS, causing great impact on the infrastructure and possibly cascading to interdependent critical systems.
Performance	and the related softgoals (see Section 3.3) are very important requirements for any module deployed within a CPCS, especially those modules providing security and performance. The adequate operation of the protection mechanisms influences directly the performance and correct operation of the surveilled infrastructure. The IDS components must adequately perform its tasks in the stipulated time and without introducing delays in the system.
Robustness	and its related requirements (see Section 3.3) for the IDS module have a direct impact on the security and the robustness of the surveilled CPCS. Similarly to *reliability*, the protection modules must be robust in order to avoid failures and threats to disturb the correct operation of the IDS. The occurrence of faults within the security component can impact the surveilled infrastructure in two different ways: (i) a failure to notify a threat in the CPCS can produce a fault in the infrastructure, (ii) a fault within the IDS might impact the correct operation of the surveilled infrastructure introducing a threat in the system. Either of these possibilities must be avoided, by deploying robust, reliable and fault tolerant protection mechanisms within the CPCS.
Software Quality	a way to comply with the previous requirement, is to consider the SW quality of the IDS, where its development is guided by good practices and guidelines [154], and resulting in a SW component that complies with standards and security guidelines.
HW Dependability	in the event the IDS is deployed on HW external to the surveilled infrastructure, it is necessary to ensure its dependability, and the compliance with the constrains of the CCS infrastructure.

3.6.2 NFR Satisficing Techniques for Protection Systems

In Section 3.6.1 we identified the requirements present in our scenario, taking into account both the requirements in CCSs and the ones that constrain the deployment of an IDS solution within a critical environment. The NFR Framework has provided us with tools to represent the NFRs, or softgoals, present in this scenario. However, we now need to go a step further in order to find specific ways to analyze whether these softgoals can be satisfied, and to find determined techniques or tools to help the IDS achieve this compliance.

As previously discussed in Section 3.4, the problem of NFRs is that these goals lack a clear definition, making difficult the assessment to determine if they have been satisfied. Thus, we tackle this problem in a similar way that in Section 3.4, where we use the *Goal Question Metric* (GQM) approach [129] to continue our study on the IDS softgoals for CIs, in order to bind these abstract characteristics to specific practices.

Instead of refining our analysis in terms of questions at the operational level of the GQM, and in line with the NFR Framework, we reflect on the operations taken for reaching the identified softgoals in terms of satisficing techniques. Thus, in this section, we aim to identify those techniques that can be implemented by the system (in our case, the IDS), capable of satisficing the established NFRs. Table 3.14 presents the simplified matching of the satisficing techniques found for the softgoals of the IDS, directly linked to the requirements of the CCS.

Table 3.14: Satisficing techniques

CCS Requirements		Techniques	
Dependability	Real-Time Performance	SW Optimization Desegmentation Load Balancing Use of good practices	HW Optimization Prioritization Testing
	Survivability	Redundancy Replication Restoration Self-Healing	Diversity Reaction Intelligence Self-Consciousness
Safety Critical		Isolation Replication Desegmentation Use of good practices	Redundancy Prioritization Restoration
Sustainability		Standardization Modularization Use of good practices	Testing Design for Assurance

This simplified matching to the NFRs of the CCS can be done because the use of these satisficing techniques for the IDS will, in turn, make the IDS comply with the requirements of the CCS. To better understand the scope of the satisficing techniques, we provide a brief description of the most representative ones for our scenario. It is important to note that most of the satisficing techniques have been described and defined previously in Section 3.4, however, here we provide definitions and clarifications for some of them according to the context of application of protection systems for the CPCSs.

Firstly, we understand that additionally to the *Prioritization* of the tasks of the CPCS, the protection mechanisms deployed within it must work accordingly and provide a prioritized set of alarms and warnings that clearly indicate the system administrator which ones are the most

critical ones in need of immediate attention in order to avoid further failures and cascading effects. *Modularization* mechanisms can be aligned with the previous technique to complement it. Modularization is the design of a whole system as a set of different modules. It makes the system versatile, providing the means for modifying the structure and functionality of the system by adding or removing modules. A modularized system is also easier to maintain, given that the complete functionality of the system is not compromised when a module needs modifications. Another interesting approach is the application of *Desegmentation*, which implies uncoupling the processes in a system, so that they are not interdependent. This technique builds *robustness* into a system, as independent processes are less likely to spread a cascading effect.

In line with the previous techniques, we have the satisficing techniques for the *HW Optimization* of the system, and its *Load Balancing*. It is vital to consider when deploying new protection HW within a critical environment, where legacy interfaces and protocols, and the physical environment dynamics may condition the characteristics and configuration of the new equipment. And in case of IDS modules, it is necessary to take into account the workload that the new component would need to manage in order to provide it with adequate resources in order for it to perform its tasks in an efficient manner, complying with the real-time performance requirements of the underlying CPCS. Taking into account the optimization of the system, we can implement *Diversity* measures. In the context of IDS, it is possible to provide diversity in the detection engines of the module, i.e., the introduction of signature-based detection together with anomaly-based detection (this concept will be further expanded in Chapter 4) helps the detection system discover threatening dynamics in different ways, therefore the detection mechanisms would be more complete and sophisticated.

To further preserve our system, the own protection system should implement safeguard mechanisms for itself, such as *Self-Consciousness*, which helps provide early detection capabilities, even in the presence of sophisticated or stealthy attacks (see Chapter 2) against the IDS. Additionally, it should implement *Restoration* and *Reaction* mechanisms. An important task of the protection modules correspond to the mitigation and recovery tasks, where restoration is vital. These techniques are mainly implemented by the IPRSs and restoration modules. IPRSs are specialized in reaction, and use the information offered by the IDS to decide the best response strategies. IPRSs for CPCSs are still subject to research, since automatic reaction in a critical environment could introduce new risks into the system and cause cascading effects.

Table 3.14 describes the simplified matching of the satisficing techniques to CCS requirements, now we model how these techniques satisfice the IDS softgoals, which, in turn is related to the CCS requirements. Figures 3.13 through 3.17 represent these relationships. We have divided the satisficing techniques and IDS softgoals taking into account the CCS requirements they are related to for the sake of clarity.

Figure 3.13 represents the relationship between the satisficing techniques and the NFRs related to the survivability of the CCS. The satisficing techniques which help the system to comply with these requirements are those that build resilience into the system, e.g., redundancy. Figure 3.14 presents the satisficing techniques linked to the real-time performance of the CCS. Some of the corresponding softgoals are related to the efficiency, performance and availability of the system, they try to optimize and balance the general performance of the system, to avoid peaks of demand that the system cannot respond to, e.g., SW and HW optimization and load balancing.

In Figure 3.15 we represent those IDS requirements linked to the safety-critical of the CCS. IDS

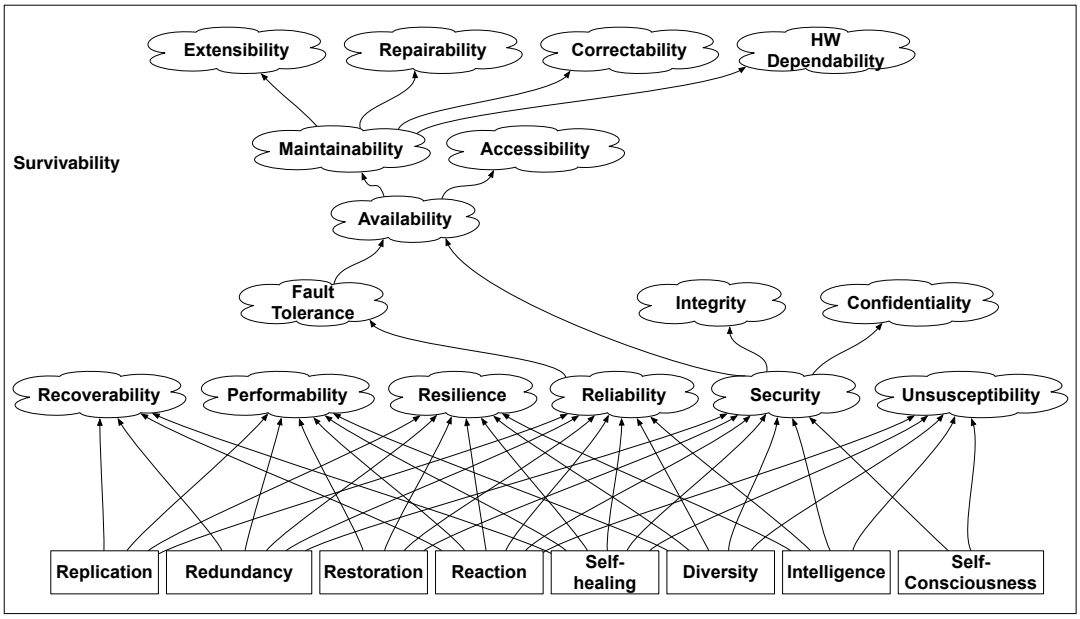


Figure 3.13: Survivability, softgoals and satisficing techniques

softgoals such as fault tolerance, safety or availability have a great impact on the safety-critical of the surveilled infrastructure. Thus, techniques focused on improving the IDS's safety and correct operation have a good influence on the general safety of the system, e.g., redundancy, replication, and development guided by good practices.

Figure 3.16 illustrates those IDS NFRs related to the sustainability of the CCS. Here, the IDS should comply with requirements such as maintainability, interoperability or scalability. Therefore the IDS should be designed in such a way as to make configuring and modifying its functionality easy. It should also make repairs or updates of their SW components easier so as to fix any problems of the system, as in modularized systems.

Finally, in Figure 3.17, we present those IDS satisficing techniques corresponding to the robustness of CCSs. Techniques that help achieve robustness are based on the design and development process of the IDS, especially the development guided by good practices. We can see in the figures that most of the aforementioned requirements and satisficing techniques overlap in the diagrams presented. This is due to the separation we previously introduced, to more clearly visualize our study.

From this study, it is possible to identify the great need for standardization and good practices when deploying new components within a critical scenario. These practices ensure that the CCSs are not affected by the addition of IDSs, and that new risks will not be introduced into the system as a source of unpredictable events or faults.

Additionally, as mentioned, there is an identified need for the CI to be protected. Some of the pillars that support this protection are the mitigation and response capabilities of the infrastructure [15, 131]. CPCSs are able to provide early detection and response to threats if their IDSs are equipped with sufficiently intelligent capabilities. The intelligence of the system will determine its adaptability to new circumstances and dynamics, providing the CPCS with tools

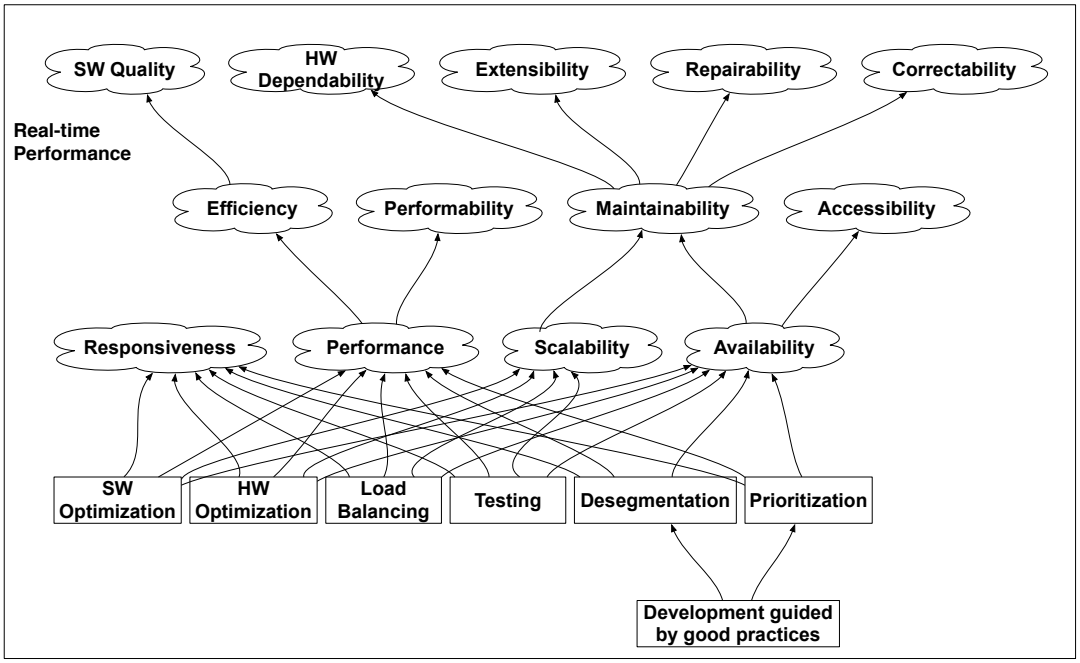


Figure 3.14: Real-time performance, softgoals and satisficing techniques

to react to unknown threatening events and restore the CI to its normal operation.

3.6.3 Metrics for Protection Systems

Throughout our approach, we have concentrated our efforts on studying the requirements present in CCS which influence and constrain the deployment of an IDS within the infrastructure (see Section 3.1). Once studied the NFRs and constraints of CCSs, we identified the requirements an IDS solution has to comply with to be deployed in this environment (see Section 3.6) following the NFR Framework [119]. In Section 3.6.2, we have outlined a set of satisficing techniques, inspiring the materialization of the operational level on the GQM approach [129]. And following this methodology, we now address the next level of the GQM, the quantitative level. This stage tries to identify different metrics to evaluate the suitability of the IDS for critical environments.

It is important to note, that most metrics have been already defined in Section 3.5 in the context of interoperability, or their definitions can be found in the general literature. However, in this section we try to give examples of application for these high-level metrics to the context of protection systems. Table 3.15 summarizes the different sets of metrics identified for this section, as well as illustrates the different standards and guidelines that serve as controls of such metrics for a further development and definition stage.

Reliability and Availability Metrics

In this subsection, we present metrics related to the *reliability and availability* of a system. These metrics can help determine and quantify the availability and reliability parameters of

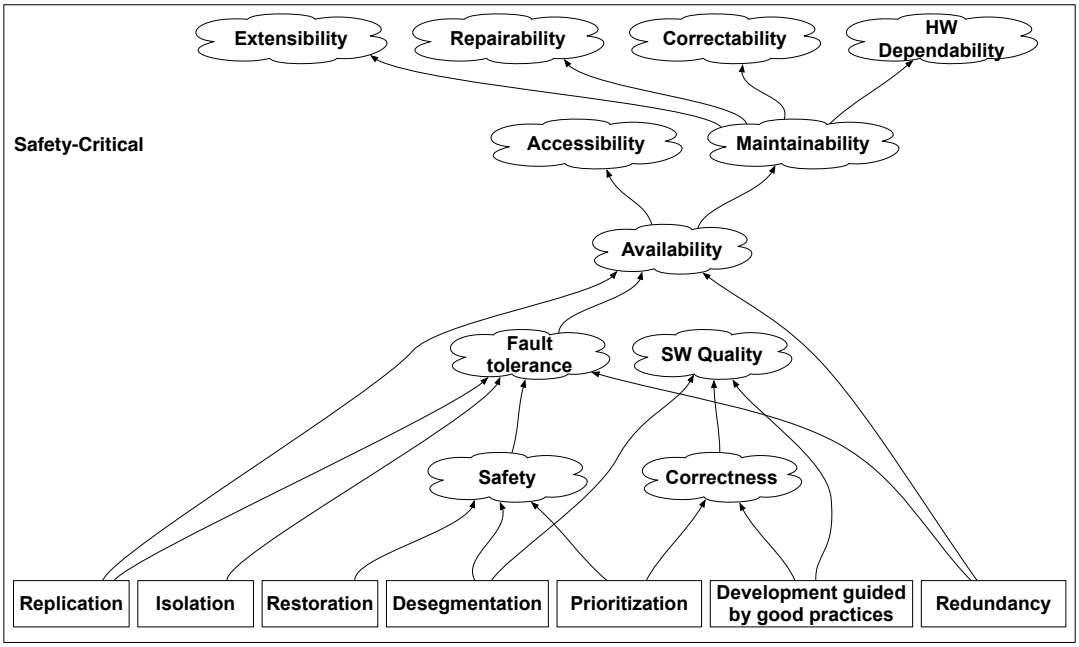


Figure 3.15: Safety critical, softgoals and satisficing techniques

Table 3.15: Metrics for protection systems

Metric Set	Examples of Metrics	Controls	
Reliability and Availability	Diversity, Replication, Uptime, Downtime, Availability	COBIT 5 [137], ISA 62443-2-1 [138], ISO/IEC 27001 [140], NIST SP800-53 [141]	CCS CSC [136], ISA 62443-3-3 [139], NISTIR 7628 [20], IEEE 2030 [11]
Performance	Compression Ratio, Speedup, Service Time, Instruction Path Length, Completion Time, Channel Capacity, Performance per Watt, Relative Efficiency, Bandwidth	IEC/TS 62351-3 [143], IEC/TS 62351-4 [144], IEC/TS 62351-5 [145], IEC/TS 62351-6 [146], COBIT 5 [137], ISA 62443-2-1 [138]	ISO/IEC 27001 [140], NIST SP800-53 [141], NIST SP800-82 [112], NISTIR 7628 [20], NIST SP800-53 [141]
Responsiveness	Jitter, Latency, Response Time, Schedulability, Throughput	NISTIR 7628 [20], COBIT 5 [137], ISA 62443-2-1 [138]	ISO/IEC 27001 [140], NIST SP800-53 [141]
Correctness	Accuracy, Precision, Currency, Sensitivity, Specificity	ISO 5725 Series [155], ISO 11843 Series [156]	ISO 2854 [157],
Maintainability	Planned Maintenance, Repair Time, HW Costs, SW Costs, Restoration Time, Failures Over Time, Maintenance Man-Hours, Upgrade Events, Recovery Time, Mean Time Between Failures, Mean Downtime	CCS CSC [136], COBIT 5 [137], ISA 62443-2-1 [138], ISA 62443-3-3 [139], ISO/IEC 27001 [140], NIST SP800-53 [141]	IEC/TS 62351-7 [142], NISTIR 7628 [20], NIST SP800-82 [112], NIST SP800-53 [141], IEC/TS 62351 [134], NERC-CIP [113]
Dependability and Safety	Mean Time to Unsafe Failure, Reliability, Stability, Safety Design Stability, Safety Requirements Traceability, Safety Integrity Level, Percent System Safety Hazards	IEC 61508 [148] , NISTIR 7628 [20], COBIT 5 [137], ISA 62443-2-1 [138]	ISA 62443-3-3 [139], ISO/IEC 27001 [140], NIST SP800-53 [141]
Security	Number of Incidents, Number of Port Probes, Successful vs. Unsuccessful Logons, Number of Patches Applied	NISTIR 7564 [158]	NIST SP 800-55 [159]

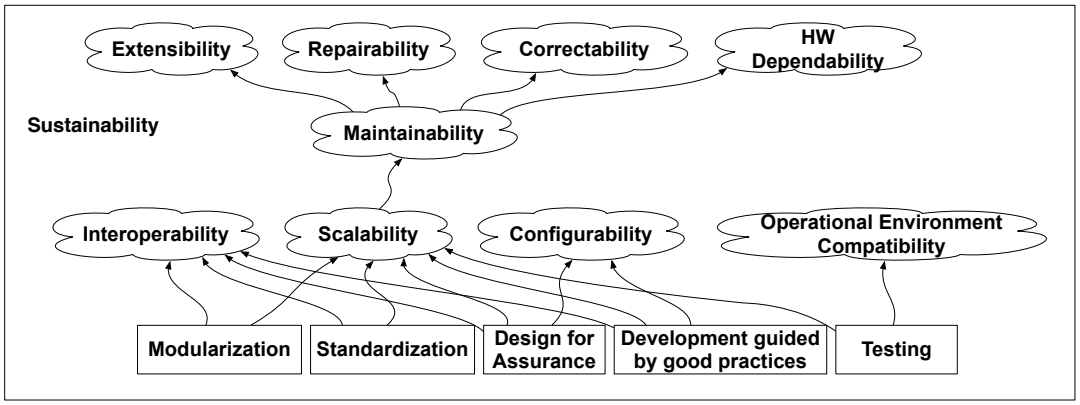


Figure 3.16: Sustainability, softgoals and satisficing techniques

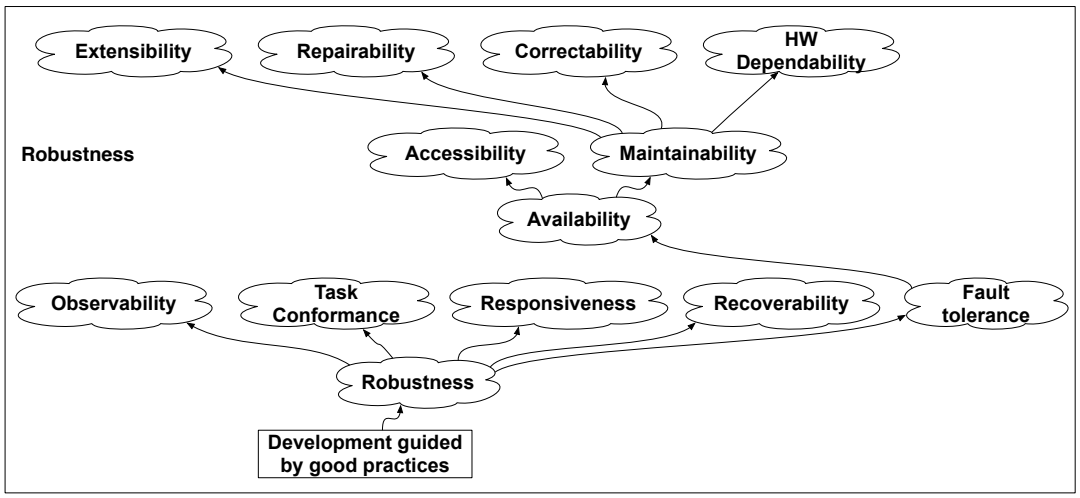


Figure 3.17: Robustness, softgoals and satisficing techniques

given IDS solution. With these measurements, we try to provide quantifiable evidence for several parameters of the system that can help determine whether or not the IDS is sufficiently reliable and available to be deployed within a CPCS. The main reliability and availability metrics we have identified are: *diversity*, *replication*, *uptime*, *downtime*, and *availability*. Here we provide examples of application of these metrics to the protection system for a CPCS:

- *Diversity*: examples are, number of platforms where the IDS can run, and number of communication protocols the IDS can understand.
- *Replication*: examples are, in RAID systems: the level of redundancy, e.g., level 0 to level 7. In general components, the percentage of replication, or the absolute number of replicated instances of a component.
- *Uptime*: examples are, the percentage of time the system is running and number of hours uptime versus number of hours of outage/downtime.
- *Downtime*: an example is the percentage of time the system is down, or the absolute

number of hours the system is down per year.

- *Availability*: examples containing this concept can include the percentage time the system is available in relation to the total amount of time the system has been running. There are other metrics containing this concept related to maintainability, such as the mean time between failures, which are discussed below.

Performance and Responsiveness Metrics

Metrics related to *performance and responsiveness* are also useful for assessing IDS within CCSs. We have selected some of them as examples, describing the ones we think are representative of our scenario: *jitter*, *latency*, *response time*, *schedulability*, *throughput*, *compression ratio*, *speedup*, *service time*, *instruction path length*, *completion time*, *channel capacity*, *performance per watt*, *relative efficiency*, and *bandwidth*. In this section, we provide definitions for those metrics that were not considered in Section 3.5:

- *Compression Ratio*: refers to the reduction of the size of the data, performed by a compression algorithm. It is usually defined as the ratio between the uncompressed size and the compressed size.
- *Speedup*: speedup is the increase of performance (speed) of a parallel algorithm versus its sequential version. In CCSs, it can be useful to measure the performance in the presence of replication mechanisms, where identical instances of a component are running in parallel.
- *Instruction Path Length*: measures the number of machine code instructions required to execute a section of a computer program, providing information about the relative efficiency of a system. It helps assess the complexity versus the efficiency of the IDS modules.

Correctness Metrics

This section discusses examples of the metrics we have identified related to the *correctness* and reliability of a detection or prediction system. These metrics are especially important for the protection mechanisms in CPCSs, since they help measure how good the IDS' detection process is into detecting the threatening events, and not confusing them with harmless system dynamics. For further insight about the evaluation on the correctness of statistical or prediction methods and the metrics that can be extracted from them it is possible to consider standards such as ISO 5725 Series [155], ISO 11843 Series [156], and ISO 2854 [157].

- *Accuracy*: is "*the degree of closeness of measurements of a quantity to that quantity's actual (true) value*" [160]. In IDSs, accuracy has to do with the rate of False Positives (FP) and False Negatives (FN) of the system, i.e., the IDS can only be considered accurate when its rate of FP and FN is minimum or negligible.
- *Precision*: is a characteristic inherent to the measurement system, statistically it is defined as the dispersion of quantitative data, regardless of its accuracy [116]. It is, therefore, the degree to which repeated measurements, taken in unchanged conditions, show the same results.

- *Currency*: in terms of correctness, currency is defined as the degree to which data remain current, i.e., not obsolete. It is also known as the *freshness of the data* [116].

In the same context of correctness, there are two specific metrics that are truly interesting in the case of an IDS: *sensitivity* and *specificity*. They are statistical measures related to both the accuracy and precision of a classification system, and are defined as follows [161]:

- *Sensitivity*: also "*true positive rate*" or "*recall*", measures the proportion of actual positives correctly identified as such. Sensitivity shows how good a test actually is by calculating how often the test will correctly identify a positive.
- *Specificity*: measures the proportion of negatives that are correctly identified as such. This measurement shows how accurate the test is with false positives, and can be considered as the percentage of times a test will correctly identify a negative result.

Maintainability Metrics

This section presents those metrics related to *maintainability*. As we have discussed, those metrics have a strong relationship with the set of metrics in charge of assessing the availability of the system. In this set of metrics we find very specific metrics that are commonly used in ICT systems, and which can be applied directly to CPCSs or CIs: *planned maintenance*, *repair time*, *HW costs*, *SW costs*, *restoration time*, *failures over time*, *maintenance man-hours*, *upgrade events*, *recovery time*, *mean time between failures*, and *mean downtime*.

All these metrics have been previously reviewed in Section 3.5, where different examples of application are also provided. In order to further concretize the set of metrics, it is necessary to provide a real scenario of application and evaluation. In Table 3.15 we can find examples of this set of metrics, as well as the standards and guidelines that can help define the concrete formulae and values for the final metrics.

Dependability and Safety Metrics

Dependability metrics and *safety* metrics are really representative of CPCSs. In this section we present an example of some of these dependability and safety metrics that can be useful to evaluate in CPCSs. In the field of safety, we have found metrics that are mostly based on statistics that provide insight into the probable occurrence of certain events over time. Dependability metrics have a strong link to those of survivability, mostly related to availability, maintainability, etc. For a more comprehensive list of this type of metric, we refer the interested reader to the IEC 61508 Standard [148].

The set of metrics we have selected as an example of this category is: *mean time to unsafe failure*, *reliability*, *stability*, *safety design stability*, *safety requirements traceability*, *safety integrity level*, and *percent system safety hazards*. Here we provide definitions for the new metric examples, which have not been previously examined:

- *Reliability*: as a function of time, or survivor function $R(t)$, is the probability of a system which does not fail in a determined time interval [147]. From $R(t)$ it is possible to compute the reliability of the whole system.

- *Stability*: metrics, such as *Mean Square Stability* (MSS) [162], refer to the equilibrium and stability properties of a system. Systems with high stability will suffer less faults and provide a better service.

Security Metrics

The last set of metrics we have identified are those metrics related to *security*. These metrics are usually strongly linked to the specific context where the security is to be implemented. In our study, they are distributed among the other sets of metrics. According to the NISTIR 7564 [158], the security metrics are based on two aspects: *correctness* and *effectiveness*.

- *Correctness*: usually related to the process of the development of the system, and to the compliance of its operation with the expected behavior. Standardization, quality assurance or development guided by good practices are targeted to improve the correctness of the system, and therefore to improve their security. Examples of these metrics are described above.
- *Effectiveness*: measuring effectiveness is usually based on the security-enforcing mechanisms, determining how well they function and if the system shows signs of vulnerabilities. The aforementioned metrics can help evaluate the effectiveness of the system, and thus help gain insight into the security of the system. Examples of these metrics are: number of incidents, number of port probes, successful vs. unsuccessful logons and number of patches applied.

3.7 Discussion

The main objective of this study is to provide a structured analysis which illustrates the steps to follow to determine the requirements and constraints that challenge the creation of a secure decentralized interoperability architecture for the control systems (especially the CPCSs) of the SG, and the ways to satisfy and assess the fulfillment of these requirements through different sets of techniques and security metrics. To this end, in this chapter, we devise a procedure to follow in order to deeply understand the non-functional requirements of a complex system. Once this procedure is shaped, we make use of it to analyze the characteristics and needs of a secure interoperability platform for CCSs, and re-apply it to understand the features and requirements any component added to this platform has to provide in order to be compatible with this environment.

The analytical procedure we follow is based on the guidelines of two main methodologies, the NFR Framework [119] and the GQM approach [129]. Based on these two methods, we divide the process of analysis of the system into three stages: *the analysis of requirements*, *the identification of satisficing techniques* and *the identification of representative metrics*. We apply this strategy to understand the characteristics and needs of a secure interoperability architecture for CCSs.

The first stage has been developed in Sections 3.1.1 and 3.1.2, where the CCS requirements and special constraints are identified in an iterative way. In Section 3.3 this analysis has been

extended in order to discern the requirements a secure interoperability architecture would have to comply with in order to provide adequate and robust services for the interconnection and interoperability of critical control systems, especially considering the CPCSs of the SG. The requirements have been analyzed following the NFR Framework guidelines [119].

These NFRs, in contrast to functional requirements, represent high-level characteristics and information about the system under study. In SW engineering, the capture and definition of requirements is an iterative process that is refined in each cycle. In our work, we have performed several iterations to determine the NFRs that are most suitable for the secure interoperability architecture of generic CSs (see Figures 3.3 and 3.4), stopping our analysis at this point. If this process is continued, given a concrete set of interconnected CIs and the needed instances of the different security tools to evaluate, a further cycle of refinement focused on this concrete scenario would provide the final NFRs that define the specific needs of this particular interoperability infrastructure.

Our analysis, however, stops before this last cycle of specialization, remaining purely theoretical, since we would like for our study to remain as open and versatile as possible to be applied to the different types of CIs in existence. Once the NFRs have been refined according to a given concrete scenario, a validation process should be done to determine the completeness and validity of the NFRs selected. However, as previously stated, due to their abstract nature, NFRs are difficult to define and test, therefore, they are usually evaluated subjectively [119].

Validation of some NFRs can be done through theoretical approaches, e.g., network security requirements can be evaluated from a game-theory point of view, in the form of a game between attacker and defenders using the Nash Equilibria [163]. Or through an expert group interview process (Delphi method [149], checklists [164]) to determine the adequacy of the selected NFRs for the problem at hand, and to have access to feedback about the completeness of the requirement set.

Since we consider that the final refinement cycle of NFRs should be done taking into account the real characteristics of a CI, we provide our study as a guideline for future reference when studying and analyzing the requirements and constraints of concrete critical scenarios. Thus the final refinement cycle of the NFRs and, in consequence, the validation process of the set of requirements selected remain as future work.

The second stage of our study provides insights into the way to comply with the identified NFRs (see Section 3.4). We have outlined the *satisficing techniques* proposed to achieve and satisfice the softgoals identified in the first stage of the study. We have therefore described sets of techniques that help achieve certain goals for the secure interoperability infrastructure at different levels: techniques allowing the protection of the critical control systems, techniques that provide architectural robustness and integrity to the interoperability platform and the communication infrastructure, techniques that enable the interoperability itself, techniques that help the integration of industrial IoT nodes within the infrastructure, and techniques related to the virtualization of services within the platform.

In an effort to concretize the results of a study based on abstract characteristics (NFRs), we use the GQM approach [129] to identify sets of metrics that could help analyze from a quantitative point of view the selected set of requirements. The metrics we have identified for our study are merely examples of important quantitative information that can be used to evaluate a critical

system, and any component that we want to add to this system, especially to ensure that the newly added sub-system will not introduce new risks and threats into the CI. This part of our analysis is developed in Section 3.5, where we present an abstraction model of the secure interoperability system, where the different metrics can be applied according to their scope and area of application within the interoperability platform (see Figure 3.11).

We have separated our metrics into ten large sets of metrics for the sake of clarity, however, they can overlap. On the other hand, these five large sets have been characterized from a high abstraction level due to the scope of this research, which ends in a refinement cycle before the concretization of the scenario. In practice, it is necessary to formalize the exercise by considering, for example, the templates offered by ISO 27004 [165] or NIST SP 800-55 [159]. Both standards provide sufficient guidance to compute, through more tangible calculation, the quantitative level of a study resulting in concrete sets of well defined metrics. In this way, the refinement of the metrics offers the means to develop and use assessment measures to determine the effectiveness of an information system and its controls.

It is important to note, that any metric we want to apply to a given system (e.g., CIs, the cloud, the IoT) has to be validated in order to learn whether these metrics are suitable for this system's evaluation, i.e., if it makes sense to use these measurements in this scenario, and whether or not they are representative for what we want to validate. However, this part of the study needs the refined final set of NFRs that takes into account the actual CIs for interoperability and the concrete interoperability solutions that will be implemented, which compose the final scenario of application. In our study, we have not included the validation stage of our proposed metrics.

However, for each of the metrics, we have provide a high-level definition, indicated their scope and the requirements they aim to assess, i.e., whether or not the matched requirements would have been fulfilled in the concrete secure interoperability scenario. And in a later step, we have provided a set of standards and related guidelines that would serve as controls or representative references in order to provide useful information about the application, representativeness and evaluation of these metrics in a specific context of application.

As previously discussed, it is possible to tackle the validation of the metrics using two methods: through a formal process of validation, or through an expert group interview process, such as the Delphi method [149]. In such a complex scenario, where numerous actors and interdependencies are in place, we consider the most viable method of validation for these metrics is through the expert group interview process, a task that additionally can result in very valuable feedback for the world of academia and also for the CI's management. A supplementary study which could provide additional information in the validation process consists in testing the selected metrics within other non-critical environments.

A particular case of application of our three-stage methodology of analysis, is the application of this study to the determination of whether a protection solution module (e.g., an IDS solution) is suitable for deployment within this new industrial environment (Industry 4.0) and the new generation of CCSs accompanying it (see Section 1.1), as described in the case of study in Section 3.6. As we previously discussed, we focus our analysis on IDS solutions, however, with little modification, it is possible to extend this work in order to determine whether or not any given security component or process is suitable for its deployment within a CI through this same three-stage analysis or requirements, techniques and metrics.

We follow our methodology and perform iterations of refinement to identify and analyze the requirements, techniques and lastly metrics that help evaluate the suitability and characteristics of detection and protection solutions deployed within the CS of a CI. It is also possible to understand this study as a particularization and further iterations of study of protection solutions deployed within our secure interoperability architecture scenario.

As it has been discussed previously, we have performed several iterations to determine the NFRs that are most suitable for a generic IDS for CIs (see Figure 3.12), stopping our analysis at this point. If this process is continued, given a concrete CI (e.g., smart grid) and an instance of the IDS tool to evaluate, a further cycle of refinement focused on this concrete scenario would provide the final NFRs that define the needs of this particular infrastructure and IDS.

The results of this experimentation would provide additional insight to the experts when reviewing the metrics for the given IDS within a CI. This information would provide the experts an overview of, for example, the performance of the IDS in general networks, indicating whether the IDS solution is a priori too costly for a critical setting or not. However, these experiments are difficult to set, and of course, removing the IDS from a critical scenario would only be relatively useful in terms of evaluating the metrics. This is because the requirements of the CI could constrain the application of the IDS too much, which would perform perfectly in a general environment. Thus, due to the need to refine and validate NFRs according to a determined scenario, and the need of concretize the metrics as previous steps, this last stage of the study also remains as future work.

Therefore, our analysis stops at a theoretical point, where the actual scenario (concrete CIs and IDS solutions) have not yet been defined. This implies that our study is applicable to different settings, but also that further refinement of the NFRs and metrics is needed to take into account the particularities of each scenario. Moreover, since the refinement cycles are not finished at this stage, the necessary validation procedures of NFRs and metrics have not been performed in our study, they remain as future work. Validation of NFRs and metrics are difficult, we believe a valid strategy would be the expert group interview processes, such as the Delphi method, providing experts with the sufficient insight into the concrete scenario through simplified simulations. From our point of view, our three-stage analysis and the outcome of this study as a model would help sustain the improvement of security and interoperability of CIs by ensuring the introduction of efficient and compatible components within the CCSs.

Chapter 4

Services for the Secure Interoperability in CPCS

This chapter is devoted to the study and analysis of the diverse security services and procedures necessary for the correct operation of CIs transitioning to the Industry 4.0 paradigm as described in Section 1.1. These security services constitute a key characteristic of the fourth generation of CSs, and ensure the correct operation of the systems, the interoperability components and avoid that threatening events occurring within this infrastructure cascade through the different interrelated components of the system. In this chapter we provide an overview of the security services for general CIs' control systems, and later on, we focus our analysis on the scenario of the SG, as an example of critical infrastructures transitioning to the Industry 4.0 paradigm and including CPCSs in their control infrastructures (e.g. smart meters, WSNs, etc).

We divide the services for the secure interoperability of CPCSs in particular, and control systems for CIs in general, into four different components (see Figure 4.1): the *Basic Security Services*, the *Prevention and Detection* services, the *Awareness and Reaction* services, and the *Restoration* services. These different modules or sets of services help introduce secure interoperability for the CPCSs of CIs; in our analysis we particularize the CI's scenario focusing on the smart grid infrastructures (as decided in Chapter 1).

The first module or set refers to the basic security services that must be present at each level of the interoperability infrastructure in order for it to be secure. The second module corresponds to the different prevention and detection services that should be deployed in order to monitor and control the status of the secure interoperability architecture in order to avoid and early detect any threatening effect within the critical systems.

The next module contains those security services focused on implementing awareness and reaction mechanisms within the secure interoperability infrastructure, their aim is to continuously analyze the status of the interoperability architecture and each of the CPCSs, in order to provide adequate, proportional and effective protection and mitigation responses to the threatening events occurring within the critical system. The last module or set of services contains those restoration mechanisms that any critical system should implement, in order to make the secure interoperability infrastructure capable of returning (restoring) to its normal operation after an error or failure within the critical infrastructure. Figure 4.1 provides an abstraction of the

different sets of services mentioned, which will be discussed in detail in the remainder of this chapter.

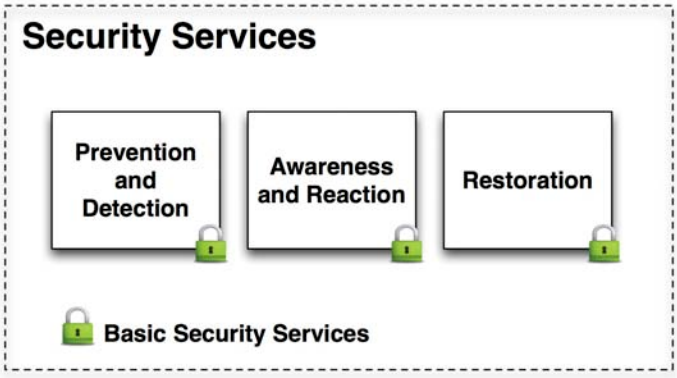


Figure 4.1: Security services

4.1 Basic Security Services

In the context of smart grid and the CPCS networks, there are three main basic security services (see Figure 4.1) that allow the standard and secure interoperation of these critical systems: *Authentication*, *Authorization* and *Accountability* mechanisms. According to NIST, authentication is defined as "*the process of verifying the claimed identity*", and can be applied to control access to systems and networks [112]. Authorization is the process of granting the user access privileges, determined by policy rules applied to the authenticated identity and other relevant information, which can be enforced by access control mechanisms [112].

According to NISTIR 7628 [110], periodic audits and logging procedures of the SG CPCSs need to be implemented for accountability purposes. These mechanisms help validate the security processes deployed within the critical systems and determine their operation condition. The implementation of accountability services imply the application of security controls to ensure the compliance of the systems with the established security policy procedures. The accountability service must be protected by the authentication and authorization services [11].

Standards such as the IEC/TS 62351 [134], the NIST SP800-82 [112], the NISTIR 7628 [110], the NIST SP1108 [6] and the IEEE2030 [11] among others, gather the need for the critical systems, specially the ones belonging to the SG, to implement these three basic security services. These will reduce the risk of malicious or accidental cybersecurity events while, at the same time, allow access to the system to the relevant stakeholders [6].

These basic security services can be performed either in a distributed or centralized approach. If a distributed paradigm is used, every system performs these processes on their own, being responsible for storing the credentials (user accounts, roles, etc.), performing the identification and authentication of the user, and maintaining logging information of the main system events. The decentralized approach does not usually require additional infrastructure, however, it presents

several problems regarding the scaling capability when the size of the system increases, and the realization of credentials maintenance tasks [112].

Alternatively, the centralized implementation for the basic security services procedures is widely used, since it can be easily scaled and used to manage large amount of users and accounts. A centralized approach uses central authentication systems and protocols (e.g., *Lightweight Directory Access Protocol* (LDAP), Kerberos, etc.) to perform the authentication of the users [112]. For authorization purposes, in both the distributed and centralized approaches, it is possible to use role-based access control mechanisms, such as IEC/TS 62351-8 (RBAC) [28], or ABAC [96].

However, besides the improved scalability provided by the centralized approach, numerous concerns arise concerning the impact of a centralized system in a critical environment. The main concerns and considerations are the following ones [112]:

- The centralized authentication system must be highly secured, since it becomes a highly critical solution, and an interesting entry point for cyber attacks (see Chapter 2 for further information).
- The centralized authentication and authorization system has a high availability requirement, since its failure may prevent administrators from authenticating to a system during an emergency. Additional protection and robustness techniques (e.g., redundancy) need to be implemented (see Chapter 3 for extended information).
- The networks that are used to perform the authentication and authentication tasks must be reliable and secure to ensure authentication attempts are not hindered.

We therefore propose that a hybrid infrastructure that combines the characteristics of the centralized and decentralized security services would allow the proper application and use of the basic security services, leveraging the good qualities of both approaches.

4.2 Prevention and Detection

Critical infrastructures around the globe provide the most necessary services to society, so their continuous correct operation is of paramount importance. Control systems, such as the CPCSs, perform the management and the regulation of behavior of the internal devices and systems of these infrastructures. They are considered a fundamental component within CIs, having an impact in the overall performance of other interconnected critical infrastructures [1]. Thus, the protection of CIs and their control infrastructures is currently seen as an essential part of national security in numerous countries around the world [150].

Recent reports, as reviewed in Chapter 1, show that security incidents and cyber-attacks against control systems are increasing, and they are getting more aggressive and sophisticated. For this reason, and as dictated by governments and institutions around the globe, the integrity and availability of all these critical systems have to be protected against the numerous threats they face every day [150, 112]. Approaches for CIP arise from several perspectives: preparedness and prevention, detection and response, mitigation and recovery, international cooperation, etc. [42, 112, 15, 16]. As a tool to respond to this need for protection, intrusion detection has been

at the center of intense research in the last decade, due to the rapid increase of cyber-attacks on computer systems.

Intrusion detection refers to a variety of techniques for detecting threats in the form of system faults (anomalies) or malicious and unauthorized activities. A technique that focalizes this detection effort is the intrusion detection system [109]. IDS solutions have been proposed for multiple environments, and they could result in very valuable protection tools for CS environments. However, their application to the protection of critical systems must comply with the strict constraints of CSs [21]. Systems such as the one presented by N. Goldenberg and A. Wool [166] or the one proposed by H. Hadeli et al. [167] are examples of IDSs especially designed to be deployed in a critical environment.

However, when intrusive behavior is detected by the IDS in a critical scenario such as CPCSs, it is desirable to take evasive and/or corrective response actions to prevent these attacks from succeeding, and ensure the safety of the computing environment [15, 4]; such countermeasures are referred to as intrusion response. Incidentally, as threats become more abundant and sophisticated, and given the special characteristics of CIs, apart from detection mechanisms, new and more powerful solutions have to be deployed in order to safeguard them and to avoid faults and consequent cascading effects. To fight this domino effect, besides providing efficient detection mechanisms, we need to focus on the response, mitigation and recovery needs of CIs.

Solutions that can provide these functionalities are the intrusion prevention systems, also called *Intrusion Response Systems* (IRS) [4], IPRS, and *Intrusion Detection, Prevention and Response Systems* (IDPRS) [168]. An IPS/IRS/IPRS/IDPRS is "*software that has all the capabilities of an intrusion detection system and can also attempt to stop possible incidents*" [109]. In the remainder of the text, we will refer to these systems as IDPRS, since we will use the term *Response System* (RS) to denominate a specific element of the whole system. The IDPRS is often integrated as an extension of the IDS, but it usually receives less attention than the IDS research due to the intrinsic complexity of developing the mechanisms to offer an automated and correct response to certain events.

Traditionally, and particularly in CPCSs, the response to a threat was manually triggered by the system's human administrator, and required a high degree of expertise. However, the increasing complexity and speed of the cyber-attacks in recent years, and the intricate possible ramifications of a system's faults show the acute need for complex intelligent dynamic RS [4]. Therefore it has become necessary to use sophisticated advanced techniques from autonomic computing, machine learning, artificial intelligence and data mining to build intelligent and smart IDPRS. Together with the deployment of IDS solutions in these contexts, automatic and intelligent response mechanisms have to be put in place to help protect CIs and prevent cascading failures to other interdependent infrastructures [169].

4.2.1 Monitoring and Detection

Within the prevention and detection set of security services represented in Figure 4.1, we find that the main tool of mechanism used to provide these services is the IDS, the monitoring and detection component of the IDPRS. IDSs have several possible mechanisms to provide detection, which can influence the efficiency and applicability of the IDPRS' reaction component; thus, it is important to note the technique used for detection. According to the implementation of the

IDS' engine, there are three main methods of detection: *anomaly-based detection*, *signature-based detection* and *specifications-based detection*:

- *Anomaly-based detection*: the IDS compares definitions of activity that is considered normal against observed events in order to identify significant deviations [170]. This method has the advantage of being very effective in identifying previously unknown threats. Its main drawbacks are the generation of a large rate of false positives in dynamic environments, and the difficulty that arises when analyzing the causes of a given alert [109].

- *Signature-based detection*: according to NIST, a signature is "*a pattern that corresponds to a known threat*" [109]. A signature-based IDS analyzes the information it gathers from the system under surveillance and compares it to signatures of known threats in order to identify possible incidents [170]. Using this method, the system looks for already identified and known attacks, thus this solution is very effective in detecting known threats, but rather ineffective in detecting previously unknown threats, threats disguised by the use of evasion techniques, and many variants of known threats.

- *Specifications-based detection*: similarly to anomaly detection, specification-based systems detect attacks as deviations from normal behavior. However, these approaches contain specifications of the system under surveillance (usually manually developed models) that capture legitimate behavior, instead of using previously seen behaviors as in the case of anomaly detection. To develop the models, legitimate systems' behaviors can be extracted from security policies and protocol specifications [152]. According to NIST [109], there are three main problems that arise when using stateful protocol analysis. First, the reliance on vendor-developed universal profiles that specify the use of particular protocols ties the IDS to a specific environment, since the implementation of the protocols may differ from system to system. Second, this kind of system is highly resource-intensive because of the complexity of the analysis and the overheads caused by the state tracking. And lastly, these IDSs cannot detect attacks that do not violate the characteristics of acceptable protocol behavior. In general terms, the type of detection engine used in the IDS depends on the needs and characteristics of the surveilled systems, where their selection is guided by parameters such as precision (low false alarm rate) or efficiency [170].

In a critical context, the parameters studied to select the adequacy of a given IDS for a constrained environment are different from the general networks' IDS solutions. Here, characteristics such as precision vary from the desire to obtain low false alarm rates (low rate of false positives) to the need to achieve a low rate of false negatives (the true attacks are not missed by the detection engine). It is also necessary to observe the techniques implemented (to avoid resource-intensive algorithms) and the level of automation achieved by the IDS for CIP [169]. Therefore, the selection of the detection engine must be aligned with the needs and constraints of the critical system under surveillance, and the type of environment where the system has to perform its tasks.

In general, taking into account their aforementioned characteristics, specifications-based detection as well as signature-based detection will work better for environments where the dynamics and behaviors are well-known [152]. Contrarily, anomaly-based detection can be recommended when the IDS is placed in an environment that constantly faces unknown behaviors and dynamics [170]. In the literature, we find different types of IDS specifically designed for CIP [152, 171, 172, 173], using and adapting different types of detection engines.

4.2.2 Intelligence and Learning Techniques

Each of the aforementioned detection engines might implement different degrees or adaptability, these capabilities are introduced by the selected *Machine Learning*, *Data Mining* or *Statistical* techniques. Machine learning and data mining are sometimes considered as synonyms, since the methods and techniques used overlap significantly. Machine learning focuses on learning from data, i.e. it compiles all the disciplines that design and construct automatic systems capable of learning from examples. Data mining is defined as the "*extraction of implicit, previously unknown, and potentially useful information from data*" in a automatic or semi-automatic way [174]. The main distinction, which makes some authors consider data mining as a sub-field of machine learning, is that machine learning is based on the known properties of the training data (prediction), and data mining focuses on leveraging unknown properties and patterns in the data (discovery). In this study we will refer to these two concepts at the same level of importance, since it is necessary to apply both approaches (discovery and prediction) in order to build a secure automatic detection and prevention system for CSs, and we will use machine learning to denominate all of them.

Learning techniques are varied, and they originate from very different fields of knowledge, such as optimization, statistics, logic, etc. It is interesting to study these methods taking into account characteristics like their knowledge scheme and the level of supervision of the training, since the constraints they impose to the underlying system impacts the possibility of introducing them in a critical context (for further information on these constraints see Chapter 3). In our analysis, we identify three different ways to classify the intelligence techniques for the detection engines, i.e., considering the *Knowledge Scheme*, according to the degree of *Supervision* applied to the technique, and by the field of knowledge from which the technique comes.

The knowledge scheme indicates the level of knowledge that is feed to the system prior to the training. If we take this scheme into account, it is possible to divide the learning techniques into two different sets [170]:

- *Prior knowledge-based systems*: these systems, in order to function, are fed with the knowledge and experience of an expert. For example, this knowledge can be transferred to the system by designing and writing a set of rules.
- *Prior knowledge free systems*: this kind of system is based on the knowledge extracted through an automatic (or semi-automatic) procedure of training. The advantage of this method is that it is not necessary to have any knowledge of the system in advance.

It is also interesting to consider those solutions that can add the knowledge of an expert to the model of the system obtained through training. We can denominate these solutions *hybrid knowledge-based systems*, and we theorize that they would result of extreme interest in the context of intrusion detection within critical systems, since their performance would be higher in these complex scenarios.

The learning of a system consists on generalizing behaviors from unstructured data, and inducting knowledge from examples. It is possible to differentiate two main different schemes of learning according to the level of *supervision* (by a human operator) involved in the learning process: *supervised and unsupervised learning* [175].

- *Supervised learning* implies providing the system with certain knowledge about the vari-

ables that it is desired that the automatic system learns, i.e. the operator provides examples (labeled training data) of anomalous and normal behaviors in order to "teach" the learner the model of the behavior of the system.

- *Unsupervised learning* implies that no knowledge is provided to the system when training it; the algorithms implied in the learning process handle the data and learn on their own. Pure unsupervised learning is difficult to achieve in real life contexts, and the accuracy of these methods are low. There is usually a step of supervised preprocessing of the data before the training, and a step of supervised parameter tuning after the training, to adjust the models obtained.

There are other schemes and procedures that the algorithms follow, for example the *semi-supervised learning* approach, that combine labeled and unlabeled training data. However, we will only focus in the main schemes of supervision: supervised and unsupervised learning, to study which of them is better suited in the context of CPCS monitoring.

Taking into account the field of knowledge from which the different techniques for detection engines originate, we can differentiate three main categories, i.e., *Data Mining-based* monitoring, *Statistical and Statistical Learning-based* monitoring, and *other* mixed techniques for monitoring.

Data Mining-based Monitoring

- *Classification-based* techniques: these correspond to classifiers in charge of assigning data instances to (normal or anomalous) classes [176]. Within this category, the *decision trees* (e.g., ID3, C4.5) are the most representative structures which deal with mapping observations into conclusions using hierarchical rules under the assumption of *divide and conquer*. This assumption consists of recursively breaking down a problem into sub-problems until these become atomic units. There are two types of decision trees: *classification* and *regression* trees, the results of which depend on the type of data managed and the desired outputs of the models. These tree-like structures are capable of providing fast computations and decisions since each data instance (in the testing phase) is compared against a precomputed model. Their advantages are the speed of classification and the comprehensibility degree of their outputs to humans. Nonetheless, their shortcomings are the tolerance to redundant or highly interdependent data, as well as, their reliance on predefined models primarily based on labels [177, 178].
- *Association Rule Learning-based* techniques: unsupervised schemes which try to identify the relationships between categorical variables, using strong rules and thresholds to prune. As part of this classification, we highlight the *Apriori* algorithm and the *FP-growth* algorithm. The former is an influential algorithm for mining frequent patterns, which tries to find rules in large datasets to predict the occurrence of an item based on the occurrences of others. In fact, its main property is: "any subset of a frequent pattern must be frequent"; and its pruning principle is "if there is a pattern which is infrequent, its superset should not be generated". Similarly, FP-growth has the same goals, but uses a compact frequent-pattern tree (FP-tree) structure under the assumption of *divide-and-conquer*. This assumption consists of finding frequent rules/patterns to decompose mining tasks into smaller ones, the aim of which is to recursively delete all the data items that

are not frequent; instead of generating candidates for each study. As mentioned, the technique itself has to make use of pruning approaches to reduce the sets of rules. Hence, the effectiveness of the learning process depends heavily on the parameters that configure the pruning operations and their algorithms, and on the number of rules that have to be launched, where the processing time may increase exponentially regarding the number of attributes [178]. Nonetheless, the comprehensibility of the results is an advantage.

- *Clustering-based* techniques: these aim to classify data instances in clusters through an unsupervised or semi-supervised method; i.e., no knowledge of threats, attacks or anomalies are needed in advance during training. This feature helps the testing phase process the evidence quickly, where the unsupervised models only compare the instances with a small number of clusters. To do this, the technique needs an evaluation function (e.g., a distance function, density, etc.) to compute the distances between data points, where each instance is evaluated according to its entire cluster. Although there are several clustering algorithms (e.g., hierarchical, centroid-based, distribution-based, density-based, etc.), the most popular approach is the *k-means*. Clustering-based techniques are quite dependent on the algorithm's parameters, which consequently have associated computational costs, which are mainly influenced by the type of dataset and the parameters selected [176, 179]. Most approaches follow a quadratic order, except those based on heuristics (e.g., k-means), which take a linear complexity. In addition, the tolerance of the algorithms to different constraints in the data are quite dependent on the configuration of the parameters selected [176].

- *Rule-based* techniques: these focus on learning rules that interpret the normal behavior of the system with the capability of multi-class and one-class settings. Their main strengths are the accuracy, comprehensibility, handling of simple parameters and low complexity. In contrast, they are weak in incremental learning, dependence on expert knowledge, tolerance to noise and are unsuitable for anomaly detection. Within this class, we highlight the *rule learners* (e.g., Ripper). These algorithms use rules from trained data to construct a rule-based decision engine under the assumption *divide-and-conquer* by looking at one class at a time and producing rules that match the class. This procedure, apparently simple, requires exploring the whole dataset where their learners become slow and inaccurate with low tolerance to missing irrelevant and redundant data. Nonetheless, they provide speedy classifiers with comprehensible results, and allow easiness to manage system parameters.

Statistical and Statistical Learning-based Monitoring

This class defines those statistical techniques that compute statistical models to apply interference tests so as to verify whether or not a specific instance belongs to a statistical model. Within these techniques, it is possible to find:

- *Parametric and Nonparametric-based* methods: these refer to inference engines with a strong dependence on the data observed and which are composed of well-known statistical models, such as Gaussian or histograms [176, 179]. These statistical models are in general accurate and tolerant to noise and missing values. Additionally, the statistical analysis provides additional information to the detection systems, such as the confidence interval associated with the anomaly. However, depending on the dataset, these methods can be

sensitive to subtle changes and the output results are difficult for humans to understand. Moreover, depending on the dynamics of the problem, the efficiency of the model can be reduced, and in some cases, these techniques can potentially have quadratic complexity if dealing with large databases [176]. In contrast, the chief disadvantage here is that these techniques assume that the production of the data follows a particular distribution, which in real life scenarios is not true [176], consequently there are difficulties in determining the best distribution to fix such data. This category also includes the *operational models*, the observations of which are evaluated according to counters, bounded by predefined (upper and lower) thresholds. If these boundaries are not efficiently computed, the approach itself can then hamper the dynamic detection of anomalous events. In general, operational models may not be suitable for those dynamic scenarios that regularly change their normal behavior [178].

- *Time Series-based* techniques: these, can be both non-parametric and parametric [180], basically aim to provide behavior forecasting using times series, which are sequences of data points, measured at successive and uniformly distributed time intervals. These methods are generally suitable for detecting those threats launched in series form with subtle perturbations (e.g., stealth attacks), but its effectiveness decays when there are drastic changes [178]. There are several methods of time series analysis; one of the most useful for detection are the *smoothing techniques*, which provide weighted data instances. The smoothing mechanisms provide accurate observations and their approaches are tolerant to insignificant changes and missing values, in addition they help optimize parameters. Unfortunately, as in the case of the rest of the statistical methods, they tend to be difficult to understand for humans and have great difficulty handling parameters. The smoothing techniques also produce weak models for medium or long-range forecasting, which heavily rely on past history and on the smoothing factor to predict the future; the variant *exponential smoothing models*, in particular, cannot easily forecast future events in the presence of fluctuations in recent data [181].

- *Markov Models*: are mathematical representations with quantitative values that help predict the future behavior of a system according to the current evidence. There are many types of Markov models, and all them have functionalities and features in common, such as operations based on successive data and dependence on a state transition (probabilistic) matrix to illustrate activity transactions without having knowledge of the problem in hand. However, and unfortunately, the Markov models are highly complicated when addressing complex situations with multiple dimensions, the complexity of which increases when leaving the most simple (first order) Markov chains, in favor of more precise and complicated models [182] (e.g., the *Hidden Markov Models* (HMMs)). In addition, abrupt changes in the normal activity sequence within a system becomes unmanageable, so that this feature may become undesirable in critical contexts [178].

- *Bayesian Networks* (BNs): these networks are composed of directed acyclic graphs, where the nodes represent states that have associated probabilities, and parameters encoded in tables. The BN first learns from structures of the (either unknown or known) networks, and then computes the parameters of the model. This category can be well-applied in intrusion detection models as powerful and versatile solutions, but may become computationally complex if the networks are unknown a priori [177], or present too many features (large BNs). Despite its ideal accuracy, this technique is too expensive in terms of time

and storage, and tends to be infeasible for constrained scenarios. An extension of BNs are the *Naïve Bayes networks*, where their digraphs only hold one parent for each node and the probabilistic parameters of the network are calculated using conditional probabilities. These types of probabilities in the form of a product can be transformed into a sum through the use of logarithms, allowing the decision system to be computationally efficient and fast [177]. Other benefits, given the simplification of the model, are the diminished computational overhead for training, understandability of their networks, and the possibilities for handling parameters and introducing incremental learning. However, a disadvantage of this model is that it is not as accurate as a BN due to the existing independence between the child nodes, which imposes strong constraints on its behavior [177].

- *Instance-based Learners*: are lazy-learners based on the similarity of properties among instances of the same class [177], such as the *K-Nearest neighbor* (KNN). These approaches are characterized by their speedy learning with respect to the number of attributes and the number of instances present in the dataset. In addition, they are suitable for incremental learning and their parameters can be modified with fairly easily. Despite these benefits, these approaches are quite sensitive to the selection of the similarity function [179], do not provide a deterministic way of choosing the parameter k, and require storage. The size of the instance sets are also dependent on k whose value affects the time required to classify an instance; in addition to exhibiting an extensive testing phase in which their methods can reach a low tolerance to noise and a low stability depending on the parameters adjusted.

Other Monitoring Techniques

- *Artificial Neural Networks* (ANNs): these, in the artificial intelligence field, can be applied for anomaly detection using a multi-class or one-class configuration for training and learning. The models essentially consist of the computation of the sum of weighted inputs to produce weighted outputs [177]. Thus, the performance of ANNs depends on three main aspects: input and activation functions, network architecture and the weight of each connection. ANNs are generally accurate and fast classifiers, capable of tolerating highly interdependent data, whose learners can need of back propagation algorithms where the output models may not be comprehensible to humans and produce over-fitted models. These drawbacks make it difficult to ensure real-time in the operational processes since most ANN approaches need extra processing-time.
- *Support Vector Machines* (SVMs): this method is a supervised learning model based on a non-probabilistic binary linear classifier under a one-class configuration to recognize data patterns or outliers in datasets [183]. Given that SVMs work with linear combination of (data) points, the computational cost follows a quadratic order and the number of vectors selected is usually small. Thus, the complexity of an SVM is not affected by the number of features in the training data so SVMs are suitable for addressing large numbers of features. The main weaknesses found is that most real-world problems involve inseparable data for which no hyperplane exists that successfully separates the positive from negative instances in the training set; and in optimization problems, the presence of local minimums and maximums affects the accuracy and speed. Even so, SVMs are, in general terms, accurate and fast classifiers, and tolerant to irrelevant and redundant data. However, the method

itself usually presents problems with the speed of learning, the comprehensibility and the ability to handle the model and incrementally learn [177].

- *Fuzzy Logic* and *Genetic Algorithms*: Fuzzy logic consists of simple rule-based structures that define reasoning [178]. The approaches are in general simple, flexible and fast in the processing of rules and in the determination of anomalies, in which their approaches are able to establish the normality boundaries and manage large databases. The technique is also able to model complex systems and situations without requiring precision or complete databases; however, its conclusions may not reflect the confidence degree of a problem. Regarding genetic algorithms, these deal with optimization and search heuristics where their implementations can require a large number of iterations to reduce a problem, and according to a fitness function. This also means that the detection rate depends on the accuracy of this function, where the approach itself has proven be unable to detect unknown or new threats, as well as, multi-interactive attacks [184].

- *Knowledge Detection-based* techniques consist of progressively acquiring knowledge about specific attacks or vulnerabilities, guaranteeing accuracy of the technique with a low false positive rate, and flexibility and scalability for adding new knowledge. The result is a system potentially capable of ensuring resilience against threats, but this security also depends on the update frequency of this knowledge and the degree of granularity to specify the threat patterns. According to M. Gyanchandani et al. in [178], there are a few types of knowledge detection-based approaches, such as state transition, expert systems and Petri nets. *State transactions* aim to define threat models through state transaction diagrams illustrating the activity sequences and operandi mode; similarly, *Petri nets* represents state transactions using directed bipartite graphs to show events and conditions. Conversely, *expert systems* are composed of intelligence engines based on simple rules which define different models capable of reasoning about the provided knowledge like a human expert. Expert system models can be provided with varied knowledge; e.g., different types of threats or vulnerabilities, or even conditions given by the security policies.

- *Information* and *Spectral Theory*. Both theories are based on statistical approaches. Particularly, the information-based techniques focus on analyzing the data itself and its order to observe whether there are irregularities (related to meaning, features or properties) within it [176]. Through concepts of entropy, their approaches are in general efficient, but this feature depends on the size of the dataset to be compared; and they are also tolerant to insignificant changes in the data and redundancy [185]. As for spectral theory methods, these work with approximations of the data (or signals) to observe whether there are differences, more visible in other dimensions of the data. Spectral analysis is an approach fairly linked to time series analysis and the characteristics of the communication channels. It performs dimensionality reduction to handle high dimensional data; however, its efficiency varies according to the mathematical method used to translate the model into other dimensions, e.g., Fourier, and these techniques usually have a high computational complexity [176].

Artificial intelligence and machine learning are vast fields of knowledge, and there are many other methods available than the explained ones. Some of them have different properties and they are classified in a broader field known as *computational intelligence*, but we have restricted this study to the methods and techniques better suited for the constrained scenario of the intrusion

detection within CIs. However, in this critical context, the need for automation cannot overcome the need for accurate methods with good performance. Studies, such as the one performed by Sadoddin and Ghorbani [186], defend that supervised machine learning techniques have better general performance for real-life problems than the unsupervised ones. Considering this, we develop a more detailed study centered in the advantages and disadvantages of supervised machine learning techniques.

In the view of the information provided in this section, it is possible to discern methods that are better suited for intrusion detection in critical scenarios. Taking into account parameters such as the accuracy of the systems and the general performance of the models, we understand that lazy learners and Bayesian networks are difficult to implement successfully in a constrained environment. Whenever it is necessary to evaluate the correctness of a model, or it is necessary to add a certain degree of expert knowledge, ANNs and SVM are the more restrictive methods. Thus, we understand that the methods that are better suited for detection in CSs are the ones based on logics, such as the decision trees and the rule learners. Decision trees have well-balanced characteristics for this specific context, while rule learners have several drawbacks (e.g. accuracy) that can be easily overcome (e.g. implementing boosting algorithms), but they provide several capabilities (performance, comprehensibility, and easiness for introducing rules by experts), that are vital and really interesting in this critical context.

4.2.3 Automation

Automation is defined as the introduction of automatic equipment or processes within a system, to assist or replace human operators, mostly when the tasks involved are intensive in computations or the working conditions are extreme. In the CIP context, it is important to emphasize that CIs are composed of subsystems that are usually deployed in distant and isolated locations, where the automation of the tasks is of paramount importance. The objective of introducing machine learning methods in the scenario of intrusion detection for CIP is complying with the proven need [3] [187] of making certain processes automatic, and therefore assisting the human operators in these complex tasks. The idea is that systems based on these automatic methods will be capable of performing automatically, and will serve as powerful tools of *reaction*, providing methods of *prevention of cascading failures*, other than only detection of anomalies and intrusions.

In this section, we have explored the needs of detection automation in a critical scenario, and we have divided them into levels of automation. With the definition of these levels, we have designed a methodology to determine the degree of automation of a protection system, that it will also serve as a guide for designing future automatic detection and protection solutions that comply with the needs of automation of CIs without violating the control requirements and constraints identified in Chapter 3.

Levels of Automation

In the context of monitoring critical systems using machine learning techniques, the concept of automation can be split into four different dimensions or levels: the *automation of the data*

collection and feeding, the *automation of the learning process* and the *automation of the detection process*. In several contexts [109], it is also vital to talk about the *automation of the reaction process*, whenever the detection system is capable of launching prevention mechanisms automatically as the first response against a detected threat or anomaly. Thus, we define these five levels of automation as follows:

- *Automation of the data collection*: the collection of the raw data is a process that is inherently automatic, since it involves capturing and recording vast amounts of data involving measurements, logs, etc. for later processing and training.

- *Automation of the data feeding*: comprises all the procedures directed to preprocess, normalize and prepare the raw data in order to feed the inputs of the system. This process is difficult, costly and the majority of the real-life systems require a heavy preprocessing of the data that is performed manually or semi-manually. It is vital to provide automatic mechanisms with the object to adapt the functioning of machine learning-based system to face the real-life problems.

- *Automation of the learning process*: the learning process comprises several steps: *training*, *tuning* and *validation*. The training of the system is usually automatic, but the process of tuning and validation normally needs the supervision of an operator in order to set the system to a correct functioning for a context. Despite the general beliefs, the automation of the learning process has little implications regarding the supervision of the system, it is complex both for supervised and unsupervised methods, since they require following the above-mentioned steps. However, learning is performed before the deployment of the system, thus this kind of automation has less impact in the performance of the system.

- *Automation of the detection process*: the automation of the detection process is vital for the performance of the system, and it is usually referred to a deployed system that has to provide its services in real time. The need to tune the model in a later stage of the deployment of the system can impact negatively the performance of the system. In this case, the need of automation is vital, and the tuning of the models should be at least semi-automatic.

- *Automation of the reaction process*: after detecting any anomaly or intrusion, the system must take appropriate actions to avoid the problem to escalate. This reaction process can be divided into two categories, according to the nature of the reaction of the system: *passive reaction processes* and *active reaction processes*. Passive responses are typical from current IDS, and include actions such as raising alarms or logging off the system [187]; active responses are those implemented to react against the anomaly or the intrusion in order to avoid the system failure. In CIP, monitoring systems have traditionally implemented passive reaction processes based on sending warnings to the operators and making available the information for them to fix the system (this is observed through the review of the literature regarding the automation of the available solutions, and will be discussed in the following sections). These solutions are mostly semi-automatic and highly dependent on the presence and accuracy of the operators. This dimension of the automation needs the focus of the scientific community, in order to provide the IDS solutions with sophisticated automatic measures of defense. This way, there would be first response mechanisms to prevent failures to cascade through the critical systems in a rapid way [3].

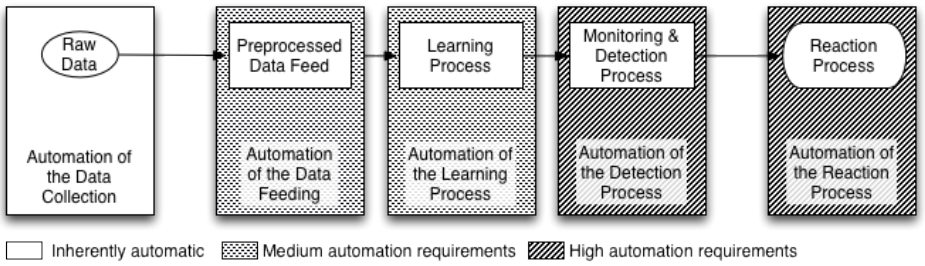


Figure 4.2: The needs of online automation of a critical IDS system

Figure 4.2 shows the levels of automation of an IDS. For the sake of clarity it is represented as a line, but each step is a cycle itself, and the improvement of the process of learning and the detection and reaction capabilities can only be made through refinement iterations. The main focus of the figure is in the levels of detection and reaction, that show the need of online (and possibly real-time) automation. There exist advanced systems that continuously refine their models after they are online.

Discussion

Besides the great need for automatic intrusion detection and reaction in CIs, and the possibility to achieve it through ML, it is necessary to consider that machine learning techniques are complex, and in practice, they pose certain difficulties whenever we try to apply them to real-world problems. Kotsiantis et al. in [177] provide an example of the process of applying machine learning techniques to a real-life problem where it is possible to appreciate that the process of generation automatic and well-tuned models of a system is composed of multiple steps of refinement to tune the learners. Despite general beliefs, the process of unsupervised learning is not much simpler than the supervised learning, since the final model has to go through very similar refinement cycles.

However, as we can see in Figure 4.2, the most significant need of online automation lays in the processes of detection and reaction. These two processes will provide the protection to the surveilled system, and once they are online, their posterior refinements are minimal. The processes of preprocessing and learning are usually performed offline, thus the costly procedures of applying machine learning techniques do not interfere with the efficient execution of the solutions, and they benefit from models that leverage complex knowledge of the surveilled system.

We believe that the introduction of machine learning techniques in critical context IDS would provide numerous benefits in two ways: in terms of *immediate automatic reaction to threats* and in terms of *understanding and leveraging knowledge about complex interdependent systems* that is beyond the expert knowledge.

Concerning the mechanisms that provide immediate automatic reaction to threats, the advantages are clear, since these systems could automatically face rapidly the numerous threats that put CIs in risk continuously. They could remove the need of displacing an operator to fix the problem, especially when the threatened systems are specially vulnerable, being isolated or with-

standing extreme conditions (e.g. remote subnetworks deployed in the field). Nonetheless, the disadvantages of automating critical decisions are great in this critical context, where human life could be at stake. However, building sufficiently intelligent automatic systems, with enough security and supervision mechanisms could provide safe solutions that deal with the real necessity of automating reaction measures for critical scenarios.

A substantial advantage of introducing machine learning techniques in the context of CSs is that these techniques (especially the techniques based on logics, due to their higher degree of comprehensibility of the generated models) are especially designed to discover the model of the (surveilled) system, and its internal complex functioning. Such tool results of incomparable power to researchers that try to model the complicated and interdependent relations and dynamics of CSs, outputting automatically-learned models of the systems that goes beyond the knowledge of an expert.

4.2.4 Review of the State of the Art: Approaches, Techniques and Automation for Detection solutions

Artificial intelligence and machine learning techniques can be applied to a variety of systems that provide protection for critical systems, mainly IDS solutions [171] [107], and few other components such as firewalls [188]. We have surveyed the literature in the search of solutions that provide specific IDS solutions for CIP, reviewing the characteristics and the degree of automation of each solution. For each of these systems, we analyze the levels of automation provided according to the classification provided in Section 4.2.3.

Table 4.1: Review of several systems according to the automation and knowledge dimensions.

System	Method	Prior Knowledge	Automation					Technique
			Data Collection	Preprocessing	Learning	Detection	Reaction	
[171]	Sup.	Free	Auto	No	Auto	Auto	Passive	Statistics, ML and Rules
[187]	Sup.	Required	N/A	N/A	N/A	N/A	N/A	Rules
[172]	Sup.	Free	Auto	No	Auto	Auto	Passive	Statistics
[189]	Unsup.	Mixed	Auto	No	Auto	Auto	N/A	Pattern Discovery
[173]	Sup.	Required	N/A	N/A	N/A	Auto	Passive	Rules
[99]	Sup.	Required	Auto	N/A	N/A	Auto	Passive	Rules, statistics
[107]	Unsup.	Free	Auto	Auto	Auto	Auto	N/A	Statistics, ML and Rules
[190]	Sup.	Required	N/A	N/A	N/A	Auto	Passive	Rules, ML
[152]	N/A	Required	N/A	N/A	N/A	Auto	Passive	Specifications
[191]	Unsup.	Free	Auto	No	Auto	Auto	Passive	Clustering

We have summarized our analysis in Table 4.1, where we can distinguish three different dimensions, namely: *prior-knowledge scheme*, *supervision* and *automation*, that categorize the reviewed systems according to the classification established in this section.

Düssel et al. [171] present a payload-based anomaly-based network IDS for CIs, capable of monitoring the traffic in real time. The IDS makes use of different techniques, the system extracts the information in the form of vectors, calculates the distance measures of similarity and compares them to a previously learned model of normality, indicating the presence or absence of an anomaly by raising alarms. Roosta et al. [187] introduce an anomaly-based IDS for wireless

process control systems. The IDS presented is theoretical, where the detection is based on expert-designed policy rules. Yang et al. present and IDS based on pattern matching, capable of detecting anomalies by analyzing the deviation from normal behavior. For this purpose they use autoassociative kernel regression models and a sequential probability ration test to discern an attack from the normal behavior. MELISSA [189] is a semantic-level IDS based on FP-Trees [174] that looks for undesirable user actions by processing logs in the SCADA control center. Carcano et al. [173] propose a state-based IDS that use rules to detect complex attack scenarios based on chains of illicit network packets. Autoscopy [99] is an IDS capable of detecting malware that tries to "hijack" pointers and routines of the system; first it learns the behavior of the system and during its operation, it uses statistics to discern anomalous behaviors and raise alarms. D'Antonio et al. present an IDS [107] that implements a rule learner to classify values into normal and attack data, it has a flow monitor component that extract statistical relations between different sessions to refine the learned model. Cheung et al. [190] propose a three-layer IDS that is based on models and patterns of the system designed by an expert. One of the layers implement a Bayesian learning module to detect changes in the availability of the surveilled system. Lin et al. [152] propose a specification-based IDS, that uses the formal specification of the system under surveillance to verify the correct use of the network packets. Raciti et al. present an IDS for SGs [191] based on clustering techniques, for detecting anomalies in the cyber and the physical levels.

In Table 4.1 we can observe the techniques and schemes used for the development of these IDS solutions, and the levels of automation implemented. It is interesting to observe, that the methods based on rules (pure transcriptions of the expert's knowledge) do not require data collection methods, since they do not undergo the training process for leveraging the knowledge. Regarding the automation of the reaction processes, we can see that current systems implement only passive methods of reaction. They are mainly based on raising warnings to the operators, and they have to manually perform the inspections, repairs of the systems and help in crisis situations (e.g. voltage peaks in pylons, high pressures in dams, etc.). The main disadvantage of the absence of automation in such systems is that help might arrive too late, and the failures of the system may cascade to other dependent systems, including other interdependent infrastructures, causing all kinds of havoc. Therefore we find it vital that effective measures are taken, in order to avoid the possible social and economical harm derived from a cascading failure. These measures must be automatic active reaction processes, maybe based on machine learning techniques, that are capable of providing effective countermeasures in the face of any kind of anomaly or attack. Next section is devoted to discuss these response mechanisms, the varied possible implementations and their suitability for CIP.

4.3 Awareness and Reaction

To understand the IDPRS, it is important to also understand the nature of the event they attempt to detect, the environment where they operate, the different kinds of processes that can be triggered to protect the surveilled system, and the possible types of solutions that can be launched. In order to provide safe IDPRS solutions to protect critical systems, we need to identify those desirable features, and most importantly, the characteristics that constrain the application and deployment of response solutions in critical contexts such as CIs.

Protection mechanisms put into place to safeguard CIs must be tailored to their environment, taking into account its constraints in order to ensure the correct operation of the system as a whole. IDPRS solutions, similarly to IDS, are designed to monitor and protect hosts (*host-based architecture*), or networks (*network-based architecture*) [109]. A host-based IDPRS must be tailored to the node where the solution is running, it must operate within the constraints imposed by the host, and therefore it should be well integrated with its environment. Network IDPRS solutions monitor the traffic of communication networks, and can be deployed in radically different contexts.

Generally, the internal networks of CIs and their control systems can be divided into three main types: *corporate networks*, the *SCADA center* and *remote substations* [112]. The first are the business local area networks connected to a SCADA to gain access to critical data streams on SCADA servers (e.g., historical data, alarms, etc.). Corporate networks are general-purpose complex infrastructures where the nodes of the network (e.g., servers, gateways, SCADA centers) have moderate to high computational capabilities and the constraints of these networks are minimal. SCADA centers are in charge of constantly monitoring the controlled infrastructures, using their communication networks to reach remote substations.

The nodes connected to SCADA centers are usually powerful, e.g., SCADA servers, gateways and some powerful RTUs in the main remote substations. However, the protocols they use are proprietary and restricted, thus the IDPRS solutions deployed in this environment could use powerful computational resources, but they have to be designed for the specific communications protocols. Remote substations constitute those control networks based on field devices (e.g., sensors, actuators) and communication interfaces (e.g., RTU, gateways, base stations) capable of transmitting commands from the central system to field devices deployed close to the controlled infrastructures, and sending sensorial measurements to the SCADA center.

In these remote networks, most of the nodes (sensors, base stations) have constrained computational power; moreover, the protocols used in the communication of their nodes are usually proprietary, or adapted to low-power devices. Thus the IDPRS solutions deployed in this context should have lightweight procedures, and they must be tailored to the specific protocols and restricted nodes of the networks. Summarizing, these are the requirements posed by the physical structures and components of CIs, however in our study we have to also determine the desirable features and characteristics for the IDPRS solutions that are needed in CIP.

Our research takes as its basis the methodological framework for situational awareness given in [3], which is based on two execution phases and a set of protection services, among them: detection, alerting and response. We expand the proposed methodology (see Figure 4.3), to describe the different methods and phases for detection and response in a critical context. The aim of our study is to analyze the features and components available for prevention and response in general-purpose networks.

To identify the characteristics that are useful for the protection of CIs, and the factors that limit the application of RS to constrained critical systems, we need a methodological framework for IDPRS solutions for CIP. In our study, we follow the situational awareness framework in [3], and consider the taxonomies for general-purpose IDPRS solutions proposed by N. Stakhanova et al. [4] and A. Shameli-Sendi et al. [5]. They constitute the basis we use and adapt to analyze the IDPRS solutions that better fit CIP. The framework and taxonomy we consider for our analysis is illustrated in Figure 4.4.

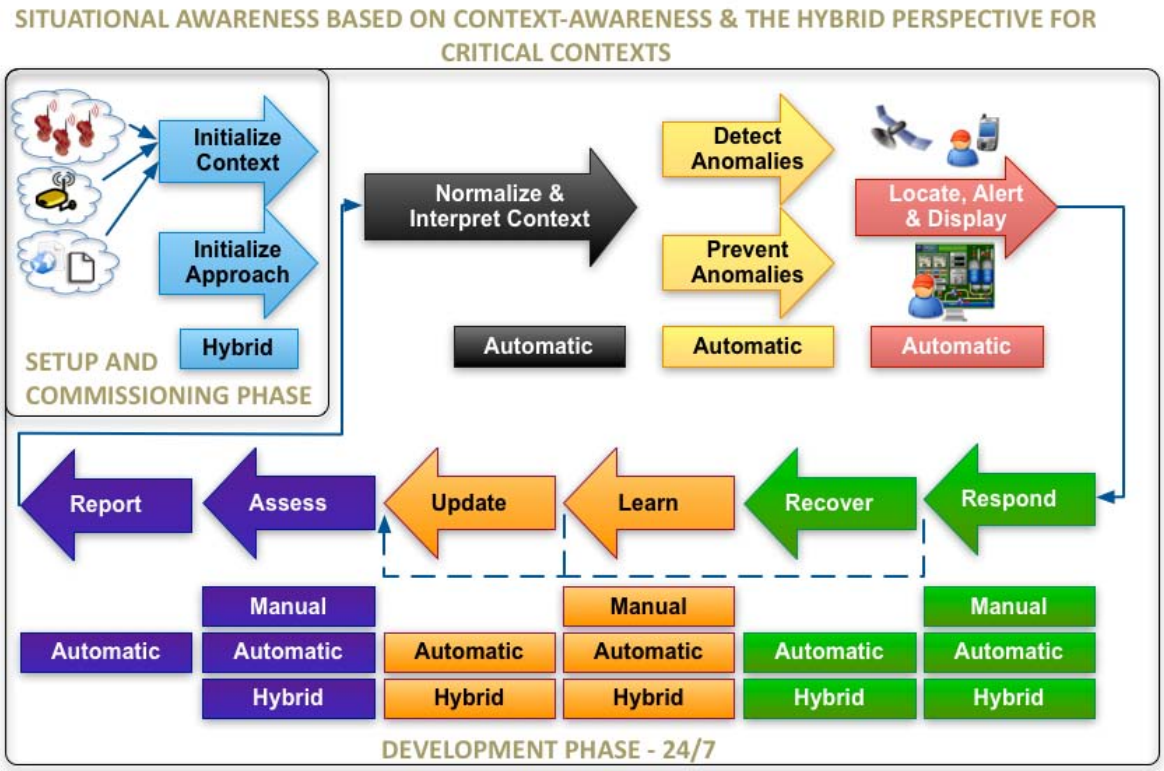


Figure 4.3: Situational awareness for critical contexts, adapted from [3]

Figure 4.4 is divided into three main modules, corresponding to the three key components or operational characteristics provided by the IDPRS: the *detection module*, the system's *automation degree* and the *response system*. These modules characterize the resulting IDPRS and the capabilities of detection, automation and response that this system will have, thus determining the degree of protection this system provides to CIs. In this section we describe the features of each element and study their advantages and disadvantages for critical systems.

4.3.1 Module 1: Detection

This module corresponds to the IDS component of the IDPRS, depicted in Figure 4.4, as previously described in detail in Section 4.2.1. As aforementioned, this component servers the vital purpose of providing detection mechanisms to feed the awareness and reaction system with valuable information about the event that threatens the system. The nature of the data provided would depend on the type of IDS method of detection, i.e., *anomaly-based detection*, *signature-based detection* or *specifications-based detection* (see Section 4.2.1).

In the case of an anomaly-based IDS, the system will provide alerts based on the deviation of the system of its known normal behavior. This will provide mainly statistic data in need for further analysis, in order to determine if the deviation of the norm is truly an abnormal behavior or simply a change on the dynamics of the surveilled system. In the case of both signature-based

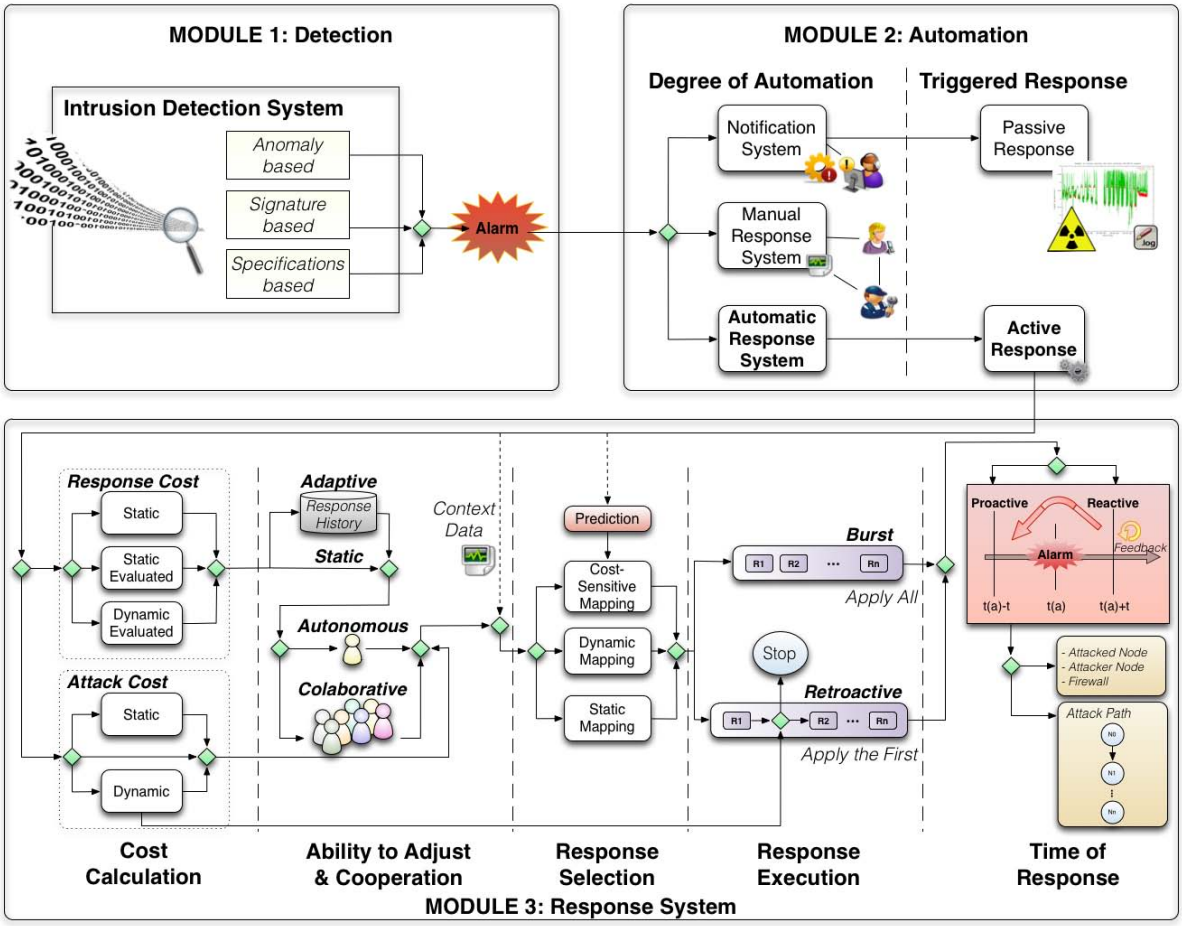


Figure 4.4: Methodological framework, containing a taxonomy of IDPRS, adapted from [4] and [5]

and specifications-based detection, it is easy to extract and analyze the detected events, since the matches against rules or specifications reveal which ones have been violated and provide exact information about the threat.

4.3.2 Module 2: Automation

A methodology to study the levels of automation of a given IDS solution for CIP is described in Section 4.2.3. From this methodology we identify that the fifth level of automation corresponds to the automation of the response mechanism implemented by the IDS. Since the IDPRS solutions provide an extension of the functionalities present in the IDS, it is necessary to determine the *degree of automation* they provide (according to the methodological framework for CIs given in [3]), in order to determine the automatic capabilities they implement. According to the level of automation, the RS are capable of providing different types of *triggered responses* [4].

Degree of Automation

Currently, research is mainly focused on providing manual or semi-manual reaction mechanisms, due to the difficulties that arise when trying to apply correct automatic responses against determined events. IDPRSs implement these responses at different levels, which can be subdivided and categorized into the following types (see Figure 4.4) [4]:

- *Notification systems*: reaction systems based on notifications principally generate alarms when threatening events are detected. They basically provide information about the occurrence, and the system administrator is the responsible for selecting the appropriate response. The majority of the existing IDS solutions provide this kind of mechanism [169]. It is important to note that this approach is not designed to prevent attacks or return the system to a safe state. The major disadvantage of this approach is the delay between the potentially harmful event and the human response [4].
- *Manual response systems*: manual response is a step ahead of the notification systems, with respect to automation. The system has a preconfigured set of actions that the administrator launches whenever a problem arises, and based on the characteristics of the event reported [5]. This process is not completely automated, however the countermeasures are configured in the system beforehand and the response of the operator is faster than in the previous scenario.
- *Automatic response systems*: are designed to be fully automated, thus (unlike the two methods mentioned above) human intervention is not required, and consequently there is no delay between the detection of the event and the response. However, due to the great difficulty in providing a high level of automation in the response mechanisms, the existing systems that provide automated response are very limited [4]. The main problem of this approach is the possibility that an inappropriate response is executed when a problem arises; it is also difficult to ensure that an automatic response is able to neutralize a problem [5].

Triggered Response

Those IDPRS that implement automatic reaction can be further categorized into two different classes, taking into account the type of their triggered response. This reaction determines the whole structure of the reaction system, and its intrinsic complexity. Accordingly, there are two different types of solutions: the *active response* mechanisms, and the *passive response* methods (see Figure 4.4):

- *Passive response*: the systems that present passive responses do not attempt to minimize the damage caused by the potentially harmful event or prevent a repeat in the future. The main objective is to notify the authority (e.g., the human operator, the SCADA center) and provide information about the occurrence [4].
- *Active response*: active systems try to locate the source of the detected event and provide active response actions to minimize the damage derived from the occurrence. The vast majority of the IDS solutions available only provide passive response, whilst the active reaction mechanisms are very limited at present [4].

Currently, most CIs are not equipped with active RS. However, the need for dynamic incident management and response systems capable of sending alerts for anomalies caused by malfunctions or intrusive presence are widely defended by the scientific and governmental communities [15, 35]. Most of the current IDS and IDPRS present in critical contexts implement passive response mechanisms [171, 173], therefore, we focus our analysis on the characteristics, functionalities and constraints of the automatic response systems.

4.3.3 Module 3: Response System

The third module of an IDPRS is the response system (see Figure 4.4), which is the component capable of selecting proper countermeasures to a given threat. This element is fed by the IDS component with insight about the threat, and has a determined degree of automation depending on the implementation of the system. The decisions taken to select the reactive measures against the threats can be made using various forms of computation. For example, they can be predefined or modified depending on the environment, they can be calculated autonomously or in a cooperative way. Also, the response can be applied in a reactive way or before the threat reaches the maximum.

Thus, there are multiple forms of implementing the IDPRS and the varied characteristics they can add. Specifically, our methodological framework considers the following features: *response cost model*, *risk assessment method*, *ability to adjust*, *cooperation ability*, *response selection method*, *response execution method* and *time of response*. These are the main features we evaluate in the different IDPRS solutions, there are other different variables that could be included in the study, such as the *applying location* and the *response lifetime* [192]. However, to determine the best solutions for CIP, we focus on the first.

Response Cost Model

Each threat (attack or fault) to a system entails a cost; likewise, each response (automatic or manual) to a threat has an impact on the system. Generally, the best responses are those that cost less, and it is always necessary that the cost of the selected response is less than the cost of the fault or attack. There are diverse forms of calculating the effects of the response actions, most of them belonging to the field of risk assessment [193]. There are three main ways of providing the automated IDPRS with the models for calculating the cost of a response in a particular situation: the *static cost model*, the *static evaluated cost model*, and the *dynamic evaluated cost model*.

- *Static cost model*: where the response cost is assigned statically using the opinion of an expert. This value has to be set for each response and it is usually preconfigured in the system [5].
- *Static evaluated cost model*: the cost is calculated and assigned statically to each response. Here, the evaluation mechanism usually computes the positive effects of the response (based on their consequences for the availability, confidentiality and integrity variables and performance metrics), and the negative impacts (in terms of availability and performance).

The combination of the two kinds of effects comprises the cost assigned to the response. The majority of cost models use the static evaluated cost model [5].

- *Dynamic evaluated cost model*: the system dynamically calculates the cost of the response based on the dependencies between the resources and the actors in the system [5]. The resulting cost-sensitive model is usually very accurate, and improves the performance of the IDPRS by allowing it to select appropriate responses that take into account the interdependencies of the system, its critical processes and the Quality of Service (QoS). However, due to its increased complexity, the majority of the IDPRS available implement static cost or static evaluated cost models.

In a critical context where the dynamics are well-known and the patterns of behavior are sufficiently static and predictable, IDPRS solutions that implement static cost models are a good option. Thanks to this static context, the range of possible threats and risks is limited to a restricted known set, and the possible countermeasure actions are equally limited. Therefore the simpler static methods perform well in these environments. However, whenever the behavior of the system has more complex dynamics, it is necessary to apply more sophisticated cost models. Especially if computationally powerful resources are available, the dynamic evaluated cost model is highly recommended, since the costs are calculated taking into account the critical processes and the multiple interdependencies existing between CIs.

Risk Assessment (Attack Cost)

Attacks or anomalies have a negative impact on the system, and through the calculation of the cost of these events, we can help the response system determine the best course of action for protection in a particular scenario. The assessment of these costs can be done *statically* or *dynamically*:

- *Static*: consists of assigning a static value to each resource of the system [5]. This type of risk assessment has a useful basic performance, and the procedures for its application are described in many existing standards (e.g., NIST [194], ISO 27005 [195]). However, static risk assessment does not provide the versatility and advantages of dynamic risk assessment for a dynamic context.

- *Dynamic*: the assessment of risk dynamically provides a real time process for evaluating of risk indices in the system [5]. The model is dynamically created by propagating the impact of the security variables through service dependency models [196], attack graphs [197], or general models based on metrics [5, 198]. Online risk assessment, although computationally complex, minimizes the costs of the threats and response events in the system, and allows the IDPRS to work, taking into account the context (state) of the system [199].

According to our study of the literature, a fully-fledged dynamic risk assessment component is present in few IDPRS existing solutions [192] (described further on in Section 4.3.4). However, given its importance in determining the best possible action responses for an RS, this component should be addressed for the IDPRS, if only in its simpler forms. Static risk assessment mechanisms such as the one proposed in [193] by A. Cárdenas et al., provide risk metrics such as the *average loss* or the *variance of the losses*. In [193], an heuristic is proposed to minimize risk by estimating the potential losses, to identify the high priority equipment (sensors) and to invest

resources in protecting them. This kind of risk assessment evaluation component is especially well suited to constrained, low-computational power sub-systems, such as the field networks in CIs.

More sophisticated approaches include dynamic risk assessment components which perform their evaluation of the risk level using different techniques. Y. Haimes et al. [200] study risk from different perspectives, and at different levels (e.g., physical risk, logical and information risk) to create models of risk. The Network Security Risk Model (NSRM) [201] is capable of assessing the risk of cyber attacks on process control networks using models. They show the different attack scenarios, and enable studying the progressions and consequences of the different attacks modeled and the risk levels introduced by the selected response strategies.

Also based on modeling, the risk-aware framework proposed in [202] contains an online component which measures the likelihood of success of an ongoing threat or attack, and the cumulative impacts (cost) of the threat and the response. These measures help the RS determine the need for activation or deactivation of the system's policies as countermeasures. Specifically, this framework is proposed for complex infrastructures and systems, with multiple interdependencies and constraints present in their normal operation. Another risk-aware RS is presented in [203], where the authors propose a framework for risk assessment in the smart grid.

R. Habash et al. identify metrics and factors evaluating the level of risk in this environment, and the level of risk remaining after applying countermeasures to tackle the threat. In their paper, A. Shameli-Sendi et al. [199] present ARITO, a response system using accurate risk impact tolerance. This RS contains a risk assessment component that measures the risk impact in real time. This system is also capable of providing feedback mechanisms for the countermeasures applied by the RS, provided by the calculations of the response goodness, which helps indicate the new risk level after applying the selected responses. After the risk evaluation, the system can decide whether to activate a response or not, and the strength of the response, according to the network risk level and the risk impact of each countermeasure.

Thus, given the different types of solutions at hand, we firmly believe that an IDPRS solution deployed within a critical infrastructure should have a risk assessment component. In order to determine the best response to anomalous situations, it is necessary to provide the RS with mechanisms to evaluate the costs of both the threat and the response actions occurring within the system. This feature is essential to apply accurate countermeasures to the threat, while avoiding causing havoc and widespread operative disruption due to a mistakenly applied response. Ideally, risk assessment should implement dynamic procedures, however, as we have discussed, in constrained areas of the network, it is possible to implement static well-adjusted procedures.

Ability to Adjust

The capability of a response system to adjust to new situations and scenarios is a desirable characteristic that supports its robustness and continuity. However, its implementation is costly in terms of computational complexity and difficulty of design, and in some cases deemed unnecessary. Thus we can find in the literature two positions regarding the adjustability of the RS: *adaptive solutions* and *static solutions*.

- *Adaptive*: the adaptability of the response is "*an ability of the system to dynamically adjust the response selection to the changing environment*" during the occurrence of a potentially harmful event [4]. The adaptive model usually adjusts its actions based on response history [5], and it can be in the form of: (i) the adjustment of the system's resources devoted to intrusion response; e.g., activation of additional IDS; or (ii) the consideration of the success or failure of the previously used responses.
- *Static (non-adaptive)*: the non-adaptive models provide a static response selection procedure, which remains the same throughout the lifetime of the IDPRS software. There is no mechanism for tracing the behavior of the applied responses, and the support to the system is manual (they can be periodically upgraded by the system's administrator). Although static, this kind of response model is simple and easy to maintain [4], so in some cases it is preferred by the administrators.

When deploying an IDPRS in a critical context, the ability of the system to adjust to its environment is in some cases very important. Since there are different types of networks within CIs, each of them having different environmental characteristics, the adjustability of a IDPRS is critical in these sections of the network where continuous changes happen, with frequent updates of the networks' nodes and continuously changing dynamics (e.g., in the corporate networks of CIs). However, other sections of the networks usually have very static patterns of behavior, and the need for an adaptive IDPRS there is not critical. This is the case, for example, in the networks that connect the SCADA with the RTU (or destination gateway), and in the communication networks between RTUs (or gateways) and sensors/actuators.

Cooperation Ability

Response systems can be designed in such a way that they perform their tasks *autonomously*, being capable of monitoring localized areas of the surveilled system or the holistic behaviors of the system. Conversely, they can be implemented *cooperatively*, which allows multiple IDPRS to communicate with each other and collaborate at different levels.

- *Autonomous*: the autonomous IDPRS solutions handle events independently, without communication with other components, at the level the threats are detected [4]. These solutions are usually aware of just a restricted part of the context of the system, thus the responses they can provide are generally localized.
- *Cooperative*: it refers to the set of RSs that combine efforts to respond to an intrusion. They can consist of several autonomous systems, capable of detecting and responding to intrusions locally, but with a final response strategy determined and applied globally. Sometimes, the IDPRS systems are directly built to operate cooperatively, which makes them perform better in terms of response speed and contained damage volume. Nonetheless, they are also more complex and require strong coordination and communication between their components [4].

Complex networks, where there are complicated dynamics and interconnected dependent systems, benefit from cooperative IDPRS solutions. It is possible to deploy multiple autonomous IDPRS modules for protection at different levels in the varied networks of CIs. However, cooperative detection systems could be able to correlate different events (occurring at different

levels or locations of the infrastructure) to provide behavior forecasting and improve the overall performance of the IDPRS (proactive protection). Nevertheless, it is not always cost-effective to increase the complexity of the security mechanisms, since they can diminish the efficiency of the operation of the infrastructure. Thus in constrained environments (e.g., remote substations), autonomous lightweight IDPRS are the recommended option.

Response Selection Method

Once the detection system has provided information about the threat that is placing the system at risk, and the costs have been calculated, the response mechanisms must determine the best actions to carry out to counteract the threat. The IDPRS is capable of making such decisions by associating the threat alerts with a determined set of actions. This association can be done in three different ways, with increasing degrees of complexity: *static mapping*, *dynamic mapping* and *cost-sensitive mapping*.

- *Static mapping*: here, each alert is mapped to a predefined response [5]. This procedure provides a model that is easy to build and maintain, however, it also makes the system predictable and thus vulnerable to intrusions (in particular, DoS attacks). Moreover, static mapping systems have an inherent inability to consider the current state of the whole system, therefore the actions triggered represent an isolated effort to mitigate a problem, without considering the current condition and impact of the response on the system. The application of this technique for large systems is infeasible and prone to errors, since the number of threat scenarios needed to be analyzed and the constant changes in system policies make this process extremely complex [4].
- *Dynamic mapping*: these RSs are more sophisticated than static mapping systems, since their response selection considers certain attack metrics (e.g., confidence, criticality, frequency) and network policies [204]. Each alert is associated with a set of response actions and when a potentially harmful event occurs, the system chooses, in real time, the best response action from the corresponding set, taking into account the characteristics of the particular threat [4]. This approach provides a fine-grained control over the automatic response of the system through adjustments to the metrics. However, its main drawback is that the RS does not learn lessons from one situation to the next; thus its intelligence level remains constant (i.e., it does not change) until the next system upgrade [5, 205].
- *Cost-sensitive mapping*: these RSs attempt to balance the cost of the damage caused by the harmful event and the cost of the response [4]. The optimal response is determined through the cost-sensitive model of the IDPRS, which includes cost and risk factors related to the event, and to each response [206]. Traditionally, cost-sensitive approaches use an offline risk assessment procedure, where the cost and risk factors are calculated in advance and the values are static. In some cases, this mechanism is completely manual, and it is the system administrator's task to update these values over time [4].

To improve the static procedures and lower the burden on the system's operators, online risk assessment components have been proposed to measure the cost of attacks, faults and automated responses [5]. Therefore, whenever the equipment capabilities allow it, it would be preferable to use the most sophisticated response selection techniques for the RS. Cost-sensitive or dynamic mapping would assist the operators in making the best decisions for incident response. However,

this advantage comes with the price of complexity and computational cost: it is necessary to analyze a vast number of factors (intrusion cause and effect, identification of optimal response, state of the system, maintainability) and to completely understand the problems addressed to provide optimal responses [4].

Additionally, it is necessary to adjust the measures of accuracy and adequacy of a response to the selection method implemented by the IDPRS. The maintainability of the system also increases whenever automatic sophisticated techniques are used for response selection. Furthermore, in ICS, the application of incorrect countermeasure actions can be devastating to the operation of the surveilled system, and can cause fatal errors to cascade throughout the infrastructure, and into other dependent interconnected CIs. Therefore, the response selection process must be supervised by the system's operator and the RS must adjust its functionality to the constraints of its environment.

In a critical context it is thus desirable to have an IDPRS well integrated with its environment. Within ICS, there are sectors with heavy-duty equipment and powerful computational capabilities, such as the SCADA center or the main remote substations (based on gateways as main interfaces), able to run IDPRS solutions that provide the most sophisticated response selection methods (i.e., cost-sensitive mapping). Also, the ICS contains sectors with constrained resources, where the equipment is not capable of performing high-complexity computations, e.g., in the field networks, where there are mainly lightweight sensors with low computational power.

In this case, it is also interesting to protect the system through IDS and IDPRS solutions, but these solutions have to be tailored to a constrained scenario. Thus, lightweight IDSs (like the one presented in [99]) and IDPRS can be deployed, where the methods used to perform their tasks consume fewer resources, e.g., providing an IDPRS with static mapping techniques for response selection. Then, by deploying the IDPRS that best suits each part of the system, it is possible to protect the infrastructure without the need to modify or add equipment to execute these tasks.

Response Execution

When the response system has selected the most suitable actions to counteract the threat, there are two ways of executing them: in a *burst*, or using *retroactive feedback*.

- *Burst*: this mode of execution does not take into account any mechanism to measure the risk once the selected response (or set of responses) is applied. This means that all the countermeasures are always applied, disregarding the possibility that a subset of the actions could be enough to mitigate the threat. This is the response execution mode usually present in the literature, its main weakness being the performance cost [5].
- *Retroactive*: in the retroactive execution, there is a feedback mechanism that measures the response effect taking into account the results of applying the most recent set of countermeasure actions. This measurement helps the system to make decisions before applying the next set of actions [5]. This kind of system was first presented by C. Mu et al. in [207], where the authors indicate several ways to implement the retroactive approach:
  - Selection window: each response has a static risk threshold associated with it, and to run the countermeasure it is necessary to consider the current risk index of the

system. If its value is higher than the static threshold of the action, the next response can be activated. With a selection window, the most effective countermeasures are selected to repel intrusions.

- Independent responses: this method involves measuring the risks associated with the countermeasure applied in order to make a decision about the application of the next one. Since responses are evaluated independently taking into account their cost impact, this step-by-step execution mechanism is more conservative than the previous approach, and more suitable to be applied in critical contexts.
- Grouped responses: when calculating the risk of a single response does not provide enough information to make the decision about running the next one, it is interesting to build groups of countermeasures. The decision to run the next round of responses is based on the general risk of the system. Once a group of countermeasures have been applied, the risk needs to be re-calculated. The challenge of this method is to determine how many responses in a round are considered enough to neutralize an attack.

ICSs are complex and delicate systems, thus when an undesired event happens, it is necessary to palliate its effects as soon as possible and in the least harmful way possible. Two approaches have been proposed: the burst response execution, which applies the countermeasures in bulk, as a whole, and the retroactive response execution. Although it is possible to find both types of RSs in the literature, the latter provides the advantage of stopping the process of responding to the event to evaluate the effects of the countermeasures already applied. Since, in a given situation, a subset of the countermeasures selected to palliate the anomalous state could be enough to restore the system to a normal operation, retroactivity capabilities are desirable.

Moreover, in a critical environment, executing countermeasure actions in a burst without evaluating their impact on the system could bring it to a critical state. Thus, automatic non-evaluated responses must not be executed in CIs without the supervision of a human operator. The reason behind this is the criticality of any action performed within CIs, since any mistaken activity (automatic or manual) can disrupt the operation of the system with unknown and potentially devastating consequences. Therefore, in this context it is better to execute countermeasures retroactively. However, these feedback mechanisms, as is the case with all adaptive approaches, face some challenges that make their use difficult, e.g., measuring the success of the most recently applied response or handling multiple threatening events.

There are several ways to provide feedback to the RS, ranging from simple static system's metrics, to a more dynamic approach such as including a risk assessment component. As suggested in Section 4.3.3, risk assessment can help determine the costs of the responses, to provide the desired feedback to the IDPRS. Here, in order to make the IDPRS more precise, risk assessment should be conducted dynamically (online). As we discussed previously, there are systems in the literature capable of tackling risk assessment in different scenarios, varying from general purpose environments [193, 199, 200] to complex cyber-physical infrastructures such as the telecommunications industry or the smart grid [202, 203].

However, since these online dynamic mechanisms make the IDPRS costlier in terms of the system's resources [5], they can be included in areas of the network with sufficient computation capabilities. In constrained areas of the network, it is possible to use simpler methods than

[193] to determine the suitability of the countermeasures for a determined situation. This can help computationally constrained devices to include RS with retroactive, although rudimentary, response execution capabilities.

Therefore, in a critical context, the unsupervised automated response execution of countermeasures in a burst should not be applied. Instead, supervised response execution, or retroactive execution mechanisms should be put in place to prevent the application of countermeasures that exceed the risk cost of the threat. It is therefore necessary, to delicately execute the responses, maintaining a control of the risk associated with the automatic procedures of the IDPRS. In this case, running and assessing the responses independently, or in small related groups can deliver adequate automatic responses, while minimizing the risks and costs of the automatic reaction.

Time of Response

It is possible to classify IDPRS solutions into *proactive* and *reactive* systems, taking into account the time instant when the IDPRS launches the response actions, with reference to whether the threat (e.g., an attack) has been already been confirmed or not.

- *Proactive (preemptive)*: proactive RSs foresee the incoming (potentially harmful) event and launch the response actions to help control the threat before it has affected the resource. This prediction is complicated to make and usually relies on probability measures and analysis of the system's behavior. Proactive solutions require the detection and response mechanisms to be tightly coupled, so the countermeasures can be triggered as soon as the event has been identified [4]. However, and although early response is highly desirable, it is difficult to guarantee the correctness of the triggered response action [208]; thus the proactivity of the system has to be balanced with the correctness of the responses provided.
- *Reactive (delayed)*: in these RSs, the reaction is delayed until the threat has been confirmed [4]. The threat can be confirmed using confidence metrics in the IDS or by the matching of the event's trace with an existing signature in the IDS. Clear distinction exists between the proactive mechanisms (calling them incident prevention systems) and the delayed IDPRS (calling them intrusion handling/response systems) [209]. The proactive response usually includes actions to restore the system state to its normal operation [4].

Reactive systems, because they are less complex compared to proactive mechanisms, are widely used in IDPRS solutions [5]. These RSs do not trigger any countermeasure actions until the threat has been detected. The problem with reactive solutions compared to preemptive systems is that, generally, the delayed response leaves the event "unattended" for a longer period, consequently allowing more damage to occur, and setting a greater burden on the recovery mechanisms and the system administrators. While this might not cause too much trouble in general networks, a delayed response is not suitable for critical systems [4].

N. Anuar et al. [168] detail the disadvantages of a reactive RS, defending the difficulty to return an affected system to its normal operation, while having to consider that the system remains in an unsafe state in the time window until the response actions are applied. However, deploying a proactive RS in a critical environment is challenging, since false positively detected threats might trigger countermeasure actions from the IDPRS and bring the system to an unstable

state. To prevent this problem and benefit from the advantages of a proactive IDPRS, it is necessary to finely tune the detection engine to make sure the rate of false positives is as low as possible.

Additionally, critical systems should not implement entirely automatic response systems, without human supervision, to avoid executing mistaken actions. Therefore, semi-supervised or supervised proactive IDPRS solutions are recommended in this environment. Of course, it is not always possible to deploy sophisticated and computationally costly solutions in several areas of CIs, due to their constrained nature. However, there are simple techniques, such as statistics or rule-based detection, which allows implementing lightweight preemptive systems. They are capable of detecting certain patterns and behaviors that deviate from the standard operation and which precede a threat, and launching adequate responses, even in a constrained environment such as the field sensor networks.

Thus, sophisticated proactive IDPRS can be applied to ICS in a context where the nodes have sufficient computational power, e.g., to the SCADA center, or to the networks that connect the SCADA center with the main remote substations. There, network dynamics are simple and the nodes are powerful. Another area of ICS that could make use of proactive solutions are the corporate networks, where there are complex dynamics and behaviors, but the resources are sufficiently powerful to apply behavior-based forecasting. Finally, as mentioned, sectors of ICS where the equipment has constrained capabilities (e.g., field sensor networks) can benefit from simpler, lightweight proactive RS solutions.

4.3.4 Review of the State of the Art: Approaches, Techniques and Tools for IDPRS solutions

A review of the literature shows the different approaches taken to build IDPRS solutions through the recent years. This analysis is based on the methodological framework and protection methods described in Section 4.3 and focuses on non-commercial general-purpose academic IDPRS solutions. There is a line of commercial tools that provide several interesting solutions regarding IDPRS mechanisms, however, the features implemented by these solutions have a proprietary nature which restricts their study to the characteristics publicized by the provider of these tools. N. Anuar et al. present a study based on Gartner's report on network IDPRS in their paper [168]. They analyze the level of response applied in commercial and research products, looking at IDS and IDRPS technologies, as well as Security Information and Event Management (SIEM) products (tools for real-time analysis and management of security alerts within a system and its network). Therefore commercial solutions are not included in our study, as we cannot identify the essential features needed for the protection of the CIs.

One of the first IDPRS systems found in the literature is DC&A (Damage Control and Assessment), defined by E. Fisch [209] in 1996. This system implements a dynamic mapping technique, specifically designed for intrusion control and assessment, with two main components: (i) a damage control processor to reduce and control the damage done by an intruder, while the intrusion is still in progress, and (ii) a damage assessment processor that determines the harm done to the system to subsequently help the recovery processes. This system selects the response actions using an index, the suspicion level of each user's activity and applies the responses assigned to the suspicion level. If this index increases, different responses are selected. When the intruder

leaves the system, the damage assessment processor determines the actions needed to revert the state of the system to a safe and healthy state.

The Cooperating Security Manager system (CSM), proposed by G. White et al. [210], is a dynamic mapping IDRPS that selects its response strategies based on IDS confidence information about the intrusive behaviors, and severity metrics associated with the attack. Although not originally designed to be proactive, CSM can be configured to counteract intrusions proactively. This IDPRS is a distributed collaborative solution equipped with autonomous response mechanisms. CSM allows hosts to share information and detect intrusive user activity in a cooperative way, but the response actions are decided and launched by each host locally [4]. One of the main drawbacks of this model is that it is not capable of learning from attack to attack [5].

Another cooperative model is EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances), proposed by P. Porras and P. Neumann in [205]. EMERALD is a distributed framework for network intrusion detection and prevention based on the insertion of different layers of monitors in the network. The component in charge of response is the resolver, which analyzes attack reports and coordinates the reaction efforts. There are several resolvers, each of them responsible for the local strategy of reaction, but able to communicate with the resolvers of the other layers, making the response selection a global procedure. As in CMS, the strategy of reaction is based on security metrics and confidence values.

In the 2000s, one of the first IDPRS to be published was the BMSL-based (Behavioral Monitoring Specification Language), by T. Bowen et al. [211]. This system has a static mapping response selection mechanism, which bases its operation in pre-specified countermeasure actions. BMSL describes the system's behavior using a finite state machine automaton, which autonomously assigns a countermeasure to each path. Another solution from 2000, the SoSMART system, by S. Mnsman and P. Flesher [212], is an agent-based IDPRS with a statically mapped response selection procedure. The incident cases are designed by the user and mapped to the appropriate countermeasures. Additionally, the system uses case-base reasoning as an adaptation mechanism in charge of determining if the current solution corresponds to intrusive behavior.

Another autonomous IDPRS is the PH system, designed by A. Somayaji and S. Forrest [213]. PH is based on a behavioral profile of the system composed of sequences of system calls. The calls that deviate from the normal behavior are considered anomalous and can therefore be marked to be counteracted. The system only implements two simple kinds of response actions: suspending the suspicious processes or aborting them permanently. As do the majority of IDPRS, PH implements a delayed response mechanism, i.e., it waits until the intrusion has been confirmed.

W. Lee et al. propose a cost-sensitive IDPRS [214] based on three cost factors: (i) *operational cost*, the cost of processing and analyzing data for detecting intrusions; (ii) *damage cost*, the amount of damage that can be caused by an attack when the IDS is ineffective; and (iii) *response cost*, the cost of applying the response when the attack has been detected. These factors combined present the total cost of the intrusion, and they help the system select the appropriate countermeasure in every case. One of the most complex dynamic mapping approaches of this decade is the Adaptive, Agent-based Intrusion Response System (AAIRS) based on an agent architecture [204].

AAIRS is a an agent-based complex system, where multiple IDSs monitor a host and generate

alarms. These agents operate at the different layers of the response process. Firstly, the intrusion alarms are processed by the master analysis agent, which calculates the confidence level based on pre-set decision tables and classifies the attack as new or ongoing. The system then passes this information to the analysis agent, which provides an action plan based on a seven-dimensions response taxonomy: degree of suspicion, attack time, attacker type, attack type, attack implications, response goal, and policy constraints. Lastly, the tactics agent decomposes the response plan into particular actions and activates the appropriate components of the response toolkit. It is capable of adapting its responses to each situation using the IDS' confidence metrics, which indicate the number of false positive alarms against the correct number of intrusions generated by each IDS. Similarly, the success metrics indicate the response actions that were successful in the past. A drawback to AAIRS is that it requires the intervention of the system administrator after each incident.

In 2001, S. M. Lewandowski et al. presented another cooperative RS, Survivable Autonomic Response Architecture (SARA) in [215]. It is composed of several components that function as: sensors (gathering of information), detectors (analysis of sensor data), arbitrators (selection of adequate response actions), and responders (implementation of response). SARA's components are arranged to provide the highest possible levels of detection and prevention. For example, each host can have an arbitrator to provide intrusion response while the selection response comes from a global (cooperative) strategy.

The Cooperative Intrusion Traceback and Response Architecture (CITRA), presented by D. Schnackenberg et al. in [216], is another cooperative agent-based system. CITRA uses a neighborhood structure to propagate the intrusion information until it reaches a centralized authority called the discovery coordinator, which determines the optimal response to the intrusion. The discovery coordinator centralizes the global response, however, the local CITRA agents are in charge of delivering the local response actions. CITRA's framework is composed of network-based IDS, security management systems and network components (e.g., routers). Their aim is to detect the intrusion, trace it back to the source and coordinate the suitable reactions. Two factors guide the response mechanism: the certainty (likelihood of the event being an actual intrusion) and the severity of the intrusion (potential damage to the system). Once these two parameters, which define the characteristics of the event, have been determined, the response action is chosen from a pre-determined set.

Also in 2001, X. Wang et al. [217] presented TBAIR, the Tracing Based Active Intrusion Response system, a dynamic-mapping non-adaptive IDPRS capable of tracing the intrusion back to the source host to dynamically select a proper response to mitigate its effects; e.g., by blocking the intruder remotely or isolating the affected hosts. The model proposed by T. Toth and C. Kruegel [206] is capable of considering the cost and benefits of the countermeasures. It implements a network RS capable of modeling dependencies between the services and resources of the system, in the form of a tree. This model can reveal priorities in targets and helps evaluate the impact of the response strategy on the system. Likewise, the algorithm for response selection can take into account the assigned static penalty cost of having a resource unavailable, so it can indicate the impact that the response strategy has on the system. Considering this model, the algorithm chooses to apply the set of actions that has the least negative impact on the system.

S. Tanachaiwiwat et al. [218] present a cost-sensitive, static IDPRS. The system is non-adaptive

given the difficulty of calculating the effectiveness of a given countermeasure. The IDPRS is based on the efficiency of the IDS, the alarm frequency per week (indicating the number of alarms triggered per attack) and the potential damage cost. These variables serve to identify the best reaction strategy from a predefined list of responses. Similar to [206], the system proposed in [196] presents a cost-sensitive, dynamic RS capable of modeling dependencies between the services of the system to identify the impact of the different countermeasures. This system implements a delayed model, which suspends any action until after the threat has been confirmed.

The IDPRS uses host IDS and provides two ways of classifying the resources of the system: a resource hierarchy or a system map. The resource hierarchy is a directed graph, where its nodes are specific system resources and the graph edges represent dependencies between them. Each node is associated with a set of reactions to restore its working state when attacked. The response is selected using: (i) the reaction cost, corresponding to the sum of the resources affected by the response; (ii) the reaction benefit, through the sum of nodes previously affected and restored to a working state; and (iii) the cost of the threatened resource. When there are alerts about nodes that have not previously considered, the system adds them dynamically to the map/hierarchy.

In 2005, B. Foo et al. presented ADEPTS, an adaptive and proactive RS, in [219]. ADEPTS uses Intrusion Graphs (I-Graph) to model intrusions, which identify attack targets, the possible spread of the intrusion, and those nodes where it is possible to apply successful responses. The RS maps the alarms provided by the IDS against the I-Graph and selects the countermeasures taking into account their calculated effectiveness, their potential to cause disruption and the level of confidence of the system being intruded. ADEPTS uses feedback mechanisms coming from the affected nodes, where parameters such as the confidence level of the attack and previous similar experiences help in estimating the success or failure of the applied response. As opposed to AAIRS [204], this system is capable of automatically updating the response effectiveness metric.

FLIPS, Feedback Learning Intrusion Prevention System [220], is another proactive IDPRS, which emulates the applications in a restricted environment before their execution. Thanks to this previous emulation, the system can recognize code injection attacks with only a few bytes of data, and prevent the system from executing their malicious code. M. Papadaki and S. Furnell present a cost-sensitive RS capable of evaluating the static and dynamic context of an attack in [221] using a database of their characteristics (e.g., target, applications, vulnerabilities). It also takes into account the characteristics of the responses available (e.g., counter-effects, stopping power, transparency, efficiency, and confidence level) to propose different kinds of countermeasures according to the attack, and it is capable of adapting the response to changes in the environment.

Similar to FLIPS [220] and ADEPTS [219], N. Stakhanova et al. propose in [222] another proactive, cost-sensitive IDPRS designed to detect anomalous behaviors in terms of system calls. The IDS tries to match the sequences of system calls with sets of normal and abnormal patterns to determine whether there is an attack or not. If the IDS finds no signature (pattern) matches, a machine learning engine is used to discern whether the behavior is normal or anomalous. Since this system is proactive, the reactions are triggered before the attack is completed. To operate in advance, the IDPRS has to have a predetermined mapping between system resources,

countermeasures and intrusion patterns. When a sequence of system calls matches an abnormal pattern, the RS chooses the proper reactions available that have the least negative effect. The effectiveness of the response is measured and its value is considered for future events.

K. Haslum et al. presented the Distributed Intrusion Prediction and Prevention System (DIPS) in [198], a cost-sensitive, real time IDPRS with prediction and risk assessment modules based on fuzzy models. Fuzzy logic is used to automatically estimate and infer risk, taking over this task from the security and risk experts. DIPS implements a hidden Markov model to represent the interaction between the attacker and the system's network. Also in 2007, M. Jahnke et al. [197] proposed a cost-sensitive IDPRS that uses graph-based mechanisms for risk assessment. Graphs model the effect of attacks in the resources, and the effects of the countermeasures in terms of availability. This system expands on the idea of T. Toth and C. Kruegel [206], using directed graphs to model dependencies between the resources, and to calculate differences between system states.

One of the few early adaptive solutions is presented by C. Strasburg et al. in [223]. The authors propose a structured methodology to evaluate the cost of a countermeasure based on three parameters: (i) operational cost: the cost of preparing and developing responses; (ii) impact of the reaction on the system: which measures the negative effect of the response action on the system; and (iii) response goodness: based on the number of possible intrusions that the countermeasure can cope with, and also the number of resources that can be protected by the countermeasure. The total response cost is a combination of these parameters.

C. Mu and Y. Li propose IDAM&IRS (Intrusion Detection Alert Management and Intrusion Response System) in [207], which includes an RS based on hierarchical task networks. Each countermeasure of the RS has an associated static risk threshold calculated using its ratio of positive and negative effects. The reaction is chosen using a response selection window that shows the most effective countermeasures. IDAM&IRS triggers a response when its value is higher than its static impact risk index. The action is selected according to the goal of the RS: analyzing, capturing or masking the attack, maximizing the system's confidentiality or integrity, recovering from the attack, or sustaining service. Each goal has its own sequence of responses according to the risks they imply, e.g., weak responses earlier in time, and strong responses later.

W. Kanoun et al. presented in [202] a risk-aware framework composed of an online model and its architecture, which allows activating or deactivating response policies. They focus on the need of having deactivation mechanisms which allow the RS to stop applying responses when it calculates that the impact of the reaction surpasses an sufficient threshold. This proactive model bases its decisions on parameters such as the likelihood of the success of an on-going threat and the accumulative impact of the threat and the response. Also in 2010, N. Kheir et al. [224] proposed a proactive solution based on dependency graphs. This IDPRS extends the propagation process developed in [197], by integrating the evaluation of the impact of an attack on the security variables Confidentiality, Integrity and Availability (CIA). Each resource in the dependency graph has an associated CIA vector, where the variables are updated by active monitoring estimation mechanisms or by extrapolation. The dependencies in the graph can be structural or functional.

In 2013, S. Wang et al. presented in [225] a middleware RS, with the aim of providing cost-benefit security hardening. The authors' approach is the use of attack-graph models together

with Hidden Markov Models to explore the probabilistic relation between system observations and states. With this probabilistic insight, the IDPRS runs heuristic searching algorithms for cost-benefit analysis, to determine the best security hardening measures available for the defense of the system. Also in 2013, A. Shameli-Sendi and M. Dagenais published in [199] a cyber-attack RS using Accurate Risk Impact Tolerance (ARITO) (see Section 4.3.3). The main component of ARITO is the online risk assessment module, which evaluates in real-time the risk impact. This model also provides a feedback mechanism for retroactive response execution, it measures the goodness of the applied countermeasure, and indicates the new risk level after the application of the selected action(s).

M. Zaghdoud and M. Al-Kahtani describe in [226] an RS based on contextual fuzzy cognitive maps and ontology-based knowledge representation. This IDRPS has three layers, the first layer uses ontologies to recognize the intrusions in the system. The second layer uses fuzzy cognitive maps to determine the context of the effect of the intrusion on the target system and to diagnose it. In the third layer, there are response agents that select the suitable remedies available and react in a passive (alerting the system administrator) or active (applying determined countermeasures) way.

In 2014, S.A. Zonouz et al. presented in [227] a game theory-based IDPRS, where the RS and the adversaries are modeled as opponents in a two-player stochastic game. Its cost-sensitive response selection method uses attack-response tree structures, which, solving partially observable competitive Markov decision processes, derive the optimal reactions in each case. The RS tries always to minimize the mathematical costs, while maximizing the benefit of the reactions applied. B. Fessi et al. specify in [228] a genetic algorithm-based IDPRS, fed by a double-IDS cooperative schema (network and host IDS). This RS uses a weighted linear combination model to standardize the multiple attribute alternatives prior to the decision analysis. The genetic algorithms are used to determine the appropriate countermeasures with the help of a decision model, which takes into account the financial cost, reputation loss, and processing resources.

After our thorough review of the literature, we find that all of the aforementioned solutions, which implement active reaction mechanisms, have been developed for general purpose networks. It is possible to identify automatic IDS solutions with notification systems capable of sophisticated alerting processes [173]; but, they rely heavily on the presence of human operators to confirm or perform the countermeasure actions needed. Therefore, we can state that (to the best of our knowledge) in the public domain there are still no specific IDPRS solutions for the field of CIP which are capable of implementing active and automated response solutions.

Additionally, there are other solutions (manual response systems) for CIP that provide passive countermeasures in the form of notifications to the operators and they are capable of automatically performing harmless actions such as logging of data records for forensic analysis, such as the reputation-based early warning system proposed in [229]. However, and as we have stated, active IDPRS solutions are widely needed for the critical infrastructure protection [15]. Thus, it is necessary to employ more effort in developing balanced solutions for automated response in critical contexts.

4.3.5 Analysis of Solutions and Countermeasures

In the previous section, we have described some of the different IDPRS solutions existing in the literature and their evolution from their first appearance in the 90's to the present day. This section is dedicated to analyzing and assessing these solutions, evaluating the different methods used regarding intrusion response and prevention against potentially harmful events occurring in CIs. Taking into account the methodological framework and the taxonomy provided in Section 4.3, the IDPRS are categorized according to the characteristics of the current solutions, and they are presented in Table 4.2.

The systems are chronologically ordered, and the fields of the taxonomy that do not apply to a specific solution are left blank. In Table 4.2, it is possible to appreciate that the majority of the existing solutions implement reactive mechanisms, executing the response actions without stopping (in a burst). We can also observe the trend over the years to move from static solutions with respect to the calculation of the costs of the intrusion and the response, to more dynamic, even cost-sensitive ones. In recent years, we have seen several attempts to provide adaptive solutions, which help adjust to the specific situation of the system.

The review of the state of the art in the previous section covers solutions from diverse fields of knowledge and application. In this section, we also evaluate these solutions according to their possible application in the field of CIP. In order to do so, it is important to analyze the most common intrusion responses, and provide a simple taxonomy to study which kinds of solutions better fit in a critical scenario. Table 4.3, originally based on the study available in [4] and later extended for our purposes, provides an overview of the main types of responses that are found in the literature (note that this table focuses mainly on academic work, therefore it is not exhaustive).

In Table 4.3, and according to our IDPRS taxonomy (see Figure 4.4), we divide the possible countermeasures that an IDPRS can provide into: *passive* and *active* responses. In the first place, it is important to consider passive reactions, which are the most abundant solutions in the literature and they are usually included in the normal operation of some IDS solutions [4]. We classify *passive solutions* into three categories, namely: *administrator notification*, *prevention measures* and *others*. The first category corresponds to those systems whose mission is to log the system's information and state, and also alert the system's administrator or human operators to control the situation. As we have mentioned, notifications to administrators are the most common operations implemented in deployed IDS/IDPRS systems for CIP. Prevention measures are mechanisms that are sometimes present in protected systems. In our study, we have distinguished 6 main types of *preventive mechanisms*:

- Cryptography: using cryptography for data encryption is an effective approach to prevent attackers from understanding captured data [230]. Bus and memory encryption increases the difficulty of successfully attacking a device [231]. It is also possible to support the OS security by providing secure execution of cryptographic primitives as OS services [232]. There are other mechanisms such as link-layer encryption and authentication, multi-path routing, identity verification, bidirectional link verification, and authenticated broadcast that can help protect networks against intrusions and attacks [233].
- Security Policies: are the definitions of the measures taken by the organizations to provide security to their entity. Usually they address constraints, restrictions and rules imposed

Table 4.2: Classification of the existing IDPRS solutions according to our taxonomy

IDPRS	Year	Response Cost	Attack Cost	Risk Assessment	Ability to Adjust	Cooperation Ability	Response Selection	Response Execution	Response Time
[209]	1996	Static			Static	Cooperative	Dynamic Mapping	Burst	Reactive
[210]	1996	Static			Static	Autonomous	Dynamic Mapping	Burst	Reactive
[205]	1997	Static			Static	Cooperative	Dynamic Mapping	Burst	Reactive
[211]	2000	Static			Static	Autonomous	Static Mapping	Burst	Reactive
[212]	2000	Static			Static	Cooperative	Static Mapping	Burst	Reactive
[213]	2000	Static			Static	Autonomous	Static Mapping	Burst	Reactive
[214]	2000	Static	Static		Static	Autonomous	Cost Sensitive	Burst	Reactive
[204]	2000	Static Evaluated			Adaptive	Autonomous	Dynamic Mapping	Burst	Reactive
[215]	2001	Static			Static	Cooperative	Dynamic Mapping	Burst	Reactive
[216]	2001	Static			Static	Cooperative	Dynamic Mapping	Burst	Reactive
[217]	2001	Static			Static	Cooperative	Dynamic Mapping	Burst	Reactive
[206]	2002	Dynamic Evaluated	Static		Static	Cooperative	Cost Sensitive	Burst	Reactive
[218]	2002	Static	Static		Static	Autonomous	Cost Sensitive	Burst	Reactive
[196]	2003	Dynamic Evaluated	Dynamic	Dependencies	Static	Autonomous	Cost Sensitive	Burst	Reactive
[219]	2005	Static	Static		Adaptive	Autonomous	Cost Sensitive	Burst	Proactive
[220]	2005	Static Evaluated	Static		Static	Autonomous	Static Mapping	Burst	Proactive
[221]	2006	Static Evaluated	Static		Static	Autonomous	Cost Sensitive	Burst	Reactive
[222]	2007	Static Evaluated	Static		Adaptive	Autonomous	Cost Sensitive	Burst	Proactive
[198]	2007	Static	Dynamic	Attack Metrics	Static	Cooperative	Cost Sensitive	Burst	Proactive
[197]	2007	Dynamic Evaluated	Dynamic	Attack Graph	Static	Autonomous	Cost Sensitive	Burst	Reactive
[223]	2009	Static Evaluated	Static		Adaptive	Autonomous	Cost Sensitive	Burst	Reactive
[207]	2010	Static Evaluated	Dynamic	Attack Metrics	Static	Cooperative	Cost Sensitive	Retroactive	Reactive
[202]	2010	Static Evaluated	Dynamic	Dependencies	Adaptive	Autonomous	Cost Sensitive	Burst	Proactive
[224]	2010	Dynamic Evaluated	Dynamic	Dependencies	Static	Autonomous	Cost Sensitive	Burst	Proactive
[225]	2013	Dynamic Evaluated	Dynamic	Security Metrics	Static	Autonomous	Cost Sensitive	Burst	Reactive
[199]	2013	Dynamic Evaluated	Dynamic	Metrics	Adaptive	Autonomous	Cost Sensitive	Retroactive	Reactive
[226]	2013	Dynamic Evaluated	Dynamic	Graphs	Adaptive	Autonomous	Dynamic Evaluated	Burst	Reactive
[228]	2014	Dynamic Evaluated	Dynamic	Graphs	Adaptive	Cooperative	Cost Sensitive	Retroactive	Reactive
[227]	2014	Dynamic Evaluated	Dynamic	Graphs	Adaptive	Cooperative	Cost Sensitive	Retroactive	Reactive

Table 4.3: Taxonomy of intrusion response actions

Triggered Response	Type / Field of Response	Response Action	Details and Examples	
Passive	*Administrator Notification*	Generate Alarm	e-Mail	Online / Pager Notification
		Generate Report	Attack Target	Criticality
			Time	Source IP / User Account
			Description of Suspicious Packets	Intrusion Statistics
	Prevention Measures	Cryptography	Authentication Mechanisms	Masking: Dummy Operations
		Security Policies	Access Control	Information Protection
			User Account	
		Monitoring	IDS / IDPRS	Host-Based Vulnerability Scans
			Security Self-Awareness	
		Protective / Defensive Infrastructure	Demilitarized Zones	Firewall
			Proxy	
		Low-Level Prevention	Synchronized Docks	Directional Antennas
		Session Measures	Cookies	Packet-Leashes
	Others		Enable Local / Remote / Network Activity Logging	Enable Additional IDS
			Enable Intrusion Analysis Tools	Backup Tampered Files
			Trace Connection for Information Gathering	
Active	*Host - Based Response*	Operations on Files	Deny Full / Selective Access to File	Delete Tampered File
			Restore Tampered File from Backup	Allow to Operate on Fake File
		Operations on User Accounts	Restrict User Activity	Disable User Account
		Operations on Processes and Services	Shutdown Compromised Service/Host	Restart Suspicious Process
			Terminate Suspicious Process	Disable Compromised Services
			Abort Suspicious System Calls	Delay Suspicious System Calls
		Trust Mechanisms	Reputation	Credit and Token-Based Trust
	Network - Based Response	Disable / Block Operations	Block Suspicious Incoming / Outgoing Connections	Enable / Disable Firewall Rules
			Block Connection to Port / IP Addresses	
		Isolation Actions	Isolate / Quarantine Node	
		Routing	Multi-Hop Routing	Trust-Based Routing
			Secure Routing	Routing Discovery
			Evolutionary Algorithms for Routing Discovery + Trust	
		Deceiver Devices	Honeypots and Honeypets	

on the systems and on their operators/users. Since each organization is free to define or adopt the security policies to be implemented, there are many variants of them. Security policies for CIs are called *CIP policies*, and they follow governmental guidelines [150]. IDPRS solutions, following these guidelines, can identify violations in the security policies of the surveilled system [109].

- Monitoring: the monitoring system par excellence is the IDS, which supervises the local (host or network) system's operations and state [230]. IDS solutions for CIP implement different degrees of automation using varied types of detection engines, e.g., P. Düssel et al. [171] present anomaly-based network IDS, capable of analyzing the traffic in real time; A. Carcano et al. [173] propose a state-based IDS using rules to detect complex attack scenarios; H. Lin et al. [152] have designed a specification-based IDS, relying on the formal specification of the system under surveillance to verify the correct use of the network packets.

- Protective/defensive infrastructure: consists of devices or system configurations designed as prevention mechanisms, capable of performing protection tasks. Following the National Infrastructure Security Co-ordination Centre (NISCC) good practices guidelines [133], this line of defense can include *firewalls*, *Demilitarized Zones* (DMZ) and *proxies*. Firewalls, are software or hardware security systems that control the incoming and outgoing network traffic based on specific rule sets [234]. The DMZ are physical or logical subnetworks that also provide services to an external untrusted network, allowing only the access from the outside to the DMZ nodes, but not to other nodes of the internal system [109] (e.g., to gain access to historical servers). Proxies represent intermediary interfaces capable of helping their clients to make indirect network connections to other network services, and filtering incoming requests [234]. These measures are usually present in current CIs at several different levels of the infrastructures, and highly appreciable in SCADA architectures.

- Low-level preventive mechanisms: are physical measures or procedures implemented at the lower layers of the communication systems to prevent intrusions. Examples of such measures are directional antennas in wireless devices or synchronized clocks [230, 235], which limit the possibilities of traffic interference and disruptions by restricting the access to the communication channels. These measures do not impose any overheads to the protected infrastructures, thus they are highly recommendable mechanisms to add to CIs.

- Session/communication measures: are techniques that add security at the session or communications level, e.g., packet leashes or cookies. A *leash* is the information added inside a packet to restrict its transmission distance [235]. *Cookies* are small pieces of data sent from a website and stored in the user's web browser to remember stateful information [84]. These measures are frequent in general-purpose networks, however, they might not be present in CIs' industrial protocols. Nevertheless, they can be implemented in sections of the CIs' networks that use general-purpose communication protocols, e.g., the corporate networks.

Lastly, in Table 4.3 we provide also another category to contain a broader range of passive response methods that are not directly related to the categories mentioned above. Examples are the activation of additional IDS, logging mechanisms, and intrusion analysis tools, backups of tampered files, tracing connections to their source, etc. It is important to note that we have reflected in Table 4.3 only a sample of the many possible countermeasures applicable to

critical infrastructures, always taking into account the need to observe the constraints of CIs and the specific context where the response actions are applied. Concerning the *active reaction mechanisms*, we can divide them into two groups [4]: *host-based* and *network-based* response actions.

Host-based responses refer to those local procedures which perform operations to modify parameters or processes that run within the affected nodes; e.g., operations on files (restore, delete files), operations on user accounts (restrict activities, disable accounts), operations on processes and services (shutdown, restart, disable, abort, delay actions), and also trust mechanisms. Trust-based mechanisms such as the use of reputation, credit-based trust or token-based trust have gained relevance in the protection of information in communication networks [233, 235]. Network-based response, conversely, corresponds to those activities performed in the communications network and that affect communication's services and parameters. Here we can differentiate responses such as disabling or blocking network operations [234], isolating segments of the network [235], modifying routing parameters [233] or setting up deceiver devices [236].

Currently, IDS/IDPRS solutions designed specifically for CIP implement some of the passive methods present in Table 4.3, such as the generation of reports and logs for later forensic analysis [229], security policies [152, 190], the presence of defensive infrastructure [229], etc. However, some CIs still lack important passive prevention mechanisms; for example, systems based on the Modbus/TCP protocol [190], which is commonly used in SCADA and DCS networks for process control, lack authentication of the source of a request. This gives a chance for an adversary to attempt to gather information on the industrial system and the network in general [190]. Active reaction solutions, consequently, are rarely applied to critical systems.

The reason for the absence of active reaction mechanisms is twofold. In the first place, CIs need stable well-behaved environments to perform their functions and it is critical that this restriction is always observed, since any change in this scenario could have a big impact on the correct operation of the infrastructures and cause disruptions in the critical services they provide. Thus any active reaction mechanism must not (directly or indirectly) cause disruptions in the normal operation of CIs. Secondly, the legacy systems and proprietary protocols and components traditionally present in CIs [1] make the development of new security mechanisms very difficult for the research community, therefore there is little research into this topic.

However, and despite these difficulties, it is necessary to develop and deploy active response solutions in CIs, thus we have selected some of the possible countermeasure mechanisms that come from IDPRS solutions for general-purpose networks, which can be adapted to protect critical systems (see Table 4.3). We divide the responses into: host-based response, where the response system applies the security measures to the host system, such as modifications to the user accounts and permissions and trust mechanisms; and network-based response, focused on network operations, e.g., the installation of deceiver devices (honeypots, honeypets, etc.), active routing techniques, etc.

4.3.6 Discussion

In the previous sections, based on a thorough study of the literature, we have designed a framework to integrate IDPRS solutions in critical contexts such as the control networks of CIs. As

seen in Figure 4.4, our methodological framework is composed of three main modules, each of them corresponding to the main requirements needed to be addressed in this type of networks: *detection*, *automation* and *response*. The detection module provides the system with insight about the behavior of the system and sends alerts in the case of occurring threats or anomalies. These processes can be performed by an IDS and, optionally, an alert management component [229].

The automation module (see Section 4.3.2) compiles the different levels of automation that can be present in an IDPRS solution [169]. The level of automation selected for the infrastructure will determine the components integrated in the third module, the RS. And, in the event that the ICS or CPCS implements active response mechanisms, the methodological framework in its RS module categorizes the different characteristics that it can implement. This third module of our framework is adapted from the work of N. Stakhanova et al. [4] and A. Shameli-Sendi et al. [5].

The approaches of both N. Stakhanova et al. [4] and A. Shameli-Sendi et al. [5] present taxonomies of RS designed for general-purpose networks. It has been our task to carefully analyze these sub-components and properties in order to establish whether or not they can be used in a critical environment. From Section 4.3.3 to Section 4.3.3 we have presented our assessment and made recommendations on their use according to the characteristics of the networks where the different RSs can be deployed. We have therefore studied the general-purpose RS components present in these taxonomies, contrasted them with the literature on ICS protection systems, examined their constraints and scopes, and reflected in our framework the different possibilities for integrating IDPRS solutions in critical environments.

The methodological framework, together with our assessment and recommendations, integrates the most important components that should be included when designing a IDPRS solution for CSs. The main objective of this framework is to create a framework focused on preparedness and response, to guide the design and development of IDPRS for critical infrastructures. Some of the components are not present in current ICS-dedicated IDPRS, however they can help address the challenges of prevention and protection set by the institutions around the globe [42, 112, 15, 16, 36].

Once the different components that can be present in the RS have been analyzed, we examine the possible countermeasures that the system can implement. In order to do this, we have reviewed the academic literature in search of different types of protective mechanisms. We did not restrict our search to ICS-related protection, but rather to different types of networks, and we have tried to assess whether these measures can help protect critical environments. We discuss some of the academic non-proprietary methods of protection, providing examples of response actions in Table 4.3. We have divided them into passive and active solutions.

Passive methods, such as security policies, cryptographic mechanisms, etc., are put in place to prevent malicious attacks and anomalies within the system, addressing the need to enhance the *preparedness and prevention* in CIs, one of the five pillars of CIP identified by the European Network and Information Security Agency (ENISA) [112, 15, 16]. Active methods, such as dynamic routing techniques, are those deployed to address the need for more efficient early warning and response capabilities in the CIs, corresponding to the second pillar of CIP identified by ENISA, *detection and response* [16]. Therefore, and given these analyses on IDPRS constraints and countermeasure actions, it is possible to reach several conclusions and recommendations about

the inclusion of IDPRS solutions in critical contexts:

- An IDPRS solution deployed in a critical context should implement both passive and active defense mechanisms, to be able to protect the surveilled infrastructure by providing early detection and reaction measures, but avoid any interference with its correct and normal operation. Most CIs are equipped with some type of IDPRS solution that mainly provides passive protection, however, both institutions [42, 112, 15, 16] and researchers [3, 237] report the need to have active response solutions in order to provide early detection and prevention mechanisms.

- Our review of the existing systems in the literature show that, to the best of our knowledge, in the public domain there are currently no specific IDPRS solutions that implement automatic active response mechanisms for the protection of CIs. However, through the revision of more recent literature on IDPRS solutions for general-purpose networks, and through the analysis of the countermeasures taxonomy, it is possible to evaluate the advantages and disadvantages of the different implementations of their components, and determine the capabilities that they offer, and constraints they pose for CIs.

- Whenever the resources of the infrastructure are sufficiently powerful to allow the necessary computations, the decisions taken by the RS should be balanced, taking into account the costs of the anomalies and the costs of the response actions applied in each case. According to N. Stakhanova et al. [4], the ideal cost model should be evaluated dynamically. Due to a better accuracy achieved using dynamic evaluated cost models, the performance of the IDPRS increases and is able to select the optimal responses to threats, taking into account the interdependencies and criticality of the affected components of the system.

- It is generally better to provide proactive responses instead of delayed mechanisms, since the last need to wait until the intrusion or the anomaly is finally confirmed to trigger the reaction. In the case of a cyber attack, this postponement of the response gives the adversary time to start and maybe complete malicious actions, and the system is susceptible to deteriorating fast and unleashing harmful cascading effects. In the case of an anomaly, the continuation of the normal operation of the system in the presence of faults may make the errors cascade through the interdependent systems before the countermeasures take place. Contrarily, proactive mechanisms could detect changes in the dynamics of the system earlier and send an alert or try to correct them from an early stage. However, proactive responses for CIP must always be evaluated (taking into account costs and risks of the countermeasures) and preferably supervised by human operators.

- The response selection method implemented by the IDPRS is also important, as we consider that dynamic mapping and cost-sensitive mapping methods provide higher benefits to the IDPRS than the static mapping-based solutions. Dynamic mapping systems take into account metrics and policies to select adequate countermeasures, while cost-sensitive mapping systems consider the costs of the attack and the response to make better choices in the selection of the reaction triggered. Thus, these two mechanisms are more likely to perform better than a static selection procedure, and make the IDPRS more effective and safe for CIP.

- Risk assessment methods can help assess the costs and implications of the attacks and anomalies occurring within the system. An IDPRS including a risk assessment component

can determine the impact of a given threat, but also the impact of the countermeasure actions that are applied to palliate the problem. This module can introduce feedback mechanisms into the RS, in order to execute the countermeasures gradually, preventing the responses applied from generating undesired effects. According to our analysis, and if the system has sufficient capabilities, it is better to provide the IDPRS with a dynamic online risk assessment component, since it will provide real time evaluation of the system's risk status.

- Our taxonomy of response actions (see Table 4.3), provides a general vision of the main passive and active protection mechanisms for critical contexts that can be implemented in an IDPRS. However, it is difficult to provide insight about which of the techniques perform better on a generic critical scenario, particularly as the active reaction mechanisms are usually not present in such scenarios. Nevertheless, we provide a categorization of the different techniques taking into account their nature and where are they applied, which will allow a further study of their applicability within CIs.

In light of these facts, it is possible to identify the needs that current detection and prediction solutions have in the context of critical systems. As we have said, according to the institutions around the globe [42, 112, 15, 16], it is necessary to tackle the issues of preparedness and prevention, and detection and response for the protection of the critical infrastructures. One of the technologies that help protect ICSs are the IDPRS solutions, since these tools serve as elements of detection and preparedness, and when implementing effective RSs, they act as prevention and response components.

Thus, we believe that further research is needed in order to develop efficient and cost-effective IDPRS solutions capable of automatic active reactions for CIP. Through the literature on general-purpose IDPRS, it is possible to examine the components and features usually present in these systems and determine the ones that have the best fit for critical contexts. Through the analysis of the internal IDPRS features and components we gain insight about their capabilities and requirements (computational complexity, QoS demands, etc.), thereby identifying the ones that would behave better in a constrained environment.

Firstly, we find that there is a great need to include dynamic adaptive online evaluation of costs in the IDPRS for ICS. There have been approaches addressing this topic for general-purpose networks [199, 227, 228], however in the field of CIP, there are few systems that implement such mechanisms. It would be especially useful to channel this evaluation of costs through a risk assessment component in charge of determining the impact of threats and responses in ICS [203]. Additionally, these feedback mechanisms would positively affect the capability of IDPRS to implement retroactive response execution procedures.

Few of the general-purpose RSs in the literature implement retroactive execution, something which would be very useful, particularly in critical contexts. As discussed above, critical environments are very sensitive to changes in dynamics and configurations, thus the countermeasures must be applied most carefully. Retroactive feedback mechanisms would therefore be especially useful in this context, because they would allow the system to always measure the risk levels and evaluate the system's state before launching the responses. Therefore, research efforts should be put in place in order to develop effective risk and cost evaluation mechanisms capable of performing correctly in a critical environment.

In the same vein, it is important to consider improving proactive RSs. These systems should be capable of detecting subtle changes in the behavior patterns of the system, and detect and identify threatening dynamics for the ICS early on. Once these early detection mechanisms have been deployed and tuned to the system, they can trigger actions from the RS capable of stopping attacks and palliate system anomalies in order to avoid errors and threats to cascade through the infrastructure to other interdependent systems. The analysis and prevention of cascading (domino) effects is currently a very active area of research in the field of CIP, much effort is being made to tackle this problem [238, 239, 240].

It is our belief that systems such as IDPRS which provide the main characteristics mentioned above would be highly useful in critical contexts. Capabilities such as dynamic online risk and cost evaluation would make the control systems sufficiently trustworthy for ICS operators to transfer some of their functions and supervision tasks to the automated management of the infrastructure. Note that most critical countermeasures should be always supervised by human operators, however, in the event of a semi-supervised control system implementing automatic countermeasure responses, the IDPRS could always report its cost and risk analysis to the operator to determine the best actions to take. It is our aim that this study will therefore be useful as a guide to future design and development of IDPRS solutions specifically created to protect CIs.

4.4 Restoration

Recovery mechanisms comprise all those actions related to resilience and fault-tolerance that help the underlying system to return its natural state and operating configuration [241, 242]. Replication-based techniques are one of the most popular fault-tolerance techniques by replicating functionalities and adding redundancy within the system. At this point, it is important the type of data consistency (linearizability, sequential and casual) and the replication mode, active or passive. The passive replication aims to activate the backup systems only when needed, in which primary devices are the only ones that can manage replicas. In contrast, an active replication consists in constantly replicating evidence and configurations of the primary entity so as to preserve assets (e.g., data, links) and maintain updated the backup elements at all times.

Although the active mode is also characterized by individually managing evidence in multicast mode, favoring the response, the redundancy management tends to increase complexities. For example, when replicas need to be compared to identify Byzantine faults, a voting process in distributed networks is normally required to manage consensus according to events detected. *Process Level Redundancy* (PLR) is another example of active replication. It is in charge of detecting transient faults, which are less severe than Byzantines but hard to diagnose. For the detection, their algorithms demand software-centric approaches for the detection of transient faults, resources to reduce overhead and redundant processes to schedule the processes across all assets of the system.

Rollback is another well-known recovery approach. It includes a checkpoint-based rollback with dependency on storage points containing current information, and a log-based rollback (also known as message logging protocol) composed of checkpoints and a record of non-deterministic events. Within the checkpoint class, it is possible to find coordinated and uncoordinated ap-

proaches. The former centers on synchronizing checkpoints to restrict the rollback propagation, but hampering the recovery time and their functionality in critical contexts. Uncoordinated checkpoints, to the contrary, aim to individually execute checkpoints to later combine them with a message logging protocol, thereby guaranteeing a complete picture of the execution of a process. According to Treaster in [241], there are three main log-based techniques: pessimistic/synchronous, optimistic/asynchronous and causal. Pessimistic logging techniques consist in registering each message received by an entity to subsequently be re-sent, and only if necessary, during the rollback stages; whereas optimistic protocols focus on registering events to a volatile memory to later (but periodically) store them in disk. Although this last protocol can simplify storage complexities, since it avoids for each received event the writing in disk, the recovery process can become much more complex whether the logs have not been stored properly over time. Finally, causal protocols log non-deterministic events in a casual manner, but they add the problems of the optimistic protocols in which events temporarily registered may be lost unexpectedly.

Table 4.4: Classification and characteristics of the restoration techniques

			Low Complexity	Capacity for storage	Performance	Consistency	Accuracy	Responsiveness	Adaptability	High rate of redundancy	Configurability	Handles N-faults	Handles Byzantine faults	Handles transient faults
Replication-based	Active	In general	x	✔	∗	✔	✔	✔	✔	✔		✔		
		Voting	x	✔	∗	✔	✔	✔	✔	✔		✔	✔	
		PLR		✔	∗	✔	✔	x	x	✔	x			✔
	Passive			✔	∗	✔		x	✔	x		✔		
Rollback-based	Checkpoints	Coordinated	x	✔	∗	✔		x		✔		✔		
		Uncoordinated	x	✔	∗			x		x		✔		
		Replication	x	✔	∗	✔	✔	✔		✔		✔		
	Message logging	Pessimistic	x	✔	∗			✔	✔			✔		
		Optimistic	x	✔	∗		x	x	x			✔		
		Casual	x	✔	∗	✔	x	x	x			✔		
Fusion-based			x	✔	∗	✔				x		✔		

As the checkpoints are in general terms costly, experts [243, 244] recommend applying heterogeneous replication-based checkpoints to enhance performance and guarantee tamper-resistance to faults. But their implementation can bring about complexities in the recovery processes due to redundancy; a characteristic that has also been considered by the fusion-based techniques. These techniques address the problem by relying on fewer backup devices as fusion points, instead of actively configuring replication-based approaches in all the devices. Nonetheless, this characteristic also tends to increase implementation costs and complexities for recovery by maintaining updated the fusion points.

Table 4.4 encompasses all the aforementioned properties, which are also sustained by Bansal et al. in [242]. These authors stress that the performance of each approach (denoted with ∗ in Table 4.4) depends on a set of factors. For example, replication-based schemes vary according to the number of replicas produced within the system (the performance decreases as number of replicas increases); checkpoints and rollback depend on the frequency and size of the checkpoints; fusion-based on the rate of faults (low rate of faults improves the recovery); and PRL on the

set of faults. If in addition, the approaches are equipped to incorporate multiple fault detectors, the level of reliability, accuracy and adaptation can become quite noteworthy, thereby favoring their use for cyber-physical contexts.

4.5 Context-Awareness and Situational Awareness in CPCS

In the previous sections we have reviewed the main approaches and characteristics of the protection solutions for critical infrastructures, and their control systems in particular. We have reviewed the basic security services, as well as the advanced solutions for protection which include prevention and detection, awareness and reaction, and restoration mechanisms. We now study the applicability of these solutions to our SG scenario (see Section 1.1). Smart grid systems contain seven chief domains: control systems, energy (production, transmission and distribution) substations, providers, market and end-users. In this context, the operational tasks related to monitoring, supervision and data acquisition become fundamental to the correct use of the power grid. This also means that any anomaly in the system, computational overheads or a misunderstanding of the situation can trigger a (slight or serious) change in the control of the entire grid, and therefore cause an undesirable or contrary effect in its stability.

Since the correct operation of the control systems of the SG is highly important for the integrity and general operation and performance of the entire grid, we focus on the analysis of applicability of the aforementioned context and situation awareness strategies, techniques and practices for the CPCSs of the SG, in the frame of the Industry 4.0. The strong focus of our analysis is centered on the prevention and detection mechanisms (see Section 4.2), since as we have previously mentioned, CIs do not usually implement awareness and reaction mechanisms yet, therefore, we can only speculate about the applicability of these mechanisms in our scenario.

We therefore explore in this section those approaches that can be found in the current literature, so as to evaluate their functionalities and applicability in the context of SGs, paying particular attention to those conditions that entail a degradation in control. These conditions are related to a set of requirements associated with the monitoring and security of the entire SG (see Chapter 3), and the natural conditions of the communication infrastructures. The result of this investigation gathering all the knowledge presented in previous sections, is a guideline to which approaches are most suitable for each section of the grid control systems.

4.5.1 General Architecture and Technologies

In order to better understand the context and situational awareness mechanisms better suited for the protection of CPCSs within the SG infrastructure, we need to comprehend the communication infrastructures and technologies that comprise the control infrastructure of a smart grid [245]. This was partially introduced in Section 1.1.2, however in this section we expand this knowledge with the particular characteristics of the SG.

The central architecture of the SG control systems corresponds to a decentralized control network capable of remotely communicating with the rest of the sub-domains of the SG, e.g., substations. At this point, smart meters, gateways, remote terminal units, sensors and a set of control objects interact with each other through large and small communication infrastructures such

as backhaul, *Wide Area*, *Field Area*, *Neighborhood Area*, and *Local Area Networks* (WANs, FANs, NANs and LANs, respectively). All these infrastructures base their communications on wired and wireless systems such as mobile cellular technology, satellite, WiMAX, power line communications, microwaves systems, optical fiber, Bluetooth, Wi-Fi, WSNs, Ethernet, and so on. These infrastructures are in charge of distributing monitored evidence (e.g., commands, measurements or alarms) occurring at any point of the SG system, where backhaul and the Internet are the chief infrastructures that connect the different sub-domains with the rest of the networks, including *Advanced Metering Infrastructures* (AMIs). An AMI is a bidirectional interface with the capability to manage and interact with smart meters and utility business systems, thus substituting the traditional one-way advanced meters.

This interconnection map primarily focuses on the secure monitoring of services and the effectiveness of energy production according to the real demand. These services mainly addresses the means of notifying electricity pricing at any time and provide the end-users with customizable services to efficiently manage energy consumption. Continuing with the topic of monitoring services, the control transactions between the control system and substations are led by communication interfaces (e.g., RTUs, gateways, servers, etc.) which serve as intermediary nodes between the remote substation and the *Master Terminal Units* (MTUs) of the central system. An RTU is a device working at $\sim$ 22MHz-200MHz with 256 bytes-64MB RAM, 8KB-32MB flash memory and 16KB-256KB EEPROM. These hardware and software capabilities are enough to compute data streams, operate mathematical formulations and identify those sensors or actuators in charge of executing a specific action in the field. These interfaces are also able to establish connections with other substations, allowing an inter-RTU communication with the ability to ensure store-and-forward using one of the two existing communication modes, serial (e.g., IEC-101) or TCP/IP (e.g., Modbus/TCP).

Control objects can be classified according to the type of micro-controller (weak, normal, and heavy-duty), the type of radio transceiver (wideband and narrowband radios) and the type of communication (synchronous/asynchronous) [18]. Within the category weak, we find those limited devices such as home-appliances and sensors with extremely constrained capabilities such as $\sim$ 4MHz, 1KB RAM and 4KB-16KB ROM, but with sufficient capacity to execute simple applications. Conversely, those classed as normal are those nodes that are able to comply with any kind of collaborative network. A node belonging to this category usually has a micro-controller of $\sim$ 4MHz-8MHz, 4KB-16KB RAM and 48KB- 256KB ROM. Finally nodes belonging to the heavy-duty category are expensive devices (e.g., handled devices) that are able to execute any simple or complex critical application. Their microprocessors are quite powerful working at around 13MHz-180MHz, 256KB-512KB RAM and 4MB-32MB ROM. With respect to transceivers, most of the sensory devices follow the IEEE-802.15.4 standard working with wideband radios (e.g., CC2420) at frequencies of 2.4GHz with certain demand restrictions in power. The narrowband radio-based transceivers (e.g., CC1000/C1020), to the contrary, work at lower frequencies and are more susceptible to noise, but they have less power consumption and faster wake up times.

Within the heavy-duty class, we highlight the industrial WSNs which are normally deployed close to critical systems (e.g., generators, transformers, pylons, etc.). Their capabilities are slightly greater than the conventional ones equipped with a $\sim$ 4MHz-32MHz micro-processor, 8KB-128KB RAM, 128KB-192KB ROM, and specific sensors to measure physical events associated with the industrial context such as temperature, voltage load, etc. They have the

possibility to be directly linked to energy suppliers or industrial equipment in order to maximize their lifetime with self-capacity for processing and transmitting measurements to a base station (e.g., a gateway or an RTU). With similar features, smart meters can become heavy-duty devices working at ∼ 8-50MHz, 4KB-32KB RAM and 32-512KB flash memory. An electrical meter is a device capable of logging the consumption values in synchronous and frequent intervals, sending this information back to the control utility for monitoring and billing purposes. Depending on the type of network, the communication can also vary [245]. For example, in power generation, transmission and distribution substations the communication can depend on specific or property protocols such as IEC-61850, Modbus, Zigbee, WirelessHART, ISA100.11a, and so on. Many of these offer technical solutions for customizing and optimizing the conditions of the application context in order to improve its quality of service, or avoid, for example, industrial noise, interferences or obstacles.

4.5.2 Context and Situation Awareness Approaches

Given that the great majority of the control objects (e.g., sensors, smart meters, etc.) are distributed over large-scale distributions where the control generally relies on only a few (or perhaps none) human operators in the field, topics related to dynamic and reliable context-awareness solutions should therefore be considered. Specifically, we explore a set of existing anomaly-based techniques as a support to these solutions. But as the number of techniques is significant within the current literature, we also particularize and stress here the main requirements and conditions described in Chapter 3 for the protection solutions (see Section 3.6), to ensure a better defense of the critical systems contained within a smart grid.

The concept of *context* was introduced by A. Dey in [246] as "*any information that can be used to characterize the situation of an entity*", where entity can be a person, place or object. This characterization is widely used by dynamic context-aware computing systems to detect, prevent and alert to unforeseen changes in the normal behavior of the system being observed [3]. An example of these detection systems are the IDSs, the configurations of which should respect the intrinsic requirements of the control (see Section 3.1) not to perturb the normal behavior of the entire grid. These requirements are as follows:

- *Performance* (*[R1]*) is part of the control of a SG. This includes the availability in-real time of assets and data from anywhere, at any time and in anyway; in addition to ensuring a fast supervision, data acquisition and response, avoiding communication and computational delays as much as possible.
- *Reliability and integrity* in the control (*[R2]*). Any change in the system can cause serious deviations in the power production and distribution, putting the stability of the power grid at risk.
- *Resilience* (*[R3]*) to address anomalies or unexpected incidents, which might also come from intrusive actions. Likewise, aspects related to *security* (*[R4]*) at the different levels of the communication and architecture must therefore also be considered as primary requirements for resilience.

Working within these requirements, the IDSs need to ensure a set of conditions to guarantee a fast, integral and reliable monitoring of evidence. That is to say, detectors need to show their

potential to quickly find pattern sequences that prove the existence of a deviation within a set of data instances; i.e.:

- Low computational complexity through optimized algorithms and handling of parameters, in addition to guaranteeing a speedy classification, learning and comprehensibility of the data instances. In this way, it is possible to meet the operational requirement (*[R1]*).

- Reliability through accurate detection with a low false positive/negative rate, comprehensibility of the results obtained, easiness to handle parameters, and tolerance to highly interdependent data, noise, missing values or redundancy. The idea is to offer the best way of understanding a situation so as to act accordingly (*[R2]*).

- Capacities for incremental learning to update the knowledge of the system with new (discrete/continuous) values, states, threat patterns or parameters. This will permit the underlying infrastructures to provide an updated protection layer for survivability (security and resilience against future threats, *[R3, R4]*).

- Ability to control drastic or persisting changes in the normal behavior of the system, as these deviations can mean the proximity or existence of intrusive actions, affecting *[R3, R4]*.

Taking into account the general architecture and technologies present in this SG scenario (see Section 4.5.1), the subset of essential requirements for the protection systems we consider in this section, and the different methods and techniques for detection studied in Section 4.2.1, we can better understand the approaches and solutions applicable for achieving a context and situational awareness status in the SG control networks, their function, and application area, as summarized in Table 4.5.

Table 4.5: Protection approaches applied in smart grid environments

Technique	*Ref.*	*Application*	*Application area*
ANNs	[247]	Fault diagnosis	Substations
Decision Trees	[248]	Intrusion detection	Control and substations
	[249]	Fault detection	Substations
BNs	[250]	Intrusion detection	HANs
Naïve Bayes	[251]	Islanding detection	Power systems
SVMs	[252]	Fault detection and classification	Transmission lines
	[253]	Intrusion detection	HANs, NANs, WANs
Rules	[254]	Intrusion detection	WANs, NANs and HANs
Statistical	[255]	False-data injection detection (Markov graph-based)	Control and substations
	[256]	Load/Price Forecasting, Demand (time series)	HANs, Control and substations
Fuzzy logic	[257]	Diagnosis and maintenance	Substations
	[258]	Optimization for power storage	Microgrid networks
Petri Nets	[259]	Fault diagnosis	Distribution substations
Examples of combined solutions			
ANNs and Rules	[260]	Fault diagnosis	Control and substations
BNs on an Expert System	[261]	Fault diagnosis	Substations (distribution feeder)
Fuzzy logic and Decision Trees	[262]	Islanding detection	Substations

4.5.3 Suitability of Detection Approaches for Smart Grid Domains

In this section, we analyze the suitability of the context and situational awareness techniques previously described to be applied in different particular areas of the SG. To do this, we compare the functional benefits of each of the schemes reviewed in Section 4.2.1, compared to the control and security requirements and conditions, and the characteristics of the communication systems (dimension, traffic and capabilities of the network devices). To illustrate this analysis, Table 4.6 provides a summary containing the different variables considered for the study.

Utilities: Control Centres and Corporate Networks

Control and corporate networks of a smart grid may range from large distributions with potentially thousands of nodes (e.g., servers) with connections to backhauls, WANs or NANs, to small and local networks. Depending on the type of domain and utility, they may have different kinds of protocols and topologies to connect different networks (e.g., control and AMI, providers and AMI). However, this interconnection mode and its relation to public networks, like the Internet, forces us to consider heavy-duty IDSs that help detect potential (anonymous, unknown, concurrent or stealthy) threats, and thereby comply with the minimum security requirements *[R3, R4]*. As part of this information belongs to users or the business itself, and the other part corresponds with control transactions for the protection and stability of the entire power grid, topics related to reliability of the data itself *[R2]* should also be considered. Therefore, and observing Table 4.6, the most suitable techniques for this section of the grid could be:

- *Knowledge-based* methods: the dynamic features of the knowledge-based approaches, such as expert systems, make them be one of the most attractive approaches to be applied in complex and dynamic contexts. However, this protection will highly depend on the degree of granularity of their knowledge and the frequency to with which this knowledge is updated; two conditions that should be well-specified in the security policies.
- *Statistics*: statistical-based techniques, as described in Section 4.2.1 are powerful methods that can be adapted to different scenarios, from simple to complex and dynamic contexts, and serve as anomaly-detection engines in multiple IDSs in the literature [176, 179]. Statistical methods could be useful for detection at any level of a communication network because of their great accuracy (except the operational models) despite being computationally complex. Note that *Markov models* may also be considered due to their inherent characteristics, but their transaction matrices should be well-fixed to control drastic changes. Specifically, HMMs are useful tools for detecting hidden dynamics and extracting knowledge when there are gaps in the information received. Thus in the presence of encrypted traffic, their use would be useful to detect certain hidden evidence (*[R3, R4]*).

As mentioned, there are other methods that could equally be applicable to theses types of networks, e.g., rule learners, SVMs, Markov models or clustering techniques. However they could be more difficult to adapt to the scenario, or present more challenges and inconveniences than benefits due to their inherent characteristics. For example, these methods tend to produce overfitting, a characteristic that makes them inappropriate when the environment is continuously adapting new dynamics and new constraints (e.g., frequent upgrades and maintenances). As for the detection modes, the use of a *signature-based* IDS seems to be a good option since utility

networks might apply existing and complex databases with diverse types of signatures defining threat patterns or known undesirable dynamics related to the network. The main problem found in this detection mode is the need to keep the threat databases up-to-date.

Substations: Production, Transmission and Distribution

The communication between the control centre and the remote nodes (i.e., RTU/gateway) is done through MTU, where the data traffic between the MTU-RTU /gateway is generally regular and standardized, and operational performance ([*R1*]), reliability in the control transactions ([*R2*]) and security ([*R3, R4*]) are all required. As mentioned in Section 4.5.1, RTUs are powerful enough to be able to execute a set of operations or instructions, as well as, advanced algorithms such as machine learning ones. Their hardware capacities also allows them to run specialized detection techniques capable of detecting sophisticated threats in an environment that has a regular behavior with a monotonous activity (note that this consideration is dependent on the security policies). Assuming that the communications are configured to be synchronous with regular traffic, the most suitable techniques for this section of the grid would be those related to *knowledge*. However, and as described above, the implementation of knowledge-based systems also depends on the functional features of the interfaces and the maintenance of these intelligent systems. As an alternative, it is also possible to choose those approaches that do not infringe, at least, [*R1, R2*] to ensure control at all times, such as:

- *Rule-based* techniques: this method is characterized by its simplicity (*[R1]*) and accuracy (*[R2]*), which should not degrade the main conditions for control. For the effectiveness of the approach and its use for protection, it is necessary to specify in detail, the rules, exposing all the possible threat scenarios that can arise in the connectivities between the MTU and the substations.
- *Support vector machines*: this method is powerful and well-suited to dealing with large numbers of features. SVMs are accurate (*[R2]*) and they have low complexity models (*[R1]*) [177]. However they present problems in the speed of the learning process, a handicap that makes SVMs difficult to implement in networks with constrained resources, and particularly in the presence of dynamic scenarios. Nevertheless, this can be easily overcome in networks with sufficiently powerful nodes (e.g., gateways) deployed in rather static scenarios, where the set of representative training instances is small.
- *Statistics*: optimized parametric or non-parametric solutions can become effective approaches for these sections of the SG, but without considering those related to operational models, since these do not guarantee the fulfillment with *[R2]*.

Table 4.6: Detection approaches in relation with the SG characteristics and requirements

	Complexity	Speed of classification	Speed of learning	Handle parameters	Comprehensibility	Accuracy	Learning from observation	Control - interdep. data	Control - missing data	Control - redundancy	Control - noise	Control - subtle changes	Control - drastic changes	Incremental learning	Unlimited networks	Limited networks	*Requirements* R1	R2	R3	R4	*Control - Corporative net.*	*Control - RTU/Gateway (Subst.)*	*RTU/Gateway - sensors (Subst.)*	*Gateway - embedded dev. (NAN)*	*Embedded devices (HAN)*
	$\pm^1$	$\pm$	$\pm$	$*\pm$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$\circ$	$\circ$	$\circ$		$\bullet^2$					$*\circ$	$\pm*\circ$	$\pm*\circ$	$\circ$	$\pm$
Classification trees	-	✓	✓	✓	✓	✓	-	✗	-	✗	-	-	-	-	✓	•	✓	✗	-	-	•	✓	-	-	✓
Regression trees	-	-	-	✓	-	-	✓	✗	✓	✗	✓	-	-	✓	✓	•	✗	✗	✓	✓	•	•	•	•	-
Association rule learners	✗	-	✓	✗	✓	-	✓	✓	-	✓	✗	-	-	✗	✓	✗	✗	•	✗	✗	•	•	•	•	-
Clustering	✓	✗	✗	✗	✓	✓	✓	✗	✓	✓	✗	-	-	✓	✓	•	•	✓	✓	✓	•	•	•	•	•
Parametric and non-parametric	✓	-	✗	✗	✗	✓	-	-	✓	-	✓	-	-	-	✓	✓	✓	✗	•	•	✓	-	•	-	•
Operational models	✓	-	-	✗	✗	✗	-	-	-	-	-	✗	-	-	✓	•	✓	✗	•	✗	-	✓	-	-	•
Smoothing	✗	-	-	✗	✗	✓		-	✓	-	✓	-	✗	-	✗	✓	✗	✓	✗	•	✓	✗	✓	-	✓
Markov chains	✗	-	✗	✗	✗	✓	✓	-	-	-	✗	-	-	-	✗	-	✗	✗	✗	✗	•	✓	✗	-	-
Artificial neural networks	✗	✓	✗	✗	✗	✓	✓	✓	✗	✗	✗	-	-	-	✓	✓	✗	✗	✗	-	•	✓	✗	-	-
Bayesian networks	✗	✓	-	✓	✓	-	✓	-	✓	✗	✓	-	-	-	✓	✓	✓	✓	•	-	•	✓	✗	-	-
Naïve Bayes networks	-	✓	✓	✓	✓	✗	✓	✗	✓	✗	-	-	-	✓	✓	•	✓	✗	•	•	•	✓	-	•	-
Support vector machines	✓	✓	✓	✗	✗	✓	✓	✓	-	✓	✗	-	-	✗	✓	•	✓	•	✗	✗	•	✓	✓	-	•
Rule-based techniques	✓	-	-	✓	✓	✓		-	-	-	✗	-	-	✗	✓	•	✓	✓	✗	✗	•	✓	✓	-	✓
Rule learners	-	✓	✗	✓	✓	✗	✓	✗	✗	✗	✗	-	-	✗	✓	-	•	✗	✗	✗	•	-	-	-	-
Nearest neighbor	-	✗	✓	✓	✗	✓	✓	✗	✓	✗	✗	-	-	✗	✓	-	✓	✗	•	•	-	-	-	•	-
Fuzzy logic	✓	✓	✓	✓	✓	✗	✓	-	✓	-	✓	-	-	✓	✓	✓	✓	✗	-	-	•	-	-	-	✓
Genetic algorithm	✗	-	-	✓	-	✗	-	-	-	-	-	-	-	-	✓	✗	✗	✗	✗	✗	-	-	-	-	-
Knowledge-based	✓	-	-	✓	-	✓	-	-	-	-	-	-	-	-	✓	•	•	•	•	•	✓	✓	✓	✓	-
Information-spectral theory	-	-	-	✗	✗	-	-	-	✓	✓	✗	✓	-	✓	✓	•	✓	✓	✓	✓	•	✓	✓	-	✓

Anomaly-based IDS	✓	✗	✗	✓	•	-	•	✓	✓	-	-
Signature-based IDS	✓	✓	✗	✗	-	✓	•	✓	•	•	•
Specification-based IDS	✓	✓	✓	✓	•	✓	✓	✓	✓	✓	✓

In addition, the detection methods that have problems in addressing over-fitting do not have as big an impact as the corporate networks, because the stability and periodicity of the scenario makes the classification instances very similar to the training datasets. However, signature-based IDSs configured inside RTUs can become complex since these IDSs requires big databases with known threat patterns to be kept, forcing the RTU to depend on external databases. However, *specification-based* IDS could be a good candidate since legitimate specifications of the interfaces are well known, and sometimes limited in terms of specification, favoring the definition of threat patterns according to the technical characteristics of the devices. This criteria is also applicable for constrained devices such as sensors or smart meters [250].

Another important part of a substation is the communication between RTU/gateways working in ISA100.11a/ WirelessHART (or coordinators in ZigBee) and the industrial sensor nodes. These sensors are heavy-duty devices with restrictions on executing complex operations and algorithms, and these generally maintain a regular and static traffic where their functions consist of constantly monitoring an object or an infrastructure, and sending this information to the gateway. Assuming that the communications are completely synchronous, our goal is now to find those lightweight solutions that ensure, at least, *[R1, R2]*; and in this way, do not degrade the operational activities in the field, such as:

- *Rule-based* techniques: this method, described above, is a simple approach that can be computed by constrained devices, but its effectiveness will depend on how the rules and threat scenarios are defined.
- *Support vector machines*: SVM methods, as we discussed, have good qualities to be used as detection engines for the IDSs deployed in constrained networks (favoring *[R1, R2]*). But to apply the method, it is necessary that these networks need to ensure regular and static traffic patterns to avoid triggering the learning processes with frequency.
- Optimized *statistic-based* solutions: as mentioned, these can also become quite effective for *[R1, R2]*, since their approaches present a moderate complexity and a high efficiency. However, the feasibility also depends on the optimization degree to avoid overhead costs.

The communication between industrial sensors is thoroughly analyzed in the next section because home appliances and smart meters present similar behaviors.

Neighborhood and House Areas: Metering and Control

The type of data managed in the hierarchical communications (NANs) between data aggregation point and metering devices (smart meters) and its relation to the end-users, makes the topics related to security and privacy prevail over questions of control; but this control must exist as well. Depending on the characteristics of the interfaces and assuming a constant communication, *knowledge-based* approaches can be good candidates to ensure [*R3, R4*] together with those related to the *statistics* (e.g., smoothing approaches). As regards HAN networks, the communication between embedded devices ((weak, normal or heavy-duty) sensors, smart meters and home appliances) becomes the most predominant infrastructure for the constant monitoring

[1] $\mp$ means that the property complies with *[R1]*; $*$ with *[R2]*; and $\circ$ with *[R3, R4]*.

[2] $\bullet$ states the benefits for the network device but with 'dependence' on the functional features of the approach (e.g., data structure, abilities to control noise, changes, etc.) regarding the hardware or software constraints.

and reporting of consumption evidence to smart meters. Their efficiency, however, depends on the type of energy consumption of these activities (many of these devices are very dependent on batteries), software and hardware capabilities, and even, on the type of configuration of their networks, overwhelmingly ad-hoc in nature. Therefore, the selection of techniques should primarily be focused on complying with [*R1*], such as:

- *Decision trees*, *Fuzzy logic*, *rule-based* techniques: these three approaches are in general fast and efficient learners and classifiers (*[R1]*); a set of functional features for those application scenarios built on strong restrictions and constrained devices [249]. Nonetheless, we could also consider the *operational models* for their simplicity, but always keeping in mind the need to define appropriate normality thresholds.
- Optimized *statistic* and *clustering* techniques: a well-configured simple approaches could result in a lightweight detection tool (*[R1]*). In the case of clustering, this solution would be more valid and useful in a scenario where the patterns of behavior suffer few variations, and the learning and testing mechanisms are seldom triggered.

On the other hand, ad-hoc networks could be used by human operators for local control, acquisition and management of controllers, sensors, actuators, smart meters and other related devices for control. In this regard, the control establishes a collaborative environment where human operators can directly operate in the field or in populated areas (e.g., to locally check neighborhood areas, status values of energy charging spots) without going through the control centre; thereby facilitating the execution of actions in real-time and the mobility within the area. This collaboration is generally based on very diverse kinds of technologies (e.g., PDA, cellular devices) with similar capacities to the technical specifications defined for the heavy-duty devices in Section 4.5.1, and hence, they can adopt similar approaches to those described for sensors. But due to their relativity to control ([*R2, R3, R4*]), their lightweight IDS solutions should also consider supplementary mechanisms, such as secure aggregation and reputation methods, to provide extra layer of protection and improve the detection procedures in the face of sophisticated threats. At this point, we also conclude that methods with costly training processes are less appropriate for dynamic networks regardless of the computational power of their nodes. This is because the constant changes and new dynamics constantly appearing in those networks make the IDSs trigger the learning mechanisms more frequently, and in this case they are computationally costly. However, in networks with regular and constant traffic, the training procedures are triggered only a few times, thus the use of these methods does not produce overhead excess in the system

4.5.4 Adequacy of Awareness Mechanisms of Prevention, Response and Restoration in the CPCSs of the SG

In order to assess the applicability of each of the aforementioned methods for CPCSs, it is necessary to take into account their computational features and the sensitive nature of the application context. Control systems generally demand [21]: operational performance, reliability and integrity, resilience and security, and safety-critical (for extended information see Chapter 3). But ensuring these requirements implies considering the complexities and characteristics of the different approaches, summarized in Tables 4.4 and 4.7, and their suitability to be supported by the different cyber-physical devices.

For the case of prevention, it is necessary to take into account parameters related to the complexity, the accuracy and the general performance of the models, without discarding that supervised techniques have better general performance for real-life problems than the unsupervised ones [186]. Under these conditions, we also observe that solutions such as BNs are difficult to implement successfully in a complex constrained environment, since their computational cost and complexity of implementation increases with the complexity of the system being modeled. Whenever it is necessary to evaluate the correctness of a model, or it is necessary to add a certain degree of expert knowledge, ANNs and SVMs are the more restrictive methods; e.g., SVMs have low complexity models but with problems in the speed of the learning process, which add implementation overheads. Thus, we understand that the methods that are better suited for detection and prevention in constrained scenarios are the ones based on logic, such as decision trees, optimized rule learners and fuzzy logic, and on simplified computation models such as operational or rule-based models. Concretely, decision trees have well-balanced characteristics for this specific context, while rule learners have several drawbacks (e.g., accuracy) that can be easily overcome (e.g., implementing boosting algorithms), but they provide several capabilities (performance, comprehensibility, and easiness for introducing rules by experts), that are vital and really interesting in critical contexts.

Knowledge-based systems and rule-based, optimized SVNs and statistical methods are contrarily suitable to be integrated inside heavy-duty devices because of their accuracy capacities. Depending on the regularity of the traffic, the application context and its capacity for changes, optimized parametric or non-parametric solutions, with exception to the operational models, can become effective approaches since their approaches present a moderate complexity and a high efficiency for detection. Similarly, powerful-duty devices can also adapt the knowledge-based schemes and statistics methods due to their ability to autonomously detect slight or abrupt anomalies with a high accuracy. For example, hidden Markov models are potential tools for detecting hidden dynamics and extracting knowledge when there may exist obfuscation in the traffic received. Nonetheless, many sophisticated ML-based solutions can be applicable for powerful-duty environments, however, some of them (e.g., SVN, rule learners, ANNs, etc.) have costly training processes that can result less appropriate for dynamic networks with irregular traffic. In fact, the occurrence of frequent asynchronous disturbances in these types of dynamic scenarios may also make the IDSs trigger its learning mechanisms repeatedly, increasing the computational overhead in the underlying systems.

These restrictions can be translated to the IRSs, since the applicability of given countermeasures heavily depends on the characteristics of the environment where the response is launched. Currently, IRSs designed specifically for critical systems, implement some of the response mechanisms mentioned in Section 4.3, particularly passive methods, as indicated in Section 4.2.4. However, most systems still lack important passive prevention mechanisms (e.g., authentication of the source of a request), mainly due to the use of legacy equipment, the proprietary protocols and components traditionally present in these environments [21]. Nevertheless, as observed in Table 4.7, most passive and preventive mechanisms are easy to implement (e.g., logging, security policies) and introduce little overheads in the system (it is important to note that Table 4.7 contains a subset of the response actions identified in Table 4.3 in Section 4.3.5, especially analyzed for CPCSs' protection). Thus, they are effective and adequate to be applied in CPCSs regardless of the computational power of the environment, but are especially indicated for constrained contexts (e.g., home-appliances, sensors). Active reaction solutions, however, are rarely present

Table 4.7: Classification and characteristics of the response mechanisms

Triggered Response	Type / Field of Response	Response Actions	Low Complexity	Easy to Implement	Low Use of Resources	Requires additional HW	Adaptable to Changes	Low Impact of Response	Low-risk Automation
Passive	*Administrator Notification*	Generate System Logs	✔	✔		∗	✔	✔	✔
		Generate Alarm	✔	✔	✔		✔	✔	✔
		Generate Report	✔	✔	✔		✔	✔	✔
	Prevention Measures	Cryptography	∗		∗			✔	✔
		Security Policies	✔	✔	✔			✔	✔
		Monitoring	✔			∗	✔	✔	✔
		Protective/Defensive Infrastructure	✔	✔	✔	✔		✔	✔
		Low-Level Preventive Mechanisms	✔	✔	✔	∗		✔	✔
		Session/Communication Measures		✔				✔	✔
Active	*Host - Based Response*	Operations on Files		✔	∗				
		Operations on User Accounts		✔	∗				
		Operations on Processes and Services			∗		✔		
		Trust Mechanisms			∗		✔		∗
	Network - Based Response	Disable / Block Operations		✔	✔				
		Isolation Actions	✔	✔	✔				
		Routing	✔	✔	✔		✔		
		Deceiver Devices				✔	✔	✔	∗

✔: the technique fulfills the property. ∗: some implementations of the technique can fulfill the property.

in these scenarios, since they usually entail the introduction of more sophisticated mechanisms, therefore making a higher use of computational resources and equipment. Methods that modify the behavior of the system or the network (operations on files, re-routing techniques, etc.) are only adequate for sections of the CPCSs containing powerful equipment (RTUs, servers) capable of devoting sufficient computational power to the IRS. Additionally, other methods that require extra HW or need to run powerful intelligent algorithms (e.g., deceiver devices, intelligent monitoring and response) are only applicable to powerful-duty environments of the CPCS, since they need to perform complex operations with high requirements on computational power.

Regarding restoration, it is possible to note from Table 4.4 that the vast majority of techniques present significant computational and spatial complexity since their approaches require a high rate of redundancy. Logging events in an optimistic or casual manner, specification of multicast protocols and configuration of hybrid networks, in which the handling of replicas and checkpoints could be concentrated on some heavy-duty or powerful-duty nodes, could, for example, resolve the implicit overheads of the models built on constrained environments. Also external storage systems (e.g., cloud computing, external memory) can support the data storage and the restoration phases in those nodes classified as weak. In less restrictive and distributed scenarios, where the rate of redundancy can become higher, more complex reparation mechanisms can be adapted (e.g., rollback based on checkpoints and replication, message logging based on pessimistic protocols, voting, etc.). However, the provision of lightweight and dynamic fault tolerance systems composed of adaptive models in these types of application contexts is still required. At this point, responsiveness, adaptability, accuracy and performance of the models are fundamental criteria to consider when developing methods, without forgetting the need to

Table 4.8: Adapting WASA techniques to cyber-physical environments

	Weak	Heavy-duty	Powerful-duty
Prevention	Decision trees Rule learners Operational models Knowledge-based Rule-based Fuzzy logic	Knowledge-based SVN Rule-based Statistical-based	Knowledge-based Statistical-based
Response	Administrator notification (logs, alarms, reports) Security policies Low-level prevention Cryptography	Monitoring Session/communication measures Modification operations (files, accounts, processes) Routing (isolation, blocking)	Monitoring Operations on processes and services Deceiver techniques Trust mechanisms
Restoration	Passive replication Message logging (optimistic, casual)	Active replication Checkpoint replication-based Message logging (pessimistic, optimistic)	Active replication Checkpoint replication-based Message logging (pessimistic, optimistic)

launch optimized strategies that give satisfactory average-time, with linear approximations as stated in [263].

Table 4.8 summarizes the analysis performed in this section, where we determine that it is still necessary to continue exploring new strategies that help simplify the implicit overheads in awareness and response tasks, providing more dynamic lightweight solutions where the rate of redundancy reaches minimum values, and the degree of accuracy and responsiveness reach high values. These goals should be part of future work where experts in the field of CIP should join efforts to foster the concept of WASA in all the sections that compose a CI, without degrading the existing cyber-physical interdependencies and guaranteeing a suitable tradeoff between operational performance and protection [21].

4.6 Summary

This chapter is devoted to the analysis of the security services necessary to provide a secure interoperability infrastructure for the CPCSs of the smart grid. We have first discussed the basic security services which must be implemented at each level of any critical CS in order to operate securely, i.e., *authentication*, *authorization*, and *accountability* mechanisms. These services should be implemented and used at any time by the actors in the infrastructure. Once this basic security layer is put into place, it is possible to analyze further advanced security services at three different levels (see Figure 4.1), namely *prevention and detection*, *awareness and reaction*, and *restoration*.

The first of these advanced security services focuses on the early detection and prevention of threatening events within the infrastructure. These services should be located at different sections of the control infrastructure, and also monitor the operation of the interoperability infrastructure itself in order to detect any possible dangerous dynamics in the systems and launch mitigation mechanisms in order to avoid cascading faults through the interconnected critical systems. As stated in these sections, the main tools that provide these capabilities are the IDS solutions. We defend the provision of intelligent automated monitoring solutions, capable of learning and adapting to the different dynamics of the surveilled system, whenever

the subjacent infrastructure allows the deployment of advanced security mechanisms. It is always necessary that the deployed detection and prevention solution observes and complies with the requirements and constraints of the surveilled infrastructure, and its control system (see Chapter 3).

A second approach to the security advanced services, is the inclusion of awareness and reaction mechanisms in the infrastructure. These mechanisms help establish and maintain a state of situational awareness and monitored operation of the infrastructure (the surveilled CI, the CPCS and the interoperability infrastructure), which allows the early management of the different occurring events in the system, in a measured, intelligent and automated way. There are varied possible implementations of these intelligence and automation capabilities, however, we defend the need to provide proactive, intelligent reaction solutions that help prevent threats within the infrastructure at the earliest moment possible, to avoid any damage to the delicate operation of the critical systems.

The third step in the inclusion of advanced security services for the interoperability of CPCSs in the SG is the provision of restoration solutions for the infrastructure. These restoration mechanisms are launched whenever a threatening event has caused an impact within the infrastructure (e.g., it has produced a fault in a system, degraded the performance of a subnetwork, etc.) and it is necessary to recover the system, and return its operational characteristics to the correct regular ones. There are varied techniques to implement restoration measures, however they usually imply the introduction of redundancy and backup elements within the infrastructure, in order to be able to perform the recovery tasks. Restoration must be present at every level in the architecture of a critical system, since any malfunction in the system should be remedied immediately.

Once described the main security services we believe important for the operation of a secure interoperability infrastructure of CPCSs in the SG, we have devoted the last sections of this chapter to the analysis of these different techniques and measures in the context of cyber-physical systems, and the implementation of awareness mechanisms within the CPCSs at each of the levels of the SG infrastructure. This concrete study analyzes specific techniques and technologies applicable to different domains of the SG, i.e., *utilities*, *substations*, and *NaNs*, studying at each case the characteristics and specific requirements of these subdomains, and the adequacy of the techniques for each of them. This last step brings the general study on abstract security services for the control systems to the concrete scenario of the CPCSs for the SG, providing insight about the better suited techniques and methods to protect the infrastructure. As in the previous sections, we defend the inclusion of intelligent automated mechanisms which make the surveillance system capable of achieving early detection, prevention and protection.

Chapter 5

Design of interoperable and Secure CPC Systems

This chapter is devoted to the design of a secure interoperability architecture according to the requirements and recommendations reviewed in the previous chapters. We base our design on the requirements, recommendations and techniques discussed in Chapter 3. The security practices and guidelines follow the standards and recommendations of secure practices por interoperability analyzed in Chapter 4. With these guidances in mind, we propose an architectural solution for the secure interoperability of CSs, particularly the CPCSs of the SG, in the framework of Industry 4.0 and in line with the new generation of CSs that accompanies this new industrial paradigm.

We firstly aim to describe the problem and application scenario where we focus our analysis, in an effort to bring to light the objectives of the proposed design. To better understand our model, we provide a brief description of the main technologies and guidelines applied. Sections 5.3 and 5.4 are devoted to detail the components and functionalities included in the design, while in Section 5.5 we provide an analysis of the operation of our proposed system's model, where we can understand, through a detailed description and examples, the capabilities of the infrastructure proposed.

5.1 Definition of the Application Scenario and Objectives of the Design

In this thesis, we aim to provide a solution to the problem of interoperability among critical complex control systems for the SG, belonging to different areas and different SG providers. This is a difficult task given the high complexity of the control networks due to the vast amount of nodes present and the high level of heterogeneity in these networks, where different industrial protocols, legacy and new equipment, devices from different manufacturers, coexist in these network and need to interact and work in a precise and finely tuned manner in order to ensure the service availability and continuity in the infrastructure.

We therefore try to tackle this problem by proposing a design for the secure interoperability

architecture for the CSs of the SG, framed in a transitional state towards the Industry 4.0 paradigm as described in Chapter 1, and considering the key characteristics of the new generation of control systems accompanying it, i.e., decentralization, security and interoperability. Additionally, we need to consider the requirements and guidelines for secure interoperability previously identified in Chapter 3. These requirements can be categorized into 6 groups (i.e., *CIP softgoals*, *industrial IoT softgoals*, *architecture softgoals*, *interoperability softgoals*, *cloud softgoals* and *virtualization softgoals*), containing different kinds of NFRs each. From these groups of requirements, it is possible to extract some information about the main desired characteristics that our interoperability platform should have, in order to be compatible and operate well in the context of CPCSs for the smart grid:

- *Decentralization*: is one of the pillar characteristics shaping the fourth generation of CSs. As stated in Section 3.4, decentralization allows the transference and distribution of functions over the different nodes of a network, this characteristic is highly useful in the control and balance of the workload in the interoperability infrastructure, avoiding centralized authorities subjects to threats. Decentralization can be achieved by making available to the infrastructure's devices different access points, where legitimate actors in the interoperability platform can always be authenticated and authorized to perform their duties, regardless of their scope, provider or constraints. This improves the availability and survivability of the interoperability infrastructure, and of the CIs interconnected through the platform.

- *Security*: security is one of the vital requirements for the correct operation of any CI (see Chapter 3), and one of the key requirements and characteristics that describe the new generation of CSs. This requirement is especially important when considering an interoperability scenario for different CIs. Given the high complexity and criticality of such infrastructure, it is necessary to provide strong security mechanisms that protect the dynamics occurring within the CIs, their control networks, and the interoperability infrastructure that enables their interconnection and cooperation.

- *Interoperability*: itself, is the third identified characteristic or requirement for the CSs in their fourth generation (see Section 1.1.1). Additionally, this requirement is key in the design of the architectural model we aim to propose, which is highly focused on providing interoperability and connectivity means for the different CPCSs coexisting in the infrastructure, in order to allow these systems to perform their tasks in a cooperative way.

- *Survivability and dependability*: these requirements need to be especially considered within any critical system. Survivability and dependability are key requirements identified for the CCSs of the CIs (see Section 3.1). They need to be considered with high priority whenever designing our secure interoperability platform for the CPCSs of the SG, given that they grant the CI will continue its operation even in the presence of system faults. Our secure interoperability architecture must, therefore, inherit these characteristics from the CCSs and ensure they are always observed and contemplated in the design of additional services or infrastructures.

- *Performance*: the performance of the critical systems is not only an important NFR to take into account in the design processes (see Section 3.3), but an essential constraint that need to be observed and enforced at all times when dealing with CIs (see Section 3.1.2). As reviewed, critical systems need to provide real-time performance, thus these characteristics need to be transmitted to the interoperability infrastructure used for the communications

and interoperability of CPCSs.

- *Maintainability*: maintainability is one of the requirements that descend from the dependability softgoal of the CCSs (see Section 3.1). This requirement conveys the need for the infrastructure to be continuously maintained in order to keep it free of errors, something that is vital in a critical context, since any error or fault within the system can cause service interruptions or cascade through the infrastructure affecting other interdependent systems. These cascading failures and service outages need to be avoided and kept to a minimum in order to ensure the availability and service continuity provided by the CI. This maintainability characteristic need to be inherited by the secure interoperability architecture, since it must maintain the same standards needed by the interconnected infrastructures in order to grant the correct operation of the whole system.

- *IoT capabilities*: within an interoperability architecture for the CPCSs of the SG, it is necessary to contemplate the presence of industrial sensors and IoT devices to adjust to the scenarios framed by the Industry 4.0 paradigm, where different types of sensors, CPSs and "things" automatically interact and cooperate to perform tasks within an industrial scenario. When translating this paradigm to our scenario, where we attempt to design a platform for the secure interoperability of CPCSs for the SG industry, it is necessary to consider a scenario where CPSs, cloud systems, IoT devices and other technologies interact and cooperate in a fluent way.

- *Cloud capabilities*: the use of cloud computing technologies in the scenario of CIP provides helpful tools to support control operations in critical constrained scenarios [264]. As discussed in Section 3.3, the use of cloud computing introduces within the interoperability architecture a high degree of flexibility, allowing the infrastructure to include virtualization techniques, it also enables distribution and redundancy, and it improves interoperability and accountability procedures. Since this technology belongs to the Industry 4.0 paradigm, it is especially interesting to consider it within our scenario. Another interesting technology to consider is the use of *Fog computing* capabilities, this new paradigm is an extension or variation of cloud computing, oriented to the improvement of the cloud characteristics in order to meet the demands of a more exigent scenario. Fog computing will be discussed in depth in the following sections.

- *Virtualization capabilities*: virtualized services are enabled in the interoperability platform thanks to the introduction of cloud and fog computing technologies within our scenario. Virtualized services and network functions allow the system to introduce functionalities within the interoperability infrastructure without the need to add extra dedicated and costly HW to the infrastructure. These new services and functions are placed within the cloud and fog services of the interoperability infrastructure, introducing a new degree of flexibility and desirable characteristics within our scenario (e.g, portability of services, migration, etc.) as well as resulting in a reduction of costs on HW for the infrastructure (see Section 3.3).

Therefore, taking into account this set of important characteristics for the secure interoperability platform, we can provide a first approach to the design of our architecture. A first draft of an interoperability architecture combining the discussed characteristics is provided in Figure 3.11 in Section 3.5, where we address the main requirements and satisficing techniques for interoperability in order to analyze their applicability to a real scenario through security metrics.

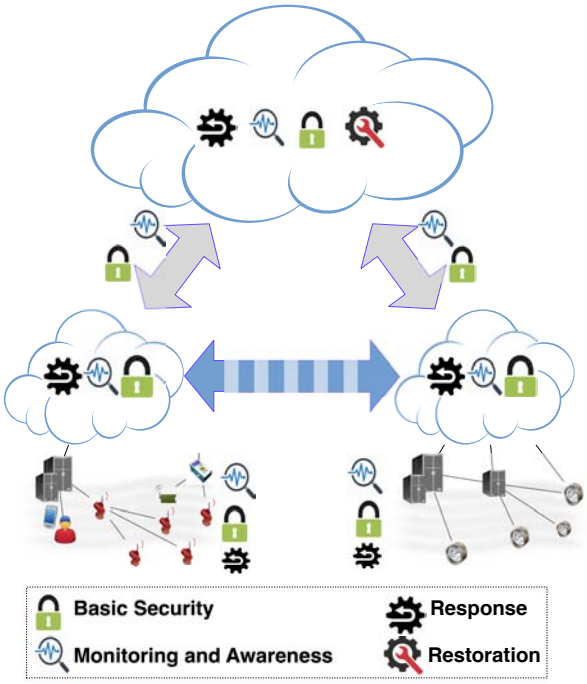


Figure 5.1: Secure interoperability conceptual infrastructure for the smart grid

This abstraction provided in Figure 3.11 can be concretized using the satisficing techniques for security recommended in Section 3.4, together with the identified set of characteristics identified in this section in order to produce a first model of the interoperability infrastructure. Another important contribution to our design comes from the discussion in Chapter 4 on the services that aid the secure interoperability in CPCSs, where we find the studied services can be categorized into four sets: *basic security*, *monitoring and awareness*,*response* and *restoration*, each of them are studied in depth in the previous chapter. Therefore, taking into account the security recommendations gathered from Chapters 3 and 4, we provide a first draft of the conceptual model for the secure interoperability architecture for the CPCSs of the SG in Figure 5.1.

In Figure 5.1, we can observe a model for a distributed interoperability architecture based on the concept of cloud and fog computing, where the CPCSs and IoT devices present in the SG infrastructures can communicate and interact with other systems belonging to other areas of influence through these technologies. It is important to note that the interoperability platform provides the basic and advanced security services discussed in Chapter 4, where we observe the presence of monitoring and awareness services, together with response and restoration mechanisms. These security services are present at different sections of the interoperability infrastructure, supporting the main requirements for the critical systems mentioned above (i.e., security, survivability, dependability, performance, maintainability).

Other considerations

In the design of our secure interoperability architecture, we therefore contemplate the requirements and recommendations for interoperability reviewed in Chapter 3, together with the secu-

rity services for the secure interoperability studied in Chapter 4. However, it is necessary to go a step further and consider the threat model faced by the interoperability infrastructure. To take this into account, we refer to the analysis of threats to the CPCSs developed in Chapter 2. As discussed, CIs are subject to many threats, from unexpected faults occurrences within the system, to sophisticated attacks targeting specific devices or subsystems in the infrastructures.

Chapter 2 discusses the threats caused by sophisticated stealthy attacks, and Section 4.5 provides a brief overview on system anomalies and situational awareness. Therefore, taking into account this consideration on the threat model, we can give further shape to the design of the secure interoperability infrastructure for the CPCSs of the SG. Additionally, it is necessary to consider the different standards, protocols and guidelines that will be interacting within the interoperability platform in order to implement into this infrastructure design their main characteristics.

The main protocols for the communications networks that can be integrated within the secure interoperability architecture (see Sections 3.5 and 4.1) are industrial protocols such as Modbus [13] (or Modbus/TCP [265]), IEC-60870-5-101 [266], IEC-60870-5-104 [7], IEC 61850 [97], DNP3 [14]; wireless industrial protocols such as Zigbee [267], WirelessHART, ISA100.11a [115].

The main standards and guidelines of good practiced reviewed in this thesis for the security of CIs (see Sections 3.5 and 4.1) are the IEC/TS 62351 series [134] (comprised of 8 parts), the NIST SP800-82 [112], the NIST SP800-53 [141], the NISTIR 7628 [110], the ISO/IEC 27000 series [268], the NIST SP800-144 [127], the NERC-CIP [113], he CCS CSC [136] and COBIT 5 [137], the NISCC Guide [133], the ISA 62443-2-1:2009 [138] and the ISA 62443-3-3:2013 [139]. And we also consider additional guidelines and standards such as the NIST SP1108 [269] and the IEEE2030 [11] on interoperability or the NFV Guides [126] on virtualization of network functions.

Additionally, it is necessary to consider, that when working with CIs, we need to deal with a high level of complexity in the resulting designs, since these infrastructures are highly interdependent among them [31]. Therefore, a system designed to connect these infrastructures and make them interoperable through a common platform infrastructure needs to take this essential characteristic into account. The resulting interoperability model needs to take this into account to provide efficient solutions that do not interfere with the communications and service provision between CIs.

5.2 Background and Technologies for the Design

In this section we tackle the analysis of the main techniques, guidelines and technologies we use in our architectural model which help us address the problem of interoperability in our complex scenario, where heterogeneous devices and technologies interact in a cooperative way to perform the control tasks in the federation of SG infrastructures, as mentioned Section 5.1.

Access Control Mechanisms

As we discussed in Section 2.4 access control is a preventive mechanism that can help the infrastructure avoid multiple types of (stealthy) attacks. There are many methods (DAC, MAC, ABAC, RBAC, etc.) however, the separation of identity from privileges and the improved dynamic management of the security roles makes RBAC [28], in our opinion, a good candidate for the management of access control in large critical environments. The the IEC/TS 62351-8 [28] standard specifies and defines the access control process in power systems. The access control process is needed enterprise-wide, covering multitude of system's users, from members of the staff to external providers, suppliers and other partners/stakeholders. RBAC follows the principle of least privilege (previously described in Section 3.4), which states that no subject should be given more rights than necessary for performing that subject's job [28].

To do this, RBAC helps create an infrastructure based on subjects, roles and rights (i.e., permissions) for the authorization process. This division is done in order to separate the most frequently changing profiles in the system, from the access control mechanisms itself, i.e., subject names change more frequently than role names, and role names change more frequently than the rights of a data model. Therefore, this separated model, together with the fact that RBAC can be implemented as distributed services, simplify the maintenance of the access control system in a wide and complex system such as the SG.

Cloud and Fog Infrastructures

According to NIST, *cloud computing* is defined as "*a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [270]. Cloud services can be provided following three different services models [270], i.e., *Software as a Service* (SaaS), *Platform as a Service* (PaaS), and *Infrastructure as a Service* (IaaS). Additionally, cloud infrastructures can be managed according to four different deployment models [270]:

- *Private cloud*, where the cloud infrastructure is owned, managed and used in exclusive by a single organization.
- *Community cloud* or federated cloud, where a community or federation of partner organizations own, manage and operate the cloud infrastructure, following shared security requirements and policies, and complying with common requirements and considerations previously agreed between the members of the federation.
- *Public cloud* is used when the cloud infrastructure is provided (owned, managed and operated) by a public cloud provider, which can be a business, academic or government organization (or a combination of them).
- *Hybrid cloud* is a composition of two or more of the above-mentioned cloud deployment models, which work cooperatively by means of common enabling technologies for the portability of data and applications.

Additional to the cloud computing, another technology interesting to introduce within our secure interoperability architecture is *Fog Computing*. As introduced in Section 5.1, the fog is a new

paradigm based on the convergence of a set of technologies coming from different areas, e.g., cloud computing, sensor networks, peer-to-peer networks, network function virtualization, or configuration management techniques, which are integrated in order to meet the demands of device ubiquity, agile network and service management and data privacy that are now surfacing with the new models and practices of today's information technologies.

More formally, fog computing is defined as "*extension of the cloud computing paradigm (that) provides computation, storage, and networking services between end devices and traditional cloud servers*" [271]. L. Vaquero and L. Rodero-Merino provide a more extensive definition: *Fog computing is a scenario where a huge number of heterogeneous (wireless and sometimes autonomous) ubiquitous and decentralized devices communicate and potentially cooperate among them and with the network to perform storage and processing tasks without the intervention of third parties. These tasks can be for supporting basic network functions or new services and applications that run in a sandboxed environment. Users leasing part of their devices to host these services get incentives for doing so* [8].

Table 5.1: Cloud vs. fog features, extracted from [8]

	Cloud	Fog
Latency	High	Low
Access	Fixed and wireless	Mainly wireless
Control	Centralized	Distributed
Service Access	Through core	At the edge / on handheld device
Availability	High	Volatile - High Redundancy
Main Content Generator	Humans	Devices / sensors
Content Generation	Central	Anywhere
Content Consumption	End devices	Anywhere
SW Virtual Infrastructure	Central servers	Edge devices

The fog computing paradigm has emerged in the recent years alongside other so called "*edge*" paradigms such as *mobile edge computing* and "*mobile cloud computing*" trying to meet the needs and requirements that the cloud cannot fulfill, e.g., low latency and jitter, context awareness, mobility support, that are crucial for some applications [271]. Table 5.1 provides a comparison between some important characteristics of the cloud and how they change in the fog paradigm. This table is an extraction from the work of L. Vaquero and L. Rodero-Merino [8], and provides interesting insight about the main differences between paradigms, and the new characteristics introduced by fog computing in latencies, control location, contents productions and consumptions, accesses, and availability.

Software Defined Networking

To implement virtualization techniques using fog computing, it is possible to make use of the *Software Defined Networking* (SDN) paradigm [272]. SDN is a new networking paradigm where the *data plane* (i.e., the actual data packets in the network) is decoupled from the *control plane*, i.e., the network control decisions; the network intelligence is located in SW-based controllers, and the physical network devices become simple packet forwarding HW, managed by the controllers via an open interface such as OpenFlow. This paradigm promises to simplify network management through network programmability, allowing the SW developers to control network resources as easily as computing and storage resources [272].

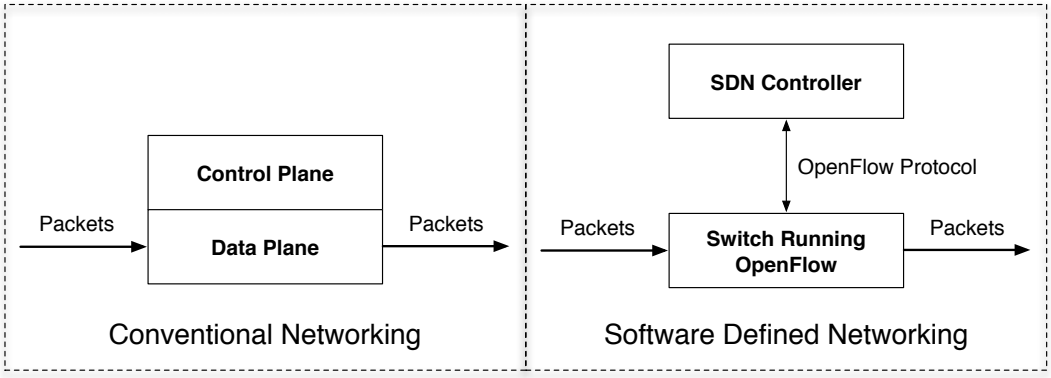


Figure 5.2: Conventional networking vs. SDN

Figure 5.2 illustrates the comparison between the schema of operation of conventional networking versus the operation of the SDN. In conventional networking, the control plane provides data to build the forwarding table, and the data plane looks into this forwarding table in order to make routing decisions. Conversely, in SDN, the switch running the open interface solution, such as the OpenFlow protocol [273] (there might be others), forwards the packets to the SDN controller for it to make control decisions, according to the SDN controller data installed as a forwarding table in the switch. The main difference between conventional networking and SDN is, therefore, the way in which data is handled and forwarded.

The OpenFlow protocol [273] enables these forwarding operations between switches and controllers, and the programming of the OpenFlow-enabled switches (or routers) tables. Generally, in the SDN architecture, we have an OpenFlow-enabled switch, generally a physical device (although it is possible to have a *virtual switch*) working with the SDN paradigm, where its SDN controller can be deployed in any device capable of supporting its computational and storage needs. It is possible to talk about a SDN controller, when the controller runs directly on HW, or about a *virtual controller* when it runs as a virtualized service in a computer or server.

A concept related to SDN is the *Network Functions Virtualization* (NFV) [126], is a network architecture concept where virtualization techniques are used to implement virtualized network functions, in order to create communications services. NFV uses standard virtualization technologies to transfer different dedicated network equipment functions onto industry standard high volume servers, switches and storage, allowing the system administrators to avoid purchasing costly dedicated networking HW. According to the documentation, NFV is "*applicable to any data plane packet processing and control plane function in fixed and mobile network infrastructure*", and is "*highly complementary to SDN*" [126]. An example of the use of NFV together with the SDN paradigm would be the usage of a *virtual switch* in charge of the data plane of SDN, instead of deploying an SDN-enabled switch, where this virtual switch is a virtualized service running in a general-purpose computer or server.

In terms of security, the use of the SDN paradigm has its own advantages as well as it introduces new kinds of threats within the system. As a security strong point, it is important to remark that SDN enhances the network monitoring and the detection of abnormal behaviors, since the SDN controller has a global perspective of the network dynamics (it receives traffic from all the

nodes in the network), additionally, the controller can launch its own response procedures, which are easily programmed and updated by the system administrators, instead of requiring updates of the operative system of the physical network device [274]. Works such as [275] illustrate the use of SDN to build interesting protective solutions.

However, the use of SDN has weaknesses from the point of view of security. Given the nature of the SDN, where instead of a networking interface, the system is separated into different planes, there are more attack entry points within the system, e.g., the SDN switch, the links between switches, the SDN controller, the links between controllers and switches, etc. Additionally, the characteristics of the SDN controller makes it especially interesting to compromise and manipulate by adversaries. This is because this module is designed to provide many open programmable interfaces to adjust perfectly the networking system to its desired operation. These available interfaces make the system open at many points to manipulation if not properly protected [274]. Common cyberattacks such as man-in-the-middle or denial of service attacks can easily take advantage of the SDN elements in the network.

Therefore, the SDN modules must always be properly protected and monitored in order to avoid introducing new vulnerabilities to threat the underlying system. Different measures can be applied to increase security and prevention in this area, such as the use of secure channels, e.g., using *Transport Layer Security* (TLS) for the OpenFlow links between controllers and switches, the use of network IDSs to monitor also the traffic around the SDN controllers taking into account the OpenFlow links, monitoring agents to manage DoS attacks, etc.

5.3 Design of the Secure Interoperability Architecture Model

Given the characteristics and requirements gathered in the previous section for the design of our secure interoperability architecture model, we devote this section to concretize the specific components needed for the secure interoperability among CPCSs of the SG. Based on the first drafts and proposals from the interoperability infrastructure provided in Section 3.5 (see Figure 3.11) and in Section 5.1 (see Figure 5.1), we propose a design for the secure interoperability of the CPCSs of different providers of the SG, which we illustrate in Figure 5.3. As we can observe in this figure, we propose an interoperability scenario for several SG infrastructures, each of them comprised of different subnetworks and remote subsystems. We present a distributed and decentralized environment which makes use of cloud and fog computing technologies and enables the inclusion of CPS and IoT devices within the infrastructure, contemplating therefore an scenario where some of the new technologies of the Industry 4.0 are represented (see Section 5.1).

Figure 5.3 presents, therefore, our approach to the solution of secure interoperability for the CPCSs of the SG. In the figure it is possible to differentiate a central module representing the *cloud infrastructure*, which will guide the interoperability between the actors in the platform at a global level. There are different smaller clouds or *cloudlets* that represent the *fog computing infrastructure* deployed at the ground level in the architecture, i.e., at each independent sector of the infrastructures, e.g., substations, remote networks, etc. Within each of the cloudlets, it is possible to find one or several *controllers*, which will serve as intelligent access points (see Figure 3.11 in Section 3.5) to the secure interoperability platform, and perform the main interoperability tasks in the infrastructure. It is also present in the figure, depicted as brown dots,

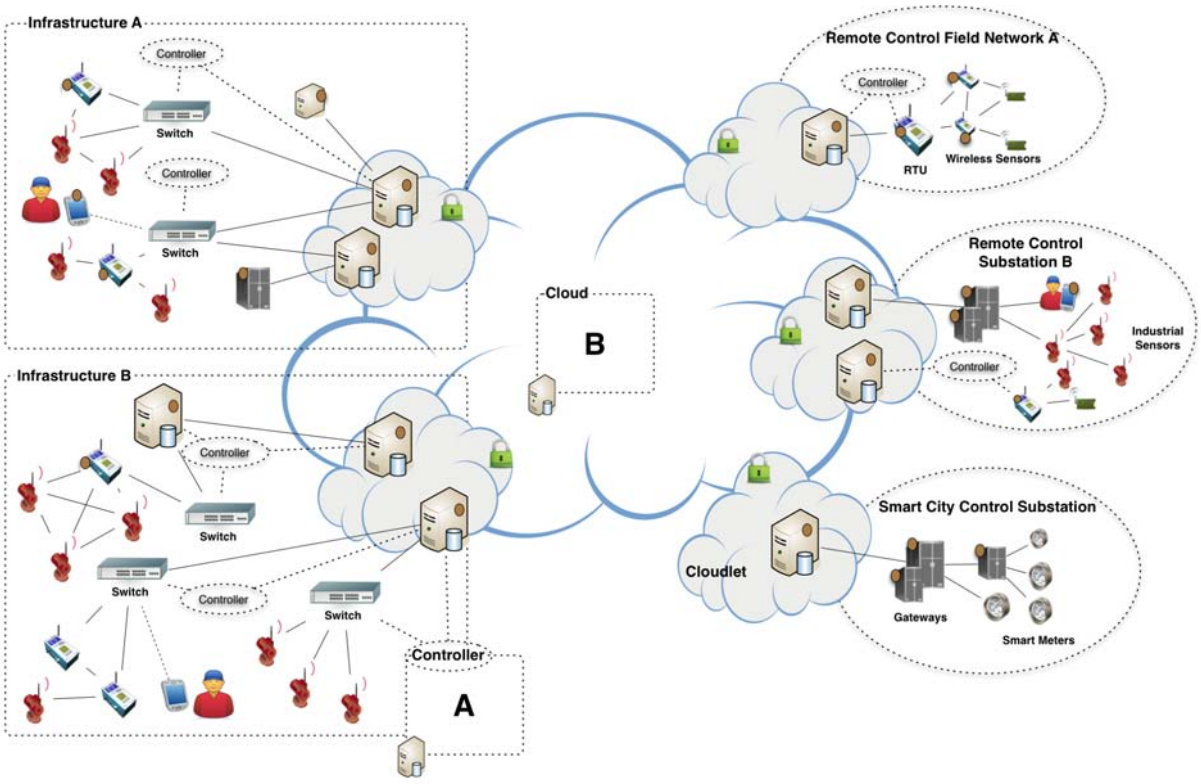


Figure 5.3: Secure interoperability infrastructure for the smart grid

a distributed diagnosis system based on software objects that gather contextual information for awareness and control tasks. And finally, we represent several types of control networks belonging to the SG infrastructures, e.g., corporate networks, remote control networks, SG substations, IoT control networks.

Within each of the controllers, we have depicted in the figure the need to specify their controllers. This is denoted with the letter A, which corresponds to the modules and services provided by each controller, as illustrated in Figure 5.4. As we can see in the figure, the functionalities offered by the controller modules are complex, and will be described in depth in the next sections. Additionally, the services provided by the cloud, denoted by module B are illustrated in a separate image, given in Figure 5.5. Moreover, there is another module illustrated within the platform represented by a green lock that refers to the security and the advanced monitoring services included within the infrastructure, which will be discussed in depth in the following sections.

The remainder of this section is devoted to the analysis and description of the main services and technologies needed for the construction of this secure interoperability architecture model, and in the next section we present and describe in depth the service modules offered by the controllers and cloud components of the infrastructure.

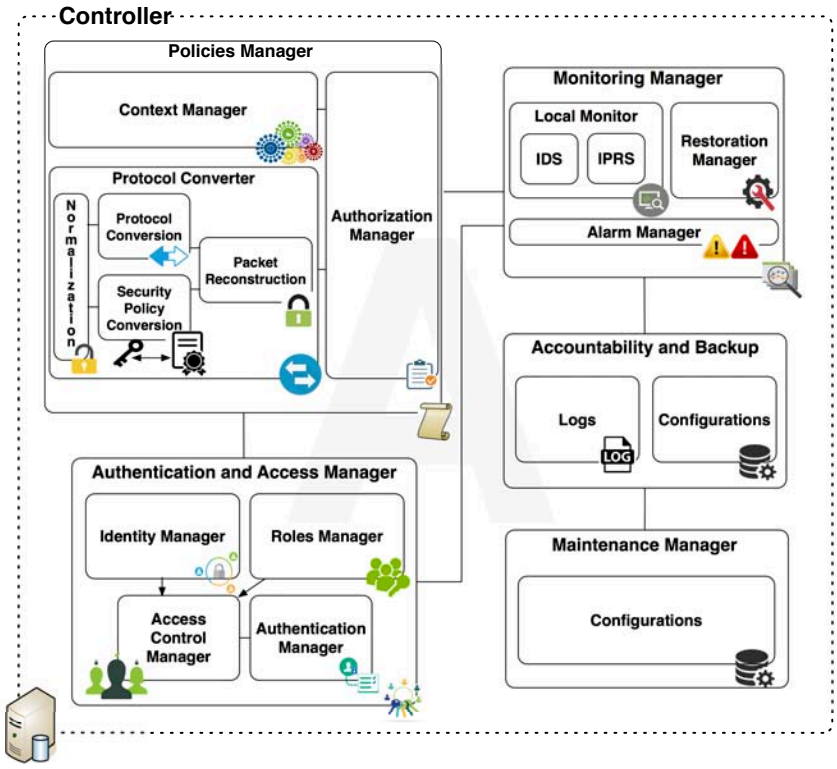


Figure 5.4: Services offered by the controller modules

5.3.1 Access Control Mechanisms

In the scenario of critical infrastructures, the critical systems must be protected at all times in order to avoid malfunctions and attacks to the system that can threaten its continued and correct operation (see Chapter 2), given that it is critical that the service provided by the infrastructure is not interrupted at any moment (see Section 3.1). Therefore it is of vital importance to provide the infrastructure with access control mechanisms (both cyber and physical measures) to avoid unauthorized actors to perform threatening actions to the system. In our secure interoperability platform, a complex and interconnected scenario where many subsystems of the SG interact with each other in a cooperative way among SG providers, it is of paramount importance to implement powerful access control mechanisms.

As previously discussed, in our scenario, it is particularly critical that the authorization mechanisms are finely configured and allow its users to rapidly perform their tasks, without granting access to forbidden segments of the interoperability platform, e.g., unauthorized access to sensitive data, unauthorized access from one infrastructure to another, etc. We therefore use RBAC, as defined in the IEC/TS 62351-8 [28], to provide access control mechanisms to our infrastructure, as introduced in Section 5.2. When a subject (user) needs authorization credentials to perform tasks according to the job description, the system manager assigns this user a role or set of roles to accomplish this task, following the principle of least privilege, ensuring that the rights assigned to the role for the user allow him access only to the functions in the system

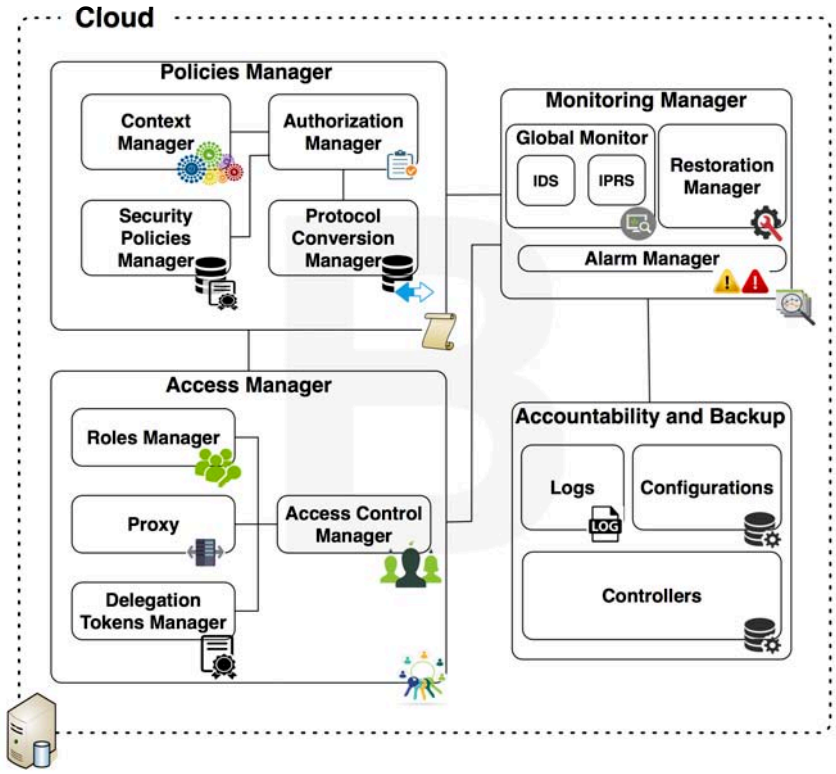


Figure 5.5: Services offered by the cloud

needed to do his job.

5.3.2 Cloud Infrastructure

According to the characteristics of the cloud provided in Section 5.2, it is possible to define the type and features of the cloud infrastructure and the cloud services for the inclusion of cloud technologies within our secure interoperability infrastructure design model. Reviewing the requirements for interoperability we identified in Section 3.3, we observe that related to cloud, there are two main NFRs that need to be fulfilled when deploying cloud services within the interoperability architecture for CPCSs of the SG: *elasticity* and *measured service*. As we can see, these two requirements are naturally satisfied by the aforementioned essential characteristics of the cloud [270].

The use of cloud computing in an interoperability architecture for CIs can result of extreme interest, since this cloud infrastructure supports the operation of the control infrastructure, adding with this operational layer, new means to fulfill the sustainability, dependability and survivability requirements of the CCS (see Sections 3.1 and 3.3) by adding redundancy, distribution, load balancing and virtualization (among other techniques) as described in Section 3.4. The new infrastructure added by the cloud is external to the already existing HW and networks of the CI and its control system.

Cloud computing is a powerful and useful tool for critical systems [264], however, the cloud becomes even more interesting whenever dealing with a federation of partners which want to establish a secure interoperability platform for cooperation motives, in which critical information and control is shared among the different federated providers. Therefore, there are two different possible cloud infrastructures present in our scenario: *dedicated cloud* and *community cloud* (see Section 5.2).

Each SG provider can acquire and manage its own dedicated cloud, either by acquiring the equipment and creating a privately-maintained cloud infrastructure, or contracting the cloud services to a public cloud provider. This dedicated cloud can be used as redundancy mechanisms, to maintain a coherent and actualized backup of the control configurations of the infrastructure, etc. Each SG provider chooses the configurations and dedicated use of its cloud infrastructure, without any influence from the outside.

However, in the interoperability scenario depicted in Figure 5.3, it is very interesting to contemplate the inclusion of a community-managed cloud infrastructure to support the secure interoperability for the different partners. This cloud infrastructure would be deployed according to one of the four deployment models mentioned above, according to the needs and characteristics of the federation of SG providers in need of cloud services. Given that it is a federation of CIs, it is probable that the preference of the SG partners is to acquire and manage its own private cloud infrastructure, creating a *community-managed cloud* infrastructure where one or several partners participate in the maintenance and management of the system. The location of this community-managed cloud infrastructure can take diverse configurations according to the agreements of the federation partners regarding the management of the cloud:

- *Single manager*: one of the partners of the federation is designated as the *cloud manager*, which is in charge of managing the cloud services. In our model for the secure interoperability platform, a management role must be created, the *CloudMNT* following the standard IEC/TS 62351-8 [28] (this role is not pre-defined by the standard, but can be created using the private role values of the standard). This role will be in charge of managing the federation cloud whenever it is necessary, and the permissions to the role must be assigned using the principle of least privilege technique as described in Section 3.4.

- *Multiple managers*: several partners of the federation are designated as cloud managers, and are in charge of managing the cloud services. In this scenario, each of the partners acting as cloud managers should create the management role *CloudMNT* following the standard IEC/TS 62351-8 [28], and its permissions assigned using the principle of least privilege (see Section 3.4). The CloudMNT will be in charge of managing the federated cloud, regardless of its location, since this responsibility becomes a shared task among the cloud manager partners, and the existence of different partners dedicated to the management of the infrastructure helps achieve reliability, accountability and service assurance in the system.

- *On the premises*: the servers and HW that compose the physical layer of the cloud infrastructure are stored on the premises of one or more partners of the federation, which are in charge of maintaining and managing its operation, and configure them in order for the services provided by the cloud to be stable, continued and available for each and everyone of the partners of the federation. This deployment model corresponds to the community cloud described in [270].

- *Out of the premises*: the infrastructure servers of the cloud are located in a location designated by the federation providers, which are external to their premises. This extraction of the cloud infrastructure from the location of the different CIs can suppose an advantage for the operation of the physical servers, since industrial environments are usually surrounded by interferences and industrial noise, which might affect the operation of the equipment. This configuration should not be confused with the deployment scheme denominated public cloud model [270].

Since this interoperability infrastructure is set in the context of CI's cooperation and the information exchange implies the transference of highly sensitive and critical data, it is most probable that the federation partners prefer to acquire and maintain a shared private cloud infrastructure. However, it is possible that some of the less sensitive data generated by the federation can be exported to an external public cloud contracted by the partners. As stated, the public cloud [270] is contracted with an external cloud service provider, which is in charge of the management of the physical layer and the configuration of its servers in order to ensure a quality service for the partners of the federation. In an scenario where the cloud is private or maintained by the community, it is not necessary to describe its service model [270], however, in the case of a public cloud usage, it is possible to model the services required from the cloud as SaaS or PaaS, depending on the implementation of the services required by the federation partners.

The use of public cloud services would help the interoperability architecture to achieve additional backup and redundancy mechanisms for the data in a distributed fashion. Nevertheless, this configuration could be less interesting for the federation of SG providers given the need to trust a third party to manage their services and data. As described in Figure 5.3, the cloud infrastructure serves as means of interoperability among the different subnetworks belonging to the SG providers of the federation. Additionally, it provides varied security services which allow the interoperability platform to operate in a secure, well balanced and solid way. In general terms, the cloud infrastructure provides the secure interoperability platform with the basic security services described in Section 4.1.

In line with with the use of these basic services, and with the objective of maintaining the privacy of the operations of each member of the federation, all connections to the cloud must be encrypted, and the access to the cloud must follow a normalized process of authentication and authorization through the modules dedicated to this access, i.e., the *controllers* as illustrated in Figure 5.4 which will be described in depth in the next sections. Information sharing and communications among different providers can be achieved making use of the PRE technique [132] as described in Section 3.4, a technique that will be discussed for this scenario in the following sections.

5.3.3 Fog Infrastructure

According to the NF requirements elicited in Chapter 3, our secure interoperability architecture must comply with the requirements of flexibility, scalability, robustness, and resilience (among others), which are parent softgoals containing many other related requirements as leafs in the diagram (see Figures 3.3 and 3.4). Thus, in order to fulfill these requirements, it is possible to include in the interoperability architecture certain practices or techniques which help satisfice the requirements, e.g., virtualization, decentralization, distribution, load balancing, redundancy

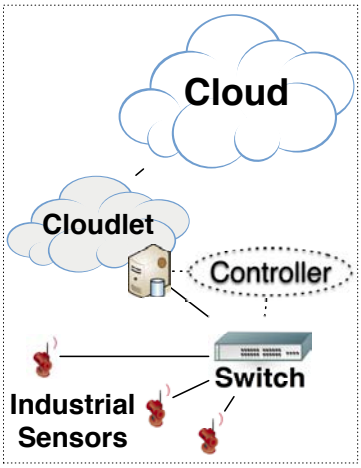


Figure 5.6: Main components of the secure interoperability infrastructure

or diversity (among others, see Figure 3.5).

Techniques and practices as aforementioned can be implemented using technologies such as the ones provided by *edge computing* (i.e., fog computing and the other similar edge paradigms), as introduced in Section 5.2. In our architecture, we apply an adaptation of the concept of fog computing or edge computing, as seen in Figure 5.1, in order to include in our model the most interesting characteristics of this paradigm together with cloud computing, to achieve the characteristics required for a secure interoperability platform for the CPCSs of the SG. Observing Figure 5.3, we see the way in which we combine these two paradigms (i.e., cloud and fog computing) in order to obtain the best characteristics of each one for the interoperability scenario.

In our architecture for the secure interoperability of CPCSs for the SG, we observe mainly two different types of abstractions: the *cloud* and the *cloudlets* (see Figure 5.3). The cloud present in the infrastructure corresponds to the community-managed cloud infrastructure for the federation of SG providers discussed in Section 5.3.2. This cloud helps orchestrate, manage and monitor the internal communications and procedures of the secure interoperability infrastructure between the different subnetworks of the federated CIs. It provides global security services, supporting the robustness, resilience and accountability of the architecture, as well as means to obtain resilient interoperability and control of the CPCSs, even in the presence of critical events. The specific services functionalities implemented by the cloud, illustrated in Figure 5.5, will be discussed in detail in the next sections.

The second abstraction corresponds to the cloudlets present in the architecture, depicted by the small yellow clouds in Figure 5.3. Each cloudlet contains a subsection or subnetwork of the SG infrastructure of one of the SG providers, this section can contain: a corporate network of the infrastructure, one or several remote substations, a remote field control network, etc. (see Section 4.3). These cloudlets are generally composed of a subnetwork with *edge servers* and *access points*, as depicted in the abstraction presented in Figure 3.11. We denominate the access points as *controllers* (as introduced in Section 5.3 and Figure 5.4), and their function is the advanced control, monitoring and management of the control network in their area of influence, supported by the edge servers of the cloudlet. Figure 5.6 illustrates these main components

present in our architecture.

Virtualization

Introducing advanced management and control mechanisms within the edge devices in a critical infrastructure is not always trivial. These devices are mainly RTUs, industrial sensors and actuators, and other legacy equipment with restricted capabilities. Due to the computational and storage constraints these devices present, it is necessary to find an alternative way to place the advances functionalities of the controllers. We can accomplish that making use of virtualization techniques, and placing the computationally and storage-costly functions of the controller within the edge servers of the substation.

These servers are either heavy-duty or powerful-duty CPSs deployed at each substation or cloudlet, such as general-purpose servers, industrial proxies and gateways (see Section 1.1.2). Each cloudlet in the secure interoperability architecture can include one or several controllers, and their main mission is to act as secure access points for the edge devices present in the area of influence of each controller, providing advanced security and interoperability capabilities, as well as the basic security services described in Section 4.1. The specific services and components implemented by the controllers in the architecture (see Figure 5.4) will be described in depth in the following sections.

5.3.4 Software Defined Networking

SDN, as introduced in Section 5.2, presents many advantages over conventional networking [272], such as *increased scalability*, *dynamism*, and *adaptability*, *easiness for migration operations*, *configuration automation*, *quality of service control*, and *reduced costs* for the dedicated networking HW, since only simple switches are needed instead of costly routers and gateways. The networking costs can be further reduced if the system administrators resort also to using the NFV instead of deploying physical SDN-enabled switches or routers. In our scenario, SDN provides the possibility to add advanced intelligent and sophisticated network solutions to an environment where the legacy equipment has very few computational and storage capabilities, but there exists heavy constraints with respect to the real-time performance of the system. Therefore advanced interoperability, security and networking capabilities can be added to the CIs without incurring in the high cost and complicated procedures for replacing current dedicated HW and networking solutions.

In our secure interoperability architecture, we include the SDN paradigm in our belief that this inclusion leverages the aforementioned advantages of the SDN, and helps build the secure interoperability in the complex context of CPCSs, where the constraints and requirements (see Chapter 3) highly restrict the inclusion of any additional component to the control infrastructure. Therefore, within each cloudlet (see Section 5.3.3), we include one or several access points for the subnetwork, in the form of a SDN-enabled switch, whose SDN controller resides in the fog servers (see the previous section) of the cloudlet. For the remainder of the chapter, we will refer to this infrastructure composed of the SDN switch and the SDN virtual controller, as a *controller* (see Section 5.3 and Figure 5.4).

The aim of each controller is to serve as an access point in the cloudlet to the secure interoperability platform designed for the CPCSs of a federation of providers of the SG, in order to allow the different CPSs and other control devices to interact among them, and with other systems in the infrastructure in a secure and efficient manner. Each controller must provide the basic security services described in Section 4.1, and additionally they must also provide the advanced secure interoperability mechanisms which will be described in depth in the following sections.

5.3.5 Defense in Depth

To build a secure interoperability architecture for the CPCSs of the SG, it is necessary to provide basic security services (see Section 4.1), however, the correct operation of the interoperability architecture also relies on the correct operation of the system in the occurrence of threatening events. As discussed in Chapter 4, it is also necessary to provide this critical scenario with advanced security services for its protection. In this section we discuss the inclusion of these security services at the level of controller (see Section 5.3.4) and cloudlet (see Section 5.3.3), where there exist the main access points to this secure interoperability infrastructure (see Figure 5.3).

In order to protect the system, the interoperability platform implements the *Defense in Depth* methodology. Defense in depth is defined as an "*information security strategy integrating people, technology, and operations capabilities to establish variable barriers across multiple layers and missions of the organization*" [141]. The term defense in depth originally referred to a military strategy for the prevention of the advance of an attacker by yielding space to buy time. In computer science, it refers to the placement of protection mechanisms and security policies creating a multiple-layered schema of defense against attacks. This method is widely recommended for the protection of critical systems, where the measures try to prevent security breaches, and allow the organization to early detect them and mitigate them before they inflict any harm to the CI.

There are several forms to implement defense in depth, however, the main configuration used in critical environments is the triad: DMZ network, firewall, and antivirus, as discussed in Section 3.6. Nevertheless, defense in depth can be constructed by integrating two or more of the following security solutions together: antivirus, authentication mechanisms, DMZs, encryption, firewalls, IDSs, IDPRSs, accountability mechanisms, physical security, *Virtual private networks* (VPNs), or access control mechanisms (this list does not contain all the possible security mechanisms implemented in defense in depth).

For our purposes, we choose to implement the security in depth mechanisms including four of the aforementioned security mechanisms: DMZ network, firewall, IDS and VPN, as depicted in Figure 5.7. These security mechanisms will help protect the access points to the secure interoperability platform, therefore any access to a controller in the infrastructure by any edge device will be conditioned and protected by these security procedures. The selected algorithms to act as IDS engines vary from one scenario to another, taking into account the characteristics of the infrastructure and network dynamics, as summarized in Tables 4.5 and 4.6 in Section 4.5. These tables reflect the suitability of the different IDS approaches and techniques taking into account the deployment area of the infrastructure, additionally Table 4.8 illustrates the best

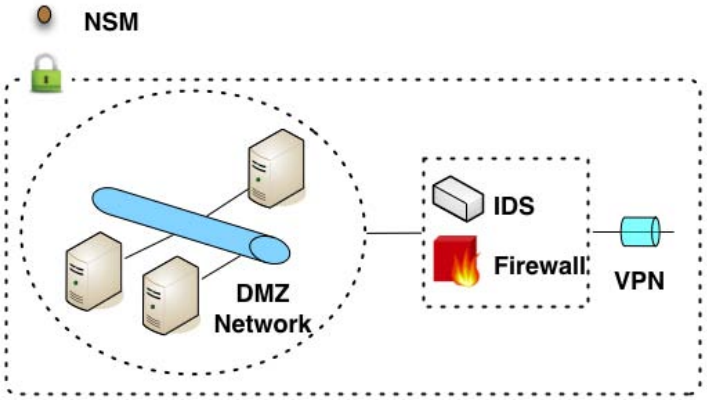


Figure 5.7: Defense in depth and diagnosis system

techniques according to the CPS capabilities.

Taking into account these studies, it is possible to find the best approaches according to the different areas of our interoperability infrastructure, i.e., the IDPRs mechanisms deployed in the cloud or in fog computing nodes (see Figure 5.3) can be complex and sophisticated, since these nodes are powerful-duty (mixed knwoledge-based, with powerful detection algorithms). However those IDSs deployed in field networks where the CPCSs have low computational capabilities, should be lightweight in order to properly function (signature-based solutions with simple detection algorithms such as rules or decision trees implemented). According to the capabilities of the nodes hosting them, they can include already existing solutions such as Snort [102], Bro [152], or other proprietary solutions. The defense in depth is represented by a green lock in the Figure 5.3, where we can observe it is present at all the different segments of the secure interoperability infrastructure.

5.3.6 Diagnosis System

Among the different services provided within the secure interoperability architecture, we include the presence of a *Diagnosis System* (DS), see Figure 5.7. The DS will be in charge of monitoring the status of the varied systems and networks present in the interoperability platform. This diagnosis system is based on the IEC/TS 62351-7 standard [142], which describes the use of the *Network and System Management* (NSM) and *Management Information Base* (MIB) Data Object Models. These objects provide security to the system thanks to the monitoring of the network status and the different nodes and systems working within the infrastructure.

NSM objects deployed across the platform are in charge of retrieving useful information about the health and operation of the infrastructure. This information can be used for intrusion detection, for protection, and additionally, it provides a detailed knowledge about the *context* of each subsystem in the infrastructure, a highly important source of information for control decisions. According to the IEC/TS 62351-7 standard [142], NSM objects can be present at many different components in the secure interoperability platform (see Figures 5.3 and 5.7, where these objects are depicted by a brown dot).

The main objectives of the NSM objects within our secure interoperability architecture is the context monitoring and control, applied taking advantage of the available functionalities described in the IEC/TS 62351-7 standard [142]. As illustrated in Figure 5.3, NSM objects manage and monitor diverse objects in the system, providing the controllers in each cloudlet, and the cloud with vital information about the health and operation of the interoperability infrastructure, the different CSs and its separate CPCSs.

In our infrastructure, the main devices containing NSM objects are the cloudlet's edge servers, which host the controllers among other services, and the medium to powerful-duty devices present at each cloudlet (see Figure 5.3) e.g., computers, servers, gateways, routers, RTUs, mobile devices, etc. Some industrial sensors and wireless sensors are powerful enough to host NSM objects, therefore they will be deployed in those sensor networks with sufficient capabilities to manage their operation. However, there are some devices in which it would be difficult or computationally draining to deploy NSM objects, such as IoT devices, which will not carry this functionality for our architecture. The information in these sections of the networks would be gathered by the network's cluster heads or similar and more computationally powerful devices.

Each controller in the interoperability infrastructure, and the cloud would have a *Context Manager* (CM) module in charge of gathering, correlating and analyzing the data collected by the NSM objects. In the case of the controllers, they will work with the data transmitted by the devices present in their area of influence, the cloud, on the other hand, will gather de data coming from the entire interoperability platform, to generate a global perspective on the operation and behaviors of the infrastructure as a whole. The detailed operation of the CM modules will be explained in depth in the next sections.

5.3.7 Delegation Mechanisms

Delegation is the process of handing over credentials between system's users. There are two types of delegation: *delegation of authentication* and *delegation of authorization*. The first one refers to the use of an identity by an user, different to the one it was authenticated in the system with, given that this operation is permitted (e.g., the `sudo` command in UNIX). The second one is the most common way of delegation, and serves to ensure critical tasks in the system do not get unattended when using security techniques such as the *principle of the least privilege* (see Section 3.4). Therefore, using authorization delegation, the permissions to perform a delegated job are transferred or enabled from a delegator role to a delegatee role. This characteristics is usually found when making use of the RBAC standard (IEC/TS 62351-8 [28]).

We find delegation mechanisms of vital importance in a critical environment such as the SG's control systems, since critical events must be addressed at all times, and if the administrator or operator authorized to address the problem is not available, the situation could escalate rapidly and cause extensive harm to the infrastructure. Therefore, the use of delegation mechanisms ensure that there is always an active user in the system capable of addressing the arising situations in case of necessity, through delegation. To this end, our architecture contains the implementation of mechanisms for authorization delegation within the different roles in the secure interoperability infrastructure for the CPCSs of the SG.

In a normal situation, when a member of the staff wants to access the system e.g., a field

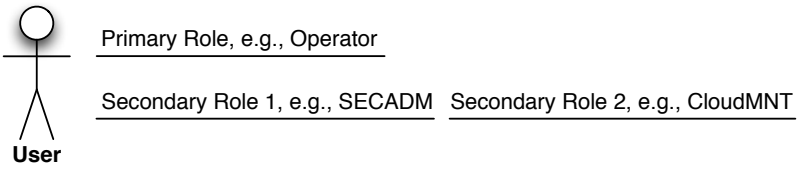


Figure 5.8: Primary and secondary roles

operator effectuating control operations with her laptop, she sends an access request to the controller in the area of influence of the connected laptop. This controller has to verify if the used client satisfies the minimum security requirements for access, i.e., if the operator has the adequate credentials to access the system. This step is managed by the authentication services available within the controller (see Figure 5.4), which will be explained in detail in the following sections.

Once the authentication is achieved, the authorization module is launched, together with the *Policies Manager* (PM) module (see Figure 5.4), the context and status of the target system, its environment and the situation of the interoperability platform are determined, identifying and selecting the security policies regulating the access, and the applicable roles and permissions to the authenticated user. The operation of these modules will be explained in detail in the following sections. In this step, if the context of the system is critical and the user accessing the system has the adequate credentials, the delegation mechanisms are launched. This is better illustrated with an example:

A field operator is effectuating scheduled control operations with her laptop in a SG substation, while working, one of the electrical transformers in the substation breaks down and stops working, the backup equipment fails and the remote control mechanisms are not able to activate it, needing a manual activation of the backup units. This operation is only authorized to be performed by security administrators of the system (RBAC `SecAdm` *pre-defined roles [28]). The operator is the member of the staff that is present at the nearer position to the failing equipment, however, in a normal situation, this maintenance operation is not permitted to her role. The operator has the RBAC pre-defined role* `Operator` *[28] assigned for the regular aspects of her job, additionally, as secondary roles she has the roles* `SecAdm` *and* `CloudMNT` *assigned for delegation in case of necessity, as illustrated in Figure 5.8. When this critical situation arises, the interoperability system is able to identify the critical context of the infrastructure through the information gathered by the context manager and the NSM objects (see Section 5.3.6), activating priority access to the operator in situ, and activating automatically the secondary role of* `SecAdm` *for her, in order to procede as soon as possible to the mitigation tasks needed.*

Therefore, as mentioned in the example, when a critical situation arises in the interoperability architecture, thanks to the delegation mechanisms, a member of the staff which primary role is not allowed to perform certain actions in regular situations, can be authorized to perform emergency mitigation tasks through the activation of the secondary roles (see Figure 5.8). Due to the fact that we are designing an interoperability infrastructure for the members of a federation of SG providers, it is possible that the maintenance of given substations or cloudlets is shared. In this case, the infrastructure has to take into account the cooperation between federated

partners and the need to establish *inter-organization delegation mechanisms* for cooperative and emergency situations.

To provide mechanisms for the delegation among different partner infrastructures, we make use of the proxy re-encryption technique [132]. As described in Section 3.4, the PRE is a type of public-key encryption which allows a proxy to transform ciphertexts generated with a public key, into ciphertexts decryptable by another key. Therefore, this technique allows the cooperation between infrastructures of different providers, given that each organization generates its own delegation schema, providing the proxy the re-encryption keys for the trusted partners, e.g., infrastructure A provides the re-encryption keys Rk_A for the controller present in remote control substation B $RCS_B(Rk_{A \rightarrow RCS_B})$ (see Figure 5.3). The proxy is present in the federated cloud of the secure interoperability platform (see Figure 5.5), allowing access to the partners from all subsystems in the infrastructures. Following the previous example for delegation, we can picture the following scenario:

A field operator from infrastructure A is effectuating scheduled control operations with her laptop in a SG substation shared by infrastructures A and B. While working, one of the electrical transformers belonging to the substation B fails without activating the corresponding backup equipment, which remains offline without possibility of remote activation. There are no field operators from the infrastructure B nearby, therefore, it is needed that the field operator from infrastructure A tends to the equipment as soon as possible to activate the backup units. The operator present at the substation is correctly authenticated in the system with the role `SecAdm` *from the infrastructure A, however, to do this job, a specific authorization is needed to perform tasks reserved to security administrators from infrastructure B. To obtain the authorization, a connection is established to the federation cloud, in order to retrieve the delegated credentials for the operator, according to the security policies in place, and with high priority given that the context of the system is critical and immediate actions are needed.*

Therefore, thanks to the deployed delegation schema in the secure interoperability architecture, it is possible to rapidly grant authorizations for mitigation and restoration actions to the different members of the federation, allowing to rapidly tend to emergency situations, and therefore prevent cascading failures and shortages of service in the different infrastructures belonging to the federation of providers of the SG. This technique is also useful for data sharing and cooperation between the different partners of the federation, where PRE allows translating the ciphered data from the infrastructure A with the public key Pk_A into the public key of the subsystem in need of information belonging to infrastructure B (with the public key Pk_{SB}). Thus, the needed data `X` is re-ciphered: $E_{Pk_A}(X) \rightarrow E_{Pk_{SB}}(X)$.

Another interesting scenario for delegation is the management of the federated cloud infrastructure (see Section 5.3.2). As previously described, our security interoperability architecture makes use of a cloud infrastructure shared among the SG partners (see Figure 5.3). This cloud would be most likely managed by two or more of the partners in the federation, building a federated cloud with a multiple manager schema, as described in Section 5.3.2. Therefore, the role `CloudMNT` will be present in more than one of the partner organizations, and the need arises to manage and control the permissions and tasks assigned to these roles. In the first place, the permissions assigned to the CloudMNT role have to be assigned following the principle of least privilege [28] technique as described in Section 3.4, this would increase the security within the infrastructure, given that this global role would not be able to perform any tasks related with

other infrastructure than the cloud.

CloudMNT roles will connect to the federation cloud through the controller in the area of influence, which establishes a secure connection to the cloud. Usually, CloudMNTs will manage the infrastructure and access to the needed information and configurations stored and administered by the partner's infrastructure the CloudMNT belongs to. However, there are occasions where further accesses are needed to data belonging to other partner of the federation. Given that for security and privacy reasons, the information stored in the cloud is ciphered with each organization's public key (Pk_{O_i}), this information is not generally accessible to other members of the federation.

To allow cooperation, enabling and facilitating the needed data sharing between partners, the re-encryption mechanisms are used. The proxy in the cloud (see Figure 5.5) will re-cipher the data encrypted with the public cloud of the organization who owns the data (Pk_{O_A}), into the public key of the controller from where the CloudMNT is managing the infrastructure ($Pk_{Cnt_{B_x}}$). Since this access to the shared data also follows the principle of the least privilege, the security and privacy of the partner's data is maintained, allowing the transference of data among organizations in a secure way.

Further applications of delegation mechanisms and PRE are the use of these technologies to enable the processes of auditing. As we mentioned, auditing is vital for implementing accountability processes, and helps maintain a set of quality standards highly important for the correct operation of the system (see Chapter 3). Audits can be classified into *internal audits* and *external audits*, the first one refers to the process of auditing carried out by personnel belonging to the federation of SG providers, the later refers to the performance of audits by external auditors (from the government, from auditing firms, etc.). In order to enable the audits in the system, the auditors need to access the information stored by the system (among other types of data) during the processes for accountability (logs, configurations), generated by all the systems in the secure interoperability platform and gathered by the controllers (see Figure 5.4) and the cloud (see Figure 5.5).

Therefore, the auditors (either internal or external), need to access the system in order to access the data. Since this data is protected by the diverse infrastructures belonging to the federation, it is necessary to create a profile associated to the auditor, and assign it an RBAC role, SecAud [28], applying the adequate permissions and security accesses. Each of these SecAuds will be able to audit the infrastructure subject to the audit. However, due to the need to provide the auditor access to the shared data in the secure interoperability platform, the PRE mechanisms are put to use in a similar way to the previously mentioned scenario for the management of the cloud by the CloudMNTs. In this case, the proxy will re-encrypt the data requested by the auditor from the public key of the infrastructure source of the information (Pk_{O_A}), to the public key of the auditor in the system ($Pk_{Auditor_x}$), allowing information only to the auditor to this sensitive data. This way, external audits such as governmental audits for the monitoring of the correct operation of the CIs and their adherence to the procedures stated by the law can be performed.

5.4 Design of the Services Modules for the Cloud and Controllers Components of the Infrastructure

This section is devoted to the description and analysis of the service modules present in the controller (see Figure 5.4) and cloud (see Figure 5.5) components of the secure interoperability infrastructure (see Figure 5.3). As we have discussed in previous sections, the main function of the controller is to serve as access point to the interoperability platform to the actors of the infrastructure, providing security services among others. The cloud services aim to orchestrate and guide the interoperability infrastructure behavior at a global level, providing accountability and redundancy mechanisms to avoid service interruptions and improve the mitigation and restoration tasks in the federated systems.

Therefore, in this section we describe in depth each of the five different services modules of the controllers and cloud components in the platform, and discuss the variations in their functionalities according to their location, i.e., whether they are present at the controllers or at the cloud.

5.4.1 Authentication and Access Manager

The first module we address is the *Authentication and Access Manager* (AAM) service module, which is present at both the controller and the cloud level in the secure interoperability infrastructure (see Figure 5.3). This module is focused on the provision of authentication to the different actors of the infrastructure, in order to be able to access their resources. The AAM is composed of six different components: the *Roles Manager* and the *Access Control Manager* (ACM) which are implemented both in the controllers and in the cloud, the *Identity Manager* and the *Authentication Manager* that are only present at the controller's level, the *Proxy* and the *Delegation Tokens Manager* (DTM) which are present only at the cloud's level.

Identity Manager

The *Identity Manager* is a service module residing at the controllers level in the secure interoperability platform, which is in charge of storing and managing, i.e., creating, modifying and deleting, the identities of the actors in the system. This service can be implemented in different ways, from using simple XML databases to more specialized solutions such as *Lightweight Directory Access Protocol* (LDAP) [276]. Thus, when an actor wants to access the secure interoperability platform, it must provide its credentials to the system, and they will be matched with the information provided by the identity manager on the legitimate users of the system.

This information is available only at the level of cloudlet, and it is not replicated through the interoperability platform, i.e., the system is expected to authenticate only a reduced number of actors in the system, and each controller stores the identities of the regular actors in its area of influence. Inclusion of new actors for this cloudlet is performed by system administrators on demand. However, since we are contemplating an interoperability infrastructure where varied actors belonging to different areas of influence need to be successfully authenticated in the system in order to perform cooperative tasks, the identity information need to be shared across

the platform in the event it is needed. This task is enabled by the cloud infrastructure (see Section 5.3.2), where the access manager module in the cloud (see Figure 5.5) provides support to the sharing of information among different controllers using on demand connections between controllers, through their *Access Control Manager* module, which we will discuss later.

Authentication Manager

The *Authentication Manager* module provides the service of storing and managing the authentication credentials in the secure interoperability platform. The system administrators can add, modify or delete the credentials associated to each actor in the database. This module is present at the level of controllers, and, as in the case of the identity manager module, it contains the authentication credentials information for the actors that are regularly present in the area of influence of each controller. Whenever it is necessary to authenticate a legitimate actor external to the area of influence of a given controller, it must request the information credentials of this user to its regular controller in the network (or in case of failure of that controller, to the backup copy present in the cloud) using the *Access Control Manager* module as will be discussed in the following sections.

This module, therefore, manages the authentication and access credentials of the actors in the interoperability platform. These credentials are usually presented as *Access Tokens*, as described in the IEC/TS 62351-8 standard [28], which can be in the form of certificates, and they provide access rights to certain areas of the system according to the identity of the actor, the security policies of the system (given by the *Policies Manager* module which will be described in detail later on), and the general status of the system. The consideration of the state of the system requested to be accessed is of high importance in a critical environment, where all actions and tasks need to be prioritized in order to ensure that high priority or critical tasks are tended to immediately, leaving these less urgent tasks with a minor degree of priority and therefore carried out in a later instant of time.

Roles Manager

The *Roles Manager* service is in charge of managing the roles in the secure interoperability platform according to the RBAC standard (IEC/TS 62351-8) [28], as described in Section 5.3.1. This service module is present at each controller in the secure interoperability architecture and within the cloud AAM. The role manager present at the controller's level is in charge of managing the roles of the actors in the system belonging to the area of influence of this given controller, the same module in the cloud is in charge of the management of the global roles in the federation infrastructure, as well as maintain a backup copy of the roles configuration at each of the controllers present in the secure interoperability platform.

This service module is also in charge of the assignment of roles to rights (permissions) according to the RBAC standard (see Section 5.3.1). This procedure enables the mapping of roles to tasks performed in the system, and the management at the controller's level provides the system with additional flexibility, allowing the system administrators of each subnetwork to manage the profiles and permissions locally, although always following the system's security policies and the established *principle of the least privilege* (see Section 3.4) and the *Dynamic Separation of*

Duties (DSD), which limits the availability of rights to the users by establishing constraints on the roles they can have activated [28]. Therefore the role manager (either in the controller or the cloud) enables the system administrators to create, modify and delete any role in the system and their associated rights and tasks.

Access Control Manager

The *Access Control Manager* (ACM) service module resides both at the controller's level in the secure interoperability infrastructure, and at the cloud's level. Its main function is to provide access to the systems in the platform to the actors according to their credentials, assigned roles and permissions, and taking into account the status of the system and the security policies present. To perform these functions, this module receives input from different modules in the system, within the own AAM it receives information from the identity manager, the roles manager and the authentication manager (described above); from the external modules, the ACM receives information from the *Policies Manager* and the *Monitoring Manager*, which will be described in detail in the following sections.

Therefore, the ACM, in order to make the decision whether or not to grant access to an actor to the system, receives the actor's credentials, which can be mapped to the information present in the authentication manager in order to determine if it is a legitimate actor of the interoperability platform or not. Once established the legitimacy of the actor, the system can verify its identity using the information stored on the identity manager module. The ACM also receives information on the tasks and requests the actor needs to perform on the interoperability platform from the received credentials and verifies the roles and rights necessary to perform them through the roles manager.

With all this information, together with the analysis on the system status and context provided by the *Policies Manager* module, and the identification of any suspicious behavior on the part of the actor provided by the *Monitoring Manager* module, the ACM can take the decision whether or not to authenticate and grant access to the system to the requesting actor. This process is performed at each controller locally, with the support of the ACM module present in the cloud, which is capable of providing absent information (i.e., present in other controllers or in the backup copy) to the local controller in order to perform the authentication in the system.

Delegation Tokens Manager and Proxy

The *Delegation Tokens Manager* (DTM) and *Proxy* modules are services present in the cloud's AAM (see Figure 5.5). They act in case of colapse or failure of the ground servers of a cloudlet, which contain the controllers of the cloudlet. The DTM and proxy provide robustness to the secure interoperability platform, since it allows the actors in the area of influence of the failing system (cloudlet server, specific controller, etc.) to continue their tasks by redirecting their control requests to any other responsive controller in the federation they need to access for service continuity, regardless of the smart grid provider the newly accessed controller belongs to.

Therefore, the DTM and proxy modules provide delegation mechanisms to the secure interoperability infrastructure, as described in Section 5.3.7. According to the procedure described in Section 5.3.7, each controller in the federation is in charge of generating and maintaining a set of delegation tokens which grant delegation authorization to a set of trusted controllers in the federation belonging to other service providers. Each controller periodically transfers these delegation tokens to the DTM present in the cloud for storage.

In the event a legitimate actor fails to be authenticated with the controllers in the area of its own service provider, it would redirect their request to any other responsive controller. This controller may not have the information needed for authenticate and authorize this new actor, therefore, it will request the cloud the necessary credentials. The cloud would consult the requested information through the global roles manager and ACM modules to verify the credentials of the actor and its assigned roles and permissions, and re-encrypt the requested credentials for the new controller using the proxy module in the cloud and the previously generated delegation token present in the DTM. This process always takes into account the information coming from the *Monitoring Manager*, the status of the system and the prioritization of actions coming from the *Policies Manager*, as described in detail in Section 5.3.7. The new re-encrypted credentials are sent to the local AAM to grant access to the actor to the network.

5.4.2 Policies Manager

The *Policies Manager* service module is one of the main components needed for the implementation of interoperability in our secure interoperability architecture model. This module is in charge of allowing the different actors in the system access to the federation's networks and resources present in the interoperability platform. This function is a critical one, since any access to critical resources increments its workload, and in an environment where thousand of nodes coexists in a network, incorrect accesses and workload assignments can result in system faults causing service interruptions and errors cascading through the interconnected infrastructures.

Therefore, the policies manager is responsible for managing the access to the different resources of the federation, remote or local, granting permissions to perform tasks only to authorized actors of the system and only in the case that system's status (context) is compatible with these actions (see Section 5.3.6 for information on the context). An example of the actuation of the policies manager module can be observed in the following scenario: *An external auditor is scheduled to audit the cloudlet servers of infrastructure A, however, there has been a critical error in one of the cloudlets, collapsing the communications network in that area. When the external auditor (a legitimate actor in the system) requests access to these servers, the policies manager identifies the critical context in this area of the network and delays the auditor's access until the system administrator in charge of maintenance in the affected cloudlet can access the system to perform restoration and recovery tasks.*

As we can see, this module is of vital importance for the correct operation of our secure interoperability platform. The policies manager module resides at each controller in the system and also in the federation cloud. The policies manager is composed of four different services sub-modules: the *Context Manager* and the *Authorization Manager*, which are present both in the controllers and in the cloud, the *Protocol Converter* which is present at the controller's level,

and the *Security Policies Manager* and *Protocol Conversion Manager* backup modules that are present in the cloud. We devote the remainder of this section to the description of each of these modules.

Context Manager

The *Context Manager* module, as previously stated, is present at the cloud's level and within each controller of the system. This module is in charge of gathering the information collected by the diagnosis system and NSM objects [142] deployed through the secure interoperability platform (see Figures 5.3 and 5.7, where these objects are depicted by a brown dot), as previously described in Section 5.3.6. This module is responsible for the system's context awareness mechanisms (see Section 3.3), i.e., the CM modules retrieve contextual information on the status of the system, filtering and storing it for later consults.

This information is used frequently by the controllers and the cloud in order to take ad-hoc decision on the authentication and authorization processes, as well as to provide up-to-date information to the *Monitoring Manager* module to perform the analysis of the network. Based on the information provided by the CM, together with the security policies of the federation, actions and accessed are managed and prioritized in our interoperability platform. Although the functionality performed by the CM in the controllers and in the cloud is basically the same, it is necessary to differentiate the type of information that is provided by the NSM objects to the controllers is of a local perspective of the networks in their area of influence, while the information gathered by the CM in the cloud receives global information from all sections of the secure interoperability platform.

Further, it is important to stress the importance of the different CM modules, since each controller situational awareness is constructed using information which comes in an important part from the context. The information gathered by the diagnosis system and NSM objects, together with the monitoring manager and, in a lesser degree, the other managers in the controller and the cloud allow the system to perceive the system's dynamics, threats and possible risks and impacts to its operation.

Authorization Manager

The *Authorization Manager* module is in charge of providing the legitimate actors in the system authorization to access the different resources of the secure interoperability infrastructure. To this end, this module concentrates the contextual information of the system coming from the CM, information from the security policies of the system and authentication and access information coming from the AAM module. The authorization manager module resides both in the cloud and in the controllers of the system. The authorization process is generally performed at the controller's level, however, this is a vital process that needs to be extremely robust in order to ensure the reliability and survivability of the federation's infrastructure, therefore the cloud offers an authorization manager service module completely functional and capable of performing authorization in the event of critical dynamics occurring in the system.

Therefore, when a legitimate actor authenticated within the system requests access to a given

resource (local or remote), it sends the authorization manager information regarding the type of access and the target resource requested. The authorization manager retrieves information on the roles of the actor from the AAM module, verifying whether this actor has permission to access the resource and perform the indicated tasks (see Section 5.4.1). Once corroborated this information, the authorization manager enquires the CM about the context and status of the target resource and its environment. If the context is not labelled as critical, the policies manager creates the authorization token according to the RBAC (see IEC/TS 62351-8 [28]), and sends it to the *Protocol Converter* service module to proceed to the requested network.

In the event of encountering a critical situation (identified through the CM by the diagnosis system, see Section 5.3.6), only operations of maintenance and mitigation are allowed in the affected section of the system. Therefore, only the personnel holding the RBAC roles of SecAdm, Operator, etc. are authorized to access this section of the network to fix the problem. These accesses are prioritized by the CM and the authorization manager module.

This authorization process, as we previously stated, is usually performed at the controller's level, however, there are two events in which the authorization manager of the cloud is needed. First, in case of remote accesses between subnetworks, the cloud must verify the state and context of the target node and environment to determine whether a critical context impedes the access for non-critical operations. The authorization manager in the cloud would be therefore in charge of performing these verifications and exclude non-essential traffic in critical cases, allowing only authorized personnel to perform critical operations.

Additionally, in the event of failure of the local controller, its functionality is transferred to other controller in the cloudlet or even to other remote controllers by means of delegation (see Section 5.3.7). In this case, the delegated controller may not have an updated database of actors for authorization procedures, since they are external to their area of influence. To be able to perform the authorization process, the local authorization manager module transfers this operation to the cloud authorization manager, or makes use of the information available in the databases of the *Security Policies Manager* and the *Protocol Conversion Manager* service modules in the cloud to update its databases and perform the authorization operations.

Protocol Converter

The *Protocol Converter* module is a complex component that resides within each controller. Its main functionality is to provide a translation service among the protocols of the heterogeneous networks in the secure interoperability platform for the CPCSs of a federation of SG providers. Locally, each controller manages and connects multiple type of objects to the interoperability infrastructure, as we discuss an Industry 4.0 scenario (see Section 5.1) e.g,. industrial sensors, personal computers, mobile devices, RFID tags, RTUs, etc. Therefore, each object communicates with its assigned controller using its own protocol, and the need of communications among them poses the need for the controller to provide a translation and conversion platform for the communication.

However, it is also necessary to address the need for the same translation and conversion mechanisms when there is a need to access remote resources in other subnetworks of the interoperability infrastructure, e.g., a security administrator (SecAdm) in the corporate network of the Infrastructure A needs to manage a non-responsive industrial sensor in the Remote Control Substation

B (see Figure 5.3). In this scenario, the administrator connects to the local controller using its PDA, and this controller is in charge of converting the command messages to the adequate protocol for the industrial sensor in Substation B. This procedure consists on four main steps or services (see Figure 5.4): *Normalization*, *Protocol Conversion*, *Security Policy Conversion* and *Packet Reconstruction*.

Firstly, the normalization service extracts the useful information of the packet, i.e., inspects the incoming packet to determine the header and payload information, extracting the source and destination nodes, the command code and other relevant information in the package, discarding non-relevant information. Once extracted the functions codes for the commands and the relevant information from the packet, the protocol conversion module translates the control codes from the source protocol to the destination protocol. The security policy conversion module is in charge of transforming the security policy from the source to the destination node according to the IEC/TS 62351 security standard[134]. This service also participates in the normalization process since the packet is encrypted when it reaches the controller, in accordance with the security policy from the source network (e.g., symmetric key) and the resulting packet has to receive the security policy of the destination network (e.g., certificates) [28].

After determining the security policy conversion and the protocol conversion, the packet reconstruction service creates a new packet with the translated information provided by the protocol conversion and security policy conversion modules. This module avoid the need for encapsulation and other costly methods of translation that would require high computational resources available for the devices participating in the network, due to the wide variety of protocols and security policies that can be present in the federation of providers of the SG, and specially among the CCPSs networks present in our secure interoperability infrastructure.

Security Policies Manager and Protocol Conversion Manager

The *Security Policies Manager* (SPM) and the *Protocol Conversion Manager* (PCM) service modules reside only in the federation cloud. The SPM is in charge of storing and maintaining a set of databases (e.g., XML) containing the security policies implemented by each of the controllers in the federation. This manager stores the policies as backup copies, which allow the system managers to update the local controllers' security policy conversion databases in the event of malfunctions, system upgrades or in case of delegation procedures.

The PCM module also maintains a set of databases as backup and maintenance modules for each local protocol converter module. Within each controller, the protocol converter has the means to translate the traffic among the different protocols existing in its subnetwork or area of influence (e.g., Modbus [277], DNP3 [14], Bluetooth [114], WirelessHart [115]), the PCM module retrieves all this knowledge and saves it as backup in a database. This information is available for local maintenance and recovery procedures.

5.4.3 Monitoring Manager

The *Monitoring Manager* service module is in charge of scanning the systems and networks in the secure interoperability infrastructure in order to detect any anomalies or attacks to the

system which pose risks for the systems belonging to the platform (see Chapter 4). This module is present at both the controllers and cloud level, where they perform their tasks either from a local perspective or a global perspective, respectively. The monitoring manager receives information for surveillance coming from the NSM objects deployed across the platform (see Section 5.3.6). This module is composed of three main sub-modules, i.e., the *Monitor*, the *Restoration Manager*, and the *Alarm Manager* services modules. All three of them are present at both the monitoring managers at the controllers and at the cloud.

Monitor

In addition to the monitoring infrastructure for the network monitoring and defense described in Section 5.3.5, each controller contains a *Local Monitor* (LM) service (see Figure 5.4) in order to detect any abnormal event in the network, i.e., any anomaly of the system or malicious attack occurring within the range of its area of influence. These monitors, as previously stated, gather the information obtained by the NSM objects [142] which make available important behavior information from all the devices in the system, including connected "things" and legacy equipment. Within each LM, the IDS performs the detection of anomalous events using the preferred techniques configured by the system managers (see Section 4.2.1). Additionally, the LM can implement an IDPRS, which can launch countermeasure actions to prevent or react to the dangerous events occurring within the subnetwork (see Section 4.3).

On the other hand, the cloud implements a *Global Monitor* (GM) (see Figure 5.5) which gathers information from the federation subnetworks, containing alarms and context information from the NSM objects (as detailed in Section 5.3.6), and performs a global detection service. It also monitors the behaviors of the federation personnel and performs a semantic detection based on behavior patterns [189]. Since the global IDPRS runs on the computationally powerful servers of the cloud, it can perform heavy duty-tasks such as automatically determining and launching prevention and reaction operations taking into account the risks and costs of these operations. They also can provide support for automatic or semi-automatic countermeasure and mitigation activities, as described in Section 4.3.5.

Restoration Manager

The *Restoration Manager* module is in charge of returning the system to its normal state and operating configuration [241] [242]. Restoration can be implemented using varied techniques, i.e., *active replication*, *passive replication*, *rollback based on checkpoints*, *rollback based on message logging*, etc. as described in Section 4.4. This module is available in our architecture at the level of each controller, to perform local restoration tasks, and also it is available in the cloud, where the system can perform restoration actions at a global level, managing the recovery operations in different controllers and applying different kinds of restoration techniques. Each infrastructure manager will decide which restoration mechanisms to implement within each subnetwork.

Alarm Manager

The *Alarm Manager* service module's objective is to alert the system administrators and field operators of any abnormal event in the system, so they can take care of the events and avoid escalating problems that can cascade through the infrastructure if the anomalous occurrences are not mitigated on time. The alarm manager receives the information filtered and contextualized through the NSM objects deployed across the system, and the monitors in the monitoring managers. This module is present in each of the controllers, supervising the area of influence of each controller, and where they are able to send local alerts to the nearby personnel and connected devices. The alarm manager is also present in the cloud, receiving alerts from the subnetworks in the secure interoperability infrastructure, and notifying the personnel in charge of the global supervision. Additionally, the alarm manager in the cloud can provide notifications of any abnormal or dangerous event detected at a global level by the monitoring manager to the supervisors of the federated system.

5.4.4 Accountability and Backup

The *Accountability and Backup* service is in charge of gathering sufficient evidence to perform audits and forensic techniques on the data, and also to store and manage the configurations of the system devices. This module is located in both the cloud and each of the the separate controllers in the fog servers in the ground networks of the secure interoperability infrastructure (see Figure 5.3). As in previously detailed modules, the main functionality of the accountability and backup services in the controllers in the retrieval and storage of local data from the area of influence of each controller, while the equivalent module residing in the cloud performs the same activities at a global level in the whole platform, and also stores backup copies of the information stored by each controller. Each accountability and backup module belonging to the controllers contains two modules: *Logs* and *Configurations* services, while the cloud, in addition to these two modules, contains a third one called *Controllers*.

The *Logs* module is in charge of gathering and storing the events in the system in the form of logs. As described in Section 5.3.6, these events include information from the accesses to the system, state changes, faults, alarms, among others. The logs should contain valuable information capable of providing insight about the events occurring within the system, e.g., they should reflect and provide useful data about a system crash in a remote substation, including data such as the time stamp, type of event, time instant, status, device ID, etc. The methods for event recording should always follow the standards IEC/TS 62351-8 [28] and IEC/TS 62351-7 [142]. As stated, this module exists in the controllers at each subnetwork, which makes use of the NSM objects deployed in the network to gather and store the logs in the ground servers of each cloudlet, and then transfer them to the cloud for backup. As previously described, the logs service in the cloud focuses on logging useful information at a global level containing information about the communications and accesses occurring among the different subnetworks in the secure interoperability infrastructure.

The *Configurations* module's main functionality is to manage and store the configurations of each device in the system, to perform recovery duties or maintenance operations. At the controllers level, the configurations module consists on a database, e.g., an XML database, which

stores the configurations applied to each devices in their subnetwork or area of influence. These configuration's information is maintained at the ground level, and transferred to the cloud for backup or when the system manager updates the configurations of the devices. The configurations module in the cloud manages and stores the configurations database of the federation. This task can be considered as Big Data [278] and should be handled in a way that (i) each configuration's file and therefore database is as lightweight as possible, (ii) the general database is well indexed and organized for search and retrieve operations, and (iii) the cloud servers in charge of the task are sufficiently powerful to handle these operations.

The last module, present only in the cloud, is the *Controllers* service module, which is in charge of maintaining and managing a database of configurations and backups of each of the controllers of the federation. This database contains copies of the virtualized controllers running in the ground servers of the fog, at the cloudlet level, in order to be able to rapidly restore the cloudlet functionalities in the event of failures.

In addition to the aforementioned functionalities, the cloud plays an important role allowing access to the data of the whole federation for accountability reasons. The creation of logs of the system activities allows internal and external auditors to access and review the processes and behaviors of the systems in the federation to determine whether the member's security policies and performance are adequate and in accordance with the law. And also the logs provide very important information and insight about the system behavior after system crashes using e.g., forensic techniques.

5.4.5 Maintenance Manager

The *Maintenance Manager* is a service module present only at the controller's level in the secure interoperability infrastructure (see Figure 5.4). The aim of this module is to help the system administrator to manage the services and functionalities offered by the controller to its subnetwork, improve and update the configurations of the controllers and main network devices in the cloudlet. This maintenance manager module stores a database of the configurations of the system, and configuration mechanisms to restore, backup, recover or modify each of the services operating within the controller. As the controllers are virtualized, the management operations are performed at the ground servers of the subnetwork, in a context of virtual management and orchestration [126] that allows the continuous operation of the system without disruption for maintenance processes.

5.5 Analysis of the Operation of the Secure Interoperability Architecture

This interoperability architecture model is designed to enable communications between the actors of an industrial platform, despite their intrinsic characteristics (protocol, computational capabilities, location, etc.), in a secure and prioritized manner. Prioritization in this context, as defined earlier, refers to the transmission of critical communications (data, commands) without interruption or delays, even in the presence of network congestions or broken links. To

this end the platform needs to ensure that its processes are robust enough to manage critical situations.

We contemplate the communications within the platform at two different levels: *local communications* within the same cloudlet, e.g., between the nodes of the same remote substation or SCADA center, and *global communications* where a node in a subnetwork can establish a successful connection with any other node in the network of the federation, e.g., data sharing between the servers of two providers in the federation.

5.5.1 Local Communication

Local communications make use of the cloudlet infrastructure, without going to the cloud by default. These connections are established between the nodes of the same cloudlet, and therefore our interoperability platform offers the services available at the level of controller (see Figure 5.4). When a local communication is requested, e.g., a field operator performing maintenance operations requests access to an RTU in a remote control network (see Figure 5.3), the new device needs to establish a connection with the controller present in the area of influence.

The successful establishment of a connection with the controller enables the device access to the interoperability platform, therefore, valid credentials need to be provided and verified by the new actor. This process is conducted at the AAM service module. Therefore, the new node entering the network provides a set of credentials which will be validated by the ACM in coordination with the the identity manager service (see Section 5.4.1). If the platform uses the RBAC standard [28], the new actor will provide an authentication token in the form of an X.509 type A ID certificate, an X.509 type B attribute certificate, or a class C access token [279].

Once the identity manager has validated the identity of the new device with the credentials provided, the authentication manager authenticates the new actor in the network. Following the IEC/TS 62351-8 standard [28], the access control manager retrieves the roles and permissions for the new actor from the roles manager taking into account the credentials provided and assign them to it. Therefore, through this first step, a new legitimate device is authenticated in the interoperability platform, and assigned a set of roles and permissions in the system according to the RBAC standard.

The authentication process, as stated in Section 5.4.1 receives information from the policies manager service in order to determine if the system's operation is normal, or there has happened a critical event, which would mean that a non-critical connection to the platform is delayed until the system is restored to normal operation. Additionally, the AAM service also receives input from the monitoring manager module, which is in charge of detecting any abnormal behavior or connections to the system (any unusual connections could be rejected by this module before authentication).

When the authentication is performed in the system, the new node is authorized to perform diverse actions, according to the roles and permissions assigned to it during the process of authorization. If the new device requests a target node (e.g., an RTU) to perform some actions by using the same communication protocol, e.g., they both establish a connection using Modbus/TCP, the policies manager module would only extract context information for the context

manager and authorize the communication if the system's operation is normal (no critical event requires non-critical connections to be dropped, see more in Section 5.4.2).

However, if the new device connected to the network uses a communications protocol different from the target node, i.e., it is necessary to provide a protocol conversion for the two devices to communicate, e.g., a WirelessHart PLC wants to connect to a Modbus/TCP RTU. Here the policies manager's protocol converter module (see Section 5.4.2) is activated. There are several ways to implement the protocol converter, however the theoretical operation of the module when node A wants to communicate to node B in the network is as described in Sections 5.4.2 and 5.4.2. When a packet from node A to node B (it requires conversion of protocol) arrives to the controller, it is normalized, the protocol and the security policies are converted, and then the packet is built again. Once reconstructed the packet, taking information from the context manager in order to determine if the communications can proceed (they are authorized by the authorization manager), the packet is sent to node B of the network.

All this communication's traffic is analyzed and monitored by the monitoring manager service, together with the context manager module. The local monitor determines if the traffic in the network is legitimate, or there are threatening behaviors in the network (see Section 5.4.3) in which case, its response system would act to protect the system, and the alarm manager module would alert the system administrators to fix the situation (see Section 5.4.3). In case the system would be affected, the restoration manager module would act in order to return the system to its regular operation (see Section 5.4.3).

Any traffic and events in the network is saved for later analysis, this is performed by the accountability and backup module (see Section 5.4.4). Additionally, the maintenance manager service allows direct intervention within the controller to the system administrators, where its configurations database together with the configurations backups of the accountability module helps the operators to perform maintenance and optimization operations within the controller.

5.5.2 Global Communication

The other type of communication within the secure interoperability infrastructure is the global communication processes, where two nodes in the federation of provider's control network establish a connection to exchange information or commands (see Figure 5.3). To enable these communications, these nodes make use of their assigned controller's services (the controller which authenticates the node within the platform), as well as the services in the cloud (see Figure 5.5).

In order to establish a communication between local node A and remote node B, A must be first authenticated in the system, and authorized to perform the desired action using the local services of its controller, as detailed in the previous section. Once verified the need to establish a remote connection, the controller redirects the traffic through the cloud, to the remote network (and assigned controller) where node B is. In simple legitimate communications where data or commands are transferred, the functions of the cloud are limited to monitoring global dynamics and providing accountability services for the global connections in the platform (see Sections 5.4.3 and 5.4.4). The other services perform periodic updates of configurations to their databases according to the different network configurations in the platform.

When there is the need to provide interoperability services within nodes of remote networks, the protocol conversion is performed at the local level (at each controller) as described in the previous section, only if the controllers establishing the connection know the protocols used by both the source and the destination nodes in the communication. If this configuration is not available at a given controller, the protocol conversion is performed in the cloud, in a similar manner to the controller's process, while the protocol conversion and the security policy conversion services in the controller are updated with the information provided by the protocol conversion manager and the security policies manager modules (respectively) at the cloud.

The last case in which the cloud services are needed for remote communication is when a controller belonging to the platform stops working. This can be due to a malfunction, system anomaly, congestion, maintenance operations, cyberattacks, etc. Since the secure interoperability platform needs to be reliable and robust, there are methods at local and global level to avoid sections of the control network to become isolated. At the local level, each controller is configured to accept connections and authenticate into the platform those nodes that are usually in their area of influence, plus some of the nodes that usually belong to the neighboring controllers (they are cached in case there is congestion in the network or the computational efficiency of a given system is reduced).

However, there are two different scenarios where a controller needs to authenticate a previously unknown actor in the network. In the first place when a new legitimate node is added to the network in the area of influence of the controller. The second case is when a neighboring controller stops operating (due to malfunctions or maintenance) and the nodes in its area of influence become isolated and therefore request access to the interoperability platform through a new controller. The operations to enable the new actors in the (local and global) network are performed at the cloud by the access manager module (see Section 5.4.1), specifically by the delegation tokens manager and proxy modules (see Section 5.4.1).

The procedures applied in the case of a controller which stops working and the nodes in its area of influence need new access to the platform is described in Section 5.3.7, where we see the delegation process in detail. In the case of arrival of a new node to the network, when the controller does not have the node's identity in its database, the AAM module sends a query to the cloud's access manager database in order to retrieve the information from the new node from the platform's databases. Once downloaded to the local controller, the configuration is updated and the node becomes a legitimate member of this controller's network. It is possible that system administrators add manually the new identity, roles and permissions to the local controller instead of making them available in the cloud. These local configurations would be later transferred to the cloud via the maintenance manager and the accountability and backup services (see Sections 5.4.5 and 5.4.4).

5.5.3 Cases of Example of the Operation of the Platform

This section is devoted to present different examples of application of the communication processes in our secure interoperability infrastructure. In the first place, we present an scenario where elements from the IoT interact with a SG platform. Secondly, we describe an scenario where a cyber stealth attack impacts the regular operation of a network segment which belongs to the interoperability platform, and the procedures taken to palliate the threat.

The IoT Scenario

This first case of example contemplates the operation of the secure interoperability architecture when establishing the communication mechanisms between an industrial IoT platform and the SCADA network of a SG provider. There are several protocols used by the industrial IoT communications, e.g, OPC *Unified Architecture* (OPC-UA), guided by the standard IEC 62541 [280]) or *Message Queueing Telemetry Transport* (MQTT), guided by the standard ISO/IEC PRF 20922 [281]. In our example this IoT platform uses the MQTT protocol. The SCADA system of the SG provider uses the industrial protocol DNP3 [14].

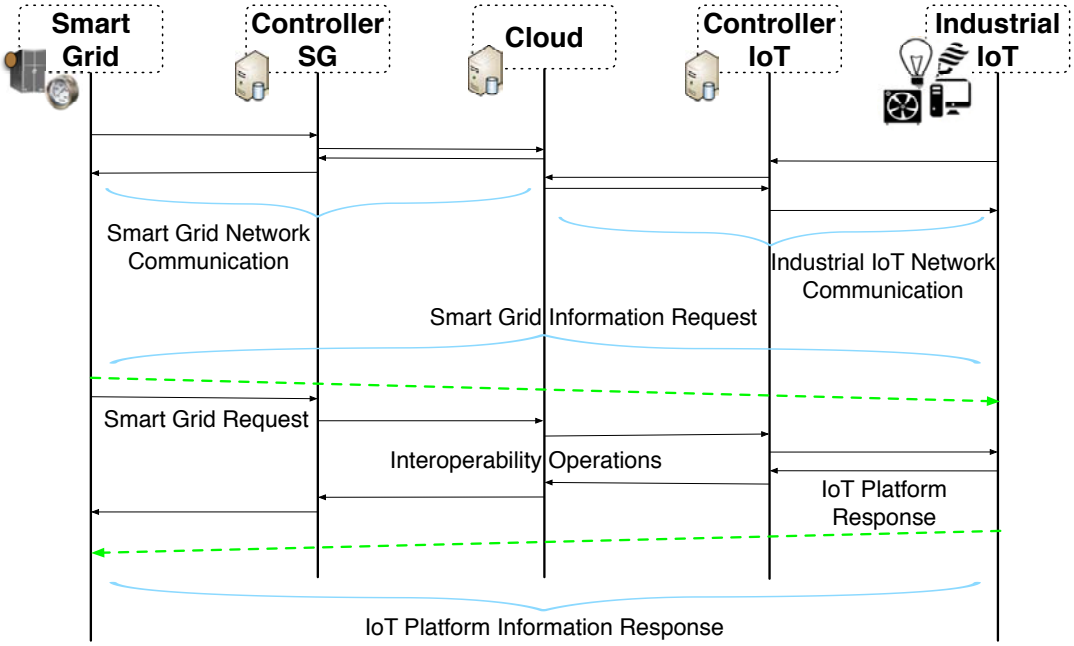


Figure 5.9: The IoT communications scenario

Figure 5.9 illustrates this example. A SCADA server authenticated within the interoperability platform requests information from the IoT platform which also belongs to the platform. This request goes through the controller in the area of influence of the server and it identifies the need of a remote connection to the IoT platform, as well as a conversion of protocol. The local controller in the area of the SG server does not implement a protocol conversion between the MQTT and the DNP3 protocols, therefore, it sends the cloud the request to update the local database of protocol and policy conversion and forwards the first communication packets to the cloud to handle them.

The cloud performs the protocol and policies conversion for this communication, and establishes that the context is not critical and the operations are authorized according to the RBAC properties assigned to the actors involved in the communication. Then the cloud forwards the reconstructed packets to the controller in the area of the industrial IoT platform, while monitoring the behavior of the two networks at a global level. Each of the local monitors at the controllers also monitors the context of their subnetwork and the dynamics in their area in order to detect any abnormal or threatening events. The controller in the IoT area delivers the packets and

awaits for the response communications. The process for the response would be symmetrical, until the configuration is updated in the controllers involved in the communication, and then the cloud only intervenes in this process as a monitor and logger for the connection.

Cyber Stealth Attack Scenario

In this second case of example, there has been a cyberattack compromising the controller in the remote control field network of provider A in the federation of SG providers. This attack's objective is to extract sensitive information from the controller, through the establishment of a side-channel timing attack (see Chapter 2). This kind of stealthy attack is highly difficult to detect for a single IDS perspective, however in this secure interoperability platform, the network dynamics of the affected node are monitored from their neighboring nodes and the cloud.

Thanks to this, the system is able to detect the attack and determine the possible countermeasures applicable to palliate the situation. Given that the system's context is not critical (established by the context manager at the local and neighboring areas), there are sufficiently robust backup mechanisms deployed in the system, and the risk assessment of the situation indicates that the attack poses a higher threat to the system than its countermeasures, the IPRS at the global monitor acts by isolating the affected controller.

This means the remote control field network of provider A is disconnected from the interoperability platform, and in order to regain access to the platform, the nodes in this network must connect to the platform through a neighboring controller. The nodes in this field network then try to access the controller in charge of the SG network of provider B. Since provider's A nodes are not usually in the area of influence of provider's B controller, it needs to send a petition to the cloud to download the appropriate information to authenticate and authorize the new actors in the system. This collaboration is enabled in the system thanks to the delegation procedures implemented in the architecture (see Section 5.3.7), and it is maintained until the system administrators restore the affected controller. Figure 5.10 illustrate this scenario with a sequence diagram.

5.6 Summary

The contents of this section is focused on proposing a design in the form of an architecture model, for the secure interoperability of CPCSs of the smart grid. In this scenario we contemplate the interactions and cooperations among different SG providers, which exchange critical information and share the management and maintenance of some of the infrastructure systems in common, and most importantly, they share the responsibility for the assistance and care of the systems in the presence of critical events, for protection, mitigation and response and restoration actions.

In order to provide this architecture model, we base our design on the wide analysis previously presented in the Chapters 3, 4 and 2 of this thesis, where we extract the main requirements, recommendations and techniques for achieving secure interoperability between critical systems and define a threat model for the platform. Since this scenario contemplates different and heterogeneous technologies, we define our architecture as belonging to the Industry 4.0 paradigm, where

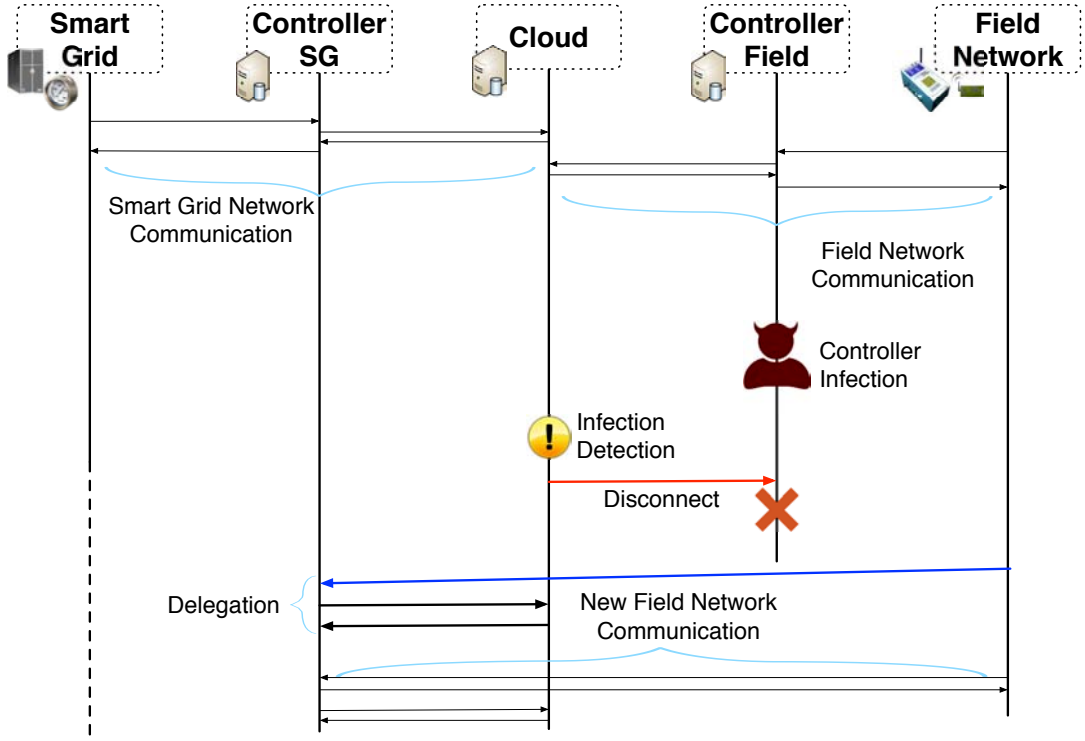


Figure 5.10: The cyberattack communications scenario

CPCSs, cloud technologies and (industrial) IoT devices coexist and interact in a cooperative way. In addition to the previously extracted guidelines and recommendations for the design, we also take into account diverse standards for interoperability and security as mentioned in Section 5.1.

Thus, we propose an architecture model for secure interoperability of CPCSs, as reflected in Figure 5.3, which makes use of the following main characteristics and technologies:

- Access control mechanisms, in order to authenticate actors within the system, always in line with the basic security services identified in Section 4.1. To provide access control, we make use of the RBAC standard defined in the IEC/TS 62351-8 [28].
- Software defined networking-based solutions, for the inclusion of new sophisticated mechanisms at the level of subnetwork, which enable new and flexible routing solutions, interoperability procedures, and advanced prevention, awareness and protection solutions (see Chapter 4).
- Cloud and Fog computing infrastructures, which allow the introduction of virtualized services, the inclusion of sophisticated routing solutions based on the SDN paradigm, additional support for interoperability procedures, as well as means for enhancing the survivability and reliability of a working interoperability infrastructure for the CPCSs of the SG.
- A defense in depth strategy that implements security procedures at different levels, ensuring the basic security services are always observed in the platform (see Section 4.1), which

include the utilization of a sophisticated diagnosis system across the platform, based on the IEC/TS 62351-7 [142]. This standard is in charge of the specification of the NSM objects, which are deployed within the infrastructure and retrieve monitoring and contextual information from the platform in order to supply guide the interoperability procedures.

- The implementation of delegation mechanisms in order to ensure that critical events are always and rapidly attended in the infrastructure, even when the assigned personnel to take care of these situations is not available. In order to implement this, we implement the RBAC model with primary (regular) and secondary roles for emergency, which can be activated in case of emergency. The use of secondary roles, as well as the delegation mechanisms with PRE techniques allow different members of the federated providers of the SG to take care of critical events arising at any part of the secure interoperability platform, providing robust protection and mitigation mechanisms for the infrastructure.

Thanks to these mechanisms, we provide an architecture (see Figure 5.3), based on two main interoperability components residing in the controllers (see Figure 5.4) and in the cloud (see Figure 5.5), which apply the aforementioned recommendations and technologies. The operation of this platform is detailed in Section 5.5 where we include two different cases of use of the platform for the Industry 4.0 and to illustrate the capabilities of this architecture when facing a cyber (stealthy) attack (see Chapter 2). Therefore this platform, as described previously, achieves the complete desired functionality for the secure interoperability of CPCSs, always observing the requirements for CIs and interoperability of critical systems described in Chapter 3.

Chapter 6

Experimentation and Validation of the Architectural Model

This chapter is devoted to the analysis and validation of the secure interoperability architecture model designed in the previous section, through the performance of various experiments. In the first sections we establish the scope of the validation of our architecture model provided in Chapter 5, and design a set of validation cases to test the proposed system through experimentation in different scenarios. In Section 6.3 we describe the different modules and features of the testbed we create for the validation, which is based on a network emulator capable of integrating real communication's protocols and therefore real network traffic. Section 6.4 provides the results of the different experiments performed in the testbed, and details the key modules implemented for the tests describing the processes followed and the resulting outcomes. In this section we also provide a performance analysis to better understand the system dynamics and help determine whether the solutions provided by our architecture model are appropriate for a critical environment.

6.1 Scope of the Validation

In Chapter 5, we propose a design for a secure interoperability architecture model for the CPCSs of the SG. Its complete design is illustrated in Figure 5.3, where it is possible to appreciate the different components and services present in the infrastructure, according to the previously identified requirements and recommendations framed within a transitional version of the Industry 4.0 paradigm. In order to test this proposed design we need to perform a series of experiments capable of determining if the technologies and services included in the architecture model are suitable for critical constrained scenarios and support the interoperability processes properly.

The secure interoperability architecture model features two main services modules residing in the controllers of the infrastructure (see Figure 5.4), and in the cloud servers (Figure 5.5), as described in Section 5.4. As we can observe in the design provided in Chapter 5, the main functionalities for interoperability are concentrated in the controller's modules of the infrastructure (see Figure 5.4), being the operations performed in the cloud mainly supporting operations for

the controllers, backup operations, and heavy-duty computing related to big data and streaming data analysis.

With these characteristics in mind, we decide to focus our experimentations on the main security and interoperability functionalities performed by the controllers in the platform. The main reason behind this decision is the utilization of relatively new technologies at the controller's level, such as the inclusion of the SDN paradigm (see Section 5.3.4), which introduces new challenges and difficulties in the implementations of the interoperability functions, and the uncertainty related to the suitability of the use of SDN for critical scenarios in terms of performance and reliability. Cloud technologies have been long studied, and the functionalities offered by the secure interoperability platform at the cloud's level have been evaluated in other works, such as delegation mechanisms [282] (see Section 5.3.7), or the IDPRS solutions for the cloud [283] (see Chapter 4).

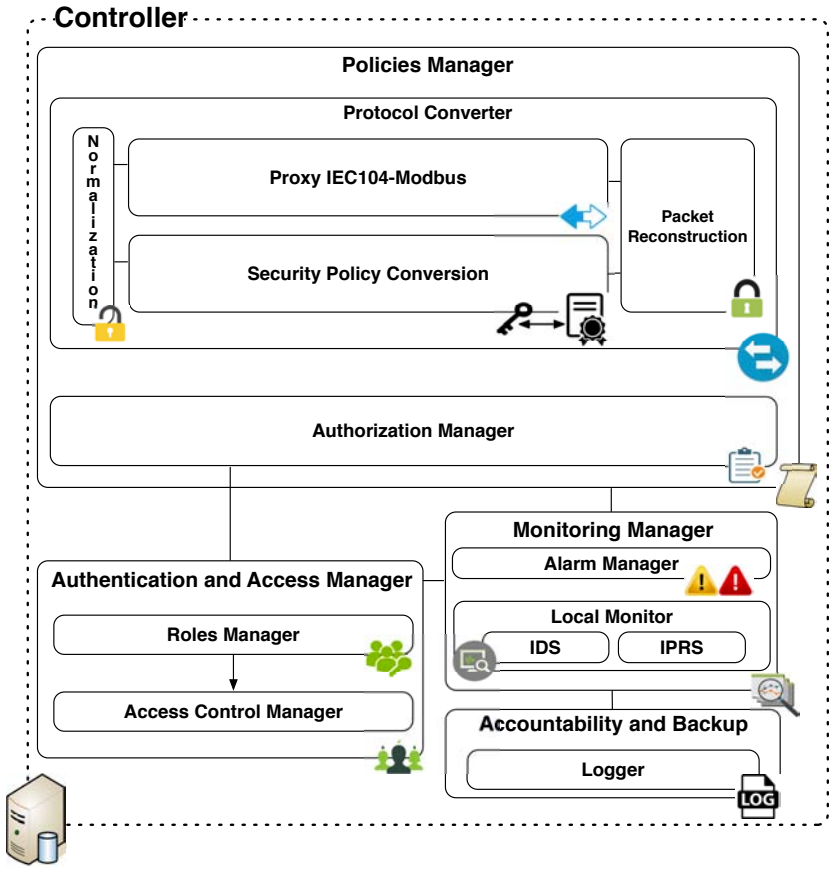


Figure 6.1: Simplified controller

We identify as important services for verification: the *protocol converter*, the *authorization manager* (together with its related services in the AAM module (see Section 5.4)), and the *monitoring manager* with its *local monitor* and the *alarm manager*, together with the *logs* module. These services are illustrated in Figure 6.1, which represents the selected modules for our simplified controller prototype. Therefore, since the prototype controller provides the

implementation of the main interoperability methods, it is possible to assess the behavior of the complete controllers at the secure interoperability infrastructure, by analyzing the operation and performance of the simplified module.

There are several services that we consider less important to validate, since we believe they add little value to our analysis of secure interoperability for critical scenarios. For example, authentication procedures, which are widely available in most information technologies systems (and they already are widely present in most CIs), have a well known behavior in critical environments [284, 285]. Therefore, we believe that adding these services to the testbed would not be very useful for the interoperability validation purposes, and would make the implementation and use of the testbed more complicated.

In the context of the advanced security services, it is our aim to provide an overview of the functionalities offered by the monitoring manager's IDPRS, however, it is not our purpose to install a complex and complete IDS solution with its response module, especially designed to work in a critical setting. This is due to de fact that currently (as explained in Section 4.2), detection and protection solutions for critical systems are subject of current intense research, where the scientific community is trying to provide efficient solutions capable of performing properly in a critical scenario. We, therefore, provide a proof of concept for the validation cases, but we do not intend to tackle the task of building a complete IDPRS for our simplified interoperability testbed.

The diagnosis system is a module which would result in a challenging implementation for our experimentation due to the characteristics of the fully-functional diagnosis system described in Section 5.3.6. This system requires the deployment of NSM objects within all the components and devices present in our testbed. However, since our focus for these experiments are centered on interoperability for a reduced set of use cases, we find that we can include and simulate in the testbed a reduced version of this module, rather than a fully-fledged diagnosis system based on the IEC/TS 62351-7 standard [142] as proposed in Chapter 5. If we focus on the use of NSM objects for the identification of threatening dynamics occurring within the system to feed information to protection systems, it is possible to find several works which feed data to the IDS and security systems using this technology [248, 286].

Additionally, we also consider that the implementation of all the backup activities to the cloudlet servers and the cloud in our testbed experimentation would not result of interest for our tests. These backup and accountability processes are very well known and are currently employed in the majority of IT systems [287, 8, 264], therefore we believe they would provide little added value to the verification of the functionalities of our architectural model.

For our interoperability tests, we need to do experiments using two or more industrial protocols in order to assess the services that provide the protocol conversion. For our tests we have chosen two industrial protocols, the *Modbus/TCP* protocol [265] and the *IEC 60870-5-104* or *IEC104* protocol [7]. These protocols have been widely used in the industry for long years, and they have still a strong presence in today's industrial networks. These protocols have a cabled nature, they are standards (Modbus is a de facto standard), and their interesting characteristic for our testbed is that without any modification to the original protocols, they do not implement security encoding mechanisms, therefore the messages are transmitted in clear. This provides our testbed with networks which use real industrial protocols, which messages are open and easy to understand and manipulate with the analysis tools, without introducing any modifications to

the protocols in order to do so.

Therefore, given these considerations, it is possible to design and analyze a series of experimentation scenarios, which allow us to test and understand the secure interoperability capabilities of a simplified version of the controller services modules designed for our secure interoperability architecture model. By testing the services present in the simplified controller in Figure 6.1, we believe it is possible to gain sufficient insight about the properties and performance of the proposed architecture model (see Chapter 5), in order to determine if the secure architecture infrastructure provided is suitable for the interoperability of critical systems.

6.2 Design of the experiment: Scenario and Validation Cases

Therefore, taking into account the considerations provided in Section 6.1, we present a testbed scenario for three different validation cases which intend to analyze the performance of the main interoperability services provided by the controllers and determine their suitability for their original purpose. Our scenario can be described, therefore, as a platform which enables the secure interoperability among different actors belonging to the smart grid (see Figure 6.2), within this complex and heterogeneous scenario, we choose to study the interactions that can occur within the environment of a hydroelectric dam belonging to the smart grid, and managed and exploited by two different energy providers (A and B), which are partners in the same federation of energy providers (see Figure 6.3).

Each of the providers has a control and monitoring network deployed in the dam in order to gather their own supervisory and control data. However, in order to minimize costs, part of the substation (cloudlet, see Section 5.3) equipment is in a shared regime, especially the cloudlet's main servers and hence the network controllers that manage the different providers' network functions. In this scenario, we consider that provider A owns a supervisory and control network that implements the IEC104 protocol (its characteristics are elaborated in Section 6.3.4) to allow the communications between the different CPCSs (industrial sensors, RTUs, personal computers, remote operations, etc.).

The energy provider B has a network deployed for the control and supervision of the CPCSs of the dam that uses the Modbus/TCP protocol (for further information see Section 6.3.3). These providers manage their own equipment separately but in a cooperative way, allowing interactions between the different CPCSs in order to gain robustness and security through the redundancy, replication, diversity and self-awareness techniques (see Section 3.4).

In the validation scenarios we contemplate in this section, the control system owned by provider A verifies the rate at which the water is being released from the dam in order to produce energy. Since this procedure is critical for the dam, e.g., too much water release can cause floods in neighboring areas, too much water contained can cause damage to the dam due to water pressure, the control equipment must make constant verifications that the obtained measures are correct. In order ensure that the taken measures are correct and prevent anomalies, provider's A CPCSs periodically request provider's B CPCSs verification measurements that allows the control systems check their correct operation and face errors such as sensor degradation, malfunctions, water theft or other malicious cyber (stealthy) attacks (see Chapter 2). Additionally, these inter-provider verifications can be launched as protection and response measures in case

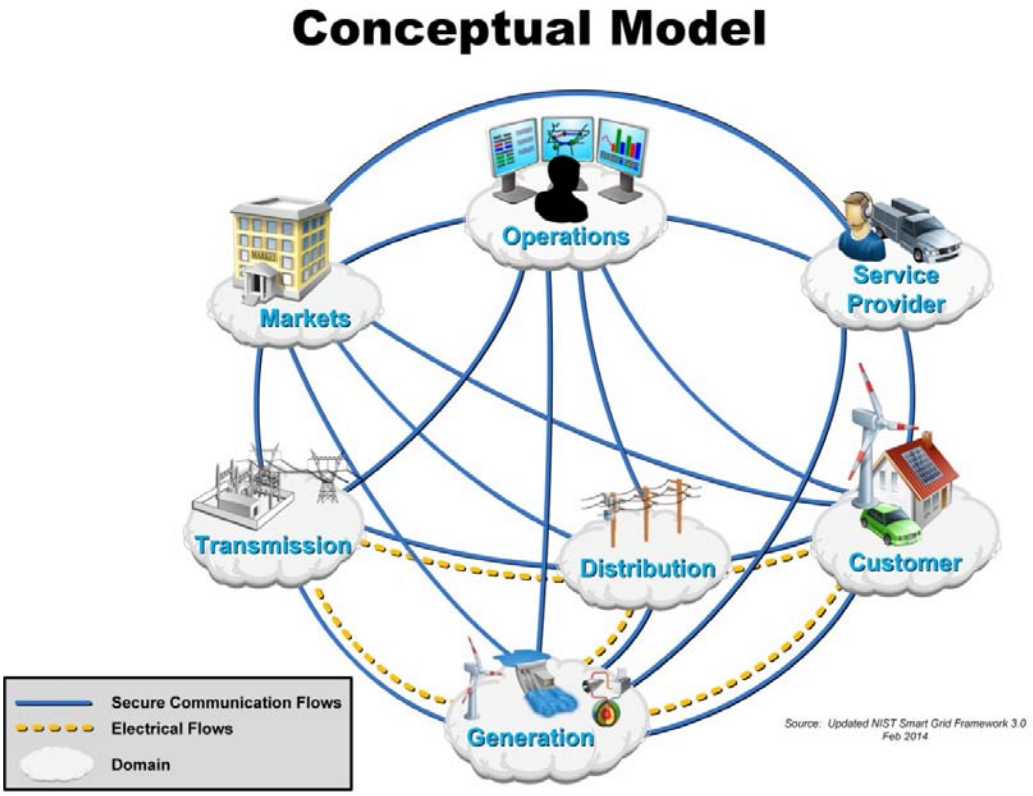


Figure 6.2: The NIST smart grid model [6]

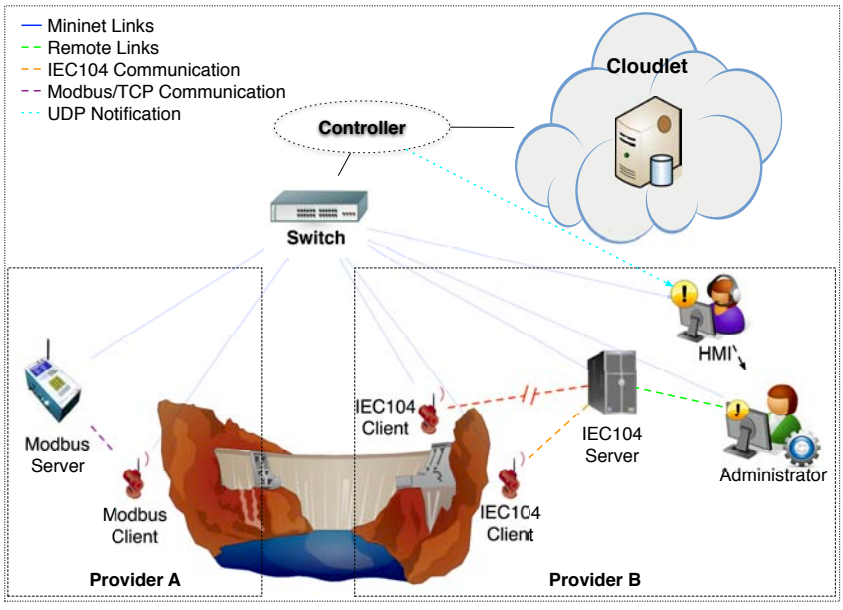


Figure 6.3: Testbed scenario for the secure interoperability infrastructure

of cyber attacks. In the remainder of this section, we describe three main validation cases which depict different scenarios for experimentation and assessment of the capabilities of our simplified controller for interoperability.

6.2.1 Validation Case 1: Protocol Converter

The *Policies Manager* module present in our controller (see Figure 6.1) is composed of two main components (see Section 5.4.2 for further information), the *Protocol Converter* and the *Authorization Manager*. This first validation case is focused on the functionalities and modules implemented by the protocol converter. This validation case is based on the aforementioned scenario set in the previous section (see Section 6.2). The main actors present are (see Figure 6.4):

- Provider A: subnetwork of CPCSs communicating in the IEC104 protocol.
- Provider B: subnetwork of CPCSs communicating in the Modbus/TCP protocol.
- Federation equipment: a virtual switch with its corresponding SDN controller.

In this scenario, the CPCSs belonging to the Provider A are taking measurements on the rate that the water is released from the dam and transmitting it to the control station through a IEC104 client-server communication. In parallel, Provider B CPCSs' are also monitoring the water release using its own network of clients and servers which communicates over Modbus/TCP. In order to make sure that everything is correct and avoid system anomalies, periodically the IEC104 client opens a connection with the Modbus server (RTU), to request the measurements taken by the Provider's B sensor network.

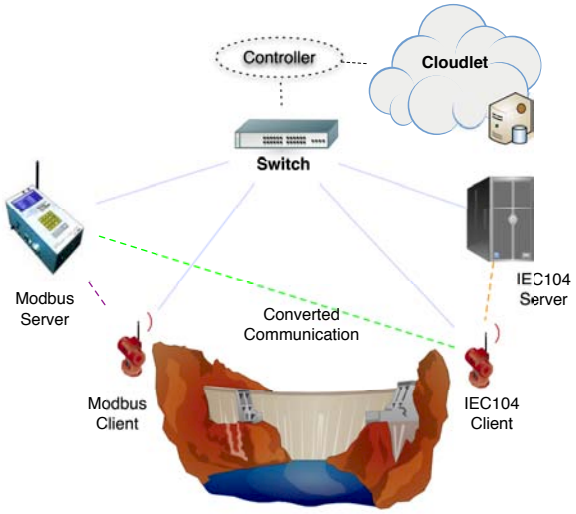


Figure 6.4: Converted traffic between nodes that use different communication protocols

This communication is facilitated in a transparent way by the SDN controller, which receives the requests from the different subnetworks and performs the protocol conversion in order to allow

this inter-protocol communication. Once the IEC104 client verifies that the measurements are correct, closes this connection with the other provider's network and continuous to work with its own control network. These periodic verifications provides the CI with additional security mechanisms based on the interoperability and cooperation of different control infrastructures.

6.2.2 Validation Case 2: Monitoring Manager

The second validation case is devoted to analyze the *Monitoring Manager* and the *Accountability and Backup* modules functionality. These modules are present in the controller designed for our architecture in Sections 5.4.3 and 5.4.4, and for our study, we have simplified some of their functions (see Figure 6.1). We validate these two services in this same scenario since they are highly interrelated. Therefore, in this section we have two main case studies for validation: the validation of the *Logger* module, and the validation of the *Local Monitor* together with the *Alarm Manager* module. The main actors involved in this scenario are (see Figure 6.5):

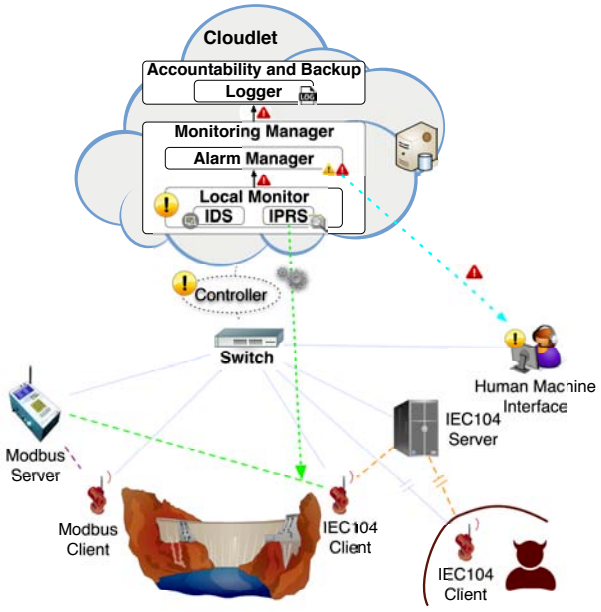


Figure 6.5: Disconnection attack

- Provider A: subnetwork of CPCSs communicating in the IEC104 protocol.
- Provider B: subnetwork of CPCSs communicating in the Modbus/TCP protocol.
- Federation equipment: a virtual switch with its corresponding SDN controller.
- Federation equipment: a database of system logs.
- Federation operator: a system operator monitoring a *Human Machine Interface* (HMI) where the system alerts are sent.
- Malicious actor performing cyber attacks.

To validate the logging module, the scenario is set as follows: the CPCSs belonging to the Provider A are taking measurements on the rate that the water is released from the dam and transmitting it to the control station through a IEC104 client-server communication. In parallel, Provider B CPCSs' are also monitoring the water release using its own network of clients and servers which communicates over Modbus/TCP. The SDN controller provides logging functionality, saving any event in the system with its timestamp and associated criticality level. As we discussed previously, in our scenario of validations, the logger module is tightly interrelated with the monitoring manager, therefore we test its operation by launching events which are first detected and handled by the monitoring manager module and then logged into the system for accountability.

For the validation of the monitoring manager module, we take aforementioned scenario of communications, and launch various cyber attacks against the system with varying degrees of stealthiness (see Chapter 2). Each of the following settings shows the application of a cyber attack against the testbed and the behavior of the system in the validation case:

Disconnection and Goodput Reduction Attack

The first type of attack we launch is one with a low degree of stealthiness, we perform a *disconnection* attack (see Section 2.3.1) in which the attacker disconnects one of the industrial sensors belonging to the control network of provider A, causing it to stop transmitting. This is detected by the IDS module in the controller, which analyzes this behavior, launching protective measures. The first measure is made through the alarm manager module, which sends an alert to the system operators monitoring the HMI, in order to re-establish the normal operation of the system. The second measure is taken by the IPRS module, which launches a petition to the Modbus server (RTU) in Provider's B sensor network, to request the measurements of the dam while the system is not working properly. This attack is illustrated in Figure 6.5.

Active Eavesdropping Attack

The second attack scenario reflects the performance of an active eavesdropping attack (see Section 2.3.2). It is done by introducing a Man-in-the-middle attack (MITM). In this setting, the malicious actor impersonates a node of the network using spoofing techniques, and tries to perform different control actions in the system to extract information. They are detected by the IDS module of the monitoring manager, and protective measures are launched to protect the system. Firstly, as in the previous case, an alarm is sent to the system operators monitoring the HMI. Secondly, the IPRS module filters non-authorized operations and traffic generated by the attacker through a whitelisting procedure. The attack is illustrated in Figure 6.6.

Covert channel attack

In the last example, we launch a stealth attack against the testbed by implementing a covert channel (see Section 2.3.4) between two nodes of the network. In this case, we work with a scenario where a sophisticated malware has infected several nodes of Provider's A network (see Figure 6.7). The objective of this malware, which operates in a similar way to Stuxnet [48], is to

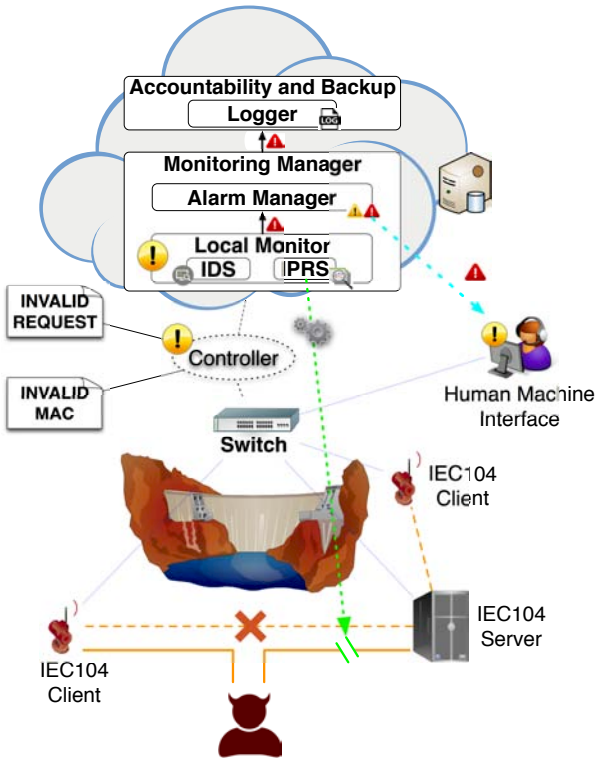


Figure 6.6: Active eavesdropping attack

stealthily degrade the operation of Provider's A infrastructure, reducing the availability of the control infrastructure, in order to cause disruptions and economic loss. This malware targets the IEC104 servers and clients of Provider's A control infrastructure.

The infection of these systems has the objective of making the RTUs and PLCs under its influence to selectively and stealthily turn off the industrial sensors under their control to simulate system failures, hence reducing the availability of the sensor nodes and producing interruptions in the supervision and control procedures. To launch such an attack in a stealthy way, an infected IEC104 RTU gathers information from compromised IEC104 sensors via a *covert channel*. In our attack scenario, the compromised RTU uses this covert channel to retrieve the number of hours each sensor has been working without interruption (uptime), in order to better select the sensor to turn off so this action goes unnoticed by the system operators. This dynamic is reflected in Figure 6.7.

Since the malware is gathering unauthorized system information by hiding it within the IEC104 packets, the IDS in the controller's local monitor detects it and alerts the system operators in the HMI. The IPRS launches a countermeasure action based on whitelisting which filters the packets belonging to the covert channel's communication. This filter allows all legitimate network packets to continue their route, while it sanitizes the packets which carry the covert channel's information by removing the information. This is done until the system administrators remove the malware from the infected devices.

All the events produced by each attack scenario are also categorized as critical and saved to

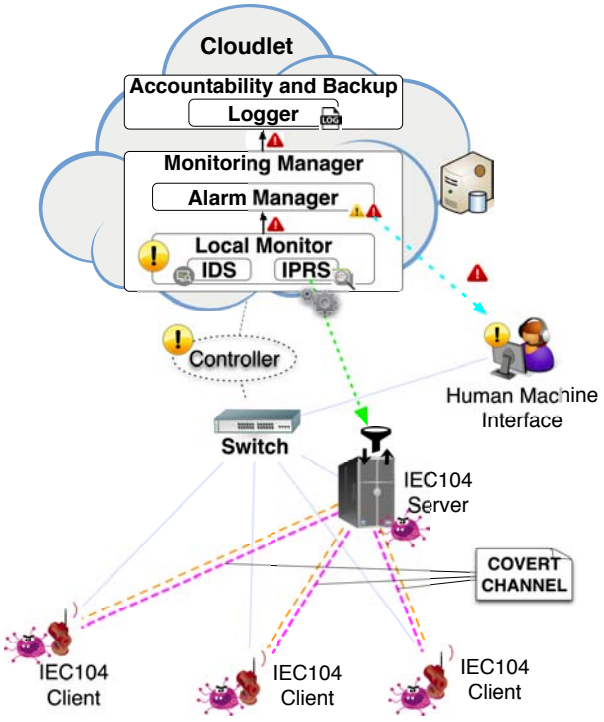


Figure 6.7: Covert channel attack

the logging database. These accountability procedures are useful for the validation of the logger module.

6.2.3 Validation Case 3: Authorization and Access Manager

The last validation scenario studies the *Authorization Manager* present in the Policies Manager (see Section 5.4.2), thus completing the validation study of this module together with Validation Case 1 in Section 6.2.1; and also analyzes the *Authentication and Access Manager* module of the controller (see Section 5.4.1). Figure 6.8 illustrates this combination of functionalities, which we call *Authorization and Access Manager* (AuthAM), where the main actors in this scenario are:

- Provider A: subnetwork of CPCSs communicating in the IEC104 protocol.
- Provider B: subnetwork of CPCSs communicating in the Modbus/TCP protocol.
- Federation equipment: a virtual switch with its corresponding SDN controller.
- Federation equipment: a database of system logs.
- Federation operator: a system operator monitoring an HMI where the system alerts are sent.
- Federation remote administrator: a system administrator that remotely connects to the substation to perform analysis and maintenance tasks.

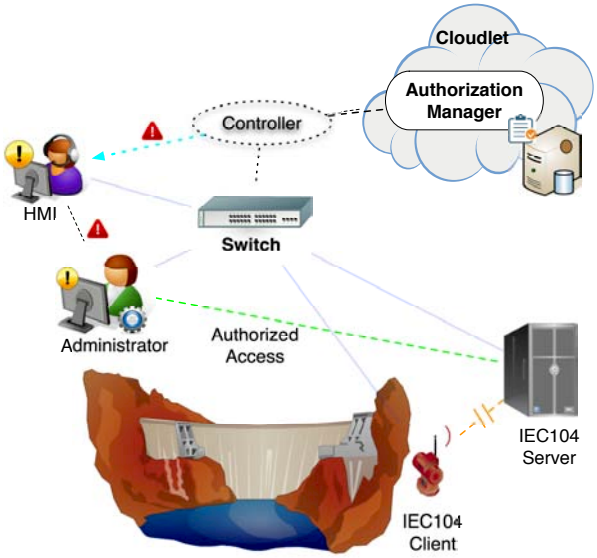


Figure 6.8: Authorization and access manager in the testbed

To validate this functionality, we set the scenario as in Section 6.2.1, where the CPCSs of the different providers communicate among them. However, we introduce a failure in one of the sensors of Provider A, which makes it to stop transmitting the measurements to the control station, thus creating a system anomaly. This is rapidly notified to the HMI, and communicated to the remote administrator to perform the proper analysis and maintenance tasks. This operator need to establish a connection with the IEC104 server of Provider A in order to retrieve specific logging information.

In order to manage the petitions to the critical CPCSs, the cloudlet implements an access control procedure based on the role-based access control standard (see Section 6.3.5 for expanded information). When an access petition arrives to the controller, it verifies the credentials sent (by the remote system administrator in this use case) and matches it to its databases, through the *Roles Manager* module and the *Access Control Manager* module (see Section 5.4.1). This verifications determine if the administrator has de proper credentials to perform the requested tasks, and depending on the situation, allow or reject the connection and petition to the critical resource.

6.3 Definition of the Testbed: Modules and Libraries

We devote this section to describe those tools, protocols, developing modules and programming libraries or relevance used to build our testbed.

6.3.1 Mininet

For our experimentation, we have decided to implement our testbed using *Mininet* [288]. Mininet is a network emulator able to create a network of virtual hosts, switches, controllers and links, where the switches support OpenFlow protocol [273] in order to provide SDN capabilities (see Section 5.3.4 for more information). Mininet networks run real code, including standard network applications and protocols. Due to this characteristic, it is posible to develop a testbed on Mininet for an OpenFlow controller, switch or host and move to a real system with minimal changes for real-world testing, performance evaluation and deployment. This means, that it is possible to move directly into HW a design correctly working within Mininet [288]. Consequently, the development of a testbed using this network emulator makes it possible to make real-world validations of network functions and services, thus we consider this tool to be highly interesting for the construction of our testbed and the validation of our design.

Within Mininet, there is a simple network simulator graphical user interface, *Miniedit*. It is possible to use this tool to create and run network simulations from a visual perspective, allowing us to configure a simple network for the interoperability testbed, with two subnetworks that use different communication protocols, a virtual switch (vSwitch) [289] and an SDN controller.

6.3.2 POX OpenFlow Controller

The controller is one of the most important components in our design, since the validation of our architecture is based on the functions implemented by the controllers in the cloudlets. We make use of the development platform POX [290] to implement the different components of our controller. POX provides a platform to develop Python-based SDN control applications (such as OpenFlow [273] controllers) supporting the interaction with OpenFlow switches, debugging, network visualization, etc.

Since POX is Python-based, it is especially interesting to use within Mininet, since this emulator runs Python-coded applications. Therefore, we find the POX development platform interesting for our testbed, since it allows the user to rapidly develop and test a versatile SDN controller prototype. In our testbed, we build the designed functions and capabilities (see Figure 6.1 and Section 5 for more information) within the POX controller we develop for our validation studies.

OpenFlow Protocol

OpenFlow, as stated in Section 5.3.4, is an open communications protocol that allows programming the flow table of (OpenFlow-enabled) routers and switches [273]. This protocol separates the control from the forwarding plane, and therefore allows sophisticated traffic management and network functions. This protocol allows direct access to the forwarding plane of virtual or physical network devices. OpenFlow is the first standard communications protocol developed to be used between the control and forwarding layers of the SDN architecture (see Section 5.3.4).

OpenFlow works on top of the Transmission Control Protocol (TCP), it allows the controller to remotely administrate a layer 3 switch's packet forwarding tables, by adding, modifying and

removing packet matching rules and actions. This characteristic entails that routing decisions can be made ad hoc by the controller, and translated into rules installed in the switch's flow table for a determined period of time. Those packets that do not match the switch's flow table are sent to the controller to decide whether to modify the existing flow table rules or to perform additional actions (e.g., drop the packet, deep packet inspection [291]).

6.3.3 Modbus/TCP Protocol

Modbus is an open serial communication protocol developed by Modicon in 1979 [277], which has become a de facto standard widely used in the industrial manufacturing environment. It provides master-slave/client-server communication between intelligent field devices for industrial control applications. Especially, it is often used to connect RTUs with SCADA systems. The Modbus/TCP protocol [265] is a variant of the Modbus protocol used to enable Modbus messaging over TCP/IP networks.

Modbus/TCP protocol is also a de facto standard, commonly used to connect PLCs or RTUs to SCADA systems using Ethernet connections that have to go through gateways or other types of networks. Advantages of Modbus/TCP are the possibility of implementing concurrent connections, and the control of each individual transaction enclosed in a TCP/IP connection, which can be identified, supervised and canceled without requiring specific actions from the Modbus client or server. Figure 6.9 represents the different fields of a Modbus/TCP packet.

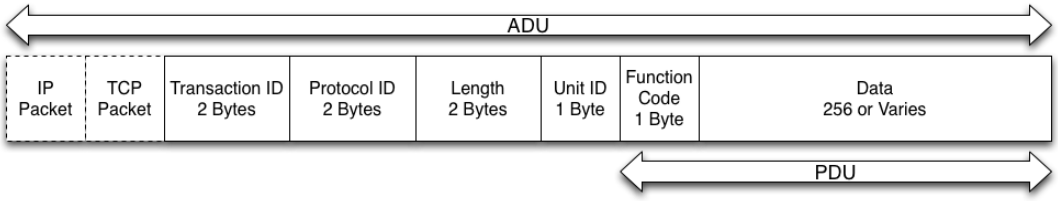


Figure 6.9: Modbus packet

Pymodbus Library

In order to include the Modbus/TCP protocol within Mininet, we make use of the Pymodbus library [292]. Pymodbus is a full Modbus protocol implementation written in Python, which allows easy scripting and stress testing of the developed solutions. This library also permits the integration of Pymodbus-developed solutions into already existing Modbus solutions. Using the Pymodbus library, we have developed a Modbus/TCP client and server which communicate with each other in an asynchronous way, where the client requests values from the server through the Modbus command `read_input_registers`.

6.3.4 IEC 60870-5-104 Transmission Protocol

The IEC 60870-5-101, or IEC101 [266] is a standard protocol for designed for power system monitoring, control and communications for remote control and protection. The IEC101 belongs to

the set of standards IEC 60870 for telecontrol (SCADA) in electrical engineering and power system automation applications, where its Part 5 is devoted to the standardization of the communication between systems. IEC101 supports multiple configurations and topologies, balanced and unbalanced modes of data transfer, priority classification of data, time synchronization and other characteristics for control of the electric power systems.

The IEC 60870-5-104, or IEC104 [7] is an extension of the IEC101 protocol, enabling communication between the control station and the remote substations via a standard TCP/IP network. The IEC104 protocol layers are modified to use standard transport profiles, but the application layer is based on IEC101, however, not all the functions available in the later are implemented for the first. The main advantage of IEC104 is the communication via a standard network, which allows simultaneous data transmission between different devices and services. Figure 6.10 represents the different fields of a IEC104 packet.

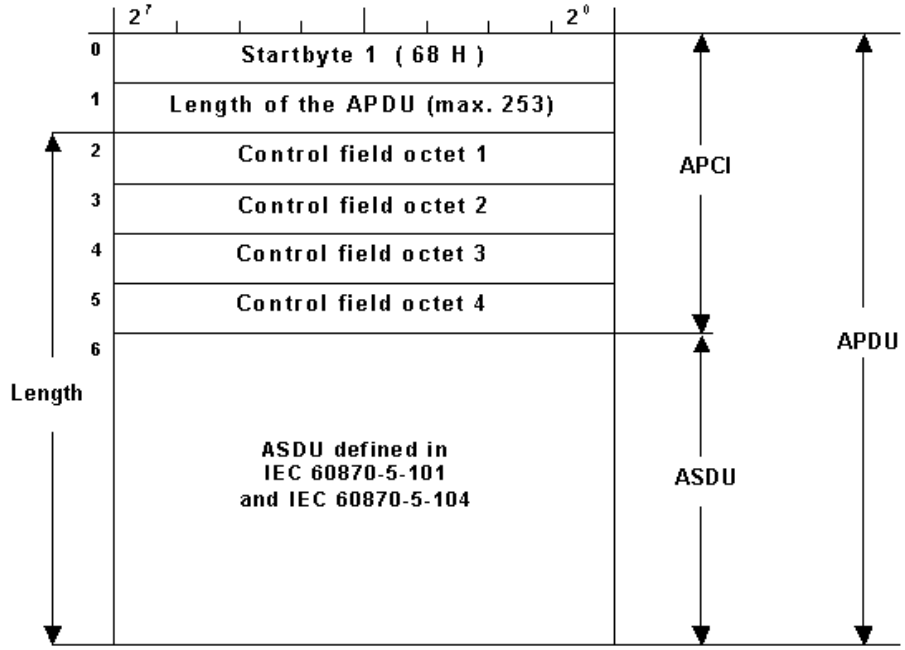


Figure 6.10: IEC104 packet [7]

One of the main commands used in the IEC104 communications is the "Interrogation Command", the *Application Service Data Unit* (ASDU) Type ID 100 `C_IC_NA_1` where the controlled station (RTU) will send the the controlling station (SCADA system) information, in order to synchronize or update the database. In this protocol, the RTU responds to the request with the corresponding values through the ASDU Type ID 11: `M_ME_NB_1`, which provides measured scaled values.

OpenMUC: j60870 Library Module

In order to introduce the IEC104 protocol [7] in our testbed in Mininet, we make used of the *OpenMUC: j60870 library module* [293]. This library is written in Java, and implements

the IEC 60870-5-104 (IEC104) communication standard. The j60870 can be used to program IEC104 clients and servers. This library belongs to the OpenMUC software framework, which helps develop monitoring, logging and control systems, providing the developer with the tools to implement controlling applications without needing to know the intricacies of the underlying communication protocol and data logging technologies.

The OpenMUC framework developed by Fraunhofer ISE has been used as a basis in various smart grid projects. In our testbed, we only make use of the j60870 library module, which can be utilized separately. Since this library is developed in Java, to include it within Mininet, we need to develop separate Python modules to launch the different clients and servers for our testbed.

6.3.5 IEC/TS 62351-8 - RBAC Standard

In our designed architecture, one of the main components for secure interoperability is the Authentication and Access Manager (see Section 5.4.1). As mentioned in Section 5.3.1, one of the main standards that support the access control is the IEC/TS 62351-8 [28] which defines role-based access control for enterprise-wide use in power systems. This widely used standard is an access control mechanism defined around roles and privileges. RBAC components: role-permissions, user-role and role-role relationships, make it simple to perform user assignments.

In an organization, a set of roles are created in relation to a set of job functions. Those roles are assigned a set of permissions. The company's staff are assigned one or several roles (see Section 5.4.1). Thus a member of the organization is assigned a role (or a set of roles) which have assigned a set of permissions to perform certain tasks within the system. The management of this structure is simple, given that permissions are not assigned directly to users, but to roles, and the security management is therefore not conducted at an individual level but through groups of roles.

In order to introduce RBAC into our testbed, we make use of the *Simple-RBAC* library [294]. This library, written in Phyton, allows us to introduce RBAC sets of roles, resources and permissions easily into our testbed.

6.3.6 Other Tools and Resources

In this section we describe other useful tools we have integrated within our testbed in order to achieve the desired functionalities, i.e., the *Scapy* tool, and the IEC104 and Modbus/TCP dissectors which help us extract important information from the testbed network traffic.

Scapy is a packet manipulation program, which allows the developer to forge or decode packets of a wide number of protocols [295]. Other interesting functionalities of Scapy are the performing of tasks like scanning, tracerouting, probing, unit tests, attacks, network discovery, and many other interactive procedures. Scapy is a helpful that we use to decode and dissect the packets in different protocols, to build new packets from scratch in the POX controller, and also can be used to generate malicious attacks to the Mininet network.

Using the Scapy tool as a programming library, we can perform the dissection of the IEC104 packets in our testbed's network. With this dissector, it is possible to obtain useful information from those packets, extracting the encapsulated IEC104 ASDU and reading the function codes and payload of the protocol. For our purposes, the IEC104 dissector identifies the different fields in the IEC104 packet, in particular the *Application Protocol Control Information* (APCI) fields in the header (see Figure 6.10 for more information), where the field `seq_received` contains the IEC104 code for the command type identification.

Additionally, using this same method in order to extract structured information from the Modbus/TCP packets, we can use a Modbus/TCP dissector (e.g., the*Modbus traffic generator* [296]) to manipulate Modbus/TCP packets, e.g., to extract the values obtained from the Modbus server making use of the command `read_input_registers`.

6.4 Experimentation Results

For the validation of our design, we build a testbed scenario using Mininet (see Section 6.3.1). In order to cover all the validation cases, we define a simple network with 8 hosts, a virtual switch and a SDN controller. This network represents the different actors present in a cloudlet or substation in our architecture design (see Section 5). Figure 6.11 illustrates our validation scenario designed using the Miniedit tool, where the Mininet (Ethernet) connections are represented in blue, and the OpenFlow (see Section 6.3.2) traffic is represented in red.

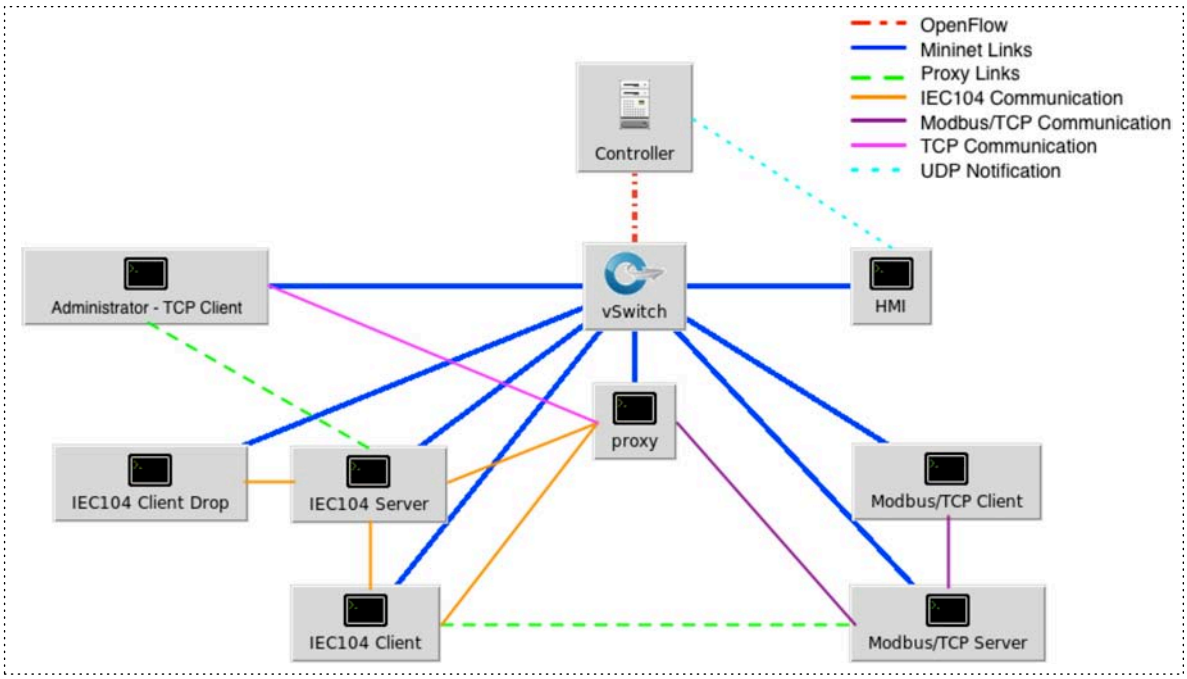
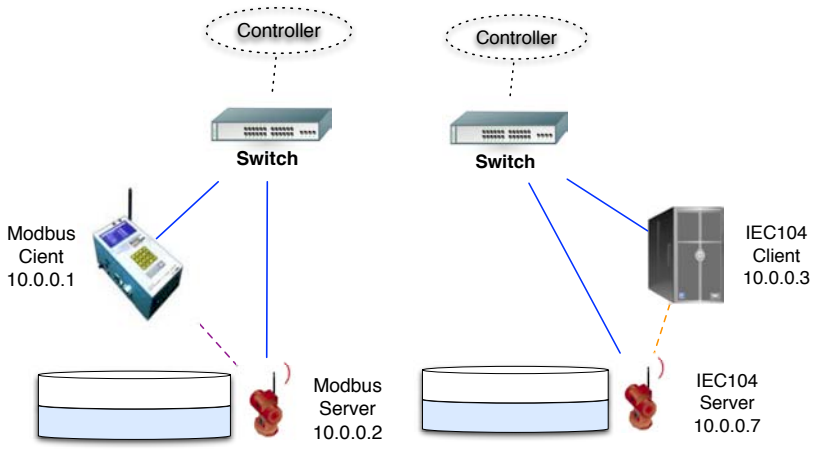


Figure 6.11: Mininet testbed scenario illustrating the communications in the network

Since this is a secure interoperability testbed, Figure 6.11 also illustrates the communications in the different protocols that coexist in the network. In purple we can see the communications

using the Modbus/TCP protocol, orange represents the IEC104 communications between the different clients and servers in the network, in pink we show the authorization communications over TCP. The dotted light blue lines represent the UDP notifications, and the green lines represent the communications through the protocol converter (see Section 6.2.1).

In detail, Figure 6.12 represents the tested scenario where it is possible to identify the Modbus communication between the Modbus client and server in the testbed and the IEC104 communication between the IEC104 client and server in the testbed (see Figure 6.11).



(a) Modbus communication

(b) IEC104 communication

Figure 6.12: Industrial protocols communications within the testbed

6.4.1 Protocol Converter

The first scenario tested is the one corresponding to the protocol converter module, in the first validation case (see Section 6.2.1). As aforementioned, this module is a fundamental part of the policies manager module of the SDN controller designed for our architecture, and is designed to allow the interoperability between different nodes in the same network, that work with different communication protocols, to interact among them and exchange information without concerns about the translation between protocols and security policies.

In Figure 6.1 we can see the SDN controller, where the main functionalities of the protocol converter module are illustrated: the *normalization* of the incoming messages, the *security policy conversion*, the *proxy* IEC104-Modbus/TCP and the *packet reconstruction* module, as described in Section 5.4.2. As we can see, the actual protocol conversion is performed through a *Transparent Interoperability Proxy* that helps translate IEC104 messages into Modbus/TCP messages.

A proxy is a computer system or application acting as intermediary in a communication between two parties, where they send requests to each other in order to gather information or resources from one another. The proxy is usually configured in a client-server way, where the client connects to the proxy server requesting a service, the proxy receives the requests and analyzes it in order to identify the best way to retrieve the information and provide it to the client. Proxies

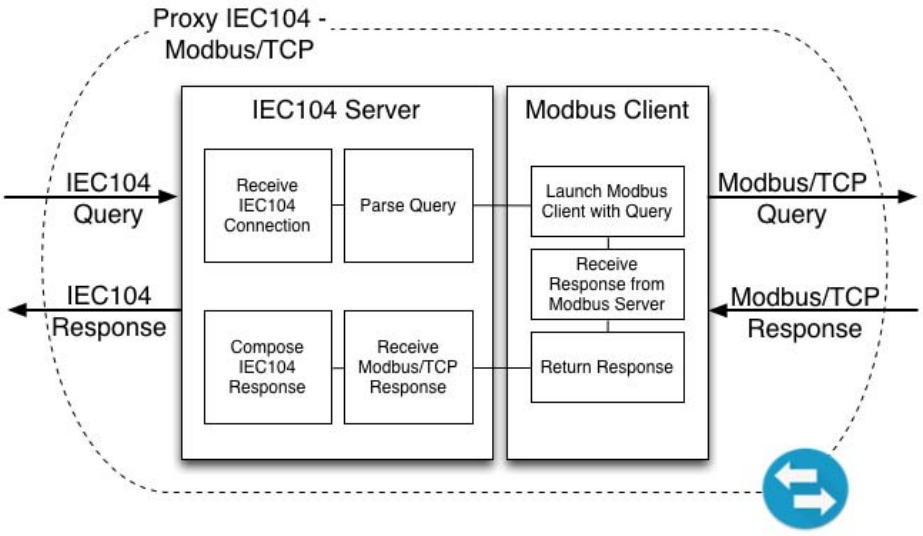


Figure 6.13: Transparent interoperability proxy

are used for multiple purposes, e.g., to provide anonymity [297] or to provide protocol conversion [298], among others.

In our testbed scenario, we are in need of the use of a proxy for the protocol conversion due to the industrial protocols used, i.e., Modbus/TCP (see Section 6.3.3) and IEC104 (see Section 6.3.4). These are protocols that use TCP in their transport layer, and are, therefore connection-oriented [299], which means that the actors establishing the communication need to open a communication session or a semi-permanent connection before any useful data can be transferred, and the data is delivered in the same order as it was sent. This characteristic of TCP causes a problem in a protocol-translation scenario, because the information exchange between the nodes connected have to follow an adequate order, and in this protocol, the nodes always keep track of the amount of data received and the amount of data remaining in the connection.

In order to consider these protocol restrictions in our testbed scenario, our transparent interoperability proxy (see Figure 6.13) is configured in a way that allows a connection with the IEC104 client, receives the query, processes it, and launches a Modbus/TCP client to retrieve the requested data from the Modbus network. In this way, it is possible to maintain proper TCP connections between the different nodes in the network through the proxy, without incurring in TCP protocol violations and NAT errors.

This configuration of the transparent interoperability proxy is similar to the approaches of *Indirect TCP* (I-TCP) [300] and the *performance-enhancing proxies* and *split TCP* mentioned in the RFC 3135 [301], where the TCP network is segmented to accomplish different purposes. I-TCP was conceived in order to provide connectivity between the wired hosts and the mobile wireless hosts in a TCP network. Split TCP breaks the TCP end-to-end connection into multiple connections to overcome and address a mismatch in TCP capabilities between two end systems. Performance-enhancing proxies are designed to improve the end-to-end performance of some communications protocols. The basic concept of these solutions consist on terminating the connection from one end system and originating a separate connection to the other end system.

The advantages of implementing this type of solution include: (i) they do not require changes to TCP at the hosts in the fixed network, (ii) errors can be identified and corrected by the proxy, thus avoiding them to spread through the network, (iii) the usage of this new configuration only affects a reduced part of the network, and (iv) the proxy can introduce optimization for the communications network (e.g., optimized TCP, header compressions, change of protocol) [300]. As disadvantages for this solution we can list: (i) the loss of TCP end-to-end semantics, (ii) handoff overheads, and (iii) the need for the proxy to be trusted.

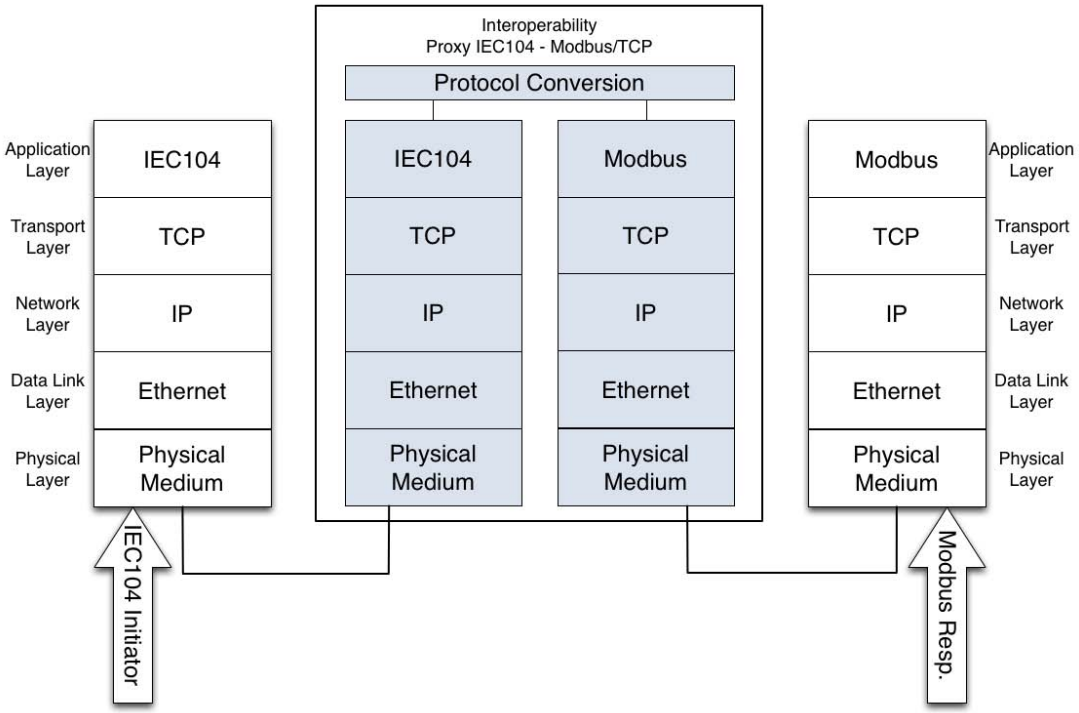


Figure 6.14: Proxy protocol conversion

In our scenario, we encounter a similar problem, there exist two networks which implement different communication protocols, and it is necessary to establish a successful connection between the nodes of these networks in order to exchange control information. Since direct translation is impossible due to the differences between the two industrial protocols (Modbus/TCP and IEC104) and the aforementioned restrictions of the TCP protocol. Thus we need to deploy a protocol conversion proxy capable of performing the mentioned segmentation of the communications in the network in order to allow the communication within our architecture, preferably in a transparent way (i.e., the nodes of the network are unaware of the proxy converter and believe they establish a direct connection). Figure 6.14 provides an overview of the protocol conversion through the transparent interoperability proxy, representing the protocol stack for both the IEC104 and Modbus/TCP protocols. It is possible to observe that each protocol layer provides an encapsulation for the network packets, and it is necessary to filter them and to extract the command codes in the application layer protocol in order to parse the queries and responses for the communications between nodes of different protocols.

Interoperability Proxy Implementation

In our testbed scenario (see the first validation case 6.2.1), the IEC104 client periodically opens a connection with the Modbus/TCP server in order to verify the correctness of the measurements taken by the IEC104 network sensors. In order to implement that, we have deployed a transparent interoperability proxy as part of the network, that would reside in the servers of the cloudlet. This proxy is in charge of translating the communications between the nodes of one network and the other in a transparent way, i.e., an IEC104 node establishes a connection to the Modbus/TCP server, without noticing that the communication is passing through a proxy.

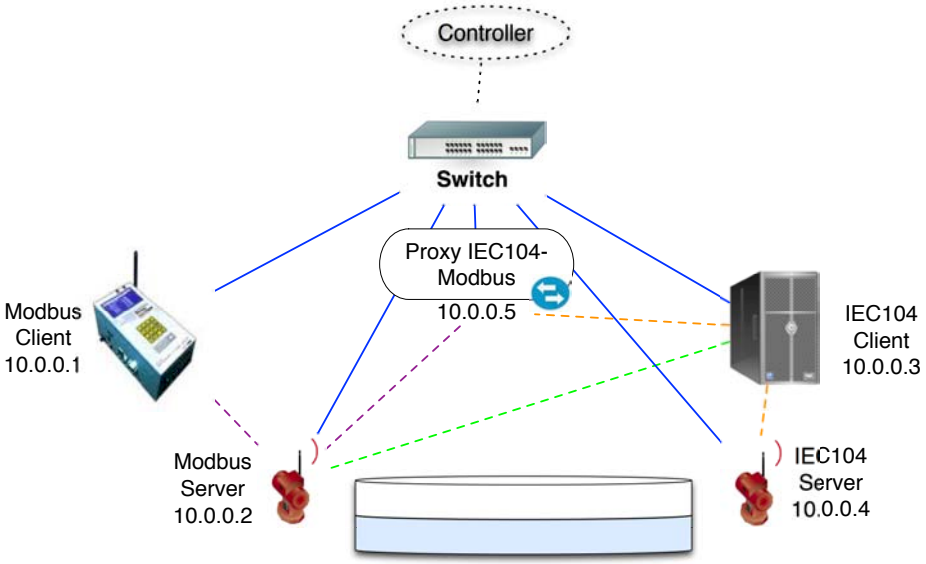


Figure 6.15: Proxy utilization in the testbed

Figure 6.15 illustrates the scenario, where the IEC104 communications are represented in orange, and the IEC104 client connects to the proxy in order to request information from the Modbus server (the Modbus/TCP communications are represented in purple). In this scenario, since the proxy is transparent, the nodes believe they are establishing a point-to-point communication, however all the communications go through the proxy (represented in green). The subsequent operations performed by the controller's proxy in order to retrieve the requested information from the Modbus server are completely transparent to the IEC104 client.

Figure 6.11 represents the communications between the nodes in Mininet, where each of the nodes in the testbed network corresponds to a Mininet host. Due to implementations requirements, in Mininet, the proxy node is deployed in a mininet network host, however, in our design, it could be implemented as a virtualized network function within the cloudlet servers together with the SDN controller. The transparency functionality of the proxy is implemented at the SDN controller's level, since it can modify the flow tables of the switch, it is possible to install redirection rules depending on the traffic that arrives from one port of the switch to another.

Listing 6.1: Log for the interoperability proxy activation

```
Client connected using TCP/IP. Will listen for a StartDT request. Connection ID: 2
Started data transfer on connection (2) Will listen for incoming commands.
Got interrogation command. Will send scaled measured values.
```

We test our design of the protocol converter through the proposed implementation of the interoperability proxy and perform some experiments to verify the correct operation of the system. Listing 6.1 shows the system's log containing the activation of the interoperability proxy when a request for communication is sent from the IEC104 client to the Modbus server (see Figures 6.15). Following to that request, it is possible to see in Listing 6.2 the logs corresponding to the behavior of the protocol converter module: (i) firstly, a connection with the Modbus Server is established and a request for information is sent to it, (ii) secondly, the response of the Modbus server (in the experiment, the water level is raising to the 250 mark level) is translated into the IEC104 protocol and sent to the IEC104 client requesting it, and (iii) finally, the data values are verified, allowing the system to continue its operation without indicating the presence of any incident.

Listing 6.2: Log for the protocol converter operation

```
#The IEC104 client needs to establish a connection with the Modbus server and sends a
    request for the information
Verifying information with Modbus server ...
successfully connected

Received ASDU:
Type ID: 100, C_IC_NA_1, Interrogation command
Cause of transmission: ACTIVATION_CON, test: false, negative con: false
Originator address: 0, Common address: 1
IOA: 0
Qualifier of interrogation: 20

#The proxy translates the request to the Modbus network and returns the information
    required
Received ASDU:
Type ID: 11, M_ME_NB_1, Measured value, scaled value
Cause of transmission: SPONTANEOUS, test: false, negative con: false
Originator address: 0, Common address: 1
IOA: 1
Information Element Set 1:
Scaled value: 250
Quality, overflow: true, blocked: true, substituted: true, not topical: true, invalid:
    true
Information Element Set 2:
Scaled value: 250
Quality, overflow: true, blocked: true, substituted: true, not topical: true, invalid:
    true
Information Element Set 3:
Scaled value: 250
Quality, overflow: true, blocked: true, substituted: true, not topical: true, invalid:
    true

Modbus server value verified, continuing ...
successfully connected
```

6.4.2 Monitoring and Accountability Managers

The second validation case contemplates the analysis of the monitoring manager and accountability and backup modules of the SDN controller (see Section 6.2.2 for extended information). As we mentioned, this validation case covers the experimentation on the logger module, together with the local monitor and the alarm manager modules, as depicted in Figure 6.1.

To test the local monitor capabilities, we have implemented a reduced IDPRS prototype capable of performing detection and response actions in our scenario. This system contains an IDS using a signature-based and specifications-based detection engine to detect threatening and anomalous dynamics happening in the testbed. The signature-based module is composed of detection rules that can be matched to known threats. The specifications-based detection module contains a partial implementation of the IEC104 protocol, where the correctness of the protocol packets (APDUs) can be verified.

We also provide a simplified IPRS module capable of launching varied countermeasure actions in response to the threats posed to the testbed in each experiment. This IPRS has a static configuration, i.e., it has a static cost calculation, a static mapping design, it provides burst response execution in a reactive manner (see Section 4.3.3 for detailed information on IPRS configurations). This selected configuration illustrates the operation of the response module in the local monitor of the controller, however, in a fully-fledged environment, dynamic adaptive solutions would be more interesting to provide more sophisticated and measured responses.

Accountability and Backup Module

Listing 6.3: Logger output for the utilization of the transparent interoperability proxy

```
2016-07-03 23:42:26,468 INFO IEC104 M_ME_NB_1
2016-07-03 23:42:26,610 INFO Modbus/TCP packet
2016-07-03 23:42:26,618 INFO Proxy not used
2016-07-03 23:42:26,662 INFO IEC104 packet
2016-07-03 23:42:26,664 WARNING Transparent proxy:  Redirecting Host1 to Host5
2016-07-03 23:42:26,670 INFO IEC104 packet
2016-07-03 23:42:26,670 WARNING Transparent proxy:  Reply
2016-07-03 23:42:26,673 INFO IEC104 packet
2016-07-03 23:42:26,674 WARNING Transparent proxy:  Redirecting Host1 to Host5
2016-07-03 23:42:26,677 INFO IEC104 packet
2016-07-03 23:42:26,678 WARNING Transparent proxy:  Redirecting Host1 to Host5
2016-07-03 23:42:26,723 INFO IEC104 packet
2016-07-03 23:42:26,724 WARNING Transparent proxy:  Reply
2016-07-03 23:42:26,731 INFO IEC104 packet
2016-07-03 23:42:26,735 WARNING Transparent proxy:  Reply
2016-07-03 23:42:26,739 INFO IEC104 packet
2016-07-03 23:42:26,739 WARNING Transparent proxy:  Redirecting Host1 to Host5
2016-07-03 23:42:27,162 INFO Proxy not used
2016-07-03 23:42:27,174 INFO Modbus/TCP packet
```

Firstly, we address the setting of the accountability and backup module for our testbed scenario. We have implemented this module using a *Logger* created using the *Logging facility for Python*

[302]. This facility helps implement a logging system using a standard library module, allowing the system to log the custom messages integrated with the messages of third-party modules. It is possible to customize the level or severity of the events in the system using the classification: `DEBUG`, `INFO`, `WARNING`, `ERROR`, and `CRITICAL`.

We introduce this logging module within our POX controller, allowing it to log any events in the system. Listing 6.3 shows the output produced by the logger module, when recording the events of the transparent interoperability proxy (see Section 6.4.1) utilization in the testbed scenario. Other validation tests for this module are dependent on the critical events occurring within the testbed, therefore we decide to test it together with the monitoring manager module, where we launch different attacks to the system and they must be properly registered for later forensics and accountability purposes.

Disconnection Attack

As discussed in Section 6.2.2, the first attack launched against the testbed has a low degree of stealthiness, where the attacker causes a sensor to stop transmitting. This action is detected by the IDS, and the *alarm manager* sends an alert to the HMI indicating a critical occurrence in the control network (see Figure6.5). This alert service is in charge of notifying the operator of any important events in the system. In the presence of an event (alert of failure, misuse of the equipment, important actions in the system), the alarm managers should provide the human operator with useful information about the occurrence. In the literature, some solutions are proposed in order to filter the amount of information arriving to the operation according to the importance and criticality of the alerts [151].

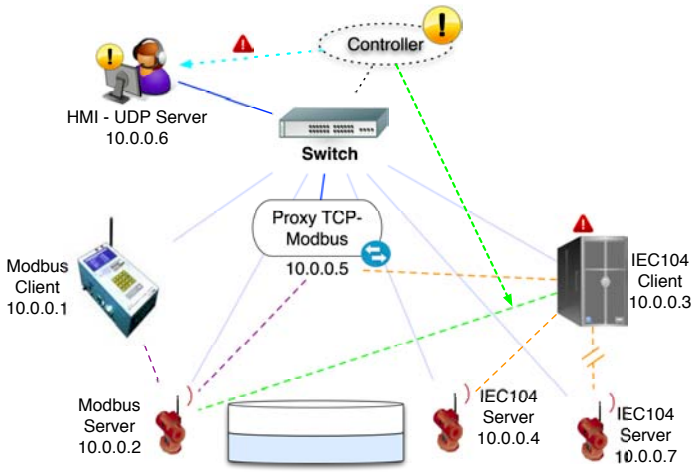


Figure 6.16: Notification of alarms to the HMI in the disconnection attack to the testbed

In this scenario, the operator is alerted of events labeled as `CRITICAL` in the controller. In order to implement this, we deploy a UDP Server in a host in our testbed, which will represent the HMI terminal for the monitoring panel in the control center. As previously described, the IEC104 client stops transmitting, this is detected by the controller's IDS and a log is created. Additionally, a notification is sent to the HMI to inform the operator of the presence of a critical

alert in the network. Listing 6.4 shows an extract of the system logs, where the controller detects that a CPCS in the network has stopped transmitting for a certain period of time and alerts the operator to fix the situation. Figure 6.16 illustrates this scenario.

The implementation of this alarm manager system is based on the UDP server deployed in the HMI host, and the capability of the POX controller and the Scapy packet manipulation tool to create UDP packets from the controller itself to create the alert notifications. At the same time the logger module records a critical log, the alarm manager creates an alert for the human operator monitoring the HMI. This alarm manager module can be extended to allow the controller to notify the operators of multiple types of threatening events to the system. Listing 6.5 shows an implementation for this alert procedure for our testbed, where we can see the usage of the Scapy tool to create the alert packets for the HMI operators.

Listing 6.4: Logger output for the alert of sensor not responding

```
2016-07-03 23:49:39,043 INFO IEC104 M_ME_NB_1
2016-07-03 23:49:39,044 INFO Proxy not used
2016-07-03 23:49:39,050 INFO IEC104 packet
2016-07-03 23:49:39,051 INFO Monitored Node H7: connection OK
2016-07-03 23:49:39,051 CRITICAL ALERT: Sensor in 10.0.0.7 not responding
2016-07-03 23:49:39,055 INFO Proxy not used
2016-07-03 23:49:39,056 INFO Transmitting alert to HMI ...
2016-07-03 23:49:39,530 INFO Proxy not used
2016-07-03 23:49:39,542 INFO Modbus/TCP packet
2016-07-03 23:49:39,544 INFO Proxy not used
2016-07-03 23:49:39,553 INFO Proxy not used
2016-07-03 23:49:40,077 INFO Proxy not used
2016-07-03 23:49:40,087 INFO IEC104 packet
2016-07-03 23:49:40,090 INFO IEC104 C_IC_NA_1
```

Once the IDS has detected the attack to the system, and a log is created, the IPRS of the local monitor is activated in order to palliate the situation. In our scenario, the IPRS countermeasures the attack by launching the interoperability proxy. This measure uses provider's B control network as redundancy and backup mechanism to reinforce the supervision operations for control purposes.

Listing 6.5: Using Scapy to create the UDP packet for the alert notification

```
#Scapy creates an UDP packet for the HMI with destination IP=10.0.0.6, port=10000, mac
    =00:00:00:00:00:06
#message contains the customized alert message sent by the controller to the HMI
a=Ether(dst="00:00:00:00:00:06")/IP(dst="10.0.0.6")/UDP(dport=10000)/message

#The controller sends the packet out of the specified switch port
#In this case the output port where the HMI is, is the port number 6 of the switch
self.send_packet(None, str(a), out_port=6, in_port=0)
```

Active Eavesdropping Attack

As described in Section 6.2.2, the adversary uses spoofing techniques to create a MITM attack against a CPCS of provider's A control network. Using this node, the attacker tries to send

control commands to the IEC104 network. The MITM attack is implemented using the the *Address Resolution Protocol* (ARP) spoofing technique as indicated in [303]. This attack has a higher level of stealthiness than the previous one, thus, the IDS has to implement mechanisms to detect this kind of situation.

In our scenario, we provide the controller's IDS the ability to detect invalid control requests through a whitelisting procedure. This, together with the knowledge of the control network (whitelist of authorized MAC addresses) enables the IDS to identify malicious requests in the system. Therefore, when the IEC104 client sends its server a forbidden request, it is detected by the controller's IDS and a critical log is created. Additionally, a notification is sent to the HMI to inform the operator of the presence of a critical alert concerning an unauthorized operation in the network. This is illustrated in Figure 6.6. Thus, when these events are detected and registered, the IPRS in the controller acts by filtering the traffic coming from this node while the system operators act in order to fix the situation and eliminate the attacker's presence in the network.

We implement this whitelisting function at the controller, where we perform *Deep-Packet Inspection* (DPI) [291] in order to extract the IEC104 function code and verify if the requested operation is allowed. We use the IEC104 dissector together with Scapy (see Section 6.3.6) in order to manipulate the packet and extract the adequate fields of the IEC104 protocol for verification. In our controller, we allow the "Interrogation Command", ASDU Type ID 100 `C_IC_NA_1` and the response command ASDU Type ID 11: `M_ME_NB_1`. However, we filter out the "Clock Synchronization" command ASDU Type ID 103 `C_CS_NA_1`. All these events are recorded in the logs of the system including its timestamp and the code indicating their criticality, for later analysis. Listing 6.6 illustrates the logger output for this experiment.

Listing 6.6: Logger output for the alert of non-authorized operation

```
2016-07-03 23:49:30,119 INFO IEC104 C_CS_NA_1
2016-07-03 23:49:30,120 CRITICAL IEC104: Operation not permitted
2016-07-03 23:49:30,140 INFO Proxy not used
2016-07-03 23:49:30,142 INFO Transmitting alert to HMI ...
```

Covert channel attack

This validation case focuses on the establishment of a covert channel between different nodes of the IEC104 (provier's A) network in the testbed (see Section 6.2.3). As shown in Figure 6.7, the malware infection compromises several IEC104 sensors, as well as an IEC104 gateway. The covert channel is established between the gateway and each of the infected sensors, in order to retrieve uptime information from the CPCSs.

There are many ways of implementing covert channels, and many variations depending on the protocol used (e.g., IP, TCP, Modbus). In our scenario, as proof of concept for this stealth attack, we have implemented a covert channel propagated by malware, taking advantage of a field in the IEC104 packet, in order to transmit information to the adversary (see Section 2.3.4 for extended information). For our example, we use the *Information Object Address* (IOA) fields in order to transmit the information for the covert channel.

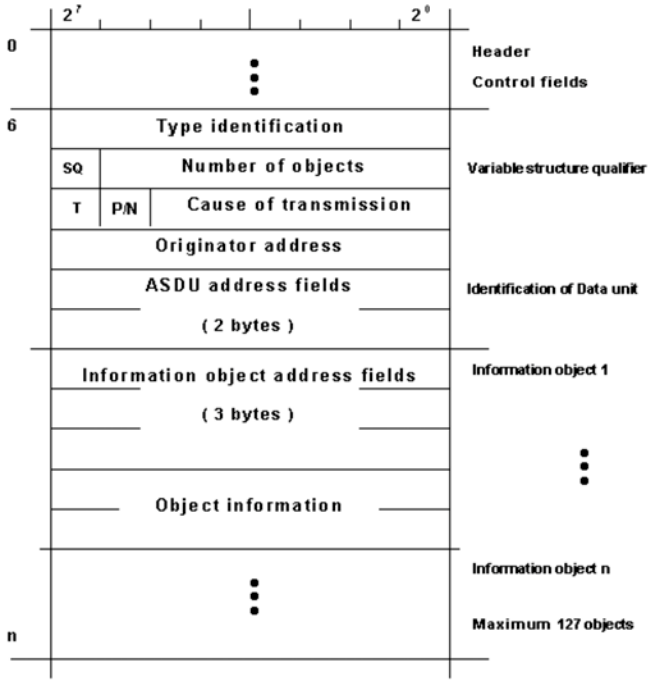


Figure 6.17: IEC104 ASDU [7]

Listing 6.7: Sample code for the covert channel implementation

```
// Covert channel implemented using the third byte of the IOA of the first information
    object in the ASDU
byte[] IOA = 0x01, 0x00, 0xa3; // third byte is a covert message
int covertMessage = new java.math.BigInteger(IOA).intValue();
// Decimal value of covert message = 65699
boolean SQ = false;
connection.send(new ASdu(TypeId.M_ME_NB_1, SQ, CauseOfTransmission.SPONTANEOUS, false,
    false, 0, aSdu.getCommonAddress(),
    new InformationObject[] {
            new InformationObject( covertMessage, // First IOA (COVERT CHANNEL)
                    new InformationElement[][] {
                            { new IeScaledValue(val), new IeQuality(true, true, true,
                                true, true) } }),
             new InformationObject(1192960,     // IOA
                     new InformationElement[][] {
                             { new IeScaledValue(val), new IeQuality(true, true, true,
                                 true, true) } })
             }
      ));
```

The IOA indicates the object addresses for each information object transmitted within the IEC104 ASDU packets, as depicted in Figure 6.17. Each IEC104 ASDU can contain from 1 to 127 information objects, where each object has an IOA field. The IOA has a length of 3 octets, however the third byte is only applied for disambiguation purposes in the case of transmitting multiple information elements within an information object. According to the protocol specifications, "*In all cases the maximum number of different object addresses is limited*

to 65535 (as for two octets). If the information object address is not relevant (not used) in some ASDUs, it is set to zero" [7].

Therefore, in our testbed scenario, we set the SQ bit to zero, indicating that "*Each single element or a combination of elements is addressed by the information object address. The ASDU may consist of one or more than one equal information object. The number of objects is binary coded (number of objects) and defines the number of the information objects.*" This indicates that we implement the IEC104 communication in a way that each information object contains only an information element. Thus, the third byte of each IOA is never used, and a covert channel can be set making use of this field. Specifically, we use the third byte of the IOA contained on the first information object of each ASDU. Listing 6.7 shows an example implementation of this covert channel, implemented using the OpenMUC: j60870 library module [293] (see Section 6.3.4).

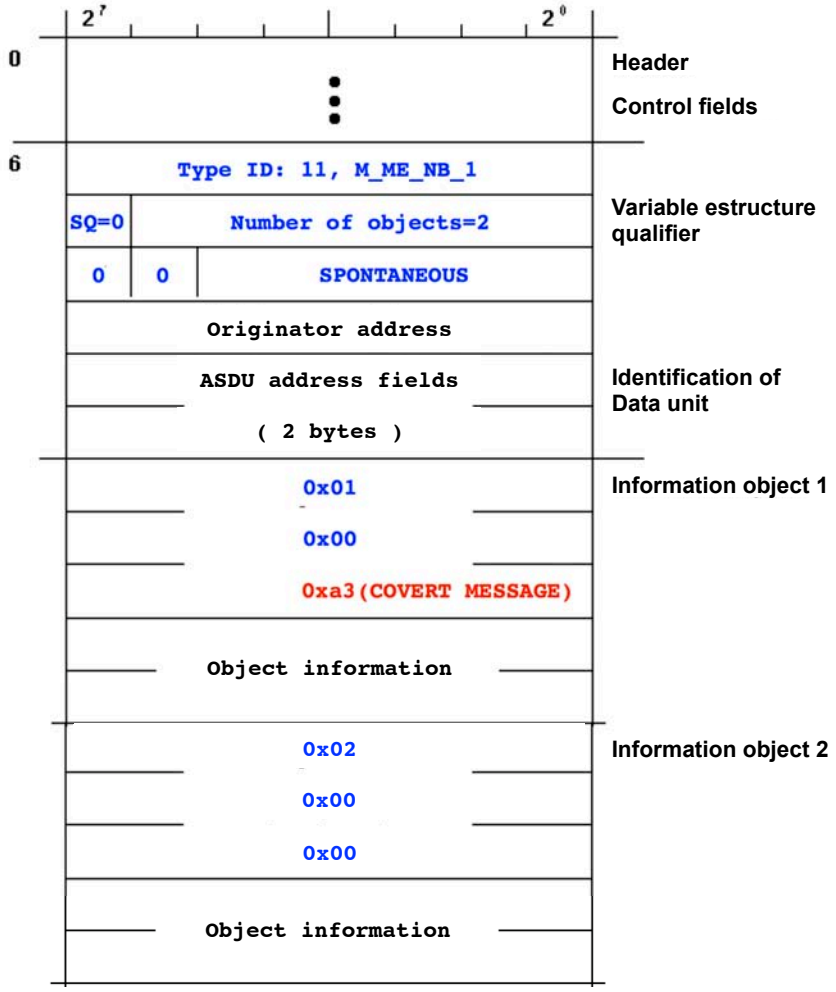


Figure 6.18: IEC104 ASDU containing covert channel

In this example, there are two information objects, each of them containing an information element with an IOA. The first IOA value would be $010000_{hex} = 65536_{dec}$, and the second

IOA value is $020000_{hex} = 131072_{dec}$. However, in the first IOA, the malware in our system introduces the covert channel. In this case, the malware indicates that this specific IEC104 CPCS has been working 163 hours without interruption. Since $163_{dec} = a3_{hex}$, the IOA value for the first information object is transformed to $0100a3_{hex} = 65699_{dec}$. This is illustrated in Figure 6.18, where we can see the ASDU packet for this example, including the covert channel information.

Listing 6.8: Log for the covert channel communication

```
Received ASDU:
Type ID: 100, C_IC_NA_1, Interrogation command
Cause of transmission: ACTIVATION_CON, test: false, negative con: false
Originator address: 0, Common address: 1
IOA: 0
Qualifier of interrogation: 20

Received ASDU:
Type ID: 11, M_ME_NB_1, Measured value, scaled value
Cause of transmission: SPONTANEOUS, test: false, negative con: false
Originator address: 0, Common address: 1
IOA: 65699
Scaled value: 34
Quality, overflow: true, blocked: true, substituted: true, not topical: true, invalid:
    true
IOA: 131072
Scaled value: 36
Quality, overflow: true, blocked: true, substituted: true, not topical: true, invalid:
    true
```

Listing 6.8 shows the system logs produced when the communication is established and the malware transmits the aforementioned covert messages. In the first place the IEC104 client interrogates the IEC104 server to obtain control information, sending an *interrogation command* `C_IC_NA_1` and the response to this query transmits the covert information together with the legitimate response ASDU which sends the requested measured value `M_ME_NB_1`.

In order to detect this type of stealthy attack, the IDS must be extremely tuned to the infrastructure's dynamics to detect those situations. However, this adjustment of the detection system to especially stealthy situations may cause the IDS to produce a high number of false positives, and hence a high level of alarms to the operator. This situation makes it necessary to configure a mixed IDS which uses signtature-based detection together with specifications-based insight of the system communication protocols (see Section 4.2.1 for further information on detection). In a real environment, this double detection engine can be improved by adding a third component, the anomaly-based detection system in order to better detect previously unknown events.

Therefore, the inclusion of knowledge about the systems and the protocol behavior within the detection engine permits the IDS to detect stealthy attacks such as the covert channel implemented in our testbed with high precision and a low rate of false positives. In our scenario, we program the IEC104 protocol in a way that we are certain that the third byte of each IOA field is never used by legitimate communications (see Figure 6.17). Thus, our IDS specification includes the indication that this byte in the ASDU must be always set to zero in the normal traffic. The information on this byte of the packet's ASDU is obtained by performing deep-packet inspection of the network's traffic. When a packet containing a value different from zero in this field is found, the IDS's engine consider it suspicious and an alert is sent to the operator in the HMI in

order to resolve the situation.

While these actions are performed by the IDS and the alarm manager, in case of determining that the existence of this illegitimate packet format is a threat to the system, the IPRS in the local monitor takes corrective measures. There are very different solutions to address this situation. In our example, since a malicious communication channel has been established via this field in the IEC104 packet, we configure our IPRS to delete the information in this field by setting it to zero, while the system operators intervene.

6.4.3 Authorization and Access Manager

In Section 6.2.3 we describe the last scenario of validation, this corresponds to the *authorization manager* present in the *policies manager*, together with the *authentication and access manager* module of our simplified controller (see Figure 6.1). As aforementioned, the authorization manager completes the validation of the policies manager module in combination with the tests performed for the first validation scenario (see Section 6.4.1) as described in Section 6.2.1.

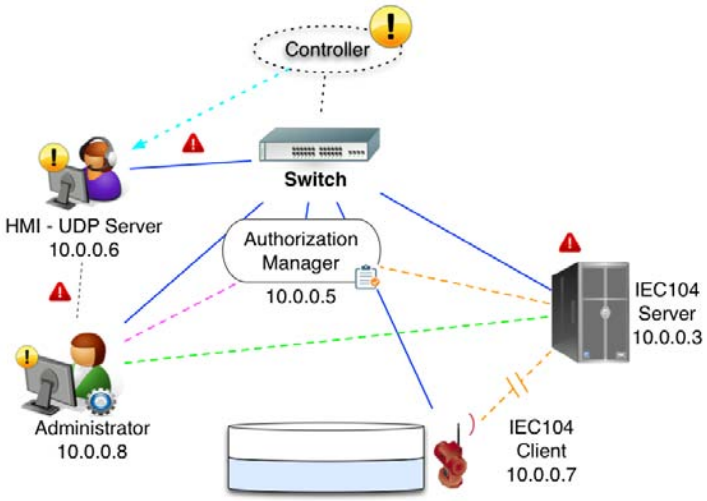


Figure 6.19: Authorization Manager utilization in the testbed

The authentication and access manager module is described in detail in Section 5.4.1, however, the validation tests will only take into account the *roles manager* and *access control manager* modules, which compose what we refer to as the *authorization and access manager* module. We have made the decision to omit the *Identity Manager* and the *Authentication Manager* modules because these solutions are widely used systems present in most IT scenarios, including CIP scenarios. Thus we consider these modules are already validated in a critical scenario and will add difficulty to the implementation or hide interesting details of our testbed validation.

To set this validation scenario, we introduce several actors in our testbed: (i) an IEC104 client that suddenly stops transmitting due to a malfunction (hence producing a system anomaly) and (ii) a remote administrator that tries to fix the situation (see Figure 6.19). The network is configured in Mininet as in the previous scenarios, with the addition of the new actors. To

implement the access control mechanisms, we make use of the IEC 62351-8 standard [28], as described in Section 6.3.5.

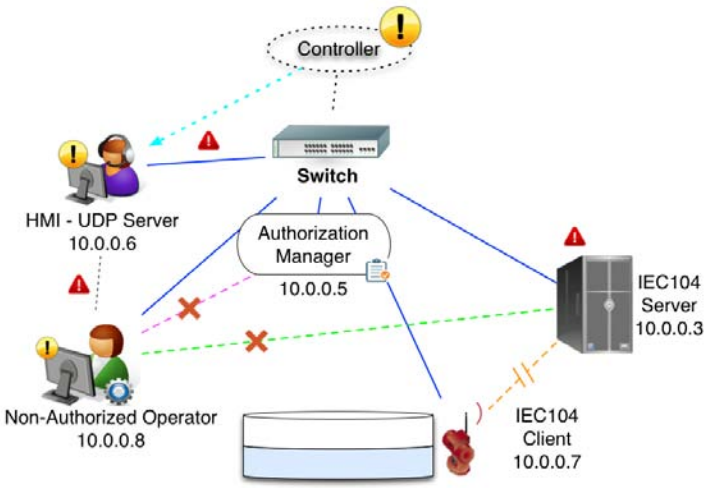


Figure 6.20: Role-based connection rejection in the testbed

In this scenario, a failure in one of the IEC104 sensors makes it to stop transmitting the measurements to the IEC104 server. This malfunction is logged to the system and an alert is sent to the operator monitoring the HMI, as in the previous test case (see Section 6.4.2). After the alert, a system administrator is notified of the failure and attempts to remotely connect to the IEC104 server in order to retrieve the logging information for analysis. In order to do so, it has to provide adequate credentials that allow the access and operation requested to the target resource.

Two different situations can emerge from this interaction:

- The administrator is granted the access, given the AuthAM receives adequate credentials that allow the administrator to retrieve the information, as depicted in Figure 6.19.

- The administrator is unable to provide adequate credentials that allow the retrieval of information, and hence is denied the access. This is depicted in Figure 6.20, where the non-authorized operator tries to establish the connection with the server and the access is denied.

In Listing 6.9 we can observe the logs produced by these two situations in the system. In the first place, an operator tries to access the IEC104 server providing the credentials: `engineer, view, IECServer`, which do not allow access to this resource, thus making the AuthAM to drop the connection. In a second place, the operator accesses the IEC104 server providing the credentials: `operator, view, IECServer`, where this role gives her access to the required resource and requested information.

Listing 6.9: Logger output for the authorization and access manager interaction

```
2016-07-05 01:34:38,564 DEBUG Socket created
2016-07-05 01:34:38,564 DEBUG Server running in: ('10.0.0.5', 10000)
2016-07-05 01:34:43,209 INFO Connection from: ('10.0.0.8', 56972)
2016-07-05 01:34:43,246 INFO Credentials provided:  engineer view IECServer
2016-07-05 01:34:43,248 ERROR Connection NOT allowed, dropping connexion...
2016-07-05 03:12:28,054 DEBUG Socket created
2016-07-05 03:12:28,056 DEBUG Server running in: ('10.0.0.5', 10000)
2016-07-05 03:16:24,975 INFO Connection from: ('10.0.0.8', 57065)
2016-07-05 03:16:25,014 INFO Credentials provided:  operator view IECServer
2016-07-05 03:16:25,015 INFO Connection allowed, starting proxy...
2016-07-05 03:16:27,406 INFO Information received, value for the registry: -5
2016-07-05 03:16:27,407 INFO Information received, closing connection
```

Authorization and Access Manager Implementation

The implementation of this module is depicted in Figure 6.21, this service is deployed within a Mininet host, together with the transparent interoperability proxy, although both services are part of the intelligence of the SDN controllers and reside in the servers of each cloudlet of our architecture. The AuthAM is composed of two main modules, the *authorization module* that receives the queries to the system and the RBAC credentials, and a *client* that connects to the server targeting the requests for information. In our case, it implements an IEC104 client in charge of retrieving the requested data.

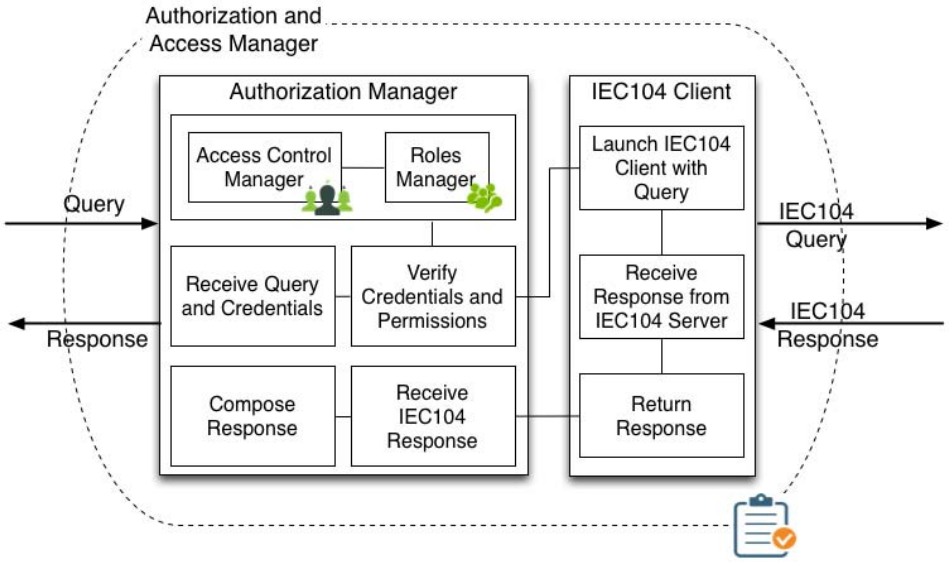


Figure 6.21: Authorization and access manager

When the human operator wants to connect to a node, it sends its credentials to the AuthAM module, it verifies the roles and permissions through the access control manager and the roles manager in the controller, and in case they are correct, it launches the IEC104 client to connect

to the IEC104 server to retrieve the system logs. When the response is returned, the AuthAM recomposes the response and sends it back to the operator. In case the credentials are not correct, the AuthAM denies the operator the connection to the server.

6.4.4 Performance Analysis

In this section, we aim to analyze the general performance and characteristics of the prototype system, as shown by the experimentations made on the testbed in previous sections. In order to do so, we gather information about the behavior of the system in each of the three main scenarios of validation (see Section 6.2). Once performed the experiments, we can retrieve the information and provide statistics and graphics on the behavior and performance of the system using tools such as the *Wireshark* network protocol analyzer. An example of such graphics is illustrated in Figure 6.22, which depicts the traffic of the network and the behavior of the testbed in the different validation scenarios described in Section 6.2. As we can see in this image, the Mininet testbed generates real network traffic in the different protocols used (illustrated by the different colors in the graph), and provides insight about the network utilization when each of the validation tests are launched in the testbed.

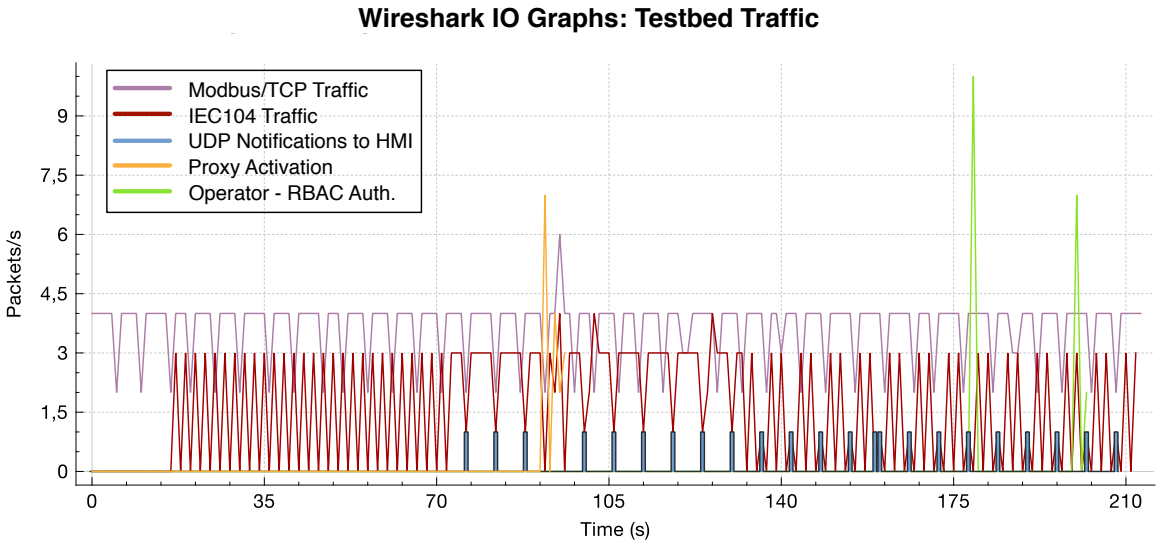


Figure 6.22: Testbed traffic graph

As depicted in Figure 6.22, we can observe in the beginning of the graph the regular traffic generated by the communications in the IEC104 network of provider B (in red), and the Modbus/TCP network of provider A (in purple). Around 70 seconds into the simulation, we begin introducing unauthorized operations petitions into the network, which are notified to the HMI as alerts (depicted in blue in the graph). It is also possible to see the activation of the interoperability proxy (in orange) and the changes in the Modbus/TCP and IEC104 networks to serve the interoperability petitions, which are reflected by a spike in the graphic. Later in the simulation it is possible to see (around 175 seconds into the simulation), the traffic generated in the network corresponding to the authorization procedures in the testbed (depicted in green).

To analyze the experiments in more detail, we break the analysis down into the three validation cases mentioned in Section 6.2. As we described, the first validation scenario is devoted to the experimentation with the protocol converter module of the simplified controller (see Figure 6.1), where we implement this conversion using a transparent interoperability proxy in order to provide communications capabilities between the IEC104 and the Modbus/TCP networks (see Section 6.4.1). Once performed the experiments described in Section 6.4.1, we retrieve data of the network traffic communications in order to analyze the behavior of our system in the testbed and analyze the specific traffic related to the transparent interoperability proxy activation. Figure 6.23 illustrates the behavior of the testbed in the moment of activation of the transparent interoperability proxy, where the functions of the protocol converter module are put to test. In the figure, the regular IEC104 is represented in red, the regular Modbus/TCP is represented in purple, and the traffic generated in the network by the transparent proxy is illustrated in orange.

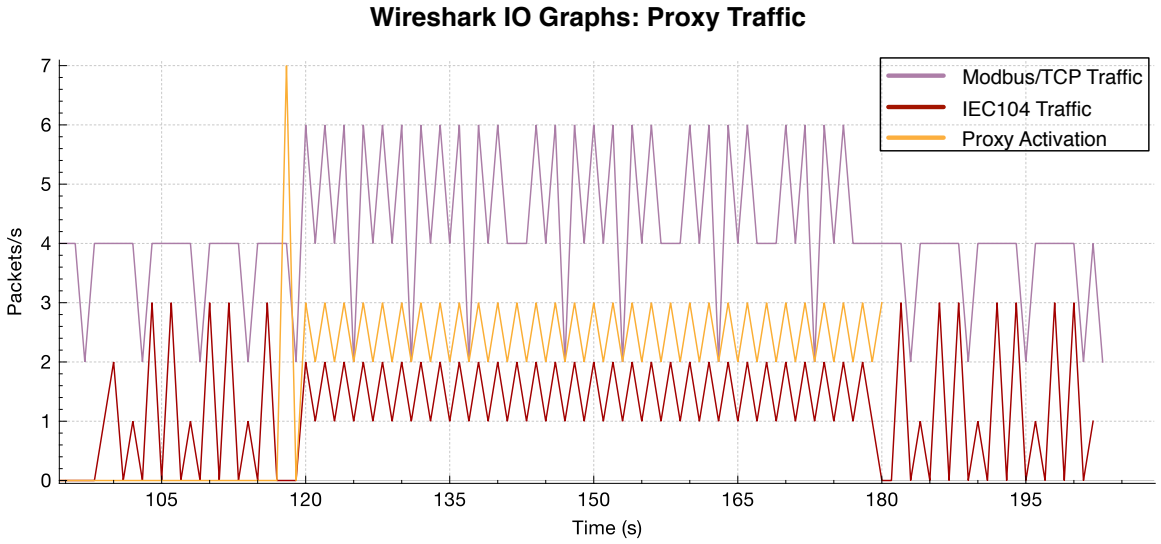


Figure 6.23: Proxy activation traffic graph

As we can see in the figure, there is a high number of injected packets in the network when the proxy is activated, returning to stable levels afterwards. We can see in the figure the increase of number of packets and the frequency of emissions in the Modbus/TCP and IEC104 protocols traffic when the communication of the proxy is established, since given the implementation of the transparent proxy, the network traffic between the devices and the proxy is augmented. The system returns to normal when the queries between the IEC104 client and the Modbus/TCP server have been responded, and the proxy ends the connection.

To better illustrate the operation of the transparent interoperability proxy, Figure 6.24 schematizes the communications flow of the transparent interoperability proxy in the testbed, i.e., the communications messages between the nodes `IP=10.0.0.4` corresponding to the IEC104 client, and the node `IP=10.0.0.1` which corresponds to the Modbus/TCP server (see Figure 6.15 in Section 6.4.1), passing through the interoperability proxy available in node `IP=10.0.0.5`. In the figure it is possible to observe the IEC104 petition to the Modbus server, which is "inter-

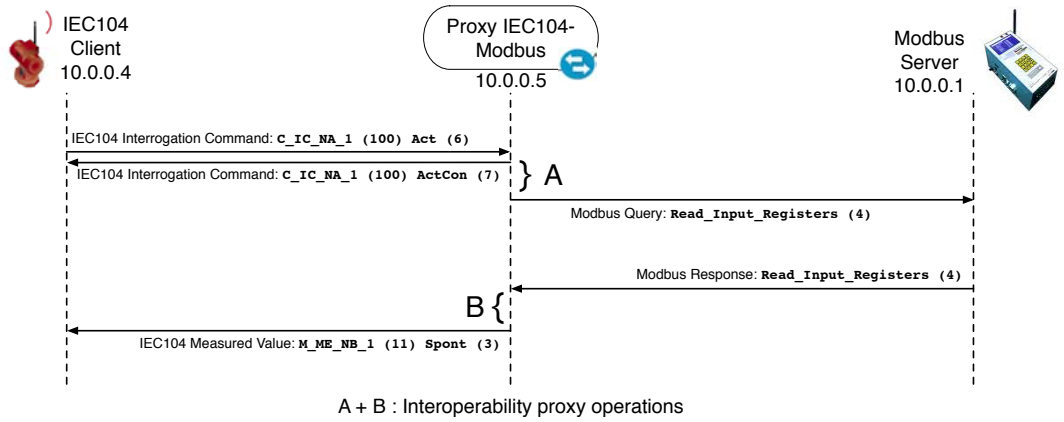


Figure 6.24: Flow diagram for the Interoperability proxy

cepted" by the transparent proxy, and converted into a Modbus query (conversion operation `A`). Similarly, when the Modbus response is sent to the IEC104 client, the proxy captures the packet and converts the response into an IEC104 response (conversion operation `B`). Operations `A` and `B` incur in a time cost, which correspond to the overhead time introduced by the interoperability proxy for the inter-network communications. In our experiments, given that the interoperability operations are not considered critical control commands (thus they are not optimized to achieve hard real-time requirements), the added overhead for the operations `A` and `B` (packet processing and normalization, deep packet inspection, protocol conversion, security policy conversion and packet reconstruction - see Figure 6.1) takes about 80 ms. These same operations can be highly optimized and prioritized in order to achieve real-time response times in case it is necessary to perform inter-network critical control tasks.

In order to further analyze the performance of the testbed network in general terms, we have generated throughput and *Roundtrip Time* (RTT) graphs using Wireshark. Figure 6.25 illustrates the throughput of the transparent interoperability proxy within the testbed, i.e., it indicates the individual performance of each of the communications packets between the IEC104 client and the Modbus/TCP server. As we can observe, in our experiments the average throughput for the proxy-enabled communications is of about 400 bits/s. Figure 6.26, on the other hand, shows the throughput of a regular connection between an IEC104 client and its server, without the need of proxy-enabled communications, where the average throughput for these communications is of about 1500 bits/s. As we mentioned before, given that the inter-network communications are marked as non-critical for our experimentations, the transparent interoperability proxy internal operations introduces delays on the network which entails a lower throughput for the inter-network connections than the throughput available for the nodes of the same network.

It is possible to also analyze and measure the performance of the testbed intra-network and inter-network communications in terms of the time it takes to send the request commands and receive the acknowledgements and responses in the network. We can observe this information in the RTT graphs generated by the Wireshark tool. Figure 6.27 provides an overview of the average length of time it takes for the request commands in the network to be sent and receive the corresponding acknowledges and responses. As we can see, in general terms the values in the graph are low, while there are certain operations that take around 80ms to be completed

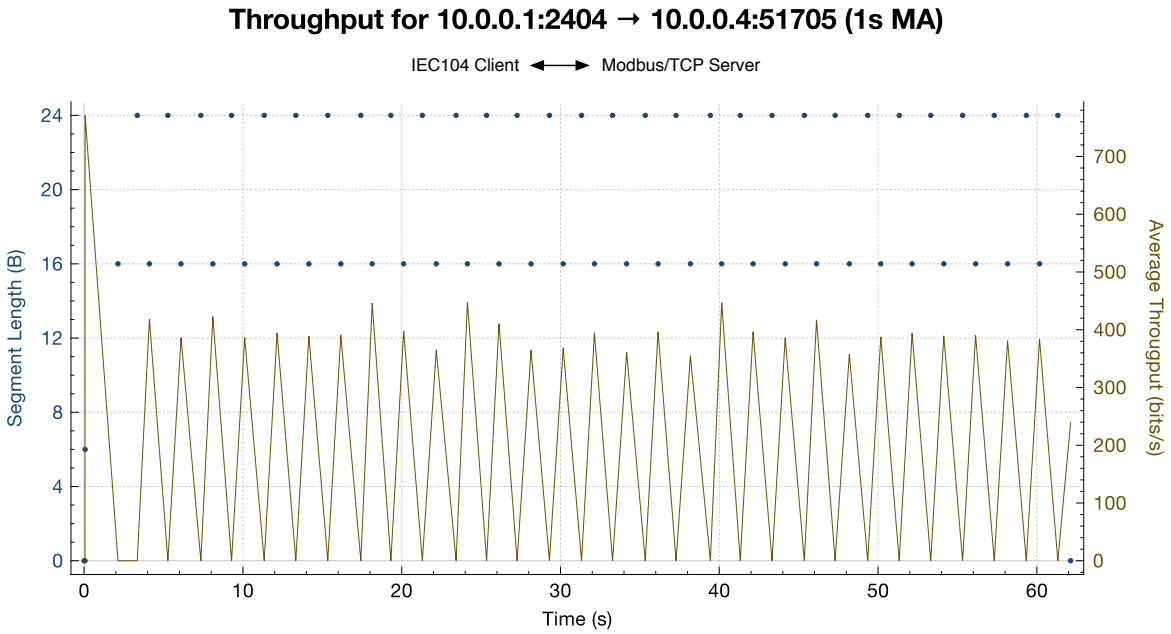


Figure 6.25: Throughput graph of the transparent proxy utilization between the IEC104 client and the Modbus/TCP server

(the higher spikes in the graph), these are the inter-network communications which precise of the proxy for interoperability between the different industrial protocols.

Following with our performance analysis, we can determine the efficiency of other services implemented by the controller module, for example, we can assess the performance of the AuthAM service, which manages the RBAC profiles of the system and controls the access to the resources of the system. Therefore, with the simulation data obtained from the third validation case (see Sections 6.2.3 and 6.4.3), we analyze the traffic in the network generated for the authorization process, together with the subsequent traffic generated when the access has been provided. Figure 6.28 illustrates these traffics, and we can observe in the graphic that the authorization process for the testbed does not incur in heavy overheads for the system since authorization is a procedure that takes a few milliseconds and introduces a low number of new packets in the network. With this information in mind, we believe this process is adequate to be implemented within a critical system, given its lightweightness.

The last module we would like to consider within this performance analysis is the behavior of the monitoring manager module (see Sections 6.2.2 and 6.4.2). As we described in Section 5.4.3, this module is in charge of scanning the systems and networks in the secure interoperability infrastructure, alerting of any threatening dynamics occurring within the system, and providing different countermeasure actions to palliate those threats. Since it is not the aim of this doctoral thesis to design a new IDPRS for the protection of CPCSs, we produce a local monitor service module operational for our validation scenario, which serves as a proof of concept.

We design this local monitor as a set of procedures installed directly on the POX controller (see Section 6.3.2). In our example we do not make use of the NSM objects technology o

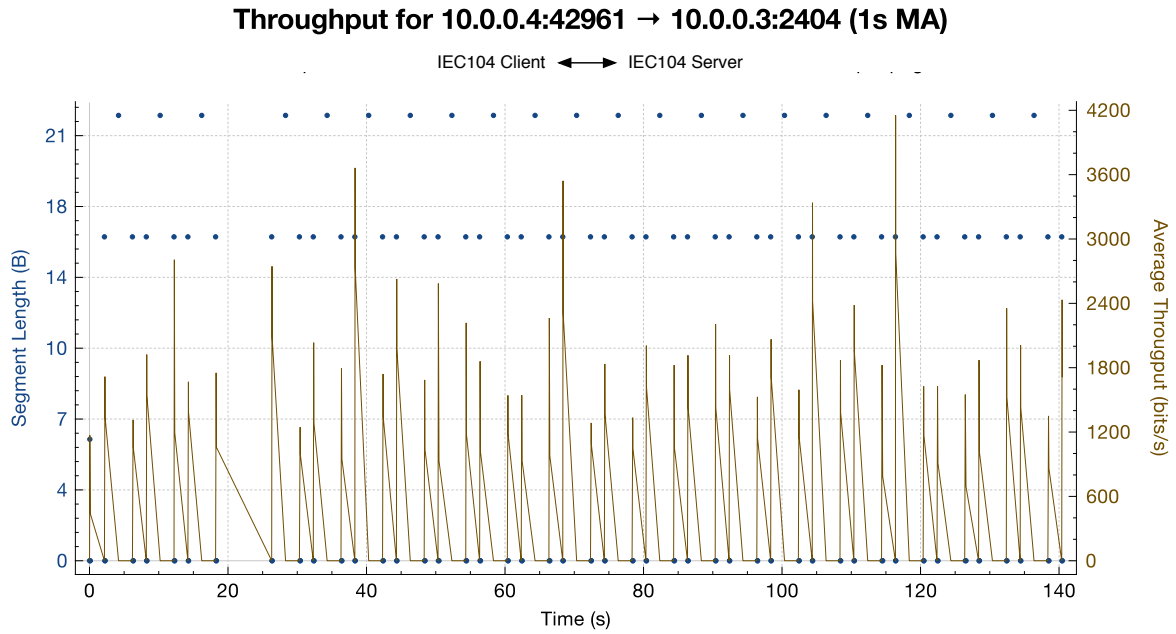


Figure 6.26: Throughput graph of the regular connection established between the IEC104 client and the IEC104 server

retrieve contextual information. Rather, we provide a simple service functionality integrated within the controller itself, in a way that introduces no overhead within our system. Listing 6.10 illustrates a simple process to control and discard forbidden operations taking using a whitelist of the IEC104 protocol operations, as we can observe, this is a process which makes use of the Scapy libraries (see Section 6.3.6) for deep packet inspection and packet manipulation and it is lightweight enough to introduce little to none overheads within the interoperability system. The alert manager service is also located at the POX controller, where we also make use of the Scapy libraries to create notification and alert packets for the operator supervising the security of the system on the HMI node, Listing 6.11 shows the simple code performing these operations.

With these considerations in mind, we can state that the performance of the monitoring manager module must be measured according to the IDPRS implemented within it. This process needs to be sufficiently powerful to contemplate all the dynamics of the network in the area of influence of the controller, but lightweight enough to avoid introducing any significative delays within the system and the network. Since this processes reside on the SDN controller, i.e., on the cloudlet servers, it is possible to use moderately sophisticated IDPRS solutions for detection, awareness and reaction, as described in Chapter 4.

Therefore, given the performance results obtained by the different experimentations and simulations carried out in order to validate our secure interoperability platform, we understand that the data obtained shows satisfactory outcomes for our simplified architectural model, and therefore promising results for the general architecture proposed in this thesis.

Listing 6.10: Monitoring manager in the POX controller, example of IEC104 functions whitelisting implementation

```
#Received packet
pkt = Ether(packet.pack())

#Identification of the IEC104 protocol packets
if pkt.haslayer(TCP) and (pkt.sport == 2404 or pkt.dport == 2404):

    logger.info('IEC104 packet')

    #Extraction of the TCP packet
    pktTCP = pkt.getlayer(TCP)

    #Discriminating the IEC104 packets
    if pkt.haslayer(Raw):

    #Extracting the payload from the TCP packet
    pktLoad = pkt.load

    #Extracting the second byte of the payload, itentifying the length of the APCI
        packet in the IEC104 protocol
    lenACPI = pktLoad[1]

    #Differentiating IEC104 START packets, which do not contain ASDU (length<=4)
    if ord(lenACPI) <= 4:
        print("lenACPI: ", ord(lenACPI))
        print "START"
    else:
        #Loading IEC104 Scapy-based packet dissector
        pack = iec104prot.APDU(pktLoad)

        #Call to Whitelisting functions - IDS functions for Deep-packet-inspection

        #[...]

        #Example of packet whitelisting by IEC104 'type identification' code:
        type_id = pack.apci.information.seq_received

        if type_id == 103:
        print "type_id: C_CS_NA_1 - Clock synchronization"
        logger.info('IEC104 C_CS_NA_1')

        elif type_id == 100:
        print "type_id: C_IC_NA_1"
        logger.info('IEC104 C_IC_NA_1')

        elif type_id == 11:
        print "type_id: M_ME_NB_1"
        logger.info('IEC104 M_ME_NB_1')

        else:
        print "Other codes"
        logger.critical('IEC104: Operation not permitted')
        self.to_HMI("ALERT on Operation: " + str(type_id))
```

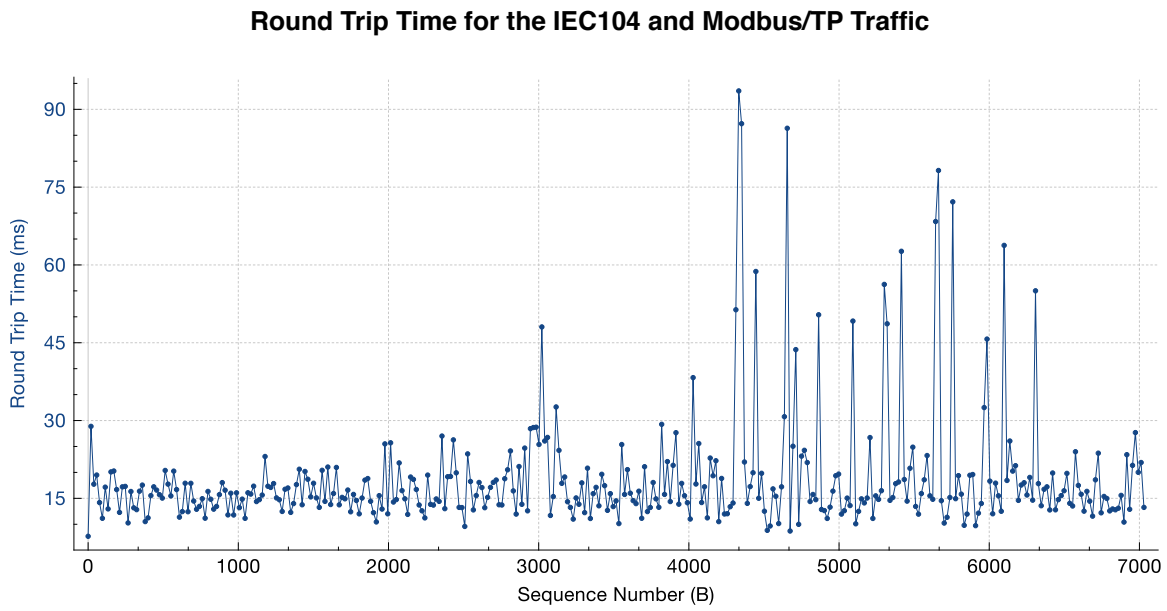


Figure 6.27: Roundtrip time graph of the traffic in the testbed networks

6.5 Discussion

It is the aim of this chapter to analyze in depth the main characteristics and behaviors of the secure interoperability architecture model designed in Chapter 5 through a series of experiments. These tests helps us understand the dynamics and problems that arise at an scenario where different CPCSs are interconnected in order to cooperate and communicate among them. To do so, we have implemented a testbed scenario where a simplified version of our design is put to test. With this testbed, we pursue the objective of validating through tests and experiments the functionalities and suitability of our architecture model for a scenario where the CSs of several critical infrastructures need to interact by means of a secure interoperability platform in order to perform cooperative tasks in regular and emergency situations.

As we have reviewed in the literature, the diversity and heterogeneity of (remote) networks present in current industrial systems, such as the energy industry (now turning towards the smart grid), makes these CIs complex to manage and maintain [1], especially when dealing with their remote field networks and substations. We, therefore, need to arrive at a solution which takes into account these considerations and characteristics and provides secure, reliable and efficient procedures for cooperation.

Our architecture model in Chapter 5 is designed taking into account an extensive capture and analysis of requirements for this type of interoperability scenario (see Chapter 3), and a detailed review of security services especially designed for CIP gathered in Chapter 4. Our architecture (see Figure 5.3) is the final product of this study. In the present chapter, we aim therefore, to review and determine if the characteristics described in the previous chapters are accomplished for our design through the implementation in the testbed.

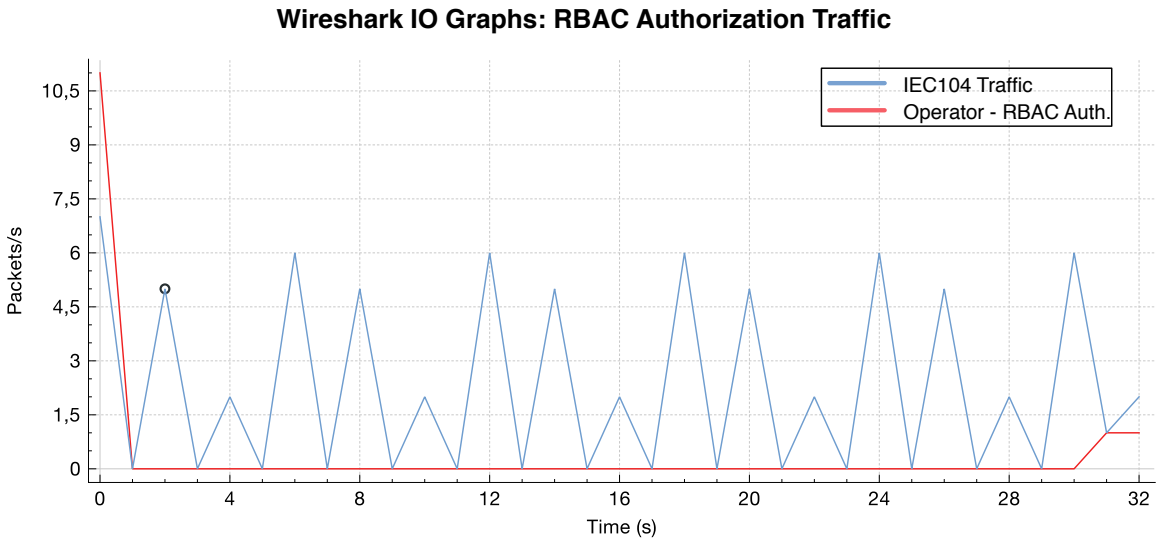


Figure 6.28: RBAC authorization traffic

Due to the complexity of the design at hand, we need to produce a reduced version of the interoperability architecture model where several of its main functionalities are tested, leaving the complete implementation of the prototype as future work of research. In this thesis, we decide to focus our validation on the review of the secure interoperability of the infrastructure at the level of cloudlet (see Section 5.3), an entity in which the different SG infrastructures cooperating through the interoperability platform can be divided on (i.e., each SG infrastructure can be subdivided into several cloudlets, which manage their different functions and networks).

The main interoperability control unit at the cloudlet's level is the controller, where they serve as access points (see Section 5.3) to the interoperability infrastructure to the different systems and devices participating in the federated infrastructure. For our experiments, we therefore focus on the functionality provided by the controllers, through the implementation of a simplified version

Listing 6.11: Monitoring manager in the POX controller, example of the alert notification procedure to the HMI

```
#This is the process for the controller that creates an UDP datagram to alert the HMI
    of the critical alerts
def to_HMI (self, mensaje):

#Using Scapy, the HMI resides in the host with IP:10.0.0.6, port=10000, mac
    =00:00:00:00:00:06
a=Ether(dst="00:00:00:00:00:06")/IP(dst="10.0.0.6")/UDP(dport=10000)/mensaje

#Sends a packet out of the specified switch port (in our testbed, port=6).
self.send_packet(None, str(a), out_port=6, in_port=0)

#Logging functions for accountability
logger.info('Transmitting alert to HMI ...')
```

of the controller as described in Section 6.1. In our validation cases, we aim to determine whether this simplified controller provides the desired interoperability characteristics described in the previous chapters.

Section 6.4 describes in detail the experiments performed to test each of the main secure interoperability functions available at the simplified controller and provides an overview of the performance of these services within the testbed. The implementation and development of the service modules for the simplified controller within the testbed is described in Sections 6.3 and 6.4. In the process of developing the services according to the designed architecture model and the requirements and recommendations guiding it, we have contemplated different particularities of the implementation of services in a real environment.

One of the important characteristics of the testbed we have developed is that it is based on the real network simulator and emulator Mininet (see Section 6.3.1), therefore it allows us to implement and run the complete industrial communications protocols IEC104 and Modbus/TCP. The use of the real protocols makes the implementation of the testbed face real-life problems such as the management of the protocols and services which we would have not been able to appreciate in the case of using a simple network simulator. What is more, given that Mininet is a network emulator, the code we have generated for our experimentations and validation cases, i.e., the different services and modules for the controller, could be deployed with little changes within a real production environment, and provide the same functionalities we have observed in our experimentation. This characteristic strengthens the validation of our architecture model, since we have been able to develop a simple version of a functional real-life industrial system. Additionally, through the experience in the process of development, we have found three main obstacles or considerations that shape our design:

- The implementation of interoperability services between real-life industrial protocols is constrained by the characteristics and capabilities of these protocols, rather than the services provided by the testbed. In architectural designs it is usual to provide solutions which take into account the protocols' application layer [304], however, the design concept can interfere with the internal operation of the communications protocols at other levels in the protocol stack, e.g., at the transport level. We encountered this kind of obstacle in our implementation of the testbed when tackling the interoperability procedures of protocol conversion, given that we are using the actual industrial protocols in our experimentations. As described in described in Section 6.1, we chose for our testbed the IEC104 and Modbus/TCP protocols, which are both based on TCP. These protocols are connection oriented, as described in Section 6.4.1, therefore, immediate protocol conversion can not be performed as conceptualized in our architecture model. We solve this problem implementing a proxy service for the interoperability between networks of the two different protocols, however, this characteristic shapes the implementation of interoperability for our validation case, and constrains the capabilities and efficiency of the service in this scenario. Other industrial protocols might bring other advantages or obstacles to light when implementing interoperability mechanisms between them. Therefore, it is necessary to take this characteristic into special consideration, since this type of perspective arises only at the moment of implementation of the conceptual model, and can alter or condition the actual characteristics of the secure interoperability architecture model.

- Automatic intrusion detection is a major pillar for the security in the Industry 4.0 scenario,

especially in an era in which cyber attacks against CIs are frequent, and they become more and more sophisticated and stealthy. Our design (see Chapter 5) includes detection procedures at a local and global level. We believe this double detection perspective will aid the protection of critical systems against the more complex attack dynamics. In our validation prototype however, we focus on the local monitor, since we devote our experiments to validate the controller. At the local level, the IDS module needs to be able to detect both system anomalies and attacks against the control network. The detection engine for this local monitor must implement a combination of detection strategies to allow it to detect stealthy attacks. Our prototype IDS provides insight about the detection capabilities of a signature-based detection which includes specification-based knowledge. This concept can be developed further in order to implement a fully-fledged detection system for the interoperability platform as indicated in its design (see Chapter 5).

- The monitoring manager module needs of a sophisticated IPRS solution capable of providing semi-to-automatic functionalities regarding reaction and mitigation characteristics, as described in Section 4.3. The main issue with this recommendation is the lack of current solutions that are demonstrated to be suitable for critical scenarios (see Section 4.2.4), providing detection, prevention and reaction mechanisms that take into account the critical scenario where they are deployed, where any action that is not correctly measured and perfectly tailored to the environment of application can unleash chains of unforeseen events that threaten the operation of the separate infrastructure and therefore the integrity of the whole interoperability platform. Given that this is currently an active open field of research, we have decided to restrict the experiments on this side of the testbed and provide only basic functionalities as a proof of concept for our experiments. However, this is an important module of the architecture model that needs further analysis.

- Both the monitoring manager and the policies manager modules in the controller of our original design need of the information gathered by the NSM objects deployed across the entire interoperability platform (see Section 5.3.6). As described in Section 6.1, we dismiss the implementation of a fully-fledged diagnosis system for our testbed experimentations at this stage, given the complexity of the target infrastructure. As detailed in Section 5.3.6, the diagnosis system retrieves data from all the main devices and networks present in the interoperability platform, and the implementation of these objects or agents is conditioned by the system where they are deployed. Intensive testing of this advanced diagnosis infrastructure must be performed at a large-scale network with real equipment, capable of simulating the characteristics and dynamics of the environment in the more realistic way possible. Due to the size and complexity of this test, we leave it as future work, however, we consider the diagnosis system to be of paramount importance in such an environment as the secure interoperability platform architecture model proposed in our design.

However, considering the requirements and recommendations for the secure interoperability platform, as compiled in Chapters 3 and 4, and the final design of our architectural model proposed in Chapter 5, we can observe the results of our experimentations as a whole. Throughout this chapter, we have focused our efforts on developing a functional testbed capable of determining the behavior and functionalities of our proposed design, and at the same time providing cases of experimentation for validating our architectural model. Our testbed, which has been developed with this objective in mind, is based on a network simulator and emulator, where we run the real industrial protocols creating realistic dynamics (and hence encountering real-life obstacles

and challenges) capable of accurately simulate the environment and dynamics of a real critical infrastructure. In our experiments, we have focused on a simplified version of the architecture, where we perform simulations to determine the behavior and efficiency of several services of the controller module (see Figure 6.1).

We test these services through three different validation cases to analyze the operation of the protocol converter, the monitoring manager and the authorization and access manager modules, obtaining satisfactory results. The protocol converter module is capable of providing internetwork transparent interoperability services for the communication among the different SG providers regardless of the industrial protocol used, and in an efficient way. The monitoring manager service is able to detect and palliate the attacks launched against the system, even in the case of stealthy attacks, and issues alerts to the system administrators when presented with anomalous situations in a lightweight and fast manner. And the authorization and access manager is capable of providing authorization services based on RBAC, without introducing significative overheads in the interoperability platform. Therefore, taking into account the features of our design and the experiments, we believe our architecture model for the secure interoperability of the CPCSs of the SG provides the interesting characteristics and functionalities needed to be deployed in a federated environment where diverse CIs interact, and we are confident that our design model is suitable for the proposed secure interoperability scenario.

Chapter 7

Conclusions

This final chapter constitutes a brief recapitulation of the work done on this thesis. We first summarize the motivation and contextual framework that incentive and shape the research performed throughout this doctoral thesis. We then describe the main contributions of our work, which come to fruition with the design of a new architectural model for the secure interconnection and interoperability of critical control systems. Finally we discuss open issues for research and future lines of work which have been opened with the development of this thesis.

7.1 Motivation and Contextual Framework

As discussed in the introduction of this thesis, in recent years a new paradigm for the industry has appeared, leading the way to the introduction of new concepts and technologies previously unseen in the world's infrastructures. This new industrial paradigm constitutes the conceptual framework defining our work. We devote this section to review the main characteristics that motivates the development of this thesis, providing the conceptual frame and starting point for the research issues addressed in this dissertation:

The arrival of the Industry 4.0 paradigm introduces new research challenges in order to accommodate the new technologies and conceptualizations within current industrial settings. Technologies such as the CPSs, the industrial IoT, the cloud, etc. bring new actors and new capabilities for the industrial systems into scene, which need to be addressed and carefully considered when developing new applications and systems in the frame of this paradigm.

The incorporation of the Industry 4.0 paradigm to current infrastructures generates a need for a **new generation of control systems** capable of taking into account the new technologies and conceptual needs of this new paradigm, and provides an adequate management and control of the new actors in the control infrastructures. We have identified for this fourth generation of control systems three key characteristics, i.e., *decentralization*, *security* and *interoperability*, which will shape the new models and architectures that arise as proposals for design of CSs in this new environment.

Therefore, this new architectural framework created based on the appearance of the new industrial paradigm and the need for CSs capable of addressing the new challenges brought by

the Industry 4.0, creates an interesting research gap that is the object of the investigation of our thesis. We aim to provide a new infrastructure model for the fourth generation of control systems which takes into account the new characteristics of the industrial paradigm, as well as the identified essential features of the new generation of CSs.

7.2 Contributions

Once established the motivation of this thesis, we review the main contributions provided in this thesis, according to the needs and challenges provided by this new scenario of research. These contributions and accomplishments are described in more detail in the following paragraphs.

Better understanding of advanced threats menacing the operation of critical infrastructures in the form of cyber stealth attacks against the control infrastructures (CCSs). Apart from well-known large-scale cyber attacks targeting CIs and their CSs, there is a second type of less-known and sometimes dismissed cyber attacks, based on common ICT characteristics, which are capable of causing great damage to critical systems. We provide a review of this type of advanced threat, analyzing the stages which these stealthy attacks can achieve, and categorize their impact and possible threatening effects according to the AICAn taxonomy. AICAn provides abstract indicators of impact and risks to critical systems, based on the security properties of the CIs. Through this taxonomy we can classify the different cyber stealth attacks targeting CIs and assess their level of stealthiness and penetration on the system, hence the dangers and risks they pose to critical systems. As part of our study we review countermeasures and prevention mechanisms capable of mitigating the risks and threats introduced by the studied cyber stealth attacks.

Identification of the requirements and characteristics of the architectural model from the point of view of SW engineering. In order to better understand the needs of the new architecture for the secure interoperability of the CPCSs of the SG we intend to build, we believe it is necessary to firstly provide a solid basis of knowledge which gathers the special constraints and requirements presented by the new scenario. Following the methodologies of the NFR Framework and the GQM approach, we create a model for our architecture which gathers its essential characteristics in the form of goals or non-functional requirements. The analysis of these softgoals opens the door to a better understanding of the needs of the target architectural model, and hence it provides insight about a possible set of satisficing techniques capable of addressing these needs and satisfying them. The last step of this analysis constitutes a concretization of the model via the review of metrics to assess the adequacy of the model to the context of application of the fourth generation of CSs, where the Industry 4.0 paradigm and the key components of the CSs have to be met and satisfied. The proposed set of metrics is based on varied reviewed standards and guidelines for the security and interoperability of critical systems.

Analysis of the characteristics and suitability of security services and solutions for the protection of control systems, particularly CPCSs. We particularize the application of the previously mentioned methodology to identify the characteristics and requirements of the security services needed for the protection of CCSs. This analysis of requirements, techniques

and metrics provides insight about the essential characteristics of interoperability and suitability that protection solutions for critical systems need to implement. The analysis is focused on the suitability of IDS and IDPRS solutions for a critical context.

Methodological framework and taxonomy for the IDPRS protection solutions for critical systems. Based on these previously identified characteristics, we review the different security services for CIs, from basic security services such as authentication, authorization or accountability; to advanced security measures, like prevention and detection, awareness and reaction or restoration mechanisms. To organize this analysis we provide a methodological framework and taxonomy for IDPRS solutions for critical systems to better understand the different components and processes they can implement, and their suitability for critical scenarios. The protection these IDPRSs can provide is very varied, and is determined by the modules they incorporate, and the mitigation and restoration processes they implement. This analysis of security services is put into context by studying their characteristics when applied to the CPCSs of the SG.

Design of a new architectural model for the secure interoperability of CPCSs for the smart grid. Following the key characteristics determined for the fourth generation of CSs, we propose a new architecture model solution which gathers the previously identified requirements and features for the secure interoperability of CPCSs in the context of SG. This architecture provides a decentralized infrastructure which implements secure interoperability at two different levels. Firstly the local (subnetwork) level, where the different CPCSs interact with the subjacent physical infrastructure and other network actors (e.g., IoT devices), implements its secure interoperability mechanisms making use of *controller modules* especially designed for this purpose. Each subnetwork contains one or many controllers capable of providing their services to the devices in their environment (area of influence). Interoperability and security services are provided at a global perspective by the designed *cloud module* which enables global connectivity and interaction among the different networks and subnetworks participating in the secure infrastructure platform. From a design perspective, the main characteristics devised for the new generation of CSs, i.e., decentralization, security and interoperability, are fulfilled with our design.

Development of a testbed and validation scenarios for the proposed secure and interoperable architecture for the CPCSs of the smart grid. To better understand the behavior and performance of our proposed design, we compose a testbed scenario for the validation through experimentation of the characteristics of our architecture. The main security and interoperability services of the controller module are implemented and tested, through three validation scenarios of the SG, which contrast the original design with the real-world implementation contemplating the real industrial protocols and technologies.

7.3 Open Research Issues and Future Work

Throughout the development of this thesis, we have identified some issues and areas in need of further research. Several of them have been discussed in their corresponding chapters, however we show them together in this chapter to have a comprehensive view of the future work and open research lines.

In the first place, we identify a need for providing advanced automatic and intelligent IDPRS solutions especially designed for their application in the context of CIP. In the literature it is possible to find numerous solutions for general-purpose networks and systems, however, their suitability for critical systems is not certain. Furthermore, there have been research efforts to provide IDS solutions especially conceived for CIP, nonetheless there are few IDPRS solutions equally conceived, and the characteristics they provide, i.e., their proactivity and response solutions, consist of a few notification features, and minimal response procedures which always need a human supervisor. Therefore, we believe further research efforts need to be placed in this area in order to create intelligent IDPRS solutions capable of establishing automatic awareness and reaction processes for CIP.

In the process of providing the analysis of requirements for the secure interoperability of CPCSs, we have uncovered two main issues which we do not address in our research, i.e., the validation of the NF requirements, and the validation of the identified set of metrics. As discussed in Chapter 3, we have performed an iterative process for the definition and refinement of the NFRs which stops at a still high level of abstraction, hence not providing a concrete definition and scenario of application for validation. This open issue can be addressed applying a further cycle of refinement of the requirements in a concrete scenario for validation, and following either a theoretical approach such as game theory, or through an expert group interview process (e.g., Delphi method) to determine the adequacy of the selected NFRs for the problem at hand, and to have access to feedback about the completeness of the requirement set.

Regarding the validation of the identified metrics, we find this is a similar scenario to the previous one. In our work, for each of the identified metrics, we provide a high-level definition, indicating their scope and the requirements they aim to assess, i.e., whether or not the matched requirements would have been fulfilled in the concrete secure interoperability scenario. As discussed, it is possible to tackle the validation of the metrics using two methods: through a formal process of validation, or through an expert group interview process, such as the Delphi method. This process remains an open issue for research given the complexity of the scenario, where there exist numerous actors and interdependencies, and where true validation procedures can only be performed in a real-life concrete critical infrastructure scenario. Therefore, we believe the validation of the proposed set of metrics can be the object of a research project that includes a partnership with the industry, where these tests and validations can be accomplished.

In order to better analyze the features and behaviors present in the architectural model for the secure interoperability of CPCSs provided in this thesis, it is possible to expand the validation and simulation cases present in Chapter 6. Due to the complexity of the architecture's design, the validation scenarios contemplate a reduced version of the model, which focus on the main services provided by the controller. It is possible however to implement the whole functionality of the proposed platform, including important components such as the fully-fledged diagnosis system or the services residing in the cloud, in order to better analyze the performance of the secure interoperability system.

Additionally, it is possible to include within our architectural model other technologies and protocols suitable for the Industry 4.0 scenario that are now in the process of research and development, such as the NFV, which can introduce new redundancy and robustness mechanisms within the CIs, and might redefine the industrial control systems of the future by allowing the infrastructures to eliminate their constrained legacy control systems and equipment.

Apéndice A

Resumen

Las infraestructuras críticas (IICC) modernas son vastos sistemas altamente complejos, que precisan del uso de las tecnologías de la información para gestionar, controlar y monitorizar el funcionamiento de estas infraestructuras. Estos sistemas informáticos se denominan sistemas de control (SSC) y, dado su carácter imprescindible para el funcionamiento correcto de las IICC, se consideran sistemas críticos en sí mismos. Las IICC, de manera interrelacionada entre ellas, conforman una intrincada red que proporciona los servicios indispensables para el buen funcionamiento de la sociedad (gestión de electricidad, agua, transporte, etc.). Debido a sus funciones esenciales, la protección y seguridad de las infraestructuras críticas, y por tanto, de sus sistemas de control, se ha convertido en una tarea prioritaria para las diversas instituciones gubernamentales y académicas a nivel mundial. La interoperabilidad de las IICC, en especial de sus SSC, se convierte en una característica clave para que estos sistemas sean capaces de coordinarse y realizar tareas de control y seguridad de forma cooperativa. Esta nueva funcionalidad se convierte en el escenario actual en algo vital para establecer un estatus de seguridad y robustez que proporcione estabilidad y herramientas de actuación a estos sistemas críticos en presencia de eventos que amenacen su buen funcionamiento. Estos eventos, ya sean fallos internos del sistema o ciberataques industriales altamente sofisticados, amenazan nuestras infraestructuras cada día, es de vital importancia, por tanto, encontrar mecanismos de cooperación y protección que ayuden a establecer un funcionamiento seguro de las mismas.

El objetivo de esta tesis se centra, por tanto, en proporcionar herramientas para la interoperabilidad segura de los diferentes SSC, especialmente los sistemas de control ciber-físicos (SCCF), de forma que se potencie la intercomunicación y coordinación entre ellos para crear un entorno en el que las diversas infraestructuras puedan realizar tareas de control y seguridad cooperativas. Los esfuerzos de nuestro trabajo están centrados en crear una base de conocimiento que permita extraer las características necesarias para crear una plataforma de interoperabilidad segura capaz de dar servicio a diversas IICC, creando un entorno de *consciencia situacional* (del inglés *situational awareness*) de alto espectro o área (*wide-area*).

A.1 Marco de la tesis, objetivos y contribuciones

Para contextualizar nuestra investigación, es importante tener en cuenta el marco industrial y las tendencias actuales que dan forma al diseño y las características de los sistemas de control que los manejan. Este contexto industrial en el que se centra esta tesis es la *Industria 4.0*, también llamada *cuarta revolución industrial*, que se refiere al movimiento que actualmente se está produciendo para guiar a la industria hacia la adopción de nuevas tecnologías y paradigmas, como son los sistemas ciber-físicos (SCF), el internet de los objetos (del inglés *Internet of Things* - IoT), etc. Como se puede observar en la Figura A.1, las diferentes revoluciones industriales han estado marcadas por la incorporación de distintas tecnologías a la industria (la mecanización, la cadena de montaje, la automatización). Esta cuarta revolución industrial se centra en la aparición de los nuevos componentes y tecnologías previamente mencionados en este escenario siguiendo los principios de *interconexión* e interoperabilidad, *transparencia de la información*, *decisiones decentralizadas*, y *asistencia técnica* [10].

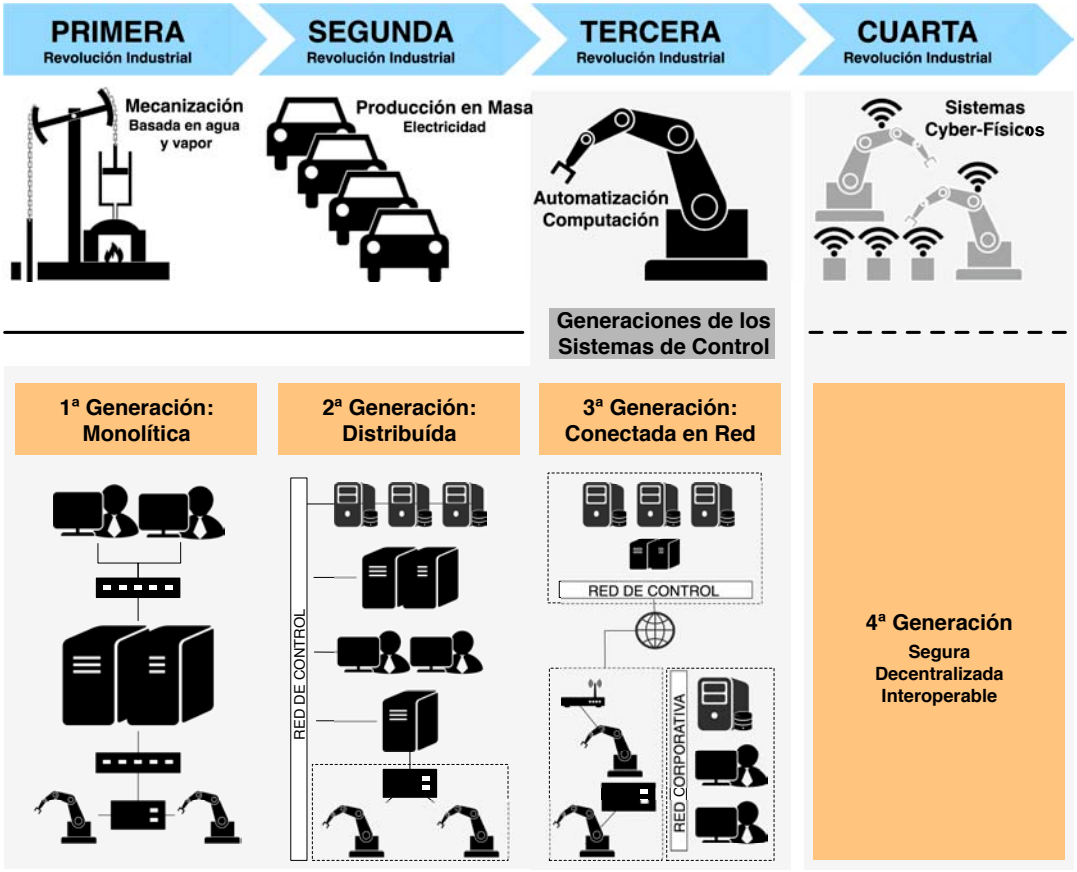


Figura A.1: Evolución de la industria y de sus sistemas de control a través de las generaciones, adaptado de [1] y [2]

Dado que los sistemas de control y monitorización son también elementos claves para la operación de los sistemas industriales (junto con los principios y tecnologías mencionados), entendemos que uno de los principales retos para la Industria 4.0 es adaptar los procesos de control utilizados

tradicionalmente durante la tercera revolución industrial a este nuevo escenario y paradigma. En la Figura A.1 podemos observar como los sistemas de control han ido evolucionando a lo largo de los años para acomodarse a las nuevas necesidades. Comprender las características y capacidades de estos sistemas de control nos resulta de interés para poder extraer los requisitos que el paradigma de la Industria 4.0 requiere para afrontar los nuevos retos y escenarios que se presentan.

En nuestro análisis no utilizaremos sistemas de control para una industria genérica, si no que centraremos nuestro estudio en los sistemas de control a cargo de gestionar el *smart grid* (SG) [11], el cual se puede considerar como la versión inteligente de la industria energética tradicional cuando se incorpora al paradigma de la Industria 4.0. En especial, y dentro del SG, nos centramos en sus sistemas de control industrial (SSCI), es decir aquellos SSC que realiza tareas de gestión y regulación del comportamiento de otros dispositivos y sistemas en las infraestructuras industriales (ejemplos de estos SSCI son los sistemas SCADA (del inglés *Supervisory Control And Data Acquisition*) o los sistemas DCS (del inglés Distributed Control Systems)), y en los sistemas de control ciber-físicos.

Como podemos ver en la Figura A.1, desde su introducción en los años 60, los sistemas de control han ido evolucionado a través de tres generaciones: la *Monolítica*, la *Distribuída* y la *Conectada en Red* [1, 12], hasta el presente en el cual entramos en una nueva generación de sistemas de control. Dadas las características de la Industria 4.0, y de los diferentes SSC existentes hasta la fecha, esta nueva *Cuarta Generación* de SSC se caracterizará por (i) *Seguridad* intrínseca al diseño, (ii) procesos de control *descentralizados*, e (iii) *interoperabilidad* garantizada entre los componentes heterogéneos que coexisten en este escenario.

Por tanto, en este entorno donde se incluyen tecnologías tales como el IoT, los SCCF o el cloud, se necesitan SSC suficientemente flexibles para aprovechar las ventajas que ofrecen (por ejemplo descentralización), siempre contemplando la seguridad del sistema desde el punto de vista del diseño. Por este motivo, el diseño y especificación de las características de esta cuarta generación de SSC es un campo de investigación puntero que presenta numerosos retos a la hora de crear un modelo arquitectural que tenga en cuenta las consideraciones mencionadas anteriormente. En esta tesis afrontamos estos retos con el objetivo de proporcionar una solución a este problema, por tanto, las principales contribuciones de esta tesis se pueden resumir en los siguientes puntos:

- Revisamos las amenazas de carácter más sofisticado que amenazan la operación de los sistemas críticos, particularmente enfocándonos en los ciberataques y los ciberataques camuflados (del inglés *stealth*) que amenazan los sistemas de control de infraestructuras críticas como el smart grid. Dado que las anomalías y fallos en el sistema se han estudiado en profundidad en el contexto de las IICC, enfocamos nuestra investigación al análisis y comprensión de este nuevo tipo de ataques que aparece contra los sistemas críticos, y a las posibles contramedidas y herramientas para mitigar los efectos de estos ataques.
- Examinamos e identificamos los requisitos especiales que limitan el diseño y la operación de una arquitectura de interoperabilidad segura para los SSC (particularmente los SCCF) del smart grid. Nos enfocamos en modelar requisitos no funcionales que dan forma a esta infraestructura, siguiendo la metodología NFR para extraer requisitos esenciales, técnicas para la satisfacción de los requisitos y métricas para nuestro modelo arquitectural. Realizamos este proceso de extracción de requisitos dos veces, en primer lugar para extraer el

modelo para la plataforma de interoperabilidad segura, y en segundo lugar para modelar los sistemas de protección adecuados a la infraestructura.

- Estudiamos los servicios necesarios para la interoperabilidad segura de los SSC del SG revisando en profundidad los mecanismos de seguridad, desde los servicios básicos hasta los procedimientos avanzados capaces de hacer frente a las amenazas sofisticadas contra los sistemas de control. Nuestro análisis se divide en diferentes áreas: prevención, consciencia y reacción, y restauración; las cuales general un modelo de seguridad robusto para la protección de los sistemas críticos.
- Proporcionamos el diseño para un modelo arquitectural para la interoperabilidad segura y la interconexión de los SCCF del smart grid. Este escenario contempla la interconectividad de una federación de proveedores de energía del SG, que interactúan a través de la plataforma de interoperabilidad segura para gestionar y controlar sus infraestructuras de forma cooperativa. La plataforma tiene en cuenta las características inherentes y los nuevos servicios y tecnologías que acompañan al movimiento de la Industria 4.0.
- Presentamos una prueba de concepto de nuestro modelo arquitectural, el cual ayuda a validar el diseño propuesto a través de experimentaciones. Creamos un conjunto de casos de validación que prueban algunas de las funcionalidades principales ofrecidas por la arquitectura diseñada para la interoperabilidad segura, proporcionando información sobre su rendimiento y capacidades.

A.2 Ataques avanzados a los sistemas de control ciber-físicos

Las IICC, especialmente sus SSC, se han convertido paulatinamente en el objetivo de diferentes y sofisticados ciberataques en los últimos años. Los sistemas críticos son especialmente vulnerables a los ciberataques, y los organismos globales de respuesta de emergencia (CERT - del inglés *Computer Emergency Response Teams*) ante estas situaciones, confirman la intensificación del número de ciberataques maliciosos cuyo objetivo es causar alteraciones en el servicio de las IICC. Especialmente peligrosos y dañinos para las IICC son los ciberataques camuflados, que son capaces de causar grandes daños a los sistemas críticos, o generar filtraciones de datos sensibles sin el conocimiento de los administradores de servicio.

Existen numerosos tipos de ciberataques dirigidos a las infraestructuras críticas, cuyos objetivos y características varían en sofisticación (desde ataques físicos sencillos para estropear ciertos dispositivos, hasta ataques a gran escala como el gusano Stuxnet [48]). Hay mucha literatura científica e informes de empresas de seguridad dedicados a la revisión de los ciberataques camuflados a gran escala. Sin embargo, teniendo en cuenta el escenario de nuestra tesis, donde nos dedicamos al estudio de los nuevos sistemas de control para la Industria 4.0, en nuestro análisis, identificamos un tipo diferente de ciberataque camuflado que no se ha estudiado previamente en el contexto de las IICC.

Nos referimos a aquellos ciberataques que emplean métodos y prácticas provinientes de las redes de TI (tecnologías de la información) de carácter general, y que en esta nueva generación de SSC, donde encontramos numerosos SCCF, presentan una amenaza creciente. Estos ataques tienen como objetivo redes y dispositivos de control que eran previamente inexistente en las IICC, y

que ahora (si no se encuentran protegidos) introducen nuevas vulnerabilidades en los sistemas críticos.

Para entender mejor la naturaleza y alcance de estos ciberataques, desarrollamos una taxonomía para clasificar los ataques, basándonos en las propiedades de seguridad *Disponibilidad* (A - del inglés *Availability*), *Integridad* (I) y *Confidencialidad* (C), AIC, y extendiéndola para introducir indicadores que caractericen los diferentes tipos de anomalías que se producen en las IICC: *Anomalías Infrasestructurales* (InfAn), relacionadas con eventos físicos que ocurren a la propia infraestructura; *Anomalías de Control* (CAn), correspondientes a la alteración de los sistemas de control causados por fallos, errores o intrusiones; y *Anomalías de Intrusión* (IntrAn), asociadas a las acciones maliciosas llevadas a cabo contra la infraestructura o sus sistemas de control y que pueden causar incidentes imprevistos. Estos indicadores se pueden utilizar de forma compuesta indicando la aparición de combinaciones de anomalías, la Figura A.2 ilustra nuestra taxonomía AICAn y los tipos de amenazas de seguridad que contempla.

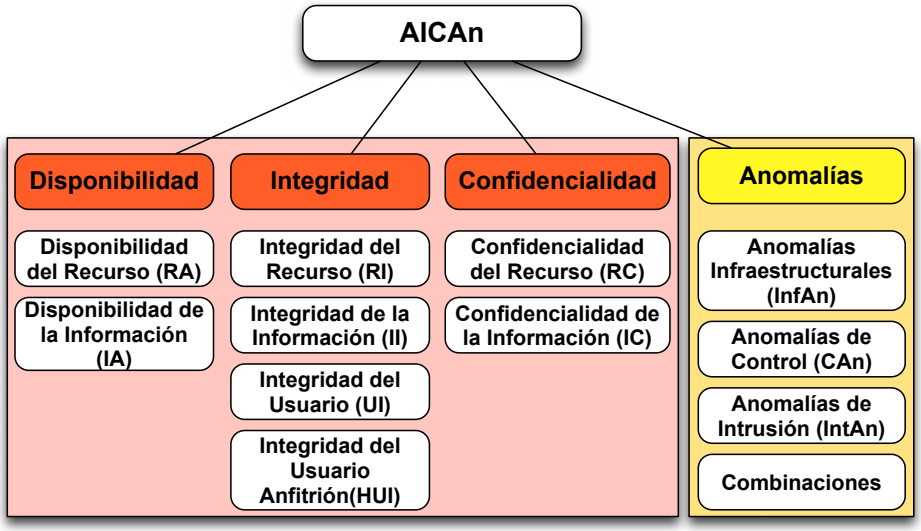


Figura A.2: Taxonomía AICAn

Como vimos previamente, los ciberataques camuflados ocurren en un escenario donde el objetivo del adversario no es únicamente llevar a cabo el ataque, si no hacerlo con el mínimo esfuerzo posible y ocultando su existencia y sus actividades en la medida de lo posible. Por tanto, los atacantes hacen uso de métodos o armas con este propósito: *suplantación*, *mentiras* y *sobrecarga*. La suplantación ataca la integridad del sistema (I) introduciendo paquetes cuyo origen declarado es diferente del origen real (algo que puede realizarse mediante spoofing de direcciones IP o frecuencias de comunicaciones aplicadas a otros). Los métodos basados en mentiras también amenazan la integridad del sistema (I), permitiendo al atacante propagar información incorrecta (por ejemplo tablas de enrutado). Las sobrecargas amenazan la disponibilidad del sistema (A), como en los ataques de denegación del servicio, donde se inyectan mensajes inválidos en la red para sobrecargarla.

Basándonos en nuestra taxonomía AICAn y en la clasificación de métodos y armas para los ataques camuflados, realizamos un análisis y clasificación de los ciberataques camuflados que

encontramos en la literatura (para redes generales) y analizamos su nivel de sofisticación, camuflaje y peligrosidad en el contexto de las IICC. Encontramos cinco categorías principales dentro de los ciberataques camuflados para los sistemas críticos según el objetivo del adversario: (i) ataques de desconexión y reducción del rendimiento de la red (*disconnection and goodput reduction*) [60], (ii) escuchas o espionaje activo (*active eavesdropping*) [60], (iii) escaneado y exploración (*scanning and probing*) [70], (iv) canales encubiertos y laterales (*covert and side channel exploitation*) [71, 72, 73], e (v) inyección de código (*code injection*) [73, 74].

La Tabla A.1 resume el análisis realizado sobre los ciberataques camuflados, proporcionando una visión sobre como éstos amenazan las IICC en relación con lo estudiado en este capítulo (Capítulo 2). En la tabla se puede ver los diferentes ataques divididos en áreas y categorías de amenazas, y se puede observar que los ataques están estrechamente relacionados entre sí debido a que los objetivos de los atacantes, independientemente del método, son comunes. En la columna denominada camuflaje se puede observar el nivel de camuflaje que aplica cada uno de los ataques estudiados con respecto a las fases de comunicación, ejecución y propagación del ciberataque, teniendo en cuenta su nivel de riesgo de acuerdo a la taxonomía AICAn y considerando el peor escenario posible, es decir, la máxima penetración del ataque en la infraestructura en caso de ser exitoso.

Por otra parte, en la tabla introducimos también una serie de contramedidas sugeridas que se pueden aplicar para prevenir o reaccionar ante estos ciberataques camuflados. Estas contramedidas provienen de un estudio de los métodos de reacción y prevención que se proponen en la literatura para redes generales, y que en casos de IICC es posible aplicar con ciertas adaptaciones para proteger los entornos críticos. Como podemos observar, las principales acciones que encontramos en caso de protección de sistemas críticos (SCCF y otros SSC) son mecanismos preventivos, tales como reputación o criptografía, dirigidas a proteger los elementos de control ante perturbaciones en las propiedades de seguridad del sistema. Asimismo, la protección de los canales de comunicación se puede basar en estas técnicas, las cuales protegen contra ataques a la confidencialidad del sistema. Sin embargo, ante ciberataques camuflados de mayor complejidad, es necesario introducir otra capa de protección para el sistema que proporcione mecanismos de respuesta y recuperación automáticos capaces de actuar antes o durante un ataque de este tipo. Herramientas que son útiles en este sentido son los módulos IDS e IPS, capaces de proporcionar detección de dinámicas sofisticadas en el sistema y alertar a los administradores de la presencia de un ataque. Estas herramientas están siendo objeto de gran investigación en la actualidad, para adaptar su funcionamiento al entorno delicado y vulnerable presente en las IICC.

Tabla A.1: Tabla resumen sobre los ciberataques camuflados

Categoría	Ataque CamufladoStealth Attack	Camuflaje			Armas	Contramedidas
Desconexión y reducción del rendimiento	Nodos inalcanzables	○			Suplantación Mentiras Sobrecarga	Criptografía Mecanismos de reputación
	Eliminación de rutas en tablas	○	❍			
	Reducción del rendimiento	○	❍			
Escucha activa	Secuestro de tráfico	○	❍			Criptografía Mecanismos de reputación
	Modificación de las tablas de enrutamiento	○	❍			
Escaneado y exploración	Escaneado TCP connect()	○				Exploración camuflada Honeypots
	EscaneadoTCP SYN	○				
	Escaneado TCP FIN	○				
	Escaneado Christmas	○				
	Escaneado Null	○				
Explotación de canales laterales	Ataque de temporización	○				Ocultar variaciones de temporización Técnicas de ofuscamiento Técnicas de enmascarado Fundas protectoras IDS y validación de cómputos Enmascaramiento de señales luminosas Codificación y enmascaramiento del canal
	Ataque de análisis del consumo energético	○				
	Ataque de análisis electromagnético	○				
	Ataque de inducción de fallos	○				
	Ataque de canal lateral óptico	○				
	Ataque de análisis de tráfico	○				
Explotación de canales encubiertos	Con virus y malware	○	❍	●		Anti-malware Monitorización de recursos Seguridad específica para datos sensibles Protocolos seguros Evaluación y corrección del diseño Uso de IDS Políticas para permisos Restricciones en los handshake Acceso restringido a recursos públicos Mecanismos de autenticación
	Con recursos importantes como objetivo	○	❍			
	Con datos sensibles como objetivo	○	❍			
	Utilizando protocolos vulnerables	○	❍	●		
	Utilizando fallos de diseño	○	❍	●		
	Utilizando cabeceras de paquete	○	❍			
	Utilizando permisos de super usuario	○	❍	●		
	Utilizando pruebas de handshake	○	❍			
	Utilizando recursos públicos	○	❍			
	Utilizando la falta de autenticación	○	❍			
Inyección de código	Basado en fallos de diseño	○	❍	●		Herramientas de monitorización Mecanismos de validación y prevención
	Basado en la propagación del malware	○	❍	●		

○: comunicación camuflada del ataque. ❍ : ejecución camuflada del ataque. ● : propagación camuflada del ataque.

A.3 Análisis de requisitos para la interoperabilidad segura de sistemas de control ciber-físicos

Como hemos visto anteriormente, prácticamente todas las infraestructuras críticas dependen de los sistemas de supervision y control, de forma que las TI se han convertido en elementos esenciales para nuestra sociedad, mejorando el rendimiento de las infraestructuras, reduciendo el coste y mejorando nuestra calidad de vida. Sin embargo, dada la gran penetración de las nuevas tecnologías de la información en las infraestructuras críticas y la migración de estas al nuevo paradigma de la Industria 4.0, estos sistemas críticos cada vez se hacen más complejos e interdependientes entre ellas. Debido a esto, surgen nuevos y sofisticados ataques a las infraestructuras que tienen como objetivo aprovechar las nuevas vulnerabilidades (como vimos en la Sección A.2) que se introducen con dichas nuevas tecnologías.

De esta situación surge la fuerte necesidad de diseñar y crear nuevos mecanismos de seguridad protección para los sistemas críticos, en particular para los SSC y los SCCF que gestionan el funcionamiento de estas infraestructuras, para garantizar su correcto funcionamiento en todo momento, manteniendo intactas las condiciones operacionales de disponibilidad, integridad, acceso, vida útil, gestión y fiabilidad. Con el objetivo de abordar este problema desde el punto de vista de la ingeniería del software, dedicamos el Capítulo 3 de esta tesis a analizar los requisitos y restricciones que tienen los sistemas críticos para implementar diferentes soluciones de interoperabilidad y protección. Para ello basamos nuestro estudio en los principios de seguridad básicos para las infraestructuras de control [21]: *operación en tiempo real, fiabilidad, supervivencia, sostenibilidad* y *seguridad crítica* (en inglés *real-time operational performance, dependability, survivability, sustainability, safety critical*).

Partiendo de este análisis, aplicamos dos metodologías, el Framework NFR [119] y el GQM [129] para realizar un estudio que nos ayude a determinar las características y el diseño para la creación de una arquitectura de interoperabilidad descentralizada y segura para los sistemas de control (especialmente los SCCF) del SG. Nuestro estudio consta de tres fases: el *análisis de los requisitos*, la *identificación de las técnicas de satisfacción* y la *identificación de métricas representativas*. Centramos la primera etapa de nuestro análisis en la identificación de requisitos no funcionales (aquellos que representan características de alto nivel e información del sistema) de forma iterativa para determinar el conjunto de requisitos que deben tenerse en cuenta a la hora de crear una plataforma de interoperabilidad segura para los SSC del SG.

La segunda fase de nuestro estudio se centra en determinar los métodos que hacen posible cumplir los requisitos previamente identificados. Para ello, siguiendo la metodología NFR, proporcionamos una serie de técnicas de satisfacción que permiten al sistema cumplir cada uno de los requisitos identificados en la primera fase del estudio. Dentro de estas técnicas encontramos diferentes subconjuntos, por ejemplo: técnicas que potencian la protección de los SSC de las IICC, técnicas para mejorar la robustez estructural y la integridad de la plataforma de interoperabilidad y de sus infraestructuras de comunicaciones, técnicas para garantizar la interoperabilidad en sí misma, técnicas que permiten la integración de los nodos IoT dentro de la infraestructura, o técnicas relacionadas con la virtualización de servicios dentro de la plataforma.

Con el objetivo de concretar los resultados del estudio basados en las características abstractas obtenidas mediante la metodología NFR, utilizamos la aproximación GQM (explicado en detalle en el Capítulo 3) para identificar un conjunto de métricas que nos ayuden a analizar

de forma cuantitativa los requisitos extraídos en las dos etapas previas del análisis. Las métricas propuestas son clasificadas en los siguientes grandes conjuntos: *mantenimiento*, *fiabilidad y disponibilidad*, *rendimiento*, *seguridad y confiabilidad*, *respuesta*, *auditoría*, *interoperabilidad*, *virtualización*, *monitorización y restauración*, y *autorización y delegación*. Cada una de las métricas va asociada a uno o varios de los requisitos obtenidos en la primera fase del estudio. Cada conjunto de métricas tiene asociado una serie de controles, estándares o guías de referencia tales como: el NIST *Framework for Improving Critical Infrastructure Cybersecurity* [131], el NISTIR 7628 [20], el NIST SP800-82 [112], la guía NISCC [133], la serie IEC/TS 62351 [134], el NERC-CIP [113], el ISO/IEC 27002 [135], el IEEE 2030 [11], el NIST SP800-144 [127] o las guías NFV [126]. Estas guías son útiles para continuar el estudio de las métricas, formalizándolas y concretándolas teniendo en cuenta las características de los sistemas donde se vayan a aplicar, en un cliclo o iteración posterior a nuestro estudio.

Un caso particular de la aplicación de nuestra metodología de análisis en tres pasos es la aplicación de este estudio para determinar si una solución de protección (modular, por ejemplo un sistema IDS) es adecuado para su despliegue en un entorno industrial crítico, perteneciente a la Industria 4.0, junto con los SSC que lo gestionan. Para determinar esta adecuación, aplicamos nuestra metodología de análisis de requisitos técnicas y métricas para identificar las características necesarias para que las soluciones de detección y protección para los SSC sean compatibles con la IC y su entorno crítico. Realizamos este proceso para un módulo IDS genérico, pero sería igualmente factible realizar el estudio para cualquier solución modular que se quiera incorporar en este contexto. Posteriormente, dados un sistema de protección concreto y una infraestructura en particular, sería posible finalizar las iteraciones de refinamiento de requisitos, técnicas y métricas para entender en profundidad las características de este sistema y su adecuación, cuantificada, a la infraestructura crítica concreta fruto del análisis.

A.4 Servicios para la interoperabilidad segura de sistemas de control ciber-físicos

En esta sección estudiamos los diferentes servicios de seguridad que precisa una arquitectura de interoperabilidad descentralizada para los SCCF que controlan el SG, de acuerdo a los componentes clave y características que impone el paradigma de la Industria 4.0 para la nueva generación de SSc. Estos servicios de seguridad forman parte de las características clave para la cuarta generación de SSC, garantizando su correcto funcionamiento, la interoperabilidad de sus componentes, y evitando que las amenazas que ocurren en el sistema se extiendan a través de los componentes interrelacionados del sistema en forma de efecto en cascada.

Dividimos los servicios para la interoperabilidad segura de los SCCF (en particular, para los SSC en general) en cuatro categorías diferentes, como podemos ver en la Figura A.3: los *Servicios de Seguridad Básicos*, los servicios de *Prevención y Detección*, los servicios de *Consciencia y Reacción*, y los servicios de *Restauración*. Estos módulos, o conjuntos de servicios ayudan a introducir la interoperabilidad segura para los SCCF en las IICC, en especial en nuestro análisis particularizamos la seguridad en el smart grid.

El primer módulo se refiere a los servicios básicos de seguridad que deben estar presentes a cualquier nivel de la infraestructura de interoperabilidad para que ésta sea segura. Servicios

Figura A.3: Servicios de seguridad

como la *autenticación*, la *autorización*, y la responsabilidad (en inglés *accountability*) deben formar parte de la infraestructura de interoperabilidad para los SCCF del SG, y ser utilizados por todos los actores de la plataforma. Una vez establecida la presencia de estos servicios ubicuos, se pueden desplegar los servicios de seguridad avanzados (ver la Figura A.3) que mencionamos previamente.

El primero de estos servicios de seguridad avanzado se centra en la prevención y detección temprana de cualquier evento amenazante para los SSC (las IICC en general). Estos servicios deben estar localizados en diferentes secciones de la infraestructura de control del sistema, y monitorizar la operación de la infraestructura de interoperabilidad en sí misma para detectar posibles dinámicas peligrosas en los sistemas y lanzar mecanismos de mitigación para evitar fallos en cascada propagándose por los sistemas críticos interconectados. En el Capítulo 4 se describen en profundidad estos servicios, de entre los cuales destacamos las soluciones IDS. En nuestro análisis defendemos la necesidad de proporcionar soluciones de monitorización inteligentes y automáticas, capaces de aprender y adaptarse a las diferentes dinámicas que ocurren en las IICC, siempre y cuando estas soluciones respeten y cumplan los requisitos y restricciones que presentan las IICC para desplegar herramientas de protección (como se vio en el Capítulo 3).

El segundo módulo de servicios avanzados de seguridad comprende aquellos servicios centrados en la implementación de estados de consciencia y mecanismos de reacción para la infraestructura. Estos mecanismos ayudan a establecer y mantener un estado de consciencia del entorno o contexto (*situational awareness*) y monitorizar el funcionamiento de la infraestructura (refiriéndonos a la IC en sí misma y a sus SSC), que permiten gestionar rápidamente los diferentes eventos del sistema de forma inteligente, automatizada y proporcionada. Existen diferentes implementaciones posibles para las capacidades que se requieren, sin embargo, en nuestro estudio defendemos la necesidad de soluciones de reacción proactivas e inteligentes capaces de prevenir amenazas a la infraestructura desde el minuto cero, para evitar cualquier daño que se pueda producir a la IC y al delicado funcionamiento de los sistemas críticos.

El tercer y último componente de los servicios de seguridad para la interoperabilidad de los SCCF del SG corresponde con las herramientas y soluciones para la restauración de los servicios de la infraestructura. Estos mecanismos son activados y se hacen cargo de recuperar el correcto funcionamiento del sistema cuando un evento amenazador ha producido un impacto en la in-

fraestructura (fallos, degradación del rendimiento del sistema o de sus redes de comunicación, etc.). Existen técnicas variadas para implementar medidas de restauración, sin embargo, normalmente implican la introducción de elementos redundantes y de *backup* en la infraestructura para realizar las tareas de recuperación. Pese a esto, las medidas de restauración deben estar presentes a todos los niveles de la arquitectura de un sistema crítico, dado que cualquier fallo o error en el sistema puede desequilibrar el funcionamiento de la IC y debe ser remediado inmediatamente.

Una vez descritos los diferentes servicios de seguridad necesarios para una plataforma de interconexión e interoperabilidad segura para los SCCF, analizamos estas técnicas y mecanismos en el contexto de las distintas redes de una infraestructura SG. En este escenario, analizamos las técnicas y tecnologías específicas aplicables a cada sección de la infraestructura, por ejemplo las *fábricas*, *subestaciones*, o las *redes de los vecindarios*, estudiando en cada caso las características y requisitos específicos para estos dominios, y la adecuación de las técnicas mencionadas para cada uno de ellos. Como en secciones anteriores, sostenemos que es necesaria la inclusión de mecanismos de seguridad automáticos e inteligentes en estos escenarios, para proporcionar sistemas de vigilancia y seguridad capaces de conseguir una detección, prevención y protección eficaces para los sistemas críticos.

A.5 Diseño de una plataforma de interoperabilidad segura para sistemas de control ciber-físicos

El Capítulo 5 de esta tesis se centra en proponer un diseño en forma de modelo arquitectural para la interoperabilidad segura de los SSC, y en particular los SCCF, del SG. En nuestro escenario, contemplamos las interacciones y cooperación entre diferentes proveedores de SG que forman una federación. Dichos proveedores intercambian información crítica sobre determinadas funciones de sus infraestructuras y comparten tareas de gestión y mantenimiento de algunas de las partes de sus infraestructuras que manejan en común. También comparten la responsabilidad para asistir y manejar sus sistemas críticos cuando ocurren eventos peligrosos dentro de sus respectivas infraestructuras o en las áreas comunes, proporcionando métodos adecuados para la protección, mitigación, respuesta y restauración de sus sistemas.

Para crear este modelo arquitectural, basamos nuestro diseño en el análisis detallado que se presenta en los Capítulos 3, 4 y 2 de la tesis, donde se extraen los principales requisitos, recomendaciones y técnicas para conseguir la interoperabilidad segura entre sistemas críticos y se define un modelo de amenazas para la plataforma a tener en cuenta. Dado que este escenario contempla diferentes tecnologías muy heterogéneas (como mencionamos previamente coexisten sistemas anticuados con nuevas tecnologías como fog computing, sensores inalámbricos, etc.), definimos nuestra arquitectura como perteneciente al paradigma de la industria 4.0, donde se deben tener en cuenta que los SCCF, las tecnologías del cloud, los dispositivos IoT deben estar correctamente integrados e interactuar de forma eficiente y cooperativa con la infraestructura crítica subyacente.

Por tanto, proponemos un modelo arquitectural para la interoperabilidad segura de los SSC del SG, tal y como está ilustrado en la Figura A.4. Como podemos observar en la figura, proponemos un escenario de interoperabilidad para diferentes infraestructuras de SG (la federación de

proveedores de energía), cada una comprendida de diferentes subredes y sistemas remotos. Presentamos un entorno distribuido y descentralizado que hace uso de tecnologías cloud y fog para permitir la inclusión de otros sistemas dentro de la infraestructura, como por ejemplo los sensores inalámbricos y el IoT, contemplando por tanto un escenario donde las nuevas tecnologías de la industria 4.0 están representadas.

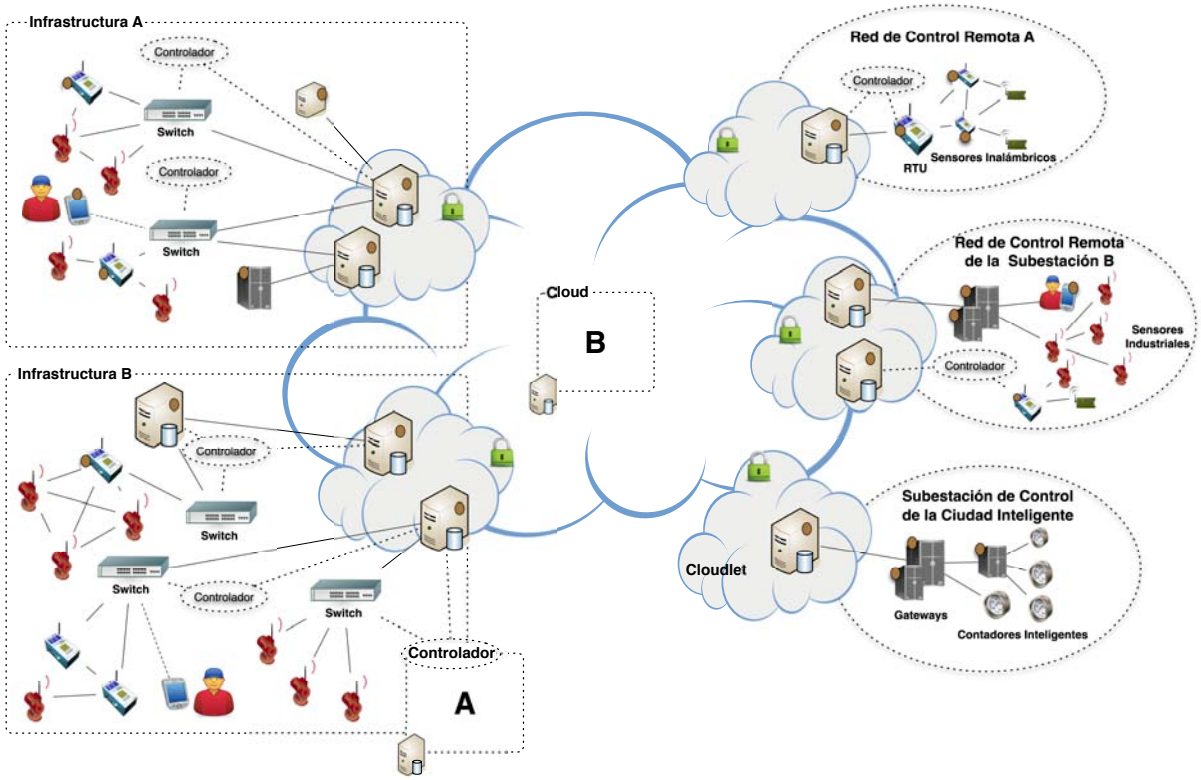


Figura A.4: Infraestructura para la interoperabilidad segura del smart grid

La Figura A.4 presenta, por tanto, nuestra aproximación para resolver la interoperabilidad segura de los SCCF del SG, en esta figura se puede diferenciar un módulo central representando la *infraestructura cloud*, que sustenta la interoperabilidad entre actores de la plataforma a nivel global. Existen diferentes clouds más pequeños, denominados *cloudlets*, que representan al a infraestructura de *fog computing*, desplegada a nivel más bajo en la arquitectura, es decir, relacionadas con cada sector independiente de las infraestructuras (subestaciones, redes remotas, etc.). Dentro de cada uno de estos cloudlets, es posible encontrar uno o varios *controladores*, que sirven como puntos de acceso inteligentes para los dispositivos de la plataforma, y que permiten realizar las principales tareas de interoperabilidad para la infraestructura. Finalmente, podemos observar que en la figura están representadas diversos tipos de redes de redes de control pertenecientes a las infraestructuras SG, tales como redes corporativas, redes remotas de control, subsestaciones SG, o redes de control IoT.

Cada uno de los módulos controladores presentes en la Figura A.4 (denotados por A) proporciona una serie de servicios a la infraestructura que están descritos a parte en la Figura A.5. Igualmente, los servicios proporcionados por el cloud (denotados por B) están ilustrados separadamente en la Figura A.6. Adicionalmente, existe otro módulo ilustrado dentro de la plataforma que está

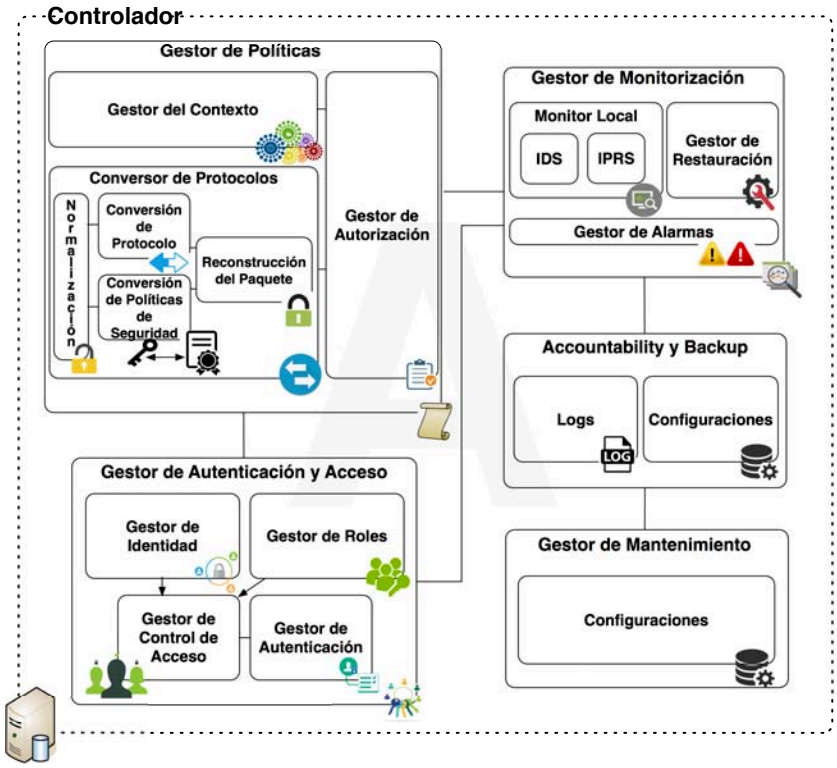


Figura A.5: Servicios ofrecidos por los módulos controlador

representado por un candado verde, este se refiere a los servicios de seguridad y monitorización avanzados incluidos dentro de la infraestructura, que quedan ilustrados en la Figura A.7. Como podemos ver en estas figuras, nuestro modelo propuesto contiene las siguientes tecnologías y características:

- *Mecanismos de control de acceso*, para permitir autenticar a los diferentes actores del sistema, siempre en línea con los servicios de seguridad básicos previamente identificados (ver Sección A.4). Para proporcionar control de acceso, hacemos uso del estándar RBAC definido en el documento IEC/TS 62351-8 [28].

- *Soluciones de redes definidas por software* (del inglés Software Defined Networking - SDN), que permiten la inclusión de nuevos y sofisticados mecanismos a nivel de subred, que permiten soluciones de enrutado novedosas, al igual que procedimientos de interoperabilidad, prevención avanzada, consciencia y protección.

- *Infraestructuras cloud y fog*, que permiten la introducción de servicios virtualizados, la inclusión de soluciones de enrutado sofisticadas basadas en el paradigma SDN, soporte adicional para los procesos de interoperabilidad, al igual que medios para mejorar la fiabilidad y supervivencia de una infraestructura operativa para los SCCF del SG.

- *Una estrategia de defensa en profundidad* (ver Figura A.7) que implementa procedimientos de seguridad a diferentes niveles, garantizando que siempre están presentes los servicios de seguridad básicos para la plataforma (ver Sección A.4). Esta estrategia incluye la uti-

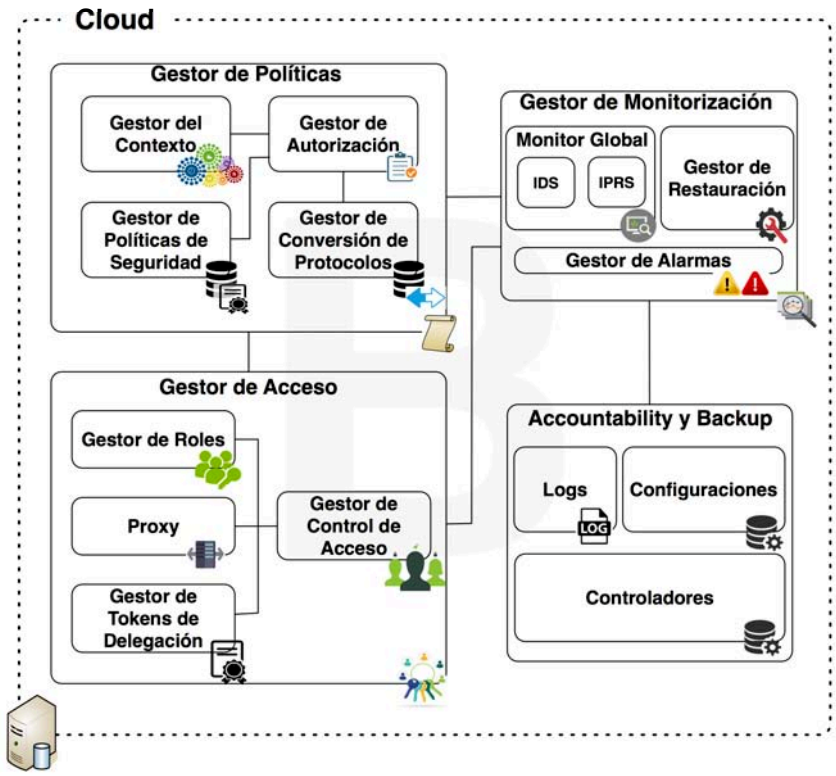


Figura A.6: Servicios ofrecidos por el cloud

lización de un sistema de diagnóstico sofisticado en todo el conjunto de la plataforma, basado en el estándar IEC/TS 62351-7 [142]. Este estándar describe la especificación de los objetos NSM, desplegados dentro de la infraestructura de control para recoger información de monitorización y del contexto de la plataforma para proporcionar guías a los procedimientos de interoperabilidad y seguridad.

- La implementación *de mecanismos de delegación* para asegurar que los eventos críticos se atienden rápidamente y en todo momento dentro de la infraestructura, incluso cuando el personal asignado para hacerse cargo de estas situaciones no está disponible. Para implementar esta característica, hacemos uso del modelo RBAC con roles primarios y secundarios que pueden ser activados en casos de emergencia. El uso de roles secundarios, al igual que los mecanismos de delegación basados en técnicas de *proxy re-encryption* permite que diferentes miembros de la federación de proveedores del SG puedan hacerse cargo de eventos críticos que ocurren en cualquier parte de la plataforma de interoperabilidad segura, proporcionando de esta forma mecanismos de protección y mitigación robustos para la infraestructura.

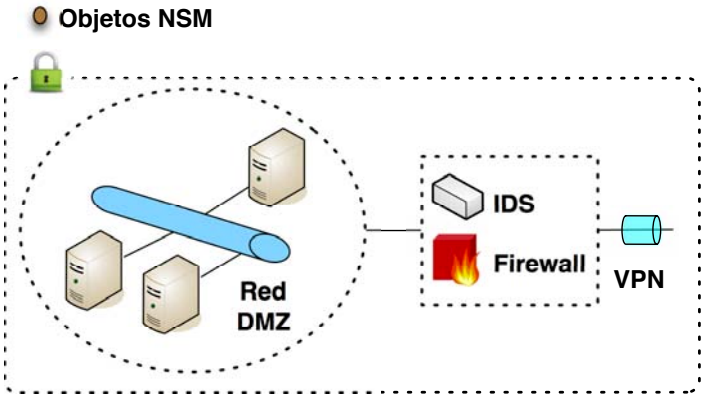


Figura A.7: Defensa en profundidad y sistema de diagnóstico

A.6 Experimentación y validación del modelo arquitectural propuesto

Una vez diseñado el modelo arquitectural para la interoperabilidad segura de SSC en entornos críticos, es nuestro objetivo analizar en profundidad las principales características y comportamientos del modelo descrito a través de una serie de experimentos. Estos experimentos nos ayudan a entender mejor las dinámicas y los problemas que aparecen en un escenario interconectado y complejo, donde los SCCF interoperan para comunicarse y colaborar entre ellos. Para ello, creamos un escenario de pruebas (*testbed*) donde implementamos una versión simplificada de nuestro diseño para comprobar su funcionamiento. Con la creación de este testbed, nuestra intención es validar a través de pruebas y experimentos la funcionalidad y la adecuación de nuestro modelo arquitectural al escenario en el cual los SSC de diferentes IICC necesitan interactuar vía la plataforma de interoperabilidad segura, para realizar tareas de forma cooperativa.

Dada la complexidad del diseño de nuestro modelo arquitectural (ver Sección A.5), para nuestros experimentos creamos una versión reducida de la plataforma para la interoperabilidad segura de SSC, donde probamos algunas de sus funcionalidades principales, dejando la implementación del prototipo en su totalidad como trabajo futuro. En esta tesis, nos centramos en la validación de la funcionalidad de interoperabilidad segura de la infraestructura a nivel de cloudlet, que es la entidad que proporciona interoperabilidad a los diferentes dispositivos conectados a la plataforma a nivel de subred.

El principal módulo de interoperabilidad a nivel de cloudlet es el controlador, que cumple las funciones de punto de acceso a la infraestructura de interoperabilidad para los diferentes sistemas y dispositivos que participan en la infraestructura federada. Para nuestros experimentos, nos centramos principalmente en la funcionalidad proporcionada por estos controladores, a través de la implementación de una versión simplificada del controlador descrito en la sección anterior. En nuestros casos de validación intentamos determinar si este controlador simplificado proporciona las características de interoperabilidad deseadas para nuestra infraestructura.

Creamos nuestro testbed para las simulaciones y experimentaciones haciendo uso del simulador y emulador de red Mininet (ver Capítulo 6), que nos permite incorporar protocolos de comunica-

ción industrial reales dentro del testbed. Hacemos uso de los protocolos IEC104 y Modbus/TCP en nuestro sistema. Esta posibilidad de incorporar protocolos reales, hace que nuestra implementación vaya más allá de una simple simulación, y nos permite adentrarnos en el funcionamiento de la interoperabilidad desde un punto de vista realista, haciendo frente a problemas reales de comunicaciones y a las dificultades que se encontrarían al desplegar el sistema real. Esta característica permite validar en la medida de lo posible nuestro modelo arquitectural, ya que hemos podido desarrollar una versión simple de un sistema industrial real operativo, que puede ser migrado a un entorno real realizando pocos cambios en la implementación.

Realizamos los experimentos en nuestro testbed con tres casos de validación centrados en analizar la operación del módulo *conversor de protocolos*, el *gestor de monitorización* en conjunto con el servicio de *accountability* y el módulo *gestor de autorización y acceso* (ver Figura A.5, para más detalles ver Capítulo 6). El módulo conversor de protocolos es capaz de proporcionar servicios de interoperabilidad transparente inter-red para los diferentes proveedores de SG presentes en la federación, de forma eficiente y sin verse afectados por el protocolo industrial utilizado. El servicio de gestión de la monitorización se encarga de detectar anomalías y ataques al sistema, independientemente del grado de camuflaje (*stealthiness*) del ataque, y de proporcionar respuestas adecuadas para paliar las amenazas generadas en el sistema, al igual que notificar vía alertas a los administradores del sistema para hacerse cargo de estas situaciones anormales. El módulo de gestor de acceso y autorización proporciona sus servicios basados en RBAC sin introducir costes (computacionales o de red) significativos en la plataforma de interoperabilidad. Por tanto, teniendo en cuenta las características de nuestro diseño, y los experimentos llevados a cabo, creemos que nuestro modelo arquitectural para la interoperabilidad segura de los SCCF del SG proporciona características y funcionalidades interesantes, muy necesarias a la hora de desplegarse en un entorno federado donde interactúan diversas IICC. Estamos convencidos de que este diseño es adecuado para el escenario propuesto de interoperabilidad segura.

A.7 Conclusiones y trabajo futuro

Esta tesis doctoral se ha centrado en el análisis y diseño de un sistema de interoperabilidad segura para los SSC de las infraestructuras críticas. En especial centramos nuestro estudio en los SCCF de las infraestructuras de SG en un entorno federado donde diversos proveedores de energía se comunican y coordinan para llevar a cabo diversas tareas de forma cooperativa. Este escenario está enmarcado en el nuevo paradigma industrial denominado Industria 4.0, el cual constituye un marco de trabajo conceptual en el que desarrollamos nuestro trabajo centrándonos en las redes de control industriales y las características que estas precisan para proporcionar las funcionalidades de interoperabilidad segura propuestas. Tres características clave guían nuestro diseño: la *descentralización*, la *seguridad* y la *interoperabilidad*. De acuerdo a estas características, desarrollamos nuestro trabajo de tesis, durante el cual afrontamos las necesidades y los retos que plantea el escenario sugerido, y podemos establecer las principales contribuciones de esta tesis:

- *Mejora de la comprensión de amenazas avanzadas* contra el funcionamiento normal de las IICC, en forma de ciberataques camuflados contra las infraestructuras de control. Realizamos un extenso análisis y revisión de la literatura relacionada, trasladando al entorno de las IICC ciberataques camuflados contra redes convencionales que pueden tener especial

impacto y relevancia en redes de control de sistemas críticos.

- *Identificación de los requisitos* y características del modelo arquitectural desde el punto de vista de la ingeniería del software. Para comprender mejor las necesidades de una nueva arquitectura para la interoperabilidad segura de los SCCF del SG, debemos proporcionar en primer lugar una base de conocimiento sólida que recoja los principales requisitos y restricciones que se presentan en este nuevo escenario.

- *Análisis de las características y adecuación de servicios de seguridad y soluciones de protección para SSC*, particularmente SCCF. Estudiamos los requisitos, restricciones y características que deben proporcionar las soluciones de protección para poder ser desplegados en entornos críticos de forma que lleven a cabo su funcionalidad de forma adecuada y sin causar ningún impacto sobre el rendimiento de la infraestructura subyacente.

- *Marco de trabajo y taxonomía para los sistemas de protección IDPRS para sistemas críticos.* Teniendo en cuenta los requisitos previamente estudiados, revisamos los diferentes servicios de seguridad disponibles para las IICC, desde aquellos servicios básicos, hasta los más avanzados y sofisticados. Estudiando en cada caso sus características, adecuación y beneficios aportados a los SSC donde pueden ser desplegados.

- *Diseño de un nuevo modelo arquitectural para la interoperabilidad segura de los SCCF del SG.* Esta arquitectura propone una infraestructura descentralizada capaz de proporcionar servicios de interoperabilidad a nivel local (subred - subestación) a través de módulos controladores, y a nivel global (infraestructura - federación de proveedores) a través de los servicios de infraestructuras cloud.

- *Desarrollo de un testbed y escenarios de validación para el modelo arquitectural propuesto*, que permiten entender mejor el comportamiento y el rendimiento del diseño propuesto, donde se prueban algunos de los principales servicios del módulo controlador de nuestro diseño.

Teniendo en cuenta estas contribuciones, también cabe destacar aquellos asuntos y áreas de interés que hemos podido identificar a lo largo del desarrollo de nuestro trabajo, cuyo ámbito excede el estudio realizado en esta tesis. En primer lugar, hemos identificado la necesidad de profundizar en la investigación para conseguir soluciones de protección IDPRS automáticas e inteligentes especialmente desarrolladas para su aplicación en el contexto de la protección de las infraestructuras críticas. Actualmente existen pocas de estas herramientas, y su funcionalidad es muy limitada y restringida.

Durante el proceso iterativo de la extracción y análisis de los requisitos para la interoperabilidad segura de los SSC, hemos tomado la decisión de diseño de parar nuestro proceso iterativo en un paso previo a la validación de requisitos y métricas, dado que para ello precisamos de un escenario de aplicación concreto y real donde llevarlo a cabo. Esta última iteración de especificación queda como trabajo futuro y será específica para cada caso de aplicación, ya que los requisitos y las métricas relevantes a cada tipo de infraestructura crítica y a cada subsistema específico variarán de acuerdo a sus características inherentes.

Para analizar en su totalidad las características y comportamiento del modelo arquitectural propuesto en esta tesis, es posible ampliar los casos de validación propuestos en el Capítulo 6 de validación, extendiendo el prototipo diseñado con las demás funcionalidades propuestas en el

Capítulo 5 de diseño. Debida la complejidad del diseño original, en esta tesis se proporciona un prototipo de validación con funcionalidades limitadas a los principales servicios de interoperabilidad segura, pero es posible completar el estudio desarrollando el modelo completo.

De forma adicional, es posible incluir en nuestro modelo arquitectural otras tecnologías y protocolos diferentes que sean compatibles con el escenario de la Industria 4.0, tales como la tecnología NFV (del inglés *Network Functions Virtualization*), que permitirán introducir nuevas características que mejoren la redundancia y la robustez de las IICC, y quizá puedan redefinir los sistemas de control del futuro, permitiendo a las infraestructuras eliminar sus sistemas y equipos más anticuados, sustituyéndolos por tecnologías modernas.

Bibliography

[1] C. Alcaraz, G. Fernandez, and F. Carvajal. Security Aspects of SCADA and DCS Environments. *Critical Infrastructure Protection*, 7130:120–149, 2012.

[2] Christoph Roser. Industry 4.0. AllAboutLean.com. [CC BY-SA 4.0 (http://creativecommons.org/licenses/by-sa/4.0)], via Wikimedia Commons. Last consulted, October 2016.

[3] C. Alcaraz and J. Lopez. Wide-Area Situational Awareness for Critical Infrastructure Protection. *IEEE Computer*, 46(4):30–37, 2013.

[4] N. Stakhanova, S. Basu, and J. Wong. A Taxonomy of Intrusion Response Systems. *International Journal of Information and Computer Security*, 1:169–184, 2007.

[5] A. Shameli-Sendi, N. Ezzati-Jivan, M. Jabbarifar, and M. Dagenais. Intrusion Response Systems: Survey and Taxonomy. *Int. J. Comput. Sci. Network Security*, 12(1):1–14, 2012.

[6] J. Bryson and PD. Gallagher. NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 3.0, September 2014.

[7] International Electrotechnical Commission et al. Telecontrol Equipment and Systems. Part 5: Transmission Protocols. Section 104: Network Access for IEC 60870-5-104 Using Standard Transport Profiles. *IEC-60870-5-104*, 2000.

[8] Luis M. Vaquero and Luis Rodero-Merino. Finding Your Way in the Fog: Towards a Comprehensive Definition of Fog Computing. *SIGCOMM Comput. Commun. Rev.*, 44(5):27–32, October 2014.

[9] Henning Kagermann, Johannes Helbig, Ariane Hellinger, and Wolfgang Wahlster. *Recommendations for implementing the strategic initiative INDUSTRIE 4.0: Securing the future of German manufacturing industry; final report of the Industrie 4.0 Working Group.* Forschungsunion, 2013.

[10] Mario Hermann, Tobias Pentek, and Boris Otto. Design Principles for Industrie 4.0 Scenarios. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 3928–3937. IEEE, 2016.

[11] IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads, 9 2011. IEEE2030.

[12] George Loukas. *Cyber-Physical Attacks: A Growing Invisible Threat*. Butterworth-Heinemann, 2015.

[13] Modbus-IDA. Modbus Application Protocol Specification, 2006.

[14] Gordon R Clarke, Deon Reynders, and Edwin Wright. *Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems*. Newnes, 2004.

[15] European Commission. *COM(2011) 163 - Communication on Critical Information Infrastructure Protection 'Achievements and Next Steps: Towards Global Cyber-Security'*. Publications Office, 2011.

[16] European Commission. *COM(2009) 149 - Protecting Europe from Large Scale Cyber-Attacks and Disruptions: Enhancing Preparedness, Security and Resilience*. Publications Office, 2009.

[17] European Commission. *Europe 2020: A Strategy for Smart, Sustainable and Inclusive Growth: Communication from the Commission*. Publications Office, 1 2010.

[18] Rodrigo Roman, Cristina Alcaraz, and Javier Lopez. A Survey of Cryptographic Primitives and Implementations for Hardware-Constrained Sensor Network Nodes. *Mobile Networks and Applications*, 12(4):231–244, 2007.

[19] Mica R Endsley. Toward a Theory of Situation Awareness in Dynamic Systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

[20] NISTIR 7628 - Guidelines for Smart Grid Cybersecurity: Vol. 1, Smart Grid Cybersecurity Strategy, Architecture, and High-Level Requirements, 2013.

[21] C. Alcaraz and J. Lopez. Analysis of Requirements for Critical Control Systems. *International Journal of Critical Infrastructure Protection*, 5(3):137–145, 2012.

[22] K DeSalvo and E Galvez. Connecting Health and Care for the Nation: A Shared Nationwide Interoperability Roadmap—version 1.0. *Health IT Buzz*, 2015.

[23] White House. The National Strategy for The Physical Protection of Critical Infrastructures and Key Assets, 2 2003.

[24] European Parliament. Cyber Security Strategy for the Energy Sector. DIRECTORATE GENERAL FOR INTERNAL POLICIES POLICY DEPARTMENT A: ECONOMIC AND SCIENTIFIC POLICY, 10 2016. IP/A/ITRE/2016-04 PE 587.333.

[25] Industrial Internet Consortium. The Industrial Internet of Things Volume G1: Reference Architecture. Industrial Internet Consortium (IIC) Technology Working Group, 2017.

[26] Industrial Internet Consortium. Industrial Internet of Things Volume G4: Security Framework. Industrial Internet Consortium Security Working Group, 2016.

[27] Cristina Alcaraz, Javier Lopez, and Stephen Wolthusen. Policy Enforcement System for Secure Interoperable Control in Distributed Smart Grid Systems. *Journal of Network and Computer Applications*, 59:301–314, 2016.

[28] International Electrotechnical Commission and others. IEC/TS 62351-8 Power Systems Management and Associated Information Exchange - Data and Communications Security - Part 8: Role-Based Access Control, September 2011.

[29] Erich Rome, Sandro Bologna, Erol Gelenbe, Eric HAM Luiijf, and Vincenzo Masucci. DIESIS: An Interoperable European Federated Simulation Network for Critical Infrastructures. In *Proceedings of the 2009 SISO European Simulation Interoperability Workshop*, pages 139–146. Society for Modeling & Simulation International, 2009.

[30] Andrij Usov, Césaire Beyel, Erich Rome, Uwe Beyer, Elisa Castorini, Paolo Palazzari, and Alberto Tofani. The DIESIS Approach to Semantically Interoperable Federated Critical Infrastructure Simulation. In *Advances in System Simulation (SIMUL), 2010 Second International Conference on*, pages 121–128. IEEE, 2010.

[31] S.M. Rinaldi. Modeling and Simulating Critical Infrastructures and their Interdependencies. In *System sciences, 2004. Proceedings of the 37th annual Hawaii international conference on*, pages 8–pp. IEEE, 2004.

[32] Min Ouyang. Review on Modeling and Simulation of Interdependent Critical I¡nfrastructure Systems. *Reliability engineering & System safety*, 121:43–60, 2014.

[33] Paolo Trucco, Enrico Cagno, and Massimiliano De Ambroggi. Dynamic Functional Modelling of Vulnerability and Interoperability of Critical Infrastructures. *Reliability Engineering & System Safety*, 105:51–63, 2012.

[34] J. Bermejo Muñoz, S. G. Galán, L. R. López, R. P. Prado, J. E. Muñoz, T. Grimstad, and D. R. López. Interoperability in Large Scale Cyber-Physical Systems. In *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies Factory Automation (ETFA 2012)*, pages 1–6, Sept 2012.

[35] C. Alcaraz and S. Zeadally. Critical Control System Protection in the 21st Century. *Computer*, 46(10):74–83, 2013.

[36] European Commission. *COM(2004) 702 - Critical Infrastructure Protection in the Fight Against Terrorism*. Publications Office, 2004.

[37] Congress of the United States of America. Public Law 107 - 56 - Uniting and Strengthening America by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism (USA Patriot Act) Act of 2001. USA PATRIOT ACT, October 2001. Washington D.C.

[38] Homeland Security News Wire. Black Hat Event Highlights Vulnerability of U.S. Critical Infrastructure. Online News, July 2013. Last Accessed May 2014.

[39] Computer News. Chinese Hacking Team Caught Taking Over Decoy Water Plant. Online News, August 2013. Last Accessed May 2014.

[40] Javier Lopez, Roberto Setola, and Stephen Wolthusen. *Critical Infrastructure Protection: Advances in Critical Infrastructure Protection: Information Infrastructure Models, Analysis, and Defenses*, volume 7130. Springer, 2012.

[41] ENISA. Analysis of Annual Incident Reports 2012. *Annual Incident Reports*, 13:1–30, 2012.

[42] US DHS ICS-CERT. Incident Response Summary Report, 2011.

[43] US DHS ICS-CERT. ICS-Monitor – Malware Infections in the Control Environment, 12 2012.

[44] US DHS ICS-CERT. ICS-Monitor – Brute Force Attacks on Internet-Facing Control Systems, 2013.

[45] European Comission. Directive 2009/140/EC of the European Parliament and of the Council. L337/37, November 2009. Last Accessed May 2014.

[46] Béla Genge, István Kiss, and Piroska Haller. A System Dynamics Approach for Assessing the Impact of Cyber Attacks on Critical Infrastructures. *International Journal of Critical Infrastructure Protection*, 2015.

[47] M. Thompson. Mariposa Botnet Analysis. Technical report, Technical report, Defence Intelligence, 2009.

[48] Aleksandr Matrosov, Eugene Rodionov, David Harley, and Juraj Malcho. Stuxnet Under the Microscope. ESET LLC, 2010.

[49] McAfee. Global Energy Cyberattacks: "Night Dragon". Technical report, McAfee Foundstone Professional Services and McAfee Labs, February 2011.

[50] E. Chien and G. O'Gorman. The Nitro Attacks, Stealing Secrets from the Chemical Industry. Symantec Security Response, 2011.

[51] Kaspersky Lab Expert. Duqu: Steal Everything, 2011. Last accessed, April 2014.

[52] K. Munro. Deconstructing Flame: the Limitations of Traditional Defences. *Computer Fraud & Security*, 2012(10):8–11, 2012.

[53] Kaspersky Lab Expert. Gauss: Abnormal Distribution, 2012. Last accessed, May 2016.

[54] US DHS ICS-CERT. ICS-Monitor – Incident Response Activity. National Cybersecurity and Communications Integration Center, April 2014. Last accessed, May 2014.

[55] US DHS ICS-CERT. ICS-CERT Year in Review, 2016.

[56] Robert M Lee, Michael J Assante, and Tim Conway. Analysis of the Cyber Attack on the Ukrainian Power Grid. *SANS Industrial Control Systems*, 2016.

[57] ENISA. ENISA Threat Landscape Report 2016. *Annual Incident Reports*, 2016.

[58] Kaspersky Lab Expert. WANNACRY On Industrial Networks: Error Correction. Kaspersky Reports, 2017. Last accessed, June 2017.

[59] Jakub Kroustek. Avast reports on WanaCrypt0r 2.0 Ransomware that Infected NHS and Telefonica. Avast Blog, May 2017. Last accesed, May 2017.

[60] Markus Jakobsson, Susanne Wetzel, and Bülent Yener. Stealth Attacks on Ad-hoc Wireless Networks. In *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, volume 3, pages 2103–2111. IEEE, 2003.

[61] Igor Nai Fovino, Andrea Carcano, Marcelo Masera, and Alberto Trombetta. *Design and Implementation of a Secure Modbus Protocol*, pages 83–96. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[62] B. Miller and D. Rowe. A Survey SCADA of and Critical Infrastructure Incidents. In *Proceedings of the 1st Annual conference on Research in information technology*, pages 51–56. ACM, 2012.

[63] Bonnie Zhu, Anthony Joseph, and Shankar Sastry. A Taxonomy of Cyber Attacks on SCADA Systems. In *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing*, pages 380–388. IEEE, 2011.

[64] C Myers, Sarah Powers, and Daniel Faissol. Taxonomies of Cyber Adversaries and Attacks: a Survey of Incidents and Approaches. *Lawrence Livermore National Laboratory (April 2009)*, 7:1–22, 2009.

[65] Howard F Lipson. Tracking and Tracing Cyber-Attacks: Technical Challenges and Global Policy Issues. Technical report, DTIC Document, 2002.

[66] ENISA. Existing Taxonomies, 2005-2013. Last Access on August 2013.

[67] F. Skopik and Z. Ma. Attack Vectors to Metering Data in Smart Grids under Security Constraints. In *Computer Software and Applications Conference Workshops (COMPSACW), 2012 IEEE 36th Annual*, pages 134–139. IEEE, 2012.

[68] T. Tsao, R. Alexander, M. Dohler, V. Daza, and A. Lozano. Routing Over Low Power and Lossy Networks, January 2012. Last Access on August 2013.

[69] E. Rescorla and B. Korver. Guidelines for Writing RFC Text on Security Considerations. *IETF, RFC-3552*, 1:1–44, 2003.

[70] Vera Marinova-Boncheva. A Short Survey of Intrusion Detection Systems. *Problems of Engineering Cybernetics and Robotics*, 58:23–30, 2007.

[71] G Joy Persial, M Prabhu, and R Shanmugalakshmi. Side channel Attack-Survey. *Int J Adva Sci Res Rev*, 1(4):54–57, 2011.

[72] Jingfei Kong, Onur Aciiçmez, J-P Seifert, and Huiyang Zhou. Hardware-Software Integrated Approaches to Defend against Software Cache-based Side Channel Attacks. In *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*, pages 393–404. IEEE, 2009.

[73] Nishi Tomar and Manoj Singh Gaur. Information Theft through Covert Channel by Exploiting HTTP Post Method. In *Wireless and Optical Communications Networks (WOCN), 2013 Tenth International Conference on*, pages 1–5. IEEE, 2013.

[74] A Hintz. Covert channels in TCP and IP headers. *Presentation at DEFCON*, 10, 2002.

[75] Markus Jakobsson, XiaoFeng Wang, and Susanne Wetzel. Stealth Attacks in Vehicular Technologies. In *IEEE 60th Vehicular Technology Conference*, volume 2, pages 1218–1222. IEEE, 2004.

[76] Gordon Fyodor Lyon. *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning.* Insecure, 2009.

[77] Chunmei Yin, Mingchu Li, Jianbo Ma, and Jizhou Sun. Honeypot and Scan Detection in Intrusion Detection System. In *Electrical and Computer Engineering, 2004. Canadian Conference on*, volume 2, pages 1107–1110. IEEE, 2004.

[78] IBM. IBM X-Force Trend and Risk Report, November 2013.

[79] Uriel Maimon, Alon Kantor, and Oded Dov. Scan Detection, January 3 2005. US Patent App. 11/025,983.

[80] Alberto Dainotti, Alistair King, Ferdinando Papale, Antonio Pescape, et al. Analysis of a/0 Stealth Scan from a Botnet. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, pages 1–14. ACM, 2012.

[81] J. Fan, X. Guo, E. DeMulder, P. Schaumont, B. Preneel, and I. Verbauwhede. State-of-the-Art of Secure ECC Implementations: A Survey on Known Side-Channel Attacks and Countermeasures. In *IEEE International Symposium on Hardware-Oriented Security and Trust*, pages 76–87, 2010.

[82] Muhammad Mahmudul Islam, Ronald Pose, and Carlo Kopp. Suburban Ad-hoc Networks in Information Warfare. In *Proc. 6th Australian InfoWar Conference, Geelong, Australia*, 2005.

[83] Jude Angelo Ambrose, Roshan G Ragel, and Sri Parameswaran. RIJID: Random Code Injection to Mask Power Analysis Based Side Channel Attacks. In *Proceedings of the 44th annual Design Automation Conference*, pages 489–492. ACM, 2007.

[84] Dieter Gollmann. Securing Web Applications. *Information Security Technical Report*, 13(1):1–9, 2008.

[85] Alfio Grasso and Peter H Cole. Definition of Terms Used by the Auto-ID Labs in the Anti-Counterfeiting White Paper Series. *Auto-ID Labs University of Adelaide, White Paper*, 2006.

[86] Z. Su and G. Wassermann. The Essence of Command Injection Attacks in Web Applications. In *ACM SIGPLAN Notices*, volume 41, pages 372–382. ACM, 2006.

[87] L. Khin Shar and H.B. Kuan Tan. Defending Against Cross-Site Scripting Attacks. *Computer*, 45(3):55–62, 2012.

[88] Matthew Van Gundy and Hao Chen. Noncespaces: Using Randomization to Enforce Information Flow Tracking and Thwart Cross-Site Scripting Attacks. In *NDSS*, 2009.

[89] A. Barth, C. Jackson, and J.C. Mitchell. Robust Defenses for Cross-Site Request Forgery. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 75–88. ACM, 2008.

[90] Trevor Jim, Nikhil Swamy, and Michael Hicks. Defeating Script Injection Attacks with Browser-Enforced Embedded Policies. In *Proceedings of the 16th international conference on World Wide Web*, pages 601–610. ACM, 2007.

[91] OWASP. The Ten Most Critical Web Application Security Risks, October 2010.

[92] Symantec. TCP MODBUS - Unauthorized Read Request. Last accessed, April 2014.

[93] National Vulnerability Database. Vulnerability Summary for CVE-2013-0663. NIST, April 2013.

[94] US DHS ICS-CERT. ICSA-13-077-01A Schneider Electric PLCs Vulnerabilities, June 2013. Last accessed, April 2014.

[95] Sylvia Osborn, Ravi Sandhu, and Qamar Munawer. Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies. *ACM Transactions on Information and System Security (TISSEC)*, 3(2):85–106, 2000.

[96] Kan Yang and Xiaohua Jia. ABAC: Attribute-Based Access Control. In *Security for cloud storage systems*, pages 39–58. Springer, 2014.

[97] RE Mackiewicz. Overview of IEC 61850 and Benefits. In *2006 IEEE PES Power Systems Conference and Exposition*, pages 623–630. IEEE, 2006.

[98] Jean-Jaques Quisquater and Franois Koene. Side Channel Attacks: State of the Art. *project CRYPTREC*, 2002.

[99] J. Reeves, A. Ramaswamy, M. Locasto, S. Bratus, and S. Smith. Intrusion Detection for Resource-Constrained Embedded Control Systems in the Power Grid. *International Journal of Critical Infrastructure Protection*, 5(2):74–83, 2012.

[100] R. Berthier and W.H. Sanders. Specification-Based Intrusion Detection for Advanced Metering Infrastructures. In *2011 IEEE 17th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 184–193. IEEE, 2011.

[101] Jing Deng, Richard Han, and Shivakant Mishra. Countermeasures against Traffic Analysis Attacks in Wireless Sensor Networks. In *Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005. First International Conference on*, pages 113–126. IEEE, 2005.

[102] Martin Roesch et al. Snort-lightweight Intrusion Detection for Networks. In *Proceedings of the 13th USENIX conference on System administration*, pages 229–238. Seattle, Washington, 1999.

[103] M Blum, LA Von Ahn, J Langford, and N Hopper. The CAPTCHA Project (Completely Automatic Public Turing Test to tell Computers and Humans Apart). *School of Computer Science, Carnegie-Mellon University, http://www. captcha. net*, 2000.

[104] P. Ratanaworabhan, V.B. Livshits, and B.G. Zorn. NOZZLE: A Defense Against Heap-spraying Code Injection Attacks. In *USENIX Security Symposium*, pages 169–186, 2009.

[105] Sandro Bologna and Roberto Setola. The Need to Improve Local Self-Awareness in CIP/CIIP. In *First IEEE International Workshop on Critical Infrastructure Protection*, pages 84–89. IEEE Computer Society, 2005.

[106] Stefano Avallone, Claudio Mazzariello, Francesco Oliviero, and Simon Pietro Romano. Protecting Critical Infrastructures from Stealth Attacks: A Closed-Loop Approach Involving Detection and Remediation. In *Critical Information Infrastructure Security*, pages 209–212. Springer, 2013.

[107] S. D'Antonio, F. Oliviero, and R. Setola. High-Speed Intrusion Detection in Support of Critical Infrastructure Protection. *Critical Information Infrastructures Security*, pages 222–234, 2006.

[108] Fabio Pasqualetti, Florian Dorfler, and Francesco Bullo. Attack Detection and Identification in Cyber-Physical Systems. *IEEE Transactions on Automatic Control*, 58(11):2715–2729, 2013.

[109] K. Scarfone and P. Mell. Guide to Intrusion Detection and Prevention Systems (IDPS). *NIST Special Publication*, 800:94, 2007.

[110] Smart Grid Interoperability Panel Cyber Security Working Group and others. NISTIR 7628-Guidelines for Smart Grid Cyber Security vol. 1-3, 2010.

[111] Terry Fleury, Himanshu Khurana, and Von Welch. Towards A Taxonomy Of Attacks Against Energy Control Systems. *Critical Infrastructure Protection II*, 290:71–85, 2009.

[112] Keith Stouffer, Joe Falco, and Karen Scarfone. NIST SP800-82 - Guide to Industrial Control Systems (ICS) Security. *NIST Special Publication*, Revision 2, 5 2015.

[113] North American Electric Reliability Corporation. NERC Critical Infrastructure Protection (CIP) Cyber Security Standards - CIP 002-009 Series. NERC, 2011.

[114] SIG Bluetooth. Bluetooth Specification, 2007.

[115] Stig Petersen and Simon Carlsen. WirelessHART versus ISA100. 11a: The Format War Hits the Factory Floor. *IEEE Industrial Electronics Magazine*, 5:23–34, 2011.

[116] Donald G Firesmith. Common Concepts Underlying Safety Security and Survivability Engineering. Technical report, DTIC Document, 2003.

[117] B. Zhu and S. Sastry. SCADA-Specific Intrusion Detection/Prevention Systems: A Survey and Taxonomy. In *Proceedings of the 1st Workshop on Secure Control Systems (SCS)*, 2010.

[118] Andrea Nicholson, S Webber, S Dyer, T Patel, and Helge Janicke. SCADA Security in the Light of Cyber-Warfare. *Computers & Security*, 31(4):418–436, 2012.

[119] Lawrence Chung, BA Nixon, E Yu, and J Mylopoulos. Non-Functional Requirements in Software Engineering. 2000.

[120] Collins Concise English Dictionary. © HarperCollins Publishers, 2013.

[121] André B Bondi. Characteristics of Scalability and their Impact on Performance. In *Proceedings of the 2nd international workshop on Software and performance*, pages 195–203. ACM, 2000.

[122] John F Meyer and William H Sanders. Specification and Construction of Performability Models. In *2nd Intr. Workshop on Performability Modeling of Computer and Communication Systems*, pages 28–30, 1993.

[123] Anne Geraci, Freny Katki, Louise McMonegal, Bennett Meyer, John Lane, Paul Wilson, Jane Radatz, Mary Yee, Hugh Porteous, and Fredrick Springsteel. IEEE Standard Computer Dictionary: Compilation of IEEE Standard Computer Glossaries. 1991.

[124] Martin Weik. *Computer Science and Communications Dictionary.* Kluwer Academic Pub, 2000.

[125] Gregory J Suski, editor. *Distributed Computer Control Systems*, 5 1985.

[126] ETSI Group Specification. ETSI GS NFV-INF 001. Network Functions Virtualisation (NFV); Infrastructure Overview. ETSI, 1 2015. V1.1.1.

[127] Wayne Jansen and Timothy Grance. NIST SP800-144 - Guidelines on Security and Privacy in Public Cloud Computing. *NIST Special Publication*, 12 2011.

[128] David Nuñez, Carmen Fernandez-Gago, Siani Pearson, and Massimo Felici. A Metamodel for Measuring Accountability Attributes in the Cloud. In *IEEE 5th International Conference on Cloud Computing Technology and Science*, 2013.

[129] Rini Van Solingen, Vic Basili, Gianluigi Caldiera, and H Dieter Rombach. Goal Question Metric (GQM) Approach. *Encyclopedia of Software Engineering*, 2002.

[130] Daniele Catteddu, Massimo Felici, Giles Hogben, Amy Holcroft, Eleni Kosta, Ronald Leenes, Christopher Millard, Maartje Niezen, David Nuñez, Nick Papanikolaou, et al. Towards a Model of Accountability for Cloud Computing Services. In *Intr. Workshop on Trustworthiness, Accountability and Forensics in the Cloud*, 2013.

[131] National Institute of Standards and Technology (NIST). Framework for Improving Critical Infrastructure Cybersecurity, February 2014.

[132] David Nunez, Isaac Agudo, and Javier Lopez. A Parametric Family of Attack Models for Proxy Re-Encryption. In *2015 IEEE 28th Computer Security Foundations Symposium*, pages 290–301. IEEE, 2015.

[133] Eric Byres, John Karsch, and Joel Carter. NISCC Good Practice Guide on Firewall Deployment for SCADA and Process Control Networks. National Infrastructure Security Co-Ordination Centre, 2 2005.

[134] International Electrotechnical Commission and others. IEC/TS 62351: Power Systems Management and Associated Information Exchange, 2011.

[135] ISO/IEC 27002:2013 - Information technology – Security techniques – Code of practice for information security controls, 2013.

[136] Council on CyberSecurity (CCS) Top 20 Critical Security Controls (CSC).

[137] ISACA for Information Technology. Control Objectives for Information and Related Technology (COBIT), 2012.

[138] ANSI/ISA-62443-2-1 (99.02.01)-2009, Security for Industrial Automation and Control Systems: Establishing an Industrial Automation and Control Systems Security Program, 2009.

[139] ANSI/ISA-62443-3-3 (99.03.03)-2013, Security for Industrial Automation and Control Systems: System Security Requirements and Security Levels, 2013.

[140] ISO/IEC 27001:2013 - Information technology — Security techniques — Information security management systems — Requirements, 2013.

[141] NIST SP800-53 - Security and Privacy Controls for Federal Information Systems and Organizations, 4 2013.

[142] International Electrotechnical Commission and others. IEC/TS 62351-7 Power Systems Management and Associated Information Exchange - Data and Communications Security - Part 7: Network and System Management (NSM) Data Object Models, September 2007.

[143] International Electrotechnical Commission and others. IEC/TS 62351-3 Power Systems Management and Associated Information Exchange - Data and Communications Security - Part 3: Communication network and system security – Profiles including TCP/IP, 10 2014.

[144] International Electrotechnical Commission and others. IEC/TS 62351-4 Power Systems Management and Associated Information Exchange - Data and Communications Security - Part 4: Profiles including MMS, 6 2007.

[145] International Electrotechnical Commission and others. IEC/TS 62351-5 Power Systems Management and Associated Information Exchange - Data and Communications Security - Part 5: Security for IEC 60870-5 and Derivatives, 4 2013.

[146] International Electrotechnical Commission and others. IEC/TS 62351-6 Power Systems Management and Associated Information Exchange - Data and Communications Security - Part 6: Security for IEC 61850, 6 2007.

[147] Todd A DeLong, D Todd Smith, and Barry W Johnson. Dependability Metrics to Assess Safety-Critical Systems. *IEEE Transanctions on Reliability*, 54(3):498–505, 2005.

[148] IEC 61508 -3 - Functional Safety of Electrical / Electronic / Programmable Electronic Safety-Related Systems, 1998.

[149] Harold A Linstone, Murray Turoff, et al. *The Delphi Method: Techniques and Applications*, volume 29. Addison-Wesley Reading, MA, 1975.

[150] M. Dunn Cavelty and M. Suter. The art of CIIP strategy: Tacking Stock of Content and Processes. *Critical Infrastructure Protection*, 7130:15–38, 2012.

[151] Z. Yu, J.J.P. Tsai, and T. Weigert. An Adaptive Automatically Tuning Intrusion Detection System. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 3(3):10, 2008.

[152] H. Lin, A. Slagell, C. Di Martino, Z. Kalbarczyk, and R.K. Iyer. Adapting Bro into SCADA: Building a Specification-based Intrusion Detection System for the DNP3 Protocol. In *Proceedings of Cyber Security and Information Intelligence Research Workshop*, page 5. ACM, 2013.

[153] K. McLaughlin. Intrusion Detection for SCADA Systems - the Smart Grid Protection Against Cyber Attacks. US CERT, 2015.

[154] Alan Gillies. *Software Quality: Theory and Management*. Lulu. com, 2011.

[155] International Organization for Standardization. ISO 5725:1994 Series - Accuracy (Trueness and Precision) of Measurement Methods and Results, 1994.

[156] International Organization for Standardization. ISO 11843:1997 Series - Capability of Detection, 1997.

[157] ISO TC69. ISO 2854:1976 - Statistical Interpretation of Data–Techniques of Estimation and Tests Relating to Means and Variances.

[158] Wayne Jansen. NISTIR 7564. Directions in Security Metrics Research. *NIST. gov-Computer Security Division-Computer Security Resource Center*, April 2009.

[159] NIST SP800-55 - Performance Measurement Guide for Information Security, 2008.

[160] IEC BIPM, ILAC IFCC, IUPAP IUPAC, and OIML ISO. *International Vocabulary of Metrology, Basic and General Concepts and Associated Terms*. JCGM, 2008.

[161] Pierre Baldi, Søren Brunak, Yves Chauvin, Claus AF Andersen, and Henrik Nielsen. Assessing the Accuracy of Prediction Algorithms for Classification: An Overview. *Bioinformatics*, 16(5):412–424, 2000.

[162] Oscar R González, Jorge R Chávez-Fuentes, and W Steven Gray. Towards a Metric for the Assessment of Safety Critical Control Systems. In *Proc. 2008 AIAA Guidance, Navigation and Control Conference*, 2008.

[163] Vicky Papadopoulou and Andreas Gregoriades. Nonfunctional Requirements Validation Using Nash Equilibria. *SCIYO. COM*, page 41, 2010.

[164] A Ananda Rao and Merugu Gopichand. Four Layered Approach to Non-Functional Requirements Analysis. *arXiv preprint arXiv:1201.6141*, 2012.

[165] ISO 27004 - Information Technology - Security Techniques - Information Security Management - Measurement, 2009.

[166] Niv Goldenberg and Avishai Wool. Accurate Modeling of Modbus/TCP for Intrusion Detection in SCADA Systems. *International Journal of Critical Infrastructure Protection*, 6(2):63–75, 2013.

[167] Hadeli Hadeli, Ragnar Schierholz, Markus Braendle, and Cristian Tuduce. Leveraging Determinism in Industrial Control Systems for Advanced Anomaly Detection and Reliable Security Configuration. In *IEEE Conference on Emerging Technologies & Factory Automation*, pages 1–8, 2009.

[168] Nor Badrul Anuar, Maria Papadaki, Steven Furnell, and Nathan Clarke. An Investigation and Survey of Response Options for Intrusion Response Systems (IRSs). In *Information Security for South Africa*, pages 1–8. IEEE, 2010.

[169] Lorena Cazorla, Cristina Alcaraz, and Javier Lopez. Towards Automatic Critical Infrastructure Protection through Machine Learning. In *Critical Information Infrastructures Security*, pages 197–203. Springer, 2013.

[170] M. Xie, S. Han, B. Tian, and S. Parvin. Anomaly Detection in Wireless Sensor Networks: A Survey. *Journal of Network and Computer Applications*, 34(4):1302–1325, 2011.

[171] P. Düssel, C. Gehl, P. Laskov, J.U. Bußer, C. Störmann, and J. Kästner. Cyber-Critical Infrastructure Protection using Real-Time Payload-Based Anomaly Detection. *Critical Information Infrastructures Security*, pages 85–97, 2010.

[172] D. Yang, A. Usynin, and J.W. Hines. Anomaly-based Intrusion Detection for SCADA Systems. In *5th Intl. Topical Meeting on Nuclear Plant Instrumentation, Control and Human Machine Interface Technologies*, pages 12–16, 2006.

[173] A. Carcano, I.Fovino, M. Masera, and A. Trombetta. State-Based Network Intrusion Detection Systems for SCADA Protocols: a Proof of Concept. *Critical Information Infrastructures Security*, pages 138–150, 2010.

[174] I.H. Witten, E. Frank, and M.A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. M. Kaufmann, 2011.

[175] K. Burbeck and S. Nadjm-Tehrani. Adaptive Real-Time Anomaly Detection with Incremental Clustering. *information security technical report*, 12(1):56–67, 2007.

[176] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly Detection: A Survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[177] SB. Kotsiantis, ID. Zaharakis, and PE. Pintelas. Supervised Machine Learning: A Review of Classification Techniques. *Frontiers in Artificial Intelligence and Applications*, 160:3, 2007.

[178] Manasi Gyanchandani, JL Rana, and RN Yadav. Taxonomy of Anomaly Based Intrusion Detection System: a Review. *International Journal of Scientific and Research Publications*, 2(12):1, 2012.

[179] Monowar H Bhuyan, Dhruba Kumar Bhattacharyya, and Jugal Kumar Kalita. Network Anomaly Detection: Methods, Systems and Tools. *Communications Surveys & Tutorials, IEEE*, 16(1):303–336, 2014.

[180] FAN Jing-qing and Yao Qi-wei. Nonlinear Time Series: Nonparametric and Parametric Methods, 2006.

[181] SAP. Demand Planning, Exponential Smoothing (SCM-APO-FCS). Last Accessed May 2014.

[182] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian Network Classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[183] V Jyothsna, VV Rama Prasad, and K Munivara Prasad. A Review of Anomaly Based Intrusion Detection Systems. *International Journal of Computer Applications*, 28(7):26–35, 2011.

[184] Bharanidharan Shanmugam and Norbik Bashah Idris. *Hybrid Intrusion Detection Systems (HIDS) Using Fuzzy Logic*. INTECH Open Access Publisher, 2011.

[185] David JC MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge university press, 2003.

[186] R. Sadoddin and A. Ghorbani. A Comparative Study of Unsupervised Machine Learning and Data Mining Techniques for Intrusion Detection. In *Machine Learning and Data Mining in Pattern Recognition*, pages 404–418. Springer, 2007.

[187] T. Roosta, D.K. Nilsson, U. Lindqvist, and A. Valdes. An Intrusion Detection System for Wireless Process Control Systems. In *Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference on*, pages 866–872. IEEE, 2008.

[188] I. Fovino, A. Coletta, A. Carcano, and M. Masera. Critical State-Based Filtering System for Securing SCADA Network Protocols. *IEEE Transactions on Industrial Electronics*, 59(10):3943–3950, 2012.

[189] D. Hadziosmanovic, D. Bolzoni, P. Hartel, and S. Etalle. MELISSA: Towards Automated Detection of Undesirable User Actions in Critical Infrastructures. In *Seventh European Conference on Computer Network Defense (EC2ND)*, pages 41–48. IEEE, IEEE Computer Society, 2011.

[190] S. Cheung, B. Dutertre, M. Fong, U. Lindqvist, K. Skinner, and A. Valdes. Using Model-based Intrusion Detection for SCADA Networks. In *Proceedings of the SCADA Security Scientific Symposium*, pages 127–134, 2007.

[191] Massimiliano Raciti and Simin Nadjm-Tehrani. Embedded Cyber-Physical Anomaly Detection in Smart Meters. In *Critical Information Infrastructures Security*, pages 34–45. Springer, 2013.

[192] Alireza Shameli-Sendi, Mohamed Cheriet, and Abdelwahab Hamou-Lhadj. Taxonomy of Intrusion Risk Assessment and Response System. *Computers & Security*, 45:1–16, 2014.

[193] A. A. Cárdenas, S. Amin, Z. S. Lin, Y. L. Huang, C. Y. Huang, and S. Sastry. Attacks Against Process Control Systems: Risk Assessment, Detection, and Response. In *Proceedings of Symposium on Information, Computer and Communications Security*, pages 355–366. ACM, 2011.

[194] Gary Stoneburner, Alice Goguen, and Alexis Feringa. Risk Management Guide for Information Technology Systems. *NIST Special Publication*, 800:30, 2002.

[195] International Organization for Standardization. ISO/IEC 27005 - Information Security Risk Management. ISO/IEC, 2008.

[196] Ivan Balepin, Sergei Maltsev, Jeff Rowe, and Karl Levitt. Using Specification-Based Intrusion Detection for Automated Responses. In *Recent Advances in Intrusion Detection*, pages 136–154. Springer, 2003.

[197] Marko Jahnke, Christian Thul, and Peter Martini. Graph Based Metrics for Intrusion Response Measures in Computer Networks. In *Proceedings of Conference on Local Computer Networks*, pages 1035–1042. IEEE, 2007.

[198] Kjetil Haslum, Ajith Abraham, and Svein Knapskog. DIPS: A Framework for Distributed Intrusion Prediction and Prevention Using Hidden Markov Models and Online Fuzzy Risk Assessment. In *International Symposium on Information Assurance and Security*, pages 183–190. IEEE, 2007.

[199] Alireza Shameli-Sendi and Michel Dagenais. ARITO: Cyber-Attack Response System Using Accurate Risk Impact Tolerance. *International Journal of Information Security*, pages 1–24, 2013.

[200] Y. Haimes, J. Santos, K. Crowther, M. Henry, C. Lian, and Z. Yan. Risk Analysis in Interdependent Infrastructures. In *Critical Infrastructure Protection*, pages 297–310. Springer, 2007.

[201] MH Henry and YY Haimes. A New Dynamic Risk Assessment and Management Model for Supervisory Control and Data Acquisition Networks. In *Society of Risk Analysis Annual Meeting*, 2006.

[202] Wael Kanoun, Nora Cuppens-Boulahia, Frédéric Cuppens, and Samuel Dubus. Risk-Aware Framework for Activating and Deactivating Policy-Based Response. In *Proceedings of Conference on Network and System Security*, pages 207–215. IEEE, 2010.

[203] Riadh WY Habash, Voicu Groza, Dan Krewski, and Greg Paoli. A Risk Assessment Framework for the Smart Grid. In *Electrical Power & Energy Conference*, pages 1–6. IEEE, 2013.

[204] J. Carver and A. Curtis. Adaptive Agent-Based Intrusion Response. Technical report, DTIC Document, 2001.

[205] Phillip A Porras and Peter G Neumann. EMERALD: Event Monitoring Enabling Response to Anomalous Live Disturbances. In *Proceedings of National Information Systems Security Conference*, pages 353–365, 1997.

[206] Thomas Toth and Christopher Kruegel. Evaluating the Impact of Automated Intrusion Response Mechanisms. In *Proceedings of Computer Security Applications Conference*, pages 301–310. IEEE, 2002.

[207] Chengpo Mu and Yingjiu Li. An Intrusion Response Decision-Making Model Based on Hierarchical Task Network Planning. *Expert systems with applications*, 37(3):2465–2472, 2010.

[208] Li Feng, Wei Wang, Lina Zhu, and Yi Zhang. Predicting Intrusion Goal Using Dynamic Bayesian Network with Transfer Probability Estimation. *Journal of Network and Computer Applications*, 32(3):721–732, 2009.

[209] Eric Andrew Fisch. *Intrusion Damage Control and Assessment: a Taxonomy and Implementation of Automated Responses to Intrusive Behavior*, volume Doctoral Dissertation. Texas A&M University, 1996.

[210] Gregory B White, Eric A Fisch, and Udo W Pooch. Cooperating Security Managers: A Peer-based Intrusion Detection System. *IEEE Network*, 10(1):20–23, 1996.

[211] Theodore Bowen, Dana Chee, Michael Segal, R Sekar, T Shanbhag, and P Uppuluri. Building Survivable Systems: An Integrated Approach Based on Intrusion Detection and Damage Containment. In *Proceedings of DARPA Information Survivability Conference and Exposition*, volume 2, pages 84–99. IEEE, 2000.

[212] S Mnsman and Pat Flesher. System or Security Managers Adaptive Response Tool. In *Proceedings of DARPA Information Survivability Conference and Exposition*, volume 2, pages 56–68. IEEE, 2000.

[213] Anil Somayaji and Stephanie Forrest. Automated Response Using System-Call Delays. In *Proceedings of the 9th USENIX Security Symposium*, volume 70, 2000.

[214] W. Lee, W. Fan, M. Miller, S.J. Stolfo, and E. Zadok. Toward Cost-Sensitive Modeling for Intrusion Detection and Response. *Journal of Computer Security*, 10(1):5–22, 2002.

[215] S.M. Lewandowski, D.J. Van Hook, G.C. O'Leary, J.W. Haines, and L.M. Rossey. SARA: Survivable Autonomic Response Architecture. In *Proceedings of DARPA Information Survivability Conference*, volume 1, pages 77–88. IEEE, 2001.

[216] D Schnackengerg, Harley Holliday, Randall Smith, Kelly Djahandari, and Dan Sterne. Cooperative Intrusion Traceback and Response Architecture (CITRA). In *Proceedings of DARPA Information Survivability Conference*, volume 1, pages 56–68. IEEE, 2001.

[217] Xinyuan Wang, D Reeves, and S Felix Wu. Tracing Based Active Intrusion Response. *Journal of Information Warfare*, 1(1):50–61, 2001.

[218] Sapon Tanachaiwiwat, Kai Hwang, and Yue Chen. Adaptive Intrusion Response to Minimize Risk Over Multiple Network Attacks. *ACM Trans on Information and System Security*, 19:1–30, 2002.

[219] B. Foo, Y.S. Wu, Y.C. Mao, S. Bagchi, and E. Spafford. ADEPTS: Adaptive Intrusion Response Using Attack Graphs in an e-Commerce Environment. In *Proceedings of Conference on Dependable Systems and Networks*, pages 508–517. IEEE, 2005.

[220] Michael E Locasto, Ke Wang, Angelos D Keromytis, and Salvatore J Stolfo. Flips: Hybrid Adaptive Intrusion Prevention. In *Recent Advances in Intrusion Detection*, pages 82–101. Springer, 2006.

[221] Maria Papadaki and SM Furnell. Achieving Automated Intrusion Response: A Prototype Implementation. *Information Management & Computer Security*, 14(3):235–251, 2006.

[222] Natalia Stakhanova, Samik Basu, and Johnny Wong. A Cost-Sensitive Model for Preemptive Intrusion Response Systems. In *AINA*, volume 7, pages 428–435, 2007.

[223] Chris Strasburg, Natalia Stakhanova, Samik Basu, and Johnny S Wong. A Framework for Cost Sensitive Assessment of Intrusion Response Selection. In *Conference of Computer Software and Applications*, volume 1, pages 355–360. IEEE, 2009.

[224] Nizar Kheir, Nora Cuppens-Boulahia, Frédéric Cuppens, and Hervé Debar. A Service Dependency Model for Cost-Sensitive Intrusion Response. In *ESORICS*, pages 626–642. Springer, 2010.

[225] Shuzhen Wang, Zonghua Zhang, and Youki Kadobayashi. Exploring Attack Graph for Cost-Benefit Security Hardening: A Probabilistic Approach. *Computers & Security*, 32:158–169, 2013.

[226] Montaceur Zaghdoud and Mohammed Saeed Al-Kahtani. Contextual Fuzzy Cognitive Map for Intrusion Response System. *Computer*, 2(3):471–478, 2013.

[227] Saman A Zonouz, Himanshu Khurana, William H Sanders, and Timothy M Yardley. RRE: A Game-Theoretic Intrusion Response and Recovery Engine. *IEEE Transactions on Parallel and Distributed Systems*, 25(2):395–406, 2014.

[228] Boutheina A Fessi, Salah Benabdallah, Noureddine Boudriga, and M Hamdi. A Multi-Attribute Decision Model for Intrusion Response System. *Information Sciences*, 270:237–254, 2014.

[229] Cristina Alcaraz, Carmen Fernandez-Gago, and Javier Lopez. An Early Warning System Based on Reputation for Energy Control Systems. *IEEE Transactions on Smart Grid*, 2(4):827–834, 2011.

[230] Kai Xing, Shyaam Sundhar Rajamadam Srinivasan, Major Jose, Jiang Li, and Xiuzhen Cheng. Attacks and Countermeasures in Sensor Networks: a Survey. In *Network Security*, pages 251–272. Springer, 2010.

[231] Tal G Malkin, François-Xavier Standaert, and Moti Yung. A Comparative Cost/Security Analysis of Fault Attack Countermeasures. In *Fault Diagnosis and Tolerance in Cryptography*, pages 159–172. Springer, 2006.

[232] Dag Arne Osvik, Adi Shamir, and Eran Tromer. Cache Attacks and Countermeasures: the Case of AES. In *Topics in Cryptology*, pages 1–20. Springer, 2006.

[233] Chris Karlof and David Wagner. Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures. *Ad Hoc Networks*, 1(2):293–315, 2003.

[234] Kyle Ingols, Matthew Chu, Richard Lippmann, Seth Webster, and Stephen Boyer. Modeling Modern Network Attacks and Countermeasures Using Attack Graphs. In *Computer Security Applications Conference*, pages 117–126. IEEE, 2009.

[235] Majid Meghdadi, Suat Ozdemir, and Inan Güler. A Survey of Wormhole-based Attacks and their Countermeasures in Wireless Sensor Networks. IETE Technical Review, 2011.

[236] Stephen M Specht and Ruby B Lee. Distributed Denial of Service: Taxonomies of Attacks, Tools, and Countermeasures. In *ISCA PDCS*, pages 543–550, 2004.

[237] Alvaro A Cárdenas, Saurabh Amin, and Shankar Sastry. Research Challenges for the Security of Control Systems. In *HotSec*, 2008.

[238] Xiang Lu, Wenye Wang, Jianfeng Ma, and Limin Sun. Domino of the Smart Grid: An Empirical Study of System Behaviors in the Interdependent Network Architecture. In *IEEE International Conference on Smart Grid Communications*, pages 612–617, 2013.

[239] Josef Allen, Sereyvathana Ty, Xiuwen Liu, and Ivan Lozano. Preventing Cascading Event: a Distributed Cyber-Physical Approach. In *Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research*, page 54. ACM, 2011.

[240] Leonardo Dueñas-Osorio and Srivishnu Mohan Vemuru. Cascading Failures in Complex Infrastructure Systems. *Structural safety*, 31(2):157–167, 2009.

[241] Michael Treaster. A Survey of Fault-Tolerance and Fault-Recovery Techniques in Parallel Systems. *ACM Computing Research Repository*, 501002:1–11, 2005.

[242] Sanjay Bansal, Sanjeev Sharma, and Ishita Trivedi. A Detailed Review of Fault-Tolerance Techniques in Distributed System. *International Journal on Internet and Distributed Computing Systems*, 1(1):33–39, 2011.

[243] Ruchika Mehresh. *Schemes for Surviving Advanced Persistent Threats*. PhD thesis, Faculty of the Graduate School of the University at Buffalo, State University of New York, 2013.

[244] Giuliana Santos Veronese, Miguel Correia, Alysson Neves Bessani, and Lau Cheuk Lung. Highly-Resilient Services for Critical Infrastructures. In *Proceedings of the Embedded Systems and Communications Security Workshop*, 2009.

[245] Ye Yan, Yi Qian, Hamid Sharif, and David Tipper. A Survey on Smart Grid Communication Infrastructures: Motivations, Requirements and Challenges. *Communications Surveys & Tutorials, IEEE*, 15(1):5–20, 2013.

[246] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a Better Understanding of Context and Context-Awareness. In *Handheld and ubiquitous computing*, pages 304–307. Springer, 1999.

[247] Mo-Yuen Chow, Sue O Yee, and Leroy S Taylor. Recognizing Animal-Caused Faults in Power Distribution Systems Using Artificial Neural Networks. *Power Delivery, IEEE Transactions on*, 8(3):1268–1274, 1993.

[248] 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. *Energies*, 5(10):4091–4109, 2012.

[249] Shubhalaxmi Kher, Victor Nutt, Dipankar Dasgupta, Hasan Ali, and Paul Mixon. A Detection Model for Anomalies in Smart Grid with Sensor Network. In *Future of Instrumentation International Workshop (FIIW), 2012*, pages 1–4. IEEE, 2012.

[250] Paria Jokar. Model-Based Intrusion Detection for Home Area Networks in Smart Grids. *University of Bristol, Bristol*, pages 1–19, 2012.

[251] Waleed KA Najy, HH Zeineldin, Ali H Alaboudy, and Wei Lee Woon. A Bayesian Passive Islanding Detection Method for Inverter-Based Distributed Generation Using ESPRIT. *Power Delivery, IEEE Transactions on*, 26(4):2687–2696, 2011.

[252] Nauman Shahid, Saad Abdul Aleem, Ijaz Haider Naqvi, and Nauman Zaffar. Support Vector Machine Based Fault Detection & Classification in Smart Grids. In *Globecom Workshops (GC Wkshps), 2012 IEEE*, pages 1526–1531. IEEE, 2012.

[253] Yichi Zhang, Lingfeng Wang, Weiqing Sun, Robert C Green, Mansoor Alam, et al. Distributed Intrusion Detection System in a Multi-Layer Network Architecture of Smart Grids. *Smart Grid, IEEE Transactions on*, 2(4):796–808, 2011.

[254] Robert Mitchell and Ray Chen. Behavior-Rule Based Intrusion Detection Systems for Safety Critical Smart Grid Applications. *Smart Grid, IEEE Transactions on*, 4(3):1254–1263, 2013.

[255] Hanie Sedghi and Edmond Jonckheere. Statistical Structure Learning, Towards a Robust Smart Grid. *arXiv preprint arXiv:1403.1863*, 2014.

[256] Shing-Chow Chan, Kai Man Tsui, HC Wu, Yunhe Hou, Yik-Chung Wu, and Felix F Wu. Load/Price Forecasting and Managing Demand Response for Smart Grids: Methodologies and Challenges. *Signal Processing Magazine, IEEE*, 29(5):68–85, 2012.

[257] CS Chang, Zhaoxia Wang, Fan Yang, and WW Tan. Hierarchical Fuzzy Logic System for Implementing Maintenance Schedules of Offshore Power Systems. *Smart Grid, IEEE Transactions on*, 3(1):3–11, 2012.

[258] Yashar Sahraei Manjili, Amir Rajaee, Mohammad Jamshidi, and Brian T Kelley. Fuzzy Control of Electricity Storage Unit for Energy Management of Micro-Grids. In *World Automation Congress (WAC), 2012*, pages 1–6. IEEE, 2012.

[259] Vito Calderaro, Christoforos N Hadjicostis, Antonio Piccolo, and Pierluigi Siano. Failure Identification in Smart Grids Based on Petri Net Modeling. *Industrial Electronics, IEEE Transactions on*, 58(10):4613–4623, 2011.

[260] Engin Karatepe, Takashi Hiyama, et al. Controlling of Artificial Neural Network for Fault Diagnosis of Photovoltaic Array. In *Intelligent System Application to Power Systems (ISAP), 2011 16th International Conference on*, pages 1–6. IEEE, 2011.

[261] Chen-Fu Chien, Shi-Lin Chen, and Yih-Shin Lin. Using Bayesian Network for Fault Location on Distribution Feeder. *Power Delivery, IEEE Transactions on*, 17(3):785–793, 2002.

[262] SR Samantaray, Khalil El-Arroudi, Geza Joós, and Innocent Kamwa. A Fuzzy Rule-Based Approach for Islanding Detection in Distributed Generation. *Power Delivery, IEEE Transactions on*, 25(3):1427–1433, 2010.

[263] Cristina Alcaraz and Stephen Wolthusen. Recovery of Structural Controllability for Control Systems. In *Critical Infrastructure Protection VIII*, pages 47–63. Springer, 2014.

[264] Cristina Alcaraz, Isaac Agudo, David Nuñez, and Javier Lopez. Managing Incidents in Smart Grids à la Cloud. In *IEEE CloudCom 2011*, pages 527–531, Athens, Greece, Nov-Dec 2011 2011. IEEE Computer Society, IEEE Computer Society.

[265] Andy Swales. Open Modbus/TCP Specification. *Schneider Electric*, 29, 1999.

[266] International Electrotechnical Commission et al. Telecontrol Equipment and Systems. Part 5: Transmission Protocols. Section 101: Companion Standards Especially for Basic Telecontrol Tasks. *IEC-60870-5-104*, 2000.

[267] ZigBee Alliance. ZigBee 2007 Specification. *Online: http://www. zigbee. org/Specifications/ZigBee/Overview. aspx*, 45:120, 2007.

[268] ISO/IEC 27000 Series — Information security Management Systems.

[269] NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 2.0, 2 2012.

[270] Peter Mell and Tim Grance. NIST SP800-145 - The NIST Definition of Cloud Computing. *NIST Special Publication*, 2011.

[271] Rodrigo Roman, Javier Lopez, and Masahiro Mambo. Mobile Edge Computing, Fog et al.: A Survey and Analysis of Security Threats and Challenges. *arXiv preprint arXiv:1602.00484*, 2016.

[272] 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. *IEEE Communications Surveys & Tutorials*, 16(3):1617–1634, 2014.

[273] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.

[274] Zhaogang Shu, Jiafu Wan, Di Li, Jiaxiang Lin, Athanasios V Vasilakos, and Muhammad Imran. Security in Software-Defined Networking: Threats and Countermeasures. *Mobile Networks and Applications*, 21(5):764–776, 2016.

[275] Giuseppe Bernieri, Federica Pascucci, and Javier Lopez. Network Anomaly Detection in Critical Infrastructure Based on Mininet Network Simulator. 2017.

[276] Kurt Zeilenga. Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map. 2006.

[277] IDA Modbus. Modbus Application Protocol Specification v1.1a. *North Grafton, Massachusetts (www. modbus. org/specs. php)*, 2004.

[278] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela H Byers. Big Data: The Next Frontier for Innovation, Competition, and Productivity. 2011.

[279] Dave Cooper. Internet X. 509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. 2008.

[280] IEC 62541 - OPC Unified Architecture, 2015.

[281] ISO/IEC 20922:2016 - Information technology – Message Queuing Telemetry Transport (MQTT) v3.1.1, 2016.

[282] David Nuñez, Isaac Agudo, and Javier Lopez. NTRUReEncrypt: An efficient Proxy Reencryption Scheme Based on NTRU. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pages 179–189. ACM, 2015.

[283] Ahmed Patel, Mona Taghavi, Kaveh Bakhtiyari, and Joaquim Celestino JúNior. An Intrusion Detection and Prevention System in Cloud Computing: A Systematic Review. *Journal of network and computer applications*, 36(1):25–41, 2013.

[284] Himanshu Khurana, Rakesh Bobba, Tim Yardley, Pooja Agarwal, and Erich Heine. Design Principles for Power Grid Cyber-Infrastructure Authentication Protocols. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10. IEEE, 2010.

[285] Mostafa M Fouda, Zubair Md Fadlullah, Nei Kato, Rongxing Lu, and Xuemin Sherman Shen. A Lightweight Message Authentication Scheme for Smart Grid Communications. *IEEE Transactions on Smart Grid*, 2(4):675–685, 2011.

[286] Frances Cleveland. Enhancing the Reliability and Security of the Information Infrastructure Used to Manage the Power System. In *Power Engineering Society General Meeting, 2007. IEEE*, pages 1–8. IEEE, 2007.

[287] Sean Marston, Zhi Li, Subhajyoti Bandyopadhyay, Juheng Zhang, and Anand Ghalsasi. Cloud Computing — The Business Perspective. *Decision support systems*, 51(1):176–189, 2011.

[288] Mininet Network Emulator. Last Accessed July 2016.

[289] OPEN VSWITCH. Open vSwitch, 2013.

[290] J Mccauley. POX: A Python-based OpenFlow Controller, 2014.

[291] Natasha Gude, Teemu Koponen, Justin Pettit, Ben Pfaff, Martín Casado, Nick McKeown, and Scott Shenker. NOX: Towards an Operating System for Networks. *ACM SIGCOMM Computer Communication Review*, 38(3):105–110, 2008.

[292] Pymodbus Library. Last retrieved, July 2016.

[293] Stefan Feuerhahn. OpenMUC: j60870 Library Module. Copyright ©by Fraunhofer-Gesellschaft. Last retrieved, July 2016.

[294] Simple-RBAC Library. Last retrieved, July 2016.

[295] P Biondi. Scapy, a Powerful Interactive Packet Manipulation Program, 2010.

[296] Modbus Traffic Generator. Last retrieved, July 2016.

[297] Zachary Weinberg, Jeffrey Wang, Vinod Yegneswaran, Linda Briesemeister, Steven Cheung, Frank Wang, and Dan Boneh. StegoTorus: a Camouflage Proxy for the Tor Anonymity System. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 109–120. ACM, 2012.

[298] Yatin Chawathe, Steve A Fink, Steven McCanne, and Eric A Brewer. A Proxy Architecture for Reliable Multicast in Heterogeneous Environments. In *Proceedings of the sixth ACM international conference on Multimedia*, pages 151–159. ACM, 1998.

[299] Kevin R Fall and W Richard Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*. addison-Wesley, 2011.

[300] Ajay Bakre and BR Badrinath. I-TCP: Indirect TCP for Mobile Hosts. In *Proceedings of the 15th International Conference on Distributed Computing Systems*, pages 136–143. IEEE, 1995.

[301] J Border et al. Performance Enhancing Proxies Intended to Mitigate Link Related Degradations. Technical report, RFC 3135, June 2001.

[302] The Logging Facility for Python. Last retrieved, July 2016.

[303] Giuseppe Bernieri, Estefanía Etchevés Miciolino, Federica Pascucci, and Roberto Setola. Monitoring System Reaction in Cyber-Physical Testbed Under Cyber-Attacks. *Computers & Electrical Engineering*, 2017.

[304] William Stallings. *Handbook of Computer-Communications Standards; Vol. 1: The Open Systems Interconnection (OSI) Model and OSI-Related Standards*. Macmillan Publishing Co., Inc., 1987.