UNIVERSITY OF GENOVA

PHD PROGRAM IN SCIENCE AND TECHNOLOGY FOR ELECTRONIC AND TELECOMMUNICATION ENGINEERING

Machine Learning based Anomaly Detection for Cybersecurity Monitoring of Critical Infrastructures

PhD Thesis

Thesis submitted for the degree of *Doctor of Philosophy* (34° cycle)

February 2022

PhD Candidate
Giovanni Battista Gaggero

Tutor
prof. Mario Marchese
prof. Paola Girdinio

Coordinator of the PhD Course:
prof. Maurizio Valle

DITEN

Electrical, Electronics and Telecommunication Engineering and Naval Architecture Department

To whoever taught me something. To Emma, who taught me love.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Giovanni Battista Gaggero
February 2022

Acknowledgements

I would like to thank my family, which helped me to manage all the inconveniences that happened during these years. I would like to express my very great appreciation to Trillo and Remì, which made me work much less than I could. I would like to thank the staff of the SCNL lab: Fabio Patrone, Alessandro Fausto, Aya Moheddine, which shared with me the burden of challenging deadlines. A special thank to Prof. Mansueto Rossi, that is the most disposable person I met during my research career. Finally, I wish to thank Professor Mario Marchese and Professor Paola Girdinio, my research supervisors, for for the trust they gave me in this challenging pathway.

Abstract

Managing critical infrastructures requires to increasingly rely on Information and Communication Technologies. The last past years showed an incredible increase in the sophistication of attacks. For this reason, it is necessary to develop new algorithms for monitoring these infrastructures. In this scenario, Machine Learning can represent a very useful ally. After a brief introduction on the issue of cybersecurity in Industrial Control Systems and an overview of the state of the art regarding Machine Learning based cybersecurity monitoring, the present work proposes three approaches that target different layers of the control network architecture. The first one focuses on covert channels based on the DNS protocol, which can be used to establish a command and control channel, allowing attackers to send malicious commands. The second one focuses on the field layer of electrical power systems, proposing a physics-based anomaly detection algorithm for Distributed Energy Resources. The third one proposed a first attempt to integrate physical and cyber security systems, in order to face complex threats. All these three approaches are supported by promising results, which gives hope to practical applications in the next future.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Industrialized societies depend on the proper functioning of a set of technological infrastructures, such as electricity, road and rail networks and telecommunications which, because of their relevance, are generally referred to as critical infrastructure. The USA Committee on National Security System defines them as (3)

> *Critical Infrastructures are systems and assets, whether physical or virtual, so vital for a state 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.*

These infrastructures, once essentially isolated and vertically integrated systems, have become increasingly interdependent to such an extent that an adverse event that occurs to one of them, in a given geographical location, may spread to other infrastructures amplifying the negative effects and afflicting subjects also located in very remote locations compared to the origin of the initial event (4).

Several episodes in the last decade have highlighted how the growing complexity of these infrastructures has meant that they are fragile, to the point that some scholars consider almost inevitable catastrophic events that lead to their damage. According to (1), the interdependencies are analyzed considering six different "dimensions" in order to capture the different elements that characterize both the behavior linked to the presence of interdependence and its arise, as shown in Figure 1.1. In particular, they identify in which directions the analysis should be developed:

- Environment: that is the structure within which owners and operators establish purposes and goals, build value systems to define their business, etc. Obviously the operational

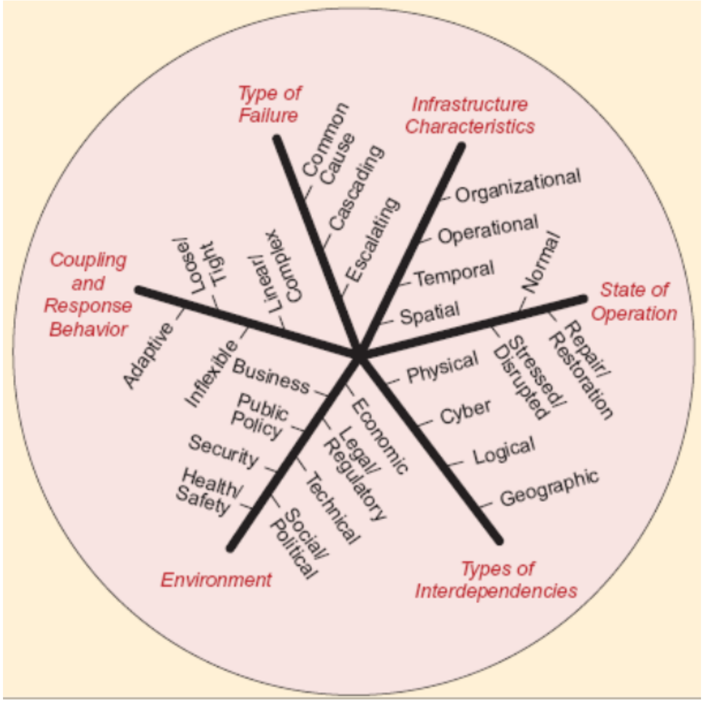

Figure 1.1 The six dimensions of interdependencies in critical infrastructures (1)

status and conditions of each infrastructure affects the surrounding environment and, in turn, the environment influences the infrastructure itself.

- Types of Interdependence: an interdependence can be classified as physical, if the two infrastructures are physically interdependent and the state of one is dependent on the material output of the other (for example, a central coal-fired electricity and its supply rail network); cyber, if the status of both depends on information transmitted through cyberspace; geographic, when a local environmental event may cause changes in the status of other infrastructures; logical, when the state of the infrastructures depends on the state of the other through a mechanism that is not anybody of those previously explicited. This classification can also be extended to include interdependent sociological relationships, caused by the (irrational) behavior of users/operators. In this way, it is possible to model phenomena such as saturation of communication channels in presence of crisis events or the fact that operators may disregard their tasks to ethical or social reasons.

- Operational status: To fully understand interdependencies it is necessary to determine, for each infrastructure, on which it depends, both in normal operation, in abnormal situations and during the recovery phase following a failure/malfunction.

- Infrastructure characteristics: In this context, elements such as spatial scale, about which a hierarchy of elements can be defined, and the time scale, since the horizon of interest may vary of seconds (for energy system operations, for example), of hours (for operations connected with the supply of water, gas or the transmission system), over the years (for improvements or capacity building of an infrastructure).

- Fault types: Interdependencies between infrastructures can be the means through which a failure can propagate. In this perspective, we speak of cascade propagation, when the malfunction causes a fault in a second infrastructure, which in turn leads to an anomaly in a third party, and so on; intensifying: when a malfunction in an infrastructure makes a malfunction, independent of the first, in a second infrastructure; for common reasons: where two or more infrastructures suffer damage in the same moment and for the same reason.

- Coupling level: depending on the degree of coupling (narrow or slack), varies both the propagation time and the transmitted intensity of any malfunction. Such interactions may be either linear, if they are the result of the design (generally known, visible and generated by a planned sequence of operations), or complex, when they occur unexpectedly to following unscheduled sequences of operations.

Note that, unlike the others, Cyber interdependence is an absolute property and not relative. This underlines how, this type of interaction, may involve an extended interdependence (substantially) with any other infrastructure using cyberspace. Many cyberattacks indeed spread further beyond the main target of the attackers. One of the most famous examples is Stuxnet (5): according to (6), as of October 2010, there were approximately 100000 infected hosts, from over 155 countries, even the vast majority (60%) was in Iran.

In this context, it is crucial to develop proper cybersecurity monitoring of such infrastructures. The purpose of this work is to address the issue of cybersecurity monitoring from different perspectives. The state of the art presents a scenario in which algorithms are becoming more specialized to detect specific attacks in specific infrastructures since the attacks themselves are growing in complexity. For these reasons, cybersecurity monitoring systems will be composed of several subsystems, each of them addressing specific issues.

The present work is structured as follows. Chapter 2 introduces the issue of cybersecurity in industrial control systems, discussing the models, procedures, standards and guidelines that are commonly used, and provides also a brief overview of real attacks towards critical infrastructures which had success in the last past years. Chapter 3 discusses the applications of Machine Learning technologies in cybersecurity monitoring, with a particular focus on anomaly detection approaches, including traditional Intrusion Detection Systems and novel approaches, like physics-based anomaly detection. Then, three main novel approaches, addressing the issue of cybersecurity monitoring on different layers of the control systems are presented. Chapter 4 presents a particular covert channel attack based on the DNS protocol, and presents an ensemble classifier built on different algorithms already in the state of the art, showing how such classifier outperforms each single algorithm. In Chapter 5 the smart grid environment for distribution networks and microgrids is analyzed, in order to highlight the main possible vulnerabilities; then, a physics-based anomaly detection algorithm for monitoring Distributed Energy Resources is presented, and the approach is validated in a simulation environment of a photovoltaic system connected to the grid. Chapter 6 proposes a novel approach that expand the possibilities of cybersecurity monitoring by integrating two traditionally separated systems: the physical security monitoring systems, like access control, and cybersecurity log systems, like Intrusion Detection or Firewalls. All of the three Chapters present also a discussion of the results and the future developments. Finally, in Chapter 7 the conclusions are drawn.

Chapter 2

Industrial Control Systems Cybersecurity

Industrial control system (ICS) is a general term that encompasses several types of control systems, including supervisory control and data acquisition (SCADA) systems, distributed control systems (DCS), and other control system configurations such as Programmable Logic Controllers (PLC) often found in the industrial sectors and critical infrastructures. An ICS consists of combinations of control components (e.g., electrical, mechanical, hydraulic, pneumatic) that act together to achieve an industrial objective (e.g., manufacturing, transportation of matter or energy) (7)

ICSs increasingly rely on Information and Communication Technologies (ICT). Many of today's ICSs evolved from the insertion of IT capabilities into existing physical systems, often replacing or supplementing physical control mechanisms.

The engineering of ICSs continues to evolve to provide new capabilities while maintaining the typical long lifecycles of these systems. The introduction of IT capabilities into physical systems presents emergent behavior that has security implications. Engineering models and analyses are evolving to address these emergent properties including safety, security, privacy, and environmental impact interdependencies.

A typical ICS contains numerous control loops, human interfaces, and remote diagnostics and maintenance tools built by using an array of network protocols on layered network architectures. A control loop utilizes sensors, actuators, and controllers (e.g., PLCs) to manipulate the process. A sensor is a device that produces a measurement of some physical property and then sends this information as controlled variables to the controller. The controller interprets the signals and generates corresponding manipulated variables, based on a control algorithm and target set points, which it transmits to the actuators. Actuators such as control valves, breakers, switches, and motors are used to directly manipulate the controlled process based on commands from the controller, as shown in Figure 2.1. Operators and engineers use human interfaces to monitor and configure set points, control algorithms, and to adjust and establish parameters in the controller. The human interface also displays process status information and historical information.

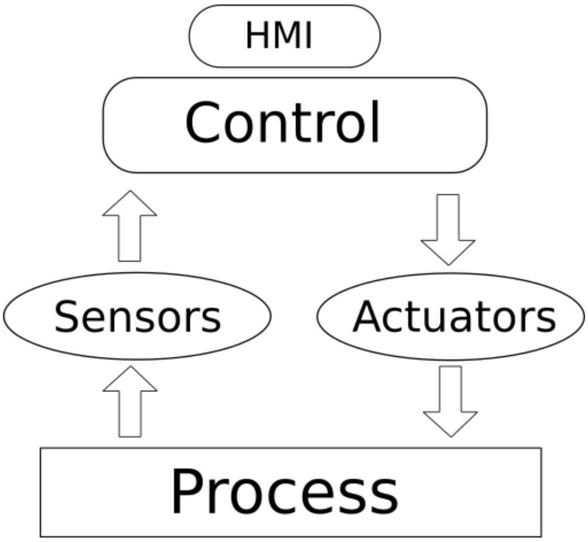

Figure 2.1 Industrial Automation logical scheme

Diagnostics and maintenance utilities are used to prevent, identify, and recover from abnormal operations or failures. Sometimes these control loops are nested and/or cascading whereby the set point for one loop is based on the process variable determined by another loop. Supervisory level loops and lower level loops operate continuously throughout a process with cycle times ranging in the order of milliseconds to minutes.

2.1 Architecture of an ICS

To cope with the complexity, different models have been developed to represent ICS systems. Most of the time, the large scale of these systems, as well as the diversity of devices and requirements, requires ICS systems to be segmented into multiple operational zones. From a cyber security perspective, ICS systems can be broadly segmented into three different zones (8):

- Enterprise zone
- Control zone

- Field zone

The Enterprise zone includes business networks and enterprise systems; it includes diverse endpoint devices that evolve rapidly and are upgraded continuously, including business networks, commonly based on the IP protocol and very often connected to external networks and the Internet. The enterprise zone is very similar to traditional IT environments found outside the realm of ICSs. Therefore, many cybersecurity solutions from the IT world can be directly applied. These networks are most of the time kept separate from the operational networks used in the other zones.

The Control zone includes the distributed control elements in SCADA systems. These zones include the control room environments. The Control zone shares a few similarities with the Enterprise zone, such as networks based on the IP protocol. The requirements of the Control zone, however, shift drastically to emphasize safety and reliability. The devices in this zone may not be updated as often and the networks may be subject to strict timing constraints. Therefore, few cybersecurity solutions from the IT world can be directly used in this zone.

The Field zone, also known as the plant, process, or operations zone, includes the devices and networks in charge of control and automation. The devices in this zone often include single-purpose embedded devices, such as Programmable Logic Controllers, which have constrained computational resources. The communication networks in this zone are much more diverse and go beyond IP networks, employing a large variety of industrial protocols and physical interfaces. Devices and networks in the field zone are subject to strict safety, reliability, and timing requirements. Therefore, the cybersecurity solutions from the IT world rarely, if ever, apply.

This three-tiered model is admittedly oversimplified. A slightly more complex architecture is presented in the so-called "Purdue Model" (9), which is adopted by the ISA-95 standard. The standard's purpose is "To create a standard that will define the interface between control functions and other enterprise functions based upon the Purdue Reference Model for CIM (hierarchical form) as published by ISA. The interface initially considered is the interface between levels 3 and 4 of that model. Additional interfaces will be considered, as appropriate. The goal is to reduce the risk, cost, and errors associated with implementing these interfaces. The standard must define information exchange that is robust, safe, and cost-effective. The exchange mechanism must preserve the integrity of each system's information and span of control."

The Purdue Model defines 5 principal layers, and two Zones, as shown in Figure 2.2:

- Level 0: consists of a wide variety of sensors, actuators, and devices involved in the basic manufacturing process. These devices perform the basic functions of the industrial automation and control system, such as driving a motor, measuring variables, setting an output, and performing key functions such as painting, welding, bending, and so on.

- Level 1: consists of basic controllers that control and manipulate the manufacturing process which its key function is to interface with the Level 0 devices (I/O, linking devices, bridges, etc). In discrete manufacturing, this is typically a programmable logic controller (PLC). In process manufacturing, the basic controller is referred to as a distributed control system (DCS).

- Level 2 represents the systems and functions associated with the runtime supervision and operation of an area of a production facility. These include operator interfaces or Human Machine Iinterfaces (HMIs), alarms or alerting systems, Process historian batch management systems, Control room workstations.

- Level 3: represents the highest level of industrial automation and control systems. The systems and applications that exist at this level manage site-wide industrial automation and control functions

- Level 4 is where the functions and systems that need standard access to services provided by the enterprise network reside. This level is viewed as an extension of the enterprise network. The basic business administration tasks are performed here and rely on standard IT services. These functions and systems include wired and wireless access to enterprise network services such as Internet access, E-mail, Non-critical production systems such as manufacturing execution systems and overall plant reporting, Enterprise applications.

- Level 5 is where the centralized IT systems and functions exist. Enterprise resource management, business-to-business, and business-to-customer services typically reside at this level. The industrial automation and control systems must integrate with the enterprise applications to exchange production and resource data. Direct access to the industrial automation and control systems is typically not required, with the exception of partner access. Access to data and the industrial automation and control network must be managed and controlled to maintain availability and stability.

The Manufacturing and Cell/Area Zone is the Security critical environment of the whole architecture. A violation of a device or of communication at any level of these areas can

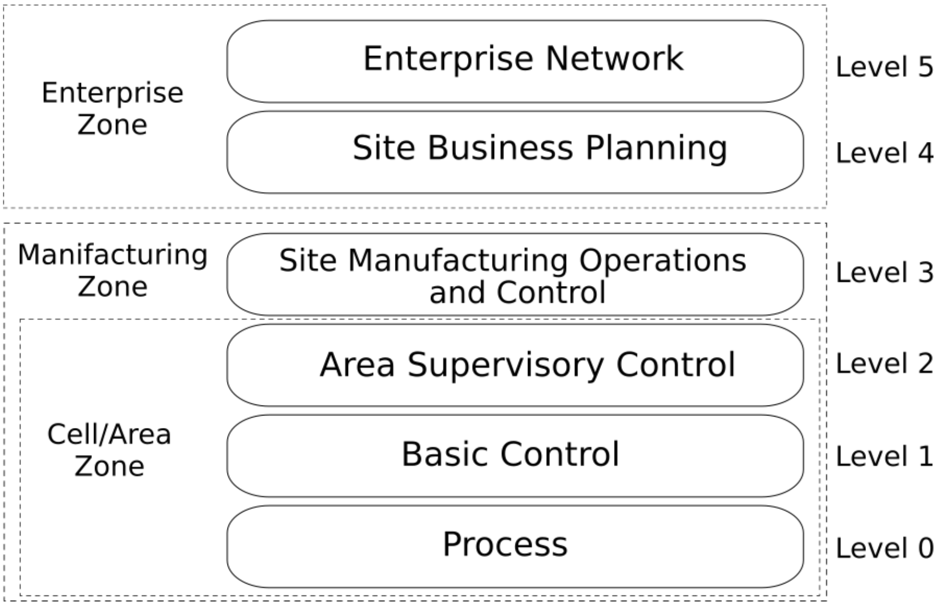

Figure 2.2 Architecture of the Purdue Model

afflict the safety of the whole process, with severe potential consequences on the integrity of devices and even people. For these reasons, the interfaces between different levels and zones of the Purdue model must be carefully designed.

Despite the Model's influence, in the IoT era, the flow of data is no longer strictly hierarchical as prescribed in the Purdue Model. As intelligence is added to sensors and actuators (Level 0 of the Purdue Model) and controllers (Level 1 of the Purdue Model), new potentials for control system exposure are occurring much further down the pyramid than the Purdue Model ever considered. And with the use of edge computing devices becoming more common, vast amounts of data can be collected at Level 1, processed, and sent directly to the cloud, thereby bypassing the hierarchical structure of data flows in the Purdue Model.

Nevertheless, the basic idea is that in order to properly defend the control network, we have to define the boundaries of the network, and to identify areas which require stronger controls for ensuring the safety and the availability of the whole industrial process.

This problem is addressed by the standard IEC 62443, which introduces the concepts of Zones, Conduits, and Security Levels in order to provide a methodology for the security assessment of an infrastructure. The IEC 62443 will be detailed in Section 2.3

2.2 Defense In Depth

An organization's cybersecurity strategy should protect the assets that it deems critical to successful operation. Unfortunately, there are no shortcuts, simple solutions, or "silver bullet" implementations to solve cybersecurity vulnerabilities within critical infrastructure ICS. It requires a layered approach known as Defense in Depth. Defense in Depth is a concept originated in military strategy to provide barriers to impede the progress of intruders from attaining their goals while monitoring their progress and developing and implementing responses to the incident in order to repel them. In the cybersecurity paradigm, Defense in Depth correlates to detective and protective measures designed to impede the progress of a cyber intruder while enabling an organization to detect and respond to the intrusion with the goal of reducing and mitigating the consequences of a breach.

According to (10), applying the Defense-in-Depth paradigm in an ICS environment involves the following procedures and Technologies (Table 2.1):

Of course, improving cybersecurity posture by implementing an ICS Defense-in-Depth strategy starts with developing an understanding of the business risk associated with ICS cybersecurity and managing that risk according to the overall business risk appetite. Risk management can be defined as the process of identification, evaluation, and prioritization of risks, followed by coordinated and economical application of resources to minimize, monitor, and control the probability or impact of unfortunate events or to maximize the realization of opportunities. This problem is addressed by many standards, including NIST 800-83 (11). Organizations also have to face many challenges in managing the human factor within ICS organizations. Large and complex systems are susceptible to mistakes made by inexperienced or untrained personnel, as well as the activities of malicious insider threats. Organizations should design clear and actionable policies and procedures, and provide security training and awareness activities to the personnel. A growing problem is also represented by Supply Chain Management. ICS manufacturers and software developers create their products in many different locations around the world. Ensuring the security of the system or application throughout its development life cycle is impossible for most ICS operators. Unfortunately, purchasing commercial off-the-shelf (COTS) technologies increases the likelihood of receiving nongenuine equipment. The same caution shall be used for cloud services. It's not the purpose of this work to discuss organizational aspects of cybersecurity. This work will focus on technological aspects, and particularly on security monitoring.

Table 2.1 Defense in Depth Procedures and Technologies

Defense in Depth Strategy Elements	
Risk Management Program	• Identify Threats • Characterize Risk • Maintain Asset Inventory
Cybersecurity Architecture	• Standards/ Recommendations • Policy • Procedures
Physical Security	• Field Electronics Locked Down • Control Center Access Controls • Remote Site Video, Access Controls, Barriers
ICS Network Architecture	• Common Architectural Zones • Demilitarized Zones (DMZ) • Virtual LANs
ICS Network Perimeter Security	• Firewalls/ One-Way Diodes • Remote Access & Authentication • Jump Servers/ Hosts
Host Security	• Patch and Vulnerability Management • Field Devices • Virtual Machines
Security Monitoring	• Intrusion Detection Systems • Security Audit Logging • Security Incident and Event Monitoring
Vendor Management	• Supply Chain Management • Managed Services/ Outsourcing • Leveraging Cloud Services
The Human Element	• Policies • Procedures • Training and Awareness

The first step to secure a control system is to define the boundaries of the control network, defining the accesses to the external network and implementing perimeter defense. The convergence of once-isolated ICSs has helped organizations simplify the management of complex environments. Nevertheless, connecting these networks and incorporating IT components into the ICS domain introduces vulnerabilities that asset owners must address before issues arise. Merging a modern IT architecture with an isolated network that may not have any countermeasures in place is challenging. Using simple connectivity (that is, routers and switches) provides the most obvious way to interconnect networks; however, unauthorized access by an individual could result in unlimited access to the ICS. Securing a control network from the design stage, the so-called "Security by Design" is, of course, much simpler than operating on existing infrastructures.

The architecture of and ICS has been discussed in Section 2.1. Dividing common control system architectures into zones can assist organizations in creating clear boundaries in order to effectively apply multiple layers of defense. Understanding how to achieve network segmentation is vital to create architectural zones and determining the best methodologies for segmenting networks within and around control system environments In respect to Figure 2.2, the main goal is to clearly define the allowed communication between the Manufacturing Zone and the Enterprise Zone, and Between the Enterprise Zone and the Internet.

From a traditional perspective, in an attack scenario, the intrusion begins at some point outside the control zone and the actor pries deeper and deeper into the architecture. Layered strategies that secure each of the core zones can create a defensive strategy with depth, offering the administrators more opportunities for information and resource control, as well as introducing cascading countermeasures that will not necessarily impede business functionality. This is guaranteed through secure network architecture, involving VLANs, Demilitarized Zones (DMZ), and devices at the "gate" of the zones, like Firewalls. A demilitarized zone is a physical and logical sub-network that acts as an intermediary for connected security devices so that they avoid exposure to a larger and untrusted network, usually the Internet. The ability to establish a DMZ between the corporate and control networks represents a significant improvement with the use of firewalls. Each DMZ holds one or more critical components, such as the data historian, the wireless access point, or remote and third-party access systems. Creating a DMZ requires that the firewall offers three or more interfaces, rather than the typical public and private interfaces. One of the interfaces connects to the corporate network, the second to the control network, and the remaining interfaces to the shared or insecure devices such as the data historian server or wireless access points on the DMZ network. In this way, by placing corporate-accessible components in a

DMZ, no direct communication paths are required from the corporate network to the control network; each path effectively ends in the DMZ. The role of Firewalls is to establish domain separation, monitor (and log) system events, authenticate users before they are allowed access, monitor ingress and egress traffic and disallow unauthorized communications. These tasks can be done by using different techniques, controlling network traffic at different layers of the OSI architecture.

Actually, this vision is only partially true. It's not mandatory that an attack begins from some areas external to the whole ICS. In some scenarios, an attacker could gain access to some point within the control network, both in the enterprise or manufacturing area. These scenarios include the presence of intruders, people that use personal devices within the control network (Bring-Your-Own-Device), but also physically dislocated areas (which is particularly common for electrical grids) for which may be difficult to guarantee physical security for all the devices. All the previously discussed countermeasures remain still valid, but simply insufficient.

The present work focuses on another fundamental element of the defense-in-depth paradigm, which is the security monitoring. The concept of Defense in Depth says a system must detect and alert an organization of an intrusion early on so they can take defensive action before critical assets are breached. Without system monitoring in place, intruders could breach the system and no one would know of the intrusion before they achieved their objective, if they know at all. Security monitoring can be achieved through different technologies, including Intrusion Detection Systems and Security Incident and Event Monitoring (SIEM). The state of the art of Intrusion Detection Systems will be presented in Section 3.2. Chapters 4, 5 and 6 present some innovative solution to address problems in different layers of the ICS infrastructure.

2.3 Standards and Best Practices

Many standards address the issue of cyber security of ICS. The main normative agencies are the International Organization for Standardization (ISO), the International Electrotechnical Commission (IEC), the National Institute of Standards and Technology (NIST), the Institute of Electrical and Electronics Engineers (IEEE), and the Internet Engineering Task Force (IETF).

The International Organization for Standardization (ISO) is an international standard-setting body composed of representatives from various national standards organizations. Founded on 23 February 1947, the organization develops and publishes worldwide technical, industrial and commercial standards. It is headquartered in Geneva, Switzerland and works in 165 countries.

The International Electrotechnical Commission is an international standards organization that prepares and publishes international standards for all electrical, electronic and related technologies – collectively known as "electrotechnology". IEC standards cover a vast range of technologies; all electrotechnologies are covered by IEC Standards, including energy production and distribution, electronics, magnetics and electromagnetics, electroacoustics, multimedia, telecommunication and medical technology, as well as associated general disciplines such as terminology and symbols, electromagnetic compatibility, measurement and performance, dependability, design and development, safety and the environment. The IEC also manages four global conformity assessment systems that certify whether equipment, system or components conform to its international standards.

The National Institute of Standards and Technology is a physical sciences laboratory and non-regulatory agency of the United States Department of Commerce. Its mission is to promote American innovation and industrial competitiveness. NIST's activities are organized into laboratory programs that include nanoscale science and technology, engineering, information technology, neutron research, material measurement, and physical measurement.

The Institute of Electrical and Electronics Engineers (IEEE) is a professional association for electronic engineering and electrical engineering (and associated disciplines) with its corporate office in New York City and its operations center in Piscataway, New Jersey. It was formed in 1963 from the amalgamation of the American Institute of Electrical Engineers and the Institute of Radio Engineers. As of 2018, it is the world's largest association of technical professionals with more than 423,000 members in over 160 countries around the world. Its objectives are the educational and technical advancement of electrical and electronic engineering, telecommunications, computer engineering and similar disciplines.

The Internet Engineering Task Force (IETF) is an open standards organization, which develops and promotes voluntary Internet standards, in particular the technical standards that comprise the Internet protocol suite (TCP/IP). It has no formal membership roster or membership requirements. All participants and managers are volunteers, though their work is usually funded by their employers or sponsors. The IETF started out as an activity supported by the federal government of the United States, but since 1993 it has operated as a standards-development function under the auspices of the Internet Society, an international membership-based non-profit organization.

We can distinguish between general standards, which can be applied to a large variety of infrastructures, and standards designed to be applicable in specific domains. ISO standards, like ISO 27001, are general standards for managing information security, which can be applied in the ICS environment. More specific for ICS are the IEC 62443 and NIST 800-82. Then, others address specific sectors. Considering the electrical domain, we can mention the ones from IEC, like IEC 62351, and some works provided by the IEEE, like the IEEE 1686. Most of them refer to RFC for the security of Internet Protocols. An overall picture of available standards is shown in Figure 2.3

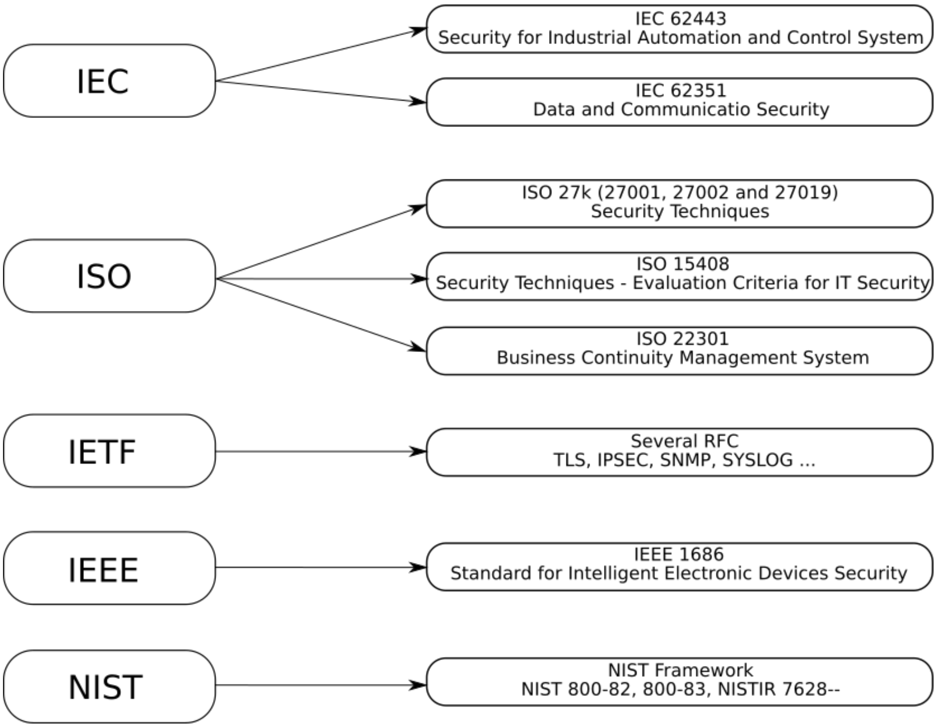
IEC
IEC 62443
Security for Industrial Automation and Control System
IEC 62351
Data and Communicatio Security
ISO
ISO 27k (27001, 27002 and 27019)
Security Techniques
ISO 15408
Security Techniques - Evaluation Criteria for IT Security
ISO 22301
Business Continuity Management System
IETF
Several RFC
TLS, IPSEC, SNMP, SYSLOG ...
IEEE
IEEE 1686
Standard for Intelligent Electronic Devices Security
NIST
NIST Framework
NIST 800-82, 800-83, NISTIR 7628--

Figure 2.3 Main Standard for Security of the Electrical Sector

The two main general standards for industrial automation and control systems are the NIST Special Publication 800-82 "Guide to Industrial Control Systems (ICS) Security" and the IEC 62443.

The NIST 800-82 is part of the NIST 800 Series. The NIST 800 Series is a set of documents that describe the United States federal government computer security policies, procedures and guidelines. NIST (National Institute of Standards and Technology) is a unit of the Commerce Department. The documents are available free of charge and can be useful to businesses and educational institutions, as well as to government agencies. The publications cover all NIST-recommended procedures and criteria for assessing and documenting threats and vulnerabilities and for implementing security measures to minimize the risk of adverse events. The publications can be useful as guidelines for enforcement of security rules and as legal references in case of litigation involving security issues. The purpose of NIST 800-82 is to provide guidance for securing industrial control systems, including supervisory control and data acquisition systems, distributed control systems, and other systems performing control functions. The document provides a notional overview of ICS, reviews typical system topologies and architectures, identifies known threats and vulnerabilities to these systems, and provides recommended security countermeasures to mitigate the associated risks. The structure of the document is reported in Table 2.2

Table 2.2 Structure of the NIST 800-82 Standard

Section	Description
Chapter 1	Introduction
Chapter 2	Overview of Industrial Control Systems
Chapter 3	ICS Risk Management and Assessment
Chapter 4	ICS Security Program Development and Deployment
Chapter 5	ICS Security Architecture
Chapter 6	Applying Security Controls to ICS

The document provides very practical guidelines for managing an ICS, like how to design the architecture of the control network in order to respect the Network Segmentation and Segregation, or how to implement a firewall, also providing recommended firewall rules for specific services.

IEC 62443 is an international series of standards on cybersecurity for ICS. It describes both technical and process-related aspects of industrial cybersecurity. It divides the industry into different roles: the operator, the integrators (service providers for integration and maintenance) and the manufacturers. The different roles each follow a risk-based approach to prevent and manage security risks in their activities. Initially, the ISA99 committee considered IT standards and practices for use in the ICS. However, it was soon found that this was not sufficient to ensure the safety, integrity, reliability, and security of an ICS. Starting in 2002, the Industrial Automation and Control System Security Committee (ISA99) of the International Society of Automation developed a multi-part series of standards and technical reports on the subject of Industrial Automation and Control System (IACS) security. These work products were submitted to the ISA for approval and then published under ANSI. The documents were originally referred to as ANSI/ISA-99 or ISA99 standards and were renumbered to be the ANSI/ISA-62443 series in 2010. The content of this series was submitted to and used by the IEC working groups. The structure of the standard is reported in Table 2.3.

Table 2.3 Structure of the IEC 62443

Part	Section	Description
General	62443-1-1	Terminology, concepts and models
	62443-1-2	Master glossary of terms and abbreviations
	62443-1-3	System security conformance metrics
	62443-1-4	IACS security lifecycle and use-cases
Policies & Procedures	62443-2-1	Establishing an IACS security program
	62443-2-2	IACS security program rating
	62443-2-3	Patch management in the IACS environment
	62443-2-4	Security program requirements for IACS service providers
	62443-2-5	Implementation guidance for IACS asset owners
System	62443-3-1	Security technologies for IACS
	62443-3-2	Security risk assessment for system design
	62443-3-3	System security requirements and security levels
Component	62443-4-1	Product security development lifecycle requirements
	62443-4-2	Technical security requirements for IACS components

The standard introduces some fundamental concepts. Part 3-2 describes the requirements for addressing the cybersecurity risks in an IACS, including the use of Zones and Conduits, and Security Levels. A Zone is defined as a grouping of logical or physical assets based upon risk or other criteria such as criticality of assets, operational function, physical or logical location, required access, or responsible organization. A Conduit is defined as a logical grouping of communication channels that share common security requirements connecting two or more zones. A key step in the Risk Assessment process is to partition the System Under Consideration into separate Zones and Conduits. The intent is to identify those assets which share common security characteristics in order to establish a set of common security requirements that reduce cybersecurity risk. Partitioning the System Under Consideration into Zones and Conduits can also reduce overall risk by limiting the scope of a successful cyber-attack. Security Level is defined as the measure of confidence that the System Under Consideration, Zone, or Conduit is free from vulnerabilities and functions in the intended manner. There are four levels:

1. Protection against casual or coincidental violation
2. Protection against intentional violation using simple means with low resources, generic skills, and low motivation
3. Protection against intentional violation using sophisticated means with moderate resources, IACS-specific skills, and moderate motivation

4. Protection against intentional violation using sophisticated means with extended resources, IACS-specific skills, and high motivation

There are three types of Security Levels that are used throughout the ISA/IEC 62443 Series:

- Capability Security Levels: are the security levels that systems or components can provide when properly integrated and configured.
- Target Security Levels: are the desired level of security for a particular Automation Solution
- Achieved Security Levels: are the actual levels of security for a particular Automation

The ISA Global Security Alliance and the ISA Security Compliance Institute recently released a co-sponsored Industrial Internet of Things (IIoT) certification study entitled, "IIoT Component Certification Based on the 62443 Standard." The study addresses the urgent need for industry-vetted IIoT certification programs, with the goal of determining the applicability of the ISA/IEC 62443 series of standards and certifications to IIoT components and systems. This included examining whether existing 62443 requirements and methods for validating these requirements under existing certification programs are necessary and sufficient for the IIoT environment.

Many standards address specific issues in specific sectors. Regarding the electrical sector for example, one of the most important standard is represented by the IEC 62351, even if is not yet largely implemented in industrial plants.

2.4 Real Cyberattacks against ICS

In the last past years, the number of attacks targeting control systems is incredibly intensifying, as shown in Figure 2.4.

The first malware who brought the attention on the topic worldwide has been without doubt the Stuxnet worm (5). Stuxnet is a malicious computer worm first uncovered in 2010 and thought to have been in development since at least 2005. The worm was at first identified by the security company VirusBlokAda in mid-June 2010.

Stuxnet differs from past malware in several ways (12). First, most malwares try to infect as many computers as possible, whereas Stuxnet appears to target industrial control systems and delivers its payload under very specific conditions. In particular, Stuxnet attacks Windows

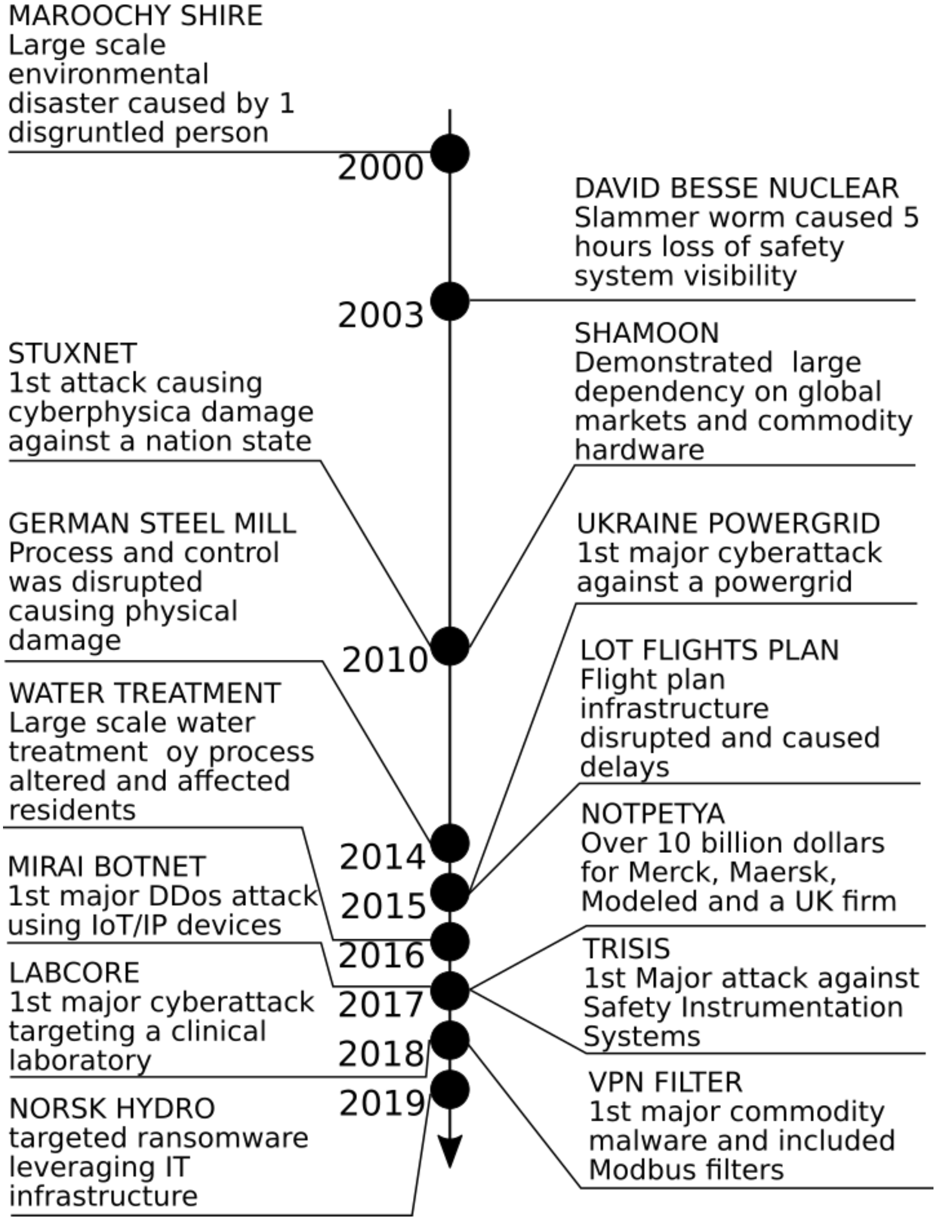
MAROOCHY SHIRE
Large scale environmental disaster caused by 1 disgruntled person
2000
DAVID BESSE NUCLEAR
Slammer worm caused 5 hours loss of safety system visibility
2003
STUXNET
1st attack causing cyberphysica damage against a nation state
SHAMOON
Demonstrated large dependency on global markets and commodity hardware
GERMAN STEEL MILL
Process and control was disrupted causing physical damage
UKRAINE POWERGRID
1st major cyberattack against a powergrid
2010
WATER TREATMENT
Large scale water treatment oy process altered and affected residents
LOT FLIGHTS PLAN
Flight plan infrastructure disrupted and caused delays
NOTPETYA
Over 10 billion dollars for Merck, Maersk, Modeled and a UK firm
2014
MIRAI BOTNET
1st major DDos attack using IoT/IP devices
2015
2016
LABCORE
1st major cyberattack targeting a clinical laboratory
2017
TRISIS
1st Major attack against Safety Instrumentation Systems
2018
VPN FILTER
1st major commodity malware and included Modbus filters
2019
NORSK HYDRO
targeted ransomware leveraging IT infrastructure

Figure 2.4 Timeline of attacks targeting ICS

PCs that program specific Siemens programmable logic controllers. When an infected PC connects to a Siemens Simatic PLC, Stuxnet installs a malicious .dll file, replacing the PLC's original .dll file. The malicious .dll file lets Stuxnet monitor and intercept all communication between the PC and PLC. Depending on specific PLC conditions, Stuxnet injects its own code onto the PLC in a manner undetectable by the PC operator. Second, Stuxnet is larger and more complex than other malware. It contains exploits for four unpatched vulnerabilities—an unusually high number. The code is approximately 500 Kbytes and written in multiple languages. Stuxnet's sophistication points to an unusually high effort level. Ilias Chantzos, director of government relations at Symantec, estimated the manpower required to develop Stuxnet to have been 5 to 10 people working for six months with access to SCADA systems. All reports examining Stuxnet have agreed on the likelihood of at least one government's involvement in its development. Besides detailed insider knowledge of the target, other aspects suggest that Stuxnet's creators expended considerable resources. The code contains an unprecedented four zero-day Windows exploits. Attackers value zero-day exploits, so four represents an unusually high investment. The Conficker worm likewise exploited the Windows Server Service RPC vulnerability, for which Microsoft issued a patch in 2008, but Stuxnet's creators seemed to know that patching Scada systems is time-consuming. Stuxnet is digitally signed by two certificates to appear legitimate. Initially, it used a stolen certificate from Realtek Semiconductor, but VeriSign revoked the certificate on 16 July 2010. The next day, Stuxnet was found to be using a stolen certificate from JMicron Technology, which was subsequently revoked on 22 July. The two companies are situated near each other, suggesting physical theft at those locations. A complete analysis of Stuxnet can be found in (6).

After Stuxnet, of course, other similar malwares have been found in the following years. Two malwares that have been called the "cousins" of Stuxnet are Duqu and Flame (13).

A quite common attack that has been pursued in the last years is Ransomware. Examples are Petya (14) and Wannacry (15). Even if they are not designed to target specifically ICS, the malware targets Microsoft Windows–based systems, infecting the master boot record to execute a payload that encrypts a hard drive's file system table and prevents Windows from booting. The Petya malware had infected millions of people during the first year of its release, and have been found in several critical infrastructures; one of the main concerns regards, in fact, the medical sector (16).

While Stuxnet has been the first complex malware targeting ICS, there have been other "first times" in ICS attacks. The first time the power system was shut down by a cyberattack has been in 2015. On December 23, 2015, the Ukrainian Kyivoblenergo, a regional electricity distribution company, reported service outages to customers. The outages were due to a third party's illegal entry into the company's computer and SCADA systems: Starting at approximately 3:35 p.m. local time, seven 110 kV and 23 35 kV substations were disconnected for three hours. Later statements indicated that the cyber attack impacted additional portions of the distribution grid and forced operators to switch to manual mode. The event was elaborated on by the Ukrainian news media, who conducted interviews and determined that a foreign attacker remotely controlled the SCADA distribution management system. The outages were originally thought to have affected approximately 80,000 customers, based on the Kyivoblenergo's update to customers. However, later it was revealed that three different distribution oblenergos (a term used to describe an energy company) were attacked, resulting in several outages that caused approximately 225,000 customers to lose power across various areas. The attackers demonstrated a variety of capabilities, including spear-phishing emails, variants of the BlackEnergy malware, and the manipulation of Microsoft Office documents that contained the malware to gain a foothold into the Information Technology (IT) networks of the electricity companies. Then, the adversaries demonstrated the capability and willingness to target field devices at substations, write custom malicious firmware, and render the devices, such as serial-to-ethernet convertors, inoperable and unrecoverable. In one case, the attackers also used telephone systems to generate thousands of calls to the energy company's call center to deny access to customers reporting outages. Another alarming element is represented by their capability to perform long-term reconnaissance operations required to learn the environment and execute a highly synchronized, multistage, multisite attack. A more complete analysis can be found in (17).

Another malware is worth mentioning is TRISIS, because it's the first targeting a Safety Instrumented System (SIS). TRISIS, also known as TRITON or HatMan, is a malware variant that targets Schneider Electric Triconex Safety Instrumented System (SIS) controllers, which consist of a Python script compiled with py2exe, a publicly available compiler (it is done that way to allow TRISIS to execute in an environment without requiring the prior installation of Python, which often would not make sense in an industrial environment) whose objective is to change the logic on a target SIS. Trisis has been discovered in mid-November 2017 by the Dragos, Inc. team (18), when it has been deployed against at least one victim. Safety Instrumented Systems are those control systems, maintaining safe conditions if other failures occur. It is not currently known what the specific safety implications of TRISIS would be in a production environment. However, alterations to logic on the final control element imply that there could be a risk to operational safety. Safety controllers are designed to provide robust safety for critical processes. Typically, safety controllers are deployed to provide life-saving stopping logic. These may include mechanisms to stop rotating machinery when a dangerous condition is detected, or stop inflow or heating of gasses when a dangerous temperature, pressure, or other potentially life-threatening condition exists. Safety controllers operate independently of normal process control logic systems and are focused on detecting and preventing dangerous physical events. Safety controllers are most often connected to actuators which will make it impossible for normal process control systems to continue operating. This is by design since the normal process control system's continued operation would feed into the life-threatening situation that has been detected. Safety controllers are generally a type of programmable logic controller (PLC). They allow engineers to configure logic, typically in IEC-61131 logic. While on their face they are similar to PLCs, safety controllers have a higher standard of design, construction, and deployment. They are designed to be more accurate and less prone to failure. Both the hardware and the software for these controllers must be designed and built to the Safety Integrity Level (SIL) blanket of standards (IEC-61508). This includes the use of error-correcting memories and redundant components and design that favors failing an operation safety over continuing operations. Each SIS is deployed for specific process requirements after a process hazard analysis (PHA) identifies the needs for a specific industrial environment. In this way, the systems are unique in their implementation even when the vendor technology remains the same. A safety controller's output cards usually have a firmware, and a configuration, which allows the output card to fail into a safe state should the main processors fail entirely. This may even include failing outputs to a known-safe state in the event that the safety controller loses power. Many safety controllers offer redundancy, in the form of redundant processor modules. In the case of the Triconex system, the controller utilizes three separate processor modules. The modules all run the same logic, and each module is given a vote on the output of its logic function blocks on each cycle. If one of the modules offers a different set of outputs from the other two, that module is considered faulted and is automatically removed from service. This prevents a module that is experiencing an issue such as an internal transient or bit-flip from causing an improper safety decision. Safety controller architecture has been debated in the industry. Even in this case, connecting these devices to the control network makes them prone to cyberattacks. TRISIS represents, in several ways, a 'game-changing' impact for the defense of ICS networks since an attack on an SIS is a considerable step forward in causing harm. Even indirectly, targeting SIS, the malware could even cause a loss of life.

A complete analysis of the malware in the ICS scenario is not the aim of the present work. This short overview aims to show how the scenario is continuously evolving, broadening the attacker's capabilities. Different types of critical infrastructures already suffered successful attacks, and there is no reason to believe that in the next few years many others won't be targeted.

Chapter 3

State of the art of Machine Learning Applications for Cybersecurity Monitoring

Machine Learning can be defined as the use and development of computer systems that are able to learn and adapt without following explicit instructions, by using algorithms and statistical models to analyze and draw inferences from patterns in data. Applications of machine learning methods to large databases is called data mining. The analogy is that a large volume of raw material is extracted from a mine, which when processed leads to a small amount of very precious material; similarly, in data mining, a large volume of data is processed to construct a simple model with valuable use, for example, having high predictive accuracy.

A complete overview of the state of the art of Machine Learning and its relative application in Industrial Control Systems and Cybersecurity is beyond the scope of the present work. After a brief introduction, the present work focuses on the application of ML in Intrusion Detection Systems, and on a particular field of ML called Anomaly Detection which results, for many reasons, particularly interesting for security monitoring of ICS.

3.1 Introduction to Machine Learning

The first important distinction is between Artificial Intelligence (AI) and Machine Learning (ML). While the terms are frequently used interchangeably, there are fundamental differences. Artificial Intelligence can be defined as the theory and development of computer systems able to perform tasks normally requiring human intelligence (19). This is a much broader definition of the one of machine learning, whose aim is to mimic a specific task of human intelligence, that is to learn. Finally, Deep Learning (DL) is a specific mathematical model that can be applied in Machine Learning. For these reasons, we can say that Machine Learning is a branch of Artificial Intelligence, and Deep Learning is a branch of Machine Learning, as shown in Figure 3.1.

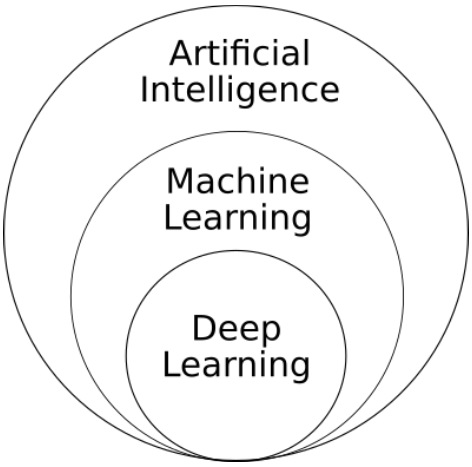

Figure 3.1 Differences between AI, ML and DL

Classical problems of Machine Learning comprehend (20):

- Regression: predict numerical valuer
- Classification: predict the belonging to one class
- Clustering: group similar examples

Regression, Classification and Clustering can be thought as answers to the question "what" ML can do. Regarding the "how", another common classification of ML algorithm is:

- Supervised Learning: given data in the form of examples with labels, the algorithm will learn to approximate the exact nature of the relationship between examples and their labels. When fully trained, the supervised learning algorithm will be able to observe a new, never-before-seen example and predict a good label for it.
- Unsupervised Learning: the algorithm would be fed a lot of data and given the tools to understand the properties of the data. From there, it can learn to group, cluster, and/or organize the data in a way such that a human (or other intelligent algorithms) can come in and make sense of the newly organized data.

- Reinforcement Learning: algorithms that improve upon themselves and learn from new situations by using a trial-and-error method.

Actually, this classification is only partially correct. There are families of algorithms that don't fit with the previous classification. One of them is anomaly detection. Anomaly detection is the task of recognizing samples that differ from those considered normal; usually these algorithms utilize datasets that are partially labeled, for example only normal data are labeled. In this sense, anomaly detection could also be defined as semi-supervised learning.

Talking about Deep Learning, algorithms have a much broader field of application (21). Deep Learning algorithms can be used for complex tasks like artificial vision, speech recognition and data generation (images, sounds, videos ...). The basic element (cell) of a neural network is the artificial neuron. The basic mathematical model of an artificial neuron is shown in figure 3.2 and Equation (3.1):

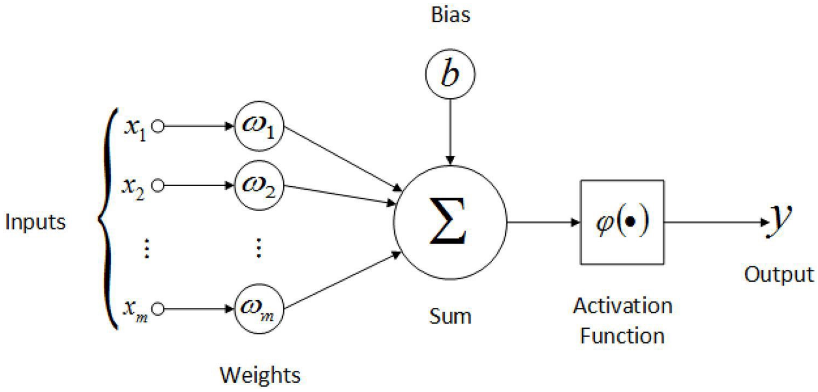

Figure 3.2 Model of an artificial neuron

$$y = \phi(\sum \omega_i x_i + b) \tag{3.1}$$

Where the activation function is can be represented by a particular function, as one of the ones shown in Figure 3.3

Other types of cells are Convolutional and Recurrent. Convolutional cells are much like feed-forward cells, except they're typically connected to only a few neurons from the previous layer. In convolutional neural networks are often utilized layers that are not actually neurons, but perform different operations like Pooling and interpolating cells. A particular type of cell is represented by recurrent cells. Recurrent cells have connections not just in the realm of layers, but also over time. Each cell internally stores its previous value. They are updated just like basic cells, but with extra weights: connected to the previous values of the cells and most of the time also to all the cells in the same layer. These weights between the

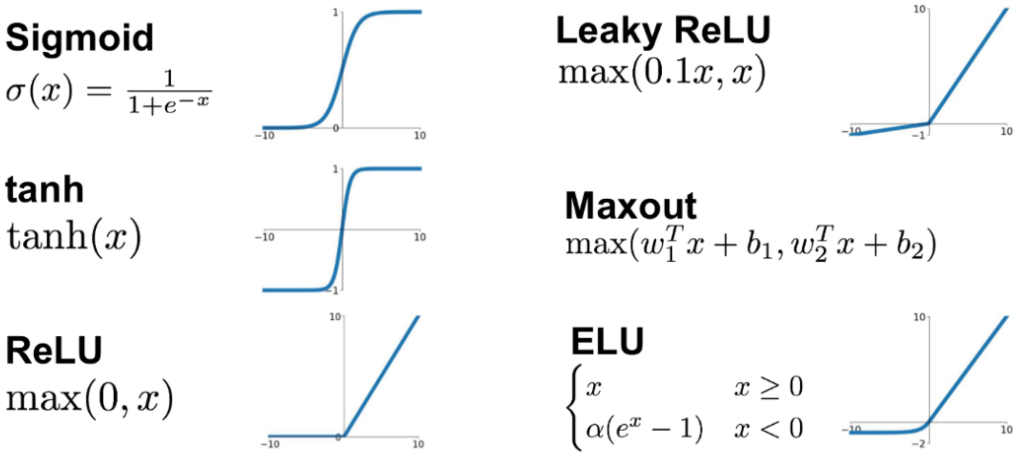

Figure 3.3 Activation Functions in Neural Networks

current value and the stored previous value work much like a volatile memory (like RAM), inheriting both properties of having a certain “state” and vanishing if not fed. Because the previous value is a value passed through an activation function, and each update passes this activated value along with the other weights through the activation function, information is continually lost. In fact, the retention rate is so low, that only four or five iterations later, almost all of the information is lost.

The most basic way of connecting neurons to form graphs is by connecting everything to absolutely everything. After a while it was discovered that breaking the network up into distinct layers is a useful feature, where the definition of a layer is a set or group of neurons which are not connected to each other, but only to neurons from other group(s). The idea of using layers is nowadays generalized for any number of layers and it can be found in almost all current architectures.

A Deep Neural Network is trained basically by minimizing the loss function which maps the input of the network with the desired output. The intuitive way to do so is to take each training example, pass through the network to get the number, subtract it from the actual number we wanted to get and square it. Once defined the loss function, the goal is to minimize it. When we start off with our neural network we initialize our weights randomly. Then, through an iterative process, weights are set. In particular, in order to minimize the loss function, it is necessary to use strategies like the Stochastic Gradient Descent, which basically iteratively tries to reduce the loss function by adjusting weights. The weights that will be used are the ones that minimize the loss function.

The use of complex architectures of artificial neurons creates a broad field of applications. In Chapter 5 a neural network architecture called autoencoder is used to build an anomaly detection algorithm.

3.2 Machine Learning for Intrusion Detection Systems

Intrusion Detection Systems (IDS) are devices or software applications that monitor a portion of the systems and try detecting malicious activities and policy violations. IDS can be classified from different viewpoints. We can identify two big families:

- Network Intrusion Detection Systems (NIDS) that analyze network traffic collected from one or more points of the communication network; and
- Host Intrusion Detection Systems (HIDS) that analyze the activity of a single host (i.e., a terminal) of the network.

Other classifications can be based on the strategy used to detect the malicious activity (signature-based or anomaly-based) or on the action that the system implements after detecting an attack (IDS can be purely passive or block traffic flows/applications, usually referred to as Intrusion Prevention System (IPS)), as shown in Table 3.1.

Table 3.1 IDS classification.

IDS Classification		
By monitored element	Network-IDS	Host-IDS
By actions	Passive (IDS)	Active (IDS and IPS)
By detection methods	Signature-based	Anomaly-based

NIDS are usually passive elements of the network. Even if it significantly slows down the responses to attack, it would be dangerous to implement an active element in a safety-critical control network due to the possible high false positive rate that would affect the whole system safety.

Several works proposed NIDS specifically designed for SCADA networks and protocols. SCADA networks are characterized by regular traffic patterns and a limited set of telecommunication protocols. The number of connections is mainly permanent, while the connectivity of particular nodes depends on their functions in the network. Such features are inherently suitable for the development and implementation of anomaly-based intrusion detection techniques. According to (22), NIDS for SCADA follows three main approaches:

- Statistical-based techniques: use statistical properties and tests to determine whether the observed behavior deviates significantly from the expected behavior. They include a number of techniques based on univariate, multivariate, time-series models and cumulative sum
- Knowledge-based techniques: try to capture the claimed behavior from the available system data. They involve techniques based on finite automata, description languages and expert systems.
- Machine learning-based techniques establish an explicit or implicit model that allows the classification of analyzed patterns. Well-known machine learning-based techniques are Bayesian networks, Markov models, neural networks, fuzzy logic, genetic algorithms, and clustering and outlier detection algorithms

(22) also provides a review of the main recent works following the mentioned approaches. Some works focus on specific protocols. Regarding the electrical sector, (23) proposes a multidimensional IDS for IEC 61850-Based SCADA networks which comprises access control detection, protocol whitelisting, model-based detection, and multiparameter-based detection, while (24) proposes a ML approach based on the extraction of statistical features on the usage of MMS and GOOSE protocols and One Class Support Vector Machine algorithm.

The other important elements of traditional security monitoring are HIDS. In the IT field, two common HIDS solutions are Open Source HIDS SECurity (OSSEC) and Tripwire. OSSEC is a free and open-source HIDS that supports a wide range of Operating Systems (OS), while Tripwire is a commercial solution. These solutions combine passive actions performed periodically in order to not affect the system performance, such as the identification of unauthorized file modifications (through, for example, file integrity checking by using checksum databases), of malicious processes, and of log behaviors (for instance by monitoring specific parameters), and active capabilities, similarly to host firewalls that allow blocking unauthorized network communications by adding firewall rules.

A further improvement in the field of HIDS is online intrusion detection (or "real-time" or "in-line" intrusion detection). Real-time HIDS analyze different features of the host, including OS aggregated behavior, such as CPU and memory metrics, shell commands, and system calls; application information, such as loaded modules and libraries, programming code, and processes; user behavior and host network information, such as physical and logical interfaces, and their configuration, as well as network packets (25).

Nevertheless, in order to implement HIDS in ICS devices, further considerations are necessary. Two major challenges have to be faced: the time performance in devices with severe latency requirements and low computational power, and the risk related to the implementation of active HIDS. Attributes that a HIDS suitable for ICS application should include are (26):

- Configurability: capability to be configured as specified by the requirements of the target system;
- Configuration and Knowledge Security: HIDS configuration and used data should be protected against unauthorized access and modifications;
- Resiliency: HIDS action cannot affect the availability of the device;
- Low Performance Overhead: the execution of the HIDS on the target device should not negatively influence the performance of the underlying system;
- Low Detection Time: detection and response to intrusions should be as fast as possible; and
- Interoperability: the HIDS should be able to interact with other technologies, such as Security Information and Event Management (SIEM).

In general, regarding embedded industrial devices, operational requirements for industrial environments, such as real-time capabilities, and availability must be ensured, even in the context of a cybersecurity action. Domain specific standards, guidelines, and recommendations that can be applied for specific industrial sectors must be considered to address this issue.

For example, in an electrical microgrid environment, the most time-critical devices are PLC, Remote Terminal Units (RTU), and, in particular, electrical protections. To give a few examples about electrical protections:

- IEC 60834 requires that the latency of the transmission and reception of a control signal related to a protective action has to be lower than 10 ms, while IEC 61850 imposes a latency lower than 3 ms;
- IEEE 1646-2004 requires information on protective actions to be exchanged by the devices inside the same substation in a time lower than a quarter of a period (i.e., 5 or 4 ms depending on the 50 or 60 Hz frequency); and
- less stringent limits (between 8 and 12 ms) are required for the exchange of information on protective actions with devices outside the substation.

Most PLC and RTU are based on Real-Time Operating Systems (RTOS) (27). The main characteristic of RTOS is the way they handle operations and resources, completing and executing tasks within a defined time frame due to their optimized architecture and features. Multi-tasking is still possible, thanks to task scheduling. RTOS handle priority: each task has a priority, and the task with the highest priority has a preference of execution, even if it is necessary to prevent a lower-priority task from being executed. Real-Time HIDS are sometimes implemented as a kernel module in Linux-based operating systems. This type of implementation can affect the device's performance. For this reason, even if some papers already propose HIDS specifically designed for ICD devices (26), a further effort has to be put forth to verify the applicability of these solutions to electrical devices.

3.3 Anomaly Detection Techniques

Novelty detection is the task of classifying test data that differ in some respect from the data that are available during training. This may be seen as "one-class classification", in which a model is constructed to describe "normal" training data. The novelty detection approach is typically used when the quantity of available "abnormal" data is insufficient to construct explicit models for non-normal classes.

This situation is very common in the field of security monitoring of industrial control systems, at different levels. The network traffic in the manufacturing zone is highly predictable; moreover, researchers do not dispose of high amount of data regarding malwares and attacks within control networks. On the contrary, it is very clear what should be the normal behavior of the whole system. For these reasons, utilizing an anomaly detection algorithm may allow reaching a high accuracy, even in detecting unknown threats. These considerations can be applied not only to the network traffic. Another important application is in the field of physics-based anomaly detection algorithms, that will be discussed in the next section.

According to (28), novelty detection techniques can be classified according to the following five general categories:

- Probabilistic approaches: are based on estimating the generative probability density function (pdf) of the data. The resultant distribution may then be thresholded to define the boundaries of normality in the data space and test whether a test sample comes from the same distribution or not.

- Distance-based methods: including clustering or nearest neighbor methods, are another type of techniques that can be used for performing a task equivalent to that of estimating the pdf of data. These methods rely on well-defined distance metrics to compute the distance (similarity measure) between two data points.
- Reconstruction-based methods: are often used in safety-critical applications for regression or classification purposes. They can autonomously model the underlying data, and when test data are presented to the system, the reconstruction error, defined to be the distance between the test vector and the output of the system, can be related to the novelty score. Neural networks and subspace-based methods can be trained in this way
- Domain-based methods: they require a boundary to be created based on the structure of the training dataset. These methods are typically insensitive to the specific sampling and density of the target class, because they describe the target class boundary, or the domain, and not the class density. Class membership of unknown data is then determined by their location with respect to the boundary. As with two-class Support Vector Machine (SVM), novelty detection SVMs (most commonly termed "one-class SVMs" in the literature) determine the location of the novelty boundary by using only those data that lie closest to it (in the transformed space); i.e., the support vectors. All other data from the training set (those that are not support vectors) are not considered when setting the novelty boundary. Hence, the distribution of data in the training set is not considered which is seen as "not solving a more general problem than is necessary"
- Information theoretic methods compute the information content of a dataset by using measures such as entropy, relative entropy, etc. These methods assume that novelty significantly alters the information content of the otherwise "normal" dataset. Typically, metrics are calculated by using the whole dataset and then that subset of points whose elimination from the dataset induces the biggest difference in the metric is found. This subset is then assumed to consist of novel data.

Each category of methods has its own strengths and weaknesses, and faces different challenges for complex datasets. Reconstruction-based methods are very flexible and typically address high-dimensionality problems, with no a priori assumptions about the properties of the data distribution. However, they require the optimization of a pre-defined number of parameters that define the structure of the model, and may also be very sensitive to these model parameters. A reconstruction-based approach relying on a neural network architecture called autoencoder will be discussed in Section 5.3 for the development of a physics-based anomaly detection algorithm for a photovoltaic system.

3.4 Physics-Based Anomaly Detection

Cyber-attacks against industrial systems aim to modify the physical behavior of the usual system process. In cyber-physical systems, the physical evolution of the system state is predictable. For this reason, some works propose to add a further line of defense in ICS, represented by algorithms able to rapidly notice abnormal physical behaviors based on measures extracted from the industrial process.

One of the fundamentally unique properties of industrial control—when compared to general Information Technology (IT) systems is that the physical evolution of the state of a system has to follow immutable laws of nature. For example, the physical properties of water systems (fluid dynamics) or the power grid (electromagnetics) can be used to create time-series models that we can then use to confirm that the control commands sent to the field were executed correctly and that the information coming from sensors is consistent with the expected behavior of the system. For example, if we open an intake valve, we should expect that the water level in the tank should rise, otherwise, we may have a problem with the control, actuator, or the sensor; this anomaly can be either due to an attack or a faulty device. The idea of creating models of the normal operation of control systems to detect attacks has been presented in an increasing number of publications appearing in security conferences in the past couple of years.

Monitoring the "physics" of cyber-physical systems to detect attacks is a growing area of research. In its basic form, a security monitor creates time-series models of sensor readings for an industrial control system and identifies anomalies in these measurements to identify potentially false control commands or false sensor readings. Applications include water control systems, state estimation in the power grid, boilers in power plants, chemical process control, capturing the physics of active sensors, electricity consumption data from smart meters, video feeds from cameras, medical devices, and other control systems.

Research communities from different backgrounds ranging from control theory, power systems, and cyber-security have tried to provide their own solutions to physics-based attack detection. A complete review of physics-based anomaly detection algorithms has been presented in (29). In the field of industrial processes and especially in power systems, typical anomaly detection strategies are based on the dynamic state estimation, basically composed by using the equations that describe the physical system, and on the comparison between the forecast behavior and the real measurements. Even if very efficient, this approach has some drawbacks: implementing the equations requires knowledge of the exact behavior of the system, i.e., the exact parameters of the equations; moreover, it could be very hard to write a closed-form equation that takes into account heterogeneous types of parameters, and even if possible, it would require a customized design. Machine Learning (ML) approaches could be useful to face up such types of problems.

Chapter 4

Enterprise Layer

4.1 Covert Channels

In computer security, a covert channel is a type of attack that creates a capability to transfer information objects between processes that are not supposed to be allowed to communicate by the computer security policy. The term, originated in 1973 by Butler Lampson, is defined as channels "not intended for information transfer at all, such as the service program's effect on system load," to distinguish it from legitimate channels that are subjected to access controls. In short, covert channels transfer information by using non-standard methods against the system design. Today, covert channels and their technological side – steganography – represent the new frontier of cyber-crime and cyber-espionage.

The ubiquitous presence of a small number of network protocols suitable as carriers (e.g. the Internet Protocol) makes covert channels widely available. Reference (30) identified in 2007 a list of possible covert channels techniques, based on the mechanism and not on the layers of the Open Systems Interconnection (OSI) model:

- Unused Header Bits: example include Type of Service (TOS) field or the TCP header's Flags field, the IP header's Don't Fragment (DF) bit (can be set to arbitrary values if the sender knows the Maximum Transfer Unit (MTU) size of the path to the receiver), the TCP Urgent Pointer (used to indicate high priority data that is unused if the URG bit is not set), the TCP Reset segments (TCP segments with the RST flag set abort the connection and usually contain no data), the unused code fields of some Internet Control Message Protocol (ICMP) messages and various IPv6 header fields such as Traffic Class and Flow Label

- Header Extensions and Paddings: examples include IPv6 destination options header, IPv6 Hop-by-Hop, Routing, Fragment, Authentication and Encapsulating Security Payload extension headers, IP Route Record option headers, padding of the IP and TCP header to 4-byte boundaries

- IP Identification and Fragment Offset: the IP Identification header field is used for reassembling fragmented IP packets. The only requirement from the IP standard is that each IP ID uniquely identifies an IP packet for a certain time period. The Fragment Offset is used to determine in which sequence the fragments need to be reassembled. Several techniques can be used, like multiplying each byte of the covert information by 256 and directly using it as the IP ID

- TCP Initial Sequence Number field: TCP sequence numbers are used to coordinate which data has been transmitted and received guaranteeing reliable transport. The first sequence number selected by the client is called the Initial Sequence Number (ISN). The ISN must be chosen such that the sequence numbers of new incarnations of a TCP connection do not overlap with the sequence numbers of earlier incarnations of a TCP connection. Several techniques can be used, like multiplying each covert byte with 256^3 and directly using it as the TCP ISN

- Checksum Field: The Checksum field is modified to encode the secret information and an IP header extension is added with the content chosen such that the modified checksum is correct again. The same technique could be used for the TCP header checksum.

- Modulating the IP Time To Live Field, or Address Fields and Packet Lenghts, or Modulating Timestamp Fields.

- Packet Rate/Timing: Covert information can be encoded by varying packet rates, which is equivalent to modulating the packet timing (the interpacket times)

- Message Sequence Timing: the possibility of constructing covert channels by modulating the use of protocol operations

- Packet Loss and Packet Sorting: the technique requires per packet sequence numbers and erasures are realized by skipping sequence numbers (artificially losing packets at the sender)

- Frame Collisions: exploiting the Carrier Sense Multiple Access Collision Detection mechanism, the covert sender jams any packets of another user, then it uses a back-off delay of either zero or the maximum value.
- Ad-Hoc Routing Protocols: Covert information can be encoded in header fields present in Dynamic Source Routing protocol.
- Wireless LAN: for example, embedding covert data in the RC4 initialisation vector, which is part of the IEEE 802.11 Wired Equivalent Privacy (WEP) mechanism.
- HTTP: an observer who cannot look into the content transported by HTTP cannot distinguish between harmless web surfers and the covert senders/receivers.
- DNS: like for HTTP, an observer who cannot look into the content transported by DNS cannot distinguish between harmless web surfers and the covert senders/receivers.
- Other Application Protocols: many different application protocols present possibilities to hide information.
- Payload Tunneling: Payload tunnels are covert channels that tunnel one protocol (usually the IP protocol) in the payload of another protocol. This can be particularly hard to detect in case of encrypted communication.

Firewalls may be configured in order to block all the unusual protocols or connections to unusual ports. Nevertheless, covert channels can exploit protocols that are usually allowed, like DNS or HTTP. In these cases, it is necessary that the firewall inspects also the traffic related to allowed connections. One protocol that the vast majority of firewalls don't inspect is DNS. In the next chapters, we will focus on covert channels based on DNS, presenting the attack model and a novel algorithm for the identification of the attack.

4.2 The DNS Tunneling Attack

DNS is a hierarchical and decentralised naming system whose main aim is to associate more readily memorised domain names, called URL, to the numerical IP addresses needed for locating devices, such as computers and servers, running services, such as web mailing and cloud storage. DNS concepts and specifications have been defined in the two documents RFC 1034 (31) and RFC 1035 (32). In the rest of this work, the couples of terms DNS server - server, DNS clients - clients, DNS request - request, DNS response - response, DNS query - query, DNS answer - answer, DNS channel - channel, and DNS tunnel - tunnel are intercharged for simplicity.

DNS is a client-server application: the entities that know the association between domain names and IP addresses are called DNS servers and the ones that require this information are called DNS clients. A host that needs to map an IP address to a domain name, or vice versa, directly sends a mapping request message, called DNS request, to a known DNS server. If the server has the required information, it satisfies the client sending a response message called DNS response. In this case, the server is called authoritative DNS server for the required domain name. Otherwise, the server can act in two different ways:

- Recursive: the server forwards the request to another server and waits for the response. If this contacted server does not have the required information, it contacts another server, and again until the request reaches the authoritative server for that domain. The response travels back from server to server until it finally reaches the requesting client (Fig. 4.1).

- Iterative: the server sends to the requesting client the IP address of the server that it assumes can resolve the request. The client repeats the request to this second server, and, if necessary, again until it asks to the correct server, i.e. the authoritative one. The response is sent from the authoritative server directly to the requesting client (Fig. 4.2).

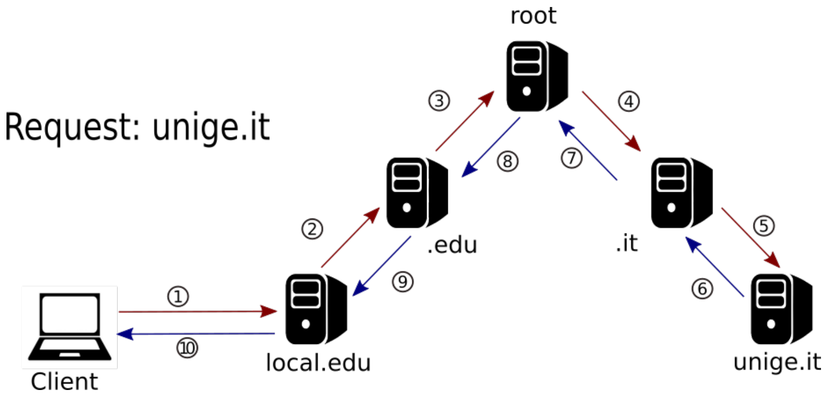

Figure 4.1 DNS Recursive Resolution

The mechanism described above can be exploited to create a covert communication within a covert DNS channel, also called DNS tunnel, between a client and a server. The covert channels exploit DNS requests and responses in order to bypass firewalls that do not implement DNS packet inspection. This can be achieved by compromising a DNS client inside a local network usually protected by a firewall and employing a malicious (rogue)

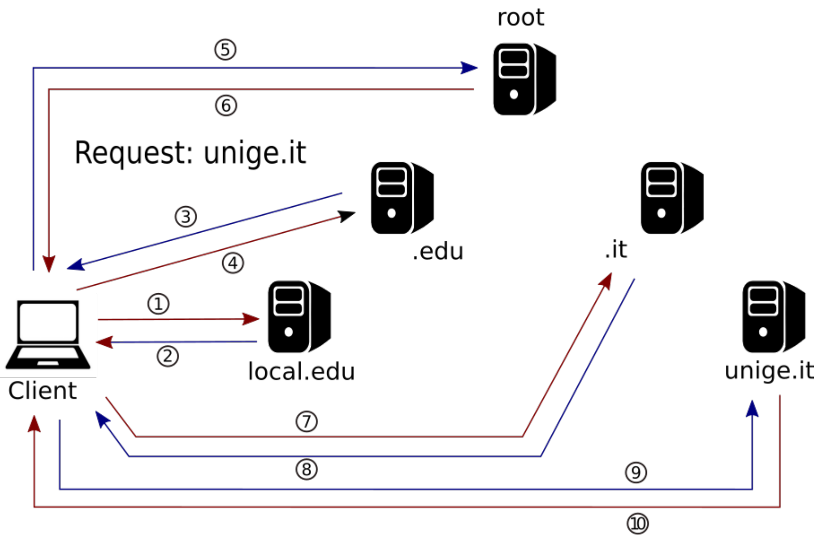

Figure 4.2 DNS Iterative Resolution

DNS server. In particular, DNS request and response packets, called DNS query and DNS reply, respectively, can be used for two main malicious purposes: i) provide an Internet connection outside of a delimited network bypassing the firewall. In this way, data from the compromised client can be encapsulated within DNS packets sent to the rough server. This kind of attack is called data exfiltration. ii) create Command and Control channels for malware, in particular, botnet. It is the DNS client the one that starts the communication. DNS server cannot do that because clients do not have a service listening for DNS requests and, most times, are behind a firewall that blocks these requests from outside. A DNS channel is activated when a client receives the response to its previously sent request.

We can identify two main scenarios about the communication between the compromised client and the rogue server:

1. Direct: the client is able to set its own server address, e.g., in the operative system's settings, and so to directly create a covert channel between the client and the rogue server. The compromised client will send all its requests to the rogue server. Typically, this configuration is fruitless since the firewall usually blocks all outgoing direct connections to port 53 (the port used for DNS packets exchange).

2. Proxy: the attacker registers a fake domain and deploys, outside the client local network, a rogue server that is authoritative for that specific domain. The compromised client will send all its requests to the nearest genuine server which will forward to the rogue server only the requests to the fake domain, as shown in Figure 4.3.

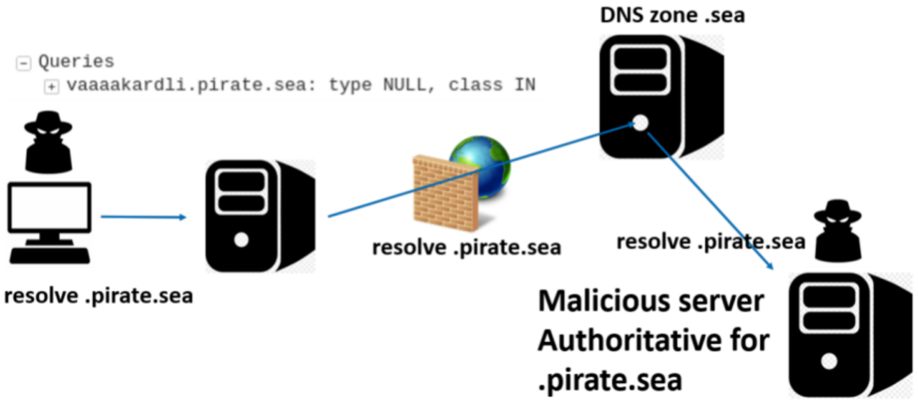

Figure 4.3 DNS tunnelling proxy attack model

There are different tools, available on the Internet, that can be used to open DNS tunnels. They can be classified depending on the abstraction layer at which the information is encapsulated. Some tools just tunnel binary data that can be used to issue Operating System commands and transfer files, while others encapsulate another protocol over DNS, such as IP or TCP, as shown in Table 4.1.

Table 4.1 Examples of tools that create DNS tunnels

Abstraction layer	Tools
Binary data	Reverse_DNS_Shell
IP over DNS	NSTX, DNSCat, Iodine, TUNS
TCP over DNS	DNS2TCP, OzimanDNS

Besides the simple tools, a lot of malware have been created to open and exploit DNS tunnels for different malicious purposes. Nowadays, the main ones are (33) (34):

- Morto Worm;
- FeederBot;
- PlugX;
- FrameworkPOS;
- Wekby;
- BernhardPOS;
- Jaku;

- Multigrain;
- DNSMessenger;

Even if they work in different ways, their common goal is to create DNS tunnels for covert command and control or data exfiltration communications.

4.3 Ensemble Classifier for Detecting DNS tunneling Attack

Several works in the literature address the issue of DNS-based covert channels. The present work proposes a classification into four big families depending on the set of information considered to extract the statistical features: Per-transaction, Per-query, Per-domain, and Per-IP approaches.

Per-transaction approaches try to discover covert channels between couples client/server entities by analysing properties related to the request/response transactions. These approaches use as input each pair of request/response grouped by client/server IP address and identified by transaction ID: $X(client_i, server_j) \leftarrow transID_1, \ldots, transID_n$. They extract arrival timing information and/or some information from both query and reply fields. The extracted features allow identifying a compromised $(client_i, server_j)$ DNS communication.

The main papers describing Per-transaction approaches and the considered features are reported in Table 4.2.

One of the first and most promising works is proposed in (35). The authors consider the inter-arrival time between DNS packets, query length, response length, and the related statistics up to the 4^{th} order (mean, variance, skewness, and kurtosis) as the features to detect the presence of active DNS tunnels. Subsequently, some binary classifiers are compared to identify malicious activities. The work has been subsequently expanded by using PCA and MI on the same features in (36) and (37), and a further performance evaluation with unsupervised ML algorithm is presented in (38). In (39), a set of 10 features is used as input to an ensemble classifier composed of three supervised binary classifiers (K-nn, Random forest, and multi-layer perceptron).

Per-query approaches try to discover tunnels analysing FQDN queries regardless of client/server interactions. These approaches consider in input some fields of every single DNS query and/or reply. The extracted features are mainly based on character space properties of the query/reply fields and allow identifying each compromised DNS packet. The main papers describing Per-query approaches and the considered features are reported in Table 4.3.

Table 4.2 Main papers investigating Per-transaction detection approaches

Papers	Features	Algorithms
(35)	inter-arrival time between DNS packets, query length, response length	linear discriminant analysis, K-nn, NN, SVM
(36), (37)	inter-arrival time between DNS packets, query length, response length	a PCA + MI
(38)	inter-arrival time between DNS packets, query length, response length	k-means, logic learning machine
(39)	query Question type, query Question length, query Question info bit, query Question entropy, response Answer length, response answer info bits, response info bit, response entropy	Ensemble classifiers (K-nn, Random forest, multi-layer perceptron)

There is a huge variety of features that can be extracted from a single query, including the ratio of vowels, consonants, numbers, special characters, but also the computation of parameters like entropy. Many works utilize a lexicographic approach for detecting anomalous queries. For example, in (40) many different features are extracted from the queries and subsequently analyzed by the one class Isolation Forest algorithm. The main drawback of this approach is that many websites utilize random strings, so that many queries are classified as false positive for the design itself of the algorithms.

Some papers propose to exploit some deep learning architectures to process the entire packets at a byte level, such as in (44), where a convolutional neural network is used to process a representation of the entire DNS packet in a supervised ML-based classification. The main paper following this approach is reported in Table 4.4

Per-domain approaches collect all the DNS packets that are sent to a specific second-level domain and compute the features over sets of these packets. Each set includes a fixed amount of packets or all the packets collected within a fixed time interval. The extracted features allow identifying a compromised $domain_i$. The main papers describing Per-Domain approaches and the considered features are reported in Table 4.5.

Table 4.3 Main papers investigating Per-query detection approaches

Papers	**Features**	**Algorithms**
(40)	character entropy, total count of characters, count of characters in sub-domain, count of uppercase and numeric characters, number of labels, maximum label length and average label length	Isolation Forest
(41)	entropy, query length, IP packet sender length, IP packet response length, encoded query name length, request application layer entropy, IP packet entropy, query name entropy	SVM, J48, Naive Bayes
(42)	entropy, length, characters ratio, upper case ratio, lower case ratio, digit ratio, number of subdomains, Responses: TXT records, upper case count, lower case count, number of digits, number of spaces, dash count, slash count, equal count, other characters count, normalized entropy	Logistic Regression, K-means clustering
(43)	Longest Meaningful Characters Ratio, N-gram Score, Entropy of Subdomain Names, Numerical Characters Ratio, Different Alphabetic Characters Ratio, Different Numerical Characters Ratio, Length of Subdomain Names, Vowel Characters Ratio, Number of Alphanumeric Swaps	PCA + ensemble binary classifier

Table 4.4 Main papers proposing a per-query featureless approach

Papers	Algorithms
(45)	1D convolutional neural network
(44)	convolutional neural network
(46)	feed-forward deep learning
(47)	Autoencoder based anomaly detection

Table 4.5 Main papers investigating Per-domain detection approaches

Papers	Features	Algorithms
(34)	character entropy, rate of A and AAAA records, non-IP type ratio, unique query ratio and volume, average query length, ratio between the length of the longest meaningful word and the subdomain length	Isolation Forest
(48)	nameservers, domains and lowest level subdomains character frequencies	compare the character ranks and frequencies with Zipfian distribution of the English language
(49)	29 feature, including statistics over subdomains and record types	Isolation Forest

Per domain approaches perform differently over specific attacks from the ones based on per-query or per-transaction strategy. For example, authors in (34) declare their solution addresses the problem of detecting low throughput DNS tunneling attacks. Per-domain approaches can also be useful to detect botnets that utilize DNS tunnels for command and control.

Per-IP approaches consider all the packets that are sent by a specific IP address. The extracted features are mainly based on timings and allow identifying a compromised $client_i$. The main papers describing Per-IP approaches and the considered features are reported in Table 4.6. In particular, authors in (50) propose a method based on 4 elements (time interval, packet size, subdomain entropy, and record types) for an overall amount of 18 features extracted from 1000 request/response pairs that share the same source IP address, destination IP address, and intended query domain, used as input to a binary classification ML algorithm.

Table 4.6 Main papers investigating Per-IP detection approaches

Papers	**Features**	**Algorithms**
(50)	4 elements (time interval, packet size, subdomain entropy and record types) for an overall amount of 18 features	binary classification ML
(51)	Average length of domain names, Average number of labels, Number of different hostnames, Average length of hostnames, Information entropy of hostnames, Average length of DNS messages, Proportion of big upstream packets, Proportion of small downstream packets, Upload/Download payload ratio	Decision Tree, Random Forest, K-nn, and SVM

Some other papers worth mentioning propose approaches that do not well fit in one of the four mentioned categories. For example, (52) addresses the issue of covert channels over encrypted traffic, that are exploited especially in some botnets, proposing to detect DGA via the amount and rate of NXDomain responses.

These papers are reported in Table 4.7.

Table 4.7 Other approaches for the detection of DNS tunnels

Papers	Features	Algorithms
(53)	number of answer, character ratio (letter/total, used character), inter arrival per domain (request and answer), number of substring, longest substring, packet length, DNS type, DNS type frequency,...	Random Forest

4.3.1 Proposed Approach

Each approach proposed in the literature shows a significant efficiency in detecting some specific tools or malware. Nevertheless, none of these works is able to assure a high efficiency in detecting all the possible types of covert channels.

The present work proposes an architecture that aims to analyze the DNS traffic in order to detect different forms of covert channels jointly exploiting the detecting capabilities of different solutions proposed in the literature. We also want to structure the solution in a modular and scalable way to be able to add new algorithms that could be proposed in the future for addressing some specific new threats. The architecture of the proposed approach is shown in Figure 4.4.

The basic idea is to run in parallel different solutions based on different ML algorithms and a different set of features, where each of them specifically targets different kinds of DNS tunneling attacks.

The main blocks are:

- **DNS filter**: it filters the DNS traffic which will be the only one considered in the DNS tunneling detection process.
- **Feature Extractor**: it periodically extracts statistical features from the input DNS traffic.
- **Feature Selector**: each family includes a feature selection block that aims to select a set of features from the ones extracted by the Feature Extraction block depending on the ML algorithm's needs.
- **ML algorithm**: each algorithm takes a "tunnel" or "no tunnel" decision concerning the portion of the DNS traffic related to the input features.

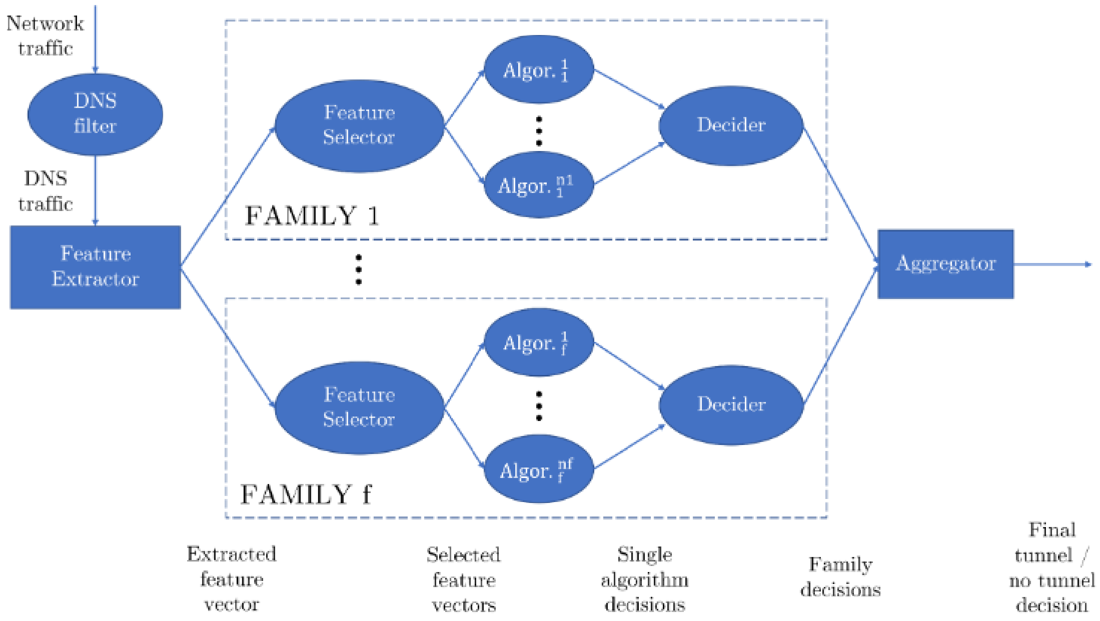

Figure 4.4 Block diagram of the proposed solution

- **Family Decider**: it merges together the decisions taken by all the ML algorithms belonging to the same family following different possible strategies in order to take a unique family decision.
- **Aggregator**: aggregate the family decisions in a final decision related to the analyzed DNS traffic portion.

This modular approach considers different algorithms of different families in order to maximize the detection rate and minimize the false positive rate. Each family takes into account a different subset of the extracted features depending on the implemented ML algorithms. Within each family, the outputs of the different algorithms are compared in order to make a family decision. Family decisions are then compared and weighted together to make a final decision. For example, considering the Per-query family, different solutions can be applied in order to analyze a single DNS query including a huge variety of features and ML algorithms. Each of the considered solutions gives as output a binary decision with an accuracy score for each single query. Family decision blocks can be based on different strategies that can be the same or not for all families. Also, the aggregator can be based on different and customizable logics.

An important aspect to highlight is that the decision process is a continuous process that goes over time on each DNS traffic flow from the first statistical feature extraction to the flow end. Features are extracted periodically on subsets of single or multiple exchanged DNS packets. However, each family can proceed with the feature selection phase, and the consequent decision making phase, with a different periodicity than the other families. Besides, even ML algorithms within the same family can make decisions asynchronously with the other algorithms. Figure 4.5 shows an example of a possible situation involving different algorithms belonging to the same family but with different decision periodicities.

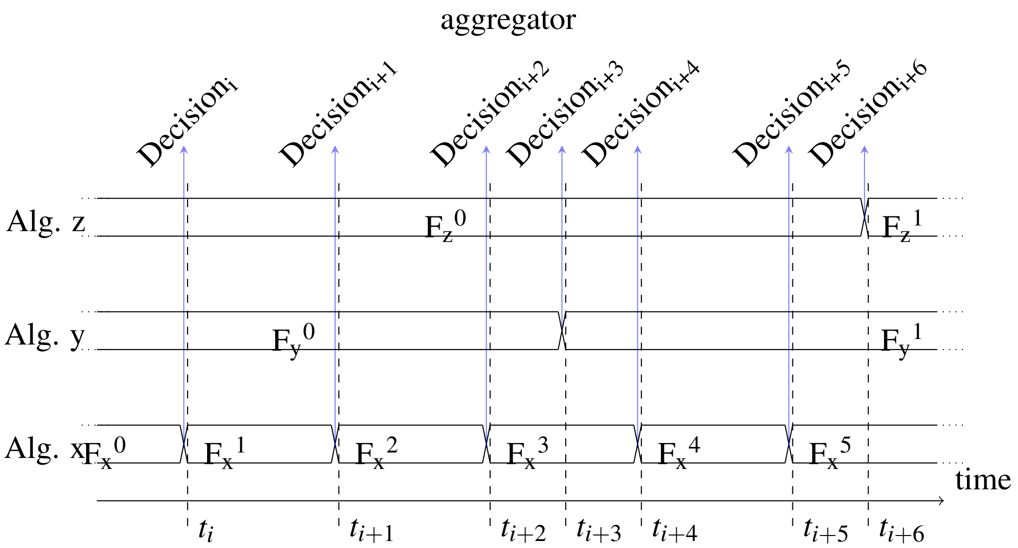

Figure 4.5 Example of a possible situation within one family: three algorithms with a different decision periodicity

Every time an algorithm Alg. x receives in input new values of the related subset of features, e.g. $F_x{}^1$, it updates its previous decision, if any. The decision block so consequently recomputes the related family decision Decision$_i$ every time even only one of the algorithm decisions has been updated.

The final stage of the architecture is an aggregator of the family decisions. It could be designed to be a simple HMI that shows the family decisions to a human operator. In this case, this HMI could be a very precious support to a human operator to help him/her recognise the source of the threat while it is still ongoing and allow him/her taking the proper actions in time. Otherwise, it could be designed to merge together decisions made on different elements (transactions, queries, domains, or IPs) aim to signal tunnel presence and possibly to infer which is the common reason, e.g., multiple transactions with the same IP address as client or server or multiple queries to the same domain.

4.3.2 Developed Testbed

In order to collect network traffic that contain a covert channel based on DNS, we developed a testbed based on two Virtual Machines (VM). Each VM has two network interfaces: one is linked to an internal network that connects the two machine, which possess a static private IP; the other is a NAT interface that allow the VM to access the internet through the host machine.

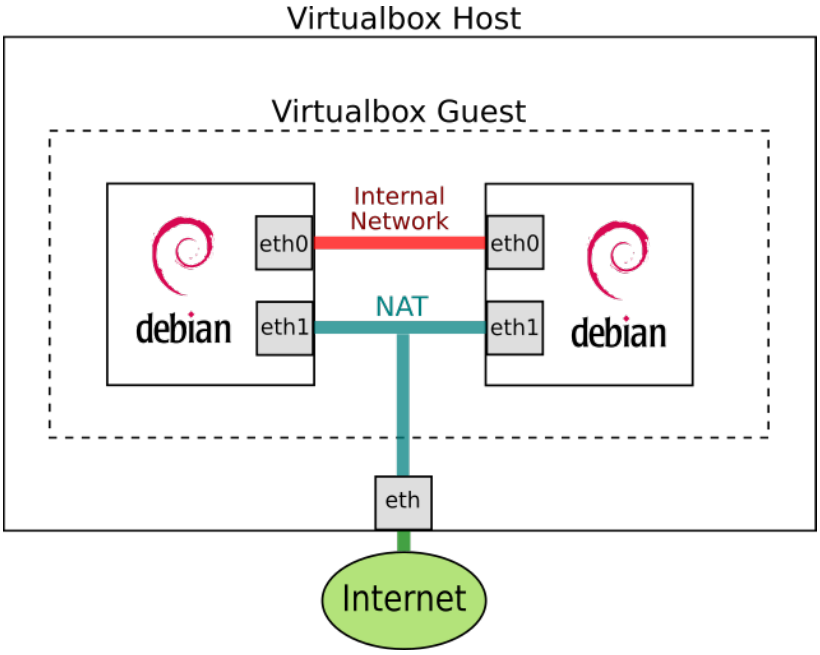

Figure 4.6 Testbed with one client and one server machines

We generated traffic traces affected by DNS tunneling attacks by using the following tool:

- **Iodine** (54): it is a tool written in C that allows tunneling IPv4 data through a DNS server. The tunnel is established by using a proxy connection. It, therefore, requires that the attacker controls a real domain, such as mydomain.com, and a rogue DNS server to run Iodine on with a public IP address to run Iodine on and delegate a subdomain (such as t1.mydomain.com) to the iodined rogue DNS server.

- **DNScat2** (55): it has been designed to create encrypted command and control (C2) channels over the DNS protocol. The client is written in C and the server in Ruby and they are both open-source. A DNS query generated by the original version of this tool can be easily recognized since it adds the word "dnscat2" at the beginning of each query. In our tests, we removed the domain prefix "dnscat2" to make the packets less easy to identify.

- **ReverseDNSshell** (56): it has been designed to create C2 channels over the DNS protocol. It is written in Python and is open-source. The DNS tunnel is used to send commands to be executed on the client. The client sends queries to the rogue server asking for a specific DNS TXT record. The rogue DNS responds with a DNS TXT record of base64 encoded commands. The client executes these commands and sends back the output by using a DNS A record query.

- **OzymanDNS** (57): it is a tool written in Perl that allows tunnelling IPv4 data through a DNS server. The tunnel is established by using a proxy connection.

- **Tcp_Over_DNS** (58): it is a tool written in Java that allows building a DNS tunnel to establish a hidden connection between two specific TCP sockets. All connections to the Tcp_Over_DNS client listening socket are forwarded through the tunnel to reach the forwarded socket port. For example, a hidden SSH connection can be established between client and server.

We chose these tools in order to test the proposed solution considering different kinds of DNS tunneling attacks that work in different ways and encapsulate different contents over the DNS packets.

Then, to assess the proposed approach, we built a testbed that implements all the blocks described in the previous Section. The code of all blocks is written in Python and the ML algorithms have been implemented by using the SciKit Learn library (59).

The developed software is able to perform an online analysis collecting traffic from the network in real-time and/or work with previously collected DNS traffic traces. We used as input traffic traces (pcap files) generated by ourselves respecting the proper starting and inter-arrival times between flows and between packets of the same flow in order to simulate as close as possible an online analysis. In detail, on one hand, we generated "genuine" DNS traffic traces, i.e., not affected by DNS tunnels, collecting DNS traffic from our University laboratory over one month. We also explicitly collected DNS traffic from special domains that could easily lead to false positive classification, such as the cloudflare domain which uses random strings to generate its subdomains.

Then, our testbed implements four approaches from four different papers among the ones mentioned. They belong to different families, are based on different ML algorithms, and

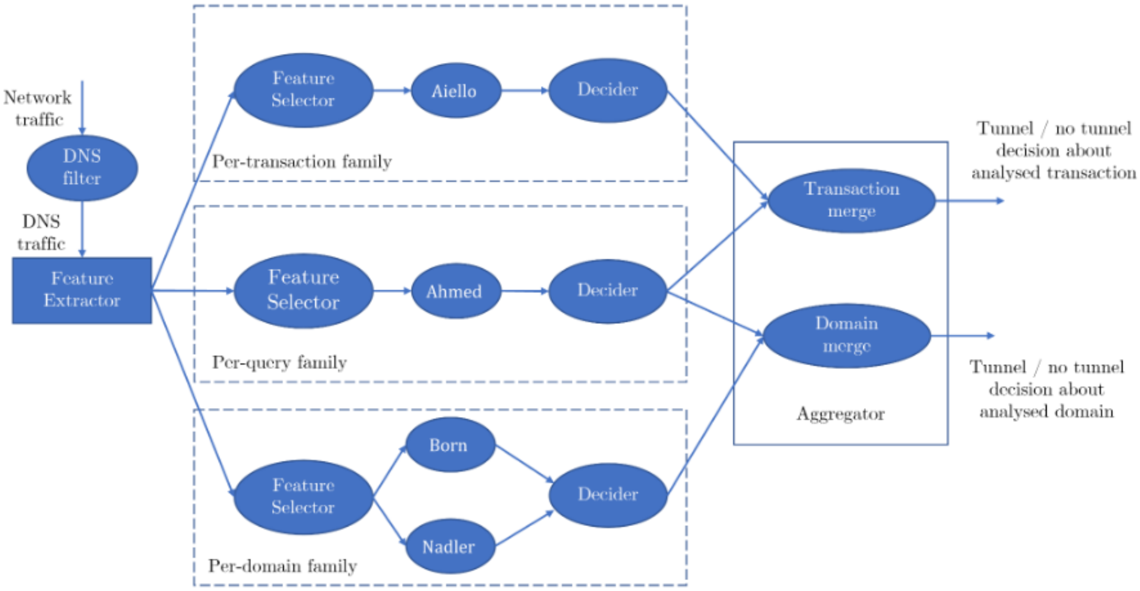

Figure 4.7 Block diagram of the implemented testbed architecture

consider different features. Since the authors did not give them an explicit name, we are going to call them with the last name of the first author:

1. **Aiello** (35): Per-transaction approach based on different ML algorithms and tested over several well-known tools, including Iodine and Dns2Tcp.

2. **Ahmed** (40): Per-Query approach based on an anomaly detection algorithm (Isolation Forest) and tested over Iodine and an open-source DNS Exfiltration Toolkit.

3. **Born** (48): Per-Domain approach which utilizes the character frequency analysis and, for this reason, it may fail in classifying some specific domains.

4. **Nadler** (34): Per-domain approach based on the Isolation forest algorithm considering as input a set of features which includes frequencies of specific request types. Authors declare that this strategy can be useful to detect low-throughput tools and botnets.

Aiello and Born approaches use supervised ML algorithms that require a training dataset containing a proper number of labelled samples belonging to both genuine and malware classes. Ahmed and Nadler approaches use anomaly detection algorithms that require an unlabelled dataset only composed of genuine traffic samples. Each of these approaches is individually trained. The datasets used for the training and testing phases have been composed of a different mix of genuine and tunneled DNS traffic. For the family decision blocks, we alternatively use three decision strategies:

- **Majority**: if more than half of the algorithms make a "tunnel" decision, the decision is "tunnel"; otherwise, the decision is "no tunnel".
- **Score voting**: a score is assigned to each algorithm decision depending on the result accuracy. If the sum of the "tunnel" decisions' score is greater than the sum of the "no tunnel" decisions' score, the family decision is "tunnel"; otherwise, the decision is "no tunnel".
- **Evil win**: if at least one algorithm makes a "tunnel" decision, the family decision is "tunnel"; otherwise, the decision is "no tunnel".

Finally, the aggregator merges the three families decisions in order to understand if they can be correlated. In detail, we correlate together decisions made by the per-transaction family with the per-query family taken from the same input information, i.e., decisions based, on one hand, on features extracted from a transaction and, on the other hand, based on features extracted from the query of that transaction. We also do the same correlating the decision of the per-domain family with the per-query family, in order to understand if a specific domain can be identified as malevolent and so blocked, for now on, with more drastic and less time-consuming methods (e.g. a black list). The two merge blocks with the aggregator in Figure 4.7 are based on the same strategies of the family decision blocks.

4.3.3 Results

In order to validate the proposed approach, we tested the algorithm over different datasets. Below are reported the performances. Tables from 4.11 to 4.10 reports the confusion matrix of the single considered approaches tested with our made test set, which is composed of 116,746 DNS packets: 100,000 genuine and 16,746 affected by tunnels generated with the considered tools (5,000 DNScat2, 260 Iodine, 5,000 ReverseDNSshell, 5,000 OzymanDNS, and 1,486 Tcp_Over_DNS). Tables from 4.8 to 4.13 report the results obtained by using the three strategies described above in both family decision and merge blocks (all these blocks always use the same strategy).

Table 4.8 Confusion Matrix - Aiello approach

		Predicted	
		Positive	Negative
Real	Positive	25 %	12.50 %
	Negative	0 %	62.50 %

Table 4.9 Confusion Matrix - Ahmed approach

		Predicted	
		Positive	Negative
Real	Positive	22.22 %	11.11 %
	Negative	0 %	66.67 %

Table 4.10 Confusion Matrix - Born approach

		Predicted	
		Positive	Negative
Real	Positive	6.25 %	0 %
	Negative	4.17 %	89.58 %

Table 4.11 Confusion Matrix - Nadler approach

		Predicted	
		Positive	Negative
Real	Positive	6.67 %	0 %
	Negative	17.78 %	75.56 %

Table 4.12 Confusion Matrix - Our approach, Majority strategy

		Predicted	
		Positive	Negative
Real	Positive	8.77 %	1.75 %
	Negative	1.75 %	87.72 %

Table 4.13 Confusion Matrix - Our approach, Score voting strategy

		Predicted	
		Positive	Negative
Real	Positive	8.77 %	1.75 %
	Negative	1.75 %	87.72 %

Table 4.14 Confusion Matrix - Our approach, Evil win strategy

		Predicted	
		Positive	Negative
Real	Positive	8.77 %	1.75 %
	Negative	19.30 %	70.18 %

Additional results have also been obtained in terms of three metrics commonly used in ML, i.e. Accuracy, Sensitivity, and Specificity, which are defined as in Equations (6.1), (6.2), and (6.3), respectively:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{4.1}$$

$$Sensitivity = \frac{TP}{TP+FN} \tag{4.2}$$

$$Specificity = \frac{TN}{TN+FP} \tag{4.3}$$

where TP: True Positive, TN: True Negative, FP: False Positive, and FN: False Negative.

The results obtained from both the single approaches and our proposed approach are reported in Table 4.15.

Table 4.15 Comparison among the single considered approaches and our proposed approach with three different decision strategies

Algorithm	**Accuracy**	**Sensitivity**	**Specificity**
Nadler	82.23 %	100 %	80.95 %
Aiello	87.5 %	66.66 %	100 %
Ahmed	88.88 %	66.67 %	100 %
Born	95.83	100 %	95.55 %
Majority	96.49 %	83.36 %	98.04 %
Score voting	96.49 %	83.36 %	98.04 %
Evil win	78.98 %	83.37 %	78.43 %

We can notice that our proposed approach generally improves the obtained performance compared to each single considered approach, even if it could not be so looking at only one of the considered metrics. For example, Nadler obtained a higher Sensitivity (100%) than the best case with our approach (majority strategy, 83.36%), but lower Accuracy (82.23% against 96.49%) and lower Specificity (80.95% against 98.04%). The best performance has been obtained by using the Majority and Score voting strategies, even if the Evil win strategy could be useful in case we want to reduce as much as possible the security risk. We noticed that each approach perform differently in detecting DNS tunnels generated by different tools.

Moreover, each algorithm produces a different number of false positive analyzing some special domains.

4.3.4 Discussion and Future Developments

The results presented in the previous section validate the hypothesis of improving the state of the art by implementing an ensemble classifier, based on the coordination of algorithms exploiting different strategies, that we classified in four main families. Results will be hopefully published soon.

We hypothesize two main strategies for further improving the performances. Each algorithm can be set in order to tolerate a higher number of false positives or false negatives without significantly decreasing the accuracy in a reasonable range. Nevertheless, given that the overall performances depend on a score voting of the output of each algorithm, the settings of each algorithm should be coordinated. This is a quite complicated procedure, since each algorithm requires a separate training phase. A strategy for coordinating the setting of the algorithms may produce significant improvements. The second strategy is based on the observation that false positives are generated by a very small amount of domains. For this reason, a simple whitelist of domains, which ensures that domains are not analyzed by the architecture but directly classified as good, may significantly reduce the number of false positives, improving the performance of the proposed solution. As a future development, we are working to increase the number of implemented approaches in order to analyze if the higher the number, the better or, at a certain point, the high number of approaches would lead to a too high complexity compared with the obtained performance improvement (if any). Moreover, we plan to investigate also the detection time, i.e., the time that the approach needs to detect an ongoing malicious attack, considering this as a parameter of primary importance in this field.

Chapter 5

The Field Layer

5.1 The Smart Grid

From its birth up to nowadays, energy generation, transmission, and distribution infrastructure evolved through many steps involving many changes and improvements. Power generation is changing paradigm, from a completely centralized structure to a distributed one. The traditional electric grid structure is unable to meet new requirements such as the need of more efficient transmission means and automated fault and risk analysis, and challenges related to the integration of renewable resources. The Smart Grid (SG) concept was developed to meet the aforementioned requirements and challenges. The evolution of the electrical grid towards a SG involves many aspects and requires many changes to both the current network architecture and the control functions. The main idea was to employ information and communication technologies in the electrical grid in order to improve efficiency, sustainability and reliability.

Of course, this introduced cyber security issues in the power sector. It is almost impossible to provide a complete taxonomy of vulnerability and threats of the electrical power system, since it involves heterogeneous types of control networks, each one with its own peculiarity. Moreover, these technologies are still evolving, actually broadening the attack surface of power systems. Nevertheless, we can identify some applications which result particularly critical. The National Electric Sector Cybersecurity Organization Resource (NESCOR) identifies six scenarios in the power system where main failures related to cybersecurity threats can happen (60): Advanced Metering Infrastructure, Distributed Energy Resources (DER), Wide Area Monitoring Protection and Control, Electric Transportation, Demand Response, and Distribution Grid Management.

Actually, these systems present a variety of architectures, which differ in terms of vulnerabilities and risks. For example, for Advanced Metering Infrastructure, a large variety of architectures, based on wired or wireless communication channels, including Low Power Wide Area protocols can be used (61). In the next sections, this work focuses on Distributed Energy Resources and Microgrids.

5.2 Microgrids and Distributed Energy Resources

With the term Distributed Energy Resources (DERs) we usually refer to a small or medium-scale unit of power generation which is connected to a larger power grid at the distribution level. DERs can be represented by generators that have different primary energy sources such as sun, wind, water movement, traditional thermal cycles and so on, but share electrical and power electronic technologies in order to convert and inject power to the main electrical grid. Renewable Energy Sources-based DERs are uncontrollable or partially controllable sources of energy. This feature implies the need for strong communication and coordination among the sources. DERs are automated systems often connected by a telecommunication network to a remote-control system, for example a Supervisory Control and Data Acquisition system. In this case telecommunication networks can be composed by multi-layer architectures, which use industrial protocols. Reference (2) describes five levels of DER system architectures, as shown in Figure 5.1:

1. Autonomous cyber-physical DER systems: is the lowest level and includes the cyber-physical DER systems. These DER systems will be interconnected to the utility grid and will usually operate autonomously according to pre-established settings. These DER systems will be running based on local conditions, such as photovoltaic systems operating when the sun is shining, wind turbines operating when the wind is blowing, electric vehicles charging when plugged in by the owner, and diesel generators operating when started up by the customer.

2. Facilities DER Energy Management Systems (FDEMS): is the next higher level in which a facility DER management system (FDEMS) manages the operation of the Level 1 DER systems. This FDEMS may be managing the DER systems in a residential home, but more likely will be managing DER systems in commercial and industrial sites, such as university campuses, shopping malls, virtual power plants, and industrial combined heat and power (CHP) installations. Utilities may also use a FDEMS to handle DER systems located at utility sites such as substations or physical power plant

sites. The settings for autonomous DER operations are modifiable by FDEMS operator preferences in coordination with utilities and REPs..

3. Information and Communications Technologies (ICT) for Utility and Retail Energy Providers (REP): extends beyond the local site to allow utilities and possibly REPs to request or require DER systems (typically through a FDEMS) to take specific actions. The settings for autonomous DER operations are modifiable by utilities and REPs. Controls include turning on or off devices, setting or limiting output, providing ancillary services (e.g. volt-var control), and other grid management functions. These requests can be automated and price-based for greater power system efficiency while commands are more likely to be safety or power system reliability related. The combination of this level and level 2 may have varying scenarios, while still fundamentally providing the same services.

4. Distribution Utility DER Operational Analysis (DERMS): applies to utility applications that are needed to determine which requests or commands should be issued to specific DER systems. Utilities monitor the power system and assess if efficiency, reliability, or market advantage can be improved by having DER systems modify their operation. This utility assessment involves many utility control center systems as well as the DERMS. Once the utility has determined that modified requests or commands should be issued, they will be sent as per Level 3.

5. Interactions with Independent System Operators/Regional Transmission Organizations (ISOs/RTOs) and the energy markets: is the highest level, and involves the larger utility environment. RTOs or ISOs may need to exchange information about the capabilities and operational status of larger DER systems and/or aggregated DER systems.

A complete analysis of the vulnerabilities and related risks of all the levels is reported in the document referenced above.

A particularly interesting scenario, which can be assimilated to level two of the previously described architecture, is the microgrid. Microgrids can be defined as small-scale, low, or medium voltage power systems with a decentralized group of electricity sources and loads, which can operate connected to or separated (“islanded”) from the main power network. To ensure proper control, microgrids often make large use of Information and Communication Technologies. With the term Smart Microgrids, we refer to microgrids that are based on networked control systems. The control network of smart grids cannot, in general, be considered as an isolated network: the control network is commonly connected

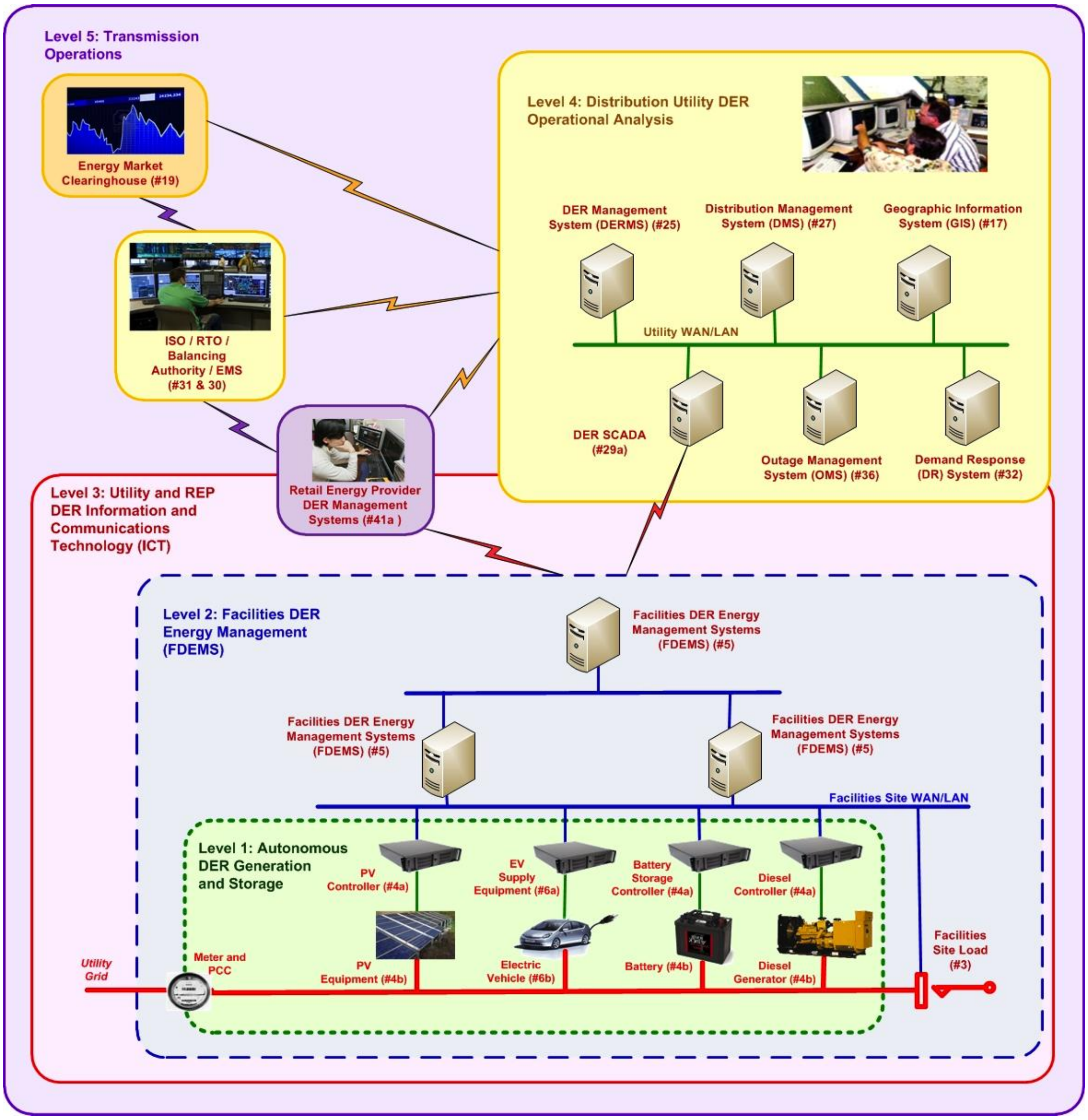
Level 5: Transmission Operations
Energy Market Clearinghouse (#19)
ISO / RTO / Balancing Authority / EMS (#31 & 30)
Level 4: Distribution Utility DER Operational Analysis
DER Management System (DERMS) (#25)
Distribution Management System (DMS) (#27)
Geographic Information System (GIS) (#17)
Utility WAN/LAN
DER SCADA (#29a)
Outage Management System (OMS) (#36)
Demand Response (DR) System (#32)
Retail Energy Provider DER Management Systems (#41a)
Level 3: Utility and REP DER Information and Communications Technology (ICT)
Level 2: Facilities DER Energy Management (FDEMS)
Facilities DER Energy Management Systems (FDEMS) (#5)
Facilities DER Energy Management Systems (FDEMS) (#5)
Facilities DER Energy Management Systems (FDEMS) (#5)
Facilities Site WAN/LAN
Level 1: Autonomous DER Generation and Storage
PV Controller (#4a)
EV Supply Equipment (#6a)
Battery Storage Controller (#4a)
Diesel Controller (#4a)
Utility Grid
Meter and PCC
PV Equipment (#4b)
Electric Vehicle (#6b)
Battery (#4b)
Diesel Generator (#4b)
Facilities Site Load (#3)

Figure 5.1 DER Architecture Levels (2)

to outside to receive remote commands or allow remote maintenance. The used network may include wireless channels, and the grid can be geographically dislocated, making some devices physically reachable and prone to attacks. Moreover, the electrical grid is a critical infrastructure, so it can be the target of attackers with huge technical and economical capacities. For these reasons, cybersecurity is a fundamental issue to improve the resilience of microgrids.

Nevertheless, technologies broadly used for IT cybersecurity cannot be directly applicable in control networks for microgrids (62). Let's consider, for example, cryptography. Control networks in SCADA systems were typically realized by using only proprietary solutions. Several application protocols were developed, each targeting specific communication constraints required by the control systems. The need for remote control and the advances in computer networks led to the blending of traditional control networks with the modern Internet. Consequently, control systems inherited security vulnerabilities that threatened the modern internet (63). In the electrical sector, broadly employed protocols to communicate data and control information are Modbus, DNP3, IEC 60870-5, and IEC 61850. In particular, although developed for substation automation, IEC 61850 suite is exploited for smart microgrids (64). Abstract data models defined in IEC 61850 can be mapped to different protocols, such as Manufacturing Message Specification (MMS), Generic Object Oriented Substation Event (GOOSE), and Sampled Measured Values (SMV), which can run over TCP/IP networks or over substation LANs by using Ethernet. As said, severe vulnerabilities affect these protocols. Different papers show possible attacks at these protocols and the related impact. In Reference (65), vulnerabilities of GOOSE are tested by using real-time simulation and industry standard hardware-in-the-loop emulation. Reference (66) shows how an attacker can launch a Man-In-The-Middle attack on the MMS communications of a photovoltaic inverter installation by using ARP spoofing.

Some of these vulnerabilities are currently addressed by IEC 62351, which is a standard developed by WG15 of IEC TC57. The main purpose is to address the problem of security of TC 57 series of protocols including IEC 60870-5, IEC 60870-6, IEC 61850, IEC 61970, and IEC 61968 series. However, its implementation in actual operation scenarios, such as overcurrent relay coordination or DER management systems, is open to interpretation. For example, IEC 62351-6 standard stipulates the use of digital signatures to ensure integrity in IEC 61850 message exchanges, but the digital signature requires a high computational time with consequent problems for practical implementation in GOOSE messages. For these reasons, IEC 62351 cannot offer a strict procedure for the implementation of cryptography techniques.

Reference (67) provides an assessment of the security of IEC 62351 and concludes that, although the standard contains some inaccuracies and unconventional choices, and does not consider new cryptographic algorithms that could provide the same security guarantees at a lower performance cost, the standard provides a significant security improvement, by assuring authenticity, integrity and confidentiality of data. Some recent papers address the issue of IEC 62351 implementation. A complete evaluation of security mechanisms for IEC 61850 message exchanges, including GOOSE, SV, routable-GOOSE (R-GOOSE), routable-SV (R-SV), MMS is presented in Reference (68). The implementations of IEC 62351-4 Security for IEC 61850 MMS Messages has been discussed in Reference (69). An analysis of the implementation of Message Authentication Code (MAC) Algorithms for GOOSE Message Security according to IEC 62351 has been presented in Reference (70). An analysis and performance evaluation of the implementation of IEC 62351-6 probabilistic signature scheme to secure GOOSE Messages is contained in Reference (71).

Software Defined Networking paradigm also has interesting applications for the security of Industrial Control Systems, especially for incident response. It allows increasing the resiliency of the control system, thanks to the possibility to dynamically re-configure the network after the detection of a fault or of a compromised device, allowing it to operate even in degraded conditions. Focusing on security applications, the SDN paradigm has been applied in different scenarios, to address attacks that target different communication layers. Reference (72) proposes an SDN architecture able to switch between wireless and power line communication to keep proper control within a direct current microgrid under a Denial of Service (DoS) attack. Reference (72) proposes an architecture that exploits SDN control plane message exchanges over the power bus, allowing the reconfiguration of the data plane connections. In this way, all generators in the microgrid operate as either voltage regulators (active agents) or current sources (passive agents), with their operating modes being determined by software-defined controls supported by the control plane communication performed over the power bus. An SDN-based attack detection to protect networked microgrids from cyber-attacks based on a botnet that targets inverter controllers of DERs is presented in Reference (73).

As already discussed, microgrids can operate in grid-connected or islanded modes. If the microgrid works in the grid-connected mode, the generators inject power by following economical logics, while the frequency and voltage are kept in the correct range by the main electrical grid. On the contrary, if it operates in islanded mode; the generators have to guarantee voltage and frequency regulation. The control of the electrical grid is usually schematized into three levels:

- Primary Control: aimed at restoring the imbalance between generation and load by changing the frequency of the power system. In inverter-based microgrids, this is achieved through droop equations; it is the fastest among the three levels.
- Secondary Control: aimed at restoring the nominal value of the frequency and the power exchange among the power systems. It acts at longer time.
- Tertiary Control: aimed at optimizing the economical aspects of load sharing, usually through an Energy Management System (EMS).

Both in islanded and grid-connected modes, EMS can periodically send the power setpoints to the generators through the control network by using different protocols. To jeopardize the control of the electrical grid acting in grid-connected mode can cause economic damages or even, in some cases, afflict the stability of the whole grid. In islanded mode, attacking control mechanism is a severe threat to the grid stability. In inverter-based microgrids, secondary control can be based on communication schemes. In these cases, attacks against the communication infrastructure can have severe consequences on the availability of the whole microgrid. The dynamic of electromagnetic system physics is so fast that the attacks targeting secondary control cannot be recognized in time by an IDS to allow the effective deployment of countermeasures. Moreover, these communication-based schemes are vulnerable to unaddressed cryptography attacks, such as DoS attacks. On the other hand, the control of electrical grids is essential for the service continuity. Resilience is topical in this field. A cyber-attack resilient control strategy for islanded microgrids is presented in Reference (74). The proposed control strategy realizes the detection and isolation of corrupted communication links and controllers in a microgrid whose secondary control is based on a distributed control system. A distributed resilient control strategy for frequency/voltage restoration, fair real power sharing, and state-of-charge balancing in microgrids with multiple Energy Storage Systems in abnormal conditions is presented in Reference (75). Reference (76) studies the impact of various kinds of cyber-attacks, such as false data injection (77), DoS (78), and replay attacks (79), on communication links based on CANBus for secondary control of the distributed generators. Reference (76) also proposes a mitigation strategy based on a reconfigurable secondary control mechanism. Reference (80) introduces a control strategy able to mitigate false data injection and DoS attacks, demonstrating the stability by using the Lyapunov theory under different scenarios, with and without false data injection, and DoS attacks. Reference (81) proposes a distributed optimal frequency control for microgrids resilient against cyber attacks on condition that they are within certain ranges, by introducing an auxiliary networked system interconnecting with the original cooperative control system.

Microgrids can present different DER scenarios, including different types of non-programmable and programmable sources. Non-programmable sources can, anyway, participate to the voltage control by injecting reactive power into the grid. Given the variety of scenarios and the complexity of the interactions of multiple sources participating to frequency control, voltage control, or both, there are still some unaddressed issues in the state of the art to be investigated.

5.3 Physics-based Anomaly Detection for a Photovoltaic System

Physics based Anomaly Detection Systems find many applications in the smart microgrid environment. They use ML approaches both for cybersecurity and fault detection.

Reference (82) proposes an IDS built on the combination of network data, together with power system and control information. After the development a consensus-based distributed voltage control architecture of isolated DC microgrids, an analytical consistency-based anomaly detection mechanism based on variables associated with the proposed algorithm is presented in Reference (83). Reference (84) shows a contextual anomaly detection method based on an artificial neural network and its use in the detection of malicious voltage control actions in the low voltage distribution grid. Reference (85) presents a high-dimensional data-driven cyber-physical attack detection and identification based on electric waveform data measured by waveform sensors in the distribution power networks. Reference (86) describes an anomaly detection algorithm to reveal attacks on PhotoVoltaic (PV) systems, such as PV disconnect, power curtailment, Volt-var attack, and reverse power flow in a portion of the distribution grid with a sufficient percentage of DER penetration. This approach exploits semi-supervised ML algorithms, such as Neural network autoencoder, One Class Support Vector Machine (SVM), Isolation Forest, Random Forest with synthetic corruption, Principal Component Analysis (PCA) with convex hulls, and Inverse-PCA techniques. A deep learning scheme composed of long and short term memory-stacked autoencoders and convolutional neural networks followed by a softmax activation layer is used in Reference (87) for fault detection in a wind turbine.

Also, many works focus on DERs. Different ML solutions have been investigated to identify anomalies/faults in power generation systems: in (88) a k-nn supervised algorithm is used to identify faults in the direct-current portion of a solar power plant composed by many arrays. A Support Vector Machine (SVM) classification algorithm is used in (89) to detect faults in Power Generation Systems Based on Solid Oxide Fuel Cells. Artificial Neural Networks (NN) are applied in (84) in order to identify malicious control of DERs in a grid with high penetration of photovoltaic (PV) generators. An artificial neural network is used in (90) to solve a regression problem in order to predict the power produced by a photovoltaic plant and detect anomalies. An algorithm based on an autoencoder to detect faults within an electric motor is proposed in (91).

In the next sections, a reconstruction-based algorithm that exploits a neural network architecture called autoencoder is presented for both cybersecurity monitoring and fault detection of a photovoltaic system connected to the grid.

5.3.1 Proposed Approach

The proposed solution uses a deep Neural Network architecture called autoencoder. Autoencoders are a type of unsupervised learning algorithms in which the neural network learns to reconstruct its input after a compression of the data, i.e., a reduction of dimensionality. In practice, the autoencoder generates a reduced representation of the set of data and tries to reconstruct a representation of the set of data from the reduced set, which is as close as possible to the input. The difference between the two sets is the reconstruction error. Autoencoders have many applications in the field of image processing but their scope can be much wider. The idea used in this paper is that after training the autoencoder with physical measures, it will be able to learn the correlations among measures. Our approach follows the reconstruction-based novelty detection paradigm where a model is trained to reconstruct normal data with low error. If the input is abnormal, the reconstruction error will be higher. In this way the magnitude of the error, and a consequent proper threshold, is used to classify new data.

The formalized process is the following. We collect all the available measures (also called features) at a discrete sampling time so to build a vector that we call state vector (5.1).

$$X(t) = \{X_1(t), X_2(t) ... X_n(t)\} \tag{5.1}$$

The aim is to detect an anomaly by the only static analysis of this vector. The entire dataset, composed of the state vectors collected over a period of time, which will be used to train the algorithm is called X^{TR}, while X^{TEST} is another collection of state vectors used in the test phase. Therefore, for a given dataset, each row contains the measures collected at the same time and each column the measures of the same type over time. See (5.2), where X is either X^{TR} or X^{TEST}. Sampling time go from t_1 to t_T in this example case.

$$X = \begin{bmatrix} X_1(t_1) & X_2(t_1) & \cdots & X_n(t_1) \\ X_1(t_2) & X_2(t_2) & \cdots & X_n(t_2) \\ \vdots & \vdots & \vdots & \vdots \\ X_1(t_T) & X_2(t_T) & \cdots & X_n(t_T) \end{bmatrix} \tag{5.2}$$

Each measure varies over time in different ranges and ways. In order to compare the measures, we must compensate these differences in a preprocessing phase. Considering the i-th column of X^{TR}, we compute the mean $\bar{X}_i^{TR}$ as in (5.3) and the standard deviation σ_i^{TR} as in (5.4), for the training dataset.

$$\bar{X}_i^{TR} = \frac{\sum_{k=1}^{T} X_i^{TR}(t_k)}{t_T - t_1} \tag{5.3}$$

$$\sigma_i^{TR} = \sqrt{\frac{\sum_{k=1}^{T} (X_i^{TR}(t_k) - \bar{X}_i^{TR})^2}{t_T - t_1}} \tag{5.4}$$

Then we normalize each single measure of the training dataset as in (5.5), so getting the matrix in (5.6):

$$x_i^{TR}(t_k) = \frac{X_i^{TR}(t_k) - \bar{X}_i^{TR}}{\sigma_i^{TR}} \tag{5.5}$$

$$x^{TR} = \begin{bmatrix} x_1^{TR}(t_1) & x_2^{TR}(t_1) & \cdots & x_n^{TR}(t_1) \\ x_1^{TR}(t_2) & x_2^{TR}(t_2) & \cdots & x_n^{TR}(t_2) \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{TR}(t_T) & x_2^{TR}(t_T) & \cdots & x_n^{TR}(t_T) \end{bmatrix} \tag{5.6}$$

Each state vector $x^{TR}(t) = \{x_1^{TR}(t), x_2^{TR}(t), \cdots, x_n^{TR}(t)\}$ is sent to the autoencoder as input and it is reconstructed as output. We call $\tilde{x}^{TR}(t) = \{\tilde{x}_1^{TR}(t), \tilde{x}_2^{TR}(t), \cdots, \tilde{x}_n^{TR}(t)\}$ the vector reconstructed by the autoencoder starting from the input $x^{TR}(t)$. We define the reconstruction error as in (5.7)

$$\begin{cases} e^{TR}(t_k) = \frac{1}{n}\sum_{i=1}^{n} (x_i^{TR}(t_k) - \tilde{x}_i^{TR}(t_k))^2 \\ e^{TR} = \begin{bmatrix} e^{TR}(t_1) \\ e^{TR}(t_2) \\ \vdots \\ e^{TR}(t_T) \end{bmatrix} \end{cases} \tag{5.7}$$

During the training phase the autoencoder is trained to reconstruct its input so to minimize the error in (5.7). Details about this action are reported in the remainder of the paper.

The next step is to decide a threshold to build a classifier to be used in the Test Phase. For this purpose we compute the mean $\bar{e}^{TR}(t)$ and standard deviation σ_e^{TR} of the error on the training dataset as in (5.8) and (5.9)

$$\bar{e}^{TR} = \frac{\sum_{k=1}^{T} e^{TR}(t_k)}{t_T - t_k} \tag{5.8}$$

$$\sigma_e^{TR} = \sqrt{\frac{\sum_{k=1}^{T}(e^{TR}(t_k) - \bar{e}^{TR})^2}{t_T - t_k}} \tag{5.9}$$

Finally we set the threshold as in (5.10), where h is a constant empirically defined.

$$E = \bar{e}^{TR} + h\sigma_e^{TR} \tag{5.10}$$

This structure to define the threshold is due to the hypothesis of a normal distribution of the error. If the probability density function of the error is Gaussian, equation (5.10) allows the setting of a defined percentage of vectors of the training dataset as normal. For example, if h is set to 3, 99.73% of the training vectors will be defined as normal. The choice will be analyzed and discussed in the Results section.

During the test phase, each test state vector at generic instant t_k is normalized by using the mean and standard deviation of the training dataset, as shown in (5.11)

$$x_i^{TEST}(t_k) = \frac{X_i^{TEST}(t_k) - \bar{X}_i^{TR}}{\sigma_i^{TR}} \tag{5.11}$$

So getting (5.12):

$$x^{TEST}(t_k) = \{x_1^{TEST}(t_k), x_2^{TEST}(t_k), \cdots, x_n^{TEST}(t_k)\} \tag{5.12}$$

Then the vector (5.12) is sent to the autoencoder. The reconstruction vector is (5.13).

$$\tilde{x}^{TEST}(t) = \{\tilde{x}_1^{TEST}(t), \tilde{x}_2^{TEST}(t), \cdots, \tilde{x}_n^{TEST}(t)\} \tag{5.13}$$

The error between input and output is computed as in (5.14)

$$\begin{cases} e^{TEST}(t_k) = \frac{1}{n}\sum_{i=1}^{n}(x_i^{TEST}(t_k) - \tilde{x}_i^{TEST}(t_k))^2 \\ e^{TEST} = \begin{bmatrix} e^{TEST}(t_1) \\ e^{TEST}(t_2) \\ \vdots \\ e^{TEST}(t_T) \end{bmatrix} \end{cases} \tag{5.14}$$

The classification is made by comparing the error with the threshold E in (5.10) as indicated in (5.15)

$$\begin{gathered} e^{TEST}(t_k) > E \rightarrow anomaly \\ e^{TEST}(t_k) < E \rightarrow normal \end{gathered} \tag{5.15}$$

To summarize, the classifier is therefore built in two phases. During the training phase (Figure 5.2) the autoencoder learns to reconstruct vectors representing the normal behavior in an optimal way by setting the parameters of the neural network. This operation is done through a gradient descent, an iterative learning algorithm that uses several hyperparameters to update a model. At each iteration the model should be improved by updating the internal model parameters. Two hyperparameters that will be used in the performance evaluation are the batch size and number of epochs. The batch size controls the number of training samples (i.e., the number of rows in the matrix (5.6)) to use before the model's internal parameters are updated. In practice, after the end of the batch, the input is compared with the output and the error is computed. Starting from the error the update algorithm is used to move down along the error gradient. The training dataset can be divided into one or more batches and this number is one parameter affecting the system performance analyzed in the Results section. The number of epochs defines the number of times that the learning algorithm will use the entire training dataset. An epoch is composed of one or more batches. The number of epochs should be large enough so that the learning algorithm can run until the error has been sufficiently minimized. The number of epochs is the second hyperparameter studied in the Results. After the training phase, a threshold E is set.

In the test phase (Figure 5.3) the state vectors are sent to the autoencoder, and the reconstruction error is used to classify vectors.

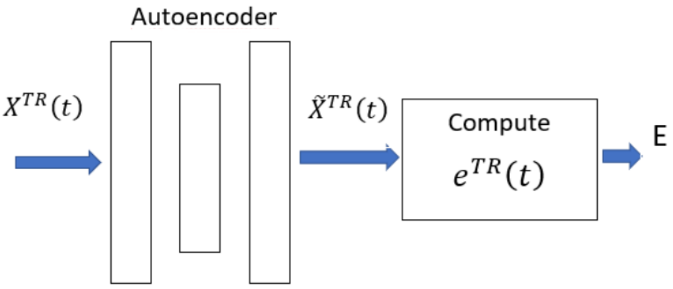

Figure 5.2 Training phase

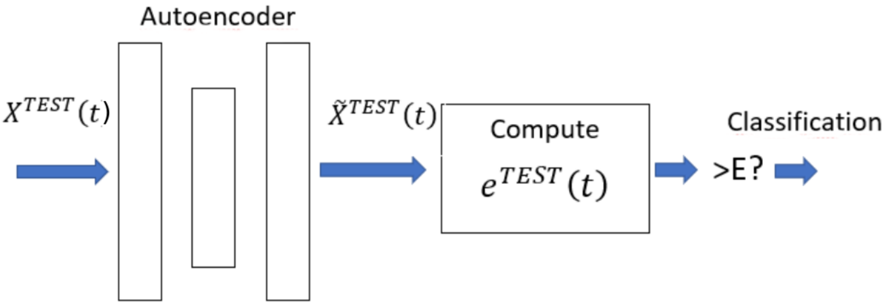

Figure 5.3 Test phase.

5.3.2 Developed Testbed

The use case of this paper is a typical small-scale photovoltaic system connected to the grid at the distribution level. From an electrical point of view it is composed of solar panels, a DC-DC boost electronic converter controlled by the maximum power point tracking algorithm, a DC link and a current source inverter. The inverter acts also as a server sending information to and receiving commands from a control unit, which can be represented by a SCADA in the case of a microgrid or by a distribution management system in the case of distribution grid. The system is also equipped with different sensors that collect heterogeneous types of measurements. We categorize all the collectible information in five groups:

- Alternating Current (AC) side electrical information: active and reactive power, voltages (Root Mean Square, RMS), currents (RMS), frequencies, total harmonic distortion (THD)
- Direct-Current (DC) side electrical information: voltages and currents
- PV information: voltage, current, temperature of the cells
- Environmental information: irradiance, temperature of the air

- Electronic information: maximum power point, dc/dc converter duty cycle

As said, we would like to detect anomalies by the only observation of the collected measures. In order to validate the proposed approach, we set up a simulation environment. The physical behavior of the photovoltaic system is simulated by using MATLAB/Simulink software, as shown in Figure 5.4.

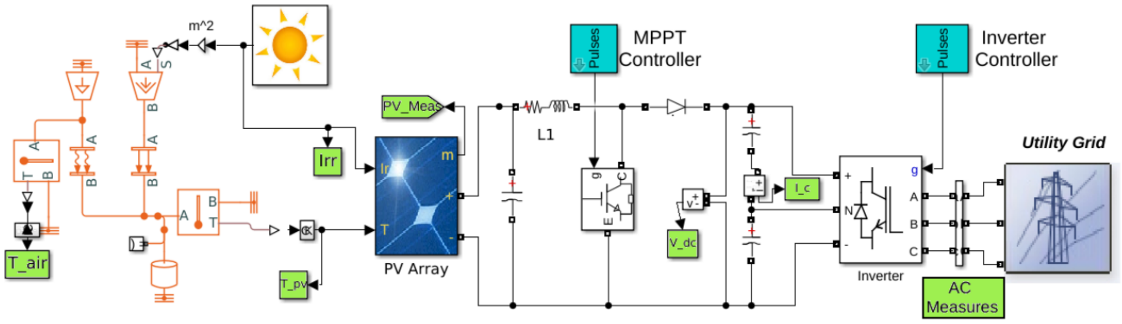

Figure 5.4 Simulated environment on Simulink.

The scheme takes into account the heat exchange between the panels and the environment (in red in the left part of Figure 5.4) starting from the external data of solar irradiance and the temperature of the air. This is simulated by a radiation heat transfer coming from the sun, a convective heat exchange with the environment considered to be an ideal temperature source, and a thermal inertia of the photovoltaic panels.

Then we implemented an electromagnetic electrical model, the portion between the PV Array and the Inverter in Figure 5.4, starting from the blocks already present in the software. The grid is modeled by a small low voltage portion with some loads, and a transformer connected to the medium voltage distribution. We extract 22 features composing vector (5.1), which represent the physical parameters that are measured in many commercial solutions, especially in microgrids. The list of these features is reported in Table 5.1. The green boxes in Figure 5.4 are the points where we extract the measures listed in Table 5.1 from the system. Concerning the boxes that indicate the points where multiple measures are extracted: “PV_Meas” refers to X_4 and X_5, i.e., V_{pv} and I_{pv}; “AC measures” block is the point where we extract the measures from X_9 to X_{22}. X_8 is measured within the MPPT converter. The model runs for different working conditions in order to create a large dataset.

We simulated three types of faults/cyber-attacks that have different impact on the measures:

- Reduction of active power injection

Table 5.1 List and description of the features.

Feature	Symbol	Description
X_1	Irr	the solar irradiance hitting the panel
X_2	T_{air}	the temperature of the environment
X_3	T_{pv}	the temperature of the PV's cells
X_4	V_{pv}	the voltage measured at the terminals of the panel
X_5	I_{pv}	the current emitted by the panel
X_6	V_{dc}	the voltage measured at the DC link
X_7	I_c	the average current in the DC capacitor
X_8	δ	the dutycycle of the DC/DC converter
X_9	V_a	the voltage of phase a (AC side)
X_{10}	V_b	the voltage of phase b (AC side)
X_{11}	V_c	the voltage of phase c (AC side)
X_{12}	I_a	the current of phase a
X_{13}	I_b	the current of phase b
X_{14}	I_c	the current of phase c
X_{15}	f_a	the frequency of phase a
X_{16}	f_b	the frequency of phase b
X_{17}	f_c	the frequency of phase c
X_{18}	THD_a	the total harmonic distortion of the voltage on phase a
X_{19}	THD_b	the total harmonic distortion of the voltage on phase b
X_{20}	THD_c	the total harmonic distortion of the voltage on phase c
X_{21}	Q	the reactive power emitted by the inverter
X_{22}	P	the active power emitted by the inverter

- Short circuit of some cells of the solar panel
- Bad data injection

In the first case, we use the remote-control capabilities of the inverter: the action implies a minor active power injection to the grid compared to the possible available power considering environmental conditions. In the second case, we consider a typical fault that can happen in a solar panel, which affects the performance of the panel itself. In the third case, we modify only one feature for sample, by changing its value of a percentage ranging from 25 to 50% of the original value, in order to create an unfeasible state vector. For instance, the injected power is modified maintaining unchanged voltage and current measures. A bad data injection attack can be dangerous for DERs because it can induce a wrong decision in a remote control system.

The classification is performed offline on the test dataset. The autoencoder, implemented in Python by using Keras library (92), is a multilevel neural network architecture that uses input and output layers composed of the same number of neurons and a series of hidden layers whose dimension is strictly lower than the dimension of input and output layers. Neurons are fully connected, which means that each single neuron acts as an input for all the neurons of the following layer. The activation function of each neuron is a sigmoid. During the training phase, all the parameters of the neurons (i.e., the weights of the connections) are set by using the gradient descent technique. The impact on the performance of the following two hyperparameters:

- Batch size
- Epochs

The variation of these hyperparameters can strongly influence the model learned by the NN.

5.3.3 Results

We extracted a training dataset composed of 7200 vectors, corresponding to about 30 days of operation under different weather conditions, and 3 test dataset composed of about 800 vectors corresponding to some hours of operation (because of a smaller sampling time with respect to the training set) in normal and abnormal conditions. Tests are made in order to evaluate the influence of different elements so to improve the efficiency of the detection method. We focused on the choice of the threshold E in (5.10), on the neural network architecture, and on the hyperparameters used to train the NN.

As indicated in (5.10) we set the threshold E as the mean error plus h times the standard deviation of the error computed on the training dataset. The choice is theoretically fully justified if the probability density function of the error is Gaussian because, in this case, the area below the curve within the range identified by $(\bar{e}^{TR} - \sigma_e^{TR})$ and $(\bar{e}^{TR} + \sigma_e^{TR})$ is always equal to 0.6827, by $(\bar{e}^{TR} - 2\sigma_e^{TR})$ and $(\bar{e}^{TR} + 3\sigma_e^{TR})$ equal to 0.9545, by $(\bar{e}^{TR} - 3\sigma_e^{TR})$ and $(\bar{e}^{TR} + 3\sigma_e^{TR})$ equal to 0.9973, and so on, allowing h to control the probability that the random variable error e^{TR} is within the range defined by $(\bar{e}^{TR} + h\sigma_e^{TR})$. Supposing that the distribution of e^{TEST} is the same of the one of e^{TR}, h would allow the setting of a defined percentage of vectors as normal. Fixing the structure of the autoencoder with a single hidden layer of dimension 15, and low epochs (=10) and high batch size (=256), Figure 5.5 reports the histogram of the normalized errors, the curve that approximates the related probability density function, and the corresponding normal curve with the same mean and standard deviation (red line). The shapes are qualitatively similar by changing hidden layer dimension, epoch and batch size.

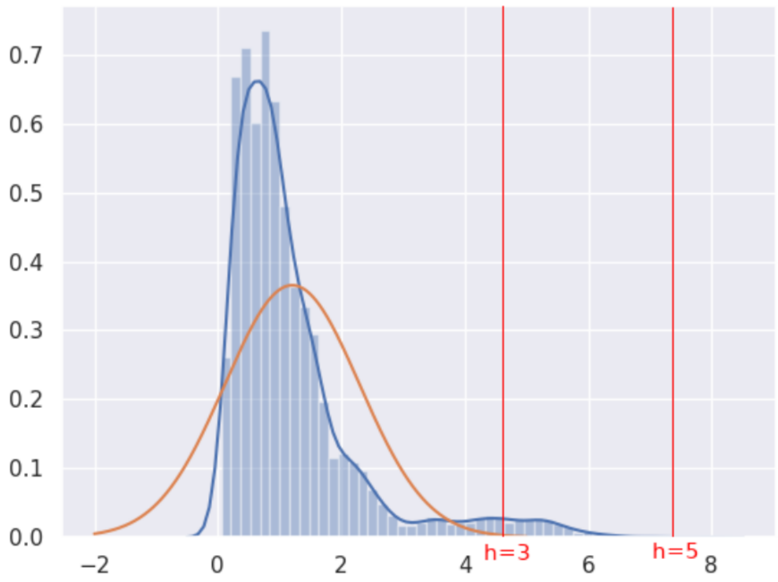

Figure 5.5 Distribution of errors computed on training dataset.

We can notice that even if the distribution is different from the Gaussian, the proposed structure for the choice of the threshold may be still applied even if the selection of the h value must be heuristic. h can still select a percentage of vectors as normal, even if more coarsely than in the Gaussian case. In Figure 5.5 two values of the threshold are reported; the choice of $h = 5$ has the intention to maintain a small number of false positives. To analyze numerically the effect on the performance we varied h from 3 to 7, as reported in Table 5.2. The motivation is that above $h = 3$ and beyond $h = 7$ results show that the performance decreases significantly. The acronyms TN, FN, FP and TP stand, respectively, for True and False Negative, and False and True Positive. Accuracy is the proportion of true results.

Table 5.2 Impact of the choice of the threshold on accuracy.

	Power Reduction	**Short Circuited Cells**	**Bad Data Injection**
$h = 3$	Accuracy: 0.729 TN: 221 FN: 0 FP: 216 TP: 361	Accuracy: 0.772 TN: 243 FN: 1 FP: 183 TP: 379	Accuracy: 0.497 TN: 125 FN: 15 FP: 339 TP: 225
$h = 4$	Accuracy: 0.948 TN: 396 FN: 0 FP: 41 TP: 361	Accuracy: 0.842 TN: 412 FN: 113 FP: 14 TP: 267	Accuracy: 0.714 TN: 314 FN: 51 FP: 150 TP: 189
$h = 5$	Accuracy: 0.995 TN: 433 FN: 0 FP: 4 TP: 361	Accuracy: 0.68 TN: 424 FN: 230 FP: 2 TP: 130	Accuracy: 0.839 TN: 440 FN: 89 FP: 24 TP: 151
$h = 6$	Accuracy: 1 TN: 437 FN: 0 FP: 0 TP: 361	Accuracy: 0.547 TN: 426 FN: 365 FP: 0 TP: 15	Accuracy: 0.839 TN: 456 FN: 105 FP: 8 TP: 135
$h = 7$	Accuracy: 1 TN: 437 FN: 0 FP: 0 TP: 361	Accuracy: 0.536 TN: 426 FN: 374 FP: 0 TP: 6	Accuracy: 0.841 TN: 463 FN: 111 FP: 1 TP: 129

These preliminary results highlight important aspects: the best threshold's choice depends on the specific case because some physical anomalies produce higher errors that are more easily separable. For examples, anomalies that involve the modification of a high number of measures cause a higher reconstruction error. Therefore, setting a high value for the threshold E (through a higher h) reduces the number of false positives. On the contrary, anomalies that induce a small amount of measures to change cause a lower reconstruction error, consequently the threshold E should be maintained lower (through a lower h), even if this choice raises the number of false positives. If we want to avoid to loose generality, we must fix a threshold that is a compromise. On the first phase, we set $h = 5$.

Subsequently, fixing $h = 5$, the impact of the NN architecture on the accuracy has been investigated. We tried different combinations of depth and number of neurons for hidden layer. For example, considering Table 5.3, "22-15-22" refers to an architecture composed of a single hidden layer whose dimension is 15 and an input and output layer each composed of 22 neurons corresponding to the number of features in Table **??**, while "22-18-15-18-22" refers to an architecture composed of 3 hidden layers whose dimensions are 18, 15 and 18, respectively. The layers are fully connected, which means that each single neuron of a layer acts as an input for each neuron of the following layer.

The number of neurons of the most compressed layer impacts on the accuracy more than the number of hidden layers. In the presented results, in some cases, the depth of the NN impacts negatively on the accuracy, but we can say, in general that the depth does not bring significant changes in the performance. A possible explanation is that the dimension of the smallest hidden layer is the element that mostly affects the reduced representation of the system and, consequently, the capability to learn the correlation between measures. If this dimension is too large, the autoencoder just reproduces its input; on the contrary, if the dimension is too small, the NN loses important information.

Table 5.3 Impact of the NN architecture on accuracy.

	Power Reduction	**Short Circuited Cells**	**Bad Data Injection**
22-18-22	0.994	0.660	0.820
22-15-22	0.995	0.687	0.839
22-10-22	0.995	0.680	0.824
22-15-10-15-22	0.995	0.612	0.825
22-18-15-18-22	0.995	0.661	0.830
22-21-(...)-13-(...)-22	0.995	0.659	0.830

Finally, we investigated the gradient descent's hyperparameters, focusing on batch size and epochs. After the previous phases, we fixed the threshold to $h = 5$ and the NN architecture to 22-18-15-18-22. We compared the results by using different learning hyperparameters. The first test focuses on the batch size, fixing the number of epochs to 10. Results are reported in Table 5.4

Table 5.4 Impact of the batch size on accuracy

	Power Reduction	**Short Circuited Cells**	**Bad Data Injection**	**Mean Accuracy**
Batch size: 256	0.995	0.650	0.837	0.827
Batch size = 64	0.993	0.702	0.820	0.838
Batch size = 32	0.992	0.789	0.825	0.869
Batch size = 16	0.992	0.819	0.801	0.870
Batch size = 1	0.992	0.801	0.801	0.864

The second test refers to the number of epochs. We fixed the batch size to 32 and we evaluated the impact on accuracy by varying the epochs (Table 5.5).

Table 5.5 Impact of the epochs on accuracy

	Power Reduction	**Short Circuited Cells**	**Bad Data Injection**	**Mean Accuracy**
Epochs = 10	0.992	0.789	0.825	0.868
Epochs = 50	0.992	0.814	0.809	0.872
Epochs = 100	0.992	0.840	0.808	0.880
Epochs = 200	0.992	0.834	0.788	0.871

There is a relevant impact of hyperparameters on the accuracy concerning anomalies caused by Short Circuited Cells and Bad Data Injection. More times the training samples are passed to the autoencoder during training phase, more accurate is the model fitted by the NN. The identification of anomalies that produces lower errors benefits from a more suitable model built around training samples.

The results of this work have been published in two subsequent works (93) (94).

5.3.4 Discussion and Future Developments

A physics-based anomaly detection algorithm for the safety and cybersecurity of a photovoltaic system has been discussed in the previous sections. The results are really promising. For this reason, we think the same methodology can be applied to different DERs.

DERs are usually deployed within microgrids or energy communities, and controlled by Supervisory Control and Data Acquisition Systems. One fundamental element of microgrids is represented by Battery Energy Storage Systems (BESSs). Storage is used in order to balance the production of uncontrollable sources like photovoltaic systems, both for economic purposes or to allow the microgrid to operate in islanded mode. Storage Battery systems are complex systems composed of different electric, electronic and communication apparatus. We consider a typical scenario of a storage system connected to a low or medium voltage microgrid controlled by a SCADA. From an electrical point of view, it is composed by:

- Module of cells (one or more) equipped with their own Battery Management System (BMS), which guarantees that the cells maintain safe ranges of voltage, current, temperature and so on.

- DC/DC converter: electronic converter that adapts the voltage of the cells to the voltage suitable for the Active Front End.

- Active Front End: electronic converter that transforms direct current into a three phase alternating current, allowing a bidirectional power flow.

The BMS is any electronic system that manages a rechargeable battery (cell or battery pack); its main tasks is protecting the battery from operating outside its safe operating area, monitoring its state, calculating secondary data, reporting data, controlling its environment, authenticating it and/or balancing it. BMS can communicate to a higher level controller through different solutions, such as different types of serial communications, CANBus, Modbus, or even utilize a specific protocol and a gateway in series.

The same solutions can be used by the power electronic converters in order to communicate between them and with a Process Control Systems (PCS). This is usually represented by an industrial PC, which acts as an interface between the SCADA and local controllers. PCS usually possess a local Human Machine Interface (HMI), which allows to interact with its monitoring and control function. The communication between the PCS and the SCADA can be based on protocols belonging to the IEC 61850 suite. The overall scheme is shown in figure 5.6

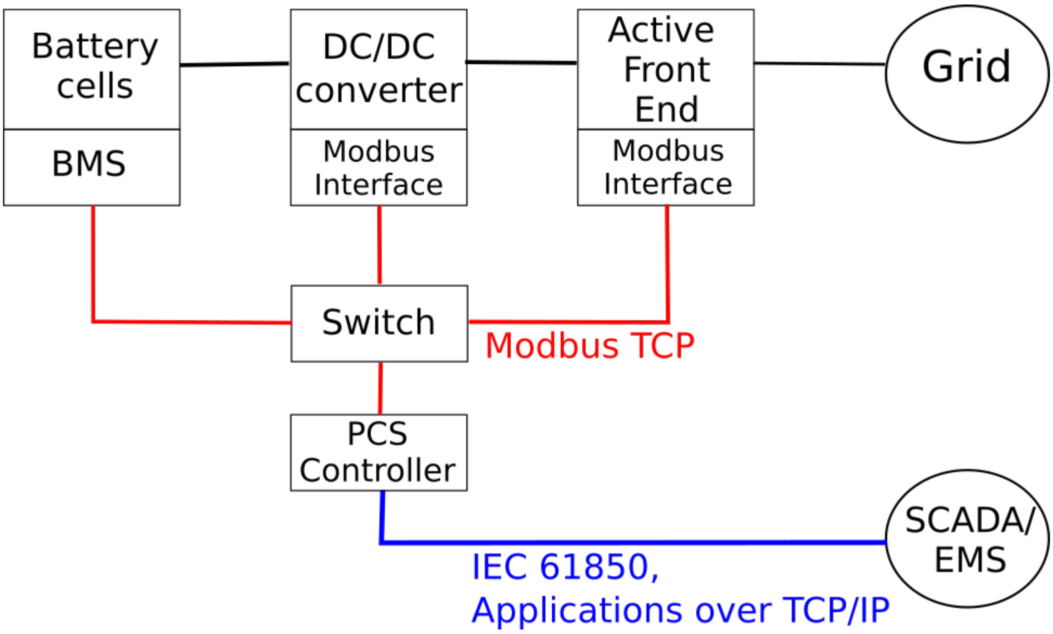

Figure 5.6 Architecture of a generic BESS

The SCADA communicates with the PCS for monitoring and control purposes. Microgrid are usually equipped by an Energy Management System (EMS) that has the role to plan the energy production of dispatchable sources. Storage systems usually operate in three different modes:

- P/Q mode: the generator inject the desired active and reactive power.

- Islanded mode: the generators maintain the voltage and the frequency constant, while providing the necessary power to balance the loads
- Droop equations mode: the frequency and the voltage maintained by the generator depends on the injected powers.

The SCADA has the role of setting the operational mode and, depending on that, giving the power setpoints to the generator. A change of operational modes is needed whenever the microgrid switch between islanded mode and grid connected mode.

Many different architectures are possible for connecting the electronic converters to the SCADA system. Nevertheless, the architecture discussed before consist in a minimal scheme, with a low number of connections and no direct connection between the control apparatus and SCADA network, in order to reduce the attack surface.

We can categorize the possible attacks on the proposed infrastructure from different points of view.

Aim of the attack: it is possible to target the storage system with the aim of damaging the apparatus, or to use the storage to cause problems to the microgrid, both from safety or economic perspective. The control of a generator can cause severe damage to the grid, depending on the size of the generator, the features of the grid and the operation mode. In fact, in the case that the microgrid operates in islanded mode with only one storage system that has the role of balancing the powers, it's obvious that the compromise of that generator would cause a complete blackout of the microgrid. In case the power of the storage system is not essential, e.g. the microgrid has multiple generators that do the voltage and frequency control or the microgrid is grid-connected, the compromise of that generator can cause damages to the grid as well. For example, a voltage or frequency variation could cause the trigger of some electrical protection, or even more protection in series.

Exploited vulnerability: if the attacker can gain access to the control network, it can first compromise the communication between the PCS and the SCADA exploiting IEC 61850 vulnerabilities; in that case, he would be able to send fake commands to the PCS or fake measures to the SCADA. Moreover, the PCS could expose different services, like web applications, Virtual Network Computing sofwares and so on. If an attacker is able to take control of the PCS through previously mentioned vulnerabilities, he would be able to communicate directly with electronic controllers and especially with the BMS, potentially causing the destruction of the whole storage system. It is worth mentioning that, even if it is a borderline case, an attacker could violate some physical security countermeasures directly accessing the telecommunication network, since the microgrid apparatus could be geographically dislocated over a wide area, making it difficult to guarantee the impossibility of accessing some apparatus.

Sophistication of the attack: of course, if the system presents severe vulnerabilities, the attacker can endanger the safety of the whole system. Nevertheless, given the complexity of a microgrid, even a simpler attack can cause severe problems. Let's consider a bad data injection on the value of the State Of Charge (SOC) that the PCS communicate to the SCADA: that compromise could cause erroneous programming by the EMS, which would suggest wrong power setpoint. If the storage present automatic actions when the SOC reaches dangerous levels, that would cause economic damages, otherwise even the safety of the system is in danger.

A complete taxonomy of possible attacks on a storage system is not feasible, since it depends on many factors, including the characteristics of the grid by which the storage system is connected. Still, the proposed evaluation of the attack model suggests that different security monitoring systems working in parallel would be useful to limit the risks. For these reasons, BESS represent one of the most interesting use-case for applying the approach developed for the photovoltaic system discussed above.

Chapter 6

Integration of Physical and Cyber Security

6.1 The need for integration of Physical and Cyber Security

We are in an era where both cybersecurity systems and physical security systems collect vast amounts of data. Although we have achieved advanced data analysis capabilities, including for example image analysis, license plate recognition and faces recognition, predictive analysis with machine learning algorithms, artificial intelligence, correlation of internal sources and external intelligence, etc., the two systems in most cases remain separate. Complex attacks, occurring simultaneously at different levels - from an illegal entry into the room to network vulnerability exploitation, are widespread. To protect against these attacks it is very important to process security data from a variety of heterogeneous sources of both physical and cybernetic levels. In the existing security systems, typically, physical and cybernetic security systems operate independently, although complex access control systems combine several types of events.

Physical Security Information Management (PSIM) systems are used to monitor and manage security in physical systems. They are largely adopted in critical environments where security and safety are the main requirements. Similarly, Security Information and Event Management (SIEM) systems are used to monitor computer systems. They detect anomalies that are a symptom of cyber security issues. Physical and cyber threats were debated separately, until few years ago. Then, the Stuxnet worm episode showed physical and cyber worlds becoming closer (6). Some examples of complex attack patterns which can afflict Cyber-physical systems, but more important could remain undetected by the only use of traditional security monitoring systems, comprehend the followings:

- USB drop, often to attack environments protected by air gaps, where USB sticks are abandoned for example in parking lots
- Physical attacks that exploit social engineering and intelligence to gather information detailed on the victim and thus bypass the security checks (cases of fake vendors with real fixed appointments, fake access passes, etc.)
- "cyber" tampering of physical security systems (like the case of the Port of Antwerp)
- insider activities (operators or suppliers, for example)
- improper behavior by operators, such as sharing administrative credentials
- Use of badges or credentials of former employees
- devices for physically breaking into the network such as "weaponized" sockets

To meet today's challenges, it is necessary to develop a integrated approach for security in cyber-physical systems.

Data mining for cyber security applications is a growing field of research. Traditional Security Monitoring technologies are Network Intrusion Detection Systems (22) and Host Intrusion Detection Systems (25). These technologies make large use of Machine Learning techniques (95). Particularly interesting in the field of ICS security are Anomaly Detection (or Novelty Detection) algorithms (28), due to the lack of sufficient data related to cyber-physical attacks. Security Monitoring of ICS increasingly keeps into account heterogeneous sources of information (96).

Some papers in the literature focus on the correlation of events within SIEM systems to reduce network complexity. They investigate strategies for alarm event pre-processing in order to reduce the number of displayed alarms and so to simplify the analysis of human operators. An overview of the most popular SIEM tools and open-source rule-based correlation engines (including IBM QRadar, HP ArcSight, Splunk and LogRhythm) is presented in (97), comparing the engine correlation mechanism, and classifying them into Similarity-based, Knowledge-based and Statistical correlation. Authors in (98) propose two novel alert correlation approaches for SIEM systems: enforcement-based correlation, aimed at classifying all possible countermeasures and their associated policy enforcement points that will implement the security rule as a defence mechanism; and metric-based correlation, aimed at deriving correlation rules from information security indicators that allow the analysis and evaluation of the SIEM effectiveness. Only a few works focus on the correlation of heterogeneous events for security reasons. One of the most challenging goals is to discover complex attack patterns through the combination of physical and cyber events. Some preliminary approaches for the integration of heterogeneous data sources and the correlation of apparently disparate events for protection against cyber-physical attacks are reported in (99). A framework for event collection and correlation that can process and analyse heterogeneous data through event pattern detectors, and integrate them into the open-source SIEM OSSIM is proposed in (100). Authors in (101) address the issue of Physical Security Information Management and Security Information and Event Management integration by using the IBM SIEM QRadar as a platform. Another framework, called synERGY, for cross-layer anomaly detection based on ML techniques to enable the early discovery of both cyber and physical attacks with impact on the cyber-physical system is presented in (102).

The discussed works present interesting solutions for the implementation of a security monitoring system by using already developed and off-the-shelf SIEM infrastructures. However, the correlation strategies of heterogeneous events for security reasons is still an open issue, as well as the techniques and algorithms that can allow exploiting this correlation. The solutions proposed in the state of the art cover only a small portion of the possible use cases and are difficult to compare with each other due to the lack of shared use-cases and datasets.

6.2 Logs Systems in the Industrial Sector

We can distinguish between three main domains for log systems, which we call Physical, Cyber and Cyber-Physical.

The information that can be collected from the physical environment is usually very heterogeneous and can be generated from different sources.We mention three main fields:

- Physical Access Control Systems (PACS);
- Video surveillance systems;
- Environmental sensor systems.

Physical Access Control Systems (PACS) are systems that control the physical access to the monitored area and across different zones of the monitored area. They are based on Personal Identity Verification (PIV) for people authentication, which is typically based on the common triad of: something you know (such as a password), something you have (such as a smart card), something you are (such as a fingerprint or other biometric information). The number of required authentication factors may depend on the restriction level of the different zones. (103). Video surveillance systems increasingly rely on communication systems that can be based on wired or wireless technologies and can use artificial vision techniques for automatic analysis of recorded videos, which is particularly useful for crowd surveillance (104). In the past few years, many camera devices have been found vulnerable to cyber attacks (105). Environmental sensor systems include a wide set of possible information sources and they also often rely on ICT solutions for data transmission. Some examples are: voltage sensors for batteries or Uninterruptible Power Supply (UPS); humidity sensors, to prevent premature ageing of equipment; temperature sensors, to detect air conditioning outages which can be very dangerous for specific devices, such as servers; fluid sensors, to detect water leakages; airflow sensors, to ensure that enough air is flowing through a particular area preventing hot spots; motion sensors, to detect people presence in secure areas with access restrictions; audio sensors, to detect noises, such as breaking glasses and alarms.

Cyber Physical System is an umbrella term that includes different kinds of systems, such as robotics, machine automation, industrial and process control systems, SCADA, Industrial Internet, and Internet of Things (IoT) (106). ICS, and in particular SCADA systems, log the events related to the industrial process in order to allow operators to supervise the process and guarantee safety and continuity of the plant operations. In industrial plants, the main goal of an attacker is to modify the physical behaviour of the process in order to cause service disruptions and/or damages to devices and even people. If the control network is compromised, an attacker can send fake commands to the actuators, but also act without allowing operators to notice the ongoing attack, as happened with the Stuxnet worm. For these reasons, from a cyber security perspective, SCADA often utilises Network and Host-based Intrusion Detection Systems (IDS) and Physical-Behaviour-based Anomaly Detection algorithms to promptly realise the presence of attacks. It is worth noticing that a shared standard for logging events in SCADA does not exist. These systems are customisable and it could be difficult for not specifically trained people to fully understand the events that trigger alarms. This may be a huge problem in correlating and integrating SCADA logs with the ones of physical and cyber domains.

In ICT systems, almost all devices can generate, store, and send information which we generally call logs. These logs can come from different and numerous sources including firewalls, IDS, Intrusion Prevention Systems (IPS), and Virtual Private Networks (VPN), Antivirus software, Identity management systems, network devices (routers, switches, wireless access points, ...), operating systems, and applications.. Complex computer systems have been developed to collect and analyse this huge amount of information. A lot of acronyms have been defined over the past few years to describe these solutions, such as Enterprise Security Management (ESM), Enterprise Event Management (EEM), Security Information Management (SIM), Security Event Management (SEM), and Security Information Event Management (SIEM). We will refer to all of them with the term SIEM. A SIEM system is generally designed to provide the following set of services (107):

- Log management: collection, storage, and analysis of all logs;
- IT regulatory compliance: audit and validate compliance or identify violations of compliance requirements imposed upon the organisation;
- Event correlation: automatically analyse and correlate data in order to promptly recognise risks;
- Active response: implement countermeasures directly acting from the SIEM system;
- Endpoint security: make adjustments to the node security on the remote system, such as configuring firewalls and updating and monitoring antivirus, antispyware, and antispam products.

Different vendors may produce the devices that generate data in input to the SIEM system, which are usually reported in different and proprietary formats. Even the way that events are reported to upstream logging server functions may not be universal (108). An example is Syslog , an industry-standard method based on RFC 5424 (109), which allows devices to record and report events.

A SIEM system can be divided into six pieces or processes, as shown in Figure 6.1.

- source device;
- log collection;
- parsing/normalisation of the logs;
- rule engine;
- log storage;
- event monitoring and retrieval.

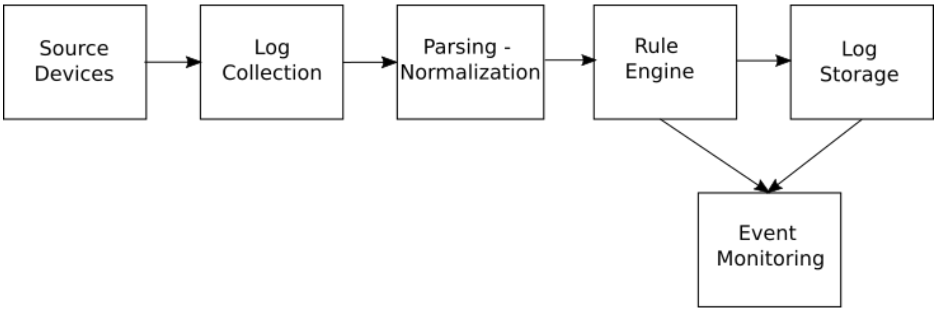

Figure 6.1 Architecture of a SIEM

Some standards address the issue of data generation by source devices, data collection, and normalisation. An interesting research field is the correlation strategies used by the rule engine. While some attack patterns can be easily detected by using simple rules, more complex attacks may require more sophisticated approaches to be detected, also exploiting the capabilities of ML algorithms.

6.3 Integrated Cyber-Physical Anomaly Detection

To start designing log management systems, three are the basic questions to keep in mind:

- What kind of information logs are useful?
- How many logs have to be measured?
- How long do the logs have to be stored?

Concerning the first question, information logs can be produced by every device on a network. Each log source can usually be tuned to provide a record of multiple kinds of information with different detail levels depending on the use. Syslog can be used for network devices. Physical domain-related systems are usually far less tunable and just offer the measure of the physical quantity, but, nevertheless, provided information can be very useful for security monitoring. Concerning the other two questions, the possible answers strongly depend on the implemented correlation logic. On one hand any information that can be of help monitor the system can be precious, even if, apparently, may seem not important. The correlation among different logs may be unexpected but fundamental to improve security monitoring. On the other hand, storing logs for longer times may help improve the efficiency of security monitoring. Observing the same log over time may reveal important details as well as establishing correlation among different logs measured at shifted time instants. Within an extended time range, the correlation engine could employ different strategies to exploit these stored data so to identify possible risky situations with higher accuracy. Moreover, other practical details are essential to complete the answer. The amount of produced logs and event information rapidly increases even in small systems, quickly becoming too big to be stored. Industry regulations and laws require that certain types of information must be stored for a given period of time (data retention). Legal and functional drivers also dictate how information can be disposed after a given period of time, also often implying data destruction. Once the whole heterogeneous data generated by multiple sources within the monitored area have been collected, possible strategies to manage the complexity and identify dangerous situations (threats) have to be properly designed.

In more detail, a threat can be represented by the occurrence of events that can be related to one of the following elements:

- Correlation by physical area. Multiple events that happen in a delimited physical or logical area can represent a situation of risk. For example, the presence of a person in a specific area can be correlated with the use of an IP address that belongs to a subnet physically accessible in that area. Besides, the correlation of different systems can help identify a contradictory situation that can represent a possible risk. For example, the PACS states that no one is present in a specific room while the room's physical sensors detect the presence of someone. This situation may be due to an error of some sensors or to a violation of physical countermeasures.

- Correlation by person. Tracking the activity of a person within the monitored area can be very useful to prevent threats, even if it involves different physical and ICT log systems. Since most humans are habitual, algorithms can notice an unusual behaviour which can be labeled as an anomaly. Additionally, correlating logical and physical accesses can reveal malicious actions. For example, a person that simultaneously gets physical and remote access to a system within the monitored area is a suspicious event. Nevertheless, this type of correlation can be difficult to detect: log systems usually memorise information about accounts, badges, RFIDs, or car registration numbers without directly referring to the people's names or to other elements that uniquely identify a person across multiple log systems.

- Correlation by time. Two events that occur in a limited time interval could be a symptom of a causal correlation between them. A simple example is the activity of port scanning within the network followed by multiple failed login attempts. However, to properly identify which events can be related by time because of possible causal reasons and which time window size is appropriate to identify correlations may not be an easy task

In order to provide a correlation between these systems, we identify different possible solutions:

- Visual Analytics. Approach that involves the design of proper Human Machine Interfaces (HMI), typically composed of different windows and based on different graphical solutions, to visually highlight the correlation among the collected data to human operators. HMI represent a very useful tool for human operators who are in charge of making decisions, even if the ever-growing amount and complexity of data can be hardly manageable without the support of automatic approaches.

- Rules. Fixed rules can be set to check simple conditions. SIEM systems can implement automatic algorithms based on "if/if then else" sentences coming, for example, from corporate policies or simple potential risk situations identified a priori. For example, a rule can be represented by: "if user1 is in the control room and user1 is remotely logged to the server through VPN -> anomaly".

- Machine Learning. In case a set of basic rules is not enough to properly depict the overall set of possible anomaly situations, ML-based solutions can be employed to extract knowledge from big datasets. ML-based anomaly detection algorithms are particularly interesting to exploit heterogeneous logs. They can be customised to the specific monitored area learning the specific patterns and habits of the employees during the training phase without explicitly declaring which behaviours are considered as anomalies and which are not.

The present work proposes an architecture, which will be discussed in the next session, able to manage both ML approaches, and rule-based approaches, trying to automatize the data analysis process, so that HMI can present pre-processed data, improving the human analysis.

6.3.1 Proposed Approach

We consider a generic industrial plant, e.g., a power plant, which is located within a fenced area. Its schematic representation is shown in Figure 6.2.

The industrial process is controlled by a SCADA system whose servers are located in a dedicated room, which we call TLC room, that also contains the HMI for the operators. All the employees are identified at the gates of the industrial plant and only a portion of them are

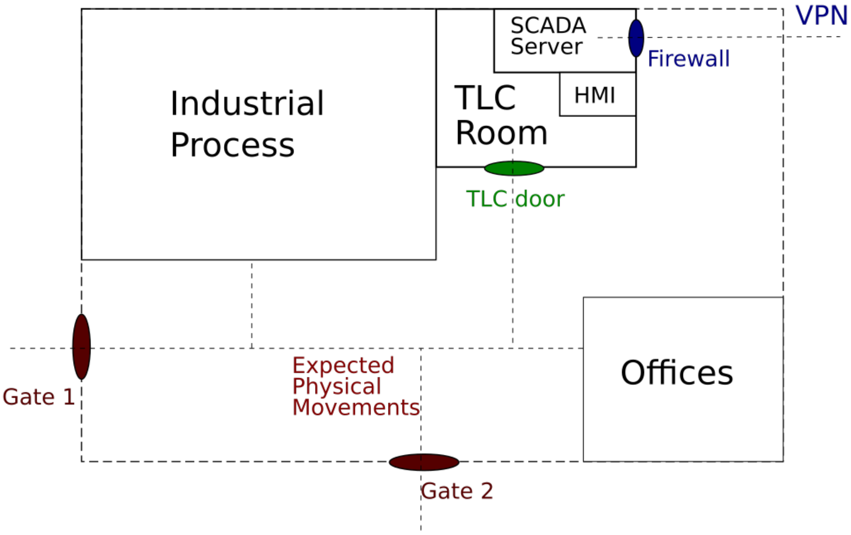

Figure 6.2 Schematic representation of the considered scenario

allowed to access the TLC room. So, the plant implements two main physical log systems: the first one that manages the accesses to the plant and the second one that manages the ones to the TLC room. The SCADA system is connected to the TCP/IP based enterprise network through a firewall. The SCADA server can be reached from outside only by using a Virtual Private Network (VPN). This situation is quite common for the industrial plants that allow authorised users to connect remotely to the SCADA server for assistance and maintenance. The VPN system logs all the accesses and some information about the exchanged traffic.

In the considered scenario, the main target of the attacks is the SCADA system. The attacks aim to interfere with the normal industrial process so creating economic damages, service disruptions, and damages to devices and even people. As we mentioned, the SCADA server can be accessed both physically and remotely. An attack against the control system can therefore be carried out by physically reaching the control device, for example, by stealing a badge of a person authorised to access the plant and the TLC room, or remotely, by exploiting vulnerabilities of the cyber defence system or by getting remote access credentials. An attack can be also carried out by a combination of physical and remote strategies. An example is acting through abandoned USB pen drives waiting to be picked up by an inadvertent employee who will plug them into work PCs possibly in the TLC room. One of the most dangerous threats is represented by insiders, i.e. people that are normally authorised to access the plant but decide to "switch sides". For these reasons, there is a huge variety of potential risk situations that have to be considered and need to be tackled.

Due to the complexity of taking into account such a variety of data simultaneously, the proposed approach is based on the idea of decomposing the complexity through different analysis levels. We propose a conceptual architecture able to manage the possible data coming from an industrial plant and also scalable to allow possible further integration of additional log systems. In order to do this, the output of each log system is pre-processed and feeds multiple ML algorithms that act in parallel. Each of them has the role to detect different kinds of anomalies depending on the related subset of log systems. The output of these algorithms will be further analysed to infer potential ongoing attacks. In order to better understand the approach description, we first define the following terms:

- **Log**: any type of information, such as raw text lines or numbers, as used in this paper, from any considered type of log devices.
- **Event**: the result of the pre-processing of one or more logs that identify an occurrence within a single log system.
- **Anomaly**: the output of a single ML algorithm that can take into account events from one or more log systems.
- **Alarm**: the signalling of a risk situation within the monitored area due to the contemporary presence of one or multiple anomalies that identify a potential ongoing attack.

The logical architecture of the proposed approach is shown in Figure 6.3. The proposed solution works in real-time. Each time a log is collected by a log system, it is sent to a related pre-processing block. This phase is particularly important to mitigate the effects of errors during the log phase that can occur depending on the used technology. For example, many systems for access control based on RFID register spurious events. The pre-processing phase is fundamental to transform raw logs into events that are here defined such as the stored information regarding really happened occurrences that are meaningful for security monitoring. In order to take into account all the events that occur within the plant, the proposed solution builds a representation of the current working conditions of the whole plant, which we call state vector. The state vector stores all the considered information, such as the physical or remote accesses of each employee through the time of the last transit or his/her actual position inside the plant or within a specific room. Each time a new event is

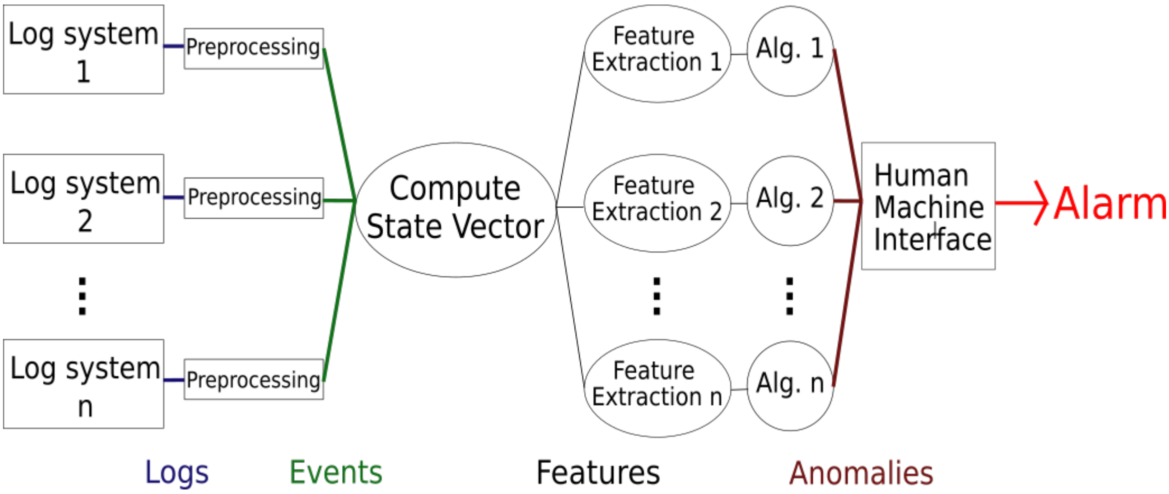

Figure 6.3 Logical Architecture of the proposed approach

processed, the state vector is updated so that it is a concise real-time representation of the plant.

The whole state vector cannot directly feed the ML algorithms. The events which contribute creating the state vector typically contain a lot of information and just a part of them is useful for the considered task. Some specific information, called features, is extracted from the state vector and used to feed the ML algorithms. Multiple algorithms run in parallel and analyse different subsets of features. Every time a new event is processed, the state vector is updated, and a set of features, contained in the feature vector, is extracted and sent to the related ML algorithms. Different types of events can trigger different ML algorithms. ML algorithms detect specific abnormal situations, if any, that are signalled to the human operators through a proper HMI which makes clear the kind of detected anomaly and the related triggered event(s). Finally, human operators have all the information to decide if detected anomalies are false alarms or if they could represent a real risky situation, starting the required countermeasures to properly manage it.

6.3.2 Developed Testbed

We considered three different log systems: the physical access to the power plant through the perimeter gate, the physical access to the TLC room, and the remote access to the SCADA server through VPN. We also considered two parallel ML algorithms: the first one focuses on the possible anomalies related to the TLC room access while the second one on the possible anomalies related to the power plant access. The overall implementation of the proposed architecture is shown in Figure 6.4. All the code has been written in python, and the modules are detailed below.

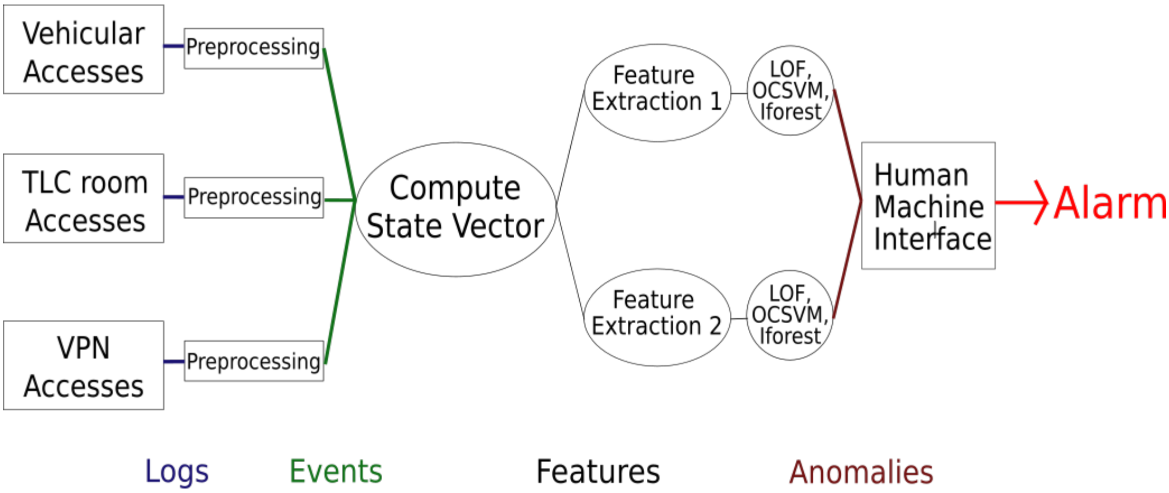

Figure 6.4 Implementation of the proposed architecture

The first module of the algorithm pre-processes the data. We used an input dataset made of data generated by the log monitoring systems of a real power plant in a period of three months. The log systems have been set up and managed by different external companies and have not been designed to allow correlating their generated data. The access log system of the power plant gate identifies authorised users through first and last name or car registration number; the access log system of the TLC room identifies authorised users through first and last name or employee ID; the access log system of the VPN identifies authorised users through an identifier which can be related to a single person or to a company. Since identifying the same person through these three log systems may not be straightforward, we pre-processed the overall log dataset to solve this user identification issue and to remove some log errors, such as double entries.

Some specific features have been designed for the available data. The related input features are reported and described in Table 6.1 and Table 6.2.

We considered three widespread anomaly detection algorithms: Local Outlier Factory (LOF), Isolation Forest (IForest), and One Class Support Vector Machine (OCSVM). One-Class Support Vector Machine is a natural extension of the support vector algorithm to the case of unlabelled data. It consists of a discriminant function that takes the value +1 in a small region that captures the majority of the data points of a set and -1 outside (110). LOF is based on a concept of a local density, where locality is given by k nearest neighbours whose distance is used to estimate the density. It is possible to identify regions of similar density

Table 6.1 TLC room + VPN access related features

Name	Description
Weekday	integer value related to the day of the week when the event occurs (from 0: Monday to 6: Sunday)
Hour	time (in 24h format) when the event takes place
TLC access error	binary value that is 1 if the system rejects a person access to the TLC room or 0 if the access request is accepted
Direction	binary feature whose value is 1 if the user enters the room or -1 if he/she leaves the room
TLC presence	integer value which is increased by 1 when the user gets in the TLC room and decreased by 1 if he/she gets out. Normally, it will be 0 or 1. In case of multiple accesses, i.e. when the log system records two entrances of the same user without recording an exit between the first and the second entrance, this feature takes the value 2. It can take higher values in case of further consecutive entrances. To avoid keeping an incorrect offset, i.e. to avoid that an anomaly goes on affecting the value of this feature after it has been identified, this value is reset under certain conditions. For example, after two following entrances of the same user, it takes the value 2 for that user, but after a first single exit the value is reduced to 0 instead of 1.
Power plant presence	integer value which is increased by 1 when the user enters the power plant and decreased by 1 if he/she leaves the power plant. This feature can behave as the "TLC presence" feature
VPN presence	integer value which is increased by 1 when the user gets remote access to the SCADA server and decreased by 1 if he/she logs out. This feature can behave as the "TLC presence" feature.

Table 6.2 Power plant + VPN access related features

Name	**Description**
Weekday	in Table 6.1
Hour	in Table 6.1
Power plant access error	binary value that is 1 if the system rejects a person access to the power plant or 0 if the access request is accepted
Direction	binary feature whose value is 1 if the user enters the power plant or -1 if he/she leaves the power plant
Power plant presence	integer value which is increased by 1 when the user enters the power plant and decreased by 1 if he/she leaves the power plant. Normally, it will be 0 or 1. In case of multiple accesses, i.e. when the log system records two entrances of the same user without recording an exit between the first and the second entrance, this feature takes value 2. It can take higher values in case of further consecutive entrances. To avoid keeping an incorrect offset, as described in the previous table, this value is reset under certain conditions.
VPN presence	integer value which is increased by 1 when the user gets remote access to the SCADA server and decreased by 1 if he/she logs out. This feature can behave as the "Power plant presence" feature.

that have a substantially lower density than their neighbours by comparing the local density of an object to the local densities of its neighbours, considered as outliers (111). Isolation Forest is an algorithm based on decision trees that explicitly identify anomalies instead of profiling normal data points. Anomalous instances in a dataset can be easier separated from the rest of the samples (isolate) than normal points by using the Isolation Forest algorithm. To isolate a data point, the algorithm recursively generates partitions of the sample by randomly selecting an attribute and then randomly selecting a split value for the attribute between the minimum and maximum values allowed for that attribute (112). All the described algorithms have been implemented by using the SciKit Learn library (59). To train the ML algorithms, we used the described dataset which does not contain any event that should be classified as anomalous, while to test them we used a test set containing different possible anomalies.

It is necessary to highlight that, unlike other types of analysis in the field of cyber security, such as malware analysis by network traffic in which it is clearly defined what portion of the traffic is related to the malware and, therefore, malicious, in the considered scenario there are no past examples of complex cyber-physical attacks, and, consequently, it is not possible to use an already available attack scenario. In this sense, we have imagined five kinds of possible anomalous events that deviate significantly from the log patterns considered normal and can be practically related to situations of risk:

- access of an employee in an unusual time and/or day to the TLC room;
- unusual VPN accesses from users already physically present in the power plant or the TLC room;
- multiple access of an employee to the TLC room;
- multiple access of an employee to the power plant;
- access of an employee to the TLC room without previously entering the power plant from the access gate.

In the field of cyber security of industrial systems, there are very few event descriptions available (and even less for the public domain) regarding complex cyber-physical attacks. For these reasons, the performance evaluation shown in this paper is not related to the ability of the proposed solution to detect ongoing attacks but to detect possible situations of potential risk. The test set is composed of 97.7 % of normal events (Negative events) and 2.3 % of anomalies (Positive events).

We included two sequence diagrams that explain more in detail our software implementation of the proposed approach and the interactions among the blocks in Figure 6.4. Figure 6.5 shows the complete end-to-end interactions among the components considering only one of the log sources and the related feature extraction block, while Figure 6.6 shows the complete view with all the three available information sources and the two feature extraction blocks. Looking at Figure 6.5, it can be noticed that a state vector computation is triggered every time a new log is registered (in this case, a new power plant access log). This action triggers in turn the extraction of a new feature vector from the related feature extraction block. A following classification takes place in order to identify if the current system state is normal or not by using one of the considered ML algorithm (in this case, the LOF algorithm). If an anomaly is detected, an alarm is triggered, otherwise the system comes back in the idle state waiting to the following log. If a log is received while the system is already processing the previous log, the related event is queued within the Compute State Vector block waiting to be considered for the following state vector computation. Looking at Figure 6.6, it can be noticed that both power plant access and TLC room access logs trigger a state vector computation and a following feature extraction only of the related feature extraction block. VPN access logs, instead, trigger the state vector computation and a following feature extraction in both implemented feature extraction blocks, considering that some anomalies, such as the considered "unusual VPN access from users already physically present in the power plant or the TLC room", correlate the VPN access data with both power plant and TLC room access data.

6.3.3 Results

The results obtained by using each of the three considered ML algorithms are shown through the confusion matrices reported in Tables 6.3 for LOF, 6.4 for Isolation Forest, and 6.5 for OCSVM.

Table 6.3 Confusion Matrix - LOF algorithm

		Predicted	
		Positive	Negative
Real	Positive	2.3 %	0 %
	Negative	1.9 %	95.8 %

LOF detects 100% anomalies and interprets correctly most normal events. Isolation Forest performance is not satisfying, in particular to detect anomalies, as well as OCSVM. These behaviours are even clearer by comparing the three ML algorithms through three

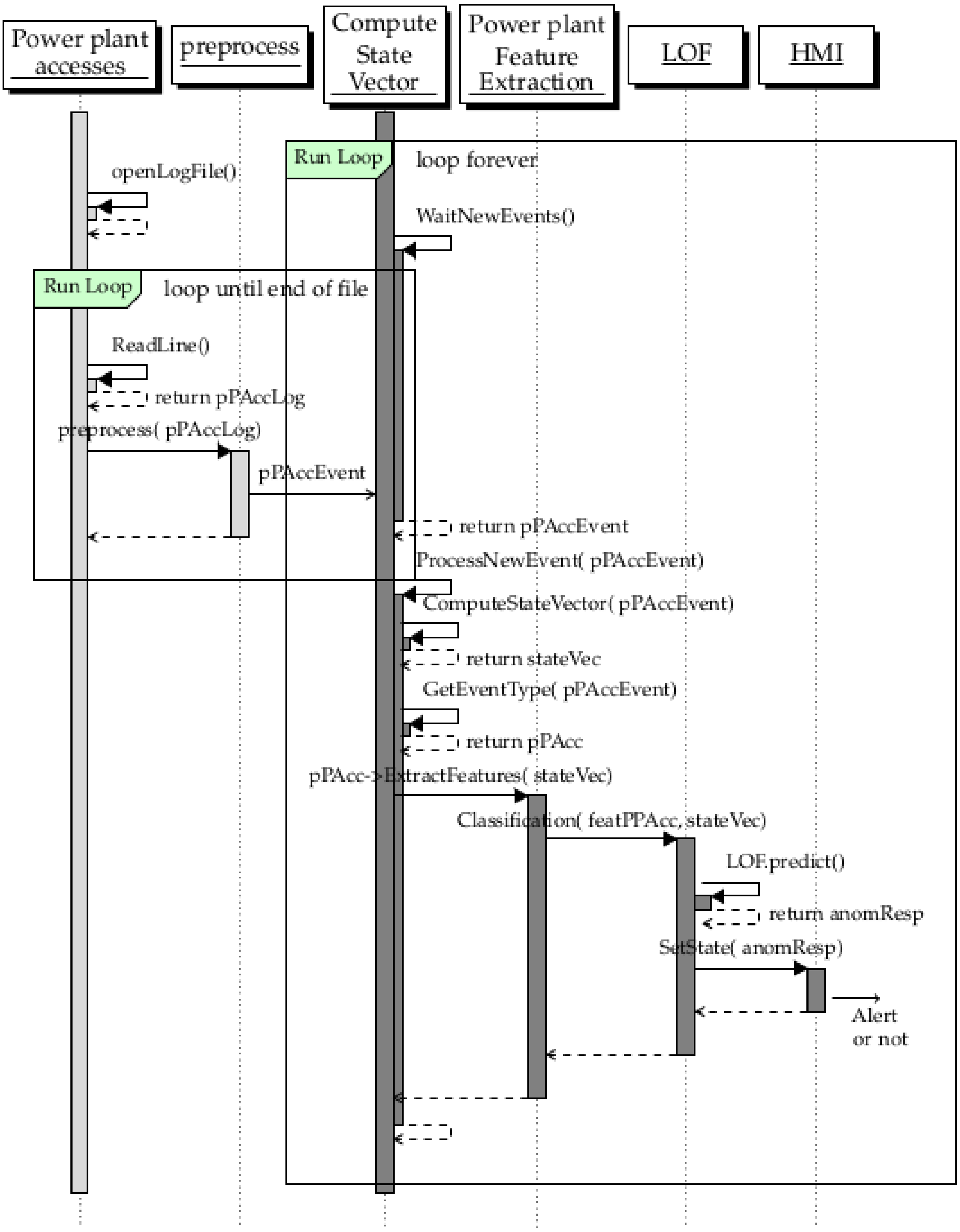

Figure 6.5 Sequence diagram showing the system operations when a new log (a power plant access log, as an example) is registered.

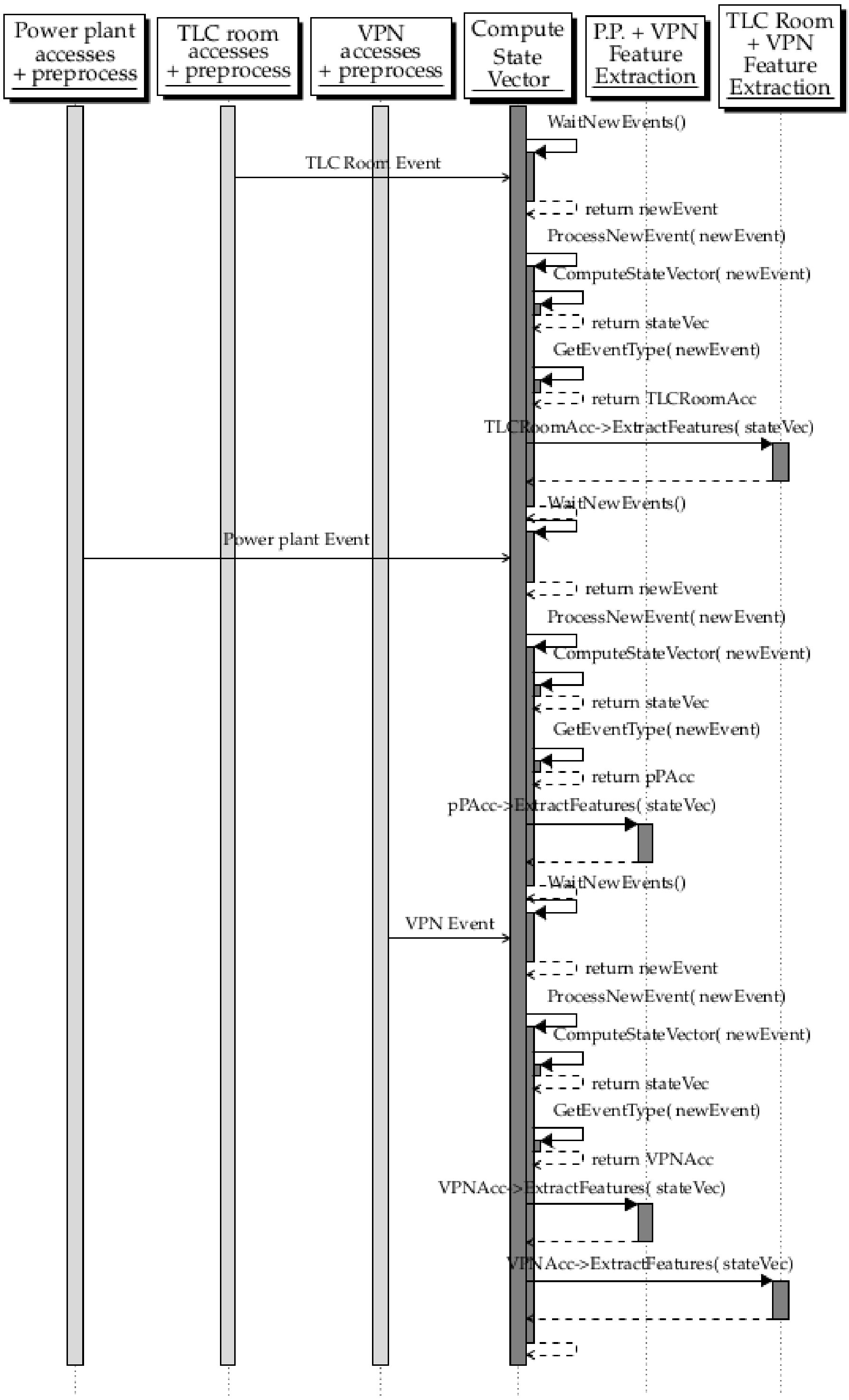
Power plant accesses + preprocess
TLC room accesses + preprocess
VPN accesses + preprocess
Compute State Vector
P.P. + VPN Feature Extraction
TLC Room + VPN Feature Extraction
WaitNewEvents()
TLC Room Event
return newEvent
ProcessNewEvent(newEvent)
ComputeStateVector(newEvent)
return stateVec
GetEventType(newEvent)
return TLCRoomAcc
TLCRoomAcc->ExtractFeatures(stateVec)
WaitNewEvents()
Power plant Event
return newEvent
ProcessNewEvent(newEvent)
ComputeStateVector(newEvent)
return stateVec
GetEventType(newEvent)
return pPAcc
pPAcc->ExtractFeatures(stateVec)
WaitNewEvents()
VPN Event
return newEvent
ProcessNewEvent(newEvent)
ComputeStateVector(newEvent)
return stateVec
GetEventType(newEvent)
return VPNAcc
VPNAcc->ExtractFeatures(stateVec)
VPNAcc->ExtractFeatures(stateVec)

Figure 6.6 Sequence diagram showing the complete first part of the implemented system with three log sources and two extraction blocks

Table 6.4 Confusion Matrix - Isolation Forest algorithm

		Predicted	
		Positive	Negative
Real	Positive	1.3 %	1 %
	Negative	10.2 %	87.5 %

Table 6.5 Confusion Matrix - OCSVM algorithm

		Predicted	
		Positive	Negative
Real	Positive	1 %	1.3 %
	Negative	7 %	90.7 %

metrics commonly used in ML, i.e. Accuracy, Sensitivity, and Specificity, which are defined as in Equations (6.1), (6.2), and (6.3), respectively:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{6.1}$$

$$Sensitivity = \frac{TP}{TP+FN} \tag{6.2}$$

$$Specificity = \frac{TN}{TN+FP} \tag{6.3}$$

where TP: True Positive, TN: True Negative, FP: False Positive, and FN: False Negative.

The obtained results are reported in Table 6.6.

Table 6.6 Comparison among the considered ML algorithms

	LOF	**IForest**	**OCSVM**
Accuracy	98.1 %	88.7 %	91.7 %
Sensitivity	100 %	57.1 %	42.9 %
Specificity	98 %	89.5 %	92.8 %

LOF offers the best performance. Even if the difference could not seem so significant by looking at the Accuracy results, actually it is. Considering the unbalanced number of positive and negative samples in the test set, even a system that detects no anomalies obtains an accuracy of 97.7 %, i.e. the percentage of negative samples in the test set. Isolation

Forest and OCSVM are unable to efficiently recognise anomalies. They also misclassify some negative events and identify them as anomalies. Sensitivity and Specificity results emphasise this. We also tried to properly set the configuration parameters of both algorithms by analysing if their set-up significantly affects the obtained performance. We found out that higher Sensitivity could have been obtained by using different parameter configurations for both Isolation Forest and OCSVM, but with a consequent lower specificity and vice versa, so without improving the overall performance.

Results of this works have been published in (113).

6.3.4 Discussion and Future Developments

Results are hardly comparable with the works in the state of the art. The works in the literature significantly differ both from the use case and type of considered data. Still, some considerations can be done. Rule-based approaches are of course useful to detect specific behaviours and working conditions that can be considered dangerous. Nevertheless, they require high customisation on the considered environment that can be extremely variable in terms of physical structure, network architecture, log systems, user behaviour, among other factors. For these reasons, Machine Learning approaches can be very helpful to address these issues. The results presented above are, although preliminary, really promising. Besides, our proposed system:

- is able to detect possible anomaly situations or attacks that cannot be detected by the traditional security mechanisms thanks to the joint use of multiple information sources;
- is more robust against actions aim to break the security systems (e.g., if stealing a badge may be enough to let a malicious person enter a power plant, it would be less easy if we have multiple security systems to corrupt);
- offers an automatic tool able to correlate data generated by heterogeneous sources thanks to its ML core, i.e., thanks to ML algorithms able to effectively identify both known and possible unknown anomalies exploiting hidden information inside the typically huge amount of raw data.
- supports human security officers that could be distracted by the huge amount of available data or by other events taking place within the monitored area.

The main limitation of the present work is certainly the need of simulating datasets related to attacks. Since we did not dispose of real data about real attacks on scenarios such as industrial plants, we had to generate the related data attack traces through a simulation environment. That condition is unfortunately common for this type of research since industries will hardly release such type of information. To relieve this limitation, it would be useful to proceed with interviews to power plan operators, allowing researchers to dispose of a set of possible complex attack patterns to test the proposed solutions.

Another interesting future development of this work would be the inclusion of logs belonging to a higher number of different systems that could allow having more accurate information even if, in some cases, at the cost of higher redundancy. For example, from the physical world, data from intelligent camera systems processed with image recognition algorithms could provide useful information to relate with other log systems. In this way, the presence of a person in a room could be related to the data from cameras, access log through badges, accounts in use on the room's terminals, and use of the user's IP address. Such a system will be more robust to attacks that have to compromise multiple systems to enter in action and keep working undetected.

Chapter 7

Conclusions

The present work presented some novel approaches for Machine Learning based Anomaly Detection algorithms for cybersecurity of ICS. After a brief introduction of the cybersecurity issues in Industrial Control Systems and an overview of the state of the art regarding cybersecurity monitoring of ICS, three Machine Learning approaches, which focus on different layers of the control network architecture, have been proposed.

The first one focuses on covert channels based on DNS protocol. As discussed in Chapter 4, the sophistication of attacks is increasing, so that the monitoring systems of network traffic have to evolve in order to face new threats. Covert channels may be particularly dangerous in ICS, since they may be used to establish a command and control connection with the outside, allowing attackers to remotely send commands to a compromised host. Many strategies can be used to establish a covert channel; one of the most interesting is based on the DNS protocol, since many firewalls do not implement a deep packet inspection on DNS. Many algorithms have been presented in the literature. Nevertheless, even a small modification in the code of tools used for establishing the covert channel (or even a change of a single parameter, like the length of the query) may lead to the failure of these algorithms. In particular, each approach performs even very well on specific tools, but not on others. The basic idea has been to run different algorithms in parallel, building an ensemble classifier; still, this task presents different problems, since each algorithm produces different outputs. An architecture, able to merge the output of different state of the art algorithms by grouping them in families has been presented. Results show how the architecture performs better than single algorithms. The implemented testbed includes only a small amount of algorithms; future developments may enlarge the number of algorithms. In order to further improve the performances of the architecture, it is necessary to develop a strategy for optimizing the architecture, keeping into account ML training parameters, features of single algorithms and the final decisor parameters.

The following Chapter focuses on the field layer of the control network, and in particular on the electrical power system. The field layer is the portion of the control network which is utilized to collect information from sensors and send commands to the actuators; in electrical distribution networks these commands may have very stringent latency constraints. For these reasons, many technologies for cybersecurity that are used in IT networks cannot be directly applicable in these networks. In the State of the art section, the use of traditional Network Intrusion Detection Systems, but also novel approaches for Host Intrusion Detection Systems have been discussed. Power systems utilize different control systems which threaten the functioning of the grid, each one presenting its own peculiarities. One of the systems which result particularly dangerous for the power system if attacked is Distributed Energy Resources. The present work proposed a physics-based anomaly detection algorithm for DERs. Physics-Based Anomaly detection is an innovative field of research, which is based on the observation that the industrial processes have to follow physics rules, so that their behavior is predictable. The rationale of using physics based AD in a DERs environment is that it would be useful to replace the supervision of human operators on the physic behavior of these systems. The proposed algorithm exploits the capabilities of a particular neural network called autoencoder, with the aim to detect both malicious commands and fake measurements sent to the controllers; also, it can be useful to notice some types of faults. In order to validate the approach, a simulation environment of a photovoltaic system connected to the grid has been developed on the MATLAB/Simulink software. Results are really promising, so that the approach can be applied to other DERS, such as Battery Electrical Storage Systems.

The third proposed approach tries to integrate information belonging to domains which traditionally work separately: cyber security and physical security. The integration of these two systems is growing in importance: it's in fact well known that many cyberattacks start from the compromise of physical countermeasurements; one famous example is represented by USB pens left outside an industrial plant, hoping that some employees insert them in a computer within the network. Also, a cyberattack may try to turn off physical security controls, steal identity or falsify logs. The problem of integration is basically managing complexity: in fact, an industrial plant may include several log systems related to both cyber and physical security, resulting in a huge amount of information. These log usually neither shares a common standard for storing information. The proposed architecture is based on the definition of a state vector, that is a vector that aims to resume the operating condition of the plant, like the number of people physically present or remotely connected, the alarms raised from intrusion detection systems or firewalls, measures from environmental sensors and so on. Each time a new event occurs, the state vector is updated. It is not useful to correlate every single information of the state vector: this would explode the dimension of the input of machine learning algorithms, lowering the performances. Instead, different subsets of information can be extracted from the state vector, and analyzed through ML. A use case of a hydroelectric power plant has been used, using a simple implementation of the architecture with a limited number of log systems in order to quickly obtain results and iterate the process of investigation, since there are no similar works in the state of the art. In the performance evaluation, data belonging to a real power plant has been utilized, showing promising results. The work can be expanded in several ways, first of all increasing the number of considered log systems, but also investigating a new set of features that can be extracted from the state vector.

In conclusion, Anomaly detection is a promising field of research for Industrial Control System Cybersecurity. While attacks are growing in complexity, and IDS has to evolve to face new threats, innovative fields of research are those which keep into account physical properties, both from the perspective of physical security and from the physical measurement of the industrial process itself. These three domains, the cyber, the physical, and the cyber-physical can, and should, work together for an effective cybersecurity monitoring of critical infrastructures.

References

[1] Steven M Rinaldi, James P Peerenboom, and Terrence K Kelly. Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE control systems magazine*, 21(6):11–25, 2001.

[2] Frances Cleveland and Annabelle Lee. Cyber security for der systems. *National Electric Sector Cybersecurity Organization Resource. Electric Power Research Institute (EPRI)*, 2013.

[3] Celia Paulsen. Glossary of key information security terms. Technical report, National Institute of Standards and Technology, 2018.

[4] R Setola and Rapporto di Ricerca. La strategia globale di protezione delle infrastrutture e risorse critiche contro gli attacchi terroristici. *Centro Militare di Studi Strategici CEMISS, http://www. masterhomelandsecurity. eu/wp-content/uploads/2012/08/Protezioneinfrastrutture-e-risorse-critiche_Setola. pdf*, 2011.

[5] David Kushner. The real story of stuxnet. *ieee Spectrum*, 50(3):48–53, 2013.

[6] Nicolas Falliere, Liam O Murchu, and Eric Chien. W32. stuxnet dossier. *White paper, Symantec Corp., Security Response*, 5(6):29, 2011.

[7] Suzanne Lightman Marshall Abrams Adam Hahn Keith Stouffer, Victoria Pillitteri. Guide to industrial control systems (ics) security. Technical Report NIST Special Publication (SP) 800-82, Rev. 2, Includes updates as of May, 2015, National Institute of Standards and Technology, 2015.

[8] Edward JM Colbert and Alexander Kott. *Cyber-security of SCADA and other industrial control systems*, volume 66. Springer, 2016.

[9] Theodore J Williams. The purdue enterprise reference architecture. *Computers in industry*, 24(2-3):141–158, 1994.

[10] Department of Homeland Security. Recommended practice: Improving industrial control system cybersecurity with defense-in-depth strategies. Technical report, National Institute of Standards and Technology. Accessed: 2021-09-30.

[11] Joint Task Force. Security and privacy controls for information systems and organizations. Technical report, National Institute of Standards and Technology, 2017.

[12] Thomas M Chen and Saeed Abu-Nimeh. Lessons from stuxnet. *Computer*, 44(4):91–93, 2011.

[13] Boldizsár Bencsáth, Gábor Pék, Levente Buttyán, and Mark Felegyhazi. The cousins of stuxnet: Duqu, flame, and gauss. *Future Internet*, 4(4):971–1003, 2012.

[14] Sharifah Yaqoub A. Fayi. What petya/notpetya ransomware is and what its remidiations are. In Shahram Latifi, editor, *Information Technology - New Generations*, pages 93–100, Cham, 2018. Springer International Publishing.

[15] M Satheesh Kumar, Jalel Ben-Othman, and KG Srinivasagan. An investigation on wannacry ransomware and its detection. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2018.

[16] Jesse M Ehrenfeld. Wannacry, cybersecurity and health information technology: A time to act. *Journal of medical systems*, 41(7):104, 2017.

[17] Defense Use Case. Analysis of the cyber attack on the ukrainian power grid. *Electricity Information Sharing and Analysis Center (E-ISAC)*, 388, 2016.

[18] R Lee. Trisis malware: analysis of safety system targeted malware. *Dragos Inc*, 2017.

[19] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2002.

[20] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[21] Andrew Glassner. Deep learning: From basics to practice. *The Imaginary Institute*, 2018.

[22] Slavica V Boštjančič Rakas, Mirjana D Stojanović, and Jasna D Marković-Petrović. A review of research work on network-based scada intrusion detection systems. *IEEE Access*, 8:93083–93108, 2020.

[23] Yi Yang, Hai-Qing Xu, Lei Gao, Yu-Bo Yuan, Kieran McLaughlin, and Sakir Sezer. Multidimensional intrusion detection system for iec 61850-based scada networks. *IEEE Transactions on Power Delivery*, 32(2):1068–1078, 2016.

[24] Hyunguk Yoo and Taeshik Shon. Novel approach for detecting network anomalies for substation automation based on iec 61850. *Multimedia Tools and Applications*, 74(1):303–318, 2015.

[25] Ming Liu, Zhi Xue, Xianghua Xu, Changmin Zhong, and Jinjun Chen. Host-based intrusion detection system with system calls: Review and future trends. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[26] Cyntia Vargas Martinez and Birgit Vogel-Heuser. A host intrusion detection system architecture for embedded industrial devices. *Journal of The Franklin Institute*, 2019.

[27] Cyntia Vargas, Michael Langfinger, and Birgit Vogel-Heuser. A tiered security analysis of industrial control system devices. In *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, pages 399–404. IEEE, 2017.

[28] Marco AF Pimentel, David A Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[29] Jairo Giraldo, David Urbina, Alvaro Cardenas, Junia Valente, Mustafa Faisal, Justin Ruths, Nils Ole Tippenhauer, Henrik Sandberg, and Richard Candell. A survey of physics-based attack detection in cyber-physical systems. *ACM Computing Surveys (CSUR)*, 51(4):1–36, 2018.

[30] Sebastian Zander, Grenville Armitage, and Philip Branch. A survey of covert channels and countermeasures in computer network protocols. *IEEE Communications Surveys & Tutorials*, 9(3):44–57, 2007.

[31] Paul V Mockapetris. RFC 1034: Domain names - Concepts and Facilities, 1987.

[32] Paul V Mockapetris. RFC 1035: Domain names - Implementation and Specification, 1987.

[33] Christian J Dietrich, Christian Rossow, Felix C Freiling, Herbert Bos, Maarten Van Steen, and Norbert Pohlmann. On Botnets that use DNS for Command and Control. *European Conference on Computer Network Defence*, pages 9–16, 2011.

[34] Asaf Nadler, Avi Aminov, and Asaf Shabtai. Detection of malicious and low throughput data exfiltration over the DNS protocol. *Computers & Security*, 80:36–53, 2019.

[35] Maurizio Aiello, Maurizio Mongelli, and Gianluca Papaleo. DNS tunneling detection through statistical fingerprints of protocol messages and machine learning. *International Journal of Communication Systems*, 28(14):1987–2002, 2015.

[36] Enrico Cambiaso, Maurizio Aiello, Maurizio Mongelli, and Gianluca Papaleo. Feature transformation and Mutual Information for DNS tunneling analysis. *International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 957–959, 2016.

[37] Maurizio Aiello, Maurizio Mongelli, Enrico Cambiaso, and Gianluca Papaleo. Profiling DNS tunneling attacks with PCA and mutual information. *Logic Journal of the IGPL*, 24(6):957–970, 2016.

[38] Maurizio Aiello, Maurizio Mongelli, Marco Muselli, and Damiano Verda. Unsupervised learning and rule extraction for Domain Name Server tunneling detection. *Internet Technology Letters*, 2(2):1–6, 2019.

[39] Saeed Shafieian, Daniel Smith, and Mohammad Zulkernine. Detecting DNS tunneling using ensemble learning. *International Conference on Network and System Security*, pages 112–127, 2017.

[40] Jawad Ahmed, Hassan Habibi Gharakheili, Qasim Raza, Craig Russell, and Vijay Sivaraman. Monitoring Enterprise DNS Queries for Detecting Data Exfiltration from Internal Hosts. *IEEE Transactions on Network and Service Management*, 17(1):265–279, 2020.

[41] Mahmoud Sammour, Burairah Hussin, and Mohd Fairuz Iskandar Othman. Comparative Analysis for Detecting DNS Tunneling Using Machine Learning Techniques. *International Journal of Applied Engineering Research*, 12(22):12762–12766, 2017.

[42] Anirban Das, Min-Yi Shen, Madhu Shashanka, and Jisheng Wang. Detection of exfiltration and tunneling over DNS. *International Conference on Machine Learning and Applications (ICMLA)*, pages 737–742, 2017.

[43] Kemeng Wu, Yongzheng Zhang, and Tao Yin. FTPB: A Three-stage DNS Tunnel Detection Method Based on Character Feature Extraction. *International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 250–258, 2020.

[44] Chang Liu, Liang Dai, Wenjing Cui, and Tao Lin. A Byte-level CNN Method to Detect DNS Tunnels. *International Performance Computing and Communications Conference (IPCCC)*, pages 1–8, 2019.

[45] Franco Palau, Carlos Catania, Jorge Guerra, Sebastian Garcia, and Maria Rigaki. DNS Tunneling: A Deep Learning based Lexicographical Detection Approach. *arXiv preprint arXiv:2006.06122*, 2020.

[46] Chia-Min Lai, Bo-Ching Huang, Shin-Ying Huang, Ching-Hao Mao, and Hahn-Ming Lee. Detection of DNS Tunneling by Feature-Free Mechanism. *Conference on Dependable and Secure Computing (DSC)*, pages 1–2, 2018.

[47] Kemeng Wu, Yongzheng Zhang, and Tao Yin. TDAE: Autoencoder-based Automatic Feature Learning Method for the Detection of DNS tunnel. *International Conference on Communications (ICC)*, pages 1–7, 2020.

[48] Kenton Born and David Gustafson. Detecting DNS tunnels using character frequency analysis. *arXiv preprint arXiv:1004.4358*, 2010.

[49] Meng Luo, Qiuyun Wang, Yepeng Yao, Xuren Wang, Peian Yang, and Zhengwei Jiang. Towards Comprehensive Detection of DNS Tunnels. *Symposium on Computers and Communications (ISCC)*, pages 1–7, 2020.

[50] Jingkun Liu, Shuhao Li, Yongzheng Zhang, Jun Xiao, Peng Chang, and Chengwei Peng. Detecting DNS tunnel through binary-classification based on behavior features. *Trustcom/BigDataSE/ICESS*, pages 339–346, 2017.

[51] Zhao Yang, Ye Hongzhi, Li Lingzi, Huang Cheng, and Zhang Tao. Detecting DNS Tunnels Using Session Behavior and Random Forest Method. *International Conference on Data Science in Cyberspace (DSC)*, pages 45–52, 2020.

[52] Constantinos Patsakis, Fran Casino, and Vasilios Katos. Encrypted and covert DNS queries for botnets: Challenges and countermeasures. *Computers & Security*, 88:101614, 2020.

[53] Anna L Buczak, Paul A Hanke, George J Cancro, Michael K Toma, Lanier A Watkins, and Jeffrey S Chavis. Detection of tunnels in PCAP data by random forests. *Annual Cyber and Information Security Research Conference*, pages 1–4, 2016.

[54] Iodine release 0.7.0: A software to tunnel ipv4 data through dns servers. https://code.kryo.se/iodine/, 2014. Accessed on 2021-10-01.

[55] Dnscat2: A software for dns tunnel. https://github.com/iagox86/dnscat2, 2020. Accessed on 2021-10-01.

[56] Dnshell v1.7: A python reverse shell that uses dns as c2 channel. https://github.com/ahhh/Reverse_DNS_Shell, 2015. Accessed on 2021-10-01.

[57] Ozyman: Dns tunneling made easy. https://github.com/splitbrain/dnstunnel, 2010. Accessed on 2021-10-01.

[58] Tcp-over-dns tunnel software howto. http://analogbit.com/2008/07/27/tcp-over-dns-tunnel-software-howto/, 2011. Accessed on 2021-10-01.

[59] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of machine Learning research*, 12:2825–2830, 2011.

[60] A Lee. Electric sector failure scenarios and impact analyses. *National Electric Sector Cybersecurity Organization Resource (NESCOR) Technical Working Group*, 1, 2013.

[61] Giovanni Battista Gaggero, Mario Marchese, Aya Moheddine, and Fabio Patrone. A possible smart metering system evolution for rural and remote areas employing unmanned aerial vehicles and internet of things in smart grids. *Sensors*, 21(5):1627, 2021.

[62] Giovanni Battista Gaggero, Paola Girdinio, and Mario Marchese. Advancements and research trends in microgrids cybersecurity. *Applied Sciences*, 11(16):7363, 2021.

[63] Anna Volkova, Michael Niedermeier, Robert Basmadjian, and Hermann de Meer. Security challenges in control network protocols: A survey. *IEEE Communications Surveys & Tutorials*, 21(1):619–639, 2018.

[64] Abedalsalam Bani-Ahmed, Luke Weber, Adel Nasiri, and Hossein Hosseini. Microgrid communications: State of the art and future trends. In *2014 International Conference on Renewable Energy Research and Application (ICRERA)*, pages 780–785. IEEE, 2014.

[65] Haftu Tasew Reda, Biplob Ray, Pejman Peidaee, Adnan Anwar, Abdun Mahmood, Akhtar Kalam, and Nahina Islam. Vulnerability and impact analysis of the iec 61850 goose protocol in the smart grid. *Sensors*, 21(4):1554, 2021.

[66] BooJoong Kang, Peter Maynard, Kieran McLaughlin, Sakir Sezer, Filip Andrén, Christian Seitl, Friederich Kupzog, and Thomas Strasser. Investigating cyber-physical attacks against iec 61850 photovoltaic inverter installations. In *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–8. IEEE, 2015.

[67] Roman Schlegel, Sebastian Obermeier, and Johannes Schneider. A security evaluation of iec 62351. *Journal of Information Security and Applications*, 34:197–204, 2017.

[68] SM Suhail Hussain, Taha Selim Ustun, and Akhtar Kalam. A review of iec 62351 security mechanisms for iec 61850 message exchanges. *IEEE Transactions on Industrial Informatics*, 16(9):5643–5654, 2019.

[69] Taha Selim Ustun and SM Suhail Hussain. Iec 62351-4 security implementations for iec 61850 mms messages. *IEEE Access*, 8:123979–123985, 2020.

[70] SM Suhail Hussain, Shaik Mullapathi Farooq, and Taha Selim Ustun. Analysis and implementation of message authentication code (mac) algorithms for goose message security. *IEEE Access*, 7:80980–80984, 2019.

[71] Shaik Mullapathi Farooq, SM Suhail Hussain, and Taha Selim Ustun. Performance evaluation and analysis of iec 62351-6 probabilistic signature scheme for securing goose messages. *IEEE Access*, 7:32343–32351, 2019.

[72] Pietro Danzi, Marko Angjelichinoski, Čedomir Stefanović, Tomislav Dragičević, and Petar Popovski. Software-defined microgrid control for resilience against denial-of-service attacks. *IEEE Transactions on Smart Grid*, 10(5):5258–5268, 2018.

[73] Yan Li, Yanyuan Qin, Peng Zhang, and Amir Herzberg. Sdn-enabled cyber-physical security in networked microgrids. *IEEE Transactions on Sustainable Energy*, 10(3):1613–1622, 2018.

[74] Quan Zhou, Mohammad Shahidehpour, Ahmed Alabdulwahab, and Abdullah Abusorrah. A cyber-attack resilient distributed control strategy in islanded microgrids. *IEEE Transactions on Smart Grid*, 11(5):3690–3701, 2020.

[75] Chao Deng, Yu Wang, Changyun Wen, Yan Xu, and Pengfeng Lin. Distributed resilient control for energy storage systems in cyber–physical microgrids. *IEEE Transactions on Industrial Informatics*, 17(2):1331–1341, 2020.

[76] Suman Rath, Diptak Pal, Parth Sarthi Sharma, and Bijaya Ketan Panigrahi. A cyber-secure distributed control architecture for autonomous ac microgrid. *IEEE Systems Journal*, 2020.

[77] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):1–33, 2011.

[78] Alefiya Hussain, John Heidemann, and Christos Papadopoulos. A framework for classifying denial of service attacks. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 99–110, 2003.

[79] Mehdi Hosseinzadeh, Bruno Sinopoli, and Emanuele Garone. Feasibility and detection of replay attack in networked constrained cyber-physical systems. In *2019 57th annual allerton conference on communication, control, and computing (Allerton)*, pages 712–717. IEEE, 2019.

[80] Quan Zhou, Mohammad Shahidehpour, Ahmed Alabdulwahab, Abdullah Abusorrah, Liang Che, and Xuan Liu. Cross-layer distributed control strategy for cyber resilient microgrids. *IEEE Transactions on Smart Grid*, 2021.

[81] Yun Liu, Yuanzheng Li, Yu Wang, Xian Zhang, Hoay Beng Gooi, and Huanhai Xin. Robust and resilient distributed optimal frequency control for microgrids against cyber attacks. *IEEE Transactions on Industrial Informatics*, 2021.

[82] A Chavez, C Lai, Nicholas Jacobs, Shamina Hossain-McKenzie, Christian Birk Jones, J Johnson, and Adam Summers. Hybrid intrusion detection system design for distributed energy resource systems. In *2019 IEEE CyberPELS (CyberPELS)*, pages 1–6. IEEE, 2019.

[83] Donghan Shi, Pengfeng Lin, Yu Wang, Chia-Chi Chu, Yan Xu, and Peng Wang. Deception attack detection of isolated dc microgrids under consensus-based distributed voltage control architecture. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 11(1):155–167, 2021.

[84] Anna Magdalena Kosek. Contextual anomaly detection for cyber-physical security in smart grids based on an artificial neural network model. In *2016 Joint Workshop on Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG)*, pages 1–6. IEEE, 2016.

[85] Fangyu Li, Rui Xie, Bowen Yang, Lulu Guo, Ping Ma, Jianjun Shi, Jin Ye, and WenZhan Song. Detection and identification of cyber and physical attacks on distribution power grids with pvs: An online high-dimensional data-driven approach. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2019.

[86] Devu Manikantan Shilay, Kin Gwn Lorey, Tianshu Weiz, Teems Lovetty, and Yu Cheng. Catching anomalous distributed photovoltaics: An edge-based multi-modal anomaly detection. *arXiv preprint arXiv:1709.08830*, 2017.

[87] Konstantina Fotiadou, Terpsichori Helen Velivassaki, Artemis Voulkidis, Dimitrios Skias, Corrado De Santis, and Theodore Zahariadis. Proactive critical energy infrastructure protection via deep feature learning. *Energies*, 13(10):2622, 2020.

[88] Fouzi Harrou, Bilal Taghezouit, and Ying Sun. Improved k nn-based monitoring schemes for detecting faults in pv systems. *IEEE Journal of Photovoltaics*, 9(3):811–821, 2019.

[89] Paola Costamagna, Andrea De Giorgi, Loredana Magistri, Gabriele Moser, Lissy Pellaco, and Andrea Trucco. A classification approach for model-based fault diagnosis in power generation systems based on solid oxide fuel cells. *IEEE Transactions on Energy Conversion*, 31(2):676–687, 2015.

[90] Anna Magdalena Kosek and Oliver Gehrke. Ensemble regression model-based anomaly detection for cyber-physical intrusion detection in smart grids. In *2016 IEEE Electrical Power and Energy Conference (EPEC)*, pages 1–7. IEEE, 2016.

[91] Emanuele Principi, Damiano Rossetti, Stefano Squartini, and Francesco Piazza. Unsupervised electric motor fault detection by using deep autoencoders. *IEEE/CAA Journal of Automatica Sinica*, 6(2):441–451, 2019.

[92] François Chollet et al. Keras. https://keras.io, 2015.

[93] Giovanni Battista Gaggero, Mansueto Rossi, Paola Girdinio, and Mario Marchese. Neural network architecture to detect system faults/cyberattacks anomalies within a photovoltaic system connected to the grid. In *2019 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, pages 1–4. IEEE, 2019.

[94] Giovanni Battista Gaggero, Mansueto Rossi, Paola Girdinio, and Mario Marchese. Detecting system fault/cyberattack within a photovoltaic system connected to the grid: A neural network-based solution. *Journal of Sensor and Actuator Networks*, 9(2):20, 2020.

[95] Zeeshan Ahmad, Adnan Shahid Khan, Cheah Wai Shiang, Johari Abdullah, and Farhan Ahmad. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150, 2021.

[96] Panagiotis I Radoglou-Grammatikis and Panagiotis G Sarigiannidis. Securing the smart grid: A comprehensive compilation of intrusion detection and prevention systems. *IEEE Access*, 7:46595–46620, 2019.

[97] S Sandeep Sekharan and Kamalanathan Kandasamy. Profiling SIEM tools and correlation engines for security analytics. In *International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 717–721. IEEE, 2017.

[98] Gustavo Gonzalez Granadillo, Mohammed El-Barbori, and Herve Debar. New types of alert correlation for security information and event management systems. In *8th International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–7. IEEE, 2016.

[99] Igor V Kotenko, Dmitry S Levshun, and Andrey A Chechulin. Event correlation in the integrated cyber-physical security system. In *19th International Conference on Soft Computing and Measurements (SCM)*, pages 484–486. IEEE, 2016.

[100] Luigi Coppolino, Salvatore D'Antonio, Valerio Formicola, and Luigi Romano. A framework for mastering heterogeneity in multi-layer security information and event correlation. *Journal of Systems Architecture*, 62:78–88, 2016.

[101] Flavio Frattini, Ugo Giordano, and Vincenzo Conti. Facing Cyber-Physical Security Threats by PSIM-SIEM Integration. In *15th European Dependable Computing Conference (EDCC)*, pages 83–88. IEEE, 2019.

[102] Florian Skopik, Max Landauer, Markus Wurzenberger, Gernot Vormayr, Jelena Milosevic, Joachim Fabini, Wolfgang Prüggler, Oskar Kruschitz, Benjamin Widmann, Kevin Truckenthanner, et al. synERGY: Cross-correlation of operational and contextual data to timely detect and mitigate attacks to cyber-physical systems. *Journal of Information Security and Applications*, 54:102544–102567, 2020.

[103] Hildegard Ferraiolo, Ketan L Mehta, Nabil Ghadiali, Jason Mohler, Vincent Johnson, and Steven Brady. Guidelines for the Use of PIV Credentials in Facility Access. *NIST Special Publication*, 800:1–71, 2018.

[104] G Sreenu and MA Saleem Durai. Intelligent video surveillance: a review through deep learning techniques for crowd analysis. *Journal of Big Data*, 6(1):1–27, 2019.

[105] Cyberattacks on cameras. https://infocondb.org/con/black-hat/black-hat-usa-2013/exploiting-network-surveillance-cameras-like-a-hollywood-hacker, 2013. [Online; accessed 15-July-2021].

[106] Houbing Song, Glenn Fink, and Sabina Jeschke. *Security and privacy in cyber-physical systems*. Wiley, 2017.

[107] David R Miller, Shon Harris, Allen Harper, Stephen VanDyke, and Chris Blask. *Security information and event management (SIEM) implementation*. McGraw Hill Professional, 2010.

[108] Karen Kent and Murugiah Souppaya. Guide to computer security log management. *NIST special publication*, 92:1–72, 2006.

[109] Rainer Gerhards. The Syslog protocol. *RFC 5424*, 2009.

[110] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, pages 582–588, 2000.

[111] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. LOF: identifying density-based local outliers. In *International conference on Management of data*, pages 93–104. ACM, 2000.

[112] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *8^{th} International conference on data mining*, pages 413–422. IEEE, 2008.

[113] Alessandro Fausto, Giovanni Battista Gaggero, Fabio Patrone, Paola Girdinio, and Mario Marchese. Toward the integration of cyber and physical security monitoring systems for critical infrastructures. *Sensors*, 21(21):6970, 2021.