Development of Security Mechanisms for a Multi-Agent Cyber-Physical Conveyor System using Machine Learning

Gustavo Silva Funchal

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Industrial Engineering, in the scope of the double diploma programme with the Federal University of Technology - Paraná.

Work oriented by:

Professor PhD Paulo Leitão

Professor PhD José Barbosa

Professor PhD Marcos Vallim

Bragança

2019-2020

Development of Security Mechanisms for a Multi-Agent Cyber-Physical Conveyor System using Machine Learning

Gustavo Silva Funchal

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Industrial Engineering, in the scope of the double diploma programme with the Federal University of Technology - Paraná.

Work oriented by:

Professor PhD Paulo Leitão

Professor PhD José Barbosa

Professor PhD Marcos Vallim

Bragança

2019-2020

Dedication

I dedicate this work to my parents who helped me get where I am today and also to my girlfriend who gave me all the supports to finish this work.

Acknowledgement

Here I would like to thank everyone who contributed to the development of this work.

Firstly, I want to thank organizers of the Double Degree program for supporting the research and collaboration between IPB and UTFPR and giving the opportunity to the students to have this amazing experience.

I want to thank my supervisors at IPB, Professor PhD Paulo Leitão and Professor PhD José Barbosa, and also Professor PhD Tiago Pedrosa, for guidance, knowledge transmitted, opportunity, motivation and advice. I'm very lucky to have all these people on my way who gave me a lot to take away with me.

Also, I want to thank my supervisor at UTFPR, Professor PhD Marcos Vallim, for the opportunity to participate in the Double Degree program.

I want to thank my laboratory colleagues, more especially Jonas Queiroz and Lucas Sakurada for sharing experiences and knowledge.

I want to thank my girlfriend Victória Melo for all the support, advice, help and especially for making this work lighter with her contagious joy.

Finally, to my family, especially to my parents who gave me all the support, encouragement and advice to achieve my goals. Nothing could be done without them.

Abstract

One main foundation of the Industry 4.0 is the connectivity of devices and systems using Internet of Things technologies, where Cyber-physical systems (CPS) act as the backbone infrastructure based on distributed and decentralized structures. CPS requires the use of Artificial Intelligence (AI) techniques, such as Multi-Agent Systems (MAS), allowing the incorporation of intelligence into the CPS through autonomous, proactive and cooperative entities. The adoption of this new generation of systems in the industrial environment opens new doors for various attacks that can cause serious damage to industrial production systems.

This work presents the development of security mechanisms for systems based on MAS, where these mechanisms are used in an experimental case study that consists of a multi-agent cyber-physical conveyor system. For this purpose, simple security mechanisms were employed in the system, such as user authentication, signature and message encryption, as well as other more complex mechanisms, such as machine learning techniques that allows the agents to be more intelligent in relation to the exchange of messages protecting the system against attacks, through the classification of the messages as reliable or not, and also an intrusion detection system was carried out.

Based on the obtained results, the efficient protection of the system was reached, mitigating the main attack vectors present in the system architecture.

Keywords: Industry 4.0; Multi-agent systems; Cyber-physical systems; Cybersecurity; Machine Learning; Intrusion Detection Systems.

Resumo

Uma das principais bases da Indústria 4.0 é a conectividade de dispositivos e sistemas utilizando as tecnologias da Internet das Coisas, onde os sistemas ciber-físicos atuam como a infraestrutura principal com base em estruturas distribuídas e descentralizadas. Os sistemas ciber-físicos requerem o uso de técnicas de Inteligência Artificial, como por exemplo, Sistemas Multi-Agentes, permitindo a incorporação de inteligência nos sistemas ciber-físicos através de entidades autônomas, proativas e cooperativas. A adoção dessa nova geração de sistemas no ambiente industrial abre novas portas para vários ataques que podem causar sérios danos aos sistemas de produção industrial.

Este trabalho apresenta o desenvolvimento de mecanismos de segurança para sistemas baseados em sistemas multi-agentes, em que esses mecanismos são utilizados em um caso de estudo experimental que consiste em um sistema de transporte ciber-físico baseado em sistemas multi-agentes. Para isso, mecanismos simples de segurança foram empregados no sistema, como autenticação do usuário, assinatura e criptografia de mensagens, além de outros mecanismos mais complexos, como técnicas de aprendizagem de máquina, que permite que os agentes sejam mais inteligentes em relação à troca de mensagens, protegendo o sistema contra ataques, através da classificação das mensagens como confiáveis ou não, e também foi realizado um sistema de detecção de intrusões.

Com base nos resultados obtidos, obteve-se uma proteção eficiente do sistema, mitigando os principais vetores de ataque presentes na arquitetura do sistema.

Palavras-chave: Indústria 4.0; Sistema multi-agente; Sistema ciber-físico; Aprendizagem de máquina; Sistema de detecção de intrusão.

Contents

List of Tables

List of Figures

Acronyms

ACL Agent Communication Language.

AI Artificial Intelligence.

ANN Artificial Neural Networks.

CNNs Convolutional Neural Networks.

CPS Cyber-Physical Systems.

DDoS Distributed Denial of Service.

DT Decision Tree.

FIPA Foundation for Intelligent Physical Agents.

IoT Internet of Things.

IT Information Technology.

JADE Java Agent DEvelopment Framework.

MAS Multi-Agent System.

MitM Man-in-the-Middle.

ML Machine Learning.

MLP	Multi-layer Perceptrons.
MQTT	Message Queuing Telemetry Transport.
NCS	Networked Control Systems.
QoS	Quality of Service.
RBF	Radial Basis Function.
RF	Random Forest.
RNNs	Recurrent Neural Networks.
SVM	Support Vector Machine.
XGBoost	Extreme Gradient Boosting.

Chapter 1

Introduction

This work is framed in the context of Industry 4.0, focusing on the concepts of "Cyber-Physical Systems", "Multi-Agent Systems" and "Machine Learning". These concepts are revolutionizing the interaction between the physical world and computational processes and mainly making systems distributed and intelligent, proposing significant improvements in various fields of application, such as manufacturing, energy, health and building automation.

1.1 Motivation and Framework

The 4th industrial revolution, also known as Industry 4.0 is driven by innovative technologies that cause profound changes in production systems and business models. This revolution promotes the connectivity of devices and systems through the use of Internet of Things (IoT) technologies, complemented with the use of other disruptive technologies, such as Big Data, Cloud Computing and Artificial Intelligence (AI). According to [1], connected devices, sensors and actuators will reach over 46 billion units by 2021.

Cyber-Physical Systems (CPS) [2], [3] constitute the backbone to implement the Industry 4.0 principles [4], enabling the next generation of production systems, that are characterized by the integration, in a network and large scale manner, of several heterogeneous, connected and smart processes, systems and devices, combining the cyber and physical counterparts, which allows improving the production systems' performance, responsiveness and reconfigurability, while efficiently managing the complexity and uncertainty [5].

Multi-Agent System (MAS) [6], [7] is a suitable approach to realize such large-scale and complex industrial CPS, due to its inherent characteristics of modularity, autonomy, decentralization and adaptation to emergence without external intervention. In particular, MAS decentralizes the control functions over distributed and intelligent software agents, running in cloud and edge computational layers, that cooperate together to achieve the system goals facing the condition change and reconfiguration needs. To tackle these issues, such agent-based CPS solutions need to exhibit several capabilities, such as reactiveness, robustness and self-organization, supporting the reconfiguration, adaptation and evolution in a very fast, automatic and self-manner [8].

In such decentralized systems, as MAS are, the data privacy and integrity assumes a crucial issue, since security vulnerabilities appears. These systems have characteristics of great data exchange to fulfill the objectives. There is a great concern regarding the breach of private information data, but one of the biggest concerns regarding IoT is related to unauthorized access or control. This high-profile security vulnerability has been hurting connected companies [9].

According to the McKinsey report [10], cyberrisk is the biggest threat to business considered by risk managers and 75% of experts consider cybersecurity to be a top priority. In addition, only 16% of executives at large companies say their companies are prepared to deal with cyberrisks and this is due to the growth of new technologies such as IoT, AI and advanced analytics that bring many benefits, but exposes companies and their customers to new cyberrisks that arrive in new ways. Some companies are investing up to USD 500 million on cybersecurity due to the growth of the threat from cyberattacks. A study by Cisco [11] showed that 53% of the attacks resulted in losses of more than USD 500.000, which includes lost revenue, opportunities, customers and out-of-pocket costs.

Having this in mind, this work addresses the development of security mechanisms to enable the implementation of secure industrial CPS based on MAS, aligned with the

Industry 4.0 principles. For this purpose, an experimental case study is used, in which simple security mechanisms were implemented, such as user authentication, signature and message encryption, in addition to other more complex mechanisms, such as Machine Learning (ML) techniques that are incorporated in the agents in order to assist in the agent's decision regarding the receipt of each message exchanged, in order to classify them as reliable or not, and also to verify sets of messages exchanged between agents, to identify whether they follow normal patterns or not, assuming the function of an intrusion detection system.

1.2 Objectives

The main objective of this work is to study and develop mechanisms to increase security in the communication between the software agents to prevent a modular and self-organized cyber-physical conveyor system based on MAS to possible cyberattacks.

In order to achieve this objective, secondary objectives have been established:

- Analyze and understand all the functioning of the experimental case study that consists of a transport system with cyber-physical components based on MAS.
- Analyze the attack scenarios that agents can be subjected to through threat modeling.
- Define possible mitigation actions.
- Implement common security techniques in the experimental case study.
- Define ML to the application scenario.
- Implement and test different ML algorithms in order to compare the results, such as accuracy, precision, recall and f-measure. Through these results, select which algorithm is a suitable solution for the security in the conveyor system.
- Define a strategy of application of the algorithms selected.

- Integrate the selected algorithms into the system.

1.3 Document Structure

This document is organized in 6 chapters, beginning with the present chapter where the proposal of the work, contextualization and the objectives were presented.

The Chapter 2, entitled "Theoretical Background", presents a review of the fundamental contents for the development and understanding of the work.

The Chapter 3, entitled "Case Study", describes the system architecture used in this work, demonstrating the global functioning of the system with a focus on the main aspects that must be considered in this work, and also presents the threat modeling performed.

The Chapter 4, entitled "Security Mechanisms in MAS", begins with a presentation of the main tools for developing the work and continues with the presentation of the implementation of simple and more complex mechanisms such as ML algorithms in the two different scenarios.

The Chapter 5, entitled "Results and Discussions", presents the results with all the ML algorithms used. The analysis of these results is performed, making it possible to choose the most appropriate algorithm for each application scenario. Lastly, the practical implementation and the graphic interface created are presented.

Finally, the last chapter, entitled "Conclusions and Future Work", rounds up the document with the conclusions and points out some future work.

The document also contains an appendix that comprises the link to access the codes developed in this work.

Chapter 2

Theoretical Background

This chapter aims to guide the reader through a literature review in the field of intelligent and distributed systems, more particularly CPS based on MAS technology. Initially, the concepts of CPS and MAS will be explained. Then, it will be provided a brief explanation of some communication protocols. Afterwards, the problems related to security will be explained. Finally, the concept of ML and also some classification methods will be presented.

2.1 Cyber-Physical Systems and Industry 4.0

The 4th industrial revolution, also known as Industry 4.0 is driven by innovative technologies that cause profound changes in production systems and business models, as shown in Figure 2.1. This revolution promotes the connectivity of devices and systems through the use of IoT technologies, complemented with the use of other disruptive technologies, such as Big Data, Cloud Computing and AI.

CPS are a backbone approach to develop smart system under decentralized structures, based on a network of cyber and physical entities that are integrated to realize a particular objective [2], [3]. The CPS can be seen as a way to transform traditional factories into intelligent factories in the Industry 4.0 context [13]. CPS is more involved with collaborative activities of sensors and actuators in order to achieve certain objectives and for

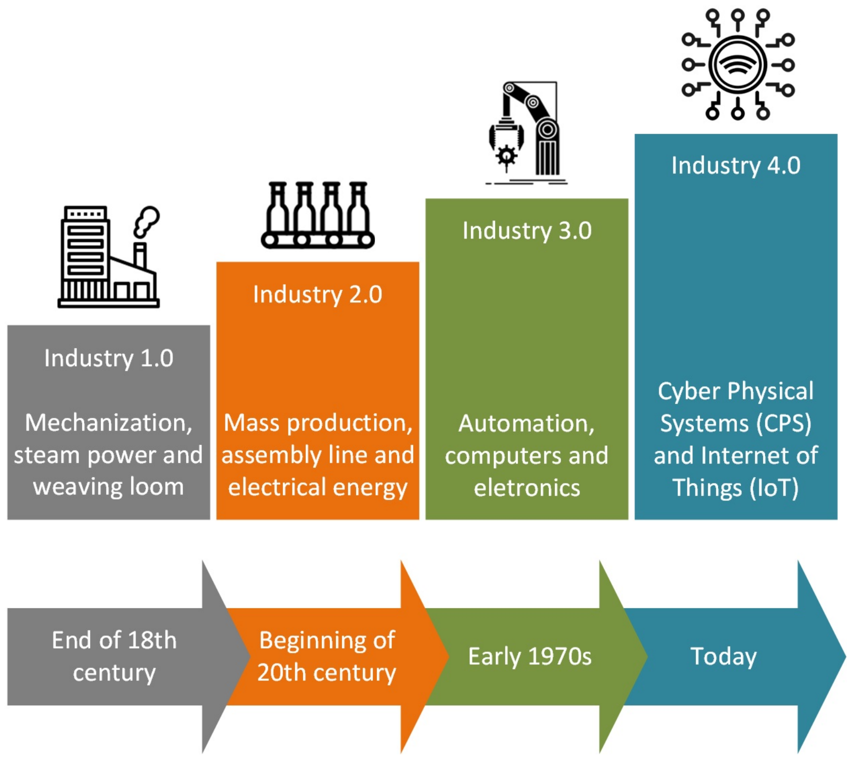

Industry 1.0
Mechanization, steam power and weaving loom
Industry 2.0
Mass production, assembly line and electrical energy
Industry 3.0
Automation, computers and eletronics
Industry 4.0
Cyber Physical Systems (CPS) and Internet of Things (IoT)
End of 18th century
Beginning of 20th century
Early 1970s
Today

Figure 2.1: Evolution of systems in industrial revolutions [12].

that, IoT technologies are used to reach collaborative work on distributed systems [14].

An overview of the CPS can be seen in Figure 2.2, which shows the requirements, applications and what these systems are.

2.2 Multi-Agent System

The intelligent and autonomous agents is one of the research areas that has emerged from AI, systems design and analysis using object-oriented methodology and human interfaces [16]. The research in this area has grown exponentially in recent years.

According to [17], problem solvers are labeled "agents" and work with autonomous, rule-based thinking in response to their internal and external environment. The optimal solutions of agents' problems determine the structures, heterogeneity and hierarchy in which they are organized.

There are many definitions of agents in the literature. These definitions are relatively

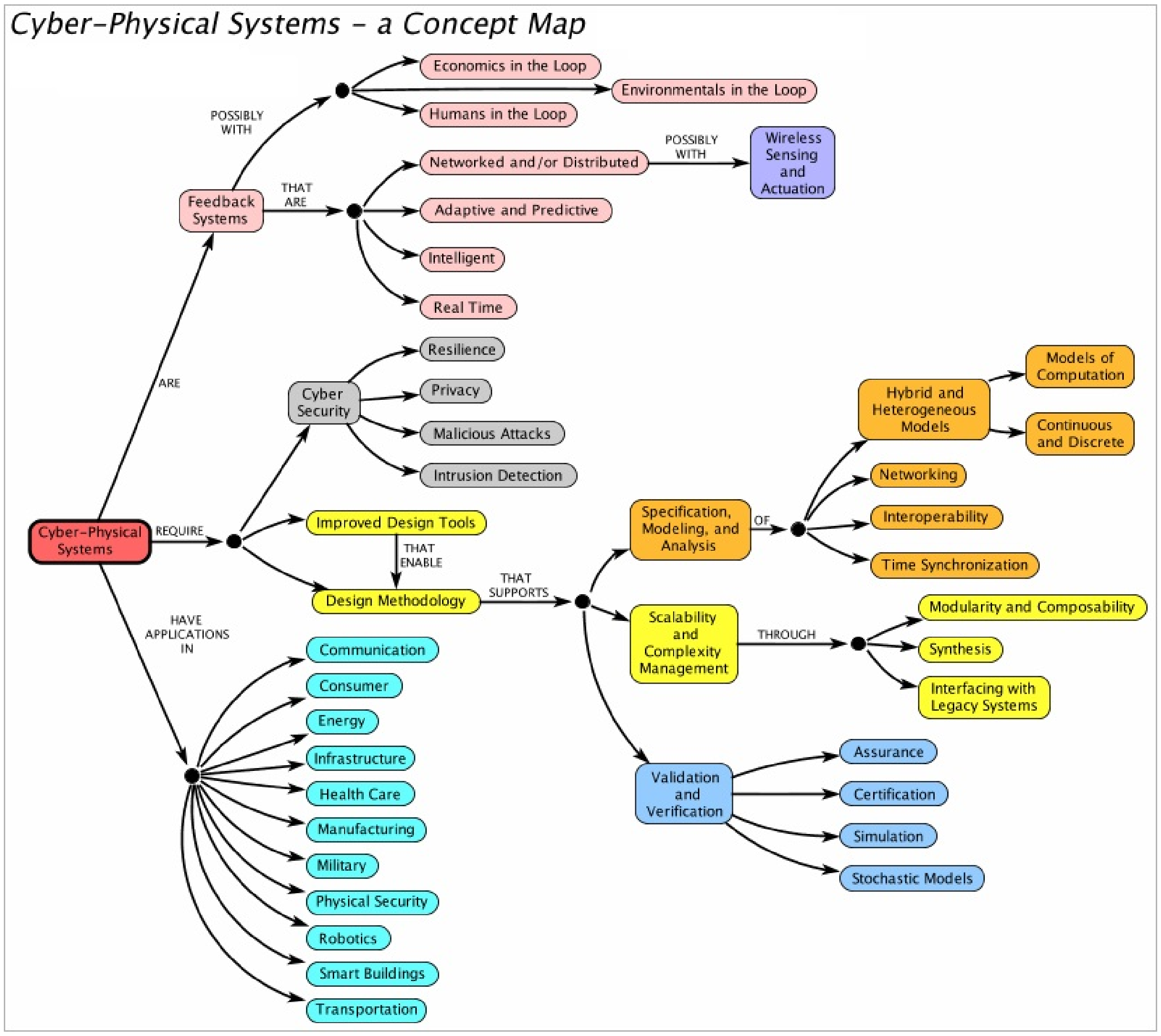


Figure 2.2: CPS concept [15].

similar, but they differ according to the type of application being used with the agent. Thus, an agent can be defined as an entity that is capable of interacting autonomously in a given environment through sensors and actuators to achieve or realize the objectives for which it was designed. In addition, the term agent has some definitions such as: it is a real or virtual (abstract) entity; it is inserted in an environment; can act on the environment; can communicate with other agents; shows autonomous behavior.

There are definitions that highlight some of the main properties and attributes of agents, [18] highlights autonomy, reactivity, proactivity and social ability.

- Autonomy: The agent must be able to perform his tasks, making decisions without interference from third parties (human agents or other computational agents) and have total control over his actions and internal state;

- Reactivity: Agents perceive and react to different changes in the environment in which they operate;

- Proactivity: The agents act in response to changes in their environment and, in addition, exhibit goal-oriented behavior, being able to take initiatives in good times.

- Social skill: The agent must be able to interact with other agents (computational and human) when the situation demands in order to complete their tasks or help the agents to finish theirs.

In addition, [17] points out that a MAS represents and models agents as autonomous reasoning, which are capable of interacting with other agents, being able to adapt to environmental changes and pursue advantages in a rational manner. The agents operate autonomously without the direct intervention of human beings or others in total control of their actions and objectives. Agents send and receive messages about their current situation to agents on another related system or on the same system and exhibit 'evolutionary' behavior in response to changes.

In summary, a MAS is a computer system composed of several intelligent and distributed agents interacting within a given environment, in order to achieve their own individual objective, as depicted in Figure 2.3. To do this, they must be sufficiently endowed with knowledge and skills to be able to achieve their own competencies and their own goals. They must have the capacity to be autonomous in order to be able to make their own decisions, in order to act in relation to the changes that have taken place in the environment in which they operate.

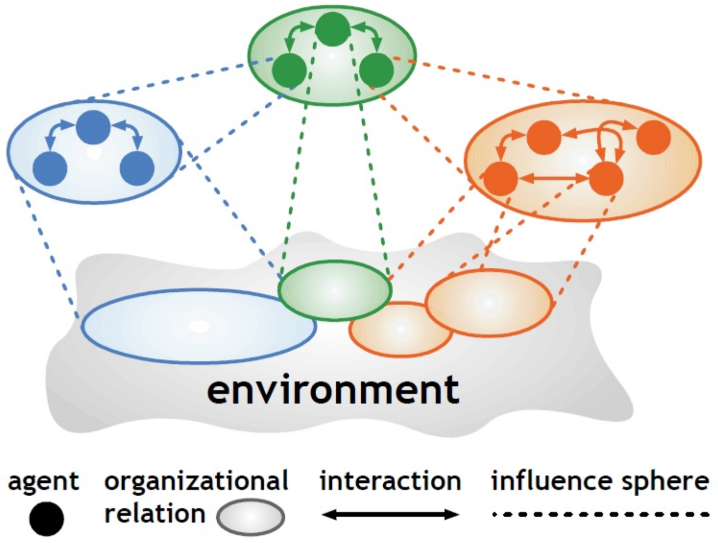


Figure 2.3: Generic scheme of a multi-agent system [19].

2.3 Communication Protocols

This section aims to describe some communication protocols that are usually used in the interconnection of physical and cyber parts in a CPS perspective. For this purpose, it will briefly overview Foundation for Intelligent Physical Agents (FIPA)-ACL protocol, which allows the communication between agents that may be located on different computational platforms, and MQTT protocols, which is commonly used in IoT applications.

2.3.1 FIPA-ACL

FIPA is an international organization that aims to promote the use of intelligent agents by establishing standards that allow the development of solutions based on agents, with the main objective of approaching communication transparently between agents, as it works with ontologies. This ensures interoperability through the standard structure for FIPA-ACL messages [20].

According to [20], [21], the FIPA-ACL message contains a set of parameters, which will vary according to the situation and which will consequently be necessary for an effective communication by the agents. In general, the message structure will be defined through

Table 2.1: ACL message parameters [21].

Parameter	**Description**
Performative	Type of the communicative act of the message
sender	Identity of the sender of the message
receiver	Identity of the intended recipients of the message
reply-to	Which agent to direct subsequent messages to within a conversation thread
content	Content of the message
language	Language in which the content parameter is expressed
encoding	Specific encoding of the message content
ontology	Reference to an ontology to give meaning to symbols in the message content
protocol	Interaction protocol used to structure a conversation
conversation-id	Unique identity of a conversation thread
reply-with	An expression to be used by a responding agent to identify the message
in-reply-to	Reference to an earlier action to which the message is a reply
reply-by	A time/date indicating by when a reply should be received

parameters that indicate the type of communicative act, the content, the participants involved (sender and receiver), the communication control and message description. Table 2.1 shows the parameters of the FIPA-ACL message.

Regarding the security of the FIPA-ACL protocol, this specification allows security management to be implemented in the message transport layer, through the use of security services available in a shared transport protocol on the agent's platform.

2.3.2 MQTT

MQTT is a publish/subscribe message protocol, being considered extremely simple and lightweight, designed for restricted devices and networks with low bandwidth, high latency or unreliable [22]. In addition, it is ideal for implementation in low power devices, such as microcontrollers, due to its small size, low power consumption, minimized data packages and efficient distribution of information to one or several receivers, facilitating the use in remote sensors of the IoT.

In this protocol, there is a broker that is the entity responsible for managing the messages and, for the exchange of messages to be carried out, it is necessary that all clients are connected to the broker. The exchange of messages is carried out by topics, i.e. when a publisher publishes a certain message topic, the broker receives this message and forwards for the subscriber who subscribed the same topic, as shown in Figure 2.4.

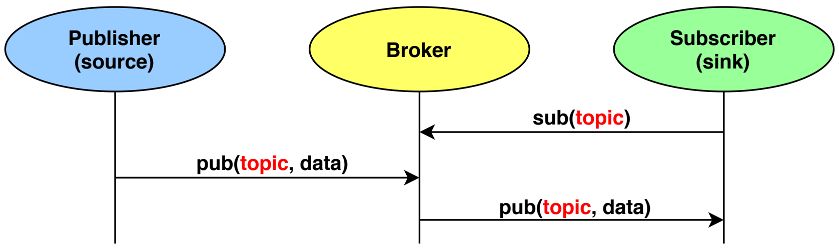


Figure 2.4: Publish-subscribe model using MQTT (adapted from [23]).

Regarding MQTT security support, this protocol by default is not secure. However, according to [22] it is possible to pass a username and password with an MQTT package and encryption on the network must be handled with SSL, but this can add significant overhead on the network since SSL is not a light protocol. In addition, additional security can be added by an application that encrypts the data it sends and receives, but this does not keep the protocol simple and lightweight.

2.4 Security Issues

There were no major concerns about the security of control networks in the past, however with the evolution of internet based communication, attacks on industrial networks have become increasingly common nowadays. A pertinent issue is related to the security of Networked Control Systems (NCS) in relation to Information Technology (IT) security and network security solutions. It is not so simple to deal only with existing solutions, as NCS applications are coupled with physical environments, new security mechanisms are required [24]. The key parameters that differentiate between IT and NCS security are the performance and reliability requirements, the operating systems and applications, the risk management goals, the security architecture, security goals and security assumptions. The main goal of IT security is related to data protection while the main goal of NCS security is related to the protection of the process, i.e. maintain the availability and the well working is the priority.

As shown in the Figure 2.2 provided by [15], CPS implementation requires cybersecurity, consisting of four main issues: resilience, privacy, malicious attacks and intrusion detection.

Resiliency refers to the ability of a system to continue operating satisfactorily when overloaded by unexpected inputs, subsystem failures or environmental conditions or inputs that are outside the specified operating range. Fault tolerance, fault detection and adaptation are all techniques that promote resilience.

Privacy is, in the context of CPS, the problem of protecting information about human beings from unauthorized access by other human beings or machines.

Regarding **malicious attacks**, all networked computing systems are subject. CPS become more vulnerable as networks become more open. However, even in closed networks there is a risk due to incomplete trust in suppliers and the possibility of accidentally introducing malicious code.

Finally, **intrusion detection** refers to physical and cyber intrusions. Technologies that can be used include: embedded vision: motion detection and tracking, human detection, face recognition; and timing models: allow the detection of timing anomalies, which can reveal invasions.

Known threats that can target smart devices include Distributed Denial of Service (DDoS), botnets and malware, Man-in-the-Middle (MitM), data breaches and perimeter attacks that connect IoT devices [25], [26]. DDoS attacks can cause serious problems when shutting down the service provided by the device. The MitM attack may include passive attacks, where the attacker has access to the communication channel (eavesdropping), or even active attacks, such as altering the routing information that may interfere with the Quality of Service (QoS) - (routing attack) or even packet alteration and sending it back to any participant (replay attack) [27]. In addition, a malware can attack the network command and control for IoT devices, and attackers can exploit the device firmware failures to execute arbitrary code by controlling IoT components for their own use. They can still spy on communications in an IoT network and capture the exchanged data over the network. Sensitive data can be exposed or lost and data corruption can also lead to false results.

MAS distributes intelligence through a society of cooperative and autonomous agents, which allows the design of complex and also of large-scale systems that are characterized by modularity, robustness, flexibility and responsiveness, being a suitable technology to perform CPS [28]. Due to the fact that these systems are based on distribution and collaboration, these systems present security constraints. Some of the attacks on industrial agents are described by [29], which are:

- Masquerading: the acquisition of internal information by external entities.
- Eavesdropping: the monitoring internal and external communication, enabling direct access to agent operations as well as their data.
- Cloning or Replacement: the clone of an agent's behavior and replace it by a malicious one, allowing to collect system data and execute malicious commands.
- Agent manipulation: the manipulation of the agent states and data to guide their behavior.

2.5 Threat Modeling

Security analysis in systems architecture plays an important role in addressing the security threats it contains. The purpose of the threat analysis is to identify, prioritize and mitigate potential security threats, eliminating possible vulnerabilities in the system architecture, which consequently reduces waste in the process and the attack vector.

One of the most used methodologies is STRIDE, being a threat analysis model created by Microsoft in 1999. It is one of the most flexible models to carry out threat modeling due to its definition that can be performed at various levels of abstraction. Threat modeling can be implemented at the component level or at the system functionality level, providing a clear understanding of the system's vulnerabilities and possible impacts of the vulnerability of each component across the system.

STRIDE refers to the analysis of security attacks in six different threat types:

- **Spoofing**: is a type of attack in which the attacker assumes the user and performs actions on his behalf, falsifying his own identity. The hacker can gain illegal access to user authentication information and use it to perform various actions on the system. Or rather, it is related to an entity that claims to be something or someone that is not.

- **Tampering**: is a type of attack that can present any form of sabotage, mainly in the intentional modification of the component to make it harmful to the system. Tampering includes unauthorized changes to data exchanged between components or stored in one of them. Regarding the device level, the tampering can be carried out with the total or partial replacement of the device software, which consequently opens the component for the spoofing attack described above.

- **Repudiation**: Repudiation threats are associated with users who deny taking an action without other parties being able to prove otherwise, e.g., a user performs an illegal operation on a system that does not have the ability to track prohibited operations. Non-repudiation is a term commonly used in security, this term refers to the ability of a system to counter repudiation threats. For example, a user who purchases an item might have to sign for the item upon receipt. The supplier can then use the signed receipt as evidence that the user did receive the package.

- **Information Disclosure**: is a term that describes a scenario in which the component can expose information to unauthorized third parties. For example, the ability of users to read a file to which they have not been granted access, or the ability of an attacker to read data in transit between two computers.

- **Denial of service**: is a type of attack that aims mainly to make the service or component temporarily unavailable or to deny service to valid users of the system. Denial of service is usually carried out by flood, i.e. sending an abnormal (huge) amount of requests to the destination service in a short period of time.
- **Elevation of Privilege**: in this type of attack, an unprivileged user gets privileged access and, therefore, has sufficient access to compromise or destroy the entire system by performing unauthorized actions on the system. This attack can be carried out using weak points of system configurations, in which the attacker effectively penetrated all the defenses of the system and became part of the trusted system itself.

Table 2.2 describes the STRIDE threat categories and security properties.

Table 2.2: STRIDE threat categories and security properties.

Threat	**Security properties**
Spoofing	Authentication
Tampering	Integrity
Repudiation	Non-repudiation
Information Disclosure	Confidentiality
Denial of Service (DoS)	Availability
Elevation of Privilege	Authorization

2.6 Machine Learning

The concept of ML is related to the area of AI. According to [30], the field of research in ML focuses on the development of computational techniques that alter aspects of their behavior as experience is acquired, making computational methods capable of making decisions based on knowledge already existing.

The algorithms of ML can be divide in several areas, being defined for the learning-based and also for the type of results that is desired to obtain. According to [31] these areas are:

- Supervised learning: where the algorithm generates a function that maps inputs to desired outputs. One standard formulation of the supervised learning task is the classification problem: the learner is required to learn (to approximate the behavior of) a function which maps a vector into one of several classes by looking at several input-output examples of the function.

- Unsupervised learning: which models a set of inputs (labeled examples are not available).

- Semi-supervised learning: which combines both labeled and unlabeled examples to generate an appropriate function or classifier.

- Reinforcement learning: where the algorithm learns a policy of how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback that guides the learning algorithm.

- Transduction: similar to supervised learning, but does not explicitly construct a function: instead, tries to predict new outputs based on training inputs, training outputs and new inputs.

- Learning to learn: where the algorithm learns its own inductive bias based on previous experience.

The focus of this work is concentrated on supervised learning and the methods that will be used will be briefly described below.

2.6.1 Decision Trees

Decision Tree (DT) belong to supervised learning methods. These algorithms are statistical models that represent a set of rules that follow a hierarchy of classes and values, with the purpose of producing a classification or regression function in a given data set.

The hierarchical set that is produced has a tree structure, similar to a flowchart. Each internal node represents an attribute to be tested and the result of the test is the possible ramifications for new nodes. Thus, the node at the top is called the root, being the first to be tested and will start the flowchart. The leaf nodes, which are at the bottom, are the terminal nodes, representing one of the classes known as the objective of the problem [32]. Figure 2.5 shows an example of a DT, in order to predict whether or not a person will play tennis.

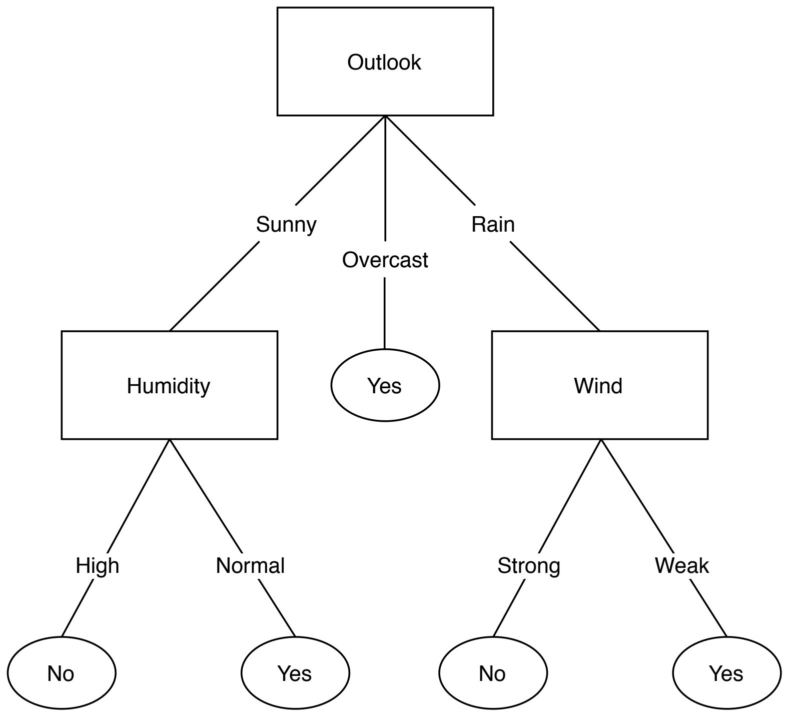


Figure 2.5: Decision trees example (adapted from [30]).

As depicted in Figure 2.5, the internal nodes are marked with rectangles and the terminal nodes are marked with an ellipse, being the attributes and classes respectively. In this tree there are three attributes that influence the classification, being Outlook, Humidity and Wind and also two possible classes as an answer, yes and no.

The division of attributes is determined based on a heuristic, which aims to select the best partition criterion. The best criterion for partitioning the data is that which selects an attribute that after the division, obtains as a result the data of a single class in each branch.

There are several heuristics, which have different characteristics that are capable of identifying the data pattern. The best known heuristics are: information gain, gain ratio and Gini index. In summary, the information gain uses entropy as a measure of impurity, selecting as a test attribute the one that maximizes the information gain. The gain ratio is nothing more than the relative (weighted) gain of information as an evaluation criterion. Gini, on the other hand, uses a statistical dispersion index [33].

There are several algorithms for the implementation of DT. The first DT algorithms were developed in the early 1980s, which consequently became popular in the area of ML and acquired new versions and metrics that improved their performance. The most used DT algorithms today are ID3 [34], C4.5 [35] and CART [36].

2.6.2 Gradient Boosting

Gradient boosting is a technique introduced by [37] and developed recently that proved to be very powerful, corresponding today to one of the most used in ML competitions. This technique deals robustly with a variety of data types, relationships and distributions and a large number of hyperparameters can be improved and adjusted for a more appropriate scenario.

The principle behind any boosting algorithm is the combination of the result of many weak classifiers (or regressors), combining to form a kind of strong decision committee. Although there are no restrictions on these classifiers and regressors.

In this type of model, weak learning is usually trees and each of these trees maps an input data point to one of the leaves that contains a continuous score. The model minimizes a regularized objective function that combines a convex loss function (based on the difference between the predicted and the target outputs) with a penalty term for the model's complexity (i.e., the tree functions) [38]. The training proceeds iteratively, adding new trees that predict waste or errors from previous trees, with which they are combined to make the final prediction. In addition, it is called gradient boosting because it uses a gradient descending algorithm to minimize loss when new models are added.

2.6.3 Random Forest

The Random Forest (RF) algorithm is based on the decision tree concept described in the previous section, however it differs in the aspect of the number of DT that instead of obtaining just one, produces multiple trees. This set of multiple trees is called a "forest" and aims to improve accuracy. For this, the algorithm combines the result of the trees through voting, returning a majority voting model [39], as depicted in Figure 2.6.

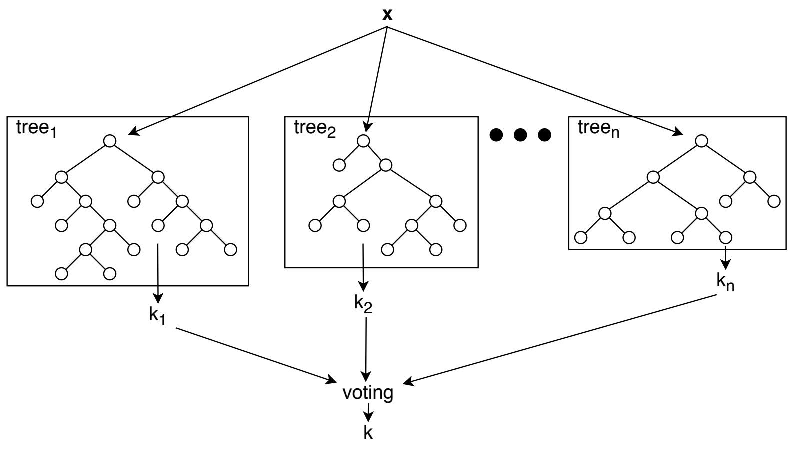


Figure 2.6: A general architecture of a random forest (adapted from [39]).

It is important to note that as the number of trees in increases, the test set error rates converge to a limit, meaning that there is no over-fitting in large RF [40].

2.6.4 Artificial Neural Networks

According to [41], Artificial Neural Networks (ANN) can be defined as *“structures comprised of densely interconnected adaptive simple processing elements (called artificial neurons or nodes) that are capable of performing massively parallel computations for data processing and knowledge representation”*.

Although ANN is an abstraction of biological neuron networks, the idea was not to replicate the operation of biological systems, but to make use of what is known about the functionality of biological networks to solve complex real-world problems. The characteristics that stand out most in relation to the information processing of the biological system are nonlinearity, high parallelism, robustness, fault and failure tolerance, learning, ability to handle with imprecise and fuzzy information and their ability to generalize [41].

The analogy between artificial neuron and biological neuron is that the connections between the nodes represent the axons and the dendrites, the weights of the connections represent the synapses and the threshold approximates the activity in the soma [41], as shown in Figure 2.7, where n biological neurons with various signals of intensity x and synaptic strength w, feeding a neuron with a threshold of b, and the equivalent system of artificial neurons.

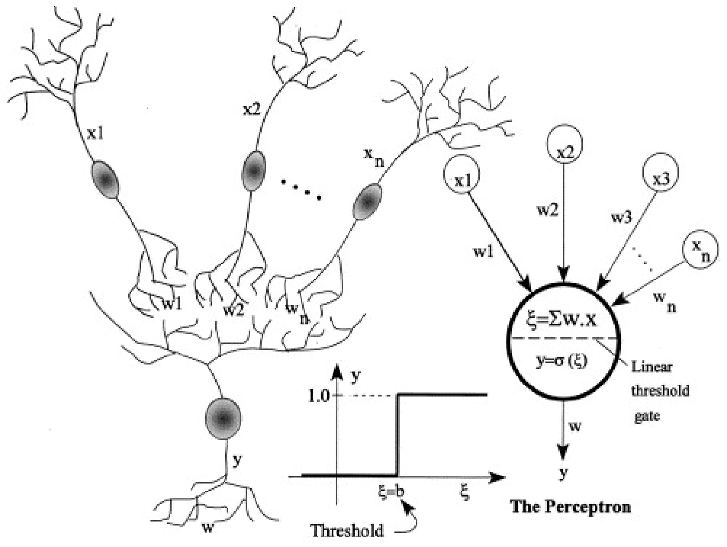


Figure 2.7: Analogy between artificial neuron and biological neuron [41].

The way of learning for both neural networks (biological and artificial) is similar. Neural networks learn by incrementally adjusting the magnitudes of the weights or strengths of the synapses [42].

There are thousands of specific ANN proposed by researchers, where the vast majority present modifications in existing models. The three classes of ANN that are best known and most used are: MLP, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). However, the main focus will be on MLP, which comprises one or more layers of neurons.

In MLP, data is fed into the input layer, there may be one or more hidden layers that provide different levels of abstraction, and predictions are made in the output layer, which is also called the visible layer. The model of a simple neural network is illustrated in Figure 2.8.

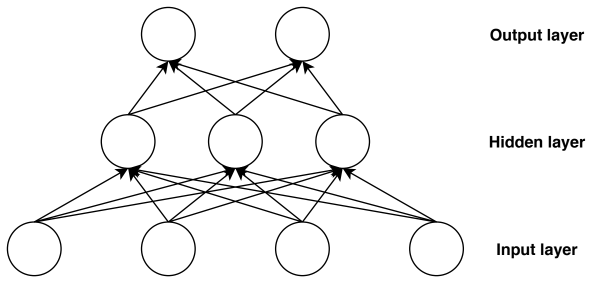


Figure 2.8: Architecture of an MLP neural network with a hidden layer (adapted from [41]).

2.6.5 Support Vector Machine

The Support Vector Machine (SVM) was introduced by [43], which consists of a supervised machine learning classifier based on statistical learning theory. The operation of the SVM consists in maximizing the margin between the instances of two classes by creating a hyperplane.

A simple example is depicted in Figure 2.9, which contains two-dimensional instances. The SVM classifier will generate a line (hyperplane) that divides the plane into two parts, with each class on each side. The SVM classifier uses four fundamental concepts which are: kernel, regularization, gamma and margin.

Kernels are used to map input instances, usually of different dimensions, to the decision boundary that can be built. Some examples of kernels are Linear, Polynomial and Radial Basis Function (RBF). Regularization informs SVM optimization when you want to mistakenly avoid each training instance. For small regularization values, optimization will be summed up in a choice of a separation hyperplane with a larger margin, even if that hyperplane incorrectly classifies more instances. On the other hand, for large values of regularization, a hyperplane with a smaller margin will be chosen. Finally, the gamma

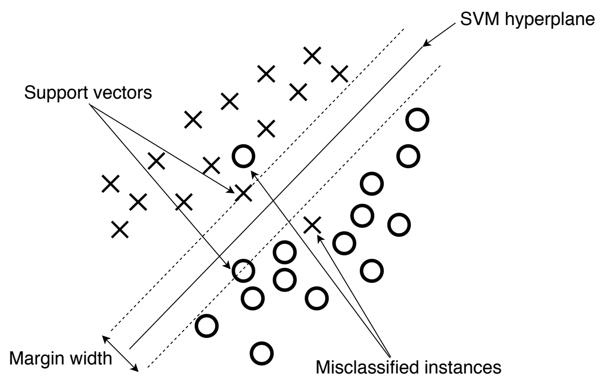


Figure 2.9: Linear support vector machine example. (adapted from [44]).

value defines the influence of a single training instance. Therefore, for low gamma values, points distant from the separation line are considered in the calculation of the separation line, and for high gamma values, only those instances close to the separation line that will be considered in the calculation of the separation line.

2.6.6 Model Evaluation

The evaluation of a supervised ML classifier is generally carried out by analyzing the performance of the model generated during the labeling of new objects, i.e. the evaluation is made based on performance measures that indicate the success rate of the predictive model [45], [46].

In certain cases, there is only one dataset, where it should be used to induce the predictor and also in the evaluation. In these cases, alternative sampling methods should be used to obtain more reliable predictive performance estimates, defining training and test subsets [47].

There are several methods in the literature, such as the holdout, k-fold and leave-one-out [48], [49]. In this work, the holdout method will be used, which consists of dividing the total set into two subsets, one set for training and one for testing. A proportion widely used in the literature is 70% of the data for the training set and 30% for the test set [48]. After partitioning, the model estimation is performed and, subsequently, the test data are applied and the prediction error is calculated.

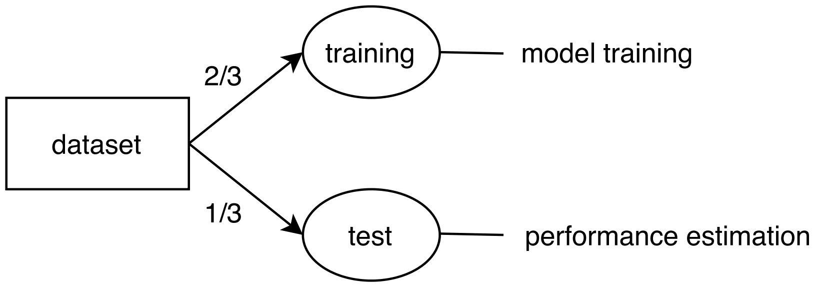


Figure 2.10: Holdout method (adapted from [47]).

According to [45], there are several ways to evaluate the performance of supervised ML algorithms and the classifiers they produce. The confusion matrix records examples recognized correctly and incorrectly for each class and from that matrix the quality measures are constructed. Table 2.3 presents a confusion matrix for binary classification.

Table 2.3: Confusion matrix example

Class / Recognized	as Positive	as Negative
Positive	TP	FN
Negative	FP	TN

Where TP are true positive, FP - false positive, TN - true negative and FN - false negative counts.

From these values, some commonly accepted performance evaluation measures can be calculated, such as accuracy, precision, recall and f-measure.

$$Accuracy = \frac{TP + TN}{P + N} \tag{2.1}$$

$$Precision = \frac{TP}{P} \tag{2.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

$$F - measure = \frac{(\beta^2 + 1) \times Precision + Recall}{\beta^2 \times Precision + Recall} \tag{2.4}$$

The most used empirical measure, accuracy, does not distinguish between the number of correct labels of different classes. On the other hand, the measures of precision, recall and f-measure are measures that separately estimate the performance of a classifier in different classes [45]. These measures have a greater focus on scenarios where the number of examples belonging to a class is generally substantially less than the overall number of examples. Recall is a function of its correctly classified examples (true positives) and its misclassified examples (false negatives). Precision is a function of true positives and examples misclassified as positive (false positives). The f-measure is evenly balanced when $\beta = 1$.

2.7 Summary

Bibliography analysis provided a good understanding of intelligent and distributed systems, particularly CPS based on MAS. These systems have mainly revolutionized production systems, through modularity, robustness, flexibility and responsiveness. Systems based on the technology of MAS is suitable for the implementation of CPS, in which the distributed systems present several agents that exchange messages between them for decision making, making it possible to achieve the objectives of the system that is based on collaborative actions.

The main vulnerability arises in the exchange of messages between agents, as they become targets for attacks. The main attacks on agents are related to the acquisition of internal information by external entities, the monitoring of communication between agents, the clone of agents, the insertion of malicious code in the agent and the manipulation of agents. These attacks can directly affect the functioning of the system, making the system unavailable or failing to achieve its objectives.

When it comes to data exchange, ML techniques are great for finding patterns in the data and therefore can be used as an aid in data classification. With this, these techniques can bring security to the systems once it is possible to extract the main characteristics and patterns of the data exchanged.

Chapter 3

Case Study

This chapter aims to present the general description of the system under study, focusing on the main and most relevant aspects for this work and also to present the security analysis, which presents a threat modeling that brings the main threats that will be mitigated with the mechanisms developed and applied.

3.1 System Description

The case study considered in this work consists of a conveyor transfer system that contains a modular sequence of Fischertechnik conveyors, where each one has the functionality of transporting parts from an initial position to a final position. Each conveyor contains a belt operated by a 24V DC motor and two light sensors responsible for detecting parts in the initial and final positions, being also operated in 24V, as depicted in Figure 3.1.

The way that parts are transferred along the conveyor modules is depicted in the Figure 3.2, starting at the input position of the first module S_{input1} and ending at the output position of the last module of the sequence $S_{outputn}$. The transfer between two consecutive conveyors is performed as follows: conveyor C_2 starts its motor when the part arrives at the sensor $S_{output1}$ and conveyor C_1 only stops its motor when the part arrives at the sensor S_{input2}.

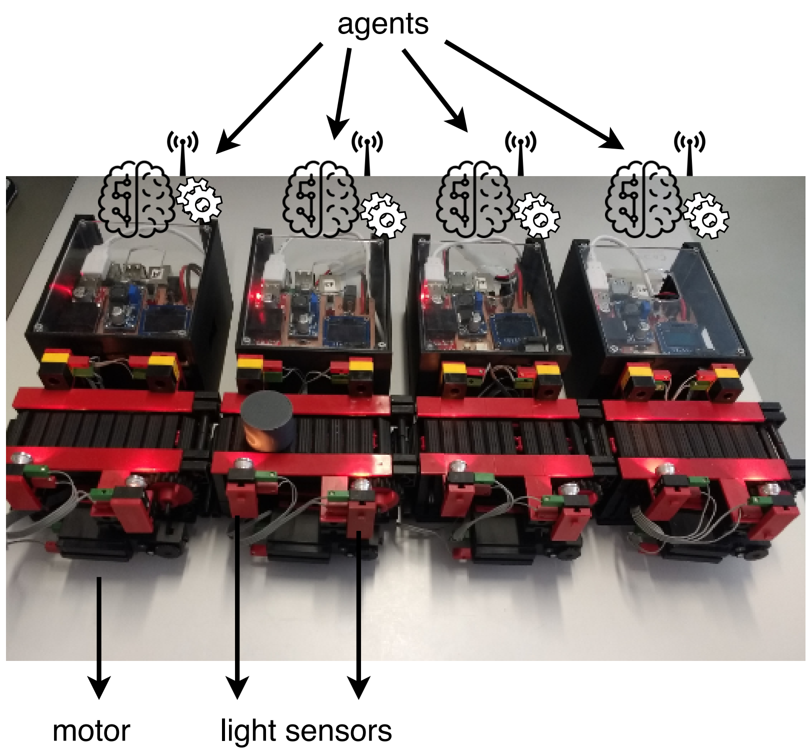


Figure 3.1: The modular conveyor transfer system.

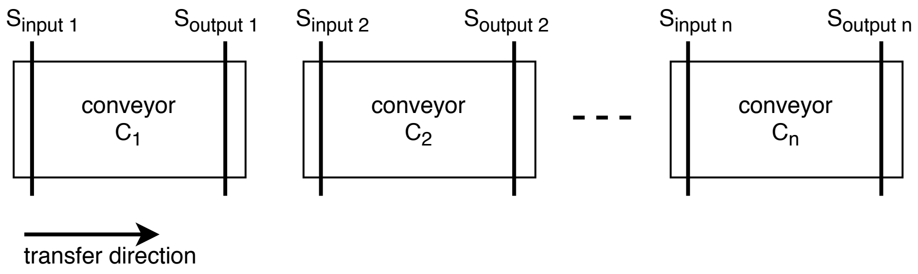


Figure 3.2: Modular and self-organized conveyor transfer system.

As described in [5], this system does not use traditional approaches for the implementation of control logic, such as the use of IEC 61131-3 or even the use of IEC 61499 [50] due to the fact that IEC 61131- 3 has limitations in terms of scalability and system reconfigurability and IEC 61499 does not completely support the intelligent and autonomous decisions, what is needed to achieve system requirements in terms of scalability and reconfigurability on-the-fly, which means that it is not necessary to stop, restart or reprogram the system in cases of adding or removing a conveyor, or even changing the sequence between the conveyors.

The approach used in this conveyor system that supports the easy and on-the-fly reconfiguration was performed through the use of MAS technology to create CPS based on several modules of cyber-physical devices, where a software agent plays the role of the cyber part controlling the physical conveyor device. For this, agents are enhanced with a self-organization mechanism.

Therefore, the cyber-physical conveyor system comprises several modular cyber physical conveyor devices, each one performing its own operation and interacting with the others to ensure the emergence of the transfer functionality of the entire conveyor system. The cyber part of each module consists of a raspberry pi, where each agent is running in this single board computer.

The communication between the different modules is achieved due to the use of MAS technology and the use of the raspberry pi's WiFi capability. Whenever there is a new event in the system, the agents communicate with each other using the FIPA-ACL standard to take the necessary measures to carry out their objectives. There is a great exchange of data between the agents, because in addition to carrying out the main individual objective, which is the transport of parts from the beginning to the end of each module, there is the objective of the system as a whole, which consists of the transport of parts from the beginning to the end of the sequence and for that, the agents must always communicate to be updated of the current sequence of the system. Also, if there is an addition or removal of a module, or even the change of position between modules, they must recognize as soon as possible to transfer the part correctly.

For this purpose, several interaction patterns were designed to exchange information with the adjacent agents regarding the position of the parts. The agents exchange several types of messages between them during their cooperation processes, namely the *informIAmAlive*, *tokenTransmissionInput*, *tokenTransmissionOutput*, *thereIsASwap*, *swapTokenFound*, *goodbye*, *pieceAtLastConveyor*, *conveyorCountWarning* and *productIDTransmission*. These communication protocols are used to maintain the operation of the system and the token is responsible for designating the position of each module. For a better analysis of all interactions, all messages exchanged are sent to a database.

The system control depends on the messages exchanged by the agents, which are exchanged over the network. Figure 3.3 depict the control architecture of the system.

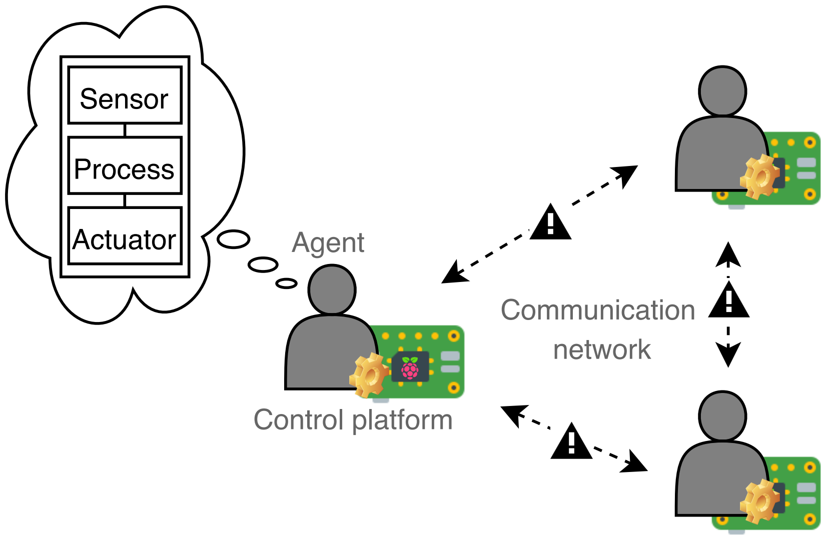


Figure 3.3: Control system architecture.

3.2 Security Analysis

Following the analysis methodology based on STRIDE previously described, the main risks found in the system architecture and which mechanisms can be used to mitigate will be presented. However, before presenting the analysis, a very important observation must be mentioned. All events that occur in the system, i.e. detection of parts in the input and output light sensors, are perceptions of the environment that will be captured by the agents making it capable of making decisions to generate actions on the system, i.e. turning on or off the motor. As it is a distributed and collaborative system, in addition to having perceptions about the environment, the agent must also communicate with other agents in order to pass his perception on to others. Therefore, the functioning of the system described in the previous section depends mainly on the exchange of messages between agents to determine their position in the transportation system sequence and to transfer parts.

In this way, the analysis will be focused exclusively on agents and the exchange of data between them to achieve the objectives of the system, since all components of the system (sensors and actuators) are controlled by the agent.

Regarding the **spoofing** threats, it is possible to find the following possibilities:

- **Attack**: Impersonate an agent.
- **Risk**: By creating a fake agent (external to the system) similar to the system agents, the attacker may be able to inject false information into the system, sending a message to other agents to perform actions on the other original agents in the system.
- **Mitigation**: Mitigation for this attack can be performed by adding user authentication on the JADE platform, so that only the user has access to the platform and only he can deploy new agents.

Regarding the **tampering** threats, it is possible to find the following possibilities:

- **Attack**: Modification of data sent by the agent
- **Risk**: By modifying the data sent from one agent to the other (in combination with the previous attack in which a false identity is created), the attacker can produce false data that is capable of totally affecting the functioning of the system.
- **Mitigation**: Mitigation for this attack is the verification of all messages received by the agent to classify them as reliable (accepting messages that will not affect the operation system) or unreliable (ignoring the messages to prevent the system from being affected or does not work properly).

Regarding the **repudiation** threats, it is possible to find the following possibilities:

- **Attack**: Sending messages with the fake signature to the agents.
- **Risk**: This attack can cause agent misbehavior. The agent will not be able to trace the origin of the package that resulted in its misbehavior.
- **Mitigation**: As a mitigation, it is necessary to perform validation of the sender by checking the package signature and if it's not valid the package is dropped.

Regarding the **information disclosure** threats, it is possible to find the following possibilities:

- **Attack**: Sniffing the communications
- **Risk**: By performing the MitM attack on the communication network between agents, the attacker will have access to all data exchanged by the agents.
- **Mitigation**: As a mitigation, it is necessary to provide secure communication between agents to protect the data sent. This can be done by encrypting the messages before sending them.

Regarding the **DoS** threats, it is possible to find the following possibilities:

- **Attack**: Flooding.
- **Risk**: The attacker can organize the flooding of the network which will result in denial of service (mainly in relation to the platform's IP address), because the services will not be able to accept any request or may exploit vulnerabilities that could interrupt the normal operation of the devices.
- **Mitigation**: To mitigate this attack the firewall should be configured to drop the traffic or limit the size of incoming ping requests.

Regarding the **elevation of privileges** threats, it is possible to find the following possibilities:

- **Attack**: Abuse the agent's functionalities and functions by exploiting vulnerabilities in the JADE platform.
- **Risk**: Once an agent is placed on the platform, by targeting platform functionality, an attacker can perform unauthorized actions, such as killing other agents on the platform.
- **Mitigation**: As a mitigation, it is necessary to establish permissions for each agent inserted in the platform.

After analyzing the main risks present in the system, some solutions were proposed to mitigate these threats, such as the use of the JADE-S add-on for the JADE platform and ML models that will be described in the next chapter. However, some categories of STRIDE are not mitigated with the present solutions but will be addressed in future works.

Chapter 4

Security Mechanisms in MAS

This chapter presents the implementation of security mechanisms in the system described in the previous chapter.

4.1 Incorporating features present in the JADE-S plug-in

To protect the data exchanged between agents, features presented in the JADE-S add-on [51] from the Java Agent DEvelopment Framework (JADE) platform were used to increase the security of the system, that provides support for security in MAS, such as end-to-end message signature and encryption, user authentication and agent actions authorization against agent permissions.

Authentication provides a guarantee that a user who starts a JADE platform and thus generates containers and agents within that platform, is considered legitimate within the secure scope of the computer system that hosts the main container on that platform. In this case, the JADE authentication process implies that the user is known by the system to have at least one valid identity and associated password.

For the case of user authentication, the special login module available was used, which allows a basic authentication using the plain text password file held in a system folder that aims to be used for initial tests, but standard modules are available like the Unix, NT and Kerberos, which are easy to apply.

As basic authentication for testing was implemented, a text file was created with the username and password. For user validation, whenever you start a new container and/or agent, it is necessary to pass the parameter on the command line (–owner user:pass) so that it is possible to perform the user verification. If you fail to pass these arguments, the action will not be authorized.

Regarding the signature and encryption of messages, these mechanisms are applied to the entire payload to protect all information such as content, protocol, ontology, etc. It is important to note that when using these features, users do not have to deal with the signing and encryption mechanisms (keys, algorithms, etc.), and it is only necessary to require that the message be signed and encrypted, as shown in a part of the code used for signing and encrypting a message to be sent to another agent.

mySecurityHelper.setUseSignature(tokenTransmission);
mySecurityHelper.setUseEncryption(tokenTransmission);

This message signing and encryption setting was performed for all message protocols that agents send. When receiving messages, there is verification of the signature and encryption so that the message can be read.

To enable these authentication, signing and encryption services, the following services must be enabled: "jade.core.security.SecurityService", "jade.core.security.signature.SignatureService" and "jade.core.security.encryption.EncryptionService". This is done on the command line when creating the platform and inserting new agents. More implementation details can be found in the tutorial for implementing these features described in [51].

4.2 Incorporating ML techniques into the system

This section aims to present the main stages of the implementation of ML techniques, starting with a brief overview of ML, data collection, pre-processing of data, definition of application scenarios and finally the choice of parameters for different models.

4.2.1 ML overview

The implementation of ML techniques comprises several stages, as can be seen in Figure 4.1 in a summarized and simplified way.

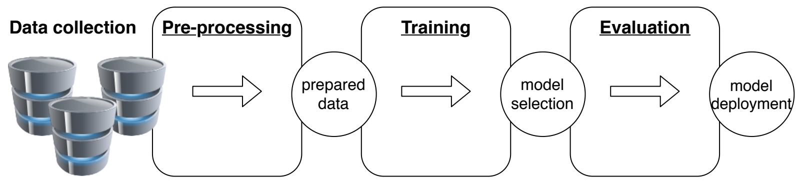


Figure 4.1: Steps for implementing ML techniques.

The process begins with the data collection, in which it is necessary to analyze which data are relevant and which have important characteristics for the acquisition of knowledge. After data collection, it is extremely important to perform the pre-processing of the data, because as these are computational models, all data must be duly in formats accepted for processing, i.e., numerical data. After obtaining the data already prepared, the next step consists of training, i.e. obtaining the characteristics of the data for the creation of a model. For this, many models can be selected for this purpose. The model evaluation stage consists of analyzing the results in relation to the model's learning, being important to analyze the success rate of this model. Finally, the last step is to implement the model, after having evaluated which model has obtained the best result and will be applied in the system.

Before implementing ML techniques, it is important to know in which scenario these techniques will be used. As it is a distributed system, two scenarios were defined for the application of these techniques.

One of the scenarios aims to implement ML for learning in the data of the system as a whole, analyzing all the data exchanged for the transport of an object from the beginning to the end of the sequence. This technique will be implemented in all agents, but only the agent that is in the last position that will use this technique, because that is when all data has been exchanged and the transport of the part has been completed. However, it must be implemented in all agents to maintain dynamic functioning, allowing the change in the sequence of the modules. This scenario can be better understood with the diagram illustration in Figure 4.2.

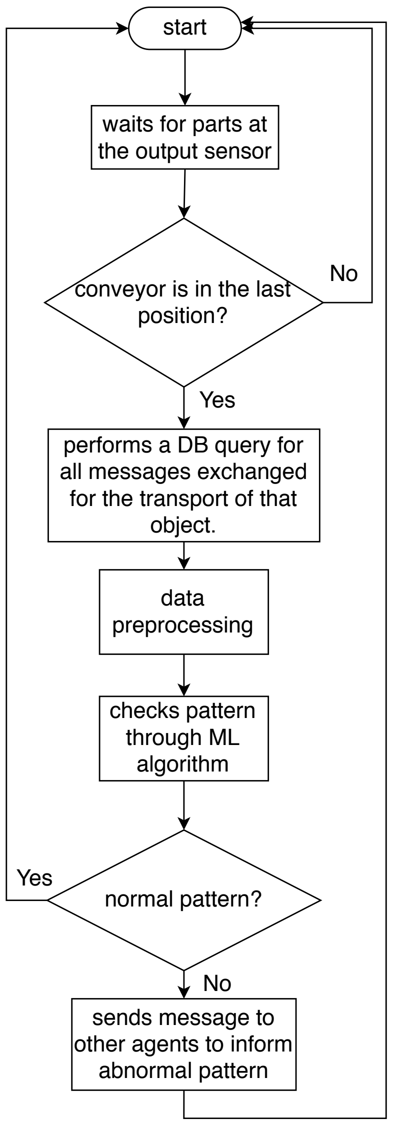

start
waits for parts at the output sensor
conveyor is in the last position?
No
Yes
performs a DB query for all messages exchanged for the transport of that object.
data preprocessing
checks pattern through ML algorithm
normal pattern?
Yes
No
sends message to other agents to inform abnormal pattern

Figure 4.2: Scenario diagram for intrusion detection.

As shown in the diagram, first the agent waits for parts in the output sensors of the modules to check if it is positioned in the last position of the transport sequence. If not, it will be waiting again for new parts in the output sensor, making sure that there is always a check in its position since there is a possibility of changing the position of the modules. In case the module is in the last position, the agent will search the database to obtain the data exchanged for the transport of the last piece and will convert the non-numeric data into numeric data. After this step, the operation pattern will be checked and if it is not normal, the agent will inform the other agents that there was an intrusion on the system.

The other scenario is to apply a ML technique to all agents, enabling them to use this technique to help them make decisions whenever they receive new messages, making decisions more intelligent. Thus, this scenario consists of applying the ML technique in a distributed way, since all agents have the implementation in their behavior to use these methods whenever they receive new messages. Figure 4.3 illustrates the diagram for this scenario. The preprocessing step described in the diagram refers to the conversion of non-numeric data into numeric data, which is performed automatically by the agents.

In short, for the two defined scenarios, ML techniques will be incorporated into all agents in the system. However, simultaneous use in all agents is performed in the scenario in which ML is used to assist decision-making by agents with the receipt of new messages while in the scenario of verifying the entire functioning of the system, the ML technique will only be used by the agent who is in the last position of the module sequence.

4.2.2 Data collection

All messages exchanged between the different agents inserted in the JADE platform can be viewed through a sniffer agent available on the platform, providing in real time the main characteristics of the message exchanged, such as the sender, the receiver, the content and the protocol. However, it is only possible to save the file manually in text format, making it impossible to collect data in real time. In this way, a function was implemented in the agent's behavior that makes it possible to insert all the characteristics of each message received by the agent in the database, in addition to additional characteristics, such as the message processing time and also to which object belongs the message exchanged. For this, the PostgreSQL database was used and all iterations between agents are saved in this program, where it is possible to obtain a file in CSV format. In addition, the use of this program also allows the agent to search the saved data.

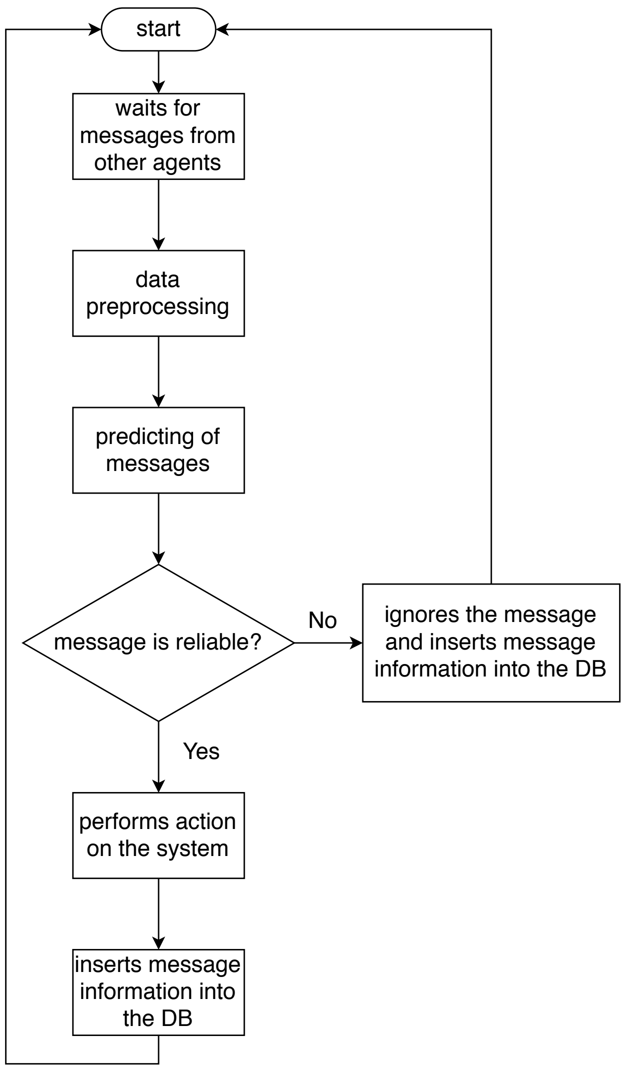

start
waits for messages from other agents
data preprocessing
predicting of messages
message is reliable?
No
ignores the message and inserts message information into the DB
Yes
performs action on the system
inserts message information into the DB

Figure 4.3: Scenario diagram for making decisions when new messages arrive.

Data related to the sender, receiver, content, protocol, product id, token, time, number of conveyors and number of messages exchanged previously were collected, as described in Table 4.1.

As supervised learning techniques will be used, the classes of the messages collected must be known. With normal operation, it was possible to collect normal data to the system. For the collection of data with the unwanted functioning, it was necessary to create some attacks.

Table 4.1: Sample of data collected

sender	receiver	content	protocol	product id	token	time (ms)	number of conveyors	number of messages exchanged previously
Agent1	Agent3	3	token Trans. Input	11716	1	1	3	7
Agent1	Agent2	3	token Trans. Input	11716	2	1	3	7
Agent1	Agent3	0	piece AtLast Conveyor	11716	1	2	3	8
Agent3	Agent1	14012	product Trans.	14012	3	10	3	1

The attacks created have a great focus on interfering with the functioning of the system, aiming to make it unavailable or to hinder its process or objective.

The attack consists of the creation of an agent external to the system that sends malicious messages to the agents of the system, i.e. messages with contents exchanged in order to interfere with the functioning of the system. One of the attacks consists of listening to all system messages and whenever it receives a message with *tokenTransmissionOutput* protocol, it replicates that message with the exchanged content, causing agents to take actions on the environment (e.g. turning on the modules' motors) in moments unwanted, causing several modules to have the motors running unnecessarily as there is no part to be transported at that time. Another attack also implemented is based on the previous attack, but the content of the message with *tokenTransmissionInput* protocol is changed, causing the module that is waiting to receive a part to turn off the motor, blocking the transport of parts throughout the system.

4.2.3 Data pre-processing

The objective of data preprocessing is to extract each data set, mainly in relation to which attributes can influence the classification model. For this, these data were analyzed and the following three datasets were created:

- 1: Interactions between agents considering main characteristics present in messages.
- 2: Interactions between agents considering main characteristics present in the messages and also additional characteristics such as the processing time and quantity of messages previously exchanged.
- 3: Set of messages exchanged to transport each piece separately.

The difference between the first and second data sets is in the amount of information they contain. The second contains some additional information that was created to try to improve the model, since the first data set does not contain as much information. For this, some variables and logics were created in the agent's behavior, allowing the verification of the number of messages that had already exchanged before receiving a new message and also the processing time of each message was analyzed in the agent's behavior.

Tables 4.2, 4.3 and 4.4 show the samples of the three datasets respectively.

Table 4.2: Sample of data of the first dataset

receiver	content	protocol	token	number of conveyors
Agent4	1	tokenTransmissionInput	2	4
Agent1	1	tokenTransmissionInput	3	4
Agent3	1	tokenTransmissionInput	4	4
Agent1	1	tokenTransmissionOutput	3	4
Agent1	2	tokenTransmissionOutput	3	4

Note that the sender information is not used in any of the datasets. This is due to the fact that it will be considered that for a hacker to send a message to the agents belonging

Table 4.3: Sample of data of the second dataset

receiver	content	protocol	token	time	number of conveyors	number of messages
Agent4	1	tokenTransmissionInput	2	5	4	2
Agent1	1	tokenTransmissionInput	3	3	4	2
Agent3	1	tokenTransmissionInput	4	2	4	2
Agent1	1	tokenTransmissionOutput	3	1	4	3
Agent1	2	tokenTransmissionOutput	3	2	4	4

Table 4.4: Sample of data of the third dataset

number of messages	average time	processing time	number of conveyors	protocols used
6	35	210	2	3
9	7	65	2	4
10	104	1041	2	3
18	1	23	3	4
25	11	286	3	6
29	3	99	3	4
42	19	831	4	4
52	13	717	4	6
62	10	622	4	4

to the system, the hacker must have the identification of any of the agents in the system, i.e., have the cloned or falsified identification. Therefore, verification of the source of the received messages was implemented in the agent's behavior and the agents only accept messages from known agents (or known agents' IP).

Regarding the amount of data collected, the first and second data sets have 22365 message information exchanged by agents each one and the third has 562 information from the data sets for object transfer.

In addition to extracting the main attributes that can influence the classification model, this data must be prepared for application in the classification models, i.e. it must contain only numerical data. For this, all non-numeric data present in the tables were converted to numerical data in order to make the data ready for application.

Each dataset was divided into training and testing for the experiments of the ML classifiers. To avoid possible overfitting, both training and test data have the same percentage for class 0 and class 1 data. In addition, the holdout method described in the section 2.5.6 was used to create the data subsets training and testing, 70% and 30% of all data respectively. Table 4.5 shows how the training and test data were organized.

Table 4.5: Separation of training and test data

Dataset	Train	Test
	number of instances	number of instances
1	15655	6710
2	15655	6710
3	360	162

The first and second datasets have 18980 instances with a label defined for class 0 and 3385 for class 1, corresponding to the proportion of 84.8%/15.2%. The third dataset has 426 instances with a label defined for class 0 and 96 for class 1, corresponding to a proportion of 81.6%/18.4%.

4.2.4 Application scenario

The three datasets were created for application in two different scenarios. The first and second datasets will be used for the first scenario, which consists of analyzing each message received by the agent and predicting whether the message is reliable or not according to the knowledge acquired in the training. The results obtained with the first two datasets will be compared between them to verify which dataset gets the best result, since one of the datasets contains additional information. Figure 4.4 illustrates the agent structure for this scenario.

The third dataset is used in the second scenario, which consists of analyzing the functioning pattern of the system as a whole for the transport of each part, i.e. from the beginning to the end of the transport of a single part, all data relating to the interaction between agents will be analyzed and verified by the ML technique to classify it as a

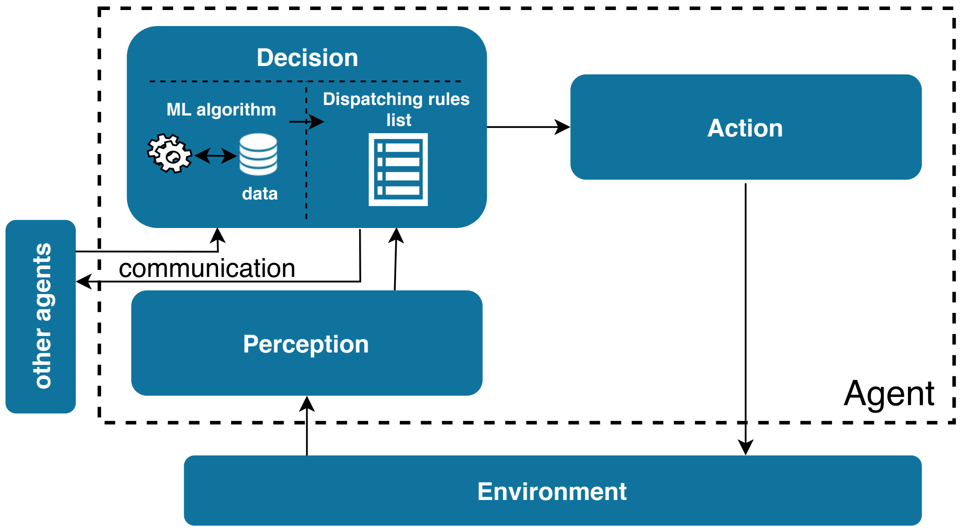


Figure 4.4: Agent structure incorporated with ML techniques for security.

normal functioning pattern or abnormal pattern. As the agents are able to consult the data stored in the database, after completing the transport of each part, it will verify the operation of the system regarding the transport of the part previously carried out. With this, the agent itself plays the role of an intrusion detection system.

4.2.5 Parameters of ML models

To choose the parameters of the machine learning models, a grid search was performed to obtain models with the best parameters. The grid search performs the application of the classifiers in the dataset created, varying the parameters of the model in order to measure the performance of each model with each parameter used and, finally, provides the parameters that had the best performance in the model. Tables 4.6 and 4.7 show the parameters of the algorithms used in this work for the different datasets.

4.3 Visualization of system security conditions

In order to better visualize the application of ML methods in the system, a graphical interface was created. For this, the Node-RED [52] platform was used, which consists of a

Table 4.6: Parameters of ML algorithms applied to the first and second dataset.

Classifier	**Parameters**
Random Forest	trees = 200 and depth = None
Decision Tree	trees = 200 and depth = None

XGBoost	booster = GBTREE
	trees = 100 and depth = 8
	learning rate = 0.3

SVM	kernel = RBF
	C = 1
	gamma = auto deprecated

MLP	hidden layers = (15, 15, 15)
	activation = relu
	alpha = 0.0001
	solver = adam
	maximum number of iterations = 500

Table 4.7: Parameters of ML algorithms applied to the third dataset.

Classifier	**Parameters**
Random Forest	trees = 100 and depth = 30
Decision Tree	trees = 100 and depth = 10

XGBoost	booster = GBTREE
	trees = 100 and depth = 8
	learning rate = 0.3

SVM	kernel = RBF
	C = 1
	gamma = 0.1

MLP	hidden layers = (11, 11, 11)
	activation = relu
	alpha = 0.0001
	solver = adam
	maximum number of iterations = 200

flow-based visual programming tool that simplifies the development of IoT applications. Figure 4.5 illustrates the flows used to create the user interface.

To identify in real time the security status of the system, a topic and content were defined to allow the connection between the publisher and the subscriber. Thus, in the Java programming of the agent, the topic was defined as "Pattern/" and the content with the

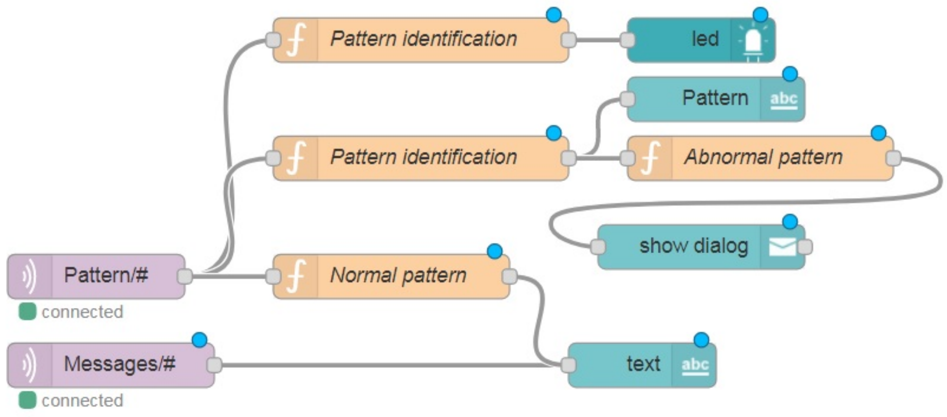


Figure 4.5: Node-RED flow for the development of the graphical interface.

messages "Normal" or "Abnormal". Therefore, the MQTT input node was used in Node-RED, ensuring that the published message from the agent to the broker can be received according to the specified topic. For this node, the topic was defined as “Pattern/#” (see Figure 4.5) to allow receiving all messages in the Pattern topic, regardless of the name of the agent posting the message.

Another topic was created, being the topic "Messages/". In case of a normal situation received in the topic Pattern described above, the content that will always be inserted will be "All messages are being processed". In the case of an abnormal situation received in the Pattern topic, agents will start to publish in the Messages topic the content with their respective name and the message in which they ignored so as not to affect the system, such as "myAgent41:Attention!! Message ignored: tokenTransmissionOutput". In the same way as before, the MQTT input node was used in Node-RED, ensuring that the published message from the agent to the broker can be received according to the specified topic. For this node, the topic was defined as “Messages/#” (see Figure figure 4.5) to allow receiving all messages in the Messages topic.

In addition to the MQTT input nodes, the function nodes responsible for identifying the content published by the agents, the nodes responsible for inserting the information on the dashboard, the node responsible for inserting a led on the dashboard (for better interaction with the user) and also a node for displaying a pop-up window were created.

Chapter 5

Results and Discussions

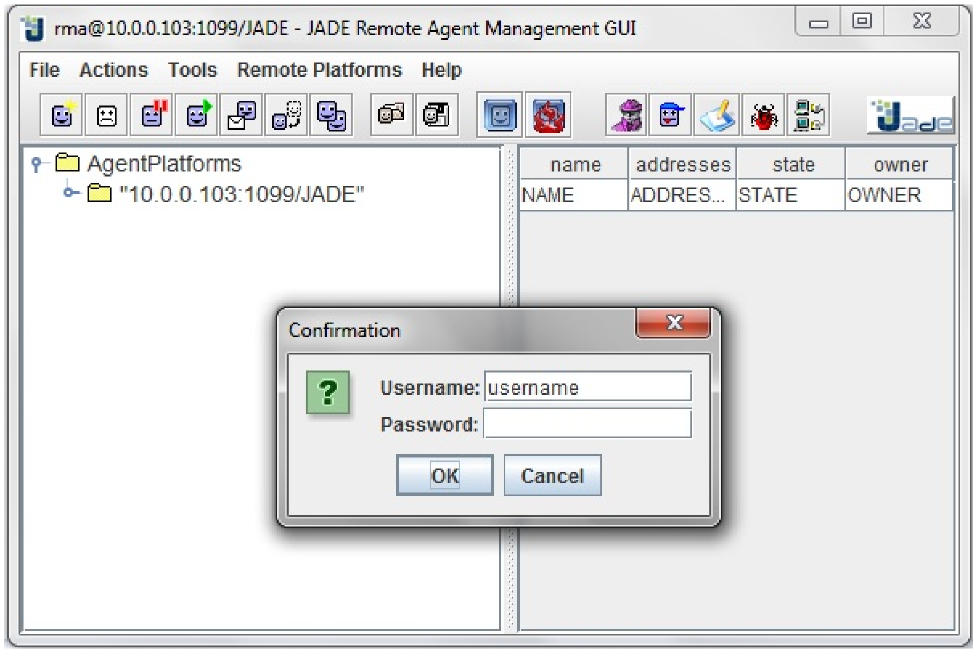
This section aims to describe the results obtained with the JADE-S add on and the ML techniques applied in the case study described above, discuss the most relevant points found in the results, as well as define the main parameters for choosing the algorithm to be loaded into the system.

5.1 JADE-S

Some features present in JADE-S were used to add security to the system in relation to authentication, non-repudiation, integrity and confidentiality.

Regarding the authentication mechanism, this feature was used to ensure that all actions performed on the platform were verified using a user and password. Thus, when performing any action on the platform, such as adding or removing agents (which are running on the Raspberry Pi in the modules), user authentication will be performed to release and perform the action, as depicted in Figure 5.1.

Regarding the signing and encryption of messages, these resources are performed and managed by the JADE platform. By default the data uses an encoding that can be easily decoded using add-ons. The use of signature and encryption features allows the origin of the message to be identified and guarantees the integrity and confidentiality of the message sent. Thus, there is a greater protection of data, especially in relation to



name	addresses	state	owner
NAME	ADDRES...	STATE	OWNER

Username: username
Password:

Figure 5.1: Authentication step on the JADE platform.

the critical data exchanged between agents, who are able to guide the functioning of the system. These resources are applied to the entire payload, but the crucial data is in the message content, which contains values that, when interpreted by the agents, are able to start or stop the motors and update the information regarding the positioning of each module in the work sequence.

5.2 Analysis of the algorithms applied to the first dataset

In this section, the analysis of the algorithms applied to the first dataset will be carried out, which consists of data referring to interactions between agents, considering the main characteristics present in the messages exchanged.

To evaluate the ML models implemented, parameters were used to measure performance, namely accuracy, precision, recall and f-measure.

These parameters used to measure performance have been explained previously, but it is necessary to explain them in the context of the case study in order to better understand the meaning provided by each measure.

Knowing that these measures directly depend on the measures of false positive, false negative, true positive and true negative, consider the situation in which the source of the message can be distinguished into a message coming from a computer belonging to the system and a message coming from an attacker.

True positive: the true positives are the messages that actually come from a computer belonging to the system and have been classified as having the source from a computer belonging to the system.

False positive: False positives are the messages that actually come from an attacker and have been classified as having the source of a computer belonging to the system.

True negative: the true negatives are the messages that really come from an attacker and have been classified as having the source of an attacker.

False negative: False negatives are the messages that really come from a computer belonging to the system and have been classified as having the source of an attacker.

These concepts can be better understood by looking at Table 5.1 where a Confusion Matrix is shown. The confusion matrix shows the number of messages belonging to each category according to the classification made, and it is also possible to understand whether the classification was performed well or not.

Table 5.1: Confusion matrix example

Actual class / Predicted label	Normal	Intrusions
Normal	True Positive (TP)	False Negative (FN)
Intrusions	False Positive (FP)	True Negative (TN)

Having this in mind, it is possible to determine the measurement values previously described, such as accuracy, precision, recall and f-measure.

First, the accuracy parameter was observed in the first data set in order to verify the performance of the five algorithms used, as described in Figure 5.2.

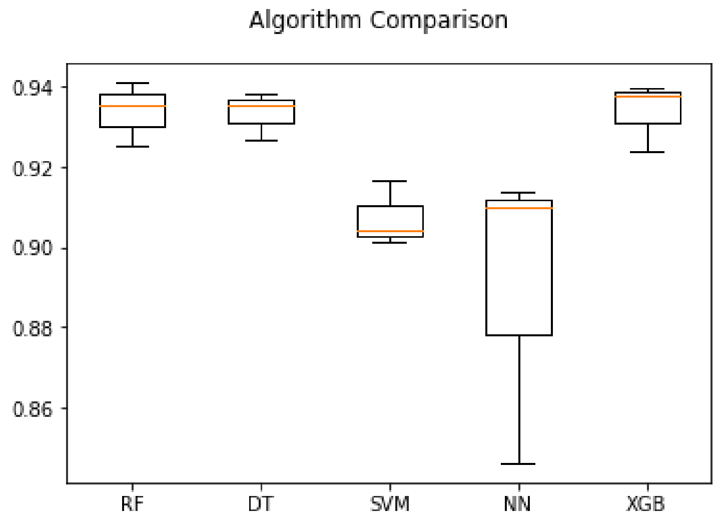


Figure 5.2: Algorithm comparison.

Note that 3 of the 5 algorithms showed excellent results, ranging from 92% to 94%. However, the accuracy measure alone is not enough to know if the algorithm will really be able to perform optimal classifications and in addition the data is not balanced, which can interfere with the accuracy result. Therefore, the other metrics were also calculated and are shown in Table 5.2.

Table 5.2: Accuracy, precision, recall and f-measure results obtained for the first data set

Algorithm	Class	Precision	Recall	F-measure	Accuracy
Random Forest	0	0.95	0.99	0.97	94.35
	1	0.92	0.68	0.78	
Decision Trees	0	0.94	0.99	0.97	94.27
	1	0.93	0.67	0.78	
Support Vector Machine	0	0.93	0.98	0.96	92.33
	1	0.84	0.60	0.70	
Multi-layer Perceptron	0	0.94	0.98	0.96	92.80
	1	0.85	0.62	0.72	
XGBoost	0	0.95	0.99	0.97	94.47
	1	0.93	0.68	0.78	

With Table 5.2, it can be seen that all the algorithms used present excellent results for the classification of data belonging to class 0 (reliable messages), which is justified due to the large percentage of this data in the database. On the other hand, for data belonging to class 1 (unreliable messages), although the algorithms present good precision values (from 84% to 93%), the values obtained for the recall (sensitivity) are very low from 60% to 68%. As such, most messages (84% to 93%) will be classified as reliable but at least 32% of these messages may not be really reliable.

In addition, the f-measure parameter (harmonic mean of precision and recall) are between 96% and 97% for class 0 (reliable) and 70% and 78% for class 1 (unreliable), indicating that on average 96% / 97% of reliable messages will be classified correctly and 70% / 78% of unreliable messages will be classified correctly, the latter being a worrying result for the functioning of the system, since most unreliable messages will be accepted by agents.

Other parameters were also collected, namely training time, predict time, false positive rate and false negative rate, as shown in Table 5.3.

Table 5.3: Training time, predict time, false positive rate and false negative rate results obtained for the first data set

Algorithm	Training time (s)	Predict time (s)	FP rate	FN rate
Random Forest	1.181	0.194	0.010	0.321
Decision Trees	1.400	0.232	0.009	0.330
Support Vector Machine	7.669	2.053	0.019	0.400
Multi-layer Perceptron	19.180	0.006	0.018	0.376
XGBoost	1.742	0.042	0.008	0.322

It is noted that the MLP method has a much higher computational cost than the other methods, but on the other hand it has the shortest time to make the predictions. The RF, DT and Extreme Gradient Boosting (XGBoost) methods have not much different computational costs, but the XGBoost has a great advantage with the low predicting time. Regarding the rates of false positives and negatives, it is expected that these rates have the lowest possible value, as this means the error rate of these techniques. Note that for the rate of false positives the value varies between 0.8% and 1.9% while for false negatives the value varies between 32.1% and 40%. This shows that in both methods most of the messages that are not really trusted have been classified as being reliable.

After analyzing the results obtained with the application of the methods in the first data set, it was observed that the application of any of the methods would not be very reliable for the functioning of the system, being necessary to resort to the extraction of new features in order to try improve the results presented by these methods.

5.3 Analysis of the algorithms applied to the second dataset

The results obtained with the second set of data will be presented and compared with the results obtained previously in order to analyze the improvement of the results with the creation of new features. Thus, the performance of the five algorithms used was calculated with the new data set, as described in Figure 5.3.

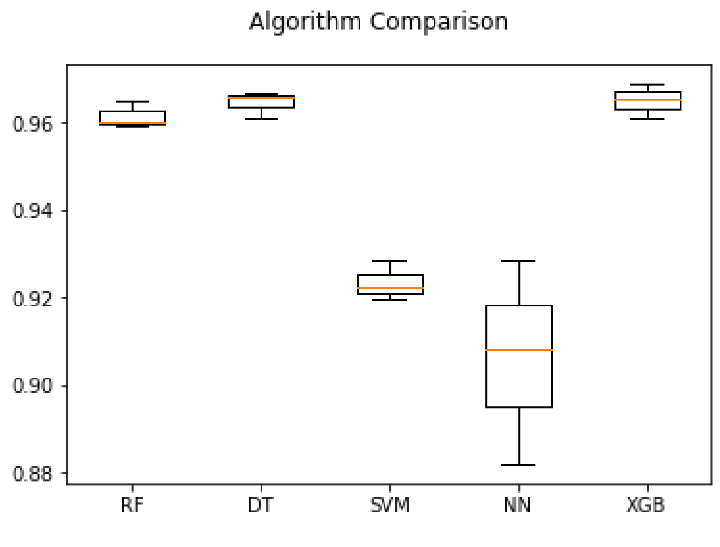


Figure 5.3: Algorithm comparison.

It is noted that all methods presented a superior performance with the use of the new data set and again the same 3 algorithms that obtained better results in the previous data set presented better results in the new data set. But as already mentioned, the accuracy values alone are not enough to know if the classification will be considerable. Table 5.4 shows the values obtained for the parameters of precision, recall and f-measure.

Table 5.4: Accuracy, precision, recall and f-measure results obtained for the second data set

Algorithm	Class	Precision	Recall	F-measure	Accuracy
Random Forest	0	0.98	0.99	0.98	97.25
	1	0.93	0.88	0.90	
Decision Trees	0	0.98	0.99	0.98	97.30
	1	0.94	0.87	0.91	
Support Vector Machine	0	0.95	0.99	0.97	94.02
	1	0.90	0.67	0.77	
Multi-layer Perceptron	0	0.96	0.97	0.97	94.48
	1	0.84	0.77	0.81	
XGBoost	0	0.98	0.99	0.98	97.37
	1	0.95	0.87	0.91	

With the values presented in Table 5.4, there is a great improvement in relation to the results obtained previously. For the recall and fscore values, the values showed an improvement of 20% and 13%, respectively. Regarding accuracy, the values showed an improvement of 3%. Regarding precision, the values remained high.

Therefore, it is noted that the creation of a new feature added a lot to the model, making all methods present good results for application.

For the selection of one of the five proposed methods, in addition to the parameters of precision, recall, f-measure and accuracy, values of training time, predict time, the rate of false positives and false negatives will also be used, in order to make a choice based on in a great classification combined with a low computational cost, a low error rate and a short time to perform the classifications. The data regarding training time, predict time, the rate of false positives and false negatives for each method are described in Table 5.5.

It is noted that the SVM and MLP methods have a high computational cost compared to the others and still have the highest false negative rates (32.6% and 22.7%), which are undesirable for the application.

Table 5.5: Training time, predict time, false positive rate and false negative rate results obtained for the second data set

Algorithm	Training time (s)	Predict time (s)	FP rate	FN rate
Random Forest	1.390	0.190	0.010	0.123
Decision Trees	1.626	0.262	0.009	0.125
Support Vector Machine	9.309	2160	0.013	0.326
Multi-layer Perceptron	10.229	0.037	0.025	0.227
XGBoost	2.017	0.039	0.008	0.13

The RF, DT and XGBoost methods prove to be more attractive for the application. As for training time (computational cost), the RF method has an advantage, but the difference between the others is not so great. In relation to the predicting time, this measure has great weight in the decision, as an algorithm is expected that is capable of making the predictions as quickly as possible so that the functioning of the system is not affected. The XGBoost method has a great advantage in this aspect of predicting time, being 80% faster than the RF method and 85% faster than the DT method. In relation to false positive and negative rates, the best false positive rate is the XGBoost method and for false negatives it is the RF method.

Thus, with the analysis of all these parameters, the XGBoost method was selected to be applied to the system, as it presents better results compared to the others for the application scenario.

As the XGBoost method was selected for application, the results obtained for the parameters of precision, recall, f-measure and accuracy by this method will be analyzed in relation to its application in the system.

First of all for class 0 (reliable messages to the system), 98% of messages will be correctly classified as trusted by the system and 1% of these messages may not be really trusted by the system, but on average 98% of messages will be classified correctly.

For class 1 (unreliable messages to the system), 95% of messages will be correctly classified as unreliable and 13% of these messages may not actually be attacks on the system. In general, 91% of messages considered to be unreliable will actually be classified as unreliable.

Finally, in the global operation of this method, an average of 97.37% of all messages will be classified correctly.

5.4 Analysis of the algorithms applied to the third dataset

This section will present the analysis of the algorithms applied to the third dataset that consists of the data related to the information of the messages exchanged for the transport of each piece that will be used for intrusion detection.

As already shown in the previous analyzes, the accuracy measure alone is not enough to know the performance of the algorithms, especially because the data are not balanced. Thus, Table 5.6 presents the values for the metrics of precision, recall, f-measure and accuracy of the 5 algorithms used with the last data set.

Table 5.6: Accuracy, precision, recall and f-measure results obtained for the third data set

Algorithm	Class	Precision	Recall	F-measure	Accuracy
Random Forest	0	0.98	1.00	0.99	98.14
	1	1.00	0.86	0.93	
Decision Trees	0	0.99	0.99	0.99	98.76
	1	0.95	0.95	0.95	
Support Vector Machine	0	0.90	1.00	0.95	90.12
	1	1.00	0.27	0.43	
Multi-layer Perceptron	0	0.93	0.99	0.96	93.20
	1	0.92	0.55	0.69	
XGBoost	0	0.98	1.00	0.99	98.14
	1	1.00	0.86	0.93	

It can be seen with the results that the algorithms showed excellent results, being between 90% and 98% of accuracy. The SVM and MLP algorithms not only present the

Table 5.7: Training time, predict time, false positive rate and false negative rate results obtained for the third data set

Algorithm	Training time (s)	Predict time (s)	FP rate	FN rate
Random Forest	0.222	0.024	0	0.136
Decision Trees	0.234	0.032	0.007	0.045
Support Vector Machine	0.060	0.004	0	0.727
Multi-layer Perceptron	1.835	0.002	0.007	0.454
XGBoost	0.152	0.001	0	0.136

worst accuracy results, but also the values for recall (sensitivity), which implies the low performance of the algorithm, as many messages will be classified as an attack without actually being, not being suitable for application.

The RF, DT and XGBoost algorithms showed similar precision, recall, f-measure and accuracy values, all of which are considered to be excellent for application.

For the selection of one of the five proposed methods, training time, predicting time, the rate of false positives and false negatives were used again, in order to make a choice based on an excellent classification combined with a low computational cost, low error rate and low time to perform the classifications. These data for each method are described in Table 5.7.

It is observed that the algorithms that were considered not suitable for the application in this case have a very high false negative rate of 45.4% and 72.7%.

Regarding the RF, DT and XGBoost algorithms, these algorithms showed much lower values. Although the RF and XGBoost methods presented a 0% false positive rate and a 13.6% false negative rate, the values presented by DT are more attractive, as they present 0.7% for false positives and 4.5% for false negatives, presenting a much smaller error than the others. On the other hand, the DT method has the longest predicting time and also the highest computational cost than the RF and XGBoost methods. However, for the scenario of application of the second approach, it was noticed that it is not necessary to have such a short predict time, since the time obtained is already sufficient for application.

Therefore, the algorithm that best fits for application is DT.

As the DT method was selected for application, the results obtained for the parameters of precision, recall, f-measure and accuracy by this method will be analyzed in relation to its application in the system.

Firstly for class 0 (normal system behavior), 99% of the message sets will be correctly classified as normal to the system and 1% of that message set may not be really normal to the system, but on average 99% of the message sets will be correctly classified.

For class 1 (intrusion detection or abnormal system behavior), 95% of the message set will be correctly classified as containing system intrusion and 5% of that message set may not actually be an intrusion to the system. In general, 95% of this set of messages considered to contain intrusion to the system will actually be classified as intrusion.

Finally, in the global operation of this method, an average of 98,76% of all messages will be classified correctly.

5.5 Integration of algorithms in the system

This section aims to describe the operation of the system in practice with the methods employed. The results obtained for the different methods were presented in the previous sections, making it possible to choose a method for each application scenario. In addition, examples of the graphical interface were presented, whose function is the real-time monitoring of system security conditions. Therefore, some rules on the agent's behavior have been agreed to simultaneously address the two application scenarios described previously in Section 4.2.4.

Initially, agents only perform the intrusion detection on the system (see diagram in Figure 4.2) for each transported part checking the information pertaining to the message set.

If an intrusion is detected in the system, the agents start to check the next received messages, checking one by one, being able to accept or ignore the message in order to protect the system (see diagram in Figure 4.3). Even when they ignore the messages, the information received is sent to the database by the agent that received it and the agents continue with the intrusion check on each part transported. After checking the normality of the next 100 transported pieces, i.e. if there is no intrusion detection for the passage of 100 consecutive pieces, the agents only perform the intrusion check again. This quantity of 100 pieces was verified experimentally, which would be the best quantity to ensure that the system was not being attacked and could disable the technique for checking new messages, but it can be modified in code if necessary. This procedure works in a constant cycle, each time an intrusion is identified, this process is repeated until normality returns. The diagram shown in Figure 5.4 illustrates this process.

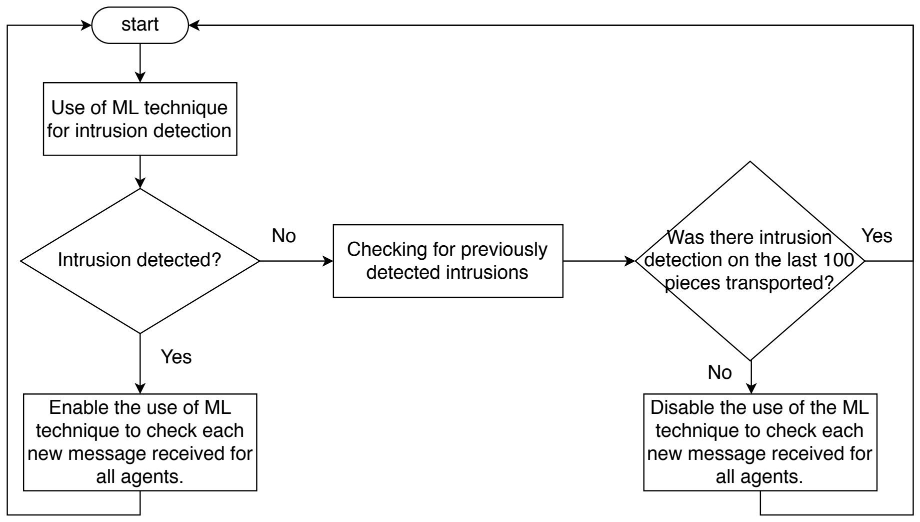

start
Use of ML technique for intrusion detection
Intrusion detected?
No
Checking for previously detected intrusions
Was there intrusion detection on the last 100 pieces transported?
Yes
Yes
Enable the use of ML technique to check each new message received for all agents.
No
Disable the use of the ML technique to check each new message received for all agents.

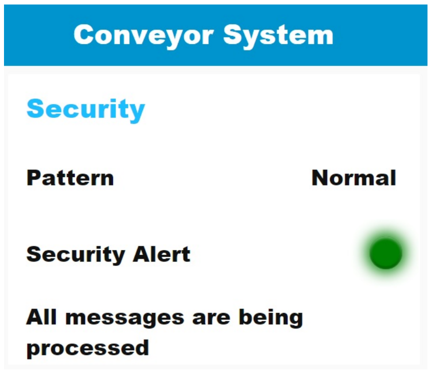
Figure 5.4: Process of operation of the two approaches in the system.

The XGBoost algorithm was chosen for the verification of intrusion in the system and DT algorithm for the individual verification of new messages received. As the agent is programmed in the Java language and the ML methods are programmed in the Python language due to the greater resources available, a socket communication was carried out between the two implemented codes, making the agent implemented in Java able to invoke methods in Python in its behavior. For testing purposes, ML methods are being compiled in the cloud, along with the JADE platform that houses the agents. In addition, the simple security mechanisms present in JADE-S were implemented in an agent that lives in a container on the computer.

5.6 Graphical interface

A graphical interface was created to monitor in real time all the security conditions of the entire system. As seen in section 4.3, the two topics created (messages and pattern) in the Node-RED diagram flow are used to receive information from all agents that publish the information to be displayed on that interface.

The visualization occurs according to the following operation: when publishing on the Pattern topic, agents send the payload with normal or abnormal status. If normal information is received, the information that all messages are being processed will appear on the graphical interface and a green light will appear, indicating safe condition, as depicted in Figure 5.5.


Conveyor System

Security

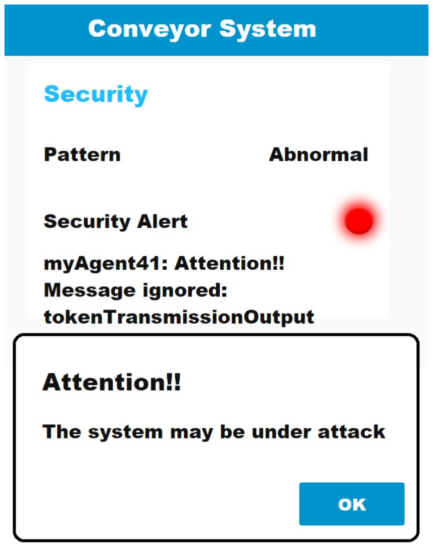
Pattern Normal

Security Alert ●

All messages are being processed

Figure 5.5: Example of the graphical interface under normal security conditions.

Otherwise, if the payload is sent with abnormal status, the light color is changed to red and a pop-up window is displayed indicating an alert condition, as depicted in Figure 5.6. In this case, whenever the agents ignore a message because they have predicted that the message is an attack, the agent publishes in the message topic saying that the received message was ignored and also the protocol information of the received message.



Pattern Abnormal

Security Alert

myAgent41: Attention!!
Message ignored:
tokenTransmissionOutput

Attention!!

The system may be under attack

Figure 5.6: Example of the graphical interface under abnormal security conditions.

Chapter 6

Conclusions and Future Work

The Industry 4.0 introduces several transformations in industry and other sectors, particularly driven by digital transformation. The adoption of the new generation of sensors, actuators and other wireless components in the industrial environment, opens up new doors for various attacks that can cause serious damage to industrial production systems. The main problems identified in industrial environments are the trust between the components of the environment, the confidentiality and integrity of the data exchanged and the low processing power of the devices participating in the processes. As CPS are systems characterized by the combination of the cyber and physical counterparts, constituting the backbone to implement principles of Industry 4.0, the implementation of these systems requires special attention to security requirements.

In this context, the present work developed mechanisms to increase security in a cyber physical conveyor system, from simpler mechanisms such as authentication, signature and encryption, to other more complex mechanisms, such as the use of embedded ML techniques to the MAS.

Before implementing the security mechanisms, the threat modeling based on the STRIDE methodology was analyzed in the case study architecture, showing the main risks and possible mitigations to reduce the attack vectors.

To implement simple security mechanisms, the JADE-S add-on was used, enabling user authentication and the signing and encryption of messages. As a result, the confidentiality and integrity of the data was obtained, preventing external entities from obtaining information from the system or entering false information.

For the development of the ML model, several classification methods were analyzed, such as Random Forest, Decision Tree, Multi-layer Perceptron, Support Vector Machine and Extreme Gradient Boosting. A comparison was made between these methods, not only in relation to performance with the use of metrics of accuracy, precision, recall and f-measure, but also in relation to computational cost and error rate for different classes. This comparison allowed the selection of the best method for each of the two scenarios used in the work, in order to maintain the on-the-fly reconfigurability of the system.

For the first scenario, which consisted of predicting each message received by agents in the reliable or unreliable classes, adding intelligence to the agent to make the decision between accepting or ignoring the message, the XGBoost method showed better results, presenting an average of 97.37% correct answers in the classification, being the best method for practical application. In addition, it was observed that only the information present in the system did not provide excellent results, and it was necessary to improve the model by creating more information in the system.

In relation to the second scenario, which consists of detecting intrusion into the system, the DT method showed better results, presenting an average of 98.76% correct answers in the classification, and was chosen for practical application.

The XGBoost and DT methods were implemented in the system, adding intelligence to the system, maintaining decentralized control, dynamic operation and mainly protecting the system against tampering.

In addition, the creation of the graphical interface allowed a great visualization and monitoring of the system's security conditions, proving to be very efficient.

Future work will be devoted to:

- Implement the permission features present in the JADE-S add on that will guarantee the reduction of the attack vector in relation to the elevation of privilege.

- Implement different attacks with the system's communication protocols and include them in the training data to make the system more robust in relation to the number of attacks.

- Generate authority certificate for communication between agents and database, because in the intrusion detection scenario the agent needs to query the database.

- Configure the firewall to eliminate traffic or limit the size of requests received by agents, protecting against DDoS.

- Make the communication channel between Java and Python more secure, using SSL/TLS.

- Perform a study in relation to computational load before and after the application of the algorithms.

Bibliography

[1] *Internet of Things: connected devices to triple by 2021*, (Accessed: 2020-02-15). [Online]. Available: `https://www.juniperresearch.com/press/press-releases/%5C%e2%5C%80%5C%98internet-of-things%5C%e2%5C%80%5C%99-connected-devices-triple-2021`.

[2] P. Leitão, S. Karnouskos, and A. Colombo, "Industrial Automation based on Cyber-Physical Systems Technologies: Prototype Implementations and Challenges," *Computers in Industry*, vol. 81, pp. 11–25, 2016.

[3] E. Lee, "Cyber physical systems: design challenges," in *Proceedings of the 11th IEEE International Symposium on Object/component/service-oriented real-time Distributed Computing*, May 2008, pp. 440–451.

[4] H. Kagermann, W. Wahlster, and J. Helbig, "Securing the Future of German Manufacturing Industry: Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0," ACATECH – German National Academy of Science and Engineering, Tech. Rep., 2013.

[5] P. Leitão, J. Barbosa, G. Funchal, and V. Melo, "Self-organized Cyber-Physical Conveyor System using Multi-agent Systems," *Accepted to be published in the International Journal of Artificial Intelligence*, 2020.

[6] M. Wooldridge, *An Introduction to Multi-Agent Systems.* John Wiley and Sons, 2002.

[7] P. Leitão, "Agent-based Distributed Manufacturing Control: A State-of-the-art Survey," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 7, pp. 979–991, Oct. 2009. DOI: `10.1016/j.engappai.2008.09.005`.

[8] J. Barbosa, P. Leitão, and J. Teixeira, "Empowering a Cyber-Physical System for a Modular Conveyor System with Self-organization," in *Service Orientation in Holonic and Multi-Agent Manufacturing, Studies in Computational Intelligence*, T. Borangiu, D. Trentesaux, A. Thomas, and O. Cardin, Eds., vol. 762, Springer, 2017, pp. 157–170.

[9] S. Naik and V. Maral, "Cyber security — IoT," in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*, May 2017, pp. 764–767. DOI: `10.1109/RTEICT.2017.8256700`.

[10] T. Poppensieker and R. Riemenschnitter, "A new posture for cybersecurity in a networked world," in *Perspectives on transforming cybersecurity*, McKinsey&Company, 2019, ch. 2.

[11] Cisco, *2018 Annual Cybersecurity Report.* 2018.

[12] L. Sakurada, "Development of Industrial Agents in a Smart Parking System for Bicycles," Master's thesis, Polytechnic Institute of Bragança, Portugal, 2019.

[13] A. Hellinger, H. S. Translation, J. Macfarlane, B. Services, and H. Galloway, "Cyber-physical Systems Driving Force for Innovation in Mobility, Health, Energy and Production Acatech (ed.)."

[14] A. B. Chebudie, R. Minerva, and D. Rotondi, "Towards a definition of the Internet of Things (IoT)," PhD thesis, Aug. 2014.

[15] *Cyber-Physical Systems*, `https://ptolemy.berkeley.edu/projects/cps/`, Accessed: 2020-02-10.

[16] N. R. Jennings, K. Sycara, and M. Wooldridge, *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 1, pp. 7–38, 1998. DOI: `10.1023/a:1010090405266`.

[17] J.-H. Lee and C.-O. Kim, “Multi-agent systems applications in manufacturing systems and supply chain management: A review paper,” *International Journal of Production Research*, vol. 46, no. 1, pp. 233–265, Nov. 2007.

[18] M. Wooldridge and N. R. Jennings, “Intelligent agents: Theory and practice,” *The Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, Jun. 1995. DOI: `10.1017/s0269888900008122`.

[19] L. Monostori, J. Váncza, and S. Kumara, “Agent-based systems for manufacturing,” *CIRP Annals*, vol. 55, no. 2, pp. 697–720, 2006. DOI: `10.1016/j.cirp.2006.10.004`.

[20] *FIPA, FIPA ACL Message Structure Specification*, `http://www.fipa.org/specs/fipa00061/SC00061G.html#_Toc26669700`, Accessed: 2020-02-08.

[21] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*. John Wiley & Sons, Mar. 13, 2007, 300 pp., ISBN: 9780470058404. [Online]. Available: `https://www.ebook.de/de/product/15097614/fabio_luigi_bellifemine_giovanni_caire_dominic_greenwood_developing_multi_agent_systems_with_jade.html`.

[22] *MQTT*, `http://mqtt.org/faq`, Accessed: 2020-02-08.

[23] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, “MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks,” in *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE ’08)*, Jan. 2008, pp. 791–798. DOI: `10.1109/COMSWA.2008.4554519`.

[24] S. Potluri, N. F. Henry, and C. Diedrich, “Evaluation of hybrid deep learning techniques for ensuring security in networked control systems,” in *Proceedings of the 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA’17)*, Sep. 2017, pp. 1–8. DOI: `10.1109/ETFA.2017.8247662`.

[25] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and Other Botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017, ISSN: 1558-0814. DOI: 10.1109/MC.2017.201.

[26] P. Varga, S. Plosz, G. Soos, and C. Hegedus, "Security Threats and Issues in Automation IoT," in *2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS)*, May 2017, pp. 1–6. DOI: 10.1109/WFCS.2017.7991968.

[27] J. D. Morenas, C. Miller, G. S. Funchal, V. Melo, M. Vallim, and P. Leitão, "Security Experiences in IoT based applications for Building and Factory Automation," in *Proceedings of the IEEE International Conference on Industrial Technology (ICIT'20)*, 2020.

[28] P. Leitão, S. Karnouskos, L. Ribeiro, J. Lee, T. Strasser, and A. W. Colombo, "Smart agents in industrial cyber–physical systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1086–1101, May 2016, ISSN: 1558-2256. DOI: 10.1109/JPROC.2016.2521931.

[29] S. Karnouskos, "Industrial agents cybersecurity," in *Industrial Agents*, Elsevier, 2015, pp. 109–120.

[30] J. G. Carbonell, R. S. Michalski, and T. M. Mitchell, "Machine learning: A historical and methodological analysis," *AI Magazine*, vol. 4, no. 3, p. 69, Sep. 1983.

[31] T. Ayodele, "Types of machine learning algorithms," in. Feb. 2010, ISBN: 978-953-307-034-6. DOI: 10.5772/9385.

[32] L. Rokach and O. Maimon, "Decision trees," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Boston, MA: Springer US, 2005, pp. 165–192, ISBN: 978-0-387-25465-4. DOI: 10.1007/0-387-25465-X_9. [Online]. Available: https://doi.org/10.1007/0-387-25465-X_9.

[33] L. Rokach and O. Maimon, "Top-down induction of decision trees classifiers - a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 4, pp. 476–487, Nov. 2005, ISSN: 1558-2442. DOI: `10.1109/TSMCC.2004.843247`.

[34] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, Mar. 1986. DOI: `10.1007/bf00116251`.

[35] J. R. Quinlan, *C4.5*. Elsevier Science & Technology, Dec. 2, 1992, 302 pp., ISBN: 1558602380. [Online]. Available: `https://www.ebook.de/de/product/3736008/j_ross_quinlan_c4_5.html`.

[36] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Taylor & Francis Ltd, Jan. 1, 1984, 368 pp., ISBN: 0412048418.

[37] J. H. Friedman, "Machine.," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001. DOI: `10.1214/aos/1013203451`.

[38] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Aug. 2016, pp. 785–794. DOI: `10.1145/2939672.2939785`.

[39] A. Verikas, A. Gelzinis, and M. Bacauskiene, "Mining data with random forests: A survey and results of new tests," *Pattern Recognition*, vol. 44, pp. 330–349, Feb. 2011. DOI: `10.1016/j.patcog.2010.08.011`.

[40] L. Breiman, *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. DOI: `10.1023/a:1010933404324`.

[41] I. Basheer and M. Hajmeer, "Artificial neural networks: Fundamentals, computing, design, and application," *Journal of Microbiological Methods*, vol. 43, no. 1, pp. 3–31, Dec. 2000. DOI: `10.1016/s0167-7012(00)00201-3`.

[42] J. Zupan and J. Gasteiger, *Neural Networks for Chemists: An Introduction*. Wiley-VCH, 1993, ISBN: 978-3-527-28603-4.

[43] V. Vapnik, *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Berlin, Heidelberg: Springer-Verlag, 1982, ISBN: 0387907335.

[44] G. Mountrakis, J. Im, and C. Ogole, "Support vector machines in remote sensing: A review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 66, no. 3, pp. 247–259, May 2011. DOI: `10.1016/j.isprsjprs.2010.11.001`.

[45] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation," vol. Vol. 4304, Jan. 2006, pp. 1015–1021. DOI: `10.1007/11941439_114`.

[46] E. Costa, A. Lorena, A. Carvalho, and A. Freitas, "A review of performance evaluation measures for hierarchical classifiers," *AAAI Workshop - Technical Report*, Jan. 2007.

[47] K. Faceli, A. C. Lorena, J. Gama, and A. C. P. d. L. F. d. Carvalho, *Inteligência artificial: uma abordagem de aprendizado de máquina*. LTC, 2011.

[48] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," in *International Joint Conference on Articial Intelligence (IJCAI)*, 1995.

[49] T. Fushiki, "Estimation of prediction error by using k-fold cross-validation," *Statistics and Computing*, vol. 21, pp. 137–146, Apr. 2011. DOI: `10.1007/s11222-009-9153-8`.

[50] V. Vyatkin, "IEC 61499 as Enabler of Distributed and Intelligent Automation: State-of-the-Art Review," *IEEE Transactions on Industrial Informatics*, vol. 7, pp. 768–781, Dec. 2011. DOI: `10.1109/TII.2011.2166785`.

[51] JADE Board, *JADE Security Guide*. TILAB S.p.A., 2005.

[52] *Node-RED: Low-code programming for event-driven applications*, `https://nodered.org/#features`, Accessed: 2020-01-29.

Appendix A

Source code developed in this work

This appendix comprises the link to access the codes developed in this work, namely ListeningToIncomingMessages.java, Resource.java, BehaviorHacker.java, Hacker.java, GridSearchCV.py, MLAgents.py, IntrusionDetection.py.

The Resource.java file refers to the implementation of the agent that is used in the Raspberry Pi, being the logical part of each module.

The ListeningToIncomingMessages.java file refers to the agent's behavior implemented in the Resource.java file. This behavior aims to configure all the actions that will be taken in the environment according to the agents' perception.

Files Hacker.java and BehaviorHacker.java refer to the attacking agent's implementation and behavior, respectively. This agent has the same characteristics as the system agent, but its behavior consists of capturing the information of messages exchanged by the system and has the action of sending messages with altered contents to the system agents.

The GridSearchCV.py file refers to the code implemented to choose the parameters of the machine learning models developed.

Files MLAgents.py and IntrusionDetection.py refer to the algorithm for checking new messages received by agents and for intrusion detection, respectively.

The access link is: Security-for-a-CPS-based-on-MAS