An approach based on behavioral models and critical states distance notion for improving cybersecurity of industrial control systems

Franck Sicard, Éric Zamaï, Jean-Marie Flaus

Version of Record: https://www.sciencedirect.com/science/article/pii/S095183201830262X
Manuscript_3e4aa4a9475c1e47ea4e1bafad7687fd

An approach based on behavioral models and critical states distance notion for improving cybersecurity of industrial control systems

Franck SICARD, Éric ZAMAI, Jean-Marie FLAUS

Univ. Grenoble Alpes, CNRS, Grenoble INP[1], G-SCOP, 38000 Grenoble, France

franck.sicard@grenoble-inp.fr

Abstract Since the beginning of the 21th century, Industrial Control Systems (ICS) have been targeted by hackers. The main motives for the interest to ICS is the ease for performing cyberattacks and the potential damages inflicted to the system and its environment in case of success. The purpose of this paper is to propose an approach for detecting malicious orders in discrete-event system. Four types of attacks (*direct, sequential, temporal* and *over-soliciting*) that affect an industrial system are studied in this work. Based on the vulnerabilities in ICS and the positioning of other techniques, an innovative methodology is exposed in this paper to develop detection mechanisms based on the "automation-knowledge". Thus, by using models of system with an improved notion of distance and trajectory, our filters based approach provides good results for detecting cyberattacks in lower levels of ICS architecture by analyzing the malicious nature of the orders sent. Different types of detection mechanisms based on the concept of distance and trajectory are detailed in this study. We also provide results on simulation examples and an industrial platform. To conclude, improvements of our approach are discussed.

Keywords: Industrial Control Systems; Cyberattacks; Cybersecurity; Critical States Distance; Filter Approach; Fault Detection; Diagnosis; Discrete Event System.

1. Introduction

1.1. Industrial Control Systems (ICS)

Industrial Control Systems (ICS) ensures productivity, reliability and safety in industrial context since the second half of 20th century. Stouffer et al. [1] in the NIST 800-82-r2 defines these systems by a combination of control components that act together to achieve an industrial objective. Nowadays, ICS are used in energy production and distribution (electricity, water, oil and gas), wastewater treatment, discrete manufacturing, transportation, and defense. Thus, these systems are critical infrastructures, highly interconnected and mutually dependent [2]. In order to organize exchanges, roles and functions within an ICS, the Computer Integrated Manufacturing (CIM) norm, also named Purdue Model [3], has been proposed in 80's. The CIM model allows representing ICS with functional levels from top management to operative part. Sicard et al. [4] details aim and constraints for the first three layers of an ICS, represented in Fig 1, that are studied in this paper. The level 0, called Operational Part (OP), is composed of actuators and sensors and transforms raw materials (initial state) into finished product (final state) by interacting with the production flow and applying actions decided by the Control Part (level 1). The level 1, named Control Part (CP), interacts with (i) the layer 0 by acquiring data from sensors and controlling actuators and, (ii) the layer 2 by exchanging information with the operators. The main component of this level is the Programmable Logic Controller (PLC). Finally, the level 2, called Supervision, acquires data from lower layers in order to inform operators about the state of the system and work orders are sent to layer 1 for modifying the control law or adjusting setpoints. Communication layers exchange data between different levels. Since the beginning of the 21th century and the emergence

[1] Institute of Engineering Univ. Grenoble Alpes

of Industry 4.0, Ethernet TCP/IP is become a standard for connecting all different layers of the ICS and introduced at the same time vulnerability inside the ICS architecture. The developed approach proposes to secure existing infrastructure based on the hierarchical architecture.

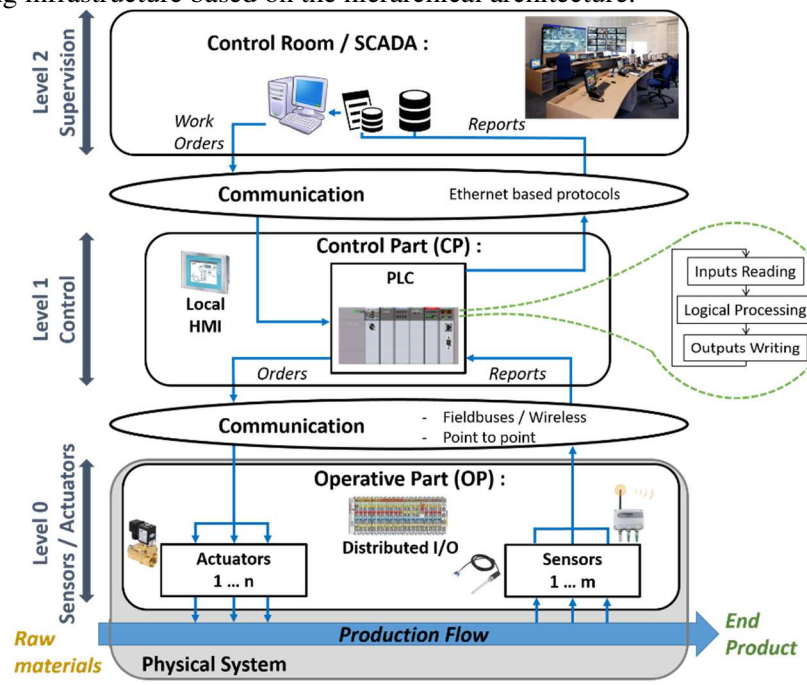

Fig 1. Illustration of functional architecture of ICS (only real-time layers have been represented)

1.2. Attacks against ICS: History, Consequences and Considered Types

Since the beginning of the century, introduction of new technologies as Ethernet based protocols, wireless networks or industry 4.0 in ICS have made it vulnerable to cyberattacks. Indeed, for almost 20 years, ICS are the new target of hackers as detailed in Sicard et al. [5]. These attacks can be performed because ICS have many vulnerabilities from a security point of view. Indeed, these systems have been designed to insure productivity and safety without considering security. Thus, these vulnerabilities give surface attacks to hackers that can be exploited to cause physical damage to the systems and its environment. Indeed, components, architecture and environment of a control system can be used to perform cyberattacks as presented in [6]. Fig 2 summarizes these main vulnerabilities in an ICS [4].

Different types of attacks can be performed on ICS [7] such as information disclosure, injecting false data or also disturbing the production, i.e. the service, the quality or the equipment. This paper focuses on attacks that execute or induce a malicious action at the PLC outputs. Some of them are inherited from IT and mainly affect the layer 2 [8,9] as *DOS* (*Denial of Service*) or *DDOS* (*Distributed Denial of Service*) whose purpose is to stop the communication between two layers, *Man-in-the-middle* (*MITM*) that intercepts and/or injects data and *replay attacks* which returns a sequence already sent. However, other attacks are inherent to levels 0-1 [4,10–12]. Thus, *random attacks* send information without taking into account the process knowledge, *direct attacks* causes great damage to the system without leaving the possibility of reacting and *sequential attacks* break the sequential logic of the control law to damage the system. Finally, *false data injection (FDI) attacks* corrupt either the control orders in the control channel or the reports in the report channel. According to McLaughlin et al [13], these attacks can be performed on several layers divided in five types : hardware, firmware, software, network and ICS process. This last layer uses the previous ones to accomplish the industrial objective. Vulnerabilities of this layer are highlighted in [14–16] and developed in section 2.

This paper focuses on attacks that target the (i) *Control channel* linking the PLC's outputs to the actuators, the (ii) *Report Channel* sending data from sensors to the control part and the (iii) *PLC/Control Room* controlling and monitoring the CP. The proposed approach objective is to detect:

- *Direct attacks*: quickly lead the system to a critical state in order to damage it,

- *Sequential attacks*: degrade the behavior of the ICS and send Order/Report sequences that gradually lead the system closer to a critical state,
- *Temporal attacks*: send orders/reports by violating time constraints (periodicity, timing…),
- *Over-soliciting attacks*: target equipment to make it unusable for production by premature wear and tear, causing maintenance or breaking equipment.

1.3. Issues, positioning and contribution

In addition of these vulnerabilities [6], ICS are subject to failures that degrade the normal behavior. This term is defined in [17] and denotes the degradation of the system or its environment but unintentionally. Contrary to attacks where the degradation is intentional. In this study, we focus on cyberattack that is to say attacks launched with digital vector. In [18], Nguyen et al. identify 4 sources of failure in ICS represented in Fig 2: products' quality, specifications of the control law, human factor and equipment wear.

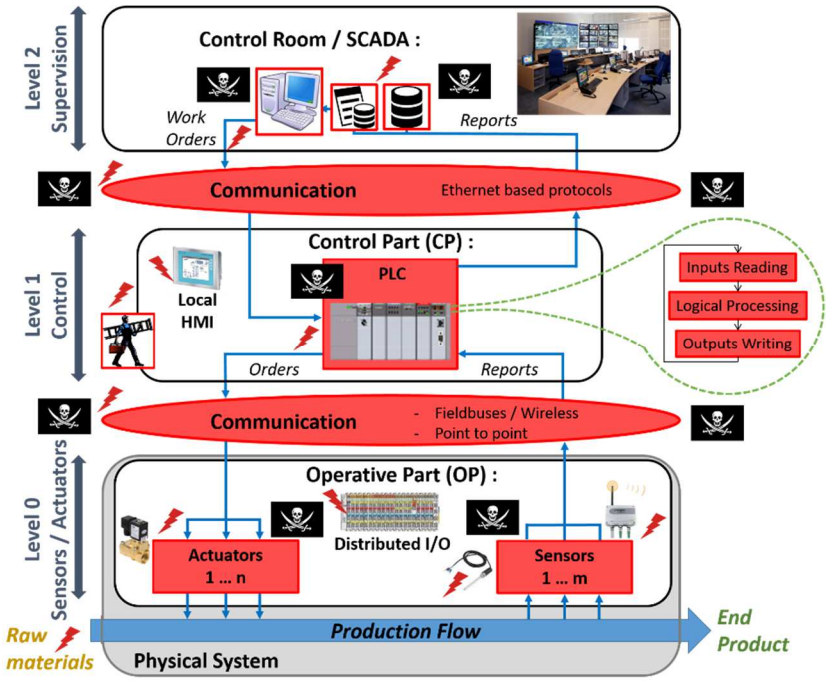

Fig 2. Main vulnerabilities (layers in red with a pirate flag) and failure sources (red flashes) in an ICS

In Information Technology (IT), these vulnerabilities are already known and solutions are proposed as cryptology, firewall, intrusion detection system or VPN [19,20]. However, ICS differ in its specificities comparing with traditional Information System (IS) [4,21] as:

- Control of physical processes,
- Performance requirements (operating in hard conditions of temperature, dust …),
- Equipment lifetime (15-20 years for ICS against 5 years in IS),
- Update policy (firmware or software updates are often not published by editors or patched in industries because production has to be stopped)

These specificities make impossible the use of these solutions or reduce their efficiency if they are applied for the entire architecture. Thus, IT solutions such that DMZ or firewall applied in high level of CIM (level 2 and higher) are effective because these levels are very similar with traditional IT. However, any IT solutions are satisfactory when they deal with real-time layer (level 1 and 0) with "automation knowledge". The developed approach provides a solution for low levels of the CIM architecture by taking into account "automation knowledge" on safety and security aspects. This approach synthetizes filters that analyze the evolution of the system. Detections mechanisms based on models, of the physical system and/or control law, and constraints, combinational and temporal, allow detecting anomaly

comparing to normal behavior. Our approach acts as the last shield for protecting the system from a cyberattack, hackers having crossed previous level of defense. In this study, the proposed approach is integrated. It implies modeling of normal and abnormal behaviors. Filters analyze information exchanged between operative and control part in order to verify if the logic expression exchanged is correct. Thus, the control filter checks if orders sent by the PLC are logical. Our approach detects normal states combined with correct orders or sequences of actions whose concurrency leads to critical state.

The main contribution of this paper is to provide an approach capable of detecting intrusions into the ICS before they damage the physical system (i.e. before occurrence). To do this, two existing approaches focused on security (IDS, Intrusion Detection System) and safety (Filter approach) are hybridized in order to perform detection mechanisms based on behavioral models. First, a generic methodology for designing filters with these models is proposed. Then, indicators allowing to detect attacks, of variable complexity, affecting the behavior of ICS are built and used. The paper focuses on two main issues decomposed into four types of attacks: (i) *direct attack* detection that act in a very short time interval requiring immediate reactivity, (ii) *stealthy attacks* these attacks act over a longer time interval such as sequential, temporal or *over-soliciting attacks*. Thus, an additional detection strategy must be proposed, in particular based on previous works on the notion of distance, in order to indicate a risk or an intention to harm through an event or temporal sequence. Finally, this research work focuses only on discrete-event systems and on existing infrastructure (ICS modeled by CIM architecture).

1.4.Threat model and global hypothesis

In this paper, the filter based approach focuses on attack detection that affects the lower levels of CIM architecture by using “automation-knowledge” of ICS. Only attacks that intend to degrade one of the following elements (assumption of attack unicity) through attacks listed in section 1.2. are considered: production equipment, product quality or the safety of goods and people. Indeed, the S.A.F.E. (Security Approach based on Filter Execution) approach is based on behavioral models from the ICS to detect the dangerousness of a sequence of events (actions/reports). Denial of service or data theft attacks are therefore excluded from this study. On the other hand, data injection attacks on the control channel, the report channel, PLC corruption or supervision control are taken into account in our approach, under the assumption that they degrade one of the elements degrading the ICS mentioned above. On the other hand, we will exclude attacks on the sensor network from the perimeter of the attacks considered. We also assume that the sensor network is considered reliable [22]. To conclude, only cyberattack are considered in this paper, thus operative part cannot be physically damaged to launch an attack.

The S.A.F.E. approach is complementary to the current solutions and recommendations [7]. In this work, the previous defenses are assumed crossed by the attacker, by computer (spam and infiltration on an office network then industrial) and/or locally (intrusion in the supervision or on a PLC). Since the higher levels are considered partially or completely corrupt, our approach is therefore the last line of defense against the attacker to protect the physical system and its environment.

Several hypotheses have been chosen in this study for performing this approach. First of all, the risk assessment identifies required and sufficient parameters to model the system without forgetting critical parameters or facing exponential complexity issues. Filters are considered as secured by design and not attackable from a hardware point of view. Communication between control and report filters is not subject to failure and attack, for example by using encryption and PKI techniques [23]. Then, we suppose that no error is present in the control law and models of the system. In the same way, temporal analysis of the ICS must correctly identify temporal windows for each action and report specified in filters. Finally, the filter-programming environment is assumed to be "secure" in the IT sense of the term and operators inside the ICS are non-hostile.

This paper focuses on the problem of detecting a malicious order and is organized as follows. In section 2, a state of art presents the main field of research for securing ICS, solutions that focus on the process layer and the proposed approach. Section 3 introduces the methodology to model and synthetize filters in function of the level of knowledge of the system. In section 4, the notions of distance and trajectory, which are the keystone of detection mechanisms implemented into filters, are introduced.

Different detection mechanisms are also described and illustrated on case studies. Then, in section 5, the methodology and approach are put into practice several simulation and real platform examples. Finally, section 6 will put into perspectives our results and future works.

2. Toward ICS cybersecurity: from the context to the proposed approach

As explained in Mc Laughlin et al. [13], several fields of research are investigated for securing ICS from cyber issues. In this section, first, we introduce general ways for improving cybersecurity. Then the state of art of techniques that focus on lower levels of CIM architecture is detailled. To conclude this section, our approach, based on security and safety techniques, is explained.

2.1. Generality about cybersecurity

Securing the ICS against the threat of cyberattacks involves security improvements of several elements. Guides written by governmental agencies [7] as ANSSI (France), NIST (USA) or ENISA (EU) recommend some enhancement of different types: technical, employee training, courses, prevention and sometimes certifications. International norms as IEC 61850 for smart grid systems [7] or ISA99 for standardizing defense of ICS against cyberattacks improve global cybersecurity of ICS. Our work focuses on technical aspects of cybersecurity by detecting attacks on specific layers of ICS.

Concerning firmware layer, several works proposed approaches for securing this component whose security issues are detailed in [24]. In Basnight et al [25], a new methodology of analysis is presented based on reverse engineering for identifying weaknesses and proposing countermeasures; then these are implemented in a controller. Other works as Schuett et al [26] or Peck et al [27], evaluate firmware in order to find vulnerabilities and backdoors in PLC or Ethernet cards to create validation algorithms. The hardware layer is also vulnerable to cyberattacks, Ren et al [28] proposed an approach based on machine learning for securing JTAG port. This interface is used for testing product and gives access to firmware and on-chip memory especially. The software layer allows configuring several components in ICS architecture (equipment, PLC, servers, HMI…) and has many vulnerabilities as reported in McLaughlin et al [29]. Solutions investigated for protecting this layer are developed in McLaughlin et al. [13]. Works for securing the network layer focus on several points [30]: architecture typology for reducing surface of attack (firewall, DMZ, communication components) [1], securing communication protocols [31], detection intrusion based on network [32]. Many studies have been carried out for securing this layer based on Intrusion Detection System (IDS) [33], a well established field of research, by using:

- Syntax and semantics of protocols as [34] that verify if communication pattern corresponds to network policies and normal communication behavior for Modbus protocol. The same work has been developed for DNP3 with Lin et al [35]. Morris et al [36] proposed to translate wireless protocol into TCP traffic for analyzing with predefine rules.
- Structure of communication flow as proposed by Barbosa et al [37]. In this work, the hypothesis of strong periodicity in ICS is assumed for detecting anomaly in communication behavior with devices.
- Telemetry that exploits data used for communication as IP addresses, number of transmitted packets, time interval between each frame, … Linda et al [38] performed machine learning techniques to know parameters detailed previously.
- Process knowledge that is based on data used to control the system. [39,40] employs information contained in frame to compute distance to critical state and launch alarms. [41] takes the same assumption with a state space model for detecting modifications due to a fault or an attack.

The ICS process layer controls the physical system especially with control law implemented inside PLC. All this layer is also called “automation knowledge”. As we see in previous section, hackers use component in this layer for performing cyberattacks. Vulnerabilities and approaches for securing this

layer are different from classical information systems. Some works that focus on this layer to highlight the weakness and proposed solutions are presented in next section.

2.2. State of the Art about cybersecurity of ICS Process Layer

Several works use the "automation knowledge" of this layer to protect it from attackers. However, several approaches are proposed for securing lower level of ICS. These approaches are based on models of operative and/or control that include a high level of knowledge of the system. Two kinds of approaches has been developed and are detailed in [42]:

- Signature approaches that are based on recognition of specific behavior as in [43].
- Behavioral approaches are based on specifications of the system. They define rules to insure safety, security and reliability of the system. Mathematical equations (quantitative method) or models (qualitative method) can be used to express these specifications.

Signature approaches are not relevant for this layer because there is no detection for zero-day attacks. By definition, these attacks are unknown when they are performed. Thus, one of the strength of this layer is the possibility to model the system instead of finding signature. In this paper, we will focus on the second type, model based approaches. Different techniques based on different layers have been developed in the literature. Thus, Pièrre-Cambacédès et al [44] model the system with Markovian processes for identifying possible attack scenarios on the physical system. Hadziosmanovic et al [45] propose to build model of the system by monitoring the network. Based on the semantic analysis of frames, variables are extracting (type and values), characterized and analyzed over time. This approach is relevant by its "process awareness" point of view but as learning technique, long time of acquisition is needed. Erez et Wool [46] have developed the same anomaly detection through network analysis in Modbus/TCP [46]. Several works use equational models for detecting intrusions in ICS as Papa et al [47] who compute a transition function and Yaseen et Bayart [48] that use a state-space representation for detecting and distinguishing anomalies. Li et al [12] model the physical system with qualitative method as finite state machine (FSM). Then, different attack scenarios are identified and computed with other FSM. The simulation defines several system evolutions classified with the characterized attack event. Finally, another use of "automation-knowledge" can be found in works about distance notion of Carcano et al [40] and Fovino et al [39]. In these works, the first step is to define the safety area of the system that implies to identify safe and critical states for each variable of the system. Then, the authors compute the distance between the current state of the system and the critical state with the minimal distance of this state. Two definitions of distance notion are given which differ in function of the system and the variables. Thus, distance to critical state is computed at every PLC cycle to prevent the system from entering in a critical area. Carcano et al. [40] provide a powerful notion that allows detecting cyberattack by using process knowledge. However, distance concept has to be improved in several aspects. Indeed, the state vector is computed with data inside of the PLC that is also a vulnerable component of ICS. Moreover, the definition of distance is applied and correct for continuous variables but not applicable for discrete event systems.

2.3. Proposed approach: filters as last shield

In the presented approach, we proposed to combine techniques used in security and in safety fields. Indeed, the issue of detection in cybersecurity is very similar with the problem of detection in safety approaches developed to find failures in a system [49]. This relation is defined in Kriaa et al [15] or in Ellis [50] to identify convergence points between safety and security analysis. Methodologies to introduce combined analysis are proposed in these works. Several works proposed to elaborate scenarios of attacks as in [51] where a model based approach is proposed to generate attack and failure scenarios. Abdo et al [52] propose the same contribution thanks to attack trees with bowtie analysis. Cardenas et al [53] develop an approach to detect computer attack affecting the normal behavior of control system by incorporating knowledge of this system. This knowledge is based on quantitative method with

equational model. To conclude, in Friedberg et al [54], the authors develop a new methodology of analysis for taking into account both safety and security aspects in cyber-physical systems based on vulnerabilities and risks. Then, the best mitigation strategy is identified for the system.

As detailed in Sicard et al [4], the proposed approach is based on:

- A security point of view. Several techniques can be used as firewall, cryptography or even antivirus. Detection mechanisms developed in IDS allow detecting attacker with probes deployed into the network that compared data exchanged with rules defining the policy of the network. Fig 3 illustrates the functioning of IDS [4]. The policy of the network is based on the respect of integrity, availability and confidentiality of data exchanged through the system. However, the rules that prevent from an attack can be modified for ICS issues. In our approach, behavior models are implemented in IDS for detecting intrusion in ICS as in Mitchell et al [42]. Thus, the knowledge to establish the detection rules as well as the location of probes in the network to ensure a good detection will be useful to our approach,
- A safety point of view, the filter approach proposed in Cruette et al [55] is compatible with the cybersecurity context in the ICS. As represented in Fig 4, this approach is based on two verification blocks that check if data exchanged between operative and control parts are correct to control and process models. These filters are able to stop the emission of an order or a report. However, in the original approach, filters are implemented inside of the control part that is not compatible with the cybersecurity context.

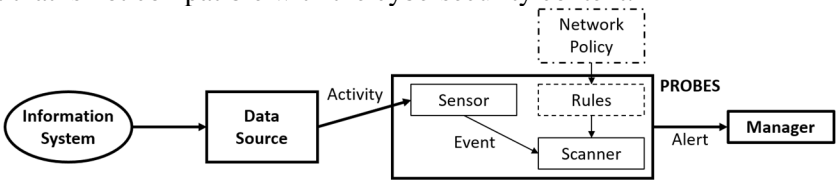

Fig 3. Illustration of the probe functioning in an information system

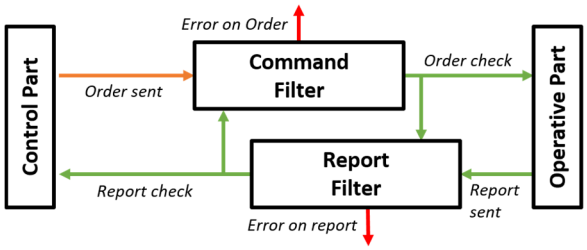

Fig 4. Filter approach developed in Cruette et al [55]

Thus, for combining these two approaches, the number of filters is limited to two: the control filter checks orders exchanged between the PLC and the actuator and the reports filter verifies data sent from sensors to PLC. Contrary to the Cruette approach, filters are outside of the PLC to avoid compromising from attackers. Indeed, this element is vulnerable according the previous section. Their locations in ICS architecture are also a crucial element for the efficiency of detection algorithm. Indeed, in previous works [14], only attack on the control channel are study. In this paper, attacks on report channel are also considered that modifies the location of report filter to optimize attack detections. This contribution is explained in section 3. Fig 5 illustrates the operating principle for each filter.

The control filter analyzes orders sent from the control part based on the information received from report filter (list of expected orders and last validated report), the control law and operative part models (named control filter model in this paper). Detection mechanisms based on rules and trajectory concept are performed on this input and are detailed in section 4. After this analysis, control filter can:

- Validate the order. The action is sent to the actuator and is applied on the physical system. The control filter transmits a list of expected reports to the other filter as well as the last checked order,
- Validate the order and raise an alarm. The actions are sent to the operative part as well as the information for the report filter but operators received an alarm indicating that something differs from normal behavior,

- Block the order. The action is not applied on the operative part and an alert is transmitted to report filter and operators. According to the system, a fallback mode can be activated.

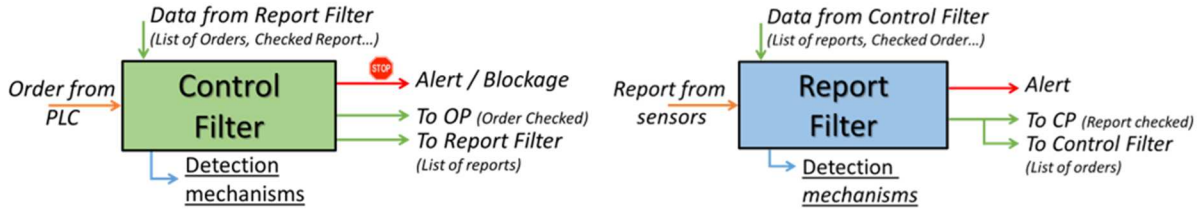

Fig 5. Illustration of filters inputs/outputs data

The report filter has the same behavior as the control filter except that it does not block data exchanged between operative and control parts. This filter analyses reports from sensors based on the information transmitted by the control filter (list of expected orders and last checked order) and process model. The analysis of reports is based on the same detection mechanisms implemented in control filter. After this analysis, reports are always transmitted to the PLC in order to guaranty the robustness of our approach. Indeed, a "wrong" report, that does not correspond to the one expected, may be due to sensor or actuator failure and may be supported and compensated by the PLC. Moreover, if this report leads to a critical state by inducing a wrong order, the control filter will stop it. Thus, the report filter sends data to the PLC and the list of expected orders and the last report checked to control filter. An alert can be raised if detection algorithm notices an anomaly.

Thus, the proposed approach is based on an IDS and filter approach for taking advantages of both according to the concept of mutual reinforcement [17]. Indeed, filters analyze information exchanged by operative and control parts based on models ensuring the different properties of the system as the correct execution of the control law or the respect of constraints of the physical system. An ICS can be characterized at each time by a unique state *s*. The different types of state sets are detailed in section 3.1. and presented in Fig 6. The notion of distance developed in section 4 aims to quantify the proximity of current state with prohibited and optimal states. The evolution of distance among time or state evolution allows defining the trajectory, which detects deviations of system and prevents *sequential attacks* and failures. These concepts are developed in next sections.

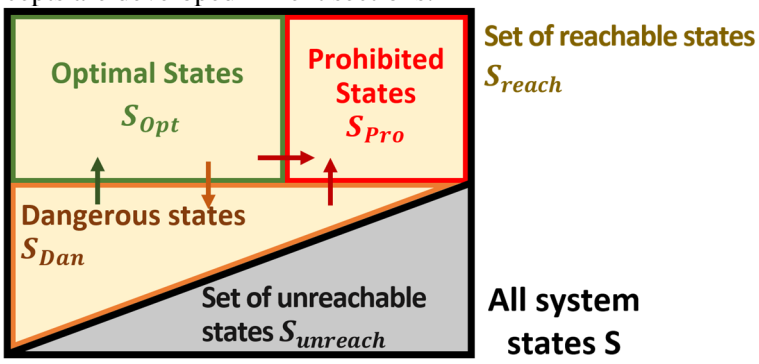

Fig 6. Illustration of the different types of states identified in an ICS

In next sections, we generalize activities monitored by filters by using events. Thus, when the control filter is considered, an event is an action that has to be analyzed by the filter based on the state of the system and the control model. On the opposite, for the report filter, an event is a report sent by the sensors that has to be compared to the expected ones and action applied on the operative part.

3. Methodology: from system modeling to filters synthesis

Previous sections highlighted vulnerabilities in ICS and notions that can be used to detect attack. The proposed approach is based on models that represent the normal behavior of the system in order to detect malicious orders sent by the PLC. However, the perimeter of this system has to be determined as

well as detection objectives for intrusion algorithms. In this section, the methodology of system modeling and filter conception is explained. Different detection objectives are presented in function of flexibility / detection ratio that is requested. The methodology of filter conception is defined in [5,14] and is composed of 3 main steps as detailed in Fig 7: parameters identification, control filter model generation and operation mode where detection mechanisms are performed. Steps 1 and 2 are performed offline while the last one is performed online. All detection algorithms presented in section 4.2. are based on each step. The type of modeling in function of detection objectives is discussed in Section 3.4.

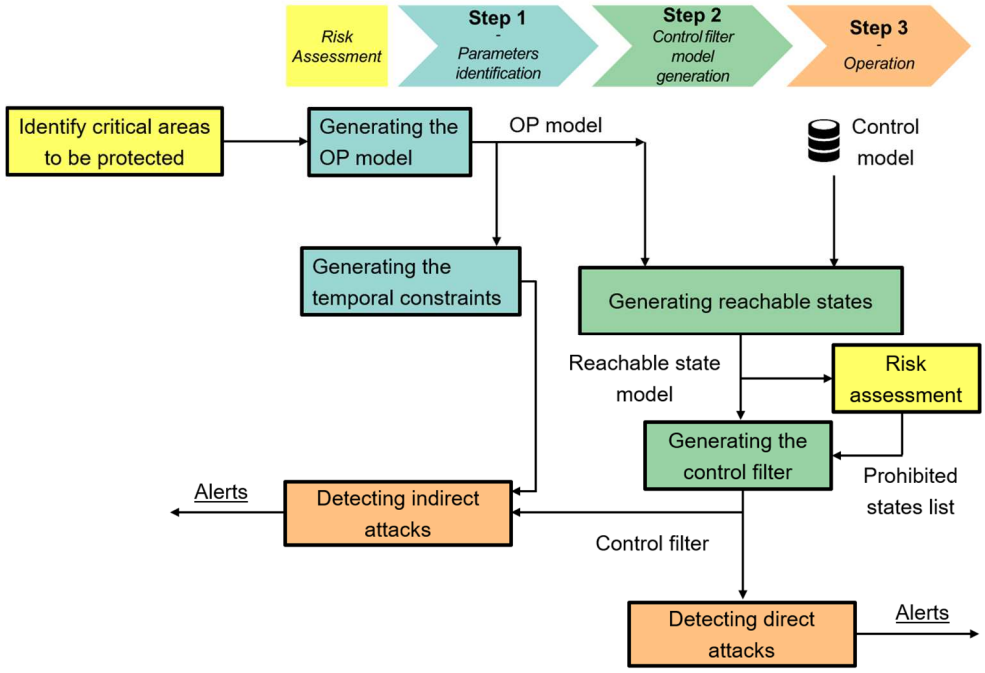

Fig 7. Methodology of filter conception for the proposed approach

Our approach assumes that a risk analysis has been previously carried out on the system. This analysis is therefore a prerequisite for the proposed methodology. More details on risk analyses for Scada systems can be found in the literature [49], indeed, several methodology can be used for analyzing an ICS by taking into account security or not [17,52]. This analysis should highlight the events feared for the system (prohibited states list) that induce parts of the ICS that must be protected as a priority (list of I/O that has to be modeled). Indeed, the S.A.F.E. approach is not intended to protect the entire system.

3.1. Step 1: Parameters identification

3.1.1. Combinational Constraints

This first step consists creating states based on the I/O list identified by the risk assessment. They correspond to the PLC's inputs/outputs and therefore define the perimeter of the system to be protected. The parameters selected in this first step then make it possible to define the states that are used in behavioral models. An expert characterizes the states belonging to the critical area identified by the risk analysis. Thus, based on Mitchell and Chen [56], the state $s \in S^i$ of an ICS, where i is the i^{th} iteration of the set of states S in the DES model, is characterized by a unique state defines, in equation 1, as the combination of the k-sensor and l-actuator values.

$$s \in S^i \subset S,\ s = [\text{Sensor}^1\ \text{Sensor}^2 \dots \text{Sensor}^k;\ \text{Actuator}^1\ \text{Actuator}^2 \dots \text{Actuator}^l] \quad (1)$$

Based on this notion of state, several types are distinguished and illustrated in Fig 6:

- Optimal states $s_{Opt} \in S_{Opt} \subset S$: these states respect both the constraints imposed by the control law and the physical system. Control part and operators always try to keep the ICS in this subset to guaranty an efficient production flow from initial to final state,

- Dangerous states $s_{Dan} \in S_{Dan} \subset S$: in this subset, only constraints imposed by the physical system are respected. Damages may be inflicted on the ICS but its severity is feeble. However, as the control law is not respected, the production flow is not optimal which may impact the quality of the final product, production time or wear of equipment,
- Prohibited states $s_{Pro} \in S_{Pro} \subset S$: these states endanger the ICS by strongly degrading the physical system. The main difference with dangerous state subset is in the damages inflicted because, for this two types of states, the control law is violated,
- Reachable states $S_{Reach} \subset S$: this set refers to the possible states that can be reached by the system using all the available actions A. This set includes previous subset S_{Opt}, S_{Dan} and S_{Pro}.
- Unreachable states $S_{Unreach} = S \setminus S_{Reach}$: the set of actions A does not lead to these states. For example, for a tank system, an unreachable state is the activation of the top-level sensor of the tank (total filling) while the bottom-level sensor is inactive (empty tank).

Finally, determining the right number of parameters to define a state becomes a critical issue. Indeed, if too many variables are considered, then modeling step faces exponential complexity issues. On the opposite, if a parameter is forgotten, then the models will not be correct which leads to errors, false detections and, at worst, non-detection of anomalies.

3.1.2. Temporal constraints

Temporal analysis is the basis for detection mechanisms to identify *temporal attacks*. These attacks respect the constraints related to the combinatory and sequentially of exchanges between the control and operative parts but modify their timing. During this type of attack, the correct sequence of reports corresponding to optimal sequence orders is sent to the control part from combinational constraints. A time constraint is a set of properties characterizing an event (action or report) over time in relation to a trigger, an activation (or deactivation) time and another event (delay, margin). Thus, if an action a_i has to be applied during a period t_a and an attacker sends a report of end of task after a period t_b with $t_b < t_a$, the quality of the system is damaged. The set of time constraints of a system defines time windows that specify the time behavior of the system for each possible action and report. Fig 8 illustrates the characteristic elements of these time windows and more information can be found in Sicard et al. [14].

Time detection has already been studied in cybersecurity but in different ways. Defensive approach has been developed by Taylor et al [57] on CAN bus. This network analysis detects *temporal attacks* that insert packets into the network. Algorithm developed in this approach computes inter-packet average time with historical in order to detect anomalies. The main limitation of this work is the use on system with periodic packets. Kleinmann and Wool [58] also take into account time in their approaches for building models or evaluate system security. The authors propose to model the traffic of the ICS between the HMI and the ICS. This approach uses the knowledge of reachable states of the PLC after a request or a response. The algorithm has been tested on Siemens S7-0x72 protocol with good accuracy and low false positive results but needs long leaning phase as method based on cyclic pattern recognition. Mohan et al [59] developed an approach based on characteristic time execution for real-time system. Thus, for each task that has to be done, a typical execution time is defined. Detection occurs when the current task activation period is too large or small by comparison with the reference time. In order to improve the robustness of the algorithm, average, worst-case and best-case execution times are computed. To conclude, another important work using time detection is Zimmer et al [60] that also analyzes task execution time. The objective is to develop mechanisms that detect code injection. Thus, synchronous and asynchronous calls are used to compare code time execution. The main limit of this approach is the compatibility with real time constraints.

To perform this detection mechanism, a characteristic execution time t_A is assigned for each action $a \in A$ as well as report reception time t_E. For considering temporal uncertainties of ICS, Δt is defined to build a temporal window. Δt_{min} and Δt_{max} represent the gap between average and respectively min/max execution time. Thus, for the control filter, the time window TW_{Order} characterizing when and how long an action has to be executed is described in equation 2.

$$TW_{Order} = [t_A - \Delta t^A_{min}; t_A + \Delta t^A_{max}] \quad (2)$$

Similarly, the time window TW_{Report} specifying the reports of an ICS is written as in equation 3 and allows defining the time and duration of occurrence.

$$TW_{Report} = [t_E - \Delta t^E_{min}; t_E + \Delta t^E_{max}] \quad (3)$$

T^{moy} represents the average execution time that is the evolution of t_A and t_E among time. On the contrary, Δt_{min} and Δt_{max} are constant. Finally, delay τ is used to compute the gap between the end of event (order or report) and the beginning of another. Fig 8 represents the expression of temporal constraints that can be defined for adding another layer of protection, complementary with combinational constraints, for ICS. To conclude, a temporal window, that is critical for the process and may damage the process, has to be defined during the modeling step.

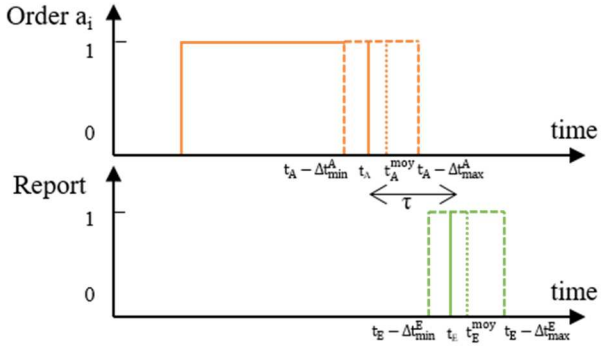

Fig 8. Illustration of temporal constraints needed for filter based detection

This step of temporal specifications is essential to perform the temporal detection algorithm presented in section 4. This mechanism adds a dimension in the characterization of the system's behavior and also allows adding another level of protection against an attacker for the ICS.

3.2. Step 2: Control filter model generation

The aim of this step is to model the industrial system with process and eventually control model (this issue is discussed in section 3.4.). These models represent the normal behavior of the ICS. For the control model, Petri nets are used for modeling the control law programmed inside the PLC. Thus, this graphical and mathematical writing allows representing constraints imposed by the control part on the system as sequential properties, parallelism, mutual exclusion and synchronization [14]. In this model, places represent actions that are applied on the system and transitions are reports received when these orders have been correctly applied. Furthermore, we assumed that the model is based on a correct PLC program as Ladder or SFC. Afterwards, the execution of the control model in a safe environment (from IT point of view) allows obtaining optimal states of the system (expected ones). An illustration of a control model is available in Fig 9.

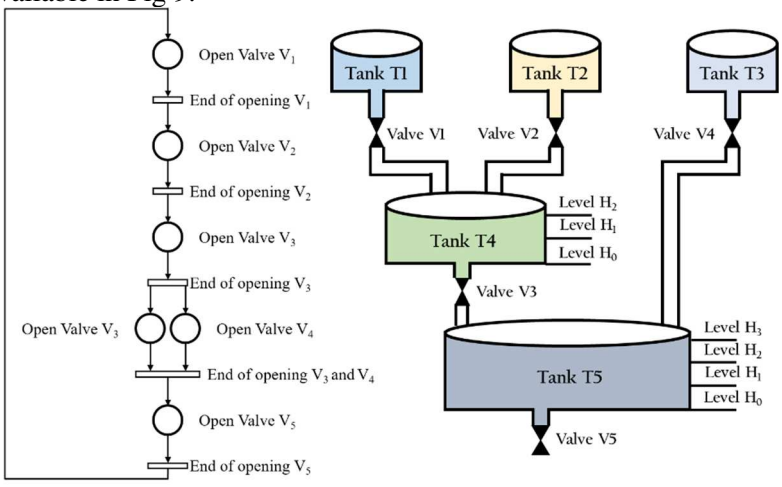

Fig 9. Illustration of the control law of the system presented in Sicard et al [14] with sequential and parallelism properties

Concerning the model of the operative part, only a limited number of actions can be applied on the system (number of actuators), so that, a finite number of states can be reached. In this paper, only steady

states are modeled. Transitional states are not necessary to obtain correct results on models and detection mechanisms and this decreases the complexity of modeling. Thus, a finite deterministic automaton, called M and represented in equation 4, is used.

$$M = \{S_n, A, \delta, S_0, S_{final}\} \text{ with } S_{final} = \{S_0, S_{Pro}, S_{Prev}\} \quad (4)$$

- S_n is the finite set of reachable states for the system, also called S_{Reach}, where n indicates the number of state. Vector s_i representing the state of the system can be in the subset of optimal states S_{Opt}, dangerous states S_{Dan} or prohibited states S_{Pro},
- A is the finite set of orders that can be performed by the system.
- δ is the transition function that represents relation between states and orders. These relationships will change the automaton by simulating the evolution of the system. Thus, δ computes the next state $s' \in S$ from current state $s \in S$ and the applied action $a \in A$. To do that, each action has to be modeled by its effect on the process (see paragraph below),
- $S_0 \subset S_{Reach}$ is the set of initial states of the system. It is the starting point of the performed algorithms,
- S_{final} represents the finite state set of stopping conditions of the algorithm. When one of these final states is reached, evolution in the automaton is stopped (initial state s_0, prohibited state $s_{Pro} \in S_{Pro}$ or state already meet $s \in S_{Prev}$).

By construction, this automaton M guarantees the condition of reachability of computed states in the process model. Indeed, the algorithm applies possible action to the system starting from the initial until a final state is reached (initial state, already reached state or prohibited state). A preliminary work is to identify the effect of each action on the system in order to define the transition function δ. Thus, with a state $s \in S^i$ and an action $a \in A$, the function δ is able to give the state $s' \in S^{i+1}$ reached by the system. Works of Henry [61] are used for defining the effect of each action on the operative part. Equation 5 represents the effect of the transition function δ and equation 6 the generalization to a set of states with the transition function on sets Δ:

$$\forall s \in S^i \subset S_n, \forall a \in A, \text{ such that } s' = \delta(s, a) \text{ with } s' \in S^{i+1} \subset S_n \quad (5)$$

$$S^{i+1} = \Delta(S^i, A) = \{ s' = \delta(s, a) | s \in S^i, a \in A \} \subset S_n \quad (6)$$

With the risk assessment presented in previous part that identifies critical state, an algorithm that detects contexts leading to prohibited states is developed. A context is a pair state/action that results in a forbidden state s_{Pro} if the action is applied on the system [14]. Equation 7 defines mathematically a context with the transition function δ:

$$\exists\, s \in S^i \subset S_n,\ \exists\, a \in A, \text{ such that } s' = \delta(s, a) \text{ with } s' \in S^{i+1} \subset S_{Pro} \quad (7)$$

Thus, the algorithm computes the projection of set of states in order to identify contexts leading to critical states. All possible actions have to be applied on each reachable state, starting from initial state set S_0, to find all possible contexts. In order to limit the exponential complexity issues, several stop conditions have to be set. Thus, the algorithm breaks the exploration of a branch when one of these conditions are met:

- Initial state set S_0: $S^{i+1} \cap S_0 \neq \{\emptyset\} \leftrightarrow \exists\, s' \in S^{i+1}$ such that $s' \in S_0$,
- Prohibited state set S_{Pro}: $s' \in S_{Pro} \subset S_{Reach}$. When this condition is met, the algorithm has identified a context,
- Previous state set S_{Prev}. When the algorithm computes a state already met, the exploration of this branch is stopped.

Thus, algorithm guaranties the exploration of all reachable possible states, by applying all combination of actions, and limits the exponential complexity issues with stop conditions. At the end of this exploration step, all the contexts of the system have been computed and by using the sequence of order contained in the control model, the set of optimal states can be computed. Thus, by knowing optimal and prohibited states and, with the assumption that all reachable states are explored, the three types of states (S_{Opt}, S_{Dan} and S_{Pro}) defined in previous step are obtained. Algorithm 1 details all these specifications as well as Fig 10.

Algorithm 1. Algorithm to explore all the possible reachable states of a system

Function Find possible reachable states
$S^i=S_0$
WHILE ($S^i \neq \{\emptyset\}$)
  $S^{i+1}=\Delta(S^i, A)$
  IF($S^{i+1} \cap S_{Pro} \neq \{\emptyset\}$) THEN
    $S^{i+1} \cap S_{Pro} = F$ #Set of states $s \in S^{i+1}$ such that $s \in S_{Pro}$
    FOR ($\exists s' \in S^{i+1}$, $s' \in S_{Pro}$)
      Context = {Context, F}
    ENDFOR
  ENDIF
  IF($S^{i+1} \cap S_0 \neq \{\emptyset\}$) THEN
    $S^{i+1} \cap S_0 = G$ #Set of states $s \in S^{i+1}$ such that $s \in S_0$
    $\exists s' \in S^{i+1}$, $s' \in S_0$
  ENDIF
  IF($S^{i+1} \cap S_{Prev} \neq \{\emptyset\}$) THEN
    $S^{i+1} \cap S_{Prev} = P$ #Set of states $s \in S^{i+1}$ such that $s \in S_{Prev}$
    FOR ($\exists s' \in S^{i+1}$ s.t. $s' \notin S_{Prev}$)
      $S_{Prev} = \{S_{Prev}, s'\}$
    ENDFOR
  ELSE
    $S_{Prev} = S_{Prev} \cap S^{i+1}$
  ENDIF
$S^i = S^{i+1} \setminus (F \cup G \cup P)$
DONE
End

Fig 10 illustrates algorithm 1 that explores all reachable states by applying all combination of actions on the system. The system is composed by three different actions a_1, a_i and a_m. When one of the three stop conditions is met (i.e. when the reached state $s \in S_{final}$), the exploration of the branch is broken. For example, in this example, we notice that:

- Action a_1 applied on state s_1' leads to the initial state $s \in S_0$,
- Action a_m applied on state s_3' leads to prohibited state S^2_{Pro}, the pair {s_3', a_m} is identified as a context,
- State s_3'' has been already met in previous iteration with state s_1': $s_3'' = \delta(s_3', a_i) = s_1'$.

The algorithm stops when the set of states that has to be explored is empty, here after four iterations. Thus, a unique sequence state/action (branch) is computed at every iteration. At the end of this step, the control filter model is obtained.

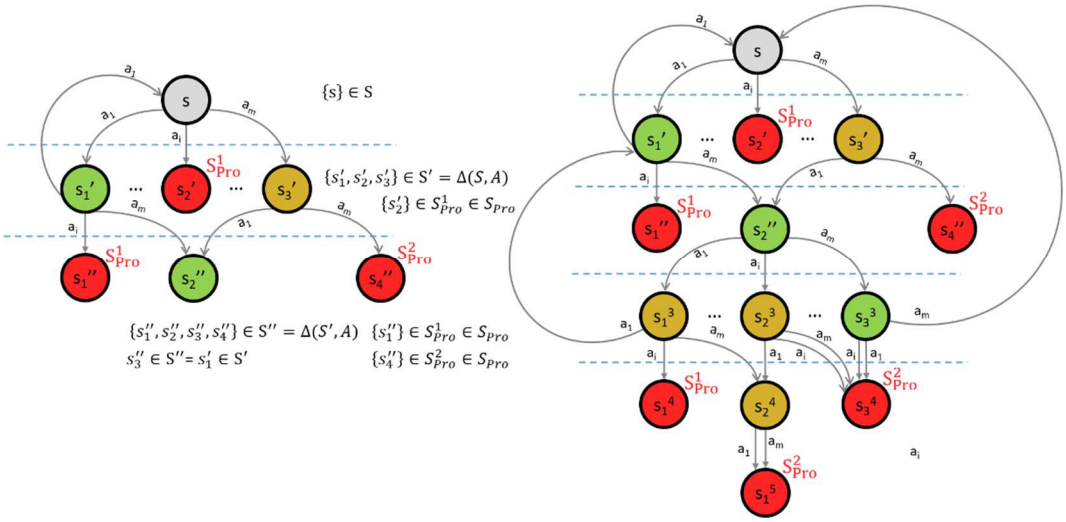

Fig 10. (left) Illustration of exploration states algorithm (Algorithm 1) with stop conditions (part of process model) – (right) Example of a complete control filter model (state s_4^4 is similar to state s_2^3 and is therefore not represented)

3.3. Step 3: Operation

This step corresponds to the online availability of control and reports filters in the ICS. The control part (control filter) and the operating part (CR filter) models support the detection mechanisms developed in this paper in section 4.

3.4. Location of Filters in the ICS architecture

As presented in section 2, filters insure the security of the system by their design. However, to be effective, the detection mechanisms have to be implemented in filters supposed non-attackable on the hardware layer. Moreover, the location of filters in the ICS architecture is a key element in detection of anomalies. In the implementation proposed by Sicard et al [5,14], the control filter is located the closest as possible from actuator. Thus, after this filter, there are only wires for linking the control part to the actuators. The location of the control filter allows to guaranty the integrity of orders after they are analyzed in order to detect malicious behavior. In the presented approach, the report filter is located just after the sensors. Thus, report filter analyzes data sent by sensors and transfers the information to the control part and the other filter. If data analyzed by report filter does not correspond to process model, then a fault can be detected. However, if data from sensors are correct but order sent by the control part violates the control law, the anomaly may be located in the PLC (fault or attack) or on the communication (attack). Several scenarios of attacks have been developed to test different cases in section 4. Fig 11 illustrates the location of these filters in the ICS architecture to detect anomalies and protect the system. To conclude, the control filter can be compared with an electrical switch that is able to isolate the operative part in the case of anomaly detection. The report filter is a trust anchor that verifies the correct execution of orders on the system. The efficiency and reliability of this approach are based on the location, the design and detection mechanism implanted inside the filters.

Finally, filters are based on combinational and temporal constraints to ensure the efficiency of detection algorithms. This aspect is developed in section 4.2. of this paper. In Fig 11, control filter is named $F_{con}(s,t)$ and report filter is defined $F_{rep}(s,t)$.

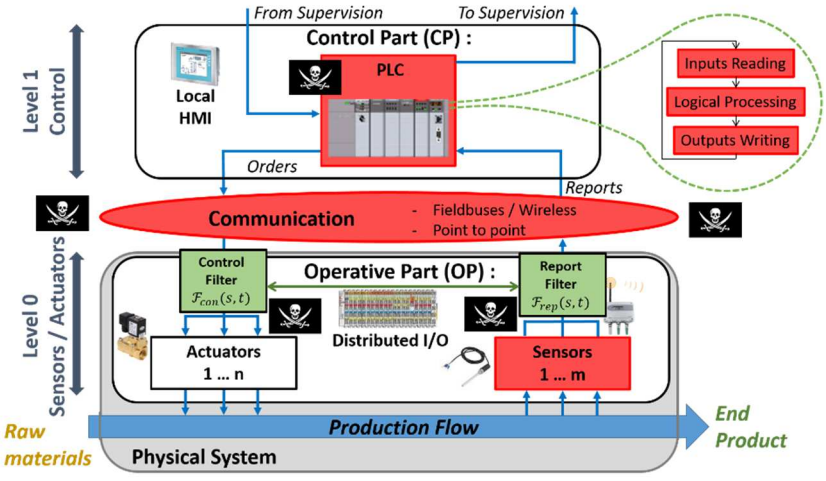

Fig 11. Illustration of filters location in the ICS architecture

3.5. Type of models: which model for which protection?

In the previous section, a methodology to synthetize filters has been presented; however, information needed to achieve the desired level of security should still be defined. For the first three layers of CIM architecture, we define three levels of security detailed bellow:

- Security of goods and people. Filters have to guaranty a set of states in which the system can evolve without danger. Thus, they analyze the information exchanged by control and operational parts in order to verify that the system does not reach a forbidden state. Only the process model is required and sufficient (level 0) for this level of security. The sequences of orders are not analyzed which implies that filters do not check the correct execution of the control law. Thus, this level of security is not able to compute optimal and dangerous states of the system so that concept of trajectory cannot be used. On the other hand, this protection needs to be changed only when the physical system is modified,
- Quality. This level of protection guaranties the correct execution of the control law in addition to monitoring the state evolution of the system. The path to evolve from the initial to the final state is imposed. The control and process models of the system have to be known to guaranty this level of protection (level 1 and 0). Thus, filters ensure that the system respects the constraints imposed by risk assessment (state evolution) and control part (sequence of orders). The concept of trajectory is used for this level of protection in order to detect anomalies in the sequence of orders sent to the system that may impact on the quality of the final product. For this protection, control model has to be modified every time that the control law is changed,
- Equipment protection. This level of protection monitors the solicitation of actuators to order that are too strong or too frequent. The objective is to prevent fatigue, maintenance or breakdowns on equipment. Thus, for each actuator, filters monitor the evolution of the solicitation $\frac{\sum \Delta u^2}{\sum \Delta y^2}$. A characteristic attack on equipment may be the opening and the closing of a valve or a cylinder in order to degrade this component without affecting directly the quality of the product or the security of goods and people.

As presented, each level of protection has benefit and inconvenient, especially the report knowledge/protection. To complete these levels of protection, two planes have to be distinguished. Control and process models can express combinational and temporal constraints. Thus, since the beginning of this section, only combinational constraints used for state evolution have been explained. The same methodology can be applied with temporal constraints. Thus, filters ensure networking to protect the system from cyberattack by using several types and planes of protections.

In this paper, filters insure the three levels of protection: security of goods and people, quality and equipment protection with combinational and temporal constraints. The next section presents the different detection mechanisms that use these models based on this three protection levels.

The following table summarizes the main notations used in this paper:

S	Set of possible states
S_{Reach}	Set of reachable states, $S_{Reach} \subset S$
$S_{Unreach}$	Set of unreachable states, $S_{Unreach} = S \setminus S_{Reach}$
S_{Opt}	Set of optimal states, $S_{Opt} \subset S_{Reach}$
S_{Pro}	Set of prohibited states, $S_{Pro} \subset S_{Reach}$
S_{Dan}	Set of dangerous states, $S_{Dan} = S_{Reach} \setminus (S_{Opt} \cap S_{Pro})$
S^i	Set of the i^{th} iteration states
s_i	i^{th} state of a state set
S_0	Set of initial states
S_{final}	Set of final states (automaton M)
S_{Prev}	Set of states already reached during previous iterations of automaton M
A	Set of possible actions
a_j	j^{th} action of a set of actions
δ	Transition function applied on states and actions
Δ	Transition function applied on sets of states and actions

4. Detection mechanisms

In this section, concepts of distance and trajectory are explained as security indicators used into detection mechanisms based on combinational constraints. Other types of detection algorithms, based on temporal constraints are also presented. Mechanisms illustrated in this section are applied on simulation examples.

4.1. Security indicators: distance notion and trajectory concept for ICS

As detailed in Sicard et al [14], the proposed approach, based on filters located between level 1 and 0, allows implementation of rules. These rules guaranty the normal behavior of the system by detecting any order that break them. Thus, these rules based approach can detect and stop an attack but only one-step before reaching a critical state. Based on this observation, our approach has to implement an algorithm that detects deviations from expected behavior. For this purpose, the bibliographical study has highlighted works of Carcano et al [40] and Fovino et al [39] that develop the notion of distance. However, this concept has been applied on continuous variables only. After a discussion of these works, an improvement of this notion of distance is presented. The concept of trajectory is explained in conclusion of this section.

4.1.1. Notion of distance

Carcano et al [40] develop an interesting notion of distance based on the ICS states. Indeed, these systems can be described by a unique combination of component values (PLC, actuators, sensors) defining a state. Thus, distance computes the gap between the current state of the system and a set of critical states, called critical formula Φ. This critical formula regroups all the prohibited states S^{Pro} of the system that ICS does not have to reach. These states are called the set of constraints c_i, the system has to respect them. Two types of distance (d_v and d_l) computation are presented in the paper and the distance d is defined as the minimum of them. Thus, equation 8 defines the distance between a state s and the set of constraint Φ that represents the conditions for entering a critical area and established by an expert (e.g., the motor permissible current ranges or the permissible temperatures for a process) as:

$$d: \Re n \times \Re n \rightarrow \Re +,\ s \in \Re n$$

$$d(s, \Phi) = \min^{j} d_{min}(s, C_j) \text{ with } d_{min} = d_1 = \Sigma^{n}_{i} |s_i - c_i| \text{ or } d_{min} = d_v = \#\{i | s_i \neq c_i\} \quad (8)$$

In the paper, distance d_l is defined as Manhattan distance [40] and counts the gap between the current state s and the constraints Φ components by components. Distance d_v computes the number of different components between the two vectors. The distance to critical states d is the minimum of distances d_l and d_v representing the gap between the current state s and the critical formula Φ.

This notion is interesting for our approach because distance gives an indication to operators about the proximity of a critical area. Moreover, by implementing control law and consequently the optimal states of the system, this notion of distance provides information about the production flow of the system. Thus, with this improvement, operators are able to monitor if the ICS follows the optimal trajectory where optimal actions are performed and avoiding the prohibited states.

In conclusion, this notion of distance, combined with models developed in section 3, provides relevant information to improve the detection with rules in [14]. However, this notion of distance is effective only for continuous variables as motor speed or temperature. Indeed, for discrete systems based on discrete sensors and actuators, distance information is useless. The system can directly go into dangerous or forbidden state when actions applied on the system differ from control law. An extension of this notion of distance to discrete values are proposed in next section.

4.1.2. Shortest Path to Critical State

Based on models of physical systems, the purpose of the following algorithm is to count the minimal number of orders from the current state that has to be applied before reaching a critical state. The smallest number of actions is called the shortest path to critical state. Thus, for each reachable state of the system, filter is able to compute the distance to critical state from a discrete point of view. Several prerequisites are necessary as the set of prohibited states that are defined with the risk assessment detailed in section

3.1. and the finite deterministic automaton representing the physical system that are explained in section 3.2. Thus, the general purpose of this section is to compute distances of each reachable state based on the process model obtained during the step 2 of the proposed methodology. These distances have to quantify the gap between a reachable state of the system and the nearest critical or optimal states. These computations are made off-line and then are implemented into filters in order to perform detection algorithm based on the trajectory tacking into account real-time constraints.

Thus, the minimal number of iterations n to reach a forbidden state starting from a state $s \in S^i \subset S_n$ is defined as the distance D in equation 9:

$$D(s|S_{Pro}) = \min_n \Delta^n(S^i, a) \in S_{Pro} \; \forall s \in S^i \subset S_n, \; \forall a \in A \tag{9}$$

Algorithm 2 provides more information for computing this distance to critical state. Thus, filters are able to compute more faithfully indicator to operators about current state and trajectory of the system.

Algorithm 2. Algorithm to find the nearest prohibited state from state $s \in S^i \subset S_n$ (shortest possible way)

Function Find nearest Prohibited State
Iteration = 0
WHILE (Stop ≠ 0)
  Iteration = Iteration +1
  S' = Δ(S_i, A)
  IF (S'∩S_{Prev} ≠ {Ø}) THEN
    P = {s∈S_{Prev}⊂S'} ∀s∈S'⊂S_n s.t. s∈S_{Prev}
    S' = S'\P
    S_{Prev} = S_{Prev} ∩ S'
  ELSE
    S_{Prev} = S_{Prev} ∩ S'
  ENDIF
  IF (S'∩S_{Pro} ≠ {Ø}) THEN
    F= {s∈S_{Pro}⊂S'} ∀s∈S'⊂S_n s.t. s∈S_{Pro}
    S' = S'\F
    Stop = 0
  ENDIF
S_i = S'
DONE
End

This algorithm is performed on every reachable state identified in the system during the second step, detailed in section 3.2. Thus, the shortest path to critical state is computed which means that the sequence order/state leading to a prohibited state is known as well as the distance $D(s|S_{Pro})$. This notion of distance for discrete variable is explained in Fig 12 below.

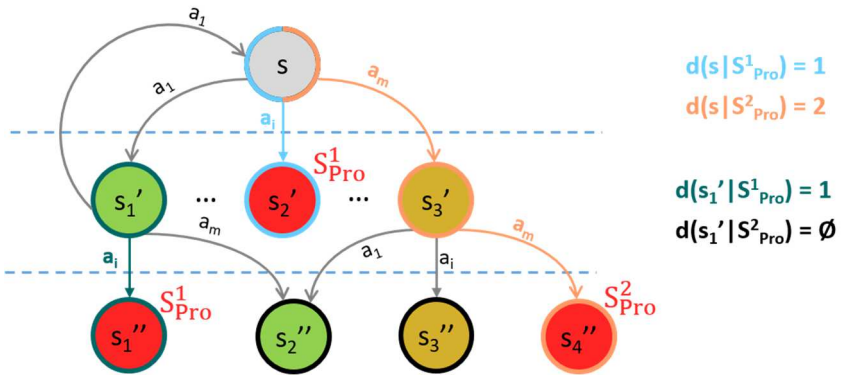

Fig 12. Illustration of distance notion for discrete variables

Fig 12 illustrates state evolutions of a system depending of applied actions. The initial state is called s and three different types of actions can be done. Moreover, two prohibited states have be identified for this system. Algorithm 2 computes the distance of state s to each forbidden state (S^1_{Pro} and S^2_{Pro}). Thus, the state s is at a distance of 1 from critical state S^1_{Pro} and 2 from prohibited state S^2_{Pro}. Indeed, when action a_i is applied on state s then the forbidden state S^1_{Pro} is reached and when actions a_m and a_m are

applied on state s then the prohibited state S^2_{Pro} is reached. To conclude, algorithm 2 computes the distance between each reachable state and each prohibited state of the system.

Depending on the type of models used for representing the system (discussed in section 3.4.), an extension to algorithm 2 presented before can be done. Indeed, if the control model representing the control law is implemented, other distance can be computed between:

- Applied action a_i and the expected order a_{Opt} (obtained with the control model), $D_{order}(a_i|a_{Opt})$,
- A state s and the optimal state s_{Opt} expected, if the control law of the system is respected, $D_{state}(s|s_{Opt})$,
- A state s and nearest prohibited state s_{Pro}, $D_{state}(s|s_{Pro})$.

In this approach, we will not focus on the value of orders (continuous or discrete components) but on the logical expression sent by the PLC. Thus, notion of distance developed by Carcano is also improved on this point because distance monitors activities sent by OP and CP without focusing on the values communicated. The distance quantifies the difference between the current state and each type of nearest prohibited state. Thus, by construction with algorithm 2, it is possible to know for each state of the system, the risk associated with this state.

4.1.3. Trajectory concept in ICS

As presented in the previous section, distance notion gives only a punctual information on the system. The study of sequences is more interesting for detecting anomalies. The concept of trajectory is defined as the evolution of distance according to state sequence or time. In this first case, distance is computed and detection algorithms performed for each steady state reached by the system. In the other case, temporal execution of the system is monitored and the evolution has to check these for respecting detection mechanisms. Thus, evolution is the behavior of the ICS during the chosen measurement scale. Equation 10 represents this concept from mathematical point of view:

$$T(s) = \partial d(s,t)/\partial s \text{ and } T(t) = \partial d(s,t)/\partial t \quad (10)$$

Trajectory represents the evolution of a system on a sequence whereas distance only describes the system at one point. Trajectory gives this information for ICS with evolution of state and time. Based on equation 10 and similarly to distance, several trajectories can be drawn:

- $T_{order}(a_i|A_{Opt}) = \partial d_{order}(a_i|A_{Opt})/\partial S$ and $T_{order}(t) = \partial d_{order}(a_i|A_{Opt})/\partial t$ check if actions sent to actuators correspond to the control model (optimal orders),
- $T_{state}(s|S_{Opt}) = \partial d_{state}(s|S_{Opt})/\partial S$ and $T_{state}(t) = \partial d_{state}(s|S_{Opt})/\partial t$ compute evolution of new states with optimal trajectory respectively based on state and temporal evolutions,
- $T_{state}(s|S_{Pro}) = \partial d_{state}(s|S_{Pro})/\partial S$ and $T_{state}(t) = \partial d_{state}(s|S_{Pro})/\partial t$ verify evolution of new states according to prohibited states. This trajectory has to be computed for each critical state.

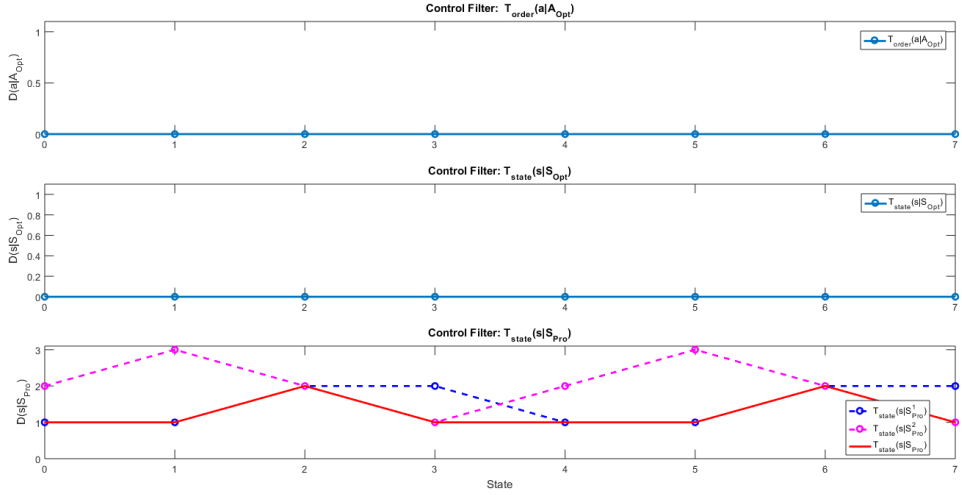

Fig 13. Illustration of different types of trajectories according to state evolution (top: $T_{order}(a_i|A_{Opt})$, middle: $T_{state}(s|S_{Opt})$, bottom: $T_{state}(s|S_{Pro})$) – Illustration of the trajectory of a system respecting the control law (control filter)

Trajectories always represent the worst scenario for the system, meaning the worst sequence of possible actions to reach a prohibited state in the sense that algorithm selects the shortest path to critical

state for discrete variables or the smallest distance to constraints for continuous variables. Thus, for $T_{state}(s|S_{Pro})$, distance to each critical state has to be computed as well as the global distance of the system. Fig 13 illustrates the concept of trajectory based on the state evolution.

The trajectories representing the evolution of the difference between the current state and the optimal state T_{order} $(a|A_{Opt})$ (top on Fig 13) and between the order to be analyzed and the expected order T_{state} $(s|S_{Opt})$ (middle on Fig 13) are equal to zero on the sequence considered in Fig 13. Indeed, during this sequence, the control law is applied to the system without the occurrence of failures. T_{state} $(s|S_{Pro})$ is shown at the bottom of Fig 13. The dotted curves illustrate the evolution of the distance from each of the prohibited states of the system (S^1_{Pro} and S^2_{Pro}). The solid curve is the critical path taking the shortest path to a critical state and whose distance is the smallest $D(s \mid S_{Pro})=\min (d(s \mid S^1_{Pro}),d(s \mid S^2_{Pro}))$. Thus, based on the concept of distance, which is defined as the minimum number of actions to be applied to the current state to reach a prohibited state, the trajectory T_{state} $(s|S_{Pro})$ always represents the worst possible scenario for the system. Detection mechanisms based on the trajectory detect anomalies sooner than distance alone or rule based model. Thus, operators have the possibility to react and correct them. An anomaly is defined as a deviation from the reference behavior based on the control law execution and which is caused, in this work, by a failure or an attack.

4.2. Detection mechanisms

In this section, detection mechanisms are presented and illustrated with examples.

4.2.1. Context detection: immediate blockage

This mechanism is based on contexts identified during algorithm 1 of exploration state step. When the control filter detects a pair state/action leading to a critical state, then the order is stopped. This security mechanism can be compared to signature approach based on a process model and is only used when other mechanisms have failed the detection. Context detection is defined with a rule *R* described in equation 11:

$$R = \text{True} \leftrightarrow \exists\, s' \in S^{i+1},\ \exists\, s \in S^{i},\ \exists\, a_m \in A \text{ such that } s' = \delta(s, a_m) \in S_{Pro} \quad (11)$$

This mechanism can be compared to an emergency braking that secures the ICS against *direct attacks*. However, the blockage of an order in such system is disadvantageous (production flow is interrupted). In order to illustrate this last resort mechanism, example presented in Fig 10 is used. In such system, the control filter blocks the order:

- a_1 when the system is in the state s or s_1' (forbidden state S^1_{Pro}),
- a_m when the system is in the state s_3' (forbidden state S^2_{Pro}).

The rules-based mechanism presented in this paragraph is a so-called "static" security measure [14] that only protects the system from immediate danger by blocking the analyzed order.

4.2.2. Anomaly detection by combinational constraints

This detection mechanism is based on the distance and trajectory indicators constructed in the previous section. They are obtained by using behavioral models of the protected industrial system, detailed in algorithm 1. Thus, due to their construction, distance and trajectory indicators are the guarantors of compliance with combinatorial constraints. These concepts have been defined as follows:

- The distance represents the minimum number of actions to be applied from the current state to reach a prohibited state.
- The trajectory is the evolution of the distance during a sequence of states or time.

Using these indicators, the filters are able to detect deviations when the trajectory deviates from the optimal states to a prohibited state. Thus, distance provides information on the level of risk, represented by distance. However, the trajectory is based on the distance that calculates for each state only the shortest path to a critical state (detailed in equation (9) and algorithm 2). Thus, the system's trajectory towards prohibited states must be correlated with the distance to optimal states and orders. Similarly, order and reporting filters exchange information and work together to monitor system development, consistency of information exchanged and relevance of actions taken. Thus, the detection mechanisms embedded in the filters correlate the information from both filters to identify deviations.

Fig 13 (control filter) and Fig 14 (report filter) illustrates the case of a system, presented in Fig 10, following its optimal trajectory with the control filter point of view. Thus, when the system approaches a prohibited state in compliance with the control law, the indicators reflect normal system behavior. Indeed, the trajectories T_{Order} (a|A_{Opt}) (for the control filter) and T_{state} (s|S_{Opt}) (for the control and CR filters) are null, which implies that the analyzed order corresponds to the order expected by the control law model and that the state that will be reached is similar to the optimal state. Under these conditions, the comparison with prohibited states is normal and taken into account in the control law. This reasoning is supported by trajectories from report filter: T_{state} (s|S_{Opt}) (zero, indicating an optimal trajectory) and T_{state} (s|S_{Pro}) (similar to the control filter). However, it is still possible for an attacker to launch a *direct attack* on the ICS that will trigger the context detection mechanism that will block the action before it is transmitted to the OP. In this paragraph, we are only interested in sequence attacks.

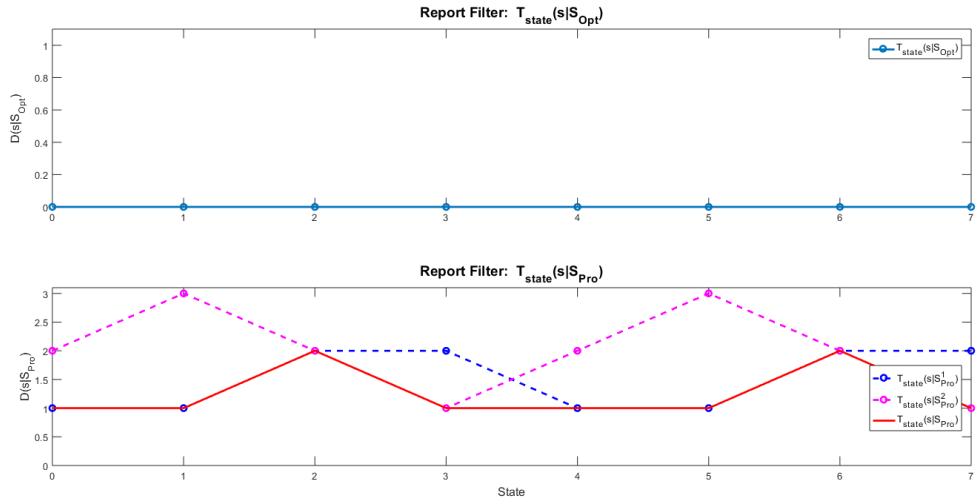

Fig 14. Illustration of the trajectory of a system respecting the control law (report filter)

In this situation, the case of sending an abnormal order that is not induced by a corrupted report is considered. The action was modified via the layer 2, by infection of the PLC or via the communication network (MITM attack between the PLC and the control filter). Thus, two consequences must be considered depending on whether the order leads the system into a prohibited or dangerous state. In the first case, the context detection mechanism will take over, while in the second case, an alert is sent for operators reporting the deviation from normal behavior without stopping the analyzed order. The analysis is carried out based on the example shown in Fig 10.

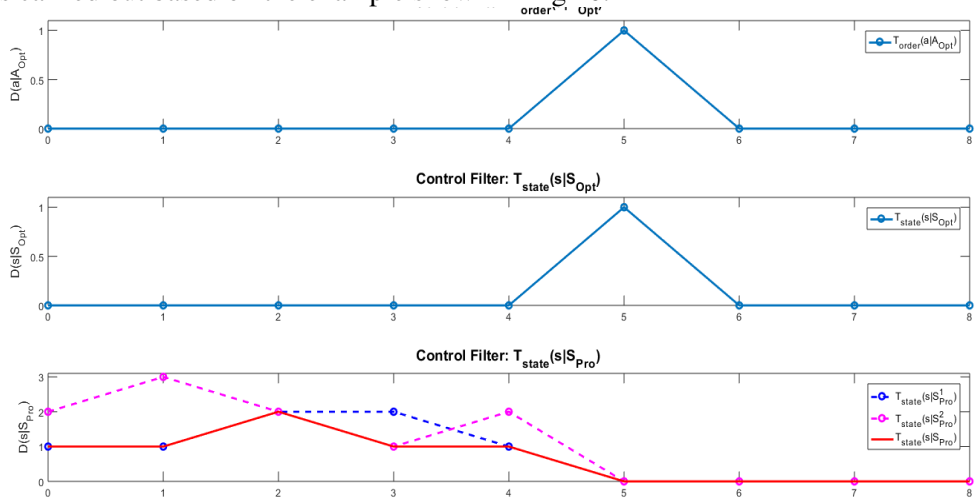

Fig 15. Trajectories of a system with an order leading to a prohibited state (control filter)

Fig 15 shows the case of an order leading to a prohibited state for the control filter. In this figure, unlike the previous case (optimal trajectory), the system does not respect the optimal trajectory when switching to state 5. The a_i order has been sent to the system instead of the order a_1. The state reaches

s'$_2$ and therefore does not correspond to the expected state s'$_1$. In addition, on the trajectory $T_{state}(s|S_{Pro})$, the distance to the prohibited states is zero, which is characteristic of a context since the state reached is also a prohibited state $s'_2 \in S^1_{Pro}$. The context detection mechanism is then triggered to block the analyzed order.

Fig 16 (control filter) illustrates the case where the analyzed order leads the system into a dangerous state. This situation does not result in a system blockage because the trajectory $T_{state}(s|S_{Pro})$ is never equal to zero, which reflects the fact that no context is encountered. Thus, the system moves from the optimal trajectory to reach a dangerous state via an order that does not respect the control law before returning to optimal states. This deduction is made from the analysis of the trajectories $T_{state}(s \mid S_{Opt})$ and $T_{Order}(a|A_{Opt})$. Indeed, the change from $T_{state}(s \mid S_{Opt})$ to one indicates that the state s'$_3$ is reached instead of the state s'$_1$. Similarly, the order a_m was sent to the system (instead of a_1), explaining this deviation. Sending the order a_1 also does not respect the command law but allows the system to regain the optimal optimal, signified by the values equal to zero on $T_{state}(s \mid S_{Opt})$. In the examples presented above, the results provided by the reporting filter are not predominant in the analysis feeding the detection mechanism. The CR filter validates that the system is moving away from the optimal trajectory and thus reinforces detection around an abnormal order.

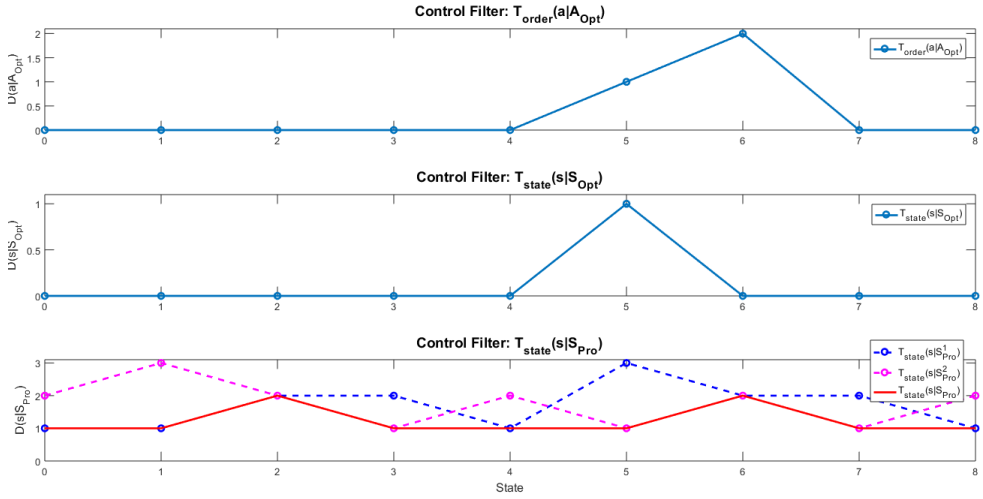

Fig 16. Trajectories of a system with an order leading to a dangerous state (control filter)

Previously, examples have highlighted different scenarios where the trajectory concept is used to detect punctual behavioral anomalies and warn operators. However, to facilitate understanding, the examples referred to simple dangerous and prohibited states. In the example below, the situation gradually deteriorates from the optimal trajectory to a prohibited state $s''_2 \in S_{Opt} \rightarrow s^3_2 \in S_{Dan} \rightarrow s^4_2 \in S_{Dan} \rightarrow s^5_1 \in S_{Pro}$). Here, the concept of trajectory makes a major contribution by allowing operators to be informed of this deviation. They can thus act to correct the trend taken by the ICS before arriving in a blocking situation (regardless of the origin of the anomaly).

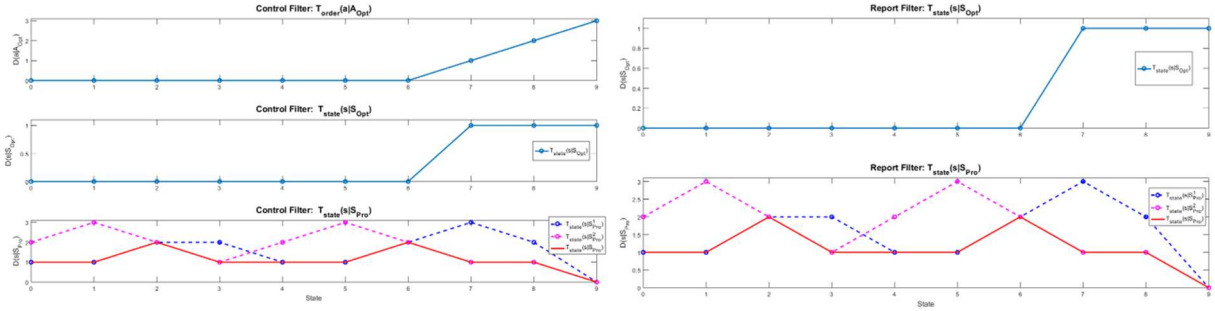

Fig 17. Trajectories of a system with a sequence leading to a prohibited state (Left: control filter, Right: report filter)

Fig 17 (left) shows the behavior from the control filter point of view (inspired from Fig 10) through the trajectories $T_{Order}(a|A_{Opt})$, $T_{state}(s \mid S_{Opt})$ and $T_{state}(s|S_{Pro})$. The control sequence deviates from state n°7 to a prohibited state, this corresponds to the application of the order a_i instead of a_m. Thereafter,

orders a_1 and a_m are sent to the operative part to reach the state $s_1^5 \in S_{Pro}^1 \subset S_{Pro}$. Fig 17 (right) confirms this deviation from the optimal trajectory through the report filter. However, we note that the PLC does not send an order to correct this deviation, suggesting that the control part and/or the communication network have been corrupted. The trajectory concept, based on taking information from filters in the ICS architecture, allows operators to be informed of this deviation before reaching a critical state. The detection of abnormal behavioral signals is done by studying the sequences and long-term evolution of exchanges between CP and OP.

To conclude this example, the control filter blocks the action a_m sent to the OP in state n°9 as it leads to a forbidden state. However, the trajectory concept allows operators to be informed about the state of the system, to anticipate deviation and to detect at the earliest possible stage an abnormal behavior that could be the effect of an attack. All the mechanisms presented in this paragraph use the combinatorial constraints defined in step 1 of the design methodology.

4.2.3. Anomaly detection by temporal constraints

This detection mechanism ensures that the time constraints specified in step 1 of the filter design methodology are respected. This mechanism is executed in parallel with the verification of combinatorial constraints. Thus, detection mechanism monitors the temporal execution of the system in order to detect anomalies in the duration of an order, or a sensor value or sequence of events. If the broken temporal constraint is critical for the process, then filters act as they would with combinational constraints (stop order and/or send alert). If temporal detection is not defined as critical, an alert is sent that is analyzed by combinational algorithm. Thus, when one or several temporal constraints are broken too long then one or several combinational constraints are also broken and mechanisms explained previously are triggered. This mechanism adds another dimension of protection for the system.

Fig 18 illustrates temporal constraints imposed by the TW_{Order} on the system presented in Fig 10. When the actions sent to the system do not respect these constraints then an alert is sent. Fig 19 illustrates a *temporal attack* that affects the quality of the product. An order of filling is sent to the process that leads to the activation of sensor H_0^{T4} (low-level sensor) in the correct time window. In the same time, attacker is sending a report of activation of sensor H_1^{T4} (intermediate level sensor) to the PLC. This attack respects the combinational constraints but damages the quality of the product (timing is not respected). Temporal constraints detect this anomaly and give information for post-detection analysis.

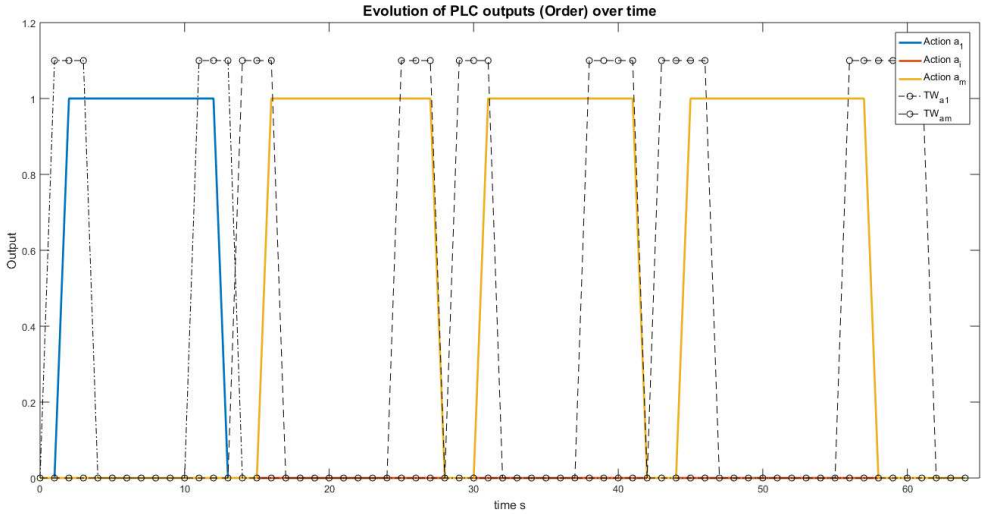

Fig 18. Illustration of temporal constraints on the actions send to the system

The second example illustrates a *temporal attack* that affects the security of goods and people. In the tank example, after a filing order a_1 is sent to the system, attacker modifies the report from sensors. The integrity of data is degraded by inhibiting activation of level sensors. Thus, PLC always send an order of opening valves without being aware of the tank filing. Temporal detection allows to stop filling order when $t > t_{a1}+\Delta t^{a1}{}_{max}$ and when correct report is not received. The limit of this mechanism is the choice of the parameter values for defining temporal constraints which depends on risk assessment made by the expert.

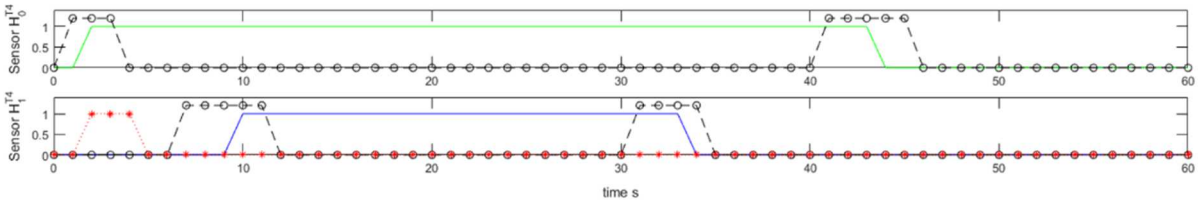

Fig 19. Illustration of *temporal attack* in the tank example presented in Sicard et al [14] (solid line: sensor value, black dashed line: time windows, red dashed line: attack)

4.2.4. Equipment degradation detection

The purpose of this detection mechanism is to identify order sequences that comply with the combinatorial constraints specified for the system but whose application leads to over-solliciting the actuators. The consequences then range from premature wear and tear, which leads to preventive maintenance on the equipment, to the breakage of the equipment. This detection mechanism is based on the third level of security presented in previous section. Filters monitor and analyze the response of operative part with command from control part. Thus, an over solicitation may introduce maintenance issue and in the end, damages on equipment. This mechanism provides information for operators on operative part with long-term perspectives.

To illustrate this protection, tank example illustrated in [14] can be used again. From combinational point of view, the system can follow the control law, and so that the optimal state, without respecting the correct use of actuators from temporal point of view. Indeed, the order of filling the tank until a level sensor may be correct but the PLC may order this filing by opening and closing quickly the valve until the level sensor is reached as illustrated in Fig 20 inspired by system represented in Fig 10. This mechanism has been developed to prevent this kind of anomalies by alerting operators and is also complementary to temporal detection mechanism.

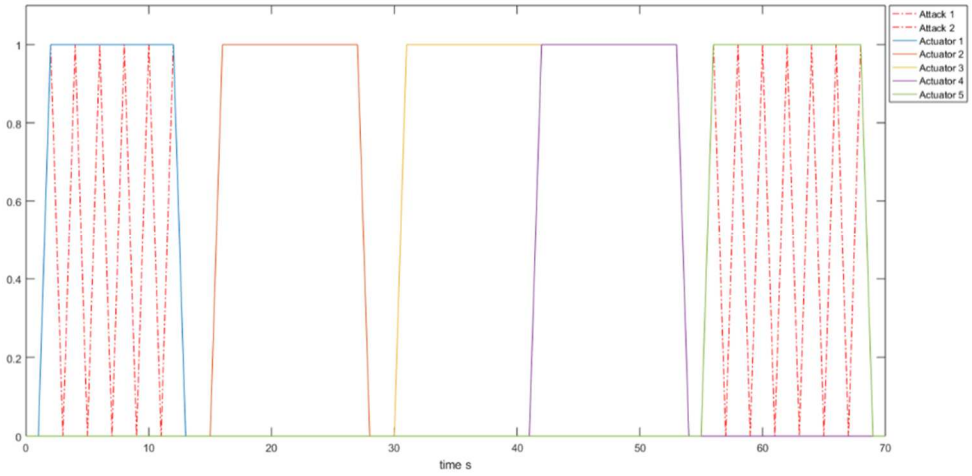

Fig 20. Study of over-soliciting on a system with 5 actuators

5. Experimentation and results

5.1.Generality

As explained in previous sections of this paper, several simulation examples have been used for testing the efficiency of detection mechanisms as in Sicard et al [4,14]. Thus, wagon/switch and 3 tanks systems, with few I/O, allow to detect *direct attacks* while the 6 tanks system, with significant numbers of I/O, allows to validate the detection mechanisms developed within the framework of *indirect attacks* (by *sequence*, *time* and *over-soliciting*). Finally, an example of an industrial platform validates the

applicability of the S.A.F.E. approach to a "real" industrial system [5]. In this section, the methodology is applied on the wagon/train system.

The combinatorial explosion is contained by the approach location between level 1 and 0. This originality, as close as possible to the operational part, provides protection adapted to ICS and above all minimizes complexity. Indeed, when the level 2 (supervision) manages a large quantity (thousands) of variables, the control part (level 1) only controls a maximum of about a hundred, and more traditionally about thirty. This placement is essential to ensure the convergence of the algorithms and detection mechanisms that will be proposed. This convergence is ensures by stop conditions that allow the system to explore possible states that can only be reached once. The complexity of the generated state space varies according to the values taken by the sensors (y) and actuators (u) and their numbers, respectively k and l. Thus, it will be equal to $y^{k}*u^{l}$. Within the framework of this work, DES are used as well as all-or-nothing actions, which results in a complexity of 2^{k+l}.

This complexity can be high since a system composed of 20 sensors and 10 actuators (average number of sensors and actuators connected to PLC in an ICS) will have more than 1 billion states that can be achieved. However, as shown in Table 1, the transition function δ reduces the number of states that can actually be reached and therefore the generated state space. The size of the achievable number of states depends on the physical system under consideration. However, this approach requires a methodology to model automation knowledge, and thus generate models of the control and operational parts. In addition, limits inherent to the placement of the approach appear:

- The execution time of the mechanisms will be related to the layers considered and therefore close to real time. To deal with these real-time constraints, two levels of detection have been presented in this paper: (i) the context detection that block an action leading to a prohibited state (immediate detection), and (ii) detection mechanisms based on distance/trajectory concepts and temporal constraints that act on long sequences (acceptable time to detect). To conclude, the modeling step (step 1 and 2 of the methodology) is computed off-line which does not impact online detection performance.
- The lack of a global vision related to the localization of the approach. As explained above, a PLC cell (level 1) has access to fewer variables than a global supervision (level 2) since it only manages a subset of the physical system. This hypothesis allows detection of abnormal behavior (for one PLC only) but cannot identify the source of the anomaly.

This approach is obviously complementary to IT solutions both in terms of location and detection mechanisms implemented (network analysis / automation knowledge) [4]. Finally, detection mechanisms against cyberattacks have been explained in section 4 and are the same for all the systems in terms of detection rate and limits.

5.2.Application of the S.A.F.E. approach to the Train/Switch system

The system, illustrated in Fig 21, represents a wagon travelling on a railway network equipped with a switch. The wagon connects 3 points A, B and C and can move forward, backward or brake. The switch can be controlled with two orders (high and low) and has three states (up, down and intermediate). Position sensors give the location of the wagon. At the initial state, the wagon starts from point A to reach the point B (order of moving forward) with the switch in high position. When the wagon is in point B, action of going back is sent to the system. Order of moving the switch in low position is sent and when this position is reached, the wagon moves from point A to point C. As before, when the point C is reached, the order of move backward is sent to the wagon to return in point A.

Let us consider that the risk assessment has identified a single critical area for the operational part corresponding to the wagon derailment, which can be reached when the wagon moves backwards when it is at point A, moves forward when it is located at point B or point C, or when the switch is in the intermediate position. In order to characterize these events in our modeling, the necessary and sufficient parameters to describe our system are identified as:

- State vector s∈S, which represents the normal behavior of the system as failures are not modeled. It is composed by the values of the carriage and switch position sensors and the actuator values such as in equation (12):

$$s = [\text{Pos}_{\text{Carriage}}\ \text{Pos}_{\text{Switch}};\ \text{Pos}_{\text{Switch}}\ \text{Pos}_{\text{Switch}}]$$
$$= (\text{Pos}_{\text{Carriage}}, \text{Pos}_{\text{Switch}}, \text{Pos}_{\text{Switch}}, \text{Pos}_{\text{Switch}}, \text{Pos}_{\text{Switch}}, \text{Pos}_{\text{Switch}}) \quad (12)$$
$$\text{Avec} : \text{Pos}_{\text{Carriage}} \in \{0..4\}, \text{Pos}_{\text{Switch}} \in \{0..2\}, \text{Pos}_{\text{Switch}} \in \{0,1\}, \text{Pos}_{\text{Switch}} \in \{0,1\},$$
$$\text{Pos}_{\text{Switch}} \in \{0,1\}, \text{Pos}_{\text{Switch}} \in \{0,1\}.$$

In equation (12), $\text{Pos}_{\text{Carriage}}$ indicates the active position sensor (0: carriage in A, 1: carriage in B, 2: carriage in C and 3: carriage in E), $\text{Pos}_{\text{Switch}}$ provides the position of the switch (0: top, 1: intermediate and 2: bottom). For this system, the set of possible states S is composed of at most 192 states and is noted S_{192}

- Actions vector a∈A which represents the actions that can be executed on the system such as $a = (a_1, a_2, a_3, a_4)$ avec $a_i \in \{\text{Pos}_{\text{Switch}}, \text{Pos}_{\text{Switch}}, \text{Pos}_{\text{Switch}}, \text{Pos}_{\text{Switch}}\}\ i \in \{1..4\}$. The wagon stops by simultaneously activating orders a_1 (Forward) and a_2 (Backward). This set A is composed of 16 actions named A_{16}
- All achievable states $S_{\text{Reach}} \subset S$ which contains 113 states including 18 optimal states (obtained using the order model shown in Figure XX), 1 prohibited state and 94 dangerous states $s_{\text{dan}} = s_{\text{Reach}}(s_{\text{dan}} \cap s_{\text{dan}})$
- All the unreachable states $s_{\text{Unreach}} = s \backslash s_{\text{Reach}}$ which contains 79 states.

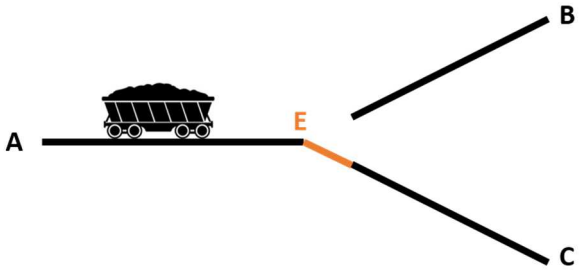

Fig 21. Illustration of the Train/Switch system

A control part model is created based on the specifications of the control law. In the same way, the operative part model is constructed in equation (13) using our automata M :

$$M = \{s_{192}, s_{16}, s, s_0, s_{\text{final}}\} \quad (13)$$

The set S_{192} represents all the achievable states for the system. A_{16} is the set of orders that can be applied to the system (all possible combinations). δ is the transition function that allows the system to evolve and whose effects are synthesized in Table 2. s_0 is the initial state of the system defined by $s_0 = [0\ 2\ 0\ 0\ 0\ 0]$. S_{final} represents all the final states of our PLC and therefore our stop conditions (s_{dan}, s_0 and s_{final}). Table 1 lists the conditions on the State/Action pair (context) that lead to the derailment's prohibited state.

Prohibited State S_{Pro}	**Order $a \in A_{16}$**	**State $s \in S_{192}$ of the system before action $a \in A_{16}$**
Carriage derailment	Backwards ($a = [0\ 1 \times\times]$)	Carriage at point A : $s = [0\ \times\times\times\times\times]$
	Forward ($a = [1\ 0 \times\times]$)	Carriage at point B or C: $s = [1\ \times\times\times\times\times]$ or $s = [2\ \times\times\times\times\times]$
	Forward or Backwards ($a = [1\ 0 \times\times]$) or ($a = [0\ 1 \times\times]$)	Switch in intermediate position: $s = [\times\ 1\ \times\times\times\times]$

Table 1. Summary of conditions leading to the prohibited state of derailment

Order $\boldsymbol{a} \in \boldsymbol{A_{16}}$	State $\boldsymbol{s} \in \boldsymbol{S_{192}}$ of the system before action $\boldsymbol{a} \in \boldsymbol{A_{16}}$
Forward ($\boldsymbol{a_1} = [\mathbf{1\ 0} \times\times]$)	$s = [Pos_{Carriage} \times\times\times\times\times]$ $s' = \delta(s, a_1)$ with $s' = \begin{cases} [3\ 1 \times\times\times]\ si\ Pos_{Carriage} = 0 \\ [1 \times 1 \times\times\times]\ si\ Pos_{Carriage} = 3\ et\ Pos_{Switch} = 0 \\ [2 \times 1 \times\times\times]\ si\ Pos_{Carriage} = 3\ et\ Pos_{Switch} = 2 \\ \emptyset\ si\ Pos_{Carriage} = \{1,2\}\ ou\ Pos_{Switch} = 1 \end{cases}$
Backwards ($\boldsymbol{a_2} = [\mathbf{0\ 1} \times\times]$)	$s = [Pos_{Carriage} \times\times\times\times\times]$ $s' = \delta(s, a_2)$ with $s' = \begin{cases} [0 \times 1 \times\times]\ si\ Pos_{Carriage} = 3 \\ [3 \times\times 1 \times\times]\ si\ Pos_{Carriage} = 1\ et\ Pos_{Switch} = 0 \\ [3 \times\times 1 \times\times]\ si\ Pos_{Carriage} = 2\ et\ Pos_{Switch} = 2 \\ \emptyset\ si\ Pos_{Carriage} = 0\ ou\ Pos_{Switch} = 1 \end{cases}$
Stop ($\boldsymbol{a_3} = [\mathbf{1\ 1} \times\times]$)	$s = [Pos_{Carriage} \times\times\times\times\times]$ $s' = \delta(s, a_3)$ with $s' = [Pos_{Carriage} \times 1\ 1 \times\times]$
Switch up ($\boldsymbol{a_4} = [\times\times\ \mathbf{1} \times]$)	$s = [\times\ Pos_{Switch} \times\times\times]$ $s' = \delta(s, a_4)$ with $s' = \begin{cases} [\times\ 1 \times\times\ 1 \times]\ si\ Pos_{Switch} = 2 \\ [\times\ 0 \times\times\ 1 \times]\ si\ Pos_{Switch} = 1 \\ [\times\ 0 \times\times\ 1 \times]\ si\ Pos_{Switch} = 0 \end{cases}$
Switch down ($\boldsymbol{a_5} = [\times\times\times\ \mathbf{1}]$)	$s = [\times\ Pos_{Switch} \times\times\times]$ $s' = \delta(s, a_5)$ with $s' = \begin{cases} [\times\ 2 \times\times\times\ 1]\ si\ Pos_{Switch} = 2 \\ [\times\ 2 \times\times\times\ 1]\ si\ Pos_{Switch} = 1 \\ [\times\ 1 \times\times\times\ 1]\ si\ Pos_{Switch} = 0 \end{cases}$
No order ($\boldsymbol{a_{16}} = [\mathbf{0\ 0\ 0\ 0\ 0}]$)	$s = [\times\times\times\times\times\times\times]$ $s' = \delta(s, a_{16}) = [\times\times\ 0\ 0\ 0\ 0]$

Table 2. Summary of the actions' effects on the Train/Switch system

We will note that these effects, as a whole, can be combined with each other. To conclude the second step of the methodology, the algorithm 2 for generating the control filter model is executed based on the OP / CP models, which leads to 113 achievable states and 466 contexts leading to prohibited states. The computation time taken by the algorithm (executed offline) is about 130 ms with a computer equipped with an Intel Core i5-63000 processor (CPU 2.40 GHz 2.50 GHz - 8GB RAM). The time is given for the algorithm programmed under Eclipse in Java language. The third step of the methodology is then applied and corresponds to the exploitation of the filter and the detection strategy whose results are presented in previous section (section 4).

6. Discussion and further works

The proposed approach has shown good results for detection of cyberattacks that affect physical system. The blockage of orders before they lead the system in a critical state and detection of *sequential attacks* using combinational or temporal constraints are huge improvement of this method. However, several points have to be changed for increasing the detection rate as:

- State modeling. The S.A.F.E. approach allows the detection of malicious orders by analyzing the evolution of the operative part in stable states. Future works should focus on the evolution of the system between two stable states. This in-depth knowledge of the

operational part will make it possible to detect new types of attacks and improve the functioning of the mechanisms developed, such as over-solicitation.

- Risk assessment step. This stage is decisive for building filters able to detect correctly anomalies toward models. In this approach, this analyze is based on a safety point of view that is why detection algorithm is efficient with attack on security of goods and people. However, none security analyze is made. Thus, some attacks can still be performed on protected system by using this vulnerability. Several works [52] that melt security and safety analysis has to be incorporated in our approach.
- Machine learning. In section 2.2, several approaches have been evaluated [38] and machine learning was not considered for our approach because many data are needed to learn the system for a long period. Moreover, no failure or attack has to be present during the learning phase. These conditions make impossible the use of machine learning in the methodology of filter during the models building. However, incorporation of machine learning for temporal constraints may improve the efficiency of detection algorithm, in particular for systems that are more complex. However, integration of such techniques has to pay attention to attacks where the hackers slowly modify the value of variables in order to degrade the process,
- Filters architecture in lower level. In section 3.3., the proposed filter architecture is used between levels 0 and 1 to the nearest location of operative part. Thus, synchronization issues between filters and the system will be increased.
- ICS architecture. The developed approach is used to protect a small but most critical part of a system as the last shield before damaging the ICS and the environment. Thus, two filters are deployed and analyze exchanged data by considering security of goods and people, quality and equipment solicitation. The same architecture can be generalized horizontally and vertically. In the first case, several critical parts that are protected by filters may have to communicate for improving the system security. This evolution highlights new constraints in modeling step. In the second improvement, architecture proposed in this paper can be used as it is. Additional filters monitor exchanged between control part and the supervision. Work orders and recipes are analyzed by considering the states of the operative part and of the PLC. Moreover, correlation algorithm can be performed between system state, reports sent to level 2 and set points returned by operators. Thus, this new architecture completes approach based on network by using “automation-knowledge”.

7. Conclusion

In this paper, we first identify the cybersecurity context in which this study takes place. A reminder of main attacks that can be performed on an ICS has been made, as well as their diversities and their effects on an ICS. This part allowed highlighting the specificities of industrial control system toward information system belonging to traditional IT. By considering these specificities, solutions deployed nowadays for securing IT systems cannot provide satisfactory results on ICS. The use of models that represent the normal behavior of lower levels of CIM architecture has to be considered. Our approach has been positioned faced to the issue and existing works as well as our contributions. Thus, the major contribution of this paper is to give detection mechanism for detecting 4 types of attacks that can be performed against DES. To conclude, several assumptions and threat models considered in this paper are detailed.

Thus, in a second part, this approach has been detailed. Based on the bibliography, security and safety approaches has been hybridized to ensure detection in lower layers of ICS by using “automation knowledge”. The filter approach, based on verification blocs inside of the PLC, allows incorporating models of physical system in the detection system. On the other hand, Intrusion Detection System is based on probes deployed in a network that check if exchanged data respect the policy. Our approach uses these two techniques for taking advantage of both in mutual improvement. Thus, control and report filters are located outside of the PLC, between operative and control part, in order to be the last shield against hackers in case of attacks.

In a third part, the methodology of system modeling and conception filters as well as algorithms used in the approach are explained. This methodology is based on three steps: after a risk assessment on the system parameters for modeling the system are identified as well as temporal constraints, an exploration state algorithm is performed on the system in order to generate a control filter model. Finally, detection algorithms are implemented into the control and report filters. The operative part is modeled with a finite deterministic automaton while the control part is described with a Petri net. Thus, after explaining the functioning of control and report filters, the paper detailed the different types of models that can be used for each level of protection.

In the fourth section, detection algorithms based on temporal and combinational constraints are explained. Distance and trajectory notions are detailed as they are the basis of our detection algorithms. After detailing the original concept that can be used for continuous variables, an adaptation has been made for discrete event systems. Thus, the shortest path to critical state computes the minimal number of actions that has to be applied on the system for reaching a prohibited state. By decomposing a system into discrete and continuous variables, distance notion can be used for ICS. Distance is an indicator for protecting a system to know how far the system is from a prohibited state with the worst possible sequence of actions. Finally, trajectory concept is defined as the evolution of distance among time and state evolution. Based on different types of models, trajectory of the system with forbidden states or trajectory of orders or state sequences with optimal ones can be computed. As for the distance, trajectories always compute the shortest path to critical states, which is important for analyzing data from filters. Then, the four detection mechanisms are explained, they are based on context detection (deduced from exploration state algorithm of step 2 (section 3.2.)), trajectory analysis (combinational constraints), equipment solicitation monitoring and time windows (temporal constraints). After, our study focuses on post detection algorithm. All these mechanisms have been illustrated on an example for presented the contribution and also the limit.

The fifth part introduces discussion about the feasibility of the methodology on small but most critical parts of ICS based on several examples where the approach is implemented. Thus, results obtained on both simulation examples and industrial platforms prove the implementation of this approach as the last protection to evaluate if an order is malicious. Thus, real-time constraints and combinatorial explosion are discussed. In conclusion, the modeling steps of the methodology are applied on a simulation example.

In the sixth and last section, the approach is discussed to identify improvement ways. Thus, the step of risk assessment, which is a prerequisite for our approach, has to be improved by taking into account a security analysis. The use of security/safety approaches will improve the detection algorithm efficiency by increasing the number of attacks taken into account. Discussions around decentralized architectures for securing ICS conclude this section. A major way of improvement will be the transposition of the proposed approach from levels 1-0 to levels 1-2. To conclude this study, machine learning can be used in the proposed approach for establishing the temporal constraints. Finally, the evolution of the system between two stable states has to be considered in future works.

Acknowledgments

This research is supported by the French organization Direction Générale de l'Armement (DGA) – Maîtrise de l'information based in Bruz, France. Thanks to the S.mart Grenoble-Alpes platform for the open access to the experiments.

References

[1] Stouffer K, Pillitteri V, Lightman S, Abrams M, Hahn A. Guide to Industrial Control Systems (ICS) Security. National Institute of Standards and Technology; 2015. doi:10.6028/NIST.SP.800-82r2.

[2] Ani UPD, He H (Mary), Tiwari A. Review of cybersecurity issues in industrial critical infrastructure: manufacturing in perspective. J Cyber Secur Technol 2017;1:32–74. doi:10.1080/23742917.2016.1252211.
[3] Purdue Research Foundation. A Reference Model For Computer Integrated Manufacturing (CIM) : A Description from the Viewpoint of Industrial Automation. Purdue Research Foundation; 1991.
[4] Sicard F, Zamaï É, Flaus J-M. Critical States Distance Filter Based Approach for Detection and Blockage of Cyberattacks in Industrial Control Systems. Diagn. Secur. Saf. Hybrid Dyn. Cyber-Phys. Syst. by M. Sayed-Mouchaweh, Springer International Publishing; 2018, p. 117 (28p).
[5] Sicard F, Escudero C, Zamaï É, Flaus JM. From ICS Attacks' Analysis to the S.A.F.E. Approach: Implementation of Filters based on Behavioral Models and Critical State Distance for ICS Cybersecurity. 2nd Cyber Secur. Netw. Conf., Paris, France: 2018, p. 8.
[6] Cheminod M, Durante L, Seno L, Valenzano A. Detection of attacks based on known vulnerabilities in industrial networked systems. J Inf Secur Appl 2017;34:153–65. doi:10.1016/j.jisa.2016.06.003.
[7] Leszczyna R. A review of standards with cybersecurity requirements for smart grid. Comput Secur 2018;77:262–76. doi:10.1016/j.cose.2018.03.011.
[8] Wang W, Lu Z. Cyber security in the Smart Grid: Survey and challenges. Comput Netw 2013;57:1344–71. doi:10.1016/j.comnet.2012.12.017.
[9] Knapp ED. Industrial network security: securing critical infrastructure networks for smart grid, scada, and other industrial control systems. 2nd edition. Waltham, MA: Elsevier; 2014.
[10] Giraldo J, Urbina D, Cardenas A, Valente J, Faisal M, Ruths J, et al. A Survey of Physics-Based Attack Detection in Cyber-Physical Systems. ACM Comput Surv 2018;51:1–36. doi:10.1145/3203245.
[11] Wang Y, Xu Z, Zhang J, Xu L, Wang H, Gu G. SRID: State Relation Based Intrusion Detection for False Data Injection Attacks in SCADA. In: Kutyłowski M, Vaidya J, editors. Comput. Secur. - ESORICS 2014, Springer International Publishing; 2014, p. 401–18.
[12] Li W, Xie L, Deng Z, Wang Z. False sequential logic attack on SCADA system and its physical impact analysis. Comput Secur 2016;58:149–59. doi:10.1016/j.cose.2016.01.001.
[13] McLaughlin S, Konstantinou C, Wang X, Davi L, Sadeghi A-R, Maniatakos M, et al. The Cybersecurity Landscape in Industrial Control Systems. Proc IEEE 2016;104:1039–57. doi:10.1109/JPROC.2015.2512235.
[14] Sicard F, Zamaï É, Flaus J-M. Filter based approach with temporal and combinational constraints for Cybersecurity of Industrial Control Systems. 10th IFAC Symp. Fault Detect. Superv. Saf. Tech. Process. SAFEPROCESS 2018, Warsaw, Poland: IFAC; 2018, p. 8.
[15] Kriaa S, Pietre-Cambacedes L, Bouissou M, Halgand Y. A survey of approaches combining safety and security for industrial control systems. Reliab Eng Syst Saf 2015;139:156–78. doi:10.1016/j.ress.2015.02.008.
[16] Fovino IN, Masera M, De Cian A. Integrating cyber attacks within fault trees. Reliab Eng Syst Saf 2009;94:1394–402. doi:10.1016/j.ress.2009.02.020.
[17] Piètre-Cambacédès L, Bouissou M. Cross-fertilization between safety and security engineering. Reliab Eng Syst Saf 2013;110:110–26. doi:10.1016/j.ress.2012.09.011.
[18] Nguyen DT, Duong QB, Zamai E, Shahzad MK. Fault diagnosis for the complex manufacturing system. Proc Inst Mech Eng Part O J Risk Reliab 2016;230:178–94. doi:10.1177/1748006X15623089.
[19] Jarmakiewicz J, Parobczak K, Maślanka K. Cybersecurity protection for power grid control infrastructures. Int J Crit Infrastruct Prot 2017;18:20–33. doi:10.1016/j.ijcip.2017.07.002.
[20] Slay J, Miller M. A security architecture for SCADA networks. ACIS 2006 Proc 2006:12.
[21] Weiss J. Industrial Control System Cyber Security and the Critical Infrastructures. INSIGHT 2016;19:33–36.
[22] Villemeur A. Reliability, Availability, Maintainability and Safety Assessment, Volume 2, Assessment, Hardware, Software and Human Factors. Wiley; 1992.
[23] Rao N, Srivastava S, Sreekanth KS. PKI Deployment Challenges and Recommendations for ICS Networks. Int J Inf Secur Priv 2017;11:38–48. doi:10.4018/IJISP.2017040104.

[24] Milinković SA, Lazić LR. Industrial PLC security issues. 2012 20th Telecommun. Forum TELFOR, 2012, p. 1536–9. doi:10.1109/TELFOR.2012.6419513.
[25] Basnight Z, Butts J, Lopez J, Dube T. Firmware modification attacks on programmable logic controllers. Int J Crit Infrastruct Prot 2013;6:76–84. doi:10.1016/j.ijcip.2013.04.004.
[26] Schuett C, Butts J, Dunlap S. An evaluation of modification attacks on programmable logic controllers. Int J Crit Infrastruct Prot 2014;7:61–8. doi:10.1016/j.ijcip.2014.01.004.
[27] Peck D, Peterson D. Leveraging ethernet card vulnerabilities in field devices. SCADA Secur. Sci. Symp., 2009, p. 1–19.
[28] Ren X, Blanton RD, Tavares VG. A Learning-Based Approach to Secure JTAG Against Unseen Scan-Based Attacks. 2016 IEEE Comput. Soc. Annu. Symp. VLSI ISVLSI, 2016, p. 541–6. doi:10.1109/ISVLSI.2016.107.
[29] McLaughlin S, Zonouz SA, Pohly DJ, McDaniel PD. A Trusted Safety Verifier for Process Controller Code. Proceeding 2014 Netw. Distrib. Syst. Secur. Symp., California (USA): 2014.
[30] Tuptuk N, Hailes S. Security of smart manufacturing systems. J Manuf Syst 2018;47:93–106. doi:10.1016/j.jmsy.2018.04.007.
[31] Amoah R, Camtepe S, Foo E. Securing DNP3 Broadcast Communications in SCADA Systems. IEEE Trans Ind Inform 2016;12:1474–85. doi:10.1109/TII.2016.2587883.
[32] Goldenberg N, Wool A. Accurate modeling of Modbus/TCP for intrusion detection in SCADA systems. Int J Crit Infrastruct Prot 2013;6:63–75. doi:10.1016/j.ijcip.2013.05.001.
[33] Clotet X, Moyano J, León G. A real-time anomaly-based IDS for cyber-attack detection at the industrial process level of Critical Infrastructures. Int J Crit Infrastruct Prot 2018;23:11–20. doi:10.1016/j.ijcip.2018.08.002.
[34] Cheung S, Dutertre B, Fong M, Lindqvist U, Skinner K, Valdes A. Using model-based intrusion detection for SCADA networks. Proc. SCADA Secur. Sci. Symp., vol. 46, Citeseer; 2007, p. 1–12.
[35] Lin H, Slagell A, Di Martino C, Kalbarczyk Z, Iyer RK. Adapting Bro into SCADA: Building a Specification-based Intrusion Detection System for the DNP3 Protocol. Proc. Eighth Annu. Cyber Secur. Inf. Intell. Res. Workshop, New York, NY, USA: ACM; 2013, p. 5:1–5:4. doi:10.1145/2459976.2459982.
[36] Morris T, Vaughn R, Dandass Y. A Retrofit Network Intrusion Detection System for MODBUS RTU and ASCII Industrial Control Systems, IEEE; 2012, p. 2338–45. doi:10.1109/HICSS.2012.78.
[37] Barbosa RRR, Sadre R, Pras A. Exploiting traffic periodicity in industrial control networks. Int J Crit Infrastruct Prot 2016;13:52–62. doi:10.1016/j.ijcip.2016.02.004.
[38] Linda O, Vollmer T, Manic M. Neural Network based Intrusion Detection System for critical infrastructures. 2009 Int. Jt. Conf. Neural Netw., 2009, p. 1827–34. doi:10.1109/IJCNN.2009.5178592.
[39] Fovino IN, Coletta A, Carcano A, Masera M. Critical State-Based Filtering System for Securing SCADA Network Protocols. IEEE Trans Ind Electron 2012;59:3943–50. doi:10.1109/TIE.2011.2181132.
[40] Carcano A, Coletta A, Guglielmi M, Masera M, Nai Fovino I, Trombetta A. A Multidimensional Critical State Analysis for Detecting Intrusions in SCADA Systems. IEEE Trans Ind Inform 2011;7:179–86. doi:10.1109/TII.2010.2099234.
[41] Manandhar K, Cao X, Hu F, Liu Y. Detection of Faults and Attacks Including False Data Injection Attack in Smart Grid Using Kalman Filter. IEEE Trans Control Netw Syst 2014;1:370–9. doi:10.1109/TCNS.2014.2357531.
[42] Mitchell R, Chen R. Behavior-rule based intrusion detection systems for safety critical smart grid applications. IEEE Trans Smart Grid 2013;4:1254–1263.
[43] Pan S, Morris TH, Adhikari U, Madani V. Causal Event Graphs Cyber-physical System Intrusion Detection System. Proc. Eighth Annu. Cyber Secur. Inf. Intell. Res. Workshop, New York, NY, USA: ACM; 2013, p. 40:1–40:4. doi:10.1145/2459976.2460022.
[44] Pièvre-Cambacédès L, Bouissou M. Attack and defense modeling with BDMP. Int. Conf. Math. Methods Models Archit. Comput. Netw. Secur., Springer; 2010, p. 86–101.
[45] Hadziosmanovic D, Sommer R, Zambon E, Hartel P. Through the eye of the PLC: towards semantic security monitoring for industrial control systems. Int Comput Sci Inst Berkeley 2013.

[46] Erez N, Wool A. Control variable classification, modeling and anomaly detection in Modbus/TCP SCADA systems. Int J Crit Infrastruct Prot 2015;10:59–70. doi:10.1016/j.ijcip.2015.05.001.
[47] Papa S, Casper W, Nair S. A transfer function based intrusion detection system for SCADA systems. Homel. Secur. HST 2012 IEEE Conf. Technol. For, 2012, p. 93–8. doi:10.1109/THS.2012.6459831.
[48] Yaseen AA, Bayart M. Towards distinguishing between faults and cyber-attacks in the networked control system. 2016 World Congr. Ind. Control Syst. Secur. WCICSS, 2016, p. 1–8. doi:10.1109/WCICSS.2016.7882611.
[49] Cherdantseva Y, Burnap P, Blyth A, Eden P, Jones K, Soulsby H, et al. A review of cyber security risk assessment methods for SCADA systems. Comput Secur 2016;56:1–27. doi:10.1016/j.cose.2015.09.009.
[50] Ellis A. Integrating Industrial Control System (ICS) safety and security #x2014; A potential approach. 10th IET Syst. Saf. Cyber-Secur. Conf. 2015, 2015, p. 1–7. doi:10.1049/cp.2015.0294.
[51] Kriaa S, Bouissou M, Laarouchi Y. A model based approach for SCADA safety and security joint modelling: S-cube. 10th IET Syst. Saf. Cyber-Secur. Conf. 2015, 2015, p. 1–6. doi:10.1049/cp.2015.0293.
[52] Abdo H, Kaouk M, Flaus J-M, Masse F. A safety/security risk analysis approach of Industrial Control Systems: A cyber bowtie – combining new version of attack tree with bowtie analysis. Comput Secur 2018;72:175–95. doi:10.1016/j.cose.2017.09.004.
[53] Cárdenas AA, Amin S, Lin Z-S, Huang Y-L, Huang C-Y, Sastry S. Attacks against process control systems: risk assessment, detection, and response. Proc. 6th ACM Symp. Inf. Comput. Commun. Secur., ACM; 2011, p. 355–366.
[54] Friedberg I, McLaughlin K, Smith P, Laverty D, Sezer S. STPA-SafeSec: Safety and security analysis for cyber-physical systems. J Inf Secur Appl 2017;34:183–96. doi:10.1016/j.jisa.2016.05.008.
[55] Cruette D, Bourey JP, Gentina JC. Hierarchical specification and validation of operating sequences in the context of FMSs. Comput Integr Manuf Syst 1991;4:140–156.
[56] Mitchell R, Chen I-R. A survey of intrusion detection techniques for cyber-physical systems. ACM Comput Surv 2014;46:1–29. doi:10.1145/2542049.
[57] Taylor A, Japkowicz N, Leblanc S. Frequency-based anomaly detection for the automotive CAN bus. 2015 World Congr. Ind. Control Syst. Secur. WCICSS, 2015, p. 45–9. doi:10.1109/WCICSS.2015.7420322.
[58] Kleinmann A, Wool A. Automatic Construction of Statechart-Based Anomaly Detection Models for Multi-Threaded Industrial Control Systems. ACM Trans Intell Syst Technol 2017;8:55:1–55:21. doi:10.1145/3011018.
[59] Mohan S, Bak S, Betti E, Yun H, Sha L, Caccamo M. S3A: Secure system simplex architecture for enhanced security and robustness of cyber-physical systems. Proc. 2nd ACM Int. Conf. High Confid. Networked Syst., ACM; 2013, p. 65–74.
[60] Zimmer C, Bhat B, Mueller F, Mohan S. Time-based Intrusion Detection in Cyber-physical Systems. Proc. 1st ACMIEEE Int. Conf. Cyber-Phys. Syst., New York, NY, USA: ACM; 2010, p. 109–118. doi:10.1145/1795194.1795210.
[61] Henry S, Zamaï E, Jacomino M. Logic control law design for automated manufacturing systems. Eng Appl Artif Intell 2012;25:824–36. doi:10.1016/j.engappai.2012.01.005.