Unknown Sensor Attacks in Supervisory Control of DES

Michel R. C. Alves * **Patrícia N. Pena** ** **Karen Rudie** ***

* *Graduate Program in Electrical Engineering - Universidade Federal de Minas Gerais - Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brazil (e-mail:michelrodrigo@ufmg.br).*
** *Department of Electronics Engineering - Universidade Federal de Minas Gerais - Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brazil (e-mail: ppena@ufmg.br)*
*** *Department of Electrical and Computer Engineering, and Ingenuity Labs Research Institute, Queen's University, Kingston, Ontario K7L 3N6, Canada (e-mail: karen.rudie@queensu.ca)*

Abstract: The background for this work is the supervisory control of discrete-event systems under partial observation. Attackers that are able to insert or erase occurrences of particular output symbols can modify the supervisor's observation and, by doing so, can lead the controlled system to undesirable states. A scenario with multiple attackers is considered, each one being an element of a set, called an *attack set*. We also assume that only one of the attackers within an attack set is acting, although we do not know which one. According to previous results in the literature, a supervisor that enforces a given legal language, regardless of which attacker is acting, can be designed if the legal language is controllable and satisfies a property called P-observability for an attack set. The latter is an extended notion of observability and is related with the supervisor's ability to always distinguish between outputs that require different control actions, even if the outputs were attacked. We present a new approach for checking if a given language is P-observable for an attack set when it is represented as an automaton.

Keywords: Discrete-event system - Supervisory control - Attacks on output symbols - P-observability

1. INTRODUCTION

In recent years, the extensive use of communication networks used in industry increases vulnerabilities for malicious attacks, thus making these networks unreliable. This concern did not exist in classical control systems (Li et al., 2020), which justifies the research effort on the subject. In order to make networks reliable, defense strategies have to be considered, which can be roughly classified as detection of attacks and prevention of the attack's effects.

In Discrete-Event Systems (DES), several approaches that address the problem of attacks have been proposed. In the context of supervisory control, there are works that present methods for designing robust supervisors ((Meira-Goes et al., 2021) (Wang and Pajic, 2019), (Wakaiki et al., 2019) and (Su, 2018), among others) and the design of a detection module (as in (Li et al., 2020), (Gao et al., 2019), (Carvalho et al., 2018), (Lima et al., 2018) and (Lima et al., 2017)). Furthermore, some authors focus on studying the design methods for the attackers, using as argument the claim that a good understanding about the adversaries can provide better insights on how to defend against them ((Lin and Su, 2021), (Zhang et al., 2021), (Mohajerani et al., 2020), (Fritz and Zhang, 2018)).

There are some different types of attacks considered in the literature. In Meira-Goes et al. (2021) Zhang et al. (2021) and Mohajerani et al. (2020), for example, attackers can insert and erase events from the sensor channel. Alternatively, the attackers considered by Lin and Su (2021), Lima et al. (2019) Khoumsi (2019) and Zhu et al. (2019), among others, are able to modify sensor and actuator events.

In this paper we study attacks on the output symbols and the attacker's goal is to lead the controlled system into an undesirable state. In this context, the authors of (Wakaiki et al., 2019) proposed a design method for robust supervisors that, regardless of which symbols are under attack, always enforce the desired behavior. Two conditions are imposed over the language that represents the desired behavior: i) controllability and ii) P-observability for an attack set. The latter is a modified version of the classical notion of observability and is related to the supervisor's ability to always distinguish between observations that require different control actions. A test for checking this property is also presented, consisting of a series of pairwise classical observability tests (Wakaiki et al., 2019).

In this work, we improve on the test of P-observability for an attack set by providing a new method that actually checks for the property itself and can be done in a single run, as opposed to the multiple observability tests.

The paper is organized as follows. In Section 2, we provide some basic preliminaries which are needed for understanding this work. In Section 3 we introduce the attack model considered and in Section 4 we present a new test for verifying P-observability. Finally, we present the conclusions in Section 5.

2. BACKGROUND ON DISCRETE-EVENT SYSTEMS

The behavior of a DES is modeled using strings of symbols, called *events*, taken from a finite set Σ. The set of all finite strings of events in Σ, including the empty string ε, is denoted by Σ^*. Given two strings, $s, u \in \Sigma^*$, their concatenation is written as su. A string $s \in \Sigma^*$ is a prefix of $t \in \Sigma^*$, represented as $s \leq t$, if there exists $u \in \Sigma^*$ such that $su = t$. A language L is any subset of Σ^*. The prefix-closure $\overline{L}$ of a language $L \subseteq \Sigma^*$ is the set of all prefixes of strings in L. It is defined as $\overline{L} := \{s \in \Sigma^* | s \leq t \text{ for some } t \in L\}$. A language L is said to be prefix-closed if $L = \overline{L}$.

Automata are a formalism to represent languages. An automaton G is defined as a tuple $G := (Q, \Sigma, \delta, q_0)$, where Q is a finite set of states, Σ is the nonempty finite set of events, $\delta : Q \times \Sigma \to Q$ is a partially defined transition function and q_0 is the initial state. The notation $\delta(q, \sigma)!$ represents that $\delta(q, \sigma)$ is defined for some $q \in Q$ and $\sigma \in \Sigma$. The transition function δ can be extended to a function $Q \times \Sigma^* \to Q$ according to $\delta(q, \varepsilon) := q$ and $\delta(q, s\sigma) := \delta(\delta(q, s), \sigma)$, with $q \in Q$, $s \in \Sigma^*$ and $\sigma \in \Sigma$. With abuse of notation, we sometimes treat δ as a set and $(q, \sigma, q') \in \delta$ if and only if $\delta(q, \sigma) = q'$. The automaton is said to be deterministic if $(q, \sigma, q'), (q, \sigma, q'') \in \delta$ always implies that $q' = q''$. The map $\Gamma : Q \to 2^{\Sigma}$ defined as $\Gamma(q) := \{\sigma \in \Sigma | (q, \sigma, q') \in \delta, \text{ for any } q' \in Q\}$ is the set of *feasible* events at a given state $q \in Q$. The language generated by G, denoted by $\mathcal{L}(G)$, is defined as $\mathcal{L}(G) := \{s \in \Sigma^* | \delta(q_0, s)!\}$.

2.1 Supervisory Control

The set of events Σ can be partitioned as $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc}$, where Σ_c is the set of *controllable* events and Σ_{uc} is the set of *uncontrollable* events. When G models the uncontrolled behavior of a DES, some of its states can be undesirable states, representing situations we want to avoid, as blocking or insecure operation. The legal behavior is modeled as a *desired language* $K \subseteq \mathcal{L}(G)$, that is, K is a subset of $\mathcal{L}(G)$, containing only the legal strings. We can enforce the desired language K over G by using a structure called a *supervisor*, denoted by S, that acts over the set of controllable events. Since the supervisor cannot prevent uncontrollable events from happening, we say that a desired language K is *controllable with respect to* $\mathcal{L}(G)$ if $\overline{K}\Sigma_{uc} \cap \mathcal{L}(G) \subseteq \overline{K}$.

If a language K is controllable, then there exists a supervisor that implements K. If K is not controllable, then a supremal controllable sublanguage, $\mathcal{S}$, that implements the most permissive and nonblocking behavior that does not violate the behavior imposed by K can be obtained.

2.2 Partial Observation

Some of the events in a DES may not be observable. In such case the system is said to be under *partial observation* and the event set Σ can be partitioned as $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, where Σ_o is the set of *observable* events, whereas Σ_{uo} is the set of *unobservable* events. The partial observation can be represented by an observation map $P : \Sigma^* \to \Sigma_o^*$, also called *natural projection*, that maps strings in Σ^* into observations in Σ_o^*. It is defined as

$$P(\varepsilon) := \varepsilon$$
$$P(\sigma) := \begin{cases} \sigma & \text{if } \sigma \in \Sigma_o \\ \varepsilon & \text{if } \sigma \in \Sigma_{uo} \end{cases}$$
$$P(s\sigma) := P(s)P(\sigma) \text{ for } s \in \Sigma^*, \sigma \in \Sigma.$$

The inverse observation map P^{-1}, called *inverse projection*, is defined as $P^{-1}(t) := \{s \in \Sigma^* | P(s) = t\}$. A prefix-closed language $K \subseteq L$ is P-observable *with respect to* L if

$$\ker P \subseteq \text{act}_{K \subset L} \tag{1}$$

where $\ker P$ denotes the relation on Σ^* defined by

$$\ker P := \{(w, w') \in \Sigma^* \times \Sigma^* | P(w) = P(w')\} \tag{2}$$

and $\text{act}_{K \subset L}$ is the binary relation on Σ^* defined by

$$\begin{aligned} \text{act}_{K \subset L} := \{&(w, w') \in \Sigma^* \times \Sigma^* | \\ &w, w' \in K \implies \nexists \sigma \in \Sigma \text{ s.t.} [w\sigma \in K, \\ &w'\sigma \in L \setminus K] \text{ or } [w\sigma \in L \setminus K, w'\sigma \in K]\}. \end{aligned} \tag{3}$$

The relation $\text{act}_{K \subset L}$ has all pairs of strings $w, w' \in K$ such that the new strings $w\sigma$ and $w'\sigma$ are either both in K or both in $L \setminus K$, for all $\sigma \in \Sigma$. In words, a language is P-observable when a supervisor is always capable of distinguishing between strings that require different control actions.

3. SUPERVISORY CONTROL UNDER ATTACKS

In this work we consider attackers whose goal is to prevent the supervisor from achieving the prefix-closed desired language $K \subseteq \mathcal{L}(G)$, as in Wakaiki et al. (2019). The attackers have full observation of the communication channel between plant and supervisor and can corrupt the string of output symbols $P(w)$, $w \in \Sigma^*$ in multiple ways, by erasing and/or inserting specific output symbols. In this work, output symbols are the symbols sent from the plant to the supervisor, as an outcome of sensor readings. Also, we assume the supervisor sends a new control action to the plant whenever it receives new information.

Figure 1 represents a closed-loop controlled system under attack. The plant executes the string $w \in \mathcal{L}(G)$, but only the string $P(w)$ can be observed by the supervisor, because of the partial observation. Nevertheless, an attacker placed in the communication channel from the plant to the supervisor can corrupt the string $P(w)$ by changing it to string $y \in \Sigma_o^*$. Upon reception of string y, the supervisor will then issue the control action $S(y)$. Depending on how the attacker chooses to modify $P(w)$, it can induce the supervisor to issue a control action that will make the plant reach an undesirable state. Next we characterize how the

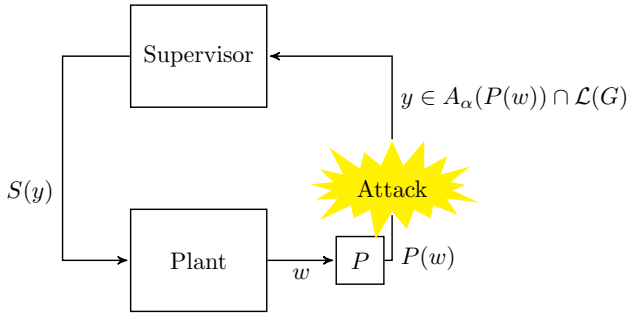

Figure 1. Closed loop controlled system under attacks (Adapted from Wakaiki et al. (2019)).

attacker can modify $P(w)$.

Given a set of symbols $\alpha \subseteq \Sigma_v \subseteq \Sigma_o$ in the observation alphabet, where Σ_v is the set of *vulnerable* events, the map $R_{\neg\alpha} : \Sigma \to (\Sigma_o \setminus \alpha) \cup \{\varepsilon\}$ is called *α-removal observation map* and is defined as

$$R_{\neg\alpha}(t) := \begin{cases} \varepsilon & t \in \alpha \\ t & t \notin \alpha \end{cases} \tag{4}$$

and can be extended to a map defined for strings of output symbols in the same way as a natural projection P. An *observation attack* is a set-valued map $A_\alpha : \Sigma_o^* \to 2^{\Sigma_o^*}$ that assigns to each string $u \in \Sigma_o^*$ the set of all strings $v \in \Sigma_o^*$ that can be obtained from u by an arbitrary number of insertions or deletions of symbols in α. It is given by

$$A_\alpha(u) := \{v \in \Sigma_o^* | R_{\neg\alpha}(u) = R_{\neg\alpha}(v)\}. \tag{5}$$

If $\alpha = \emptyset$, then $A_\emptyset$ corresponds to the absence of attack.

Thus, instead of receiving the string $P(w)$, the supervisor receives one string among the set $A_\alpha(P(w))$. The map A_α is called an *observation attack* or simply *attacker* and the map $A_\alpha P : \Sigma^* \to 2^{\Sigma_o^*}$ obtained by the composition $A_\alpha P := A_\alpha \circ P$ the corresponding *attacked observation map*. In other words, the map $A_\alpha P(w)$ results in all strings that can be formed with the manipulation of events in $\alpha \subseteq \Sigma_v \subseteq \Sigma_o$, by inserting events into or erasing events from the observation $P(w)$. The goal of the attacker is, by changing the observation string, to make the supervisor accept a string that is not in K or reject a string that is in K. We can also interpret the attack as a factor that increases the supervisor's uncertainty about which state the plant is really in.

We assume that if $w \in K$ is the string executed in the plant so far, then the attacker, which has knowledge about the system, won't insert an event $\sigma \in \alpha$ in the observation if $w\sigma \notin K$, since that would reveal the attacker's presence if an intrusion detection system was in place.

A more interesting scenario consists of multiple attackers, instead of only one. By multiple attackers, we mean that there is an *attack set*, denoted by $\mathcal{A}$, such that $\mathcal{A} = \{A_\emptyset, A_{\alpha_1}, A_{\alpha_2}, \ldots, A_{\alpha_M}\}$, where each $\alpha_i \subseteq \Sigma_v$ and $A_\emptyset$ represents the absence of attack. We assume that only one of the attackers in an attack set $\mathcal{A}$ is acting, but we do not know which one.

3.1 P-Observability for an Attack set

The authors of (Wakaiki et al., 2019) extend the notion of P-observability, presenting the P-observability *for an attack set* $\mathcal{A}$, which is given by

$$R_{A_{\alpha_i}, A_{\alpha_j}} \subset \text{act}_{K \subset L} \tag{6}$$

where the relation $R_{A_{\alpha_i}, A_{\alpha_j}}$, with $A_{\alpha_i}, A_{\alpha_j} \in \mathcal{A}$, contains all pairs of strings that may result in attacked observation maps $A_{\alpha_i}P$ and $A_{\alpha_j}P$ with a common string of output symbols, i.e.,

$$R_{A_{\alpha_i}, A_{\alpha_j}} := \{(w, w') \in \Sigma^* \times \Sigma^* | A_{\alpha_i}P(w) \cap A_{\alpha_j}P(w') \neq \emptyset\} \tag{7}$$

and $\text{act}_{K \subset L}$ is as in (3). In Wakaiki et al. (2019), the authors present a result that gives the conditions for the existence of a supervisor that, regardless of which attacker $A_\alpha \in \mathcal{A}$ is acting, can enforce a desired language K. These conditions are: *i*) K must be controllable with respect to G and; *ii*) K must be P-observable for the attack set $\mathcal{A}$.

The fact that the controllability of K is a necessary condition for the existence of a supervisor is to be expected. Regarding the P-observability for an attack set $\mathcal{A}$, if this condition does not hold, it is possible to find two attacks $A_{\alpha_i}, A_{\alpha_j} \in \mathcal{A}$ that would result in the same observation $y \in \Sigma_o^*$ for two distinct strings $w, w' \in K$, and $w\sigma$ would transition to an element in K and $w'\sigma$ to an element outside K, or vice-versa. This means that, upon observing y, the supervisor cannot decide which control action to take.

The next theorem shows how the test for P-observability for an attack set $\mathcal{A}$ can be done by reducing it to a classical observability test.

Theorem *1.* (Wakaiki et al., 2019) For every nonempty prefix-closed set $K \subseteq L$ and attack set $\mathcal{A} = \{A_{\alpha_1}, A_{\alpha_2}, \ldots, A_{\alpha_M}\}$ consisting of $M \geq 1$ observation attacks, K is P-observable for the set of attacks $\mathcal{A}$ if and only if K is $(R_{\neg\alpha} \circ P)$-observable (in the classical sense, i.e., without attacks) for every set $\alpha := \alpha_i \cup \alpha_j$, $\forall i, j \in \{1, 2, \ldots, M\}$. □

Theorem 1 states that the P-observability for an attack set $\mathcal{A}$ can be tested by picking every possible pair of two attackers A_{α_i}, A_{α_j} and checking if regular observability is obtained if the symbols affected by the two attackers are removed from the observation. The authors claim that, using the test for classical observability presented in (Cassandras and Lafortune, 2007, Section 3.7.3), P-observability for an attack set can be tested with time complexity $O(M^5)$, where $M = |\mathcal{A}|$. However, this complexity is obtained from a very specific attack set and a generalization is not provided.

4. NEW TEST FOR P-OBSERVABILITY FOR AN ATTACK SET

This section presents a new test for P-observability for an attack set. Rather than realizing multiple tests of classical observability, the proposed test checks P-observability for

an attack set itself. The test is applied over an automaton that implements a desired language K. To illustrate the concepts introduced, we present a running example.

An attack can increase the uncertainty for an observer about which state the plant is in, by manipulating the symbols in the observation. This uncertainty can be represented as a relation of pairs of states and each attacker induces a different one. This concept is formalized in Definition 2.

Definition *2.* (Indistinguishable states with respect to attacker A_α) For a given attacker $A_\alpha \in \mathcal{A}$ and automaton G, the relation $\Pi_\alpha \subseteq Q \times Q$ defined as

$$\Pi_\alpha := \{(q, q') \in Q \times Q | (\forall wu \in \mathcal{L}(G))[\delta(q_0, w) = q \wedge \delta(q_0, wu) = q' \wedge u \in (\alpha \cup \Sigma_{uo})^*]\} \quad (8)$$

is the relation of indistinguishable states with respect to attacker A_α. □

In words, Π_α has all pairs of states (q, q') such that state q' is reachable from state q with events in α, the events that attacker A_α can manipulate, or with events in Σ_{uo}. This definition is inspired by (Wang et al., 2007), where pairs of indistinguishable states arise due only to partial observation. Note that a state is always indistinguishable with itself.

Example *3.* Suppose a conveyor belt transports two types of unfinished products. These products are transported through a test unit, whose automaton representation is shown in Fig. 2. Whenever a new product arrives (event n), the conveyor belt stops and a sensor identifies the type of the product (a or b). A test is then performed according to the type of the product. If it passes the test (p), the unfinished product continues its path in the conveyor belt (event A). However, if it fails (f), then the unfinished product must be discarded (R). We want to avoid accepting or rejecting a product before performing the test. The uncontrolled behavior of this system is shown in Fig 2. State 6, in gray, is a bad state, since it represents the test unit accepting or rejecting an unfinished product before testing it.

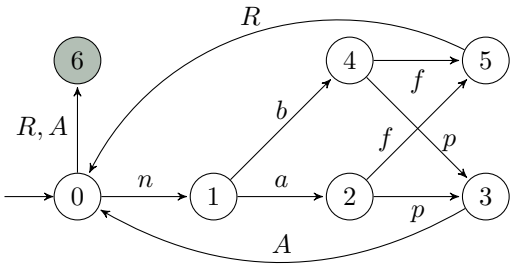

Figure 2. Automaton G of Example 3. $\Sigma_c = \{A, R\}, \Sigma_u = \{a, b, f, n, p\}$ and $\Sigma_v = \{a, f, n, p\}$.

Now suppose that the test unit has some known vulnerabilities. In one of these vulnerabilities, events n and a are compromised, while in other vulnerability, events f and p are the ones compromised. We know that one of these vulnerabilities was exploited by attackers, but we do not know which one. The test of P-observability for an attack set allows us to determine if it is possible to obtain a supervisor that will enforce the desired language, regardless of which attacker is acting. Firstly, we apply Def. 2. To do this, we need to find an automaton H that represents the desired language for our system. This automaton can be obtained by taking the automaton of Fig. 2 and removing state 6. Fig. 3 shows the new automaton H' obtained, where we also omit the transitions labeled with non-vulnerable events (automaton H' is the automaton H after the omission of some transitions), since they are not relevant for our analysis.

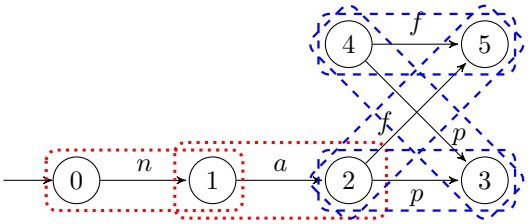

Figure 3. Automaton H'. The rectangles with dotted (red) and dashed (blue) borders represent the effect of the attackers $A_{\{n,a\}}$ and $A_{\{f,p\}}$, respectively

From the information given in the example, we have $\mathcal{A} = \{A_{\{n,a\}}, A_{\{f,p\}}\}$. We introduce here a visual interpretation of the attack's effect over H'. We add a rectangle around each pair of states that are connected by a transition labeled with a vulnerable event. This is done for each attacker $A_\alpha \in \mathcal{A}$. Each pair of states inside a rectangle represents states that can be made indistinguishable by an attacker, by inserting or erasing the corresponding event. To obtain the relations $\Pi_{\{n,a\}}$ and $\Pi_{\{f,p\}}$, one can apply Def. 2 or, alternatively, obtain it by inspecting Fig. 3, where the states inside the rectangles are indistinguishable:

$\Pi_{\{n,a\}} = \{(0,0), (0,1), (0,2), (1,1), (1,2), (2,2)\}$ and

$\Pi_{\{f,p\}} = \{(2,2), (2,3), (2,5), (3,3), (4,3), (4,4), (4,5), (5,5)\}$. □

When checking P-observability for an attack set, one operation that is very convenient is to check if two given states $q, q' \in Q$ can be confused with each other. As we do not know which attacker is actually acting, we have to consider the effect of all attackers. This is done by joining all relations of indistinguishable states into one, according to Definition 4.

Definition *4.* (Indistinguishable states with respect to the attack set $\mathcal{A}$) For an attack set $\mathcal{A}$, the relation $\Pi_\mathcal{A} \subseteq Q \times Q$ defined as

$$\Pi_\mathcal{A} := \bigcup_{A_\alpha \in \mathcal{A}} \Pi_\alpha \quad (9)$$

is the relation of indistinguishable states with respect to attack set $\mathcal{A}$. □

The relation $\Pi_\mathcal{A}$ is obtained by taking the union of all the pairs in Π_α. A pair $(q, q') \in \Pi_\alpha$ means that state q' can be reached from state q after the insertion of some events by an attacker $A_\alpha \in \mathcal{A}$ and/or after the occurrence of unobservable events, making them indistinguishable. Thus, the relation $\Pi_\mathcal{A}$ has all pairs of states that are indistinguishable with each other, which is a consequence of the insertion of events by all attackers in the attack set.

Example *5.* Continuing from Example 3, if we apply Def. 4 over the relations $\Pi_{\{n,a\}}$ and $\Pi_{\{f,p\}}$, we obtain:

$\Pi_{\mathcal{A}} = \{(0,0),(0,1),(0,2),(1,1),(1,2),(2,2),(2,3),(2,5),(3,3),(4,4),(4,3),(4,5),(5,5)\}$. □

Definition 6 allows us to obtain the effect of all attackers combined pairwise.

Definition *6.* (Relation of indistinguishable states for pairwise combined attacks) The relation $\Pi^2_{\mathcal{A}} \subseteq Q \times Q$ defined as

$$\Pi^2_{\mathcal{A}} := (\Pi_{\mathcal{A}} \circ \Pi_{\mathcal{A}}) \cup \widehat{\Pi}_{\mathcal{A}} \tag{10}$$

where $\widehat{\Pi}_{\mathcal{A}}$ is defined as

$$\widehat{\Pi}_{\mathcal{A}} = \{(q_1,q_2),(q_2,q_1) \in Q \times Q | (\exists q \in Q) [(q,q_1),(q,q_2) \in \Pi_{\mathcal{A}}]\} \tag{11}$$

and $\Pi_{\mathcal{A}}$ is given by Def. 4, is the relation of indistinguishable states for the pairwise combination of attacks. □

The idea behind $\Pi^2_{\mathcal{A}}$ is to get the effect of the attack if any two attackers were allowed to cooperate with each other. Although we assume that only one attacker is acting, considering that the attackers are acting together allows us to find if there are at least two attackers that can produce the same observation starting from two different strings. w, w', which is related to the relation $R_{A_{\alpha_i},A_{\alpha_j}}$, defined in (7).

Example *7.* Continuing from Example 5, we apply Def. 6 over the relation $\Pi_{\mathcal{A}}$ in order to obtain the relation of indistinguishable states for pairwise combined attacks:

$\widehat{\Pi}_{\mathcal{A}} = \{(0,0),(0,1),(1,0),(0,2),(1,1),(1,2),(2,0),(2,1),(2,2),(2,3),(2,5),(3,2),(3,3),(3,4),(3,5),(4,3),(4,4),(4,5),(5,2),(5,3),(5,4),(5,5)\}$,

$\Pi_{\mathcal{A}} \circ \Pi_{\mathcal{A}} = \{(0,0),(0,1),(0,2),(0,3),(0,5),(1,1),(1,2),(1,3),(1,5),(2,2),(2,3),(2,5),(3,3),(4,3),(4,4),(4,5),(5,5)\}$ and

$\Pi^2_{\mathcal{A}} = \{(0,0),(0,1),(0,2),(0,3),(0,5),(1,1),(1,0),(1,2),(1,3),(1,5),(2,0),(2,1),(2,2),(2,3),(2,5),(3,2),(3,3),(3,4),(3,5),(4,3),(4,4),(4,5),(5,2),(5,3),(5,4),(5,5)\}$.

Any pair in $\Pi^2_{\mathcal{A}}$ represent states from which attackers can insert or erase events in a way that they can generate the same observation, thus making the states indistinguishable. □

The requirement of P-observability for an attack set is only relevant when dealing with strings $w, w' \in K$ that can generate the same observation after the attack and such that there exists an event $\sigma \in \Sigma$ such that $w\sigma \in K$ and $w'\sigma \in \mathcal{L}(G) \setminus K$ or $w'\sigma \in K$ and $w\sigma \in \mathcal{L}(G) \setminus K$. When $w\sigma \in \mathcal{L}(G) \setminus K$ or $w'\sigma \in \mathcal{L}(G) \setminus K$, it means that event σ should be disabled after w or w', respectively. In the same way, when $w\sigma \in K$ or $w'\sigma \in K$, it means that event σ should be enabled after w or w', respectively. Let $\xi : Q \to 2^{\Sigma}$ be a map defined as

$$\xi(q) := \{\sigma \in \Sigma | (\exists w \in \mathcal{L}(G))[\delta(q_0,w) = q \wedge w\sigma \in K]\} \tag{12}$$

that gives, for a state $q \in Q$, the set of *enabled* events at state q. Also, let the map $\phi : Q \to 2^{\Sigma}$ defined as

$$\phi(q) := \{\sigma \in \Sigma | (\exists w \in \mathcal{L}(G))[\delta(q_0,w) = q \wedge w\sigma \in \mathcal{L}(G) \setminus K]\} \tag{13}$$

be the map that gives, for a given state $q \in Q$, the set of *disabled* events at state q. Inspired by (Su and Wonham, 2004), where the authors present a binary relation of pairs of states that are consistent with respect to their control action and to their marking, we present the following definition.

Definition *8.* (Relation of control inconsistent states). The binary relation $\mathcal{I} \subseteq Q \times Q$ defined by

$$\mathcal{I} := \{(q,q') \in Q \times Q | \xi(q) \cap \phi(q') \neq \emptyset \vee \xi(q') \cap \phi(q) \neq \emptyset\} \tag{14}$$

is the relation of control inconsistent states.

In words, according to Definition 8, a pair of states is in the relation of control inconsistent states $\mathcal{I}$ if the control action at these two states is conflicting. It is important to notice that the relation $\mathcal{I}$ is not transitive but is symmetric. For testing P-observability for an attack set, it is not necessary to consider all pairs of strings $w, w' \in K$, but only the pairs such that their control action is conflicting. In other words, we need to consider only the states that are control inconsistent.

Example *9.* By comparing Figures 2 and 3, we can conclude that events A and R should be disabled at state 0, while they can occur at states 3 and 5, respectively. Thus, applying Def. 8, we obtain: $\mathcal{I} = \{(0,3),(0,5),(3,0),(5,0)\}$. □

Before presenting the main result of this work, we introduce Lemma 10.

Lemma *10.* Let $G = (Q, \Sigma, \delta, q_0)$ be an automaton, $q, q' \in Q$ be states such that $(q,q') \in \Pi_{\alpha}$ for some attacker $A_{\alpha} \in \mathcal{A}$ and $P : \Sigma^* \to \Sigma_o^*$ be a natural projection. Then, for all strings $w \in \mathcal{L}(G)$ and $v \in w(\alpha \cup \Sigma_{uo})^*$, such that $\delta(q_0,w) = q$ and $\delta(q_0,v) = q'$, it holds that

1) $v \in P^{-1}(A_{\alpha}P(w))$;
2) $w \in P^{-1}(A_{\alpha}P(v))$.

Proof The proof is omitted due to space limitation. □

Finally, we are able to present Theorem 11, which is our main result.

Theorem *11.* Let G be a deterministic finite automaton that represents the behavior of a system, K the desired language, with $K \subseteq \mathcal{L}(G)$. The set $\Sigma_{uo} \subseteq \Sigma$ is the set of unobservable events, $\Sigma_v \subseteq \Sigma$ is the set of vulnerable events ($\Sigma_{uo} \cap \Sigma_v = \emptyset$) and $\mathcal{A} = \{A_{\alpha_1}, A_{\alpha_2}, \ldots, A_{\alpha_M}\}$ is the attack set, with $\alpha_i \subseteq \Sigma_v$, $i = 1, \ldots, M$. Let $\mathcal{I}$ be the relation of control inconsistent states of the automaton that implements K. The language K will be P-observable for $\mathcal{A}$ and $\mathcal{L}(G)$ if and only if $\Pi^2_{\mathcal{A}} \cap \mathcal{I} = \emptyset$. □

Proof The proof is omitted due to space limitations. □

Theorem 11 states that a given desired language K is P-observable for an attack set $\mathcal{A}$ if and only if there are no mutual pairs of states between the relations $\Pi^2_{\mathcal{A}}$ and $\mathcal{I}$. One can apply Theorem 11 to verify if a language K is P-observable for an attack set $\mathcal{A}$ by checking if $(q,q') \notin \Pi^2_{\mathcal{A}}$ holds for every pair (q,q') in the relation of control inconsistent states $\mathcal{I}$.

Example *12.* From Examples 7 and 9, we have that $\Pi^2_{\mathcal{A}} \cap \mathcal{I} \neq \emptyset$, which allows us to conclude that the desired language of our problem is not P-observable for the attack set $\mathcal{A}$. Because of this, there are at least two strings, e.g., $w' = \varepsilon$ and $w'' = nbg$, such that they can be modified by two different attackers in a way that they will look alike to the supervisor. In such a case, if the supervisor

observes string $y = nb$, it is not sure if string nb really happened or if it was strings ε or even nbg, where events n and b could have been inserted by an attacker or event p could have been erased by other attacker, respectively. Note that string $y = n$ can also be the one that happened. However, since state 1 ($\delta(q_0, n) = \{1\}$) is not control inconsistent with any other state, considering this case does not contribute to our analysis. □

5. CONCLUSION

The increasing use of communication networks in control systems also increases the interfaces between devices and the outside world. These interfaces have vulnerabilities associated, which are entry points for malicious agents. This justifies the effort of providing reliable communication. Within the supervisory control context, one of these solutions is to design resilient supervisors, that can guarantee the legal behavior of the system, regardless of the attack.

For attacks in the output symbols, one of the conditions that allows robust supervisors to be designed is the P-observability for an attack set. Although a test for this property already existed (Wakaiki et al., 2019), it is based on a series of tests of the classical observability property. Our contribution is to introduce a new test, that checks for P-observability for an attack set itself and considers the effect of all attackers in a single run. We do not claim that our test is necessarily faster or more efficient than the previous one. Nevertheless we provided an approach that gives a better understanding about a recently described property, whose application is still incipient in the literature.

Additionally, despite the fact that our attack model may not seen very practical, since many assumptions are made, our results open new fronts for the research of more realistic models. For future works, we plan to investigate the effect of considering multiple attackers acting at once, as well as the effect of having them acting on events from supervisor to plant. Furthermore, we intend to provide algorithms that will allow us to apply our results using a computational tool.

ACKNOWLEDGMENTS

This work has been supported by the National Council for Scientific and Technological Development - CNPq under grant 443656/2018-5, CAPES, Brazil, Fapemig and PRPq-UFMG and the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

Carvalho, L.K., Wu, Y.C., Kwong, R., and Lafortune, S. (2018). Detection and Mitigation of Classes of Attacks in Supervisory Control Systems. *Automatica*, 97, 121–133.

Cassandras, C. and Lafortune, S. (2007). *Introduction to Discrete Event Systems*, volume 11. Springer, New York, 2nd edition.

Fritz, R. and Zhang, P. (2018). Modeling and detection of cyber attacks on discrete event systems. *IFAC-PapersOnLine*, 51(7), 285–290.

Gao, C., Seatzu, C., Li, Z., and Giua, A. (2019). Multiple Attacks Detection on Discrete Event Systems. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 2352–2357.

Khoumsi, A. (2019). Sensor and Actuator Attacks of Cyber-Physical Systems: A Study Based on Supervisory Control of Discrete Event Systems. In *Proceedings of the 8th International Conference on Systems and Control*, 176–182. IEEE.

Li, Y., Tong, Y., and Giua, A. (2020). Detection and Prevention of Cyber Attacks in Networked Control Systems. In *Proceedings of the 15th IFAC Workshop on Discrete Event Systems*, 7–13.

Lima, P.M., Alves, M.V., Carvalho, L.K., and Moreira, M.V. (2017). Security Against Network Attacks in Supervisory Control Systems. *IFAC-PapersOnLine*, 50(1), 12333–12338.

Lima, P.M., Carvalho, L.K., and Moreira, M.V. (2018). Detectable and Undetectable Network Attack Security of Cyber-physical Systems. *IFAC-PapersOnLine*, 51(7), 179–185.

Lima, P.M., Alves, M.V.S., Carvalho, L.K., and Moreira, M.V. (2019). Security Against Communication Network Attacks of Cyber-Physical Systems. *Journal of Control, Automation and Electrical Systems*, 30(1), 125–135.

Lin, L. and Su, R. (2021). Synthesis of covert actuator and sensor attackers. *Automatica*, 130(109714), 1–11.

Meira-Goes, R., Lafortune, S., and Marchand, H. (2021). Synthesis of Supervisors Robust Against Sensor Deception Attacks. *IEEE Transactions on Automatic Control*, 1–8.

Mohajerani, S., Góes, R.M., and Lafortune, S. (2020). Efficient Synthesis of Sensor Deception Attacks Using Observation Equivalence-Based Abstraction. In *Proceedings of the 15th IFAC Workshop on Discrete Event Systems*, 28–34.

Su, R. (2018). Supervisor synthesis to thwart cyber attack with bounded sensor reading alterations. *Automatica*, 94, 35–44.

Su, R. and Wonham, W.M. (2004). Supervisor Reduction for Discrete-Event Systems. *Discrete Event Dynamic Systems*, 14, 31–53.

Wakaiki, M., Tabuada, P., and Hespanha, J.P. (2019). Supervisory Control of Discrete-Event Systems Under Attacks. *Dynamic Games and Applications*, 9(4), 965–983.

Wang, W., Lafortune, S., and Lin, F. (2007). An Algorithm for Calculating Indistinguishable States and Clusters in Finite-State Automata with Partially Observable Transitions. *Systems and Control Letters*, 56(9-10), 656–661.

Wang, Y. and Pajic, M. (2019). Supervisory Control of Discrete Event Systems in the Presence of Sensor and Actuator Attacks. In *Proceedings of the 58th IEEE Conference on Decision and Control*, 5350–5355. IEEE.

Zhang, Q., Seatzu, C., Li, Z., and Giua, A. (2021). State Estimation Under Attack in Partially-Observed Discrete Event Systems. URL `https://arxiv.org/abs/1906.10207v3`.

Zhu, Y., Lin, L., and Su, R. (2019). Supervisor Obfuscation against Actuator Enablement Attack. In *Proceedings of the 18th European Control Conference*, 1760–1765.