Supervisory Control under Indefinite Actuator and Sensor Attacks: Prescribed Resilience

Shoma Matsui, Julien Bourgain-Wilbal, Yudai Saito,
Ziyue Ma, Gregory Faraut, Kai Cai, and Karen Rudie

***Abstract*— We study a new security problem in discrete-event systems with *indefinite* actuator and sensor attacks: Any controllable events disabled by the supervisor can be re-enabled, and any observable events can be concealed by attackers. We consider that the plant contains unsafe states, each of which is associated with a safety requirement specifying the number of attacks on actuators and the number of attacks on sensors that should be withstood by the supervisor to prevent the plant from entering the unsafe state. Thus, these safety requirements capture the *prescribed resilience* of the supervisor to be designed. To that end, we first construct an "unreliable plant" which includes all possible attacks in the plant, and derive a model of limited behavior of the unreliable plant according to the safety requirements, which is called an "attack-compromised plant". We then identify the unsafe states in the attack-compromised plant, and develop a synthesis procedure to compute a feasible partial-observation and *physically-admissible* supervisor (if one exists) with prescribed resilience. Physical admissibility is a novel property which indicates that no disabled event is subject to a sensor attack.**

***Index Terms*— Supervisory control, discrete-event systems, automata, security**

I. Introduction

Actuators and sensors in many industrial systems are among the primary targets for attackers as they are crucial interfaces to external environments, and may cause fatal incidents once they are compromised. Thus, it is a typical requirement for system designers to have controllers resilient against potential attacks on actuators and sensors. One challenge for system designers in creating resilient supervisor is that attack targets are usually unknown *a priori*.

In this paper, we model our target system as a *discrete-event system* (DES) [1], [2], and consider that the attacker's goal is to steer the system to its set of unsafe states. For example, an unsafe state can be the malfunction of production machines in a factory or a car collision at an intersection. In particular, we consider that the attacker may conduct two types of attacks against the system: *actuator attacks* and *sensor attacks*. Under actuator attacks, the attacker may enable any of the controllable events disabled by a supervisor so that the system may execute disabled events, disobeying the supervisor's intended control commands. On the other hand, sensor attacks disrupt the supervisor's observation by making observable events unobservable, thus rendering the supervisor no longer able to make decisions based on the partial observation.

Several attack schemes on actuators and sensors have been studied in the DES community. For example, in [3], [4], the plant is equipped with a security module to mitigate actuator attacks, and the attacker can enable several *vulnerable* actuator events. The main idea is that the security module disables all controllable events once it detects attacks. In [5], on the other hand, the security module only disables necessary controllable events to prevent damage. The work of [6] considers a situation where a subset of controllable events is subject to attack and the rest of the controllable events can be definitely disabled at high costs; the goal is to maximize the specification behavior while minimizing disablement of defensible events. Estimation and prevention of actuator attacks are addressed in [7]. Covert actuator attackers and the corresponding decidability problem are studied in [8].

For sensor attacks, the work of [9] introduced the system model in which the supervisor knows multiple observation maps possibly under attack. The work of [10], on the other hand, proposed a framework of sensor deception attack where the attacker has the same observability as the supervisor and can insert or delete a subset of observable events, called "attackable events". The subsequent work in [11] extended the synthesis methodology of robust supervisors in [10] to a bounded attack in which the attacker has limited time to edit sensor events. A further study of sensor deception attack is in [12]. Focusing on detecting sensor attacks, the authors of [13] introduced a joint state estimator which tracks all possible attacks and sets of state estimation. Besides the attack prevention, from the attacker's viewpoint, several works

S. Matsui is with SCREEN Semiconductor Solutions Co., Ltd., Kyoto, Japan. (s.matsui@screen.co.jp)

K. Rudie is with the Department of Electrical and Computer Engineering and Ingenuity Labs Research Institute, Queen's University, Kingston, Canada. (karen.rudie@queensu.ca)

J. B. Wilbal and G. Faraut are with the Department of Mechanical Engineering, University Paris-Saclay, ENS Paris-Saclay, France. (J. B. Wilbal: julien.bourgain--wilbal@ens-paris-saclay.fr, G. Faraut: gregory.faraut@ens-paris-saclay.fr)

Y. Saito and K. Cai are with the Department of Core Informatics, Osaka Metropolitan University, Osaka, Japan. (Y. Saito: sd24350m@st.omu.ac.jp, K. Cai: cai@omu.ac.jp)

Z. Ma is with the School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China. (maziyue@xidian.edu.cn)

S. Matsui and K. Rudie were supported by the Natural Sciences and Engineering Research Council of Canada under NSERC Discovery Grant RGPIN-2020-04279. Z. Ma was supported in part by National Natural Science Foundation of China (62373313), the Shaanxi Provincial Natural Science Foundation (2025JC-YBMS-658). Y. Saito and K. Cai were supported by JST ASPIRE Grant no. JPMJAP2519, JSPS KAKENHI Grant nos. 21H04875 and 22KK0155.

in [14], [15], [16], [17] addressed the synthesis of sensor deception strategies without tampering with the supervisor's observation, thus permitting *stealthy* attacks. In addition, a decidability problem of existence of resilient nonblocking supervisors is investigated in [18]. Sensor attacks in labeled Petri nets is also studied in [19]. More recently, methods for synthesizing a robust supervisor against both actuator and sensor attacks have been developed in [20], [21], [22], [23], [24]. Also see [25] for a review on this topic.

While most existing works of actuator and/or sensor attacks impose constraints on which events may be attacked, we consider by contrast that *all* controllable events and *all* observable events are subject to the actuator attack and the sensor attack, respectively. In other words, the attack model in this paper is *indefinite*. How to model the behavior of the plant under indefinite actuator/sensor attacks presents new complexity in supervisory control design, and most existing methods yield no solution. The work of [26] is the most relevant to this paper, as it first considered indefinite attacks on actuators, but did not consider indefinite sensor attacks.

As a main contribution of this paper, we extend the work of indefinite actuator attacks in [26] to both actuator and sensor attacks. That is, we consider a new security problem of synthesizing a supervisor which provides prescribed resilience against both indefinite actuator attacks and indefinite sensor attacks at the same time. Solving this problem enables system designers to build a supervisor based on the given demand of how resilient the supervisor should be without the need to know specific attacker policies such as where, when, and in which order to attack actuators and sensors; this is in sharp contrast to a supervisor which is resilient to only predetermined attack targets.

Extending the setting to indefinite actuator attacks in [26], we consider that the system designer is given *safety requirements* for both actuator and sensor attacks (respectively denoted by R_a and R_s) as prescribed resilience of the supervisor, which specifies the number of actuator/sensor attacks that a supervisor should endure.[1] This extended setting makes the supervisory control design for prescribed resilience significantly more challenging than that in [26], because not only actuator attacks can be either observable or unobservable (depending on whether or not the attacked controllable events are observable), but also the supervisor cannot have *a priori* knowledge of partial observation (since every observable event is subject to sensor attacks).

To address these challenges we develop several new concepts and a solution procedure as follows. First, we extend the attack model in [26] to encompass indefinite attacks on both actuator and sensors. In this model, we embed all possible attacks into the original plant (denoted by G) by introducing two types of pseudo-events: *actuator attack events* and *sensor attack events*, based respectively on subsets of controllable and observable events, and adding parallel transitions by these attack events to the original plant, resulting in an enhanced plant called an *unreliable plant* (denoted by $\widetilde{G}$). We then construct an "attack-compromised plant" (denoted by G_{com}) which is a model of limited behavior of the unreliable plant according to the safety requirements. We then formulate our problem using this extended attack model, and develop a new synthesis procedure to compute a supervisor (denoted by V if one exists) against both actuator and sensor attacks within the resilience prescribed by the given safety requirements. As the last and essential step of our procedure, we ensure that the resulting supervisor is *physically admissible* in that it makes consistent control decisions for pseudo-events and their corresponding original events.

The rest of this paper is organized as follows. In Section II we present notation and two main concepts of supervisory control theory of DES, namely controllability and observability, which we employ in this paper. Section III introduces notions and system models required to consider indefinite actuator and sensor attacks, and formulates the problem of resilient supervisory control. We then present in Section IV the procedure of designing supervisors with prescribed resilience, and we show that our procedure yields a solution whenever one exists. Finally, we conclude this paper in Section V.

II. Preliminaries

In supervisory control theory (SCT) of discrete-event systems (DES) [1], [27], a plant to be controlled is modeled as a finite-state automaton denoted by a 4-tuple

$$G = (Q, \Sigma, \delta, q_0) \tag{1}$$

where Q is a set of states, Σ is a set of events, $\delta : Q \times \Sigma \to Q$ is a (partial) transition function, and q_0 is the initial state. The transition function δ is extended to $\delta : Q \times \Sigma^* \to Q$ in the usual way (here Σ^* is the Kleene closure). We write $\delta(q, \omega)!$ to mean that $\delta(q, \omega)$ is defined. We denote the length and prefix-closure of a string ω by $|\omega|$ and $\overline{\omega}$, respectively. Moreover, the language generated by a plant G in (1) is denoted by $L(G)$, which is defined as $L(G) \coloneqq \{\omega \in \Sigma^* : \delta(q_0, \omega) \in Q\}$. The prefix-closure of a language $L \subseteq \Sigma^*$ is denoted by $\overline{L}$ which is defined by $\overline{L} \coloneqq \{\omega \in \Sigma^* : (\exists \omega' \in \Sigma^*)\, \omega\omega' \in L\}$. The language L is said to be *prefix-closed* if $L = \overline{L}$. Note that $L(G) = \overline{L(G)}$ by construction. Consider two automata $G_1 = (Q_1, \Sigma_1, \delta_1, q_{0,1})$ and $G_2 = (Q_2, \Sigma_2, \delta_2, q_{0,2})$. We denote by $G_1 \parallel G_2$ their *parallel composition* (cf. [2]).

To represent the ability of a supervisor to control and observe certain events in the plant G, the set of events Σ is partitioned into *controllable* and *uncontrollable* event subsets (denoted by Σ_c and Σ_{uc}), or *observable* and *unobservable* events (denoted by Σ_o and Σ_{uo}). That is, for a plant G, we have that $\Sigma = \Sigma_c \dot\cup \Sigma_{uc}$ and $\Sigma = \Sigma_o \dot\cup \Sigma_{uo}$. A *natural projection* $P : \Sigma^* \to \Sigma_o^*$ is defined recursively by

$$P(\varepsilon) \coloneqq \varepsilon$$
$$P(\sigma) \coloneqq \begin{cases} \sigma & \text{if } \sigma \in \Sigma_o \\ \varepsilon & \text{if } \sigma \in \Sigma_{uo} \end{cases}$$
$$P(\omega\sigma) \coloneqq P(\omega)P(\sigma) \text{ for } \omega \in \Sigma^* \text{ and } \sigma \in \Sigma.$$

[1] In practice, the safety requirements R_a and R_s are best determined based on historical attack statistics and feasible windows for applying emergency measures or human interventions. In this way, even if the number of attacks may exceed R_a/R_s, the supervisor with prescribed resilience can be used as a trigger (with R_a/R_s as thresholds) so as to apply an emergency action to reset or reconfigure the system.

Thus the natural projection P removes unobservable events from a given string, which can be extended to $P : 2^{\Sigma^*} \to 2^{\Sigma_o^*}$ by $P(L) \coloneqq \bigcup_{\omega \in L} P(\omega)$ for a language $L \subseteq \Sigma^*$.

The central notions of SCT are *controllability* and *observability*, which are specific conditions for a given language.

Definition 1 (Controllability)**.** (cf. [1]) Given a prefix-closed language $L \subseteq \Sigma^*$ and a set of uncontrollable events Σ_{uc}, a prefix-closed language $M \subseteq \Sigma^*$ is said to be *controllable with respect to* L *and* Σ_{uc} if $M\Sigma_{uc} \cap L \subseteq M$.

Definition 2 (Observability)**.** (cf. [1]) Given a prefix-closed language $L \subseteq \Sigma^*$ and a natural projection $P : \Sigma^* \to \Sigma_o^*$, a prefix-closed language $M \subseteq \Sigma^*$ is said to be *observable with respect to* L and P if for all $\omega, \omega' \in M$ and $\sigma \in \Sigma_c$,

$$P(\omega) = P(\omega') \wedge \omega\sigma \in M \wedge \omega' \in M \wedge \omega'\sigma \in L \Rightarrow \omega'\sigma \in M.$$

Roughly speaking, the controllability condition in Definition 1 states that for every string in M, its one-step continuation via an arbitrary uncontrollable event (allowed by L) remains in M. The observability condition, on the other hand, requires that for any two lookalike strings in M, control decisions after those strings should be the same, i.e., there should be no control conflict for lookalike strings.

A *partial-observation supervisor* in SCT is a mapping $V : P(L(G)) \to 2^{\Sigma_c}$ that prescribes which controllable events in Σ_c to disable after the observation of strings in $P(L(G))$. The result of supervisory control is represented by a generated language of G under control of V, denoted by $L(V/G)$, which is recursively defined as follows [2]:

1. $\varepsilon \in L(V/G)$
2. $(\forall \sigma \in \Sigma)\, \omega \in L(V/G) \wedge \omega\sigma \in L(G) \wedge \sigma \notin V(P(\omega)) \Leftrightarrow \omega\sigma \in L(V/G)$

A feasible partial-observation supervisor V must make the same control decision for strings in the plant that look the same to V, i.e., for strings $\omega, \omega' \in L(G)$, if $P(\omega) = P(\omega')$ then $V(\omega) = V(\omega')$. We say that V synthesizes the language L_K if $L(V/G) = L_K$. It has been shown that there exists a feasible supervisor V (with respect to $L(G)$) that synthesizes L_K if and only if L_K is controllable with respect to $L(G)$ and Σ_c, and observable with respect to $L(G)$ and P [28]. Since the mapping V can be represented by an automaton (called a *realization* [2]), we henceforth refer to V and its automaton representation interchangeably as "supervisor".

III. Problem Formulation

In this section, we formulate the problem of synthesizing resilient supervisors against indefinite actuator and sensor attacks. To this end, we first introduce the setting and modeling of indefinite attacks. Our aim is to synthesize a supervisor V for the original plant G, where V satisfies the resilience requirement that an imposed specification is enforced if no more than prescribed numbers of actuator/sensor attacks occur. Thus we need a representation for "an event in G has been attacked" and thereby construct an augmented plant which captures the behavior of the original plant whose actuators and sensors are subject to attacks.

A. Attack Events and Unreliable Plant

To model the system's behavior under indefinite attacks, we introduce two new types of event subsets, *actuator attack events* Σ_a and *sensor attack events* Σ_s, which represent the attacker's actions of actuator attack and sensor attack, respectively. Events in Σ_a and Σ_s will be used to count, respectively, the numbers of indefinite actuator and sensor attacks. Specifically, we define Σ_a and Σ_s by adding a subscript a to every controllable event in Σ_c and a subscript s to every observable event in Σ_o, i.e.,

$$\Sigma_a = \{\sigma_a : \sigma \in \Sigma_c\} \tag{2}$$

$$\Sigma_s = \{\sigma_s : \sigma \in \Sigma_o\}. \tag{3}$$

The actuator attack events in Σ_a correspond to the events in Σ_c that may be enabled by the attacker, while the sensor attack events in Σ_s to those in Σ_o that may be concealed by the attacker. Thus, all events in Σ_a are uncontrollable, and all events in Σ_s are unobservable. We emphasize that these special events are to model the attacker's actions, and do not exist in the original plant. In other words, events in Σ_a and Σ_s are "pseudo"-events.

Each of Σ_a and Σ_s can be further partitioned into two subsets as follows:

$$\Sigma_{o,a} = \{\sigma_a : \sigma \in \Sigma_c \cap \Sigma_o\}$$
$$\Sigma_{uo,a} = \{\sigma_a : \sigma \in \Sigma_c \cap \Sigma_{uo}\}$$
$$\Sigma_{c,s} = \{\sigma_s : \sigma \in \Sigma_c \cap \Sigma_o\}$$
$$\Sigma_{uc,s} = \{\sigma_s : \sigma \in \Sigma_{uc} \cap \Sigma_o\}$$

where $\Sigma_{o,a}$, $\Sigma_{uo,a}$, $\Sigma_{c,s}$, and $\Sigma_{uc,s}$ are called subsets of *observable actuator attack events*, *unobservable actuator attack events*, *controllable sensor attack events*, and *uncontrollable sensor attack events*, respectively. As subsets of Σ_a, events in $\Sigma_{o,a}$ and $\Sigma_{uo,a}$ are uncontrollable; and as subsets of Σ_s, events in $\Sigma_{c,s}$ and $\Sigma_{uc,s}$ are unobservable. Table I summarizes the controllability and observability status of events in these subsets. Each attack event represents either an actuator attack or a sensor attack. Since the observability status of events in Σ_o is not affected by actuator attacks, and the controllability status of events in Σ_c is not affected by sensor attacks, attack events in $\Sigma_{o,a}$ are observable, and those in $\Sigma_{c,s}$ are controllable. The reason why the events in $\Sigma_{c,s}$ can be treated as controllable is because their corresponding original events are controllable and thus can be prevented from occurring; if the original events are disabled and do not even occur, there cannot be any sensor attacks, which means effectively that the corresponding pseudo-events in $\Sigma_{c,s}$ cannot occur. In this sense, we categorize events in $\Sigma_{c,s}$ as controllable events.

Subset	Controllable	Observable
$\Sigma_{o,a}$	✗	✓
$\Sigma_{uo,a}$	✗	✗
$\Sigma_{c,s}$	✓	✗
$\Sigma_{uc,s}$	✗	✗

TABLE I: Controllability/observability status of attack events

With the introduced attacker events, we encode all possibilities of the system's behavior under indefinite attacks. The

idea is to add parallel transitions of attack events to those labeled by their corresponding original events. We call such an attack-encoded plant an *unreliable plant*.

Definition 3 (Unreliable Plant)**.** Given a plant $G = (Q, \Sigma, \delta, q_0)$ in (1), Σ_a in (2), and Σ_s in (3), an unreliable plant is a four-tuple

$$\widetilde{G} = (Q, \widetilde{\Sigma}, \widetilde{\delta}, q_0) \tag{4}$$

where

$$\begin{aligned}\widetilde{\Sigma} &= \Sigma \,\dot\cup\, \Sigma_a \,\dot\cup\, \Sigma_s \\ \widetilde{\delta} &= \delta \cup \{(q, \sigma_a, q') : \delta(q, \sigma) = q' \wedge \sigma_a \in \Sigma_a \wedge \sigma \in \Sigma_c\} \\ &\quad \cup \{(q, \sigma_s, q') : \delta(q, \sigma) = q' \wedge \sigma_s \in \Sigma_s \wedge \sigma \in \Sigma_o\}.\end{aligned} \tag{5}$$

The transition function $\widetilde{\delta}$ in (5) indicates that we add duplicated transitions labeled by the events $\sigma_a \in \Sigma_a$ and $\sigma_s \in \Sigma_s$, wherever their corresponding events are defined by the original transition function δ. This also reflects that we do not know how powerful attackers may be *a priori*.

For convenience, we henceforth denote the subset of controllable events in $\widetilde{G}$ (namely $\Sigma_c \cup \Sigma_{c,s}$) by $\widetilde{\Sigma}_c$, the subset of uncontrollable events in $\widetilde{G}$ (namely $\Sigma_{uc} \cup \Sigma_{uc,s} \cup \Sigma_a$) by $\widetilde{\Sigma}_{uc}$, the subset of observable events in $\widetilde{G}$ (namely $\Sigma_o \cup \Sigma_{o,a}$) by $\widetilde{\Sigma}_o$, and the subset of unobservable events in $\widetilde{G}$ (namely $\Sigma_{uo} \cup \Sigma_s \cup \Sigma_{uo,a}$) by $\widetilde{\Sigma}_{uo}$. In summary,

$$\begin{aligned}\widetilde{\Sigma}_c &= \Sigma_c \cup \Sigma_{c,s} \\ \widetilde{\Sigma}_{uc} &= \Sigma_{uc} \cup \Sigma_{uc,s} \cup \Sigma_a \\ \widetilde{\Sigma}_o &= \Sigma_o \cup \Sigma_{o,a} \\ \widetilde{\Sigma}_{uo} &= \Sigma_{uo} \cup \Sigma_s \cup \Sigma_{uo,a}.\end{aligned}$$

B. Safety Requirements

Consider a plant $G = (Q, \Sigma, \delta, q_0)$ as in (1), and a subset $Q_u \subseteq Q$ of *unsafe states* to be avoided. For an unsafe state $q_u \in Q_u$, a *safety requirement* on q_u is expressed by a pair of numbers $(R_a(q_u), R_s(q_u))$, which requires that even if there are up to $R_a(q_u)$ actuator attacks and $R_s(q_u)$ sensor attacks on every path from the initial state to the unsafe state q_u, this state q_u should still be avoided. Different unsafe states in Q_u may have different "danger levels", and thus may be associated with different safety requirements in general.

Formally, we define two functions: an *actuator safety requirement* $R_a : Q_u \to \mathbb{N}_0$ and a *sensor safety requirement* $R_s : Q_u \to \mathbb{N}_0$ as the required resilience against actuator attacks and sensor attacks, respectively (where $\mathbb{N}_0$ denotes the set of non-negative integers). These two functions assign two non-negative integers to each unsafe state $q_u \in Q_u$, which represent upper bounds of actuator and sensor attacks the supervisor should be resilient against. We now provide an example to illustrate the concepts introduced so far.

Example 1. Consider the plant G in Figure 1. This plant consists of seven states, namely $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$, and q_4 and q_5 are unsafe states, i.e., $Q_u = \{q_4, q_5\}$. The set of controllable events is $\Sigma_c = \{\sigma_1, \sigma_3\}$ and the set of observable events is $\Sigma_o = \{\sigma_1, \sigma_2, \sigma_3\}$. From Σ_c and Σ_o, the sets of actuator attack events and sensor attack events are $\Sigma_a = \{\sigma_{1,a}, \sigma_{3,a}\}$ and $\Sigma_s = \{\sigma_{1,s}, \sigma_{2,s}, \sigma_{3,s}\}$, respectively. Note that σ_4 is neither controllable nor observable; hence, it is not subject to attacks.

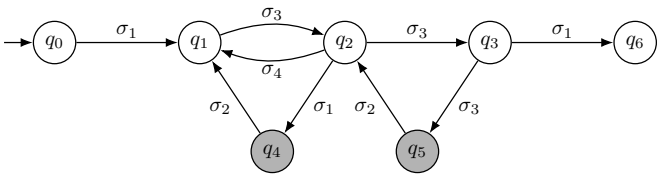


Fig. 1: Example plant G

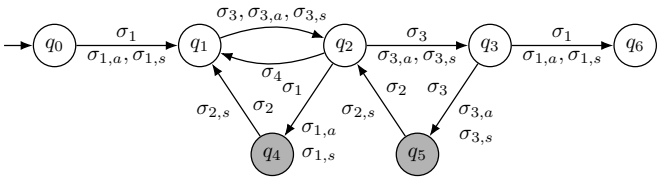


Fig. 2: Example unreliable plant $\widetilde{G}$

Following Definition 3, the unreliable plant $\widetilde{G}$ of G in Figure 1 is depicted in Figure 2. The additional transitions in $\widetilde{G}$ labeled by events in Σ_a and Σ_s are uncontrollable and unobservable, respectively.

To analyze how the attacker can affect the supervisor, let us first examine a special case where no attacks are considered, namely $R_a(q_4) = R_a(q_5) = R_s(q_4) = R_s(q_5) = 0$. This is in fact the case of conventional supervisory control. By inspecting the unreliable plant $\widetilde{G}$ in Figure 2, it can be verified that the supervisor in this case only disables σ_1 at q_2 and σ_3 at q_3.

Next, let us examine the case where $R_a(q_4) = R_a(q_5) = 2$ and $R_s(q_4) = R_s(q_5) = 0$, i.e., only actuator attacks are considered. This is a situation that the previous work [26] addresses. In this case, it can be seen in Figure 2 that σ_1 at q_0 and σ_3 at q_1 need to be disabled to eliminate other possibilities than the attacker enables σ_1 at q_0 and σ_3 at q_1, that is, string $\sigma_{1,a}\sigma_{3,a}$ occurs. This is because by $R_a(q_4) = 2$, we consider up to two actuator attacks for unsafe state q_4, and if either σ_1 at q_0 or σ_3 at q_1 occurs without attack, then the attacker can forcibly enable σ_1 at q_2, resulting in the plant which uncontrollably reaches unsafe state q_4. After observing $\sigma_{1,a}\sigma_{3,a}$, the supervisor disables σ_1 at q_2 while it enables σ_1 at q_3, since strings $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_3$ are distinguishable and the supervisor considers that no more attacks may occur (i.e., $\sigma_{3,s}$ at q_2 is not supposed to happen in this case).

Finally, let us introduce a sensor attack to this example by considering the sensor safety requirement $R_s(q_5) = 1$ (in addition to $R_s(q_4) = 0$ and $R_a(q_4) = R_a(q_5) = 2$), that is, the attacker can conduct up to one sensor attack. Given that σ_3 is observable and we consider indefinite attacks, it is possible that the attacker conceals σ_3 at q_2 after $\sigma_{1,a}\sigma_{3,a}$, i.e., $\sigma_{1,a}\sigma_{3,a}\sigma_{3,s}$ occurs in Figure 2. In such a case, the supervisor which disables σ_1 at q_2 after string $\sigma_{1,a}\sigma_{3,a}$ while enabling σ_1 at q_3 after string $\sigma_{1,a}\sigma_{3,a}\sigma_{3,s}$ is no longer feasible, since $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_{3,s}$ look the same from the supervisor's

observation. As a result, the supervisor also needs to disable σ_1 at q_3 after string $\sigma_{1,a}\sigma_{3,a}\sigma_{3,s}$.[1]

This example illustrates that even one sensor attack could make the supervisor infeasible (if it is resilient only against actuator attacks), which cannot be addressed by the method in [26]. This motivates us to take sensor attacks into account in addition to actuator attacks, and develop a new solution for this more general and challenging case.

C. Resilient Supervisory Control Problem

In this subsection, we formulate our problem of designing supervisors satisfying prescribed resilience for the unreliable plant $\widetilde{G}$ in (4) introduced in Section III-A. Note however that strings in $L(\widetilde{G})$ contain arbitrary numbers of actuator/sensor attacks. Not all of these strings need to be considered, because our target supervisor is required to endure numbers of actuator/sensor attacks *no more than* the prescribed safety requirements R_a and R_s introduced in Section III-B. Thus, in the following, we first extract the subset of behavior of the unreliable plant $\widetilde{G}$ according to R_a and R_s. For convenience, we denote natural projections extracting actuator attack events and sensor attack events from a given string in $L(\widetilde{G})$ by $P_a : \widetilde{\Sigma}^* \to \Sigma_a^*$ and $P_s : \widetilde{\Sigma}^* \to \Sigma_s^*$, respectively.

Definition 4 (Attack-Compromised Language)**.** Consider an unreliable plant $\widetilde{G} = (Q, \widetilde{\Sigma}, \widetilde{\delta}, q_0)$ in (4), a set of unsafe states $Q_u \subseteq Q$, an actuator safety requirement R_a, and a sensor safety requirement R_s. The *attack-compromised* sublanguage of $L(\widetilde{G})$ is defined as follows[2]:

$$\begin{aligned} L_{com}(\widetilde{G}, Q_u, R_a, R_s) \coloneqq \\ \big\{\omega \in L(\widetilde{G}) : |P_a(\omega)| \le r_a \wedge |P_s(\omega)| \le r_s \\ \wedge \big((\forall t \in \overline{\omega})(\forall q_u \in Q_u)\, \widetilde{\delta}(q_0, t) = q_u \Rightarrow \\ |P_a(t)| \le R_a(q_u) \wedge |P_s(t)| \le R_s(q_u)\big)\big\} \end{aligned} \quad (6)$$

where $r_a = \max_{q_u \in Q_u} R_a(q_u)$ and $r_s = \max_{q_u \in Q_u} R_s(q_u)$.

We write $L_{com}(\widetilde{G}, Q_u, R_a, R_s)$ to stress its dependence on $\widetilde{G}, Q_u, R_a, R_s$; but when these are clear from the context, we refer to $L_{com}(\widetilde{G}, Q_u, R_a, R_s)$ simply as L_{com}.

In (6), we first limit by "$|P_a(\omega)| \le r_a \wedge |P_s(\omega)| \le r_s$" the numbers of actuator and sensor attacks in each string $\omega \in L(\widetilde{G})$ to the maximum values of the respective safety requirements, since we do not need to consider the unreliable plant's behavior when the numbers of actuator/sensor attacks are beyond the maximum safety requirements. Then, we further limit by the condition in the third and forth lines of (6) the number of actuator and sensor attacks in each string ω that either reaches or passes an unsafe state q_u to the numbers $R_a(q_u)$ and $R_s(q_u)$, respectively.

[1]If q_6 represents a "desired" state in the system (i.e., a marked state in DES), one could interpret that even one sensor attack can cause somewhat catastrophic results, although we do not consider marked states in this paper and thus disabling σ_1 at q_3 merely results in possibly more conservative behavior of the system.

[2]This definition, in fact, reflects that once G reaches an unsafe state, we are not required to prevent G from reaching further unsafe states. This can be seen in many traditional supervisory control problems of DES. However, future work could include exploring how to mitigate damage or recover from damage once the systems reaches or passes through an unsafe state.

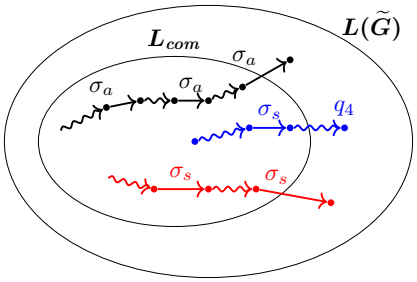


Fig. 3: Visualization of $L(\widetilde{G})$ and L_{com}; symbol $\rightsquigarrow$ represents arbitrary strings without attack events (namely strings in Σ^*), and each dot represents a state in $\widetilde{G}$ (which is arbitrary except one specified as q_4), so q_4 above a dot on the blue line indicates that the blue string reaches q_4. We abuse notation here and use the labels σ_a and σ_s to indicate some actuator or sensor attack; i.e., σ_a and σ_s can represent different actuator/sensor attack events.

Example 2. Consider again the example $\widetilde{G}$ in Figure 2, $R_a(q_4) = R_a(q_5) = 2$, $R_s(q_4) = 0$, and $R_s(q_5) = 1$. Figure 3 depicts a conceptional diagram of $L(\widetilde{G})$ and L_{com}. The arrows in black color illustrate a string going outside L_{com} by "$|P_a(\omega)| \le r_a$" where $r_a = \max_{q_u \in \{q_4, q_5\}} R_a(q_u) = 2$, since it contains more than two actuator attack events in Σ_a and thus does not need to be considered. The arrows in blue color show a string going outside L_{com} by the third and forth lines of (6), since it reaches unsafe state q_4 with $R_s(q_4) = 0$ but contains at least one sensor attack event σ_s. The arrows in red color indicate a string going outside L_{com} due to "$|P_s(\omega)| \le r_s$" where $r_s = \max_{q_u \in \{q_4, q_5\}} R_s(q_u) = 1$, since it contains more than one sensor attack event σ_s.

Thus, to construct L_{com} from $\widetilde{G}$ in Figure 2, we remove from $L(\widetilde{G})$ all strings containing more than two actuator attack events and/or more than one sensor attack events, and strings which reach q_4 and contain at least one actuator attack event. (Note that $R_a(q_4) = R_a(q_5) = r_a = 2$ and $R_s(q_5) = r_s = 1$, i.e., three of the four resilience specifications are met by ensuring that no more than the maximum numbers of attacks, namely, r_a and r_s.)

Remark 1. If R_a (resp., R_s) has the same value for all unsafe states in Q_u, then "$|P_a(\omega)| \le r_a$" and "$|P_a(t)| \le R_a(q_u)$" (resp., "$|P_s(\omega)| \le r_s$" and "$|P_s(t)| \le R_s(q_u)$") in (6) yield the same result, since $R_a(q_u) = r_a$ and $R_s(q_u) = r_s$ for all $q_u \in Q_u$. In such a special case, we can simplify (6) as

$$\begin{aligned} L_{com}(\widetilde{G}, Q_u, r_a, r_s) \coloneqq \\ \big\{\omega \in L(\widetilde{G}) : |P_a(\omega)| \le r_a \wedge |P_s(\omega)| \le r_s\big\} \end{aligned}$$

where r_a and r_s are the safety requirements identical for all unsafe states.

Now that we have defined $L_{com} \subseteq L(\widetilde{G})$ to be the subset of the unreliable plant's behavior that should our supervisor design be focused on according to the prescribed resilience, we next define the specification language (denoted by L_K) by excluding from L_{com} those strings which reach or pass an unsafe state in Q_u, i.e.,

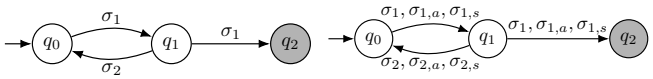
$$L_K \coloneqq L_{com} \setminus \{\omega \in L_{com} : (\exists t \in \overline{\omega})\, \widetilde{\delta}(q_0, t) \in Q_u\}. \quad (7)$$

Moreover, we consider a partial-observation supervisor V for the unreliable plant $\widetilde{G}$. Recall from Section III-A that the controllable events in the unreliable plant $\widetilde{G}$ are those original ones in Σ_c and those pseudo-events in $\Sigma_{c,s}$. Thus, supervisor V is a mapping $V : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$, where $P : \widetilde{\Sigma}^* \to \widetilde{\Sigma}_o^*$. However, the control decision of V must be consistent for each original-pseudo pair $(\sigma, \sigma_s) \in (\Sigma_c \cap \Sigma_o, \Sigma_{c,s})$. This is because if a controllable event $\sigma \in \Sigma_c \cap \Sigma_o$ is disabled by V, then σ cannot even occur and a sensor attack on σ is impossible (i.e., σ_s cannot occur); on the other hand, if σ is not disabled by V, then a sensor attack on σ is possible (i.e., σ_s can occur). In fact, supervisor V cannot physically enable/disable $\sigma_s \in \Sigma_{c,s}$, but can equivalently do so by enabling/disabling the corresponding original $\sigma \in \Sigma_c \cap \Sigma_o$. Therefore, to ensure consistency of the control decisions, we introduce a new concept of *physically-admissible* partial-observation supervisor $V : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ as follows.

Definition 5 (Physically Admissible Partial-Observation Supervisor)**.** Given an unreliable plant $\widetilde{G}$ and a natural projection $P : \widetilde{\Sigma}^* \to \widetilde{\Sigma}_o^*$, a partial-observation supervisor $V : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ is said to be *physically admissible* if

$$(\forall \omega \in L(\widetilde{G}), \forall \sigma \in \Sigma_c \cap \Sigma_o)\, \sigma \in V(P(\omega)) \Leftrightarrow \sigma_s \in V(P(\omega)) \tag{8}$$

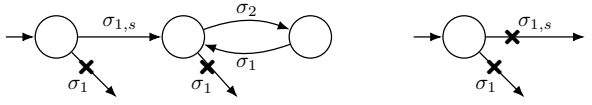
Condition (8) requires that if event σ in the original plant is disabled, then sensor attack event σ_s corresponding to σ is also disabled, and vice versa. This ensures that no disabled event is subject to a sensor attack, by applying consistent control decisions to original events and sensor attack events.



(a) Example plant G_2 (b) Unreliable plant $\widetilde{G}_2$

Fig. 4: Example plant G_2 and unreliable plant $\widetilde{G}_2$ for the illustration of physically admissible partial-observation supervisor

Example 3. To illustrate Definition 5, let us consider an example plant in Figure 4a. Suppose that $\Sigma = \Sigma_c = \Sigma_o = \{\sigma_1, \sigma_2\}$ (i.e., all events are both controllable and observable) and $Q_u = \{q_2\}$ which gives us the unreliable plant $\widetilde{G}_2$ in Figure 4b. According to Table I, in $\widetilde{G}_2$, events $\sigma_1, \sigma_{1,a}, \sigma_2$ and $\sigma_{2,a}$ are observable, events $\sigma_{1,s}$ and $\sigma_{2,s}$ are unobservable, events $\sigma_1, \sigma_{1,s}, \sigma_2$, and $\sigma_{2,s}$ are controllable, and events $\sigma_{1,a}$ and $\sigma_{2,a}$ are uncontrollable.



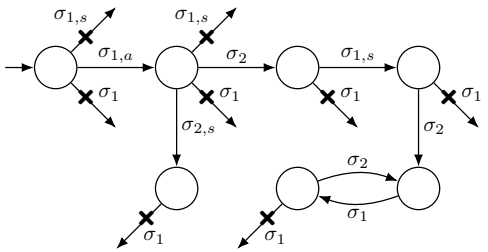
(a) Partial-observation supervisor (not physically admissible) (b) Partial-observation supervisor (physically admissible)

Fig. 5: Partial-observation supervisors based on G_2 and $\widetilde{G}_2$ in Figure 4 with $R_a(q_2) = 0$ and $R_s(q_2) = 1$; symbol ✖ on edges represents the supervisor's decision to disable events.

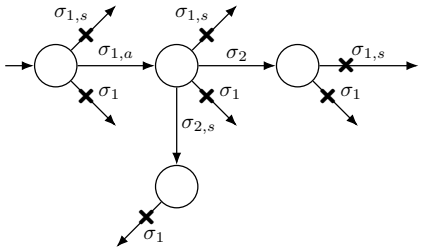
Let us first consider a special case where $R_a(q_2) = 0$ and $R_s(q_2) = 1$ for $\widetilde{G}_2$ (Figure 4b with $\sigma_{1,a}, \sigma_{2,a}$ removed), i.e., no actuator attacks are considered, while one sensor attack should be tolerated. In this case, consider a partial-observation supervisor in Figure 5a. First, at state q_0 of $\widetilde{G}_2$, the sensor attack event $\sigma_{1,s}$ is allowed since $R_s(q_2) = 1$ means that at least one sensor attack can be tolerated. If the sensor attack event $\sigma_{1,s}$ occurs from state q_0 of $\widetilde{G}_2$, the event σ_1 must be disabled. This is because the sequence $\sigma_{1,s}\sigma_1$ drives the unreliable plant $\widetilde{G}_2$ to the unsafe state q_2. Note, though, that while the string $\sigma_{1,s}\sigma_{1,s}$ also leads $\widetilde{G}_2$ to unsafe state q_2, our supervisor need not disable $\sigma_{1,s}$ at state q_1 of $\widetilde{G}_2$. The reason the supervisor may allow $\sigma_{1,s}$ to occur at this point is because the assumption that $R_s(q_2) = 1$ means that if $\sigma_{1,s}$ occurs once, we can essentially assume it does not occur again because although in the real system it may occur a second time, we guarantee a correct solution only for the situations in which there are no more than the prescribed number of sensor attacks given by R_s.

We also see that the supervisor in Figure 5a disables σ_1 at the initial state q_0 of $\widetilde{G}$. This is because σ_1 is disabled after $\sigma_{1,s}$ occurs (as reasoned above) but the strings ε and $\sigma_{1,s}$ look the same to the supervisor (since $\sigma_{1,s}$ is unobservable) so the supervisor has to make the same control decisions after seeing nothing and after seeing $\sigma_{1,s}$.

Here, it can be seen that the supervisor in Figure 5a is not physically admissible, since it enables $\sigma_{1,s}$ but disables σ_1 at q_0 of $\widetilde{G}_2$, violating condition (8) for $\omega = \varepsilon$ and $\sigma = \sigma_1$. We can, however, construct a physically-admissible (partial-observation) supervisor in Figure 5b based on Figure 5a by disabling $\sigma_{1,s}$ at q_0 of $\widetilde{G}_2$. Thus, we see that physical inadmissibility can arise as long as there is a sensor attack (even if there is no actuator attack). This issue is therefore distinctive from the previous work [26], and a new technique will be developed to ensure physical admissibility.



(a) Partial-observation supervisor (not physically admissible)



(b) Partial-observation supervisor (physically admissible)

Fig. 6: Partial-observation supervisors based on G_2 and $\widetilde{G}_2$ in Figure 4 with $R_a(q_2) = R_s(q_2) = 1$; symbol ✖ on edges represents the supervisor's decision to disable events.

Next, let us consider both actuator attacks and sensor attacks

by letting $R_a(q_2) = 1$, $R_s(q_2) = 1$ for $\widetilde{G}_2$ in Figure 4b. In this case, consider a partial-observation supervisor in Figure 6a. First, this supervisor must disable σ_1 and $\sigma_{1,s}$ at q_0 of $\widetilde{G}_2$, as allowing either event would lead $\widetilde{G}_2$ to q_1, from which a single attack ($\sigma_{1,a}$) would drive $\widetilde{G}_2$ to the unsafe state q_2, which would not be acceptable since $R_a(q_2) = 1$. These scenarios would allow $\sigma_1\sigma_{1,a}$ or $\sigma_{1,s}\sigma_{1,a}$ to occur, both of which would lead $\widetilde{G}_2$ to unsafe state q_2. Thus, at q_0 of $\widetilde{G}_2$, only $\sigma_{1,a}$ may occur. This situation indicates that the supervisor sometimes enforces a seemingly counterintuitive strategy of allowing an attack early on (in this case, allowing $\sigma_{1,a}$ to occur at q_0 of $\widetilde{G}_2$)—as long as it does not lead immediately to an unsafe state (in this case, q_2)—setting a sort of honey trap, so that the attacker will use up its allotted attack resources (in this case, captured by $R_a(q_2) = 1$) while the supervisor is then able to later on prevent entry into an unsafe state.

If $\sigma_{1,a}$ occurs at q_0 of $\widetilde{G}_2$, the supervisor again disables σ_1 and $\sigma_{1,s}$ to prevent the plant from reaching unsafe state q_2. At this point, possible events at q_1 of $\widetilde{G}_2$ are σ_2 and $\sigma_{2,s}$. Since $\sigma_{2,s}$ is unobservable and $\sigma_{1,a}$ and $\sigma_{1,a}\sigma_{2,s}$ look the same to the supervisor, σ_1 after $\sigma_{1,a}\sigma_{2,s}$ needs to be disabled because σ_1 is disabled after $\sigma_{1,a}$ (as we just reasoned).

Now, recall that the supervisor only needs to make disablement decisions for strings which contain no more than the prescribed number of attacks, namely no more than one actuator attack and one sensor attack due to $R_a(q_2) = R_s(q_2) = 1$ in this case. Consequently, the supervisor can enable $\sigma_{1,s}$ after $\sigma_{1,a}\sigma_2$, because the supervisor need not consider that $\sigma_{1,s}$ can occur after $\sigma_{1,a}\sigma_2$. However, once $\sigma_{1,s}$ after $\sigma_{1,a}\sigma_2$ is enabled, the supervisor in Figure 6a would need to disable σ_1 after $\sigma_{1,a}\sigma_2$, because unobservable event $\sigma_{1,s}$ can now occur after $\sigma_{1,a}\sigma_2$, but strings $\sigma_{1,a}\sigma_2$ and $\sigma_{1,a}\sigma_2\sigma_{1,s}$ look the same to the supervisor; hence σ_1 needs to be disabled after $\sigma_{1,a}\sigma_2\sigma_{1,s}$ to prevent $\widetilde{G}_2$ from reaching unsafe state q_2. Since the supervisor does not take into account any further attacks after string $\sigma_{1,a}\sigma_2\sigma_{1,s}$ and after strings in $\sigma_{1,a}\sigma_2\sigma_{1,s}\sigma_2\sigma_1(\sigma_2\sigma_1)^*$, this means that after $\sigma_{1,a}\sigma_2\sigma_{1,s}$ and after strings in $\sigma_{1,a}\sigma_2\sigma_{1,s}\sigma_2\sigma_1(\sigma_2\sigma_1)^*$, the supervisor need only to disable σ_1, which corresponds to disabling σ_1 at q_1 in $\widetilde{G}_2$, to prevent $\widetilde{G}_2$ from reaching q_2.

It can be seen that the supervisor in Figure 6a is again not physically admissible, since it disables σ_1 and enables $\sigma_{1,s}$ after string $\sigma_{1,a}\sigma_2$, thereby violating condition (8) for $\omega = \sigma_{1,a}\sigma_2$ and $\sigma = \sigma_1$. We can, however, construct a physically-admissible (partial-observation) supervisor in Figure 6b based on Figure 6a by further disabling $\sigma_{1,s}$ after $\sigma_{1,a}\sigma_2$.

Having introduced and motivated the need to address physical admissibility, we are now ready to formulate our problem of designing supervisors with prescribed resilience to protect the plant under indefinite actuator and sensor attacks.

Problem 1 (Resilient Supervisory Control against Indefinite Attacks Problem)**.** *Given an unreliable plant $\widetilde{G} = (Q, \widetilde{\Sigma}, \widetilde{\delta}, q_0)$ as in (4), a subset of unsafe states $Q_u \subseteq Q$, and safety requirements R_a and R_s, with the attack-compromised language $L_{com}(\widetilde{G}, Q_u, R_a, R_s)$ in (6), find a feasible partial-observation supervisor $V : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ ($P : \widetilde{\Sigma}^* \to \widetilde{\Sigma}_o^*$) such that*

C1. V is physically admissible; and

C2. Specification L_K in (7) is satisfied with respect to L_{com}:

$$L(V/\widetilde{G}) \cap L_{com} \subseteq L_K$$

namely every string in L_{com} subject to the safety requirements R_a and R_s should be made to satisfy the specification; and

C3. V is a minimally-restrictive supervisor that satisfies C1 and C2, i.e., there does not exist another supervisor $V' : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ such that C1 and C2 are satisfied and $L(V'/\widetilde{G}) \supset L(V/\widetilde{G})$.

We emphasize that the language generated by $\widetilde{G}$ under control of V, namely $L(V/\widetilde{G})$, is only required to guarantee that event sequences which do not contain more than the allowed number of sensor or actuator attacks prescribed by the safety requirements resides in our specification L_K. This is because the nature of indefinite attacks is that our controlled language, namely $L(V/\widetilde{G})$, could very well result in event sequences that reaches unsafe states if an arbitrary number of actuator attacks occurs. This is not a shortcoming of our methodology, but rather is an essential part of the characteristics of indefinite attacks. That is, in theory, if an attacker could keep attacking the target system, then the system cannot be protected against such an extremely powerful attacker. This is the main reason why we have focused on synthesizing a supervisor which provides prescribed resilience against indefinite attacks.

IV. Synthesis of a Resilient Supervisor

In this section, we develop a novel approach to solve Problem 1. The procedure of this approach is summarized in Figure 7 as a flowchart, which shows the interconnections and dependencies among different concepts/components. We first construct the unreliable plant $\widetilde{G}$ as in Section III-A. Next, we construct *attack automata* H_a and H_s based on the sets of attack events (Σ_a and Σ_s) and the safety requirements (R_a and R_s), which act as trackers of the numbers of actuator attacks and sensor attacks. Then, to encode the trackers H_a and H_s to $\widetilde{G}$, we construct intermediate automata $H = H_a \parallel H_s$ and $\widehat{G} = \widetilde{G} \parallel H$. Next, we construct an *attack-compromised plant* G_{com} and a specification automaton K by preprocessing $\widehat{G}$, to represent the attack-compromised language L_{com} in (6) and the specification language L_K in (7) respectively. Then, we compute an intermediate (maximally observable and controllable) partial-observation supervisor V_{com} for G_{com} (which need not be physically admissible). Finally, we construct based on V_{com} a physically-admissible partial observation supervisor V for the unreliable plant $\widetilde{G}$. Our main result (Theorem 1 in Section IV-D below) asserts that the resulting V is a solution to Problem 1.

A. Attack-Compromised Plant

Based on the unreliable plant $\widetilde{G}$ and safety requirements R_a and R_s, our first step is to construct an *attack-compromised plant*, denoted by G_{com}. The main idea to construct G_{com} is to track the numbers of actuator and sensor attacks of strings

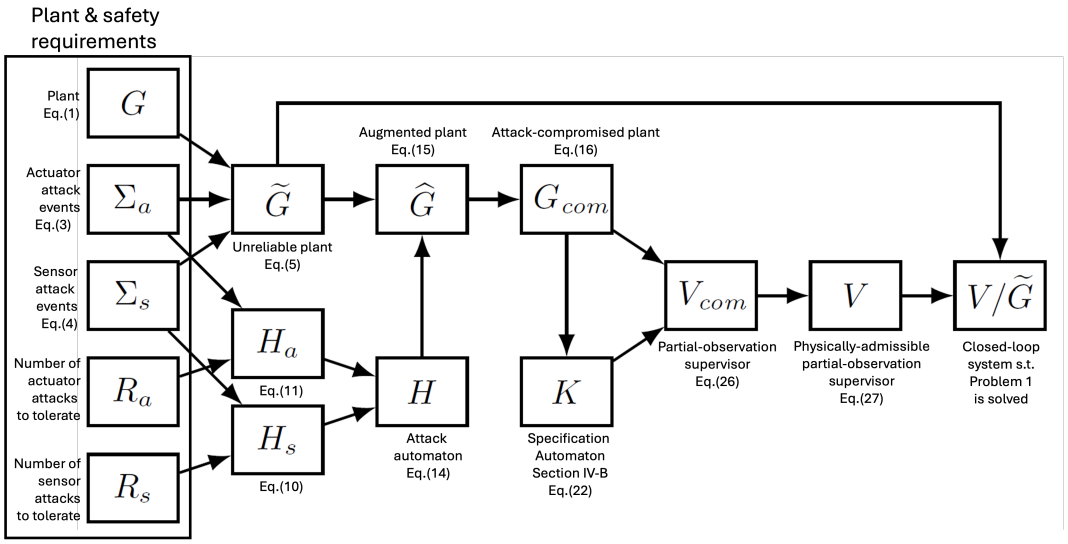


Fig. 7: Flowchart of the synthesis procedure (arrows represent dependencies between concepts/components)

in $\widetilde{G}$ according to the given safety requirements. For this, we introduce the following concepts.

Definition 6 (Sensor Attack Automaton)**.** Consider the unreliable plant $\widetilde{G} = (Q, \Sigma \cup \Sigma_a \cup \Sigma_s, \widetilde{\delta}, q_0)$, a subset of unsafe states $Q_u \subseteq Q$, and a sensor safety requirement $R_s : Q_u \to \mathbb{N}_0$. A *sensor attack automaton* is a four-tuple

$$H_s = (X_s, \Sigma \cup \Sigma_s, \xi_s, x_{0,s}) \tag{9}$$

where

- $X_s = \{x_{i,s} : i \in [0, r_s]\}$, where $r_s = \max_{q_u \in Q_u} R_s(q_u)$
- $\xi_s : X_s \times (\Sigma \cup \Sigma_s) \to X_s$ is given by

$$\xi_s(x_{i,s}, \sigma) = \begin{cases} x_{i,s} & \text{if } \sigma \in \Sigma \wedge 0 \le i \le r_s \\ x_{i+1,s} & \text{if } \sigma \in \Sigma_s \wedge r_s \ge 1 \\ & \quad \wedge 0 \le i \le r_s - 1 \\ \text{undefined} & \text{if } \sigma \in \Sigma_s \wedge i = r_s. \end{cases}$$

In words, a sensor attack automaton H_s (Definition 6) consists of $r_s + 1$ states which indicate the number of sensor attacks H_s tracks. The transitions labeled by sensor attack events increment the number of sensor attacks, and the current state of H_s moves to its next state. At every state, self-loops of original events are defined, so that original events do not increment the tracking number of attacks. Once H_s reaches its final state $x_{r_s,s}$, meaning that the numbers of sensor attacks reach the maximum of the safety requirement R_s, the automaton H_s no longer tracks attacks. By this construction, the number of sensor attack events in every string generated by H_s is less than or equal to r_s.

In a similar way, we construct another automaton to track the number of actuator attacks (cf. [26]):

$$H_a = (X_a, \Sigma \cup \Sigma_a, \xi_a, x_{0,a}). \tag{10}$$

The generated languages of H_a and H_s are, by construction,

$$L(H_a) = \{\omega_a \in (\Sigma \cup \Sigma_a)^* : |P_a(\omega_a)| \le r_a\}, \tag{11}$$

$$L(H_s) = \{\omega_s \in (\Sigma \cup \Sigma_s)^* : |P_s(\omega_s)| \le r_s\}, \tag{12}$$

respectively. That is, the numbers of actuator/sensor events in strings generated by attack automata are less than or equal to the maximum values of the respective safety requirements.

Now to simultaneously consider actuator and sensor attacks, we construct the following *attack automaton* by the parallel composition of H_a in (10) and H_s in (9), i.e.,

$$H = H_a \parallel H_s. \tag{13}$$

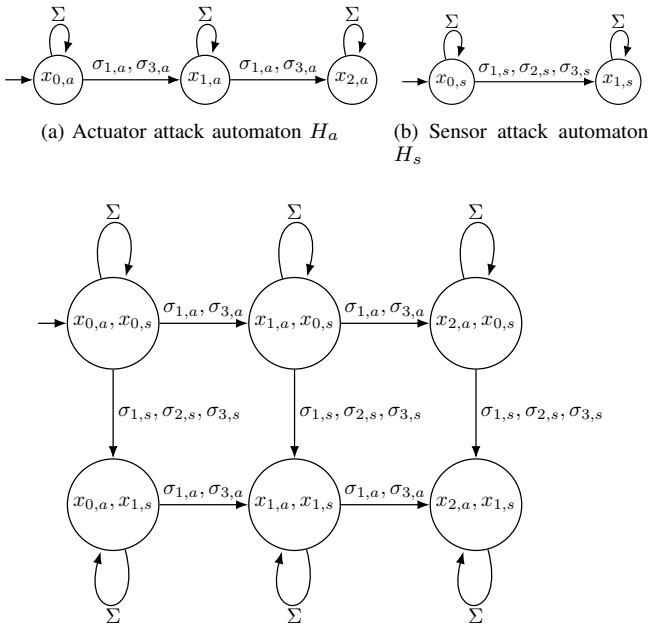
Remark 2. By setting $r_s = 0$ in Definition 6, we consider situations where there are only indefinite actuator attacks and no sensor attacks. Thus, our approach covers the previous work in [26] as a special case. Symmetrically, by setting $r_a = 0$, we consider situations where there are only indefinite sensor attacks, which has not been studied in the literature. Moreover, if $r_a = r_s = 0$, then the setup is back to the conventional supervisory control. In general, we consider situations where both indefinite actuator and sensor attacks exist, which is studied for the first time in the literature.

Example 4. Let us revisit the plant in Example 1. Since $r_a = \max_{q_u \in Q_u} R_a(q_u) = 2$ and $r_s = \max_{q_u \in Q_u} R_s(q_u) = 1$, the actuator attack automaton H_a and the sensor attack automaton H_s consist of three states and two states, are depicted in Figure 8a and Figure 8b, respectively. Figure 8c illustrates the attack automaton $H = H_a \parallel H_s$, used to track the number of attacks up to two actuator attacks and one sensor attack.

Using the attack automaton H in (13) and the unreliable plant $\widetilde{G}$ in (4), our next step is to construct an *augmented plant* as follows:

$$\widehat{G} = (\widehat{Q}, \widetilde{\Sigma}, \widehat{\delta}, \widehat{q}_0) = \widetilde{G} \parallel H \tag{14}$$

where $\widehat{Q} \subseteq Q \times X_a \times X_s$ is the set of triples of states from $\widetilde{G}$, H_a, and H_s, $\widetilde{\Sigma}$ is the same event set as that of $\widetilde{G}$, $\widehat{\delta} : \widehat{Q} \times \widetilde{\Sigma} \to \widehat{Q}$ is the transition function (based on the rules of parallel composition), and $\widehat{q}_0 = (q_0, x_{0,a}, x_{0,s})$ is the initial state. The numbers of actuator attacks and sensor attacks that have occurred are embedded respectively as the



(a) Actuator attack automaton H_a (b) Sensor attack automaton H_s

(c) Attack automaton H

Fig. 8: Attack automata for Example 1 with $r_a = 2$ and $r_s = 1$

second and third elements of each state triple in $\widehat{Q}$. Here, since H_a and H_s track up to r_a and r_s attacks which are the *maximum* values of R_a and R_s, respectively, the augmented plant $\widehat{G}$ may contain an unsafe state $(q_u, x_{i,a}, x_{j,s})$ where $i > R_a(q_u)$ or $j > R_s(q_u)$. Such a state corresponds to a trajectory in $\widetilde{G}$ which reaches an unsafe state q_u from the initial state and contains more actuator and sensor attack events than the respective safety requirements on this state q_u. Since we only need to consider unsafe state $q_u \in Q_u$ in $\widetilde{G}$ which are reachable via strings containing at most $R_a(q_u)$ actuator attacks *and* at most $R_s(q_u)$ sensor attacks, any states $(q_u, x_{i,a}, x_{j,s})$ in $\widehat{G}$ where $i > R_a(q_u)$ *or* $j > R_s(q_u)$ are unnecessary and thus should be removed.

Example 5. Consider the unreliable plant $\widetilde{G}$ in Figure 2 and H_a, H_s in Figure 8. By (14), we obtain the augmented plant $\widehat{G}$ in Figure 9. Three (dashed, grey) unsafe states labeled $A = (q_4, x_{0,a}, x_{1,s})$, $B = (q_4, x_{1,a}, x_{1,s})$, and $C = (q_4, x_{2,a}, x_{1,s})$ exceed the sensor safety requirement $R_s(q_4) = 0$. Thus, they do not need to be considered and should be removed. The state triples other than A, B, and C are omitted in Figure 9 to avoid cluttering.

Removing the superfluous states (if there are any) from the augmented plant $\widehat{G}$, we derive an automaton whose states are all within the required safety requirements on actuator and sensor attacks for individual unsafe states. We call this automaton "attack-compromised plant".

Definition 7 (Attack-Compromised Plant)**.** Given an augmented plant $\widehat{G}$ in (14), an *attack-compromised plant* is a four-tuple

$$G_{com} = (Q_{com}, \widetilde{\Sigma}, \delta_{com}, \widehat{q}_0) \tag{15}$$

where

$$\begin{aligned} Q_{com} &= \widehat{Q} \setminus Q^F_{com}, \text{where} \\ Q^F_{com} &= \{(q, x_{i,a}, x_{j,s}) \in \widehat{Q} : \\ &\qquad q \in Q_u \wedge (i > R_a(q) \vee j > R_s(q))\} \\ \delta_{com} &= \{(q_{com}, \sigma, q'_{com}) \in \widehat{\delta} : \\ &\qquad \sigma \in \widetilde{\Sigma} \wedge q_{com} \in Q_{com} \wedge q'_{com} \in Q_{com}\}. \end{aligned}$$

The subset Q^F_{com} in Definition 7 consists of unsafe states in $\widehat{G}$ which exceed their individual safety requirement R_a or R_s. In other words, to construct G_{com}, we remove from $\widehat{G}$ all states in Q^F_{com} and transitions from and to states in Q^F_{com}.

Example 6. Let us inspect Figure 9 again. By following Definition 7, we obtain the attack-compromised plant G_{com} by removing from the augmented plant $\widehat{G}$ in Figure 9 dashed grey states A, B, and C and dashed transitions associated to A, B, and C, since these three states exceed the sensor safety requirement $R_s(q_4) = 0$.

Remark 3. If the safety requirements R_a and R_s are the same for all unsafe states (i.e., $R_a(q_u) = i$ and $R_s(q_u) = j$ for all $q_u \in Q_u$ where $i, j \in \mathbb{N}_0$), then G_{com} is the same as $\widehat{G}$, since Q^F_{com} in Definition 7 is empty in this special case. Namely, there are no states to be removed from $\widehat{G}$ in order to form G_{com}.

Now, we assert that the generated language of attack-compromised plant is equal to the attack-compromised language, i.e., G_{com} generates L_{com} in (6). The significance of this result is that, for any given safety requirement, it provides a constructive way to represent the attack-compromised language by a finite-state automaton.

Proposition 1. *Given a plant $G = (Q, \Sigma, \delta, q_0)$, a subset of unsafe states $Q_u \subseteq Q$, and safety requirements R_a and R_s, with the unreliable plant $\widetilde{G}$ formed from (4), the attack-compromised language $L_{com} = L_{com}(\widetilde{G}, Q_u, R_a, R_s)$ formed from (6), and the attack-compromised plant G_{com} formed from (15), it holds that $L(G_{com}) = L_{com}$.*

Proof. Let $\widetilde{P}_a : \widetilde{\Sigma}^* \to (\Sigma \cup \Sigma_a)^*$ and $\widetilde{P}_s : \widetilde{\Sigma}^* \to (\Sigma \cup \Sigma_s)^*$ be natural projections. Since H is the parallel composition of H_a and H_s,

$$L(H) = \widetilde{P}_a^{-1}(L(H_a)) \cap \widetilde{P}_s^{-1}(L(H_s)). \tag{16}$$

Let $r_a = \max_{q_u \in Q_u} R_a(q_u)$ and $r_s = \max_{q_u \in Q_u} R_s(q_u)$. Here, from $L(H_a)$ in (11) and $L(H_s)$ in (12),

$$\begin{aligned} \widetilde{P}_a^{-1}(L(H_a)) &= \{\omega \in \widetilde{\Sigma}^* : |P_a(\omega)| \le r_a\} \\ \widetilde{P}_s^{-1}(L(H_s)) &= \{\omega \in \widetilde{\Sigma}^* : |P_s(\omega)| \le r_s\}. \end{aligned}$$

Thus, from (16), we have that

$$L(H) = \{\omega \in \widetilde{\Sigma}^* : |P_a(\omega)| \le r_a \wedge |P_s(\omega)| \le r_s\}$$

Since the set of all events in $\widetilde{G}$ is $\widetilde{\Sigma}$,

$$L(\widetilde{G} \parallel H) = L(\widetilde{G}) \cap L(H).$$

Hence,

$$L(\widehat{G}) = L(\widetilde{G} \parallel H)$$

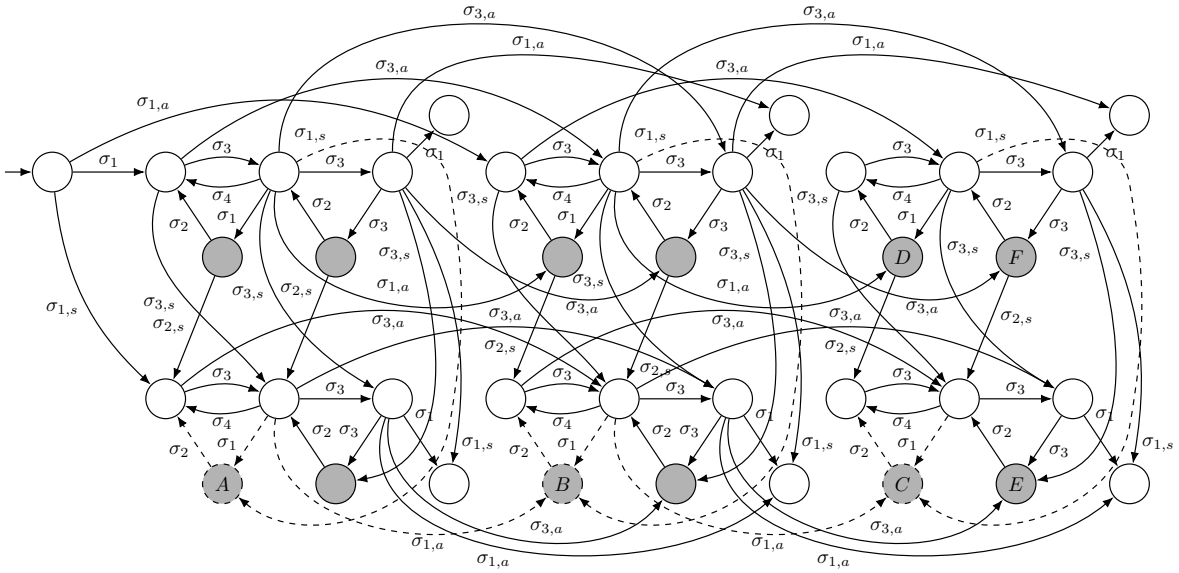


Fig. 9: Example augmented plant $\widehat{G}$; dashed grey (unsafe) states labeled $A = (q_4, x_{0,a}, x_{1,s})$, $B = (q_4, x_{1,a}, x_{1,s})$, and $C = (q_4, x_{2,a}, x_{1,s})$ and dashed transitions are to be removed from $\widehat{G}$ to form G_{com} due to the sensor safety requirement $R_s(q_4) = 0$; other grey states labeled $D = (q_4, x_{2,a}, x_{0,s})$, $E = (q_5, x_{2,a}, x_{1,s})$, and $F = (q_5, x_{2,a}, x_{0,s})$ are referred in Example 9 and Example 10.

$$= \{\omega \in L(\widetilde{G}) : |P_a(\omega)| \leq r_a \wedge |P_s(\omega)| \leq r_s\}. \quad (17)$$

This indicates that the number of attack events in strings yielded by the augmented plant is smaller than or equal to the maximum values of safety requirements.

Let $i \in [0, r_a]$ and $j \in [0, r_s]$. From Q^F_{com} in Definition 7, the state set of the attack-compromised plant Q_{com} is given by

$$Q_{com} = \{(q, x_{i,a}, x_{j,s}) \in \widehat{Q} : q \in Q_u \Rightarrow i \leq R_a(q) \wedge j \leq R_s(q)\}. \quad (18)$$

That is, the set of states of an attack-compromised plant only contains unsafe states which are reachable via a number of attacks not greater than R_a and R_s.

By the construction of $\widehat{G}$ in which the indices of states indicate the number of attacks, for all $(q, x_{i,a}, x_{j,s}) \in \widehat{Q}$ and $\omega \in L(\widehat{G})$,

$$\widehat{\delta}(\widehat{q}_0, \omega) = (q, x_{i,a}, x_{j,s}) \Leftrightarrow \widetilde{\delta}(q_0, \omega) = q \wedge |P_a(\omega)| = i \wedge |P_s(\omega)| = j. \quad (19)$$

Condition (19) essentially states that the state labels in $\widehat{Q}$ correspond to the state labels in $\widetilde{G}$ and the number of attacks. Hence, from (18) and (19), for all $\omega \in L(\widehat{G})$,

$$\widehat{\delta}(\widehat{q}_0, \omega) \in Q_{com} \Leftrightarrow \big((\forall t \in \overline{\omega})(\forall q_u \in Q_u)\, \widetilde{\delta}(q_0, t) = q_u \Rightarrow |P_a(t)| \leq R_a(q_u) \wedge |P_s(t)| \leq R_s(q_u)\big) \quad (20)$$

Condition (20) means that if string ω in $L(\widehat{G})$ leads to a state in Q_{com} which passes or reaches an unsafe state q_u, then the number of attacked events in ω should be less than or equal to $R_a(q_u)$ and $R_s(q_u)$.

Therefore, from (17) and (20),

$$\begin{aligned} L(G_{com}) &= \{\omega \in L(\widetilde{G}) : |P_a(\omega)| \leq r_a \wedge |P_s(\omega)| \leq r_s \\ &\quad \wedge \big((\forall t \in \overline{\omega})(\forall q_u \in Q_u)\, \widetilde{\delta}(q_0, t) = q_u \Rightarrow \\ &\quad |P_a(t)| \leq R_a(q_u) \wedge |P_s(t)| \leq R_s(q_u)\big)\} \\ &= L_{com} \end{aligned}$$

□

Here, we note that Proposition 1 implies that L_{com} is prefix-closed, since $L(G_{com})$ is prefix-closed by definition. We will exploit in Section IV-D this fact to show that our synthesis method produces a solution to Problem 1.

B. Synthesis via Partial-Observation Supervisory Control

Now that we have defined the attack-compromised plant G_{com}, we can construct a specification automaton from G_{com} by removing those state triples of G_{com} whose first elements are unsafe states in Q_u.

Definition 8 (Specification Automaton)**.** Given an attack-compromised plant $G_{com} = (Q_{com}, \widetilde{\Sigma}, \delta_{com}, \widehat{q}_0)$ in (15), we construct a specification automaton K by removing unsafe states from G_{com}, i.e.,

$$K = (Q_K, \widetilde{\Sigma}, \delta_K, \widehat{q}_0) \quad (21)$$

where

$$\begin{aligned} Q_K &= \{(q, \_, \_) \in Q_{com} : q \notin Q_u\} \quad (22) \\ \delta_K &= \{(q_{com}, \sigma, q'_{com}) \in \delta_{com} : \\ &\quad q_{com} \in Q_K \wedge q'_{com} \in Q_K \wedge \sigma \in \widetilde{\Sigma}\}. \end{aligned}$$

Example 7. Figure 10 depicts the specification automaton K derived from the attack-compromised plant G_{com} in Figure 9 by removing all grey unsafe states.

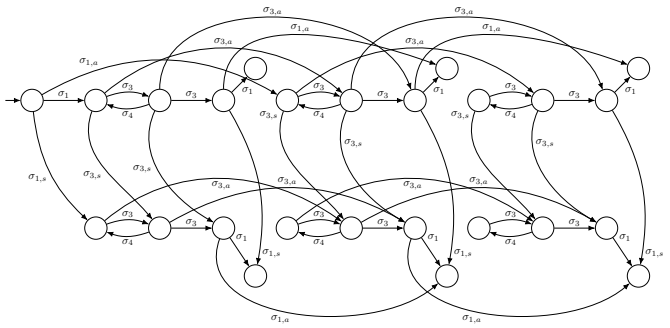


Fig. 10: Example specification automaton K

Our next result confirms that the specification automaton K in (21) generates the specification language L_K in (7).

Proposition 2. *Given a plant $G = (Q, \Sigma, \delta, q_0)$, a subset of unsafe states $Q_u \subseteq Q$, and safety requirements R_a and R_s, with unreliable plant $\widetilde{G}$ and attack-compromised language $L_{com} = L_{com}(\widetilde{G}, Q_u, R_a, R_s)$ formed as in (4) and (6), respectively, consider the specification automaton K in (21) and the specification language L_K in (7). It holds that $L(K) = L_K$.*

Proof. For convenience, let us define a mapping function $f : Q_{com} \to Q$ by $f((q, \_, \_)) = q$. Here, the set of states Q_K can be written as

$$Q_K = \{q_{com} \in Q_{com} : f(q_{com}) \notin Q_u\}.$$

By the construction of K, the generated language $L(K)$ is given by

$$L(K) = \{\omega \in L(G_{com}) : (\forall t \in \overline{\omega})\, f(\delta_{com}(\widehat{q}_0, t)) \notin Q_u\}. \tag{23}$$

That is, we remove strings from $L(G_{com})$ which pass or reach any unsafe states. By the construction of G_{com}, we know that for all $\omega \in L(G_{com})$,

$$f(\delta_{com}(\widehat{q}_0, \omega)) = \widetilde{\delta}(q_0, \omega). \tag{24}$$

Namely, the unreliable plant $\widetilde{G}$ reaches a state $q \in Q$ via string ω that leads the attack-compromised plant G_{com} to state $(q, \_, \_)$. Hence, from (23) and (24), we have that

$$L(K) = \{\omega \in L(G_{com}) : (\forall t \in \overline{\omega})\, \widetilde{\delta}(q_0, t) \notin Q_u\}.$$

Since $L(G_{com}) = L_{com}$ by Proposition 1,

$$\begin{aligned} L(K) &= \{\omega \in L_{com} : (\forall t \in \overline{\omega})\, \widetilde{\delta}(q_0, t) \notin Q_u\} \\ &= L_K \end{aligned}$$

□

Now that we have constructed a specification automaton K in (21) which corresponds to L_K in (7), and also an attack-compromised plant G_{com} in (15) which corresponds to L_{com} in (6), it is natural to proceed to supervisory synthesis. Recall that our goal is to find a feasible partial-observation supervisor $V : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ (where $P : \widetilde{\Sigma}^* \to \widetilde{\Sigma}_o^*$) which satisfies conditions C1–C3 in Problem 1. To find such V, we first compute based on plant G_{com} and specification K, a feasible partial-observation supervisor

$$V_{com} : P(L(G_{com})) \to 2^{\widetilde{\Sigma}_c} \tag{25}$$

(if one exists) with a standard partial-observation supervisory synthesis procedure (using, for example, the algorithm in [29] or the one in [30]), such that $L(V_{com}/G_{com}) \subseteq L(K)$ and $L(V_{com}/G_{com})$ is maximally controllable and observable with respect to $L(G_{com})$, the set of controllable events $\widetilde{\Sigma}_c$, and the natural projection $P : \widetilde{\Sigma}^* \to \widetilde{\Sigma}_o^*$.

We should note, however, that a feasible partial-observation supervisor V_{com} thus computed cannot be directly used to control $\widetilde{G}$ as is, because the domain of V_{com} is $P(L(G_{com}))$ instead of $P(L(\widetilde{G}))$ and, more importantly, there may exist a pair $(\sigma, \sigma_s) \in \Sigma_c \times \Sigma_{c,s}$ such that V_{com} yields inconsistent control decisions, that is, V_{com} may not be physically admissible. This point is illustrated in the example below.

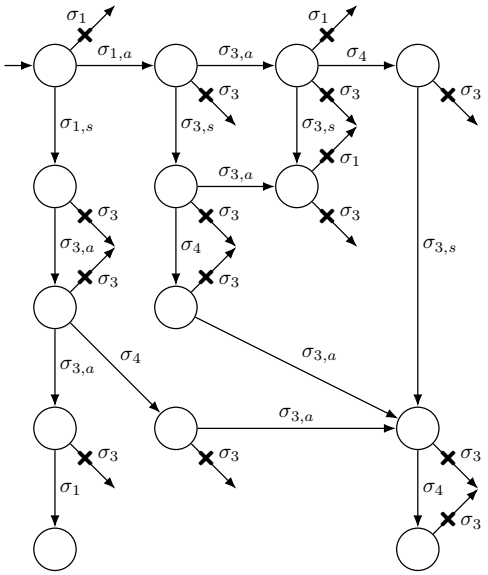


Fig. 11: Automaton representation of an example supervisor V_{com}; symbol ✖ represents the supervisor's decision to disable events.

Example 8. Figure 11 displays the automaton representation of a feasible partial-observation supervisor V_{com} for the attack-compromised plant G_{com} in Figure 9 and the specification automaton K in Figure 10 such that $L(V_{com}/G_{com}) \subseteq L(K)$ and $L(V_{com}/G_{com})$ is maximally controllable and observable with respect to $L(G_{com})$, $\widetilde{\Sigma}_c$, and $P : \widetilde{\Sigma}^* \to \widetilde{\Sigma}_o^*$.

It can be verified that V_{com} in Figure 11 is not physically admissible, because it disables σ_3 after $\sigma_{1,a}$, $\sigma_{1,a}\sigma_{3,a}$, and $\sigma_{1,a}\sigma_{3,a}\sigma_4$ but does not disable $\sigma_{3,s}$ after these strings. In addition, it disables σ_1 after ε but not after $\sigma_{1,s}$. (The reason why these inconsistencies occur will be explained in detail below in Example 9.) Consequently, V_{com} illustrated in Figure 11 fails to satisfy condition C1 in Problem 1 and therefore is not a valid solution to Problem 1.

Motivated by the need to address the issue of possible physical inadmissibility of V_{com}, in the next subsection we will further derive from V_{com} a feasible partial-observation supervisor V which satisfy all conditions C1–C3 in Problem 1.

C. Physically Admissible Supervisor

As the last but key step of finding a solution to Problem 1, we derive a feasible partial-observation supervisor V from V_{com} which not only can control $\widetilde{G}$ but also guarantees physical admissibility as defined in (8):

$$(\forall \omega \in L(\widetilde{G}))\, V(P(\omega)) = \begin{cases} V_{com}(P(\omega)) & \text{if } \omega \in L_{com} \\ \quad \cup \{\sigma_s \in \Sigma_{c,s} : \sigma \in V_{com}(P(\omega))\} & \\ \quad\quad \cup \{\sigma \in \Sigma_c : \sigma_s \in V_{com}(P(\omega))\} & \\ \varnothing & \text{if } \omega \in L(\widetilde{G}) \setminus L_{com} \end{cases} \tag{26}$$

That is, the supervisor V is constructed by disabling:

- D1. All controllable events disabled by V_{com}; and
- D2. The (controllable) sensor attack event $\sigma_s \in \Sigma_{c,s}$ if V_{com} disables the controllable event $\sigma \in \Sigma_c$ in the original plant corresponding to σ_s; and
- D3. The controllable event $\sigma \in \Sigma_c$ in the original plant if V_{com} disables the (controllable) sensor attack event $\sigma_s \in \Sigma_{c,s}$ corresponding to σ.

This construction (the disablement rules D1–D3 above) ensures that V is feasible and satisfies (8). Moreover, the supervisor V does not disable any events for strings not in L_{com}, i.e., when R_a and R_s have been exceeded. We also note that the construction in (26) does not produce an empty V from a nonempty V_{com}.

Example 9. Let us inspect the feasible partial-observation supervisor V_{com} illustrated in Figure 11. As pointed out in Example 8, this example V_{com} is not physically admissible.

Based on the attack-compromised plant G_{com} in Figure 9 (after dashed grey states A, B, C and dashed transitions in Figure 9 are removed), the behavior of the supervisor V_{com} can be described as follows.

First, V_{com} disables σ_1 after ε, since the unsafe state $(q_4, x_{2,a}, x_{0,s})$, labeled D, in Figure 9 could otherwise be reached uncontrollably by $\sigma_1\sigma_{3,a}\sigma_{1,a}$ (since we recall that actuator attack events are uncontrollable). On the other hand, $\sigma_{1,s}$ after ε is enabled, since unsafe state $(q_4, x_{2,a}, x_{1,s})$ (state C in Figure 9) does not exist in G_{com} due to $R_s(q_4) = 0$, and thus V_{com} does not take that unsafe state into account after string $\sigma_{1,s}$ occurs. Namely, string $\sigma_{1,s}\sigma_{3,a}\sigma_{1,a}$ (which goes to unsafe state C in Figure 9) is not in $L(G_{com})$.

Next, σ_3 is disabled after string $\sigma_{1,s}$ and $\sigma_{1,s}\sigma_{3,a}$, since otherwise the unsafe state $(q_5, x_{2,a}, x_{1,s})$, labeled E in Figure 9, could be reached uncontrollably by $\sigma_{1,s}\sigma_3\sigma_{3,a}\sigma_{3,a}$ and $\sigma_{1,s}\sigma_{3,a}\sigma_3\sigma_{3,a}$. Event σ_3 is also disabled after $\sigma_{1,s}\sigma_{3,a}\sigma_{3,a}$, to prevent G_{com} in Figure 9 from reaching unsafe state $(q_5, x_{2,a}, x_{1,s})$ (state E in Figure 9). Since σ_4 is unobservable, σ_3 is further disabled after $\sigma_{1,s}\sigma_{3,a}\sigma_4$, $\sigma_{1,s}\sigma_{3,a}\sigma_4\sigma_{3,a}$, and $\sigma_{1,s}\sigma_{3,a}\sigma_4\sigma_{3,a}\sigma_4$.

Also, V_{com} disables σ_1 after $\sigma_{1,a}\sigma_{3,a}$ to prevent G_{com} in Figure 9 from reaching unsafe state $(q_4, x_{2,a}, x_{0,s})$ (state D in Figure 9). As a result, since $\sigma_{3,s}$ is unobservable, σ_1 is also disabled after $\sigma_{1,a}\sigma_{3,a}\sigma_{3,s}$. Note that V_{com} need not disable $\sigma_{1,s}$ after $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_{3,s}$ since $(q_4, x_{2,a}, x_{1,s})$ (state C in Figure 9) does not exist in G_{com} and thus $\sigma_{1,a}\sigma_{3,a}\sigma_{1,s}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_{3,s}\sigma_{1,s}$ are not in $L(G_{com})$.

Furthermore, V_{com} disables σ_3 after $\sigma_{1,a}\sigma_{3,a}\sigma_{3,s}$ to prevent G_{com} in Figure 8 from reaching unsafe state $(q_5, x_{2,a}, x_{1,s})$ (state E in Figure 9). As a result, since $\sigma_{3,s}$ and σ_4 are unobservable, σ_3 is also disabled after strings $\sigma_{1,a}\sigma_{3,a}$, $\sigma_{1,a}\sigma_{3,a}\sigma_4$, $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_{3,s}$, $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_{3,s}\sigma_4$, $\sigma_{1,a}\sigma_{3,s}\sigma_4\sigma_{3,a}$, and $\sigma_{1,a}\sigma_{3,s}\sigma_4\sigma_{3,a}\sigma_4$.

It can also be seen that σ_3 is disabled after string $\sigma_{1,a}$. This is because the unsafe state $(q_4, x_{2,a}, x_{0,s})$ (state D in Figure 9) would be uncontrollably reachable by $\sigma_{1,a}\sigma_3\sigma_{1,a}$ otherwise. As a result, since $\sigma_{3,s}$ and σ_4 are unobservable, σ_3 is further disabled after $\sigma_{1,a}\sigma_{3,s}$ and $\sigma_{1,a}\sigma_{3,s}\sigma_4$.

Consequently, V_{com} in Figure 11 is not physically admissible, since it disables σ_3 after $\sigma_{1,a}$, $\sigma_{1,a}\sigma_{3,a}$, and $\sigma_{1,a}\sigma_{3,a}\sigma_4$ but does not disable $\sigma_{3,s}$ after those strings, and also disables σ_1 after ε but not $\sigma_{1,s}$.

Finally, based on the V_{com} illustrated in Figure 11, we construct from (26) a feasible partial-observation supervisor $V : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ which is physically admissible, by disabling $\sigma_{1,s}$ after ε and disabling $\sigma_{3,s}$ after $\sigma_{1,a}$, $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4$. As a result, we obtain the feasible physically-admissible partial-observation supervisor V illustrated in Figure 12.

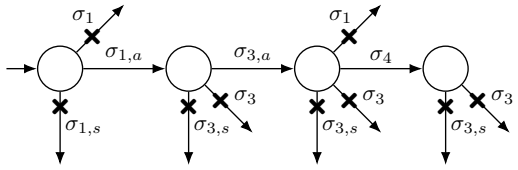


Fig. 12: Automaton representation of an example supervisor V derived from V_{com} in Figure 11 by (26); symbol ✖ represents the supervisor's decision to disable events.

Since V_{com} in Figure 11 is minimally restrictive, and there is no other way of choosing which events to additionally disable for physical admissibility, V in Figure 12 is a feasible minimally-restrictive physically-admissible partial-observation supervisor. That is, V in Figure 12 satisfies all the conditions C1–C3 in Problem 1. We will prove this fact in Section IV-D.

Recall that V_{com} is a supervisor that guarantees that $L(V_{com}/G_{com}) \subseteq L(K)$ and $L(V_{com}/G_{com})$ is maximally controllable and observable. The fact that V_{com} is *maximal* and not necessarily *maximum* underscores that, in general, there may exist multiple possible supervisors that achieve (incomparable) maximal controllable and observable sublanguages of $L(K)$. In our next example, we have exactly that situation.

Example 10. While only disabling additional sensor attack events (namely, the disablement rule D2, which corresponds to the third line of (26)) is required to derive physically admissible V in Figure 12 from V_{com} in Figure 11, there exists another feasible minimally-restrictive partial-observation supervisor, call it $V'_{com} : P(L(G_{com})) \to 2^{\widetilde{\Sigma}_c}$, such that $L(V'_{com}/G_{com}) \subseteq L(K)$ for G_{com} in Figure 9 and K in Figure 10, which is incomparable to V_{com} in Figure 11 and requires disabling additional controllable events in the original

plant (namely the disablement rule D3 after (26)) to construct V as a solution to Problem 1. Figure 13 illustrates such V'_{com}.

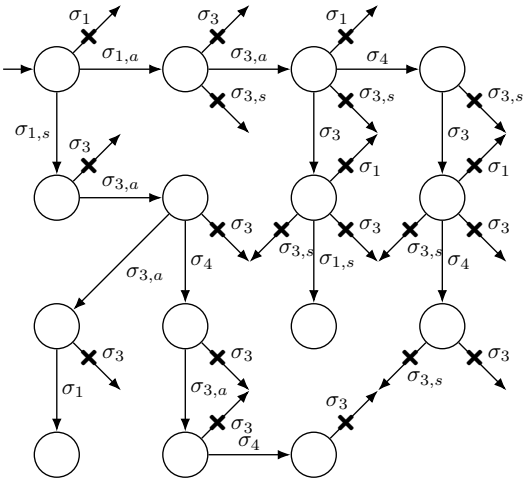


Fig. 13: Automaton representation of an example supervisor V'_{com}; symbol ✖ represents the supervisor's decision to disable events.

The main difference between V'_{com} in Figure 13 and V_{com} in Figure 11 is that V'_{com} disables $\sigma_{3,s}$ and enables σ_3 after $\sigma_{1,a}\sigma_{3,a}$ while V_{com} does the opposite. This comes from different options of how to prevent G_{com} in Figure 9 from reaching unsafe state $(q_5, x_{2,a}, x_{1,s})$ (state E in Figure 9). The supervisor V'_{com} in Figure 13 achieves this by disabling $\sigma_{3,s}$ after $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_3$, while V_{com} in Figure 11 does so by disabling σ_3 after $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_{3,s}$.

We focus now on V'_{com} and examine its disablement decisions. First, since $\sigma_{3,s}$ is disabled after $\sigma_{1,a}\sigma_{3,a}$ and σ_4 is unobservable, event $\sigma_{3,s}$ is also disabled after $\sigma_{1,a}\sigma_{3,a}\sigma_4$.

Next, V'_{com} disables σ_3 and $\sigma_{3,s}$ after $\sigma_{1,a}\sigma_{3,a}\sigma_3$ to prevent G_{com} in Figure 9 from reaching unsafe states $(q_5, x_{2,a}, x_{0,s})$ (state F in Figure 9) and $(q_5, x_{2,a}, x_{1,s})$ (state E in Figure 9), respectively. Since σ_4 is unobservable, σ_3 is also disabled after $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3$, and $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3\sigma_4$, and $\sigma_{3,s}$ is disabled after $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3\sigma_4$.

It can also be seen that $\sigma_{3,s}$ must be disabled after string $\sigma_{1,a}$, using the following argument. We have that σ_3 is enabled after $\sigma_{1,a}\sigma_{3,a}$. We also have that $\sigma_{1,a}\sigma_{3,s}\sigma_{3,a}\sigma_3$ leads G_{com} in Figure 9 to unsafe state $(q_5, x_{2,a}, x_{1,s})$ (state E in Figure 9). Since $\sigma_{3,s}$ is unobservable and thus $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,s}\sigma_{3,a}$ look the same, if V'_{com} were to disable σ_3 after $\sigma_{1,a}\sigma_{3,s}\sigma_{3,a}$, V'_{com} would no longer be feasible. Consequently, V'_{com} must disable $\sigma_{3,s}$ after $\sigma_{1,a}$.

Next, σ_1 after ε, σ_3 after $\sigma_{1,a}$, σ_3 after $\sigma_{1,s}$, σ_3 after $\sigma_{1,s}\sigma_{3,a}$ are disabled, since unsafe state $(q_4, x_{2,a}, x_{0,s})$ in G_{com} (state D in Figure 9) is uncontrollably reachable by $\sigma_1\sigma_{3,a}\sigma_{1,a}$; unsafe state $(q_5, x_{2,a}, x_{0,s})$ (state F in Figure 9) is uncontrollably reachable by $\sigma_{1,a}\sigma_3\sigma_{1,a}$; and unsafe state $(q_5, x_{2,a}, x_{1,s})$ (state E in Figure 9) is uncontrollably reachable by $\sigma_{1,s}\sigma_3\sigma_{3,a}\sigma_{3,a}$ and $\sigma_{1,s}\sigma_{3,a}\sigma_3\sigma_{3,a}$.

Note that $\sigma_{1,s}$ after ε need not be disabled for the same reason as explained in Example 9, i.e., unsafe state $(q_4, x_{2,a}, x_{1,s})$ (state C in Figure 9) does not exist in G_{com} due to $R_s(q_4) = 0$ and thus string $\sigma_{1,s}\sigma_{3,a}\sigma_{1,a}$ is not in $L(G_{com})$.

Moreover, σ_3 after $\sigma_{1,s}\sigma_{3,a}\sigma_{3,a}$ is disabled to prevent G_{com} from reaching unsafe state $(q_5, x_{2,a}, x_{1,s})$ (state E in Figure 9). Since σ_4 is unobservable, σ_3 is also disabled after $\sigma_{1,s}\sigma_{3,a}\sigma_4$, $\sigma_{1,s}\sigma_{3,a}\sigma_4\sigma_{3,a}$, and $\sigma_{1,s}\sigma_{3,a}\sigma_4\sigma_{3,a}\sigma_4$.

In addition, σ_1 is disabled after $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3$ to prevent G_{com} from reaching unsafe state $(q_4, x_{2,a}, x_{0,s})$ (state D in Figure 9). Since σ_4 is unobservable, this results in disabling σ_1 after $\sigma_{1,a}\sigma_{3,a}\sigma_3$.

Note that V'_{com} need not disable $\sigma_{1,s}$ after $\sigma_{1,a}\sigma_{3,a}$ and after $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3$ since $(q_4, x_{2,a}, x_{1,s})$ (state C in Figure 9) does not exist in G_{com} and thus $\sigma_{1,a}\sigma_{3,a}\sigma_{1,s}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3\sigma_{1,s}$ are not in $L(G_{com})$. Since σ_4 is unobservable and hence $\sigma_{1,a}\sigma_{3,a}\sigma_3$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4\sigma_3$ look the same, $\sigma_{1,s}$ is enabled after $\sigma_{1,a}\sigma_{3,a}\sigma_3$.

Consequently, the feasible partial-observation supervisor V'_{com} illustrated in Figure 13 is not physically admissible, since σ_3 is enabled but $\sigma_{3,s}$ is disabled after $\sigma_{1,a}\sigma_{3,a}$ and $\sigma_{1,a}\sigma_{3,a}\sigma_4$, and also $\sigma_{1,s}$ is enabled but σ_1 is disabled after ε. Therefore, by (26) (in particular by the disablement rules D2 and D3 after (26)), we obtain the same[3] feasible physically-admissible partial-observation supervisor in Figure 12 as Example 9.

From Example 9 and Example 10 above, it is justified that in (26) both $\{\sigma_s \in \Sigma_{c,s} : \sigma \in V_{com}(P(\omega))\}$ and $\{\sigma \in \Sigma_c : \sigma_s \in V_{com}(P(\omega))\}$ need to be included.

D. Main Result

As the main result of this paper, we prove that our automaton-based procedure presented in Sections IV-A to IV-C can produce a solution to Problem 1. This result confirms that we can construct a feasible partial-observation supervisor which provides a prescribed resilience for a given plant against both indefinite actuator and indefinite sensor attacks. This is an essential generalization of [26] which considers only the full-observation case.

Theorem 1. *Given a plant $G = (Q, \Sigma, \delta, q_0)$, a subset of unsafe states $Q_u \subseteq Q$, and safety requirements R_a and R_s, consider the unreliable plant $\widetilde{G}$ in (4), the attack-compromised plant G_{com} in (15), the specification automaton K in (21), the natural projection $P : \widetilde{\Sigma}^* \to \widetilde{\Sigma}_o^*$, the attack-compromised language L_{com} in (6), and the specification language L_K in (7). Let $V_{com} : P(L(G_{com})) \to 2^{\widetilde{\Sigma}_c}$ be a feasible partial-observation supervisor such that $L(V_{com}/G_{com}) \neq \varnothing$, $L(V_{com}/G_{com}) \subseteq L(K)$, and $L(V_{com}/G_{com})$ is maximally controllable and observable with respect to $L(G_{com})$, $\widetilde{\Sigma}_c$, and P. Then the feasible partial-observation supervisor V :*

[3]It is not surprising in this particular example that the same physically admissible V is produced because, while the two languages $L(V_{com}/G_{com})$ and $L(V'_{com}/G_{com})$ are incomparable, they differ only in whether σ_3 versus $\sigma_{3,s}$ is disabled after certain strings in $L(G_{com})$. Hence, to make each one physically admissible, the events that were not originally disabled by either V_{com} or V'_{com} become disabled by V. For example, since σ_3 (resp., $\sigma_{3,s}$) is disabled after $\sigma_{1,a}\sigma_{3,a}$ by V_{com} (resp., V'_{com}), we have to disable $\sigma_{3,s}$ (resp., σ_3) after $\sigma_{1,a}\sigma_{3,a}$ to make V_{com} (resp., V'_{com}) physically admissible. Consequently, in this example, the decisions after $\sigma_{1,a}\sigma_{3,a}$, which differ in V_{com} and V'_{com}, end up with the same result that both σ_3 and $\sigma_{3,s}$ are disabled after $\sigma_{1,a}\sigma_{3,a}$ by V. In general, though, $V_{com,1}$ and $V_{com,2}$ that are incomparable may result in incomparable V_1 and V_2.

$P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ derived from V_{com} as in (26) is a solution to Problem 1.

Proof. Let $V_{com} : P(L(G_{com})) \to 2^{\widetilde{\Sigma}_c}$ be a feasible partial-observation supervisor such that $L(V_{com}/G_{com}) \neq \varnothing$, $L(V_{com}/G_{com}) \subseteq L(K)$ and $L(V_{com}/G_{com})$ is maximally controllable and observable with respect to $L(G_{com})$, $\widetilde{\Sigma}_c$, and P. Define the feasible partial-observation supervisor $V : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ from V_{com} as in (26). We now show that V solves Problem 1. To this end, we show that V satisfies all the conditions C1–C3 in Problem 1.

Proof of C1: Condition (8) automatically holds for V by the construction of V in (26).

Proof of C2: Again by the construction of V in (26), for all $\omega \in L_{com}$ and $\sigma \in \widetilde{\Sigma}$, if V enables σ after ω, then V_{com} enables σ after ω. This means that if $\omega\sigma \in L(V/\widetilde{G})$ and $\omega\sigma \in L_{com}$, then $\omega\sigma \in L(V_{com}/G_{com})$. Thus, it holds that

$$L(V/\widetilde{G}) \cap L_{com} \subseteq L(V_{com}/G_{com}). \tag{27}$$

From $L(V_{com}/G_{com}) \subseteq L(K)$ and Proposition 2, it holds that

$$L(V_{com}/G_{com}) \subseteq L_K. \tag{28}$$

Then, from (27) and (28), we have that

$$L(V/\widetilde{G}) \cap L_{com} \subseteq L_K.$$

Proof of C3: Suppose that there is another feasible supervisor $V' : P(L(\widetilde{G})) \to 2^{\widetilde{\Sigma}_c}$ which is less restrictive than V, that is,

$$L(V/\widetilde{G}) \subset L(V'/\widetilde{G}). \tag{29}$$

We show that V' is either not feasible or violates one of C1 or C2 in Problem 1.

Expression (29) implies

$$L(V/\widetilde{G}) \cap L_{com} \subset L(V'/\widetilde{G}) \cap L_{com}.$$

For convenience, let $M' = L(V'/\widetilde{G}) \cap L_{com}$ and $M = L(V/\widetilde{G}) \cap L_{com}$. By assumption, $L(V_{com}/G_{com}) \subseteq L(K)$. Since $L(K) = L_K$ by Proposition 2, we know that $L(V_{com}/G_{com}) \subseteq L_K$. Also, by (7), we have that $L_K \subseteq L_{com}$. Furthermore, by (27), we get that $M \subseteq L(V_{com}/G_{com})$. Therefore, there are three possibilities for M', captured by the following three cases:

Case 1. $M \subset M' \subseteq L(V_{com}/G_{com})$

Case 2. $L(V_{com}/G_{com}) \subset M' \subseteq L_K$

Case 3. $L_K \subset M' \subseteq L_{com}$

First, consider Case 1, namely $M \subset M' \subseteq L(V_{com}/G_{com})$. We show that such V' is not physically admissible due to a string in $M' \setminus M$, i.e., condition C1 does not hold in this case.

By (26), V disables both $\sigma \in \Sigma_c$ and $\sigma_s \in \Sigma_{c,s}$ (where σ_s is the sensor attack event corresponding to σ) after any string $\omega \in L(V_{com}/G_{com})$ such that $\sigma \in V_{com}(P(\omega))$ or $\sigma_s \in V_{com}(P(\omega))$. Thus, by $M \subset M' \subseteq L(V_{com}/G_{com})$, there exists a string $\omega' \in M$ and an event $\sigma' \in \Sigma_c$ such that

P1. $\omega'\sigma' \in M' \setminus M$; or

P2. $\omega'\sigma'_s \in M' \setminus M$.

If P1 is true, by $M \subset M' \subseteq L(V_{com}/G_{com})$, it holds that

$$\omega'\sigma' \notin M \wedge \omega'\sigma' \in L(V_{com}/G_{com}). \tag{30}$$

By $\omega'\sigma' \notin M$ and (26), we have that $\omega'\sigma'_s \notin M$ (i.e., $\sigma' \in V(P(\omega'))$ and thus $\sigma_s \in V(P(\omega))$). From (30) we have $\omega'\sigma' \in L(V_{com}/G_{com})$ (i.e., $\sigma' \notin V_{com}(P(\omega'))$) and therefore by (26),

$$\omega'\sigma'_s \notin L(V_{com}/G_{com}).$$

Thus, by $M' \subseteq L(V_{com}/G_{com})$, it holds that

$$\omega'\sigma'_s \notin M'. \tag{31}$$

Hence, from P1 and (31), we have that $\sigma' \notin V'(P(\omega'))$ and $\sigma'_s \in V'(P(\omega'))$. This violates condition (8).

If P2 is true, by $M \subset M' \subseteq L(V_{com}/G_{com})$, it holds that

$$\omega'\sigma'_s \notin M \wedge \omega'\sigma'_s \in L(V_{com}/G_{com}). \tag{32}$$

By $\omega'\sigma'_s \notin M$ and (26), we have that $\omega'\sigma' \notin M$ (i.e., $\sigma'_s \in V(P(\omega'))$ and thus $\sigma' \in V(P(\omega'))$). From (32) we have $\omega'\sigma'_s \in L(V_{com}/G_{com})$ (i.e., $\sigma'_s \notin V_{com}(P(\omega'))$) and so by (26),

$$\omega'\sigma' \notin L(V_{com}/G_{com}).$$

Thus, by $M' \subseteq L(V_{com}/G_{com})$, it holds that

$$\omega'\sigma' \notin M'. \tag{33}$$

Hence, from P2 and (33), we have that $\sigma'_s \in V'(P(\omega'))$ and $\sigma' \notin V'(P(\omega'))$. This violates condition (8).

From the above two cases, we can conclude that V' is not physically admissible and violates condition C1.

Next, consider Case 2, namely $L(V_{com}/G_{com}) \subset M' \subseteq L_K$. We show that $L(V'/\widetilde{G})$ is either uncontrollable or unobservable with respect to L_{com}, $\widetilde{\Sigma}_c$, and P, i.e., V' in this case contradicts the assumption that V' is feasible (as only sublanguage that are both controllable and observable can be synthesized by feasible supervisors [1]). Since $L(V_{com}/G_{com})$ is already maximally controllable and observable with respect to L_{com}, $\widetilde{\Sigma}_c$, and P, from $L(V_{com}/G_{com}) \subset M'$, the language M' is uncontrollable or unobservable with respect to L_{com}, $\Sigma_c \cup \Sigma_{c,s}$, and P. Since L_{com} is prefix-closed and controllable with respect to L_{com} and $\widetilde{\Sigma}_c$, and $L(V'/\widetilde{G})$ is prefix-closed, if $M' = L(V'/\widetilde{G}) \cap L_{com}$ is uncontrollable with respect to L_{com}, then $L(V'/\widetilde{G})$ is uncontrollable with respect to L_{com} and $\widetilde{\Sigma}_c$. This is a result of the fact that prefix-closed and controllable languages are closed under intersection [2]. This contradicts the assumption that V' is feasible. Since L_{com} is prefix-closed and observable with respect to L_{com} and P, if $M' = L(V'/\widetilde{G}) \cap L_{com}$ is unobservable with respect to L_{com} and P, then $L(V'/\widetilde{G})$ is unobservable with respect to L_{com} and P, as $L(V'/\widetilde{G})$ is prefix-closed by definition and observability is closed under intersection for prefix-closed languages [2]. This contradicts the assumption that V' is feasible.

Finally, consider Case 3, namely $L_K \subset M' \subseteq L_{com}$. In this case, V' directly violates condition C2.

Therefore, any supervisor V' that is less restrictive than V cannot be a solution to Problem 1. Consequently, the supervisor V satisfies C3.

The proof is now complete. □

Remark 4. If our procedure yields an empty supervisor V, then there does not exist a nonempty sublanguage of L_K that is controllable with respect to L_{com} and observable with respect to L_{com} and P. This means that Problem 1 is not solvable, since no feasible supervisor which generates a nonempty sublanguage of L_K can be synthesized in such a case. Therefore (in combination with Theorem 1), we have shown that Problem 1 is solvable if and only if the supervisor V produced by our procedure is nonempty.

V. Conclusions

We have studied resilient supervisory control against indefinite actuator and sensor attacks, as an extension of the previous work in [26]. We have introduced a novel property called physical admissibility which ensures that no disabled event is subject to a sensor attack. Using this new notion, our problem of finding a feasible physically-admissible partial-observation supervisor which provides prescribed resilience against indefinite actuator and sensor attacks has been formulated. Moreover, we have developed an automaton-based procedure to synthesize a solution supervisor. Ensuring that the resulting supervisor is physically admissible was a new and essential step in our procedure.

In future work, we aim to develop tools for systematically determining or estimating appropriate values for the safety requirements R_a and R_s for practical systems. We also aim to consider other types of sensor attacks such as event insertion, deletion, and editing. Moreover, we are interested in extending this work to a decentralized architecture in which there are multiple supervisors with different observation capabilities, and investigate whether attackers can exploit constraints on decentralized supervisors such as fusion rules.

References

[1] W. M. Wonham and K. Cai, *Supervisory Control of Discrete-Event Systems*. Springer, 2019.

[2] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems (3rd Edition)*. Springer, 2021.

[3] L. K. Carvalho, Y.-C. Wu, R. Kwong, and S. Lafortune, "Detection and prevention of actuator enablement attacks in supervisory control systems," in *International Workshop on Discrete Event Systems*, 2016, pp. 298–305.

[4] L. K. Carvalho, Y.-C. Wu, R. Kwong, and S. Lafortune, "Detection and mitigation of classes of attacks in supervisory control systems," *Automatica*, vol. 97, pp. 121–133, 2018.

[5] P. M. Lima, L. K. Carvalho, and M. V. Moreira, "Detectable and undetectable network attack security of cyber-physical systems," *IFAC-PapersOnLine*, vol. 51, no. 7, pp. 179–185, 2018.

[6] J. Yao, X. Yin, and S. Li, "On attack mitigation in supervisory control systems: A tolerant control approach," in *IEEE Conference on Decision and Control*, 2020, pp. 4504–4510.

[7] Z. He, N. Wu, and Z. Li, "Estimation and prevention of actuator enablement attacks in discrete-event systems under supervisory control," *IEEE Transactions on Automatic Control*, vol. 69, no. 9, pp. 5963–5978, 2024.

[8] R. Tai, L. Lin, and R. Su, "On decidability of existence of fortified supervisors against covert actuator attackers," *IEEE Transactions on Automatic Control*, vol. 69, no. 3, pp. 1898–1905, 2024.

[9] M. Wakaiki, P. Tabuada, and J. P. Hespanha, "Supervisory control of discrete-event systems under attacks," *Dynamic Games and Applications*, vol. 9, pp. 965–983, 2019.

[10] R. Meira-Góes, H. Marchand, and S. Lafortune, "Towards resilient supervisors against sensor deception attacks," in *IEEE Conference on Decision and Control*, 2019, pp. 5144–5149.

[11] R. Meira-Góes, S. Lafortune, and H. Marchand, "Synthesis of supervisors robust against sensor deception attacks," *IEEE Transactions on Automatic Control*, vol. 66, no. 10, pp. 4990–4997, 2021.

[12] J. Yao, S. Li, and X. Yin, "Sensor deception attacks against security in supervisory control systems," *Automatica*, vol. 159, p. 111330, 2024.

[13] Q. Zhang, C. Seatzu, Z. Li, and A. Giua, "Joint state estimation under attack of discrete event systems," *IEEE Access*, vol. 9, pp. 168 068–168 079, 2021.

[14] R. Meira-Góes, E. Kang, R. Kwong, and S. Lafortune, "Stealthy deception attacks for cyber-physical systems," in *IEEE Conference on Decision and Control*, 2017, pp. 4224–4230.

[15] Q. Zhang, Z. Li, C. Seatzu, and A. Giua, "Stealthy attacks for partially-observed discrete event systems," in *International Conference on Emerging Technologies and Factory Automation*, vol. 1, 2018, pp. 1161–1164.

[16] R. Meira-Góes, R. Kwong, and S. Lafortune, "Synthesis of sensor deception attacks for systems modeled as probabilistic automata," in *American Control Conference*, 2019, pp. 5620–5626.

[17] R. Meira-Góes, E. Kang, R. H. Kwong, and S. Lafortune, "Synthesis of sensor deception attacks at the supervisory layer of cyber-physical systems," *Automatica*, vol. 121, p. 109172, 2020.

[18] R. Su, "On decidability of existence of nonblocking supervisors resilient to smart sensor attacks," *Automatica*, vol. 154, p. 111076, 2023.

[19] G. Xie, Y. Tong, X. Wang, and C. Seatzu, "Resilient supervisor synthesis for labeled petri nets against sensor attacks," *IEEE Transactions on Automatic Control*, vol. 70, no. 6, pp. 4045–4052, 2025.

[20] P. M. Lima, M. V. Alves, L. K. Carvalho, and M. V. Moreira, "Security of cyber-physical systems: Design of a security supervisor to thwart attacks," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 2030–2041, 2021.

[21] R. Meira-Góes, H. Marchand, and S. Lafortune, "Dealing with sensor and actuator deception attacks in supervisory control," *Automatica*, vol. 147, p. 110736, 2023.

[22] Y. Wang, A. K. Bozkurt, N. Smith, and M. Pajic, "Attack-resilient supervisory control of discrete-event systems: a finite-state transducer approach," *IEEE Open Journal of Control Systems*, vol. 2, pp. 208–220, 2023.

[23] R. Tai, L. Lin, Y. Zhu, and R. Su, "Synthesis of distributed covert sensor–actuator attackers," *IEEE Transactions on Automatic Control*, vol. 69, no. 8, pp. 4942–4957, 2024.

[24] Z. He, N. Wu, R. Su, and Z. Li, "Cyber-attacks with resource constraints on discrete event systems under supervisory control," *IEEE/CAA Journal of Automatica Sinica*, vol. 12, no. 3, pp. 585–595, 2025.

[25] S. Oliveira, A. Leal, M. Teixeira, and Y. Lopes, "A classification of cybersecurity strategies in the context of discrete event systems," *Annual Reviews in Control*, vol. 56, p. 100907, 2023.

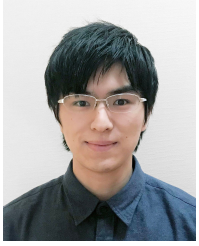
[26] Z. Ma and K. Cai, "On resilient supervisory control against indefinite actuator attacks in discrete-event systems," *IEEE Control Systems Letters*, vol. 6, pp. 2942–2947, 2022.

[27] K. Cai, *Invitation to Supervisory Control of Discrete-Event Systems: with Hands-On PyTCT*. Kindle Direct Publishing, 2024. [Online]. Available: https://www.caikai.org/invitation-scdes

[28] F. Lin and W. M. Wonham, "On observability of discrete-event systems," *Information sciences*, vol. 44, no. 3, pp. 173–198, 1988.

[29] X. Yin and S. Lafortune, "Synthesis of maximally permissive supervisors for partially-observed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 5, pp. 1239–1254, 2015.

[30] K. Cai, R. Zhang, and W. Wonham, "Relative observability of discrete-event systems and its supremal sublanguages," *IEEE Transactions on Automatic Control*, vol. 60, pp. 659–670, 2015.



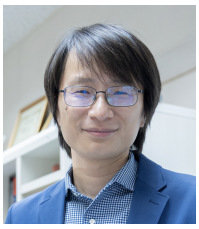
Shoma Matsui received the B.Eng. in Electrical and Information Engineering degree from Osaka City University, Japan, in 2019, the M.Sc. in Engineering degree from University of Michigan, USA, in 2020, and the Ph.D. degree from Queen's University, Canada, in 2025. His research interests include secrecy, privacy, and safety of Discrete-Event Systems, and applications to computer and network systems.

Julien Bourgain-Wilbal received the B.Eng. in Mechanical Engineering degree from ENS Paris-Saclay, France, in 2022, and a M.Sc. in Engineering degree from University of Paris-Saclay, France, in 2024. He is currently a PhD student at University Paris-Saclay, France.

Yudai Saito received the B.Eng. degree in 2024 and is pursuing the M.Inf. degree in the Department of Core Informatics, Osaka Metropolitan University, Japan. His research interest is security in discrete-event systems.

Ziyue Ma received the B.Sc. degree and the M.Sc. degree in Chemistry from Peking University, Beijing, China, in 2007 and 2011, respectively. In 2017 he got the Ph.D degree in co-tutorship between the School of Electro-Mechanical Engineering of Xidian University, China (in Mechatronic Engineering), and the Department of Electrical and Electronic Engineering of University of Cagliari, Italy (in Electronics and Computer Engineering). He joined Xidian University in 2011, where he is currently an Associate Professor in the School of Electro-Mechanical Engineering. His current research interests include control theory in discrete event systems, automata and Petri net theories, fault diagnosis/prognosis, resource optimization, and information security.

Gregory Faraut received the B.S. degree in electrical engineering and the M.S. degree in computer science from the University of Nice Sophia Antipolis, Nice, France, in 2004 and 2006, respectively, and the Ph.D. degree in automatic control from the Ampere Lab, INSA Lyon, Villeurbanne, France, in 2010. Since 2011, he has been an Associate Professor of Automatic Control with LURPA, ENS Paris-Saclay, University of Paris-Saclay, and Full Professor since 2022. His research interests concern the field of discrete-event systems with applications to cyber-physical systems, behavioral identification, resilient control, and, more recently, digital twins for cognitive systems.



Kai Cai received the B.Eng. degree in Electrical Engineering from Zhejiang University, China, in 2006; the M.A.Sc. degree in Electrical and Computer Engineering from the University of Toronto, Canada, in 2008; and the Ph.D. degree in Systems Science from the Tokyo Institute of Technology, Japan, in 2011. He is currently a Full Professor with the Department of Core Informatics, Osaka Metropolitan University. Dr. Cai's research interests include safe control of AI-physical systems, supervisory control of discrete-event systems, and cooperative control of multi-agent systems.

Karen Rudie received her Ph.D. in 1992 from the University of Toronto, in the Systems Control Group. In 1992–93 she was a postdoctoral researcher at the Institute for Mathematics and its Applications, Minnesota. Since 1993 she has been at Queen's University (Canada) where she is a Professor of Electrical and Computer Engineering and Director (Interim) of Ingenuity Labs Research Institute. She has served on the editorial board of the Journal of Discrete Event Dynamic Systems (since 2000) where she is currently a Department Editor, and was an Associate Editor for IEEE Transactions on Control Systems Technology (2015–2020), IEEE Transactions on Automatic Control (1996–1999), and IEEE Control Systems Magazine (2003). From 2004–2006 she was an IEEE Control Systems Society Distinguished Lecturer. She is a Fellow of the IEEE. Her research focuses on discrete-event systems—most recently with an emphasis on their application to the security of cyber-physical systems.