Attack-Resilient Supervisory Control with Intermittently Secure Communication

Yu Wang and Miroslav Pajic

***Abstract*— In this work, we study supervisory control of discrete event systems in the presence of network-based attacks on information delivered to and from the supervisors. The attacks are modeled by finite state transducers (FSTs), having the ability to nondeterministically rewrite a word to any word of a regular language. A desired language is called *controllable* if there exists a security-aware supervisor that ensures that the restricted language executed by the plant for any possible attack behavior is the desired one – we refer to such supervisors as *attack-resilient*. First, we solve the problem of computing the maximal controllable sub-language (MCSL) of a desired language and propose the design algorithm for an attack-resilient supervisor, in scenarios where no security guarantees exists for communication between the plant and the supervisor. Then, we consider the case where the supervisor has active but intermittent access to a size-limited secure channel, which ensures integrity and availability of the data transmitted over it. Specifically, we propose the notion of accessibility as a measure of distance between a language and its sub-language, and show that a desired language is controllable with intermittently secure communication if and only if its difference from its MCSL without secure channel is bounded by the accessibility measure. Finally, we illustrate our approach on several examples.**

I. Introduction

Control resiliency is a key issue in deployment of cyber-physical systems (CPS) in a wide range of safety-critical but attack-subjective environments, such as smart grids [1], medical devices [2], and distributed control systems [3]. In general, in these applications, adversarial attacks can happen from the cyber and/or physical domains; attacks targeting sensors or actuators of the plant, communication between the controller and the plant, or even the physical environment of the plant, have raised the urgency to develop methods to analyze such systems in the presence of attacks, as well as design attack-resilient systems (e.g., [4], [5], [6]). Such attack-resilient systems provide strong Quality-of-Control guarantees even under intelligent and coordinated attacks [7], [6] that introduce malicious behaviors more complex than merely generating random failures (which can be handled with fault tolerant control techniques).

In this work, we consider the problem of supervisory control of discrete event systems in the presence of network-based attacks on communication between the supervisor and the plant's sensors and actuators, as illustrated in Figure 1. We focus on network-based attacks, such as *Man-in-the-Middle* or *Denial-of-Service* attacks, since network connectivity, which is prevalent in CPS, allows for a remote attacker to affect control performance by tampering with the transmitted sensor measurements and actuator commands. As proposed in our companion paper [8], we model attacks on discrete-event systems using finite state transducers (FST) by exploiting nondeterminism of FSTs to capture all possible attack actions; in [8] we show that FSTs enable modeling most forms of malicious activity in CPS, including ones that may occur as result of network-based attacks that corrupt the data communicated between the plant and controller, as well as non-invasive attacks that affect the environment of the plant (e.g., GPS spoofing attacks on autonomous vehicles [9]). This attack model generalizes the ones studied in the growing literature in the attack-resilient discrete event systems, considering possible attacks between the communication between the supervisor and the plant [10], [11], [12], [13]. We also use FSTs as the supervisor for the plants modeled by deterministic finite automata, giving it the power to revise symbols instead of simple insertion or deletion as in [10], [11], [12], [13].

FSTs have found applications in a wide-range of application domains, such as applications in speech recognition (e.g., [14]). In this work, nondeterministic FSTs are employed to model all possible malicious behaviors of the network-based attacks. Additional advantage of using FSTs to model attacks is that the class of FSTs is closed under inversion and composition, which are easily computable. Consequently, the problem of designing attack-resilient supervisory control for complex system configurations and coordinated attacks from multiple attack-points, where each attack is modeled by an FST, can be simplified to a few basic configurations.

We are mainly concerned with two problems in the attack-resilient supervisory control of discrete-event plants commonly modeled as deterministic finite state automata (e.g., [15]): **(i)** control of plants in the presence of FST-modeled (i.e., regularly-rewriting) attacks on the communication between the actuators/sensors and the supervisor, and **(ii)** the same problem with the supervisor having active and intermittent access to a secure size-limited communication channel, as illustrated in Figure 1. The active and intermittent access to a size-limited secure channel captures the common practice of providing intermittent security guarantees for communication over potentially compromised networks. For example, in CAN-based automotive systems, the continuous use of cryptographic security primitives to protect real-time communication, may not be possible due to resource constraints such as network bandwidth and the processing speed of the computers [16], [17], [18]. Note that the network as well as the processors used for control computation are commonly shared between more than one control loop. Thus,

The authors are with the Department of Electrical & Computer Engineering, Duke University, Durham, NC 27707, USA. Email: yu.wang094@duke.edu, miroslav.pajic@duke.edu

This work was supported in part by the NSF CNS-1652544 grant, as well as by the ONR under agreements number N00014-17-1-2012 and N00014-17-1-2504, and the AFOSR under award number FA9550-19-1-0169.

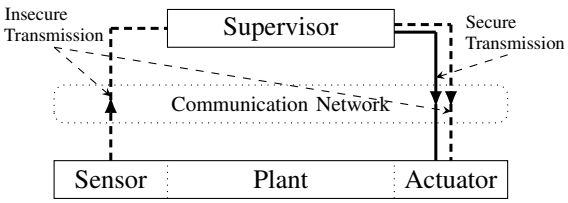


Fig. 1: Supervisory control under attack on the communication between the plant's sensors and actuators and the supervisor.

increasing communication/computation requirements in every control cycle may not be possible if done for all control loops; however, if additional resources are only intermittently used for each control loop, the resource overhead can be 'spread' around for all control loops, making the system schedulable on deployed platforms.

The major contributions of this work are twofold. First, we solve the problem of computing the maximal controllable sublanguage (MCSL) of a desired language and the corresponding attack-resilient supervisor design. These result can be viewed as an supplement and continuation of the attack-resilient supervisory control problems proposed in [8]. Second, we study the benefit of the supervisor having active and intermittent access to a size-limited secure channel in attack-resilient supervisory control. Specifically, we propose a graph-theoretic metric for the difference between an automaton and its sub-automaton, and use the metric to derive a controllability condition for a desired language, under the presence of such a secure channel; we show that the desired language is controllable and the system can be made resilient to attacks, if and only if the language's difference from its MCSL measured by accessibility is within the secure channel's repairing ability.

The rest of the article is organized as follows. The preliminaries on discrete events systems and FSTs are provided in Section II. In Section III, the problem formulation is presented. In Section IV, we provide a procedure to compute the MCSL of a desired language K without any secure transmission, and design the corresponding attack-resilient supervisor. Based on this, we present the controllability theorem and the supervisor design for a desired language with the resilient supervisor having active but intermittent access to a size-limited secure channel via a graph-theoretic approach in Section V. Finally, we conclude the work in Section VI.

II. Preliminaries

A finite-length sequence taken from a given finite set of symbols is called a *word* and the empty word is denoted by ε. A set of words is called a *language* of those symbols. The set of all *regular* languages of a set of symbols $\mathbf{I}$ is denoted by $\mathrm{Re}(\mathbf{I}^*)$. The cardinality of set A is denoted by $|A|$. For two sets A and B, let $A \backslash B = \{x \in A \mid x \notin B\}$. For $n \in \mathbb{N}$, let $[n] = \{0, 1, \ldots, n-1\}$. For a word $\mathsf{i}_1\mathsf{i}_2 \ldots \mathsf{i}_n$, we call $\mathsf{i}_1\mathsf{i}_2 \ldots \mathsf{i}_k$, with $k \leq n$, a prefix of I. For a language L, its prefix-closure is defined by $\overline{L} = \{I \mid I \text{ is a prefix of } J, J \in L\}$. The language L is *prefix-closed* if $L = \overline{L}$. The length of a word is denoted by $|\cdot|$. We adopt the following convention on generating regular expressions: for languages L_1 and L_2, L_1^* stands for repeating L_1 finitely many times, L_1, L_2 for the disjunction of L_1 and L_2, and $\overline{L}_1$ for the prefix closure of L_1.

A relation $\mathcal{R}$ between two sets A, B is a set $\mathcal{R} \subseteq A \times B$. For $a \in A$, let $\mathcal{R}(a) = \{b \in B \mid (a, b) \in \mathcal{R}\}$. The relation $\mathcal{R}$ is a partial function if $\mathcal{R}(a)$ is empty or a singleton for any $a \in A$. More generally, for $A' \subseteq A$, while slightly abusing the notation, let $\mathcal{R}(A') = \{b \in B \mid (a', b) \in \mathcal{R}, a' \in A'\}$. Clearly, $\mathcal{R}(\cdot)$ defines a function $2^A \to 2^B$. For relation $\mathcal{R} \subseteq A \times B$, its inversion is defined by $\mathcal{R}^{-1} = \{(b, a) \in B \times A \mid (a, b) \in \mathcal{R}\}$. For two relations $\mathcal{R} \subseteq A \times B$ and $\mathcal{R}' \subseteq B' \times C$, their (serial) composition is defined by $\mathcal{R} \circ \mathcal{R}' = \{(a, c) \in A \times C \mid \exists b \in B \cap B' : (a, b) \in \mathcal{R} \wedge (b, c) \in \mathcal{R}'\}$.

A. Discrete Event Systems

Discrete event systems are deterministic finite state automata (FSA) [15], [19].

Definition 1 (Finite State Automata)**.** *An FSA $\mathcal{P}$ is a tuple $\mathcal{P} = (\mathsf{S}, \mathsf{s}_{\mathrm{init}}, \mathbf{I}, \mathsf{Trans}, \mathsf{S}_{\mathrm{final}})$ where*

- S *is a finite set of* states*;*
- $\mathsf{s}_{\mathrm{init}} \in \mathsf{S}$ *is the* initial state*;*
- $\mathbf{I} \cup \{\varepsilon\}$ *is a finite set of* inputs*;*
- $\mathsf{Trans} : \mathsf{S} \times \mathbf{I} \to \mathsf{S}$ *is a* transition *relation;*
- $\mathsf{S}_{\mathrm{final}} \subseteq \mathsf{S}$ *is a finite set of* final states.

The FSA $\mathcal{P}$ is deterministic if Trans *is a partial function. The sequence* $(\mathsf{s}_{\mathrm{init}}, \mathsf{i}_0, \mathsf{s}_0)(\mathsf{s}_0, \mathsf{i}_1, \mathsf{s}_1) \ldots (\mathsf{s}_{n-2}, \mathsf{i}_{n-1}, \mathsf{s}_{n-1})$ *is called an* execution *of $\mathcal{P}$, if $\mathsf{s}_i \subseteq \mathsf{Trans}(\mathsf{s}_{i-1}, \mathsf{i}_i)$ for $i \in [n]$ with $\mathsf{s}_{-1} = \mathsf{s}_{\mathrm{init}}$. The state s_{n-1} is called reachable by the input word $I = \mathsf{i}_0\mathsf{i}_1 \ldots \mathsf{i}_n$. The input word I is* accepted *by $\mathcal{P}$, if there exists an execution ending at $\mathsf{S}_{\mathrm{final}}$. The set of* words *accepted by $\mathcal{P}$ is called the* language *accepted by $\mathcal{P}$, denoted by $L(\mathcal{P})$. On the other hand, $\mathcal{P}$ is called a* realization *or* model *of $L(\mathcal{P})$.*

B. Finite State Transducers

FSTs extend discrete event systems by nondeterministically generating a sequence of outputs; this is done by augmenting each transition with a regular output language.

Definition 2 (Finite State Transducer)**.** *An FST is a tuple $\mathcal{A} = (\mathsf{S}, \mathsf{s}_{\mathrm{init}}, \mathbf{I}, \mathbf{O}, \mathsf{Trans}, \mathsf{S}_{\mathrm{final}})$ where*

- S *is a finite set of* states*;*
- $\mathsf{s}_{\mathrm{init}} \in \mathsf{S}$ *is the* initial state*;*
- $\mathbf{I}$ *is a finite set of* inputs*;*
- $\mathbf{O}$ *is a finite set of* outputs*;*
- $\mathsf{Trans} : \mathsf{S} \times \mathrm{Re}(\mathbf{I}^*) \times \mathrm{Re}(\mathbf{O}^*) \to \mathsf{S}$ *is a partial* transition *function;*
- $\mathsf{S}_{\mathrm{final}} \subseteq \mathsf{S}$ *is a finite set of* final states.

Specially, the FST is normal *if the input and output labels on the transitions are $\mathbf{I} \cup \{\varepsilon\}$ and $\mathbf{O} \cup \{\varepsilon\}$, respectively.*

The definitions of *input/output words*, *executions*, *acceptance*, and *realization* are similar to discrete event systems (Definition 1). The automaton $\mathcal{A}_{\mathrm{in}}$ derived by removing the output symbols of the FST $\mathcal{A}$ is called *the input automaton* of $\mathcal{A}$; its acceptable language is called the *input languages* accepted by $\mathcal{A}$, denoted by $L_{\mathrm{in}}(\mathcal{A})$; similarly, the automaton $\mathcal{A}_{\mathrm{out}}$ derived by removing the input symbols of the FST $\mathcal{A}$ is called the output automaton of $\mathcal{A}$; its acceptable language is called *output languages* generated by $\mathcal{A}$, denoted by $L_{\mathrm{out}}(\mathcal{A})$.

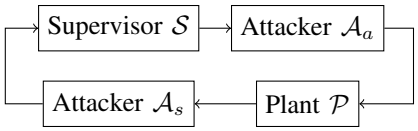


Fig. 2: Supervisor control without a secure channel.

C. Regular Relations

An FST $\mathcal{A}$ defines a relation $\mathcal{R}_{\mathcal{A}}$ between inputs $\mathbf{I}^*$ and outputs $\mathbf{O}^*$ as follows: $(\mathsf{i},\mathsf{o}) \in \mathcal{R}_{\mathcal{A}}$ if and only if there exists an execution $(\mathsf{s}_{\text{init}},\mathsf{i}_0,\mathsf{o}_0,\mathsf{s}_0)(\mathsf{s}_0,\mathsf{i}_1,\mathsf{o}_1,\mathsf{s}_1)\dots(\mathsf{s}_{n-2},\mathsf{i}_{n-1},\mathsf{o}_{n-1},\mathsf{s}_{n-1})$ such that $\mathsf{i}=\mathsf{i}_0\mathsf{i}_1\dots\mathsf{i}_{n-1}$ and $\mathsf{o}=\mathsf{o}_0\mathsf{o}_1\dots\mathsf{o}_{n-1}$. This relation is called a *regular* relation between $\mathbf{I}^*$ and $\mathbf{O}^*$. On the other hand, a relation $\mathcal{R}\subseteq\mathbf{I}^*\times\mathbf{O}^*$ is *regular*, only if it is realized by a finite state transducer. Clearly, $L_{\text{out}}(\mathcal{A})=\mathcal{R}(\mathbf{I}^*)$ and $L_{\text{in}}(\mathcal{A})=\mathcal{R}^{-1}(\mathbf{O}^*)$. Specially, a discrete event system $\mathcal{M}$ defines a regular relation $\mathcal{R}_{\mathcal{M}}=\big\{(I,I)\mid I\in L(\mathcal{M})\big\}$ on $\text{Re}(\mathbf{I}^*)\times\text{Re}(\mathbf{I}^*)$.

Despite the addition of an output, the computational complexity of solving problems on FSTs does not increase significantly from discrete event systems. A discrete event system can be lifted to a special FST with identical inputs and outputs. On the other hand, a normalized FST can be viewed as a discrete event system with labels in the set $(\mathbf{I}\cup\{\varepsilon\})\times(\mathbf{O}\cup\{\varepsilon\})$.

III. System Model and Problem Description

In this work, we consider the setup from Figure 2. Here, the supervisor $\mathcal{S}$, modeled by an FST, controls the behavior of the plant $\mathcal{P}$ by observing the symbols that the plant generates and then sending the possible control symbols back to the plant. In a companion paper [8], we show how such attacks can be modeled as nondeterministic FSTs; the attack FSTs can regularly rewrite an word, i.e., replace a symbol nondeterministically with an arbitrary word taken from some predefined regular language, including, for example, injection, replacement and deletion. In addition, we show that such FSTs can be used to capture a very wide class of attacks including all previously considered attacks on discrete-event systems (e.g., false data-injection, nondeterministic denial of service, rewriting), as well as additional attacks reported in recent security incidents (e.g., replay attack). In such models, the nondeterminism captures all possible actions of the attacker for a specific set of compromised resources (e.g., sensors, actuators), as well as all potential limitations imposed on the attacker's actions by the system design (e.g., the use of cryptographic primitives on some communication messages to prevent false-data inserting attacks over the network).

Consequently, in this work we model network-based attacks on the information delivered to and from the supervisor by two FSTs $\mathcal{A}_a$ and $\mathcal{A}_s$ affecting the inputs and outputs of the plant, respectively (Figure 2). The control objective is to restrict the symbols passing to the plant $\mathcal{P}$ within some desired language $\mathcal{K}\subseteq L(\mathcal{P})$. We first consider the case where the communication is always subject to attack, as shown in Figure 2, and study the problem of computing the maximal controllable sub-language (MCSL) $\underline{\mathcal{K}}\subseteq\mathcal{K}$ of the desired language – i.e., the largest subset of the desired language that is achievable by some supervisor under the attacks. In addition, we study the synthesis problem for such attack-resilient supervisor.

In the presence of network-based attacks captured by $\mathcal{A}_a$ and $\mathcal{A}_s$, the MCSL $\underline{\mathcal{K}}$ can be significantly smaller than the desired language $\mathcal{K}$. Thus, we also study the controllability problem with an intermittently accessible secure channel that works asynchronously with the insecure channel subject to the attacker $\mathcal{A}_a$, as shown in Figure 3. Suppose that the supervisor wants to ensure that any controls in $\mathcal{K}$ is sent to the plant safely – i.e., the plant receives words in $\mathcal{K}$ under all possible attacks. Clearly, if $\mathcal{K}\not\subseteq\underline{\mathcal{K}}$, then sending a control in $I\in\mathcal{K}\backslash\underline{\mathcal{K}}$ entirely through the insecure channel is undesirable. One solution is to send any word $I\in K$ through a secure channel, where security is usually enforced by encryption. However, this can impose excessive communication and computation overheads, limiting its use in resource-constrained systems.

A more cost-efficient way is to find another control $J\in\underline{\mathcal{K}}$ that only mismatches with I in short fragments; namely, there exist compositions $I=I_1I_2I_3I_4I_5...$ and $J=I_1J_2I_3J_4I_5...$, where $I_i,J_i\in\mathbf{I}^*$, such that for all $n\in\mathbb{N}$, $|I_{2n}|\le l_1$ and $|J_{2n}|\le l_2$ for some $l_1,l_2\in\mathbb{N}$. The idea is that in this case, the supervisor can send via insecure channel fragment-by-fragment the control $I_1'J_2'I_3'J_4'I_5'\dots$ that is modified from the original controls $I=I_1I_2I_3I_4I_5...$, and always stays within $\underline{\mathcal{K}}$ under the attacks (in Section V, we show how to compute these fragments from I and J). On the other hand, the pairs $\{(I_{2n},J_{2n})\}_{n\in\mathbb{N}}$ indicating the difference between I and J is sent via the secure channel, where J_{2n} serves as anchors for the restoration. Accordingly, the *authenticator* only receives a control within $\underline{\mathcal{K}}$ via the insecure channel; in certain cases, due to attacks it will receive J and can restore the original control I, which may be $\in\mathcal{K}\backslash\underline{\mathcal{K}}$, using the secure transmissions.

As the restoration of J to I is only performed on the small fragments J_{2n}, the secure channel merely needs the capacity of intermittently transmitting a pair of words of length less than l_1 and l_2. Note that in this scenario, the transmission of the anchors J_{2n} is the overhead cost (i.e., additional communication packets). Therefore, if $l_2\le l_1$, the savings of not having to secure every transmitted symbol (i.e., in the order of l_1+l_2) will be large due to the very high cost of protecting communication packets (e.g., with the use of standard cryptographic primitives). For example, let us assume that transmissions in the secure and insecure channels happen asynchronously and take the time to send each of fragments $\{I_n'J_n'\}_{n\in\mathbb{N}}$, as well as to send each of the pairs $\{(I_{2n},J_{2n})\}_{n\in\mathbb{N}}$ as 1. Then, the transmission of the secure channel is only needed every other time. For other time measures, the maximal frequency constraints on the secure channel changes accordingly.

Assumptions: We assume that $L_{\text{out}}(\mathcal{A}_a)=L_{\text{in}}(\mathcal{A}_s)=L(\mathcal{P})$ – i.e., the input attacks cannot generate words unacceptable to the plant and the output attacks does not receive those words. The latter is achievable by trimming the attack model $\mathcal{A}_s$; the former is achievable by restricting the plant — the results derived in the rest of paper hold for $L_{\text{out}}(\mathcal{A}_a)$ instead of $L(\mathcal{P})$ when $L_{\text{out}}(\mathcal{A}_a)\neq L(\mathcal{P})$. In addition, to

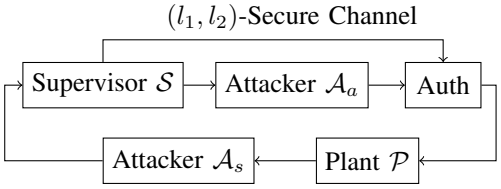


Fig. 3: Supervisor control with a secure channel.

simplify our presentation, we assume that for the plant $\mathcal{P}$, the supervisor $\mathcal{S}$, and the input and output attacks $\mathcal{A}_a, \mathcal{A}_s$, all states are final $\mathsf{S}_{\text{final}} = \mathsf{S}$, i.e., both the sets of their inputs and outputs are prefix-closed. In addition, the desired language is also prefix-closed $\mathcal{K} = \bar{\mathcal{K}}$ and regular. The regularity of $\mathcal{K}$ is to ensure that it is controllable by supervisors modeled as FSTs. The prefix-closeness requires that the supervision can be implemented step-by-step.

IV. Maximal Controllable Sub-Language

In this section, we study the problem of computing the MCSL of a desired language $\mathcal{K} \subseteq L(\mathcal{P})$, under the presence of input and output attacks, and without the use of secure channel, as shown in Figure 2.

Definition 3. *Let $\mathcal{K}$ be the desired language of the plant $\mathcal{P}$ with $\mathcal{K} \subseteq L(\mathcal{P})$, and $\mathcal{A}_a$ and $\mathcal{A}_s$ be the FSTs modeling attacks on the plant's input and output. The desired language $\mathcal{K}$ is controllable if there exists a supervisor $\mathcal{S}$ such that*

$$L(\mathcal{P}|\mathcal{S}, \mathcal{A}_a, \mathcal{A}_s) = \mathcal{K},$$

where $L(\mathcal{P}|\mathcal{S}, \mathcal{A}_a, \mathcal{A}_s) = \mathcal{K}$ is the language executed by the plant $\mathcal{P}$ in the presence of supervisor $\mathcal{S}$, and attacks $\mathcal{A}_a$, $\mathcal{A}_s$. The controllable language $\underline{\mathcal{K}}$ is the MCSL of $\mathcal{K}$, denoted by $\underline{\mathcal{K}} \subseteq_{\max} K$, if every controllable sub-language of $\mathcal{K}$ is contained in $\tilde{K}$. The desired language $\mathcal{K}$ is controllable if and only if $\mathcal{K} \subseteq \underline{\mathcal{K}}$.

The computation of MCSL $\underline{\mathcal{K}}$ of the desired language $\mathcal{K}$ depends on the operator $\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}$. For any words $k, k' \in L_{\text{out}}(\mathcal{A}_a)$, it holds that $k' \in \mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(k)$ if and only if $\mathcal{A}_a^{-1}(k') = \mathcal{A}_a^{-1}(k)$ – i.e., k and k' may result from the same supervisory control. Therefore, the following lemma holds.

Lemma 1. *For any $k, k' \in L_{\text{out}}(\mathcal{A}_a)$, it holds that (i) $k \in \mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(k)$, and (ii) $k' \in \mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(k) \Leftrightarrow \mathcal{A}_a^{-1}(k) = \mathcal{A}_a^{-1}(k') \Leftrightarrow k \in \mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(k')$.*

For any language $\mathcal{K}$, let $\mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$ be the least fixed point (LFP) of $\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}$ containing $\mathcal{K}$ – i.e., the smallest superset of $\mathcal{K}$ such that $\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}\big(\mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})\big) = \mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$. As $L_{\text{out}}(\mathcal{A}_a)$ is a fixed point of $\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}$, such an LFP exists. The LFP is characterized by the following lemma.

Lemma 2 (Least Fixed Point)**.** *For any $\mathcal{K} \subseteq L_{\text{out}}(\mathcal{A}_a)$, the LFP of $\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}$ is given by $\mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K}) = \bigcup_{i=0}^{\infty} \mathcal{R}^{i}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$, where $\mathcal{R}^{i}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}$ is the n-fold composition of $\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}$. Clearly, for any $i \in \mathbb{N}$, $\mathcal{R}^{i}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K}) \subseteq \mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$.*

When $\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}$ is idempotent on $\mathcal{K}$, namely, there exists n such that $\mathcal{R}^{n+1}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K}) = \mathcal{R}^{n}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$, then the LFP can be computed in finite time by $\mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K}) = \mathcal{R}^{n}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$. Otherwise, computing $\mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$ may be challenging.

Algorithm 1 Design of a Supervisor for Maximal Controllable Sub-Language under Input and Output Attacks

Require: Plant $\mathcal{P}$, input attacker $\mathcal{A}_a$, output attacker $\mathcal{A}_s$, Desire language $\mathcal{K}$.
1: Compute MCSL by (1) and find a model $\mathcal{M}_{\underline{\mathcal{K}}}$.
2: Compute supervisor $\mathcal{S}$ by (2).
3: **return** Supervisor $\mathcal{S}$.

In [8], we show that given the MCSL $\underline{\mathcal{K}}$ satisfying $\underline{\mathcal{K}} = \mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\underline{\mathcal{K}})$, the supervisor designed by $\mathcal{S} = \mathcal{A}_s^{-1} \circ \mathcal{M}_{\underline{\mathcal{K}}} \circ \mathcal{A}_a^{-1}$ restricts the language received by the plant to $\underline{\mathcal{K}}$. Similar to [8] and differing from previous works [13], [15], [10], the MCSL only depends on the input attack $\mathcal{A}_a$, because the supervisor, modeled by an FST instead of an automaton, has more power in generating control commands, and can totally counter the effect of the output attacker by composing with $\mathcal{A}_s^{-1}$. Given $\mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K})$, the computation of the MCSL of $\mathcal{K}$ and the supervisor can be done as captured by Theorem 1 and Algorithm 1.

Theorem 1 (Maximal Controllable Sub-Language)**.** *The maximal controllable sub-language (MCSL) of the desired regular language $\mathcal{K} \subseteq L_{\text{in}}(\mathcal{P})$ under the input and output attacks modeled by FSTs $\mathcal{A}_a$ and $\mathcal{A}_s$ is*

$$\underline{\mathcal{K}} = \mathcal{K} \backslash \mathcal{R}^{\infty}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}\big(\mathcal{R}_{\mathcal{A}_a^{-1} \circ \mathcal{A}_a}(\mathcal{K}) \backslash \mathcal{K}\big), \tag{1}$$

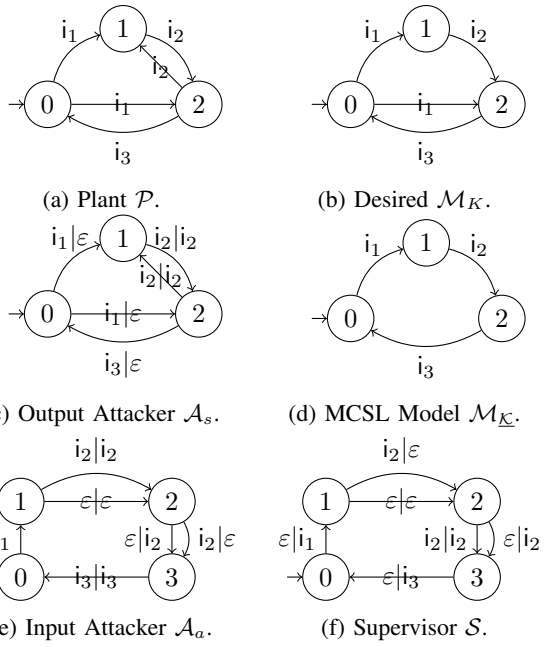
under the supervisor

$$\mathcal{S} = \mathcal{A}_s^{-1} \circ \mathcal{M}_{\underline{\mathcal{K}}} \circ \mathcal{A}_a^{-1}. \tag{2}$$

Example 1 (Supervisor design for intermittent secure transmissions)**.** *Consider the supervisory control of a plant $\mathcal{P}$ shown in Figure 4a with the set of symbols $\mathbf{I} = \{\mathsf{i}_1, \mathsf{i}_2, \mathsf{i}_3\}$. The (prefix-closed) desired language is $\mathcal{K} = (\mathsf{i}_1(\mathsf{i}_2, \varepsilon)\mathsf{i}_3)^*$, modeled by an automaton shown in Figure 4b. The input and output attacks $\mathcal{A}_a$ and $\mathcal{A}_s$ of the plant are modeled by FSTs shown in Figures 4c and 4e, respectively. By Theorem 1, the MCSL is $\underline{\mathcal{K}} = \overline{(\mathsf{i}_1\mathsf{i}_2\mathsf{i}_3)^*}$, which is strictly contained in $\mathcal{K}$, as shown in Figure 4d. It is easy to check the maximality of $\underline{\mathcal{K}}$. The supervisor computed by Algorithm 1 is shown in Figure 4f after minor simplification.* ◁

Remark 1. *By Theorem 1, attacks on the output do not affect controllability, as opposed to previous works [10], [11], [13]. This is caused by the fact that the plant is deterministic, so its state is known from the perspective of the supervisor, and the supervisor as an FST can generate controls for the next step by itself without using the sensing information of the plant. These sensing information will become useful to learn the state of the plant when it is nondeterministic, which will be studied in the future, with initial results reported in [8].*

Remark 2. *Note that the supervisor derived in Example 1 is nondeterministic — there are transition with ε input symbol that can be triggered spontaneously. The controllability theorem guarantees that in the presence of attacks, the union of all possible words received by the plant under all these allowable controls is exactly the desired language $\mathcal{K}$. In implementation, the nondeterminism can be resolved by choosing one of the allowable controls. Accordingly, the possible words received by the plant is contained in $\mathcal{K}$.*



(a) Plant $\mathcal{P}$. (b) Desired $\mathcal{M}_K$.

(c) Output Attacker $\mathcal{A}_s$. (d) MCSL Model $\mathcal{M}_{\underline{\mathcal{K}}}$.

(e) Input Attacker $\mathcal{A}_a$. (f) Supervisor $\mathcal{S}$.

Fig. 4: Example supervisor for input and output attacks.

V. Controllability with Secure Channel Access

In this section, we extend the results from Section IV to the case that the supervisor has access to a secure channel. Recalling Section III, the supervisor can decompose the desired control $I = I_1I_2I_3I_4I_5... \in \mathcal{K}\backslash\underline{\mathcal{K}}$ into fragments and find a segmentally mismatching but attack-resilient control $J = I_1J_2I_3J_4I_5....$ To counter the attacks, the supervisor can send the modified control fragments $I_1'J_2'I_3'J_4'I_5'\ldots$ with $I_1' = \mathcal{R}_{\mathcal{A}_a^{-1}}(I_1)$, $I_1'J_1' = \mathcal{R}_{\mathcal{A}_a^{-1}}(I_1J_1)$, and so on, via the insecure channel, and the pairs $\{(I_{2n}, J_{2n})\}_{n\in\mathbb{N}}$ to indicate the difference between I and J via the secure channel. Accordingly, from the insecure channel the authenticator will receive a word in $\mathcal{R}_{\mathcal{A}_a^{-1}\circ\mathcal{A}_a}(J) \subseteq \underline{\mathcal{K}}$. Upon receiving exactly J, it can use the pairs $\{(I_{2n}, J_{2n})\}_{n\in\mathbb{N}}$ to restore J back to I, where J_{2n} serves as anchors for the restoration. In other words, the control received by the plant is always in $\mathcal{K}$ and is exactly the original one I when the attacker $\mathcal{A}_a$ takes certain attacks. When $\mathcal{R}_{\mathcal{A}_a^{-1}\circ\mathcal{A}_a}(J) \subseteq \underline{\mathcal{K}} = \{J\}$ – i.e., J is revertible from all possible attacks of $\mathcal{A}_a$, the recovery to the original control I is guaranteed.

The above solution depends critically on whether for any word $\mathcal{K}$, we can find a word in $\underline{\mathcal{K}}$ with bounded fragmentary mismatches. This gives rise to a difference measure from $\mathcal{K}$ to $\underline{\mathcal{K}}$, which we define as *accessibility*.

A. Accessibility

To quantitatively calculate the accessibility for any two regular languages $\mathcal{K} \subseteq \underline{\mathcal{K}}$, we propose a method of building automata models $\mathcal{M}_{\mathcal{K}}$ and $\mathcal{M}_{\underline{\mathcal{K}}}$ realizing $\mathcal{K}$ and $\underline{\mathcal{K}}$, respectively, with the model $\mathcal{M}_{\underline{\mathcal{K}}}$ contained in the model $\mathcal{M}_{\mathcal{K}}$, and then computing it on the two graph models $\mathcal{M}_{\underline{\mathcal{K}}}$ and $\mathcal{M}_{\mathcal{K}}$ using graph-theoretic tools.

Definition 4 (Sub-Automata)**.** *Let* $\mathcal{P} = (\mathsf{S}, \mathsf{s}_{\text{init}}, \mathbf{I}, \mathsf{Trans}, \mathsf{S}_{\text{final}})$ *be an FSA with* $\mathsf{S}_{\text{final}} = \mathsf{S}$*. We call* $\mathcal{P}' = (\mathsf{S}', \mathsf{s}'_{\text{init}}, \mathbf{I}', \mathsf{Trans}')$ *with* $\mathsf{S}'_{\text{final}} = \mathsf{S}'$ *a sub-automata of* $\mathcal{P}$*, denoted by* $\mathcal{P}' \subseteq \mathcal{P}$*, if* $\mathsf{S}' \subseteq \mathsf{S}$*,* $\mathsf{s}'_{\text{init}} = \mathsf{s}_{\text{init}}$*,* $\mathbf{I}' \subseteq \mathbf{I}$*, and* $\mathsf{Trans}'(\mathsf{s}', \mathsf{i}') = \mathsf{Trans}(\mathsf{s}', \mathsf{i}')$ *for any* $\mathsf{s}' \in \mathsf{S}'$ *and* $\mathsf{i}' \in \mathbf{I}'$*. For any prefix-closed* $\mathcal{K} \subseteq L(\mathcal{P})$*, there exists a maximal sub-automaton* $\mathcal{P}' \subseteq \mathcal{P}$ *such that* $\mathcal{K} = L(\mathcal{P}')$*, and all such sub-automata are sub-automata of* $\mathcal{P}'$*.*

Algorithm 2 Constructing $\mathcal{M}_{\underline{\mathcal{K}}} \subseteq \mathcal{M}_{\mathcal{K}}$ for $\mathcal{K}$ and $\underline{\mathcal{K}}$.

Require: Languages $\underline{\mathcal{K}} \subseteq \mathcal{K}$.
1: Find minimal realization $\mathcal{M}_{\mathcal{K}}$ and $\mathcal{M}_{\underline{\mathcal{K}}}$ of $\mathcal{K}$ and $\underline{\mathcal{K}}$.
2: Convert $\mathcal{M}_{\mathcal{K}}$ and $\mathcal{M}_{\underline{\mathcal{K}}}$ to FSTs by adding ε as output and input symbol for each transition, respectively.
3: $\mathcal{M} = \mathcal{M}_{\mathcal{K}} \circ \mathcal{M}_{\underline{\mathcal{K}}}$.
4: Trim off transitions with input symbol ε in $\mathcal{M}$.
5: **return** $\mathcal{M}_{\text{in}}$ and $\mathcal{M}_{\text{out}}$.

To quantify the difference between two automata $\mathcal{P}' \subseteq \mathcal{P}$, we introduce the definition of (l_1, l_2)-step accessibility.

Definition 5 ((l_1, l_2)-Step Accessibility)**.** *For two automata* $\mathcal{P}'$ *and* $\mathcal{P}$*, such that* $\mathcal{P}' \subseteq \mathcal{P}$*, we say that* $\mathcal{P}$ *is* (l_1, l_2)*-accessible from* $\mathcal{P}'$ *if there is no execution* $(\mathsf{s}_0, \mathsf{i}_0, \mathsf{s}_1)\ldots(\mathsf{s}_{n-1}, \mathsf{i}_{n-1}, \mathsf{s}_n) \subseteq \mathsf{Trans}\backslash\mathsf{Trans}'$ *with* $n > l_2$*. In addition, for any such execution with* $n \leq l_2$*, there exists an execution* $(\sigma_0, \mathsf{i}_0, \sigma_1)\ldots(\sigma_{m-1}, \mathsf{i}_{m-1}, \sigma_m) \subseteq \mathsf{Trans}$ *with* $m \leq l_1$*,* $\sigma_0 = \mathsf{s}_0$ *and* $\sigma_m = \mathsf{s}_n$*.*

For an FSA $\mathcal{P}$, let the induced the directed graph be $\text{Graph}_{\mathcal{P}} = (\mathsf{S}, \{(\mathsf{s}, \mathsf{s}') \in \mathsf{S}^2 \mid (\mathsf{s}, \mathsf{i}, \mathsf{s}') \in \mathsf{Trans}\})$. Accessibility can be checked by the following graph-theoretic condition.

Theorem 2 ((l_1, l_2)-Step Accessibility)**.** *For two automata* $\mathcal{P}'$ *and* $\mathcal{P}$*, such that* $\mathcal{P}' \subseteq \mathcal{P}$*,* $\mathcal{P}$ *is* (l_1, l_2)*-accessible from* $\mathcal{P}'$ *if and only if (i) the subtracted graph* $\text{Graph}_{\mathcal{P}}\backslash\text{Graph}_{\mathcal{P}'}$ *is a tree with paths no longer than* l_1*; and (ii) for any such path, there is a path no longer than* l_2 *with same start and end in* $\text{Graph}_{\mathcal{P}'}$*.*

These two graph-theoretic conditions in Theorem 2 can be examined with well-studied algorithms (e.g., from [20]).

B. Supervisor Design for Attack-Resilience

To measure the distance between $\mathcal{K}$ and $\underline{\mathcal{K}}$, we build two automata $\mathcal{M}_{\underline{\mathcal{K}}} \subseteq \mathcal{M}_{\mathcal{K}}$, where $\underline{\mathcal{K}}$ is the MCSL of $\mathcal{K}$, as given in Theorem 1. This is done as follows. Let $\mathcal{M}_{\underline{\mathcal{K}}}$ and $\mathcal{M}_{\mathcal{K}}$ be the minimal realization of $\mathcal{K}$ and $\underline{\mathcal{K}}$. Then we can convert them into FSTs by adding ε as input and output symbol for each transition, respectively. Let $\mathcal{M} = \mathcal{M}_{\mathcal{K}} \circ \mathcal{M}_{\underline{\mathcal{K}}}$ and trim the transitions of $\mathcal{M}$ with labels ε input symbol. Clearly, we have $\mathcal{M}_{\text{out}} \subseteq \mathcal{M}_{\text{in}}$ $L(\mathcal{M}_{\text{in}}) = \mathcal{K}$ and $L(\mathcal{M}_{\text{out}}) = \underline{\mathcal{K}}$. This is summarized by Algorithm 2. In the rest of this section, let $\mathcal{M}_{\mathcal{K}} = (\mathsf{S}, \mathsf{s}_{\text{init}}, \mathbf{I}, \mathsf{Trans}, \mathsf{S}_{\text{final}})$ be $\mathcal{M}_{\text{in}}$ and $\mathcal{M}_{\underline{\mathcal{K}}} = (\underline{\mathsf{S}}, \underline{\mathsf{s}_{\text{init}}}, \underline{\mathbf{I}}, \underline{\mathsf{Trans}}, \underline{\mathsf{S}_{\text{final}}})$ be $\mathcal{M}_{\text{out}}$.

The transitions in $\mathcal{M}_{\underline{\mathcal{K}}}$ are resilient to attacks, thus do not require transmissions from the secure channel, while other transitions in $\mathcal{M}_{\mathcal{K}}$ are non-resilient and need secure transmission. By Definition 5, if $\mathcal{M}_{\mathcal{K}}$ is (l_1, l_2)-accessible from $\mathcal{M}_{\underline{\mathcal{K}}}$, then for any execution of $\mathcal{M}_{\mathcal{K}}$, there is at most l_1 consecutive non-resilient transitions and it is replaceable by an l_2 secure transitions. This pair of words, transmitted through

Algorithm 3 Design supervisor and authenticator with active access to secure channel

Require: Plant $\mathcal{P}$, input attacker $\mathcal{A}_a$, output attacker $\mathcal{A}_s$, Desire language $\mathcal{K}$.
1: Compute MCSL by (1) and find FSA models $\mathcal{M}_{\underline{\mathcal{K}}} \subseteq \mathcal{M}_K$.
2: Compute $\mathcal{N}_K$ by (3) and supervisor $\mathcal{S}$ by (4).
3: **return** Supervisor $\mathcal{S}$.

the (l_1, l_2)-secure channel, are used to repair the corrupted word from the insecure channel. This is summarized by the following theorem.

Theorem 3 (Controllability with Intermittent Active Access to Secure Channel)**.** *The desired regular language $\mathcal{K} \subseteq L(\mathcal{P})$ is controllable with active (l_1, l_2)-secure channel access if $\mathcal{M}_{\mathcal{K}}$ is (l_1, l_2)-accessible from $\mathcal{M}_{\underline{\mathcal{K}}}$, where $\underline{\mathcal{K}}$ is the MCSL of $\mathcal{K}$ as given by Theorem 1, and $\mathcal{M}_{\mathcal{K}}$ and $\mathcal{M}_{\underline{\mathcal{K}}}$ are given by Algorithm 2.*

Now, the supervisor resilient to such attacks can be realized by an FST with two outputs for the secure and insecure channel, respectively. Specifically, let $\mathcal{M}_{\mathcal{K}} = (\mathsf{S}, \mathsf{s}_{\rm init}, \mathbf{I}, \mathsf{Trans}, \mathsf{S}_{\rm final})$ and $\mathcal{M}_{\underline{\mathcal{K}}} = (\underline{\mathsf{S}}, \underline{\mathsf{s}_{\rm init}}, \underline{\mathbf{I}}, \underline{\mathsf{Trans}}, \underline{\mathsf{S}_{\rm final}})$ be deterministic FSA models for $\underline{\mathcal{K}} \subseteq \mathcal{K}$, with $\mathcal{M}_{\underline{\mathcal{K}}} \subseteq \mathcal{M}_{\mathcal{K}}$. We construct an FST $\mathcal{N}_{\mathcal{K}} = (\mathsf{S} \times \underline{\mathsf{S}}, (\mathsf{s}_{\rm init}, \underline{\mathsf{s}_{\rm init}}), \mathbf{I}, \mathbf{I} \times \{+, -\}, \mathsf{Trans}', \mathsf{S}_{\rm final} \times \underline{\mathsf{S}_{\rm final}})$ with

$$\begin{aligned}\mathsf{Trans}' = &\big\{\big((\underline{\mathsf{s}}, \underline{\mathsf{s}}), \mathsf{i}, (\diamond, \mathsf{i}), (\underline{\mathsf{s}}', \underline{\mathsf{s}}')\big) \mid (\underline{\mathsf{s}}, \mathsf{i}, \underline{\mathsf{s}}') \in \underline{\mathsf{Trans}}\big\} \\ &\big\{\big((\mathsf{s}, \underline{\mathsf{s}}), \mathsf{i}, (\mathsf{i}, \diamond_1), (\mathsf{s}', \underline{\mathsf{s}})\big) \mid (\mathsf{s}, \mathsf{i}, \mathsf{s}') \in \mathsf{Trans} \backslash \underline{\mathsf{Trans}}, \underline{\mathsf{s}} \in \underline{\mathsf{S}}\big\} \\ &\big\{\big((\underline{\mathsf{s}}, \underline{\mathsf{s}}'), \mathsf{i}, (\mathsf{i}, \diamond_2), (\underline{\mathsf{s}}, \underline{\mathsf{s}}'')\big) \mid (\underline{\mathsf{s}}', \mathsf{i}, \underline{\mathsf{s}}'') \in \underline{\mathsf{Trans}}, \underline{\mathsf{s}} \in \underline{\mathsf{S}}\big\}\end{aligned} \tag{3}$$

The output $(\diamond, \mathsf{i})$ stands for sending i over the insecure channel while closing the secure channel; $(\mathsf{i}, \diamond_1)$ and $(\mathsf{i}, \diamond_2)$ denote sending i to the first and second component of the pair of words transmission via the secure channel while closing the insecure channel. Then the supervisor $\mathcal{S}$ is constructed by the composition

$$\mathcal{S} = \mathcal{N}_K \circ_2 \mathcal{A}_a^{-1} \tag{4}$$

similar to [8]. Here, $\circ_2$ stands for the composition between the input of $\mathcal{A}_a$ with the second component of the output of $\mathcal{N}_K$, while keeping the first component.

Example 2 (Supervisor design for maximal controllable sub-languages)**.** *Following Example 1, the desired language $\mathcal{K} = \overline{(\mathsf{i}_1(\mathsf{i}_2, \varepsilon)\mathsf{i}_3)^*}$ is achievable with intermittent access to $(2, 1)$-secure channel. The supervisor designed by Algorithm 3 is shown in Figure 5, in which the upper half of the FST is identical to Figure 4f handling words in MCSL $\underline{\mathcal{K}}$ and the lower half handles the word in $\mathcal{K} \backslash \underline{\mathcal{K}}$. Upon i_1 at the beginning, the supervisor may actively activate the secure channel to send $(\mathsf{i}_1\mathsf{i}_2, \mathsf{i}_1)$, so that if the plant receives $\mathsf{i}_1\mathsf{i}_2\mathsf{i}_3$, the authenticator repairs it back to $\mathsf{i}_1\mathsf{i}_3$.* ◁

VI. Conclusions

In this work, we have studied the supervisory control of discrete event systems under regularly-rewriting attacks on their actuators and sensors with intermittent authentication. First, we have solved the problem of computing the maximal controllable sub-language (MCSL) of a desired language and proposed a design method for such attack-resilient supervisor,

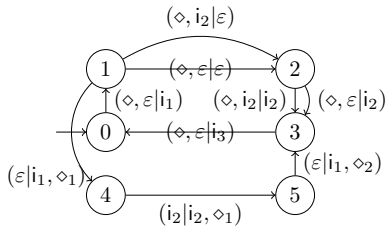


Fig. 5: Supervisor $\mathcal{S}$ with active access to secure channel.

when there is no protection for communication between the plant and supervisor. We have then extended these results for the case when the supervisor has active but intermittent access to a size-limited secure channel that protects its communication with the plant.

References

[1] C. G. Cassandras, "Smart Cities as Cyber-Physical Social Systems," *Engineering*, vol. 2, no. 2, pp. 156–158, 2016.
[2] Z. Jiang, M. Pajic, and R. Mangharam, "Cyber-Physical Modeling of Implantable Cardiac Medical Devices," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 122–137, Jan 2012.
[3] Y. Wang, Z. Huang, S. Mitra, and G. E. Dullerud, "Differential privacy in linear distributed control systems: Entropy minimizing mechanisms and performance tradeoffs," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 118–130, 2017.
[4] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, "Attack models and scenarios for networked control systems," in *Conf. on High Confid. Net. Sys. (HiCoNS)*, 2012, pp. 55–64.
[5] Y. Mo, T.-H. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber–physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.
[6] M. Pajic, I. Lee, and G. J. Pappas, "Attack-Resilient State Estimation for Noisy Dynamical Systems," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 82–92, 2017.
[7] H. Fawzi, P. Tabuada, and S. Diggavi, "Secure Estimation and Control for Cyber-Physical Systems Under Adversarial Attacks," *IEEE Trans. on Automatic Control*, vol. 59, no. 6, pp. 1454–1467, 2014.
[8] Y. Wang, A. K. Bozkurt, and M. Pajic, "Attack-resilient supervisory control of discrete-event systems," *arXiv preprint arXiv:1904.03264, submitted to IEEE Transactions on Automatic Control*, 2019.
[9] J. S. Warner and R. G. Johnston, "A simple demonstration that the global positioning system (gps) is vulnerable to spoofing," *Journal of Security Administration*, vol. 25, no. 2, pp. 19–27, 2002.
[10] M. Wakaiki, P. Tabuada, and J. P. Hespanha, "Supervisory Control of Discrete-event Systems under Attacks," *arXiv:1701.00881*, 2017.
[11] R. Su, "Supervisor synthesis to thwart cyber attack with bounded sensor reading alterations," *Automatica*, vol. 94, pp. 35–44, 2018.
[12] R. M. Goes, E. Kang, R. Kwong, and S. Lafortune, "Stealthy deception attacks for cyber-physical systems," in *IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 4224–4230.
[13] 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.
[14] M. Mohri, "Weighted Finite-State Transducer Algorithms. An Overview," in *Formal Languages and Applications*, Springer Berlin Heidelberg, 2004, vol. 148, pp. 551–563.
[15] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. New York, NY: Springer, 2008.
[16] V. Lesi, I. Jovanov, and M. Pajic, "Security-aware scheduling of embedded control tasks," *ACM Trans. Embed. Comput. Syst.*, vol. 16, no. 5s, pp. 188:1–188:21, Sep. 2017.
[17] ——, "Network scheduling for secure cyber-physical systems," in *2017 IEEE Real-Time Systems Symposium (RTSS)*, Dec 2017, pp. 45–55.
[18] I. Jovanov and M. Pajic, "Relaxing integrity requirements for attack-resilient cyber-physical systems," *IEEE Transactions on Automatic Control*, pp. 1–1, 2019, to appear.
[19] J. Sakarovitch and R. Thomas, "Elements of Automata Theory," p. 784, 2003.
[20] J. L. Gross and J. Yellen, *Handbook of graph theory*. CRC press, 2004.