Mitigation of Classes of Attacks using a Probabilistic Discrete Event System Framework ⋆

Ze Yang Wang * **Rômulo Meira-Góes** ** **Stéphane Lafortune** **
Raymond H. Kwong *

* *Department of Electrical and Computer Engineering, University of Toronto, Canada (e-mail: zeyang.wang@mail.utoronto.ca, kwong@control.utoronto.ca).*
** *Department of Electrical Engineering and Computer Science, University of Michigan, USA (e-mail: {romulo, stephane}@umich.edu)*

Abstract: Cyber-attack models and their respective mitigation strategies have been recently studied in the discrete event systems setting. Previous work focuses on whether unsafe behaviour can be prevented using supervisory control theory. When unsafe behaviour cannot be prevented with certainty, mitigation strageties are limited. This paper proposes the use of a probabilistic discrete event system (PDES) framework to incorporate a likelihood measure for unsafe behaviour in the attack models presented in Carvalho et al. (2018). The least-unsafe (LU) supervisor problem is introduced to minimize this unsafe likelihood measure and improve existing attack mitigation techniques. The LU supervisor problem under full observability is solved by reformulating it into an MDP problem, and a computable algorithm is developed. Lastly, the implementation of LU supervisors is discussed and illustrated with an example.

Keywords: Cyber-attacks; Mitigation; Probabilistic discrete event systems; Markov decision processes; Optimal control; Attack resilience

1. INTRODUCTION

Many of today's systems are controlled by embedded computers that interact closely with their physical counterparts through sensors and actuators. The increasingly tight and complex cyber-physical coupling introduces new risks that may be exploited by malicious intent (Ashibani and Mahmoud (2017)). It is becoming increasingly important to improve cyber-physical security to ensure safe and sustainable operations (Banerjee et al. (2012)).

In recent years, there have been many works that focus on modelling and analysing the effects of malicious attacks on controller-plant interactions using discrete event systems (DES), such as Lima et al. (2018), Carvalho et al. (2018), Rashidinejad et al. (2019), and Meira-Góes et al. (2019). In Carvalho et al. (2018), four types of attacks are considered: sensor erasure (SE), sensor insertion (SI), actuator enablement (AE), and actuator disablement (AD), as illustrated in Figure 1. The attacker may manipulate a predetermined subset of vulnerable sensors or actuators at any given opportunity, simulating the worst-case scenario. The attacker's actions are assumed to be unobservable. The intention is to bring the plant into an unsafe state, which the supervisor is designed to avoid at all costs. The authors developed algorithms in the DES framework that

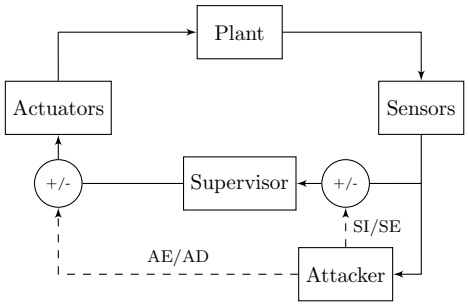


Fig. 1. Compromised closed-loop system, where the attacker manipulates the supervisor's observations (SE/SI) and control actions (AE/AD)

generate models for a compromised closed-loop system under each of the four attack types.

Building on the attack models, Carvalho et al. (2018) introduced the general-form (GF) safe controllability conditions to verify if a closed-loop system can detect and mitigate these attacks. The work assumes that attacks do not directly lead to immediate failure, but they do enable illegal behaviours that may lead to failure. The framework uses a detection module that attempts to deduce whether an unobservable attack action has happened based on the historical observable behaviour of the system. If an attack is detected, a "disable-all" mitigation strategy is used, where all controllable events are disabled to restrict the system behaviour as much as possible. If a system is GF safe controllable, the "disable-all" mitigation strategy

⋆ This work was partly supported by the Natural Sciences and Engineering Research Council of Canada (grant RGPIN-2015-04273) and by the US National Science Foundation (grants CNS-1738103 and CNS-1801342).

is effective in preventing the system from arriving at an unsafe state. If a system is not GF safe controllable, the "disable-all" mitigation strategy may be ineffective and may disable access to some safe paths which causes a higher likelihood of unsafe behaviour. This warrants a better mitigation strategy, which is the motivation of this paper.

In this work, the least-unsafe (LU) supervisor problem is posed with the goal of a better mitigation strategy. The problem considers a probabilistic discrete event system (PDES) with a predefined set of unsafe states and seeks to find the set of optimal supervisory actions that minimizes the probability of arriving at an unsafe state. In the PDES literature, the concept of optimal control has not been extensively studied, certainly not with the objectives presented in this paper. The conventional control objective is to satisfy a given specification, as for example in Pantelic et al. (2014). As an initial study, this paper presents the solution of the LU supervisor problem for fully observable systems. The results are then applied to the attack mitigation scenario to produce a supervisor that minimizes the probability of unsafe behaviour after an attack detection. The remaining sections are organized as follows. Section 2 provides an overview of the mathematical notations of PDES. The LU supervisor problem is formulated in Section 3. The solution to the LU supervisor problem is presented in Section 4. Section 5 discusses the implementation of the results. Lastly, Section 6 concludes with contributions, a brief comment on partially observable systems, and future work.

2. PROBABILISTIC DISCRETE EVENT SYSTEM

The formalism of PDES is less agreed upon and less studied compared to its deterministic counterpart. For example, the formalism in Garg et al. (1999) allows a termination probability in $[0, 1]$ at each state while Pantelic et al. (2014) allows termination probability of either 0 or 1. Another example is the difference in control mechanics between Chattopadhyay and Ray (2009), where disablement causes the event to self-loop with the same probability, and Garg et al. (1999) along with Pantelic et al. (2014), where disablement causes the redistribution of the event probabilities. In this work, we use the same framework as in Pantelic et al. (2014), which closely extends the DES framework in Wonham and Cai (2019).

A PDES is denoted as $G = (X, \Sigma, \delta, x_0, \rho)$ where X is the set of states, Σ is the set of events, $\delta : X \times \Sigma \to X$ is the partial transition function, x_0 is the initial state, and $\rho : \Sigma \times X \to [0, 1]$ is the state-wise event probability distribution. We define that $\rho(\sigma \,|\, x) = 0 \iff \delta(x, \sigma)!$ (i.e. $\delta(x, \sigma)$ is not defined).

$L(G) \subseteq \Sigma^*$ is the language generated by G. $L(G, x) \subseteq \Sigma^*$ is the language generated by G with initial state x. The corresponding p-language $L_p(G) : \Sigma^* \to [0, 1]$ is generated by the following,

(i) $L_p(G)(\epsilon) = 1$,

(ii) $L_p(G)(s\sigma) = \begin{cases} L_p(G)(s) \cdot \rho(\sigma|\delta(x_0, s)) & \text{if } s \in L(G) \\ 0 & \text{otherwise.} \end{cases}$

The conditional p-language of G of a string st given the prefix s is expressed as

$$L_p(G)(st|s) = \frac{L_p(G)(st)}{L_p(G)(s)}.$$

Controllable and uncontrollable events are denoted as Σ_c and Σ_{uc}, respectively, where $\Sigma = \Sigma_c \dot\cup \Sigma_{uc}$. The set of admissible control patterns is defined as $\Gamma := \{\gamma \in 2^\Sigma \,|\, \gamma \supseteq \Sigma_{uc}\}$. A supervisor is any mapping $V : L(G) \to \Gamma$. The automaton realization of V is denoted as S. The closed-loop system of G under the supervision of V is denoted as V/G. $L(V/G) \subseteq \Sigma^*$ is the closed-loop language generated by V/G. We denote $L(V/G, x) \subseteq \Sigma^*$ as the closed-loop language of V/G with initial state x.

There are two different control philosophies available for PDES as suggested in Lawford and Wonham (1993): *deterministic supervisor* and *probabilistic supervisor*. In this paper, we study the mitigation of attacks using a deterministic supervisor as it simplifies the problem and is suitable for this initial study on the mitigation of attacks using PDES. A probabilistic supervisor may be of interest when studying the attack model in the context of attacker-supervisor games. The p-language generated from G using supervisor $V : L(G) \to \Gamma$, denoted as $L_p(V/G)$, is defined as

(i) $L_p(V/G)(\epsilon) := 1$,

(ii) $L_p(V/G)(s\sigma) := \begin{cases} L_p(V/G)(s) \cdot p(\sigma|\delta(x_0, s), V(s)) \\ \quad \text{if } s \in L(V/G) \wedge \sigma \in V(s) \\ \quad \wedge s\sigma \in L(G) \\ 0 \quad \text{otherwise} \end{cases}$

where, given $s \in L(G)$,

$$p(\sigma|\delta(x_0, s), V(s)) := \begin{cases} \dfrac{\rho(\sigma|\delta(x_0, s))}{\sum_{\sigma' \in V(s)} \rho(\sigma'|\delta(x_0, s))} & \text{if } \sigma \in V(s) \\ 0 & \text{otherwise} \end{cases}.$$

3. PROBLEM FORMULATION

The objective is to find a supervisor that minimizes the total probability of a system, G, from entering the set of unsafe states $X_{us} \subseteq X$ starting from the initial state x_0. Let $L_{us}(G) = \{s \in L(G) \,|\, \delta(x_0, s) \in X_{us}\}$ be the *unsafe language* of G. It is assumed that the probabilistic information embedded in G has been measured by the designer. We make an important assumption about the set of unsafe states.

Assumption 1. All unsafe states X_{us} of a given plant G are terminating states and have no outgoing events: thus,

$$(\forall x \in X_{us})(\forall \sigma \in \Sigma) \quad \delta(x, \sigma)!.$$

We believe that such an assumption is reasonable because it signifies the system's inability to operate after a catastrophic failure.

Let $\mathcal{V}$ be the set of all supervisors with respect to G. The main problem of this paper is formulated as follows.

Problem 1. (The LU supervisor problem). Given a PDES G and its set of unsafe states $X_{us} \subseteq X$, find a *least-unsafe (LU) supervisor* V^*, if it exists, such that

$$\sum_{s\in L_{us}(V^*/G)} L_p(V^*/G)(s) = \inf_{V\in\mathcal{V}} \sum_{s\in L_{us}(V/G)} L_p(V/G)(s), \tag{1}$$

where $L_{us}(V/G) := L(V/G) \cap L_{us}(G)$ is the *unsafe language of G under the supervision of V*.

Let $P^G_{us}(V) = \sum_{s\in L_{us}(V/G)} L_p(V/G)(s)$, which is called the *total unsafe probability of G under the supervision of V*. Hence, the supervisor V^*, if it exists, achieves the minimum total unsafe probability given the system G. The LU supervisor problem is formulated without the complications of attack detection and partial observability, to focus on finding the optimal supervisor. Section 5 will discuss the implementation of the LU supervisor in the attack detection and mitigation framework.

4. SOLUTION AND COMPUTATIONAL METHODOLOGY

The approach in solving the LU supervisor problem under full observation is to reformulate it into a Markov decision process (MDP). Denote the set of states with a set of integers $X = \{0, 1, ..., I-1\}$, where I is the cardinality of X, $|X|$. We assume 0 to be the initial state. Let x_t be a random variable that realizes the state on X after t event evolutions. Note that in the DES under consideration, there is no sense of time. Thus, our "time" measure is the number of events that have occurred. In the MDP framework, the feedback policy $\mathbf{g} = \{g_0, g_1, ...\}$, where $g_t : X \to \Gamma$, is a state-based feedback, in contrast to the language-based supervisory control $V : L(G) \to \Gamma$. Note that the DES framework does not support a feedback controller dependent on t. Nevertheless, if a state-based feedback policy $\mathbf{g}$ is stationary, i.e. $\mathbf{g} = \{g, g, ...\}$, then $\mathbf{g}$ can be implemented by constructing the supervisor

$$V_{\mathbf{g}}(s) = g(\delta(0, s)) \quad \text{for all } s \in L(G). \tag{2}$$

Thus, hereafter, we only consider the class of stationary policies $\mathbf{g} = \{g, g, ...\}$.

4.1 The Transition Structure

Consider the stationary policy $\mathbf{g} = \{g, g, ...\}$. Let $\{x_t\}$ and $\{v_t\}$ be the resulting processes, where $v_t = g(x_t) \in \Gamma$. The processes under a fixed g can be described by the stochastic difference equation

$$x_{t+1} = f(x_t, v_t, \sigma_{t+1}), \quad v_t \in \Gamma, \quad \sigma_{t+1} \in \Sigma \tag{3}$$

where

$$f(x_t, v_t, \sigma_{t+1}) := \begin{cases} \delta(x_t, \sigma_{t+1}) & \text{if } \sigma_{t+1} \in v_t \\ undefined & \text{otherwise} \end{cases}$$

and the transition probabilities associated with f are

$$\begin{aligned} P[f(i, v, \sigma) = j] &= P\{\sigma \,|\, \delta(i, \sigma) = j \text{ and } \sigma \in v\} \\ &= \sum_{\sigma\in v\,:\,\delta(i,\sigma)=j} p(\sigma|i, v) \\ &:= P_{ij}(v). \end{aligned} \tag{4}$$

We define the *transition probability matrix* $\mathbf{P}(\mathbf{g})$ where $\mathbf{P}_{ij}(\mathbf{g}) = P_{ij}(g(i))$, which denotes the probability of going from state i to j based on policy g.

A PDES contains a mixture of terminating and non-terminating processes. If there are no events defined at i or all events at i have been disabled by the control action v, then i has a zero continuation probability $\sum_{j=0}^{I-1} P_{ij}(v) = 0$. This leads to issues in the MDP framework. To resolve this, the transition structure is modified, without loss of generality, to include the dump state, I. The new augmented state space is denoted as $X_{aug} := \{0, 1, ..., I-1, I\}$. The transition structure is modified as follows. For all $i, j \in X$

$$P_{ij}(v) = \sum_{\sigma\in\Sigma\,:\,\delta(i,\sigma)=j} p(\sigma|i, v), \tag{5a}$$

$$P_{iI}(v) = 1 \quad \text{if } [(\forall\sigma \in \Sigma) \quad \delta(i, \sigma)! \implies \sigma \notin v], \tag{5b}$$

$$P_{II}(v) = 1. \tag{5c}$$

4.2 The Cost Structure and the LU Supervisor Problem

The following per-stage cost structure $c : X_{aug} \to \{0, 1\}$ is imposed on the MDP process.

$$c(i) := \begin{cases} 1 & \text{if } i \in X_{us} \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Define the expected cost functional $J_{\mathbf{g}} : X_{aug} \to \mathbb{R}$ as

$$J_{\mathbf{g}}(i) := \lim_{N\to\infty} E^{\mathbf{g}}_{x_0=i} \sum_{t=0}^{N-1} c(x_t), \tag{7}$$

where $E^{\mathbf{g}}_{x_0=i}$ is the expectation conditioned on i being the initial state and the policy $\mathbf{g}$ being used.

Theorem 2. Given a state-based supervisor, or in other words a stationary feedback policy, $\mathbf{g}$, the cost functional satisfies $J_{\mathbf{g}}(0) = P^G_{us}(V_{\mathbf{g}})$, where $V_{\mathbf{g}}$ is obtained through (2).

Proof. First, the probability distribution of x_t given $x_0 = i$ and policy $\mathbf{g}$ we be expressed in the DES framework as

$$P(x_t = j|x_0 = i, \mathbf{g}) = \sum_{\substack{s\in L(V_{\mathbf{g}}/G,i)\,:\\ \delta(i,s)=j\wedge|s|=t}} L_p(V_{\mathbf{g}}/G, i)(s). \tag{8}$$

The proof can be shown through direct derivation.

$$\begin{aligned} J_{\mathbf{g}}(0) &= \lim_{N\to\infty} E^{\mathbf{g}}_{x_0=0} \sum_{t=0}^{N-1} c(x_t) \\ &= \lim_{N\to\infty} \sum_{t=0}^{N-1} \sum_{j=0}^{I} P(x_t = j|x_0 = 0, \mathbf{g})\, c(j). \end{aligned}$$

Substituting in (8), we get

$$= \lim_{N\to\infty} \sum_{t=0}^{N-1} \sum_{j=0}^{I} \sum_{\substack{s\in L(V_{\mathbf{g}}/G)\,:\\ \delta(0,s)=j\wedge|s|=t}} L_p(V_{\mathbf{g}}/G)(s)\, c(j).$$

The limit and the first sum reduce to sum over strings of all lengths. Substituting in the cost (6), the second sum is only over the unsafe states. Thus we arrive at

$$\begin{aligned} &= \sum_{s\in L_{us}(V_{\mathbf{g}}/G)} L_p(V_{\mathbf{g}}/G)(s) \\ &= P^G_{us}(V_{\mathbf{g}}). \end{aligned}$$

■

Using this result, we take the infimum over all stationary policies on both sides to conclude the following.

Corollary 3. Let $\mathcal{G}$ be the space of all stationary policies, we have that

$$\inf_{\mathbf{g}\in\mathcal{G}} J_{\mathbf{g}}(0) = \inf_{\mathbf{g}\in\mathcal{G}} P^G_{us}(V_{\mathbf{g}}). \tag{9}$$

Since the space of all stationary policies is finite, we can replace infimum with minimum

$$\min_{\mathbf{g}\in\mathcal{G}} J_{\mathbf{g}}(0) = \min_{\mathbf{g}\in\mathcal{G}} P^G_{us}(V_{\mathbf{g}}). \tag{10}$$

The LU supervisor problem is now an undiscounted, positive cost, infinite-horizon MDP with a finite state space X_{aug}, transition probability matrix $\mathbf{P}(\mathbf{g})$, and a finite action space Γ. Closely following notations in Bertsekas (1983), let

$$T(J)(i) = \min_{v\in\Gamma} E_\sigma \left\{c(i) + J\left[f(i,v,\sigma)\right]\right\} \tag{11}$$

and for a given policy $\mathbf{g}$, define

$$T_{\mathbf{g}}(J)(i) = E_\sigma \left\{c(i) + J\left[f(i,g(i),\sigma)\right]\right\}. \tag{12}$$

Standard results from dynamic programming state that the optimal cost functional, J^*, and the optimal policy, $\mathbf{g}^*$, satisfy the optimality equation $J^* = T(J^*) = T_{\mathbf{g}^*}(J^*)$. The cost functional induced by $\mathbf{g}$, $J_{\mathbf{g}}$, satisfies $J_{\mathbf{g}} = T_{\mathbf{g}}(J_{\mathbf{g}})$. The optimal policy is well known to be a stationary policy (Strauch (1966)). Therefore, there is no need to consider a non-stationary $\mathbf{g}$ in Theorem 2. For the general undiscounted, infinite-horizon MDP, the optimal solution may only be approximated using *successive approximation* due to

$$J = T(J) \tag{13}$$

and

$$J_{\mathbf{g}} = T_{\mathbf{g}}(J_{\mathbf{g}}) \tag{14}$$

having multiple fixed-points. We now exploit this specific Markov structure to obtain the solution to (14).

Denote $\mathcal{T}_{\mathbf{g}} \subset X_{aug}$ as the set of transient states and $\mathcal{R}_{\mathbf{g}} \subseteq X_{aug}$ as the set of recurrent states of G under the policy $\mathbf{g}$.

Lemma 4. $\mathcal{R}_{\mathbf{g}}$ does not contain any unsafe states. Thus, $c(i) = 0$ for all $i \in \mathcal{R}_{\mathbf{g}}$.

Proof. (Sketch) By assumption, unsafe states are terminating states. This implies that all unsafe states are transient states to the dump state. Thus, $\mathcal{R}_{\mathbf{g}}$ does not contain any unsafe states. ■

Proposition 5. Given a feedback policy, $\mathbf{g}$, we have that

$$J_{\mathbf{g}}(i) = 0 \quad \text{for all } i \in \mathcal{R}_{\mathbf{g}}. \tag{15}$$

Proof. Assume $i \in \mathcal{R}_{\mathbf{g}}$, $\{x_t\}$ is the resulting process given $x_0 = i$, and the stationary policy $\mathbf{g}$ is used. Using $x_0 = i \in \mathcal{R}_{\mathbf{g}}$ and $x_t \in \mathcal{R}_{\mathbf{g}} \implies x_{t+1} \in \mathcal{R}_{\mathbf{g}}$, we can conclude, by induction, that $(\forall t \geqslant 0)\, x_t \in \mathcal{R}_{\mathbf{g}}$. By Lemma 4, $(\forall t \geqslant 0)\, c(x_t) = 0$ and substituting into (7), we get that

$$J_{\mathbf{g}}(i) = \lim_{N\to\infty} E^{\mathbf{g}}_{x_0=i} \sum_{t=0}^{N-1} c(x_t) = \lim_{N\to\infty} E^{\mathbf{g}}_{x_0=i} \sum_{t=0}^{N-1} 0 = 0.$$

■

Let us examine solutions to (14) in view of Proposition 5.

Definition 6. Given a policy $\mathbf{g}$, we define a solution to (14) to be *admissible* if for all $i \in \mathcal{R}_{\mathbf{g}}$,

$$J_{\mathbf{g}}(i) = 0.$$

Expanding (14) and rearranging the state indices 0 to $\tau-1$ to be all the transient states and indices τ to I to be all the recurrent states, we get

$$\begin{bmatrix} J_{\mathbf{g}}(0) \\ \vdots \\ J_{\mathbf{g}}(\tau-1) \\ \hline J_{\mathbf{g}}(\tau) \\ \vdots \\ J_{\mathbf{g}}(I) \end{bmatrix} = \left[\begin{array}{c|c} \mathbf{P}_{\mathcal{T}_{\mathbf{g}}} & \mathbf{P}_{\mathcal{T}\cdot\mathcal{R}_{\mathbf{g}}} \\ \hline 0 & \mathbf{P}_{\mathcal{R}_{\mathbf{g}}} \end{array}\right] \cdot \begin{bmatrix} J_{\mathbf{g}}(0) \\ \vdots \\ J_{\mathbf{g}}(\tau-1) \\ \hline J_{\mathbf{g}}(\tau) \\ \vdots \\ J_{\mathbf{g}}(I) \end{bmatrix} + \begin{bmatrix} c(0) \\ \vdots \\ c(\tau-1) \\ \hline 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{16}$$

Theorem 7. Given a stationary feedback policy $\mathbf{g}$ and the per-stage cost (6), (14) has exactly one admissible solution.

Proof. Applying Proposition 5, we have $J_{\mathbf{g}}(\tau) = ... = J_{\mathbf{g}}(I) = 0$ in (16). The direct result of Theorem 3.1.1 from Kemeny and Snell (1976) can be applied to conclude that the inverse $(\mathbf{I} - \mathbf{P}_{\mathcal{T}_{\mathbf{g}}})$ exists. Thus, we may evaluate $(\mathbf{I}-\mathbf{P}_{\mathcal{T}_{\mathbf{g}}})^{-1}\cdot\mathbf{c}_{\mathcal{T}_{\mathbf{g}}}$, where $\mathbf{c}_{\mathcal{T}_{\mathbf{g}}} = [c(0)...,c(\tau-1)]^T$, to obtain $J_{\mathbf{g}}(0),...,J_{\mathbf{g}}(\tau-1)$, analytically. ■

4.3 The LU Supervisor Algorithm

Combining successive approximation and results from Theorem 7, Algorithm 1 is a proposed method to compute the LU supervisor policy. The LU supervisor policy can be implemented using a language-based supervisor $V^* : L(G) \to \Gamma$ through (2). By restricting the co-domain of g^* to Γ in (18), V^* does not violate the controllability condition in the DES framework by construction. The following is an example for Algorithm 1.

Consider the system G shown in Figure 2a, where $\Sigma_c = \{c, d\}$ and $X_{us} = \{3, 6\}$. There are uncontrollable paths to the unsafe states $b(ab)^*b \cup b(ab)^*f$. For notational convenience, denote the control actions as $cd = \Sigma$, $\bar{c}d = \Sigma - \{c\}$, $\bar{c}\bar{d} = \Sigma - \{c, d\}$, and $c\bar{d} = \Sigma - \{d\}$.

In Step 1, we augment the system to include the dump state, state 7, so we have $X_{aug} = \{0, 1, 2, 3, 4, 5, 6, 7\}$. States 3, 5, and 6 transition to state 7 with probability 1. No other state can be a terminating state through a control action, thus no other states transition to 7. The cost per stage is

$$c(0) = c(1) = c(2) = c(4) = c(5) = c(7) = 0,$$
$$c(3) = c(6) = 1.$$

In Step 2, we initialize $J_0(0) = J_0(1) = J_0(2) = J_0(3) = J_0(4) = J_0(5) = J_0(6) = J_0(7) = 0$. In iteration 1, we have

$$J_1(0) = J_1(1) = J_1(2) = J_1(4) = J_1(5) = J_1(7) = 0,$$
$$J_1(i = 3) = J_1(i = 6) = c(i) = 1.$$

In iteration 2, we have

$$J_2(0) = J_2(5) = J_2(7) = 0,$$
$$J_2(1) = 0 \quad \text{with } g(1) = c\bar{d} \text{ as the minimizing action},$$
$$J_2(2) = 0.2 \text{ with } g(2) = cd \text{ as the minimizing action},$$
$$J_2(3) = J_2(6) = 1,$$
$$J_2(4) = c(4) + (0.1)J_1(3) = 0.1.$$

In Step 3, we obtain the approximate optimal policy $\tilde{g^*}$ which is defined

Algorithm 1 LU supervisor policy: full observation

Input:

- $G = (X, \Sigma, \delta, x_0, X_m, p)$ the PDES model of the plant
- Σ_c the controllable event set
- $X_{us} \subseteq X$ the set of unsafe states

Output:

- The LU supervisor policy $\mathbf{g}^* = \{g^*, g^*, ...\}$, where $g^* : X_{aug} \to \Gamma$

Algorithm:

STEP 1: Let $X_{aug} := X \cup \{I\}$ and initialize $J_0(i) = 0$ for all $i \in X_{aug}$.

STEP 2: Perform value iteration up to an arbitrary number of iterations with

$$J_{k+1}(i) = \min_{v \in \Gamma} \left[c(i) + \sum_{j=0}^{I} P_{ij}(v) J_k(j) \right] \quad \text{for } i \in X_{aug} \tag{17}$$

where $P_{ij}(v)$ is obtained from (5). Exit if $J_{k+1} = J_k$.

STEP 3: Obtain the approximated optimal control using

$$\widetilde{g^*}(i) = \arg \min_{g(i) \in \Gamma} \left[c(i) + \sum_{j=0}^{I} P_{ij}[g(i)] J_k(j) \right] \quad \text{for } i \in X_{aug}. \tag{18}$$

STEP 4: Solve $J_{\widetilde{g^*}} = T_{\widetilde{g^*}}(J_{\widetilde{g^*}})$ for $J_{\widetilde{g^*}}$:

(a) Classify all states in X_{aug}. Collect all transient states into $\mathcal{T}_{\widetilde{g^*}}$ and all recurrent states into $\mathcal{R}_{\widetilde{g^*}}$. (See Hachtel et al. (1996) or Xie and Beerel (1998) for state classification algorithms.)

(b) Set $J_{\widetilde{g^*}}(i) = 0$ for all $i \in \mathcal{R}_{\widetilde{g^*}}$.

(c) Extract the transient-to-transient transition matrix $\mathbf{P}_{\mathcal{T}_{\widetilde{g^*}}}$ from $\mathbf{P}_{\widetilde{g^*}}$.

(d) Compute $\mathbf{J}_{\mathcal{T}_{\widetilde{g^*}}} = (I - \mathbf{P}_{\mathcal{T}_{\widetilde{g^*}}})^{-1} \cdot \mathbf{c}_{\mathcal{T}_{\widetilde{g^*}}}$, which corresponds to $J_{\widetilde{g^*}}(i)$ for all $i \in \mathcal{T}_{\widetilde{g^*}}$.

STEP 5: Check if $J_{\widetilde{g^*}}$ satisfies the optimality equation, i.e. $J_{\widetilde{g^*}} = T(J_{\widetilde{g^*}})$. If not, go back to STEP 2. Otherwise, set of optimal policy $g^* = \widetilde{g^*}$ and terminate.

$\widetilde{g^*}(0) = \widetilde{g^*}(3) = \widetilde{g^*}(4) = \widetilde{g^*}(5) = \widetilde{g^*}(6) = \widetilde{g^*}(7) = cd$ (arbitrary),

$\widetilde{g^*}(1) = \bar{c}\bar{d}$, (or $c\bar{d}$ which gives the same minimization)

$\widetilde{g^*}(2) = cd$.

Figure 2b shows the resulting closed-loop system using policy $\widetilde{g^*}$.

For Step 4a, the transient states and the recurrent states are as follows:

$$\mathcal{R}_{\widetilde{g^*}} = \{1, 7\}, \quad \mathcal{T}_{\widetilde{g^*}} = \{0, 2, 3, 4, 5, 6\}.$$

For Step 4b, we set $J_{\widetilde{g^*}}(1) = J_{\widetilde{g^*}}(7) = 0$. For Step 4c, the transient transition matrix (states 0, 2, 3, 4, 5, and 6) can be obtained from Figure 2b and is

$$\mathbf{P}_{\mathcal{T}_{\widetilde{g^*}}} = \begin{bmatrix} 0 & 0.3 & 0 & 0 & 0 & 0 \\ 0.1 & 0 & 0.1 & 0.5 & 0.1 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0.9 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

In Step 4d, we solve for the cost functional $J_{\widetilde{g^*}}$ using $\mathbf{P}_{\mathcal{T}_{\widetilde{g^*}}}$ and $\mathbf{c}_{\mathcal{T}_{\widetilde{g^*}}} = (0, 0, 1, 0, 0, 1)^T$. We arrive at

$$J_{\widetilde{g^*}}(0) = 0.077, \quad J_{\widetilde{g^*}}(1) = J_{\widetilde{g^*}}(5) = J_{\widetilde{g^*}}(7) = 0,$$
$$J_{\widetilde{g^*}}(2) = 0.258, \quad J_{\widetilde{g^*}}(3) = J_{\widetilde{g^*}}(6) = 1, \quad J_{\widetilde{g^*}}(4) = 0.1.$$

In Step 5, it is easy to check that $J_{\widetilde{g^*}}$, indeed, satisfies the optimality equation $J_{\widetilde{g^*}} = T(J_{\widetilde{g^*}})$. Hence, we set $g^* = \widetilde{g^*}$ and terminate the algorithm with $J_{g^*}(0) = P^G_{us}(g^*) = 0.077$.

Let us study the system using intuition. First, observe that we can "trap" the process in the self-loop at state 1 by disabling the events c and d, where it will self-loop forever and never reach unsafe states 3 or 6. At state 2, we enable event d to divert the probabilities from events b and f into the trap state 1. At state 4, we have a relatively high probability of the system executing event e and arriving at state 5 where it has no access to the unsafe states. Thus, event c is enabled at state 2. With the LU supervisor implemented, we have $P^G_{us}(g^*) = 0.077$. The open-loop would have P^G_{us}(enable-all) $= 0.596$. The disable-all strategy results in P^G_{us}(disable-all) $= 0.162$.

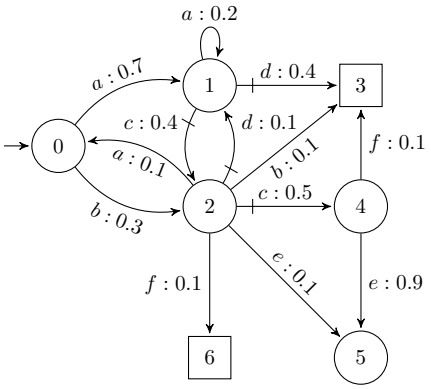
4.4 Discussion on State-based Policy vs. Language-based Supervisory Control

The question addressed in this section is whether searching in the space of state-based policies, $g : X \to \Gamma$, and not considering language-based supervisors, $V : L(G) \to \Gamma$, is too limiting? And, can a certain V achieve a lower minimum? An advantage that V may have over g is that it uses a string that describes the precise historical behaviour of the system, as opposed to knowing just the current state. However, results of dynamic programming show that state feedback is sufficient for optimality and any form of historical information does not provide further improvement. Searching in the space of all language-based feedback policies would be redundant. The optimal language-based supervisor can be found within the subset of supervisors that behave like a state-based controller. In other words, all supervisors $V_{\mathbf{g}}$ such that for all $s, t \in L(G)$,

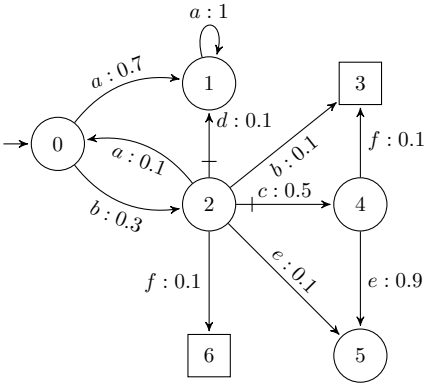
$$\delta(x_0, s) = \delta(x_0, t) \implies V_{\mathbf{g}}(s) = V_{\mathbf{g}}(t).$$

5. THE LEAST-UNSAFE ATTACK MITIGATION SYSTEM

This section discusses the implementation of the LU supervisor as an enhancement of the "disable-all" mitigation strategy in the attack detection-and-mitigation framework introduced in Carvalho et al. (2018). Figure 3 illustrates an overview of the least-unsafe attack mitigation architecture. G_a and S_a are the attack models of G and S, respectively (see Carvalho et al. (2018)). G_D is the attack detection module. An interrupt signal **INT** is sent to the switch when G_D enters an attack-certain state.



(a) G



(b) g^*/G

Fig. 2. An example for the LU supervisor policy algorithm (a) is the plant, (b) closed-loop system using the LU supervisor.

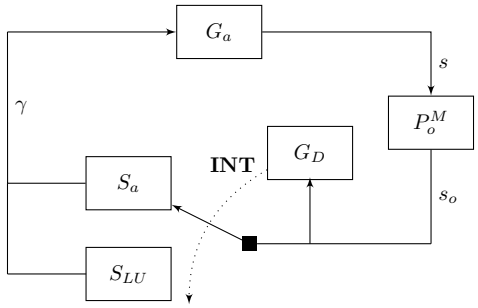


Fig. 3. Least unsafe attack mitigation architecture

The assumption of full observability limits applicability of the presented approach in studying attack mitigation systems. Nevertheless, the LU supervisor is only utilized after an attack detection. Thus, we only impose the assumption of full observability on the portion of the system after an attack detection. The LU supervisor policy generated in Algorithm 1 may be implemented under the following two conditions.

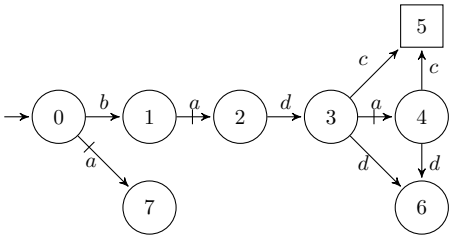
(1) The current state is known with probability 1 when an attack has been detected. Under such a situation, G_D should only have one element in its attack-certain state at the moment of detection.

(2) There are no occurrences of unobservable events in all subsequent behaviour from the attack detection state.

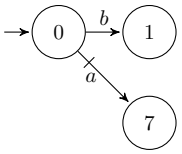
We now illustrate the LU supervisor under full observation with a simple example.

Consider the system G given in Figure 4 where $\Sigma_c = \{a\}$, $\Sigma = \{a, b, c, d\}$, and $X_{us} = \{5\}$. To avoid the unsafe state, the nominal supervisor S_{nom} is designed to be maximally permissive while preventing any uncontrollable behaviour to the unsafe state. The system is vulnerable to sensor erasure attacks on event b, hence $\Sigma_v = \{b\}$ and $\Sigma_v^a = \{b^a\}$ are the vulnerable event and corresponding attack event, respectively. G_M is the resultant attack model generated using the SE-attack algorithm from Carvalho et al. (2018). Observing Figure 4c, the attack detection module confirms that an attack has happened after observing the string ad, at which the "disable-all" mitigation strategy is deployed. Notice that at the instance of attack detection, the system's current state is 3 w.p. 1 and all downstream behaviours do not contain any unobservable events. Thus, Algorithm 1 may be applied.

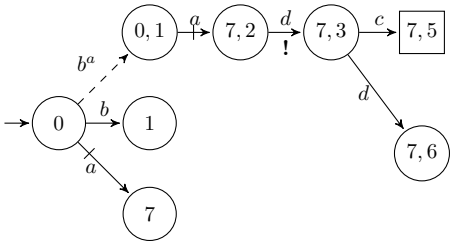
Figure 5a shows the subsystem $G_{a,3} = (X, \Sigma, \delta, 3, X_m)$ from G_a with state 3 as the initial state and its PDES description. The intuition is to divert the system into entering state 4 where it has a higher probability of arriving at the safe state 6. Enabling event a at state 3 results in $P_{us}^{G_{a,3}}(S_{LU}) = 0.34$, which is the LU supervisory action of $S_{LU,3}$. Figure 5b shows the system under the "disable-all"



(a) G

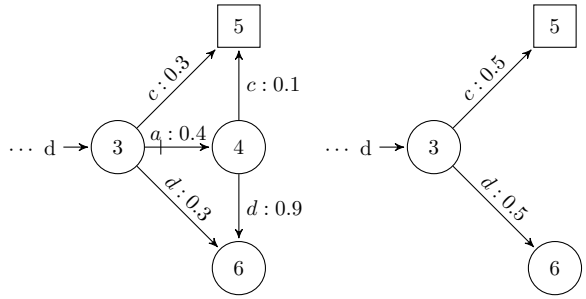


(b) S_{nom}



(c) G_M+ "disable-all"

Fig. 4. Example of an attack model and the "disable-all" mitigation strategy



(a) $G_{a,3}$ (b) $S_{disable-all}/G_{a,3}$

Fig. 5. G after attack detected and the disable-all mitigation strategy

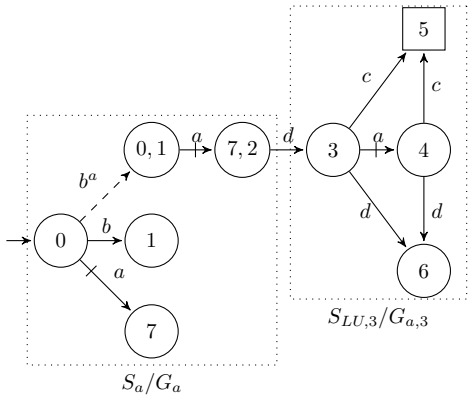


Fig. 6. Example of least-unsafe attack mitigation system under full observation assumption

mitigation strategy, which results in $P_{us}^{G_{a,3}}(S_{disable-all}) = 0.5$. Stitching the two closed-loop subsystems together, the final behaviour of the least-unsafe attack mitigation strategy is shown in Figure 6.

6. CONCLUSION

This paper introduces a probabilistic attack model that builds on Carvalho et al. (2018) by using a probabilistic DES framework, which leads to the formulation of the least-unsafe (LU) supervisor problem. For fully observable systems, the LU supervisor problem is reformulated into a positive cost, undiscounted, infinite-horizon MDP problem. Algorithm 1 is developed to compute the LU supervisor. The implementation of the least-unsafe attack mitigation system with the integration of the LU supervisor is illustrated with an example.

While not studied in this paper, under partial observation, the LU supervisor problem can be reformulated into an undiscounted, infinite-horizon, partially observable MDP (POMDP), Unfortunately, effective computational methods to solve the resulting POMDP analytically are not available. Some special cases are examined in Wang (2019).

Future work consists of studying the convergence rate of Algorithm 1 and a further in-depth study of the LU supervisor problem under partial observation. Further investigation may also be carried out on using linear programming in Algorithm 1 rather than successive approximation based on the work of de Alfaro (1999). A comparison study between the LU-supervisor and other attack modelling frameworks and mitigation strategies would also be of interest.

REFERENCES

Ashibani, Y. and Mahmoud, Q. (2017). Cyber physical systems security: Analysis, challenges and solutions. *Computers & Security*, 68, 81–97.

Banerjee, A., Venkatasubramanian, K., Mukherjee, T., and Gupta, S. (2012). Ensuring safety, security, and sustainability of mission-critical cyber–physical systems. *Proceedings of the IEEE*, 100, 283–299.

Bertsekas, D. (1983). *Dynamic programming and stochastic control*. Academic Press.

Carvalho, L., Wu, Y., Kwong, R., and Lafortune, S. (2018). Detection and mitigation of classes of attacks in supervisory control systems. *Automatica*, 97, 121–133.

Chattopadhyay, I. and Ray, A. (2009). Optimal control of infinite horizon partially observable decision processes modelled as generators of probabilistic regular languages. *International Journal of Control*, 83, 457–483.

de Alfaro, L. (1999). Computing minimum and maximum reachability times in probabilistic systems. *CONCUR'99 Concurrency Theory*, 66–81.

Garg, V., Kumar, R., and Marcus, S. (1999). A probabilistic language formalism for stochastic discrete-event systems. *IEEE Transactions on Automatic Control*, 44, 280–293.

Hachtel, G., Macii, E., Pardo, A., and Somenzi, F. (1996). Markovian analysis of large finite state machines. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15, 1479–1493.

Kemeny, J. and Snell, J. (1976). *Finite Markov chains*. Springer-Verlag.

Lawford, M. and Wonham, W. (1993). Supervisory control of probabilistic discrete event systems. *Proceedings of 36th Midwest Symposium on Circuits and Systems*.

Lima, P.M., Alves, M.V.S., Carvalho, L.K., and Moreira, M.V. (2018). Security against communication network attacks of cyber-physical systems. *Journal of Control, Automation and Electrical Systems*, 30, 125–135.

Meira-Góes, R., Kwong, R., and Lafortune, S. (2019). Synthesis of sensor deception attacks for systems modeled as probabilistic automata. *2019 American Control Conference (ACC)*.

Pantelic, V., Lawford, M., and Postma, S. (2014). A framework for supervisory control of probabilistic discrete event systems. *IFAC Proceedings Volumes*, 47, 477–484.

Rashidinejad, A., Wetzels, B., Reniers, M., Lin, L., Zhu, Y., and Su, R. (2019). Supervisory control of discrete-event systems under attacks: An overview and outlook. *2019 18th European Control Conference (ECC)*.

Strauch, R. (1966). Negative dynamic programming. *The Annals of Mathematical Statistics*, 37, 871–890.

Wang, Z. (2019). *Mitigation of Classes of Attacks using a Probabilistic Discrete Event System Framework*. Master's thesis, University of Toronto.

Wonham, W. and Cai, K. (2019). *Supervisory control of discrete-event systems*. Springer.

Xie, A. and Beerel, P. (1998). Efficient state classification of finite-state Markov chains. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17, 1334–1339.