Supervisory Enforcement of Current-State Opacity with Incomparable Observations

Yin Tong[1], Ziyue Ma[1], Zhiwu Li[2], Carla Seatzu[3] and Alessandro Giua[4]

Abstract

Current-state opacity is a key security property in discrete event systems. A system is said to be current-state opaque if the intruder, who only has partial observation on the system's evolution, is never able to establish if the current state of the system is within a set of secret states. In this work, we address the problem of enforcing current-state opacity by supervisory control. Given a system that is modeled with a finite automaton and that is not current-state opaque with respect to a given secret, the enforcement problem consists in designing a supervisor so that the controlled system is current-state opaque. We assume that the supervisor can only observe and control a subset of events. To be more general, we assume there is no specific containment relationship between the sets of events that can be observed by the intruder and the supervisor, respectively. We call this general setting *incomparable observations*. We show that the maximally permissive supervisor always exists and propose a novel approach for its design.

To appear as:

Y. Tong, Z.Y. Ma, Z.W. Li, C. Seatzu, A. Giua, "Supervisory Enforcement of Current-State Opacity with Incomparable Observations," WODES'16: 13th International Workshop on Discrete Event Systems (Xi'an, China), May 30 - June 1, 2016.

---

*This work was supported by the National Natural Science Foundation of China under Grant Nos. 61374068, 61472295, the Recruitment Program of Global Experts, the Science and Technology Development Fund, MSAR, under Grant No. 066/2013/A2.

[1]Yin Tong and Ziyue Ma are with the School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China and also with the Department of Electrical and Electronic Engineering, University of Cagliari, 09123 Cagliari, Italy yintong@stu.xidian.edu.cn; maziyue@gmail.com

[2]Zhiwu Li is with the Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau and also with the School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China zhwli@xidian.edu.cn

[3]Carla Seatzu is with the Department of Electrical and Electronic Engineering, University of Cagliari, 09123 Cagliari, Italy seatzu@diee.unica.it

[4]Alessandro Giua is with Aix Marseille Université, CNRS, ENSAM, Université de Toulon, LSIS UMR 7296, Marseille 13397, France and also with DIEE, University of Cagliari, Cagliari 09124, Italy alessandro.giua@lsis.org; giua@diee.unica.it

I. INTRODUCTION

Motivated by the concern about security and privacy in computer systems, communication protocols etc., various notions of secrecy have been formulated, such as *non-interference* [1], [2], *anonymity* [3], [4] and *opacity* [5], [6], [7], [8], [9]. In this paper we focus on opacity, and especially on current-state opacity in discrete event systems. Given a system, a subset of its states is considered as "secret". There exists a malicious observer (called intruder) who attempts to detect the secret so that an attack can be launched. It is usually assumed that the intruder knows the structure and the initial state of the system but has only partial observation of the system's evolution. The system is said to be *current-state opaque* with respect to a given secret if the intruder cannot determine with certainty if the current state of the system belongs to the secret. There are several ways to verify current-state opacity [9], [10].

The aim of this work is not to check current-state opacity for a system but to enforce this property using supervisory control. More precisely, given a system that is not current-state opaque with respect to a given secret, our purpose is to design a supervisor that minimally restricts the behavior of the system while guaranteeing that the controlled system is current-state opaque. Formally, a supervisor is a map specifying for each possible string of generated events, the set of events (which should include all uncontrollable events) that are allowed to occur at that point such that the current state remains opaque [11]. The supervisor should be maximally permissive (optimal), i.e., the system's behavior should not be unnecessarily restricted.

There has been some related work on the design of supervisors to enforce opacity properties. In [12], the authors consider the secret defined as a set of event sequences (such an opacity property is usually called language-based opacity) and more than one intruder having different observations. They assume that all events are observable and controllable to the supervisor, and show that the optimal supervisor always exists. Considering the same language-based opacity enforcement problem but with only one intruder, Dubreil et al. [13], [14] study a more general case where the supervisor may observe a set of events different from the one observed by the intruder in the presence of uncontrollable events. The authors of [15] propose methods for designing optimal supervisors to enforce two different opacity properties: initial-state opacity and infinite-step opacity, with the assumption that the supervisor can observe all events.

More recently, the common assumption that all controllable events are also observable [12], [13], [14], [15] is relaxed in [16] to enforce current-state opacity. In such a case, the maximally permissive supervisor may not be unique. We point out that all the aforementioned works are carried out in the framework of finite automata and all of them assume that the set of events observed by the intruder is a subset of the events observed by the supervisor (or vice versa). We also point out that in [14] it is assumed that the intruder knows not only the structure of the system but also the sets E_c and E_S under the assumption that $E_c \subseteq E_S \subseteq E_I$, where E_c is the set of controllable events, E_S is the set of events observable by the supervisor and E_I is the set of events observable by the intruder. Therefore, the intruder predicts the control policy by applying supervisory control theory and updates his estimation accordingly. However, this assumption results in a paradigm of a full information game between two smart-enough players, under which the solution to the opacity enforcement problem would be difficult to find since the supervisor

needs to be iteratively computed. Moreover, this assumption could hardly be satisfied in practical cases, since the intruder may not be able to acquire all the information. As a result, it is reasonable for us to assume that the intruder does not know what can be observed and controlled by the supervisor.

In this paper we tackle the current-state opacity enforcement problem in the framework of finite automata and in particular, we assume that no specific containment relationship exists between the sets of events that can be observed by the intruder and the supervisor, respectively. We call this general setting *incomparable observations*. In this sense, the problem considered here is more general than the one in [12], [13], [14], [15], [16]. Furthermore, we assume that all controllable events are also observable by the supervisor, which is a common and practical assumption when dealing with supervisory control. A structure called *parallel observer* (PO) is constructed to capture the observations of the intruder and the supervisor simultaneously. The PO of a system is a deterministic finite automaton, where each state corresponds to the current-state estimate of the intruder and the supervisor. Finally, based on the PO, the optimal supervisor enforcing current-state opacity can be designed. The contributions of this work can be summarized as follows.

- A novel finite structure, the PO, that enables one to relax the assumption $E_S \subseteq E_I$ (or $E_I \subseteq E_S$) is proposed.
- An approach to design the maximally permissive supervisor that enforces current-state opacity is developed based on the PO.

The rest of this paper is organized as follows. Basic notions on automata and supervisory control are recalled in Section II. Section III presents the definition of current-state opacity and the corresponding verification approach. In Section IV the current-state opacity enforcement problem is formalized and a method for the synthesis of the optimal supervisor is proposed. Finally, this paper is concluded in Section V, where our future work in this topic is also discussed.

II. Preliminaries and Background

In this section we recall the basics on automata and supervisory control. For more details, we refer the reader to [11], [17].

A. Automata

A *deterministic finite automaton* (DFA) is a 4-tuple $G = (X, E, \delta, x_0)$, where X is the finite *set of states*, E is the set of *events*, $\delta : X \times E \to X$ is the (partial) *transition function*, and $x_0 \in X$ is the *initial state*. The transition function can be extended to $\delta : X \times E^* \to X$ recursively: $\delta(x, \varepsilon) = x$ and $\delta(x, \sigma e) = \delta(\delta(x, \sigma), e)$ for $\sigma \in E^*$ and $e \in E$. We denote by $\delta(x, \sigma)!$ the fact that σ is defined at x. The *generated language* of a DFA $G = (X, E, \delta, x_0)$ is defined as

$$L(G) = \{\sigma \in E^* | \delta(x_0, \sigma)!\}.$$

To model the partial observation on event sequences of the intruder and the supervisor, we denote $E_I \subseteq E$ and $E_S \subseteq E$ the set of events observable by the intruder and the supervisor, respectively. The set of events observable

either by the intruder or the supervisor is $E_o = E_I \cup E_S$. The natural *projection* $P_I : E^* \to E_I^*$ on E_I is defined as i) $P_I(\varepsilon) = \varepsilon$; ii) for all $\sigma \in E^*$ and $e \in E$, $P_I(\sigma e) = P_I(\sigma)e$ if $e \in E_I$, and $P_I(\sigma e) = P_I(\sigma)$, otherwise. Given a set of sequences $L \subseteq E^*$, its projection is defined as $P_I(L) = \bigcup_{\sigma \in L}\{P_I(\sigma)\}$. Similarly, the natural projection $P_S : E^* \to E_S^*$ on E_S can be defined. Given an event sequence $\sigma \in E^*$, its projection $w_i = P_I(\sigma)$ (resp., $w_s = P_S(\sigma)$) on E_I (resp., E_S) is called an *observation* of the intruder (resp., supervisor).

Given a state x and the empty word ε, the *unobservable reach* $R_I(x, \varepsilon)$ of x with respect to the intruder is

$$R_I(x, \varepsilon) = \{x' \in X | \exists \sigma \in (E \setminus E_I)^* : \delta(x, \sigma) = x'\}.$$

Clearly, $x \in R_I(x, \varepsilon)$. Given an event $e \in E_I$, the *e-reach* $R_I(x, e)$ of x is defined as $R_I(x, e) = R_I(x', \varepsilon)$, where $x' = \delta(x, e)$. Analogously, we can define the unobservable reach $R_S(x, \varepsilon)$ and the e-reach $R_S(x, e)$ with $e \in E_S$ for the supervisor.

In this work, it is assumed that the system is modeled by a DFA. We note that since a nondeterministic finite automaton can always be converted into an equivalent DFA, this assumption does not restrict the application of the proposed approach.

B. Supervisory Control

Given a system $G = (X, E, \delta, x_0)$ and a series of specifications, the goal of supervisory control is to design a supervisor such that the controlled system satisfies the specifications. The supervisor observes a subset E_S of the events in E and is able to control (i.e., disable) a subset of events $E_c \subseteq E$. We denote $E_{uc} = E \setminus E_c$ the set of events that cannot be controlled by the supervisor. According to [11], a supervisor is a map $f : E_S^* \to 2^{E_c}$. In other words, given an observation w_s of the supervisor, the set of events enabled by the supervisor is $f(w_s) \cup E_{uc}$. Namely, based on its observation, the supervisor can disable one or more controllable events so that the system will not violate the specification. In this work, the supervisor is described by a DFA $Sup = (Y, E, \delta_s, y_0)$. Given an observation w_s of the supervisor, let $\delta_s(y_0, w_s) = y$, then the set of events enabled by the supervisor is $E_{uc} \cup \{e \in E | \delta_s(y, e)!\}$.

III. Current-State Opacity and Verification

Current-state opacity has been defined in both automata and Petri nets framework [6], [9], [10], [18]. In this section, we recall the definition of current-state opacity in finite automata and the approach to checking this property.

Given a system, it is usually assumed that the intruder knows the system's structure G but only some events occurrence can be detected by the intruder. Current-state opacity is defined as follows.

Definition 3.1: Given a system $G = (X, E, \delta, x_0)$, a secret $S \subseteq X$, and the set E_I of events observable by the intruder, the system is said to be *current-state opaque* wrt S and E_I if $\forall \sigma \in L(G)$ such that $\delta(x_0, \sigma) \in S$,

$$\exists \sigma' \in L(G) : P_I(\sigma') = P_I(\sigma) \text{ and } \delta(x_0, \sigma') \notin S. \qquad \diamond$$

In simple words, for each sequence of events σ that leads to a state in the secret, i.e., a secret state, there exists at least an event sequence that reaches a non-secret state but produces the same observation $P_I(\sigma)$ to the intruder. Therefore, when the intruder observes $w_i = P_I(\sigma)$, it cannot conclude whether the current state is contained or not

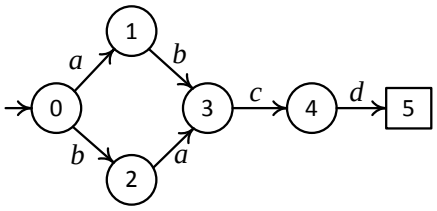


Fig. 1. System G that is not CSO wrt $S = \{5\}$ and $E_I = \{a, d\}$.

in the secret. However, based on the system's structure and its observation, the intruder can estimate the current state.

Definition 3.2: Given a system $G = (X, E, \delta, x_0)$ and an observation w_i of the intruder, the *estimate* of the intruder[1] is defined as

$$\mathcal{C}_I(w_i) = \{x \in X | \exists \sigma \in E^* : \delta(x_0, \sigma) = x, P_I(\sigma) = w_i\}. \diamond$$

Therefore, if the intruder observes a sequence of events w_i, it knows that the current state is any state in the set $\mathcal{C}_I(w_i)$. Such a set is called the set of states consistent with w_i.

Theorem 3.3: Let $G = (X, E, \delta, x_0)$ be the system, $S \subseteq X$ be the secret and E_I be the set of events observable by the intruder. The system is current-state opaque wrt S and E_I if and only if for all $\sigma \in L(G)$,

$$\mathcal{C}_I(w_i) \nsubseteq S$$

holds, where $w_i = P_I(\sigma)$. $\diamond$

Proof: Let $\sigma \in L(G)$ and $w_i = P_I(\sigma)$.

(*If*) Assume $\mathcal{C}_I(w_i) \nsubseteq S$. Let $x' \in \mathcal{C}_I(w_i) \setminus S$. According to Definition 3.2, there exists $\sigma' \in L(G)$ such that $\delta(x_0, \sigma') = x' \notin S$ and $P_I(\sigma') = w_i$. By Definition 3.1, the system is current-state opaque.

(*Only If*) Assume the system is current-state opaque. Thus, there exists at least one sequence of events $\sigma' \in L(G)$ such that $\delta(x_0, \sigma') = x' \notin S$ and $P_I(\sigma') = w_i$. According to Definition 3.2, $x' \in \mathcal{C}_I(w_i)$, i.e., $x' \in \mathcal{C}_I(w_i) \setminus S$. Therefore, $\mathcal{C}_I(w_i) \nsubseteq S$. ■

In simple words, to verify if a system is current-state opaque wrt the given secret, one needs to build the sets $\mathcal{C}_I(w_i)$ for all $\sigma \in L(G)$ and check whether $\mathcal{C}_I(w_i) \nsubseteq S$ holds. This can be done by constructing the *observer* of the system for the intruder (i.e., wrt E_I). The observer captures all state estimates of the intruder. More specifically, the state of the observer reached by w_i is equal to $\mathcal{C}_I(w_i)$. Therefore, we can use the observer to verify current-state opacity of the system. The observer is defined in Section 2.5.2 of [17], where the algorithm to construct it can also be found. Herein, it is not recalled for the sake of simplicity. Therefore, to avoid leaking the secret, we should make unreachable the states of the observer such that $\mathcal{C}_I(w_i) \subseteq S$.

[1]Analogously, given an observation w_s of the supervisor, the supervisor's estimate is defined as $\mathcal{C}_S(w_s) = \{x \in X | \exists \sigma \in E^* : \delta(x_0, \sigma) = x, P_S(\sigma) = w_s\}$.

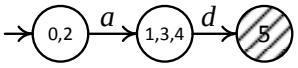


Fig. 2. Observer of the system in Fig. 1 for the intruder.

Example 3.4: Consider the system in Fig. 1. Let $E_I = \{a, d\}$ and $S = \{5\}$ (the secret state is in box). The corresponding observer for the intruder is shown in Fig. 2. Since there exists $w_i = ad$ such that $\mathcal{C}_I(w_i) = \{5\} \subseteq S$ (state in shadow), by Theorem 3.3, the system is not current-state opaque wrt S and E_I. $\diamond$

IV. Current-State Opacity Enforcement by Control

Given a system that is not current-state opaque wrt a secret, an interesting question is how to restrict its behavior or how to modify the observation structure such that the secret will never be revealed to the intruder. In this work we address the first issue using supervisory control theory [11]. The supervisor will restrict the system's behavior to prevent the evolutions that leak the secret. In this section, we present a novel approach to designing the maximally permissive supervisor enforcing current-state opacity without assuming any containment relationship between E_S and E_I.

A. Problem Formulation

Before we formalize the problem addressed in the rest of this work, the following assumptions are made.

A1) $\forall \sigma \in E_{uc}^* \cap L(G)$, it holds $\mathcal{C}_I(w_i) \nsubseteq S$, where $w_i = P_I(\sigma)$.

A2) All controllable events are also observable by the supervisor, i.e., $E_c \subseteq E_S$.

Assumption A1 claims that the secret cannot be leaked by the only occurrence of uncontrollable events. Such an assumption guarantees that there always exists a solution, because in the worst case, all controllable events are disabled at the initial state. Assumption A2 implies that there may be observable events that cannot be disabled by the supervisor and all unobservable events are uncontrollable. Note that there is no specific containment relationship between E_I and E_S, which makes the major difference of our work from [12], [13], [15], [14], [16]. The problem we want to solve in this work can be formalized as follows.

Problem Statement: Consider a system $G = (X, E, \delta, x_0)$, a secret $S \subseteq X$ and a set of events E_I observable by the intruder such that the system is not current-state opaque wrt S and E_I. Assume that the intruder does not know the supervisor. Namely, it cannot update its estimation taking into account the supervisory action. Let Assumptions A1 and A2 be satisfied, we want to synthesize an optimal (i.e., maximally permissive) supervisor such that the controlled system is current-state opaque wrt S and E_I.

Example 4.1: Consider again the system in Fig. 1. The sets of events observable/controllable by the intruder and the supervisor are shown in Table I. In this case, E_I and E_S are not comparable, i.e., neither $E_I \subseteq E_S$ nor $E_S \subseteq E_I$ holds. From Example 3.4 we know that, if the intruder observes $E_I = \{a, d\}$, the system is not current

TABLE I

OBSERVABILITY AND CONTROLLABILITY OF THE EVENTS.

Events	E_I	E_S	E_c
a	$\surd$	$\times$	$\times$
b	$\times$	$\surd$	$\times$
c	$\times$	$\surd$	$\surd$
d	$\surd$	$\times$	$\times$

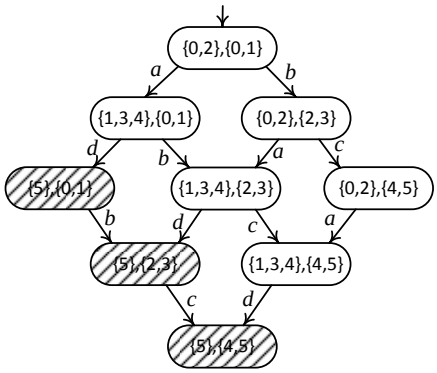


Fig. 3. Parallel composition of observers

state opaque wrt $S = \{5\}$. Assuming that only events b and c can be observed by the supervisor and c can also be controlled, we want to design an optimal supervisor, so that the system becomes opaque. $\diamond$

In the next subsection, we introduce a structure, called *parallel observer*, based on which the optimal supervisor can be designed.

B. Synthesis of the Optimal Supervisor

To design the supervisor, we have to characterize the supremal controllable and observable behavior of the system such that the secret will never be leaked. However, the absence of specific containment relationships between E_I and E_S greatly makes the problem non trivial. In the following we provide an example such that the approach in [14] fails since none of the containment relationships $E_I \subseteq E_S$ or $E_S \subseteq E_I$ holds.

Consider the problem in Example 4.1. According to [14], observers of the system for the intruder and the supervisor should be constructed first. Then we have to compute the parallel composition of them (see Fig. 3, states in shadow should be forbidden). Such a structure characterizes the behavior that should be forbidden by the supervisor. Finally, we compute the observer of the parallel composition structure for the supervisor (wrt E_S). Therefore, the complexity of the approach is $\mathcal{O}(2^{2^{2|X|}})$. Without the assumption $E_I \subseteq E_S$ or $E_S \subseteq E_I$, the parallel composition between the observers would introduce event sequences (e.g., $\sigma = ad$) not belonging to $P_o(L(G))$, where $P_o : E^* \to E_o$ and $E_o = E_I \cup E_S$ (in the case at hand, being $E_o = E$, it is $P_o(L(G)) = L(G)$). As a result,

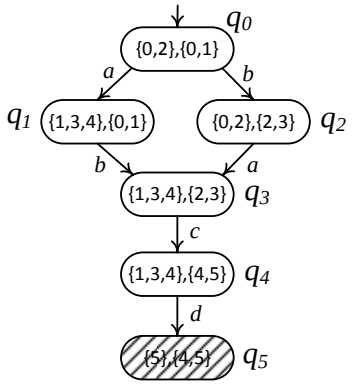


Fig. 4. Parallel Observer of the system in Example 4.2

the behavior of the system would be over restricted. For instance, sequence ab does not leak the secret. However, it should be disabled: after uncontrollable event d occurs, sequence abd will lead to a state in shadow. Therefore, the obtained supervisor would not be optimal, or even no such an opacity enforcing supervisor exists (as in the case at hand).

In this work, we introduce a structure, called a *parallel observer* (PO), based on which the optimal supervisor can be designed without increasing the complexity even if no containment relationship exists between E_I and E_S. Given an event sequence, the PO captures both the estimate of the intruder and the supervisor. The PO of the system G is a DFA denoted as $\mathcal{P} = (Q, E_o, \delta_p, q_0)$, where $E_o = E_I \cup E_S$. A state $q \in Q$ of $\mathcal{P}$ is a pair (C_I, C_S), where $C_I \subseteq X$ and $C_S \subseteq X$. The initial state of the PO is $q_0 = (R_I(x_0, \varepsilon), R_S(x_0, \varepsilon))$. Algorithm 1 illustrates the construction of the PO.

Now we explain the idea behind Algorithm 1. Given a state $q = (C_I, C_S) \in Q$ and an event $e \in E_o$, using Algorithm 1, the transition $\delta_p(q, e) = q' = (C'_I, C'_S)$ in the PO is computed as follows. If e can only be observed by the intruder, C'_I is the e-reach of states in C_I while C_S does not change, i.e., $C'_S = C_S$; if e can only be observed by the supervisor, C'_S equals the e-reach of states in C_S while C_I does not change, i.e., $C'_I = C_I$; on the contrary, if e can be observed by both the intruder and the supervisor, C'_I and C'_S are e-reaches of states in C_I and C_S, respectively. If q' is a new state, it will be added to Q, otherwise Q does not change. In Algorithm 1 all states in Q are analyzed. The maximum number of states of the PO is $2^{2|X|}$. The difference between Algorithm 1 and the parallel composition of observers for the intruder and the supervisor lies in Step 4: not all observable events are considered in the loop.

Example 4.2: Consider the problem in Example 4.1. Using Algorithm 1, the PO is constructed and shown in Fig. 4. $\diamond$

From Algorithm 1, the PO has the following properties:

i) $P_I(L(G)) = P_I(L(\mathcal{P}))$ and $P_S(L(G)) = P_S(L(\mathcal{P}))$;

ii) Given a sequence $\sigma \in L(G)$, Let σ_o be the projection of σ on E_o, and in the PO $q = (C_I, C_S) = \delta_p(q_0, \sigma_o)$.

Algorithm 1 Computation of the Parallel Observer

Input: A system $G = (X, E, \delta, x_0)$ and sets of events E_I and E_S.

Output: The corresponding parallel observer $\mathcal{P} = (Q, E_o, \delta_p, q_0)$.

$q_0 := (R_I(x_0, \varepsilon), R_S(x_0, \varepsilon))$ and assign no tag to it;
$Q := \{q_0\}$;
while $q = (C_I, C_S) \in Q$ with no tag exists, **do**
  for all $e \in E_o$ such that $\exists x \in C_I : \delta(x, e)!$ and $\exists x' \in C_S : \delta(x', e)!$, **do**
    if $e \in E_I \setminus E_S$, **then**
      $C'_I := \bigcup_{x \in C_I} R_I(x, e)$;
      $C'_S := C_S$;
    else if $e \in E_S \setminus E_I$, **then**
      $C'_I := C_I$;
      $C'_S := \bigcup_{x \in C_S} R_S(x, e)$;
    else if $e \in E_I \cap E_S$, **then**
      $C'_I := \bigcup_{x \in C_I} R_I(x, e)$;
      $C'_S := \bigcup_{x \in C_S} R_S(x, e)$;
    end if
    $q' := (C'_I, C'_S)$;
    if $q' \notin Q$ **then**
      $Q := Q \cup \{q'\}$;
    end if
    $\delta_p(q, e) := q'$;
  end for
  Tag q "old";
end while
Remove all tags;
Output $\mathcal{P}$.

It holds $C_I = \mathcal{C}_I(w_i)$ and $C_S = \mathcal{C}_S(w_s)$, where $w_i = P_I(\sigma)$ and $w_s = P_S(\sigma)$.

From the second property above and Theorem 3.3, we have Corollary 4.3, which shows that the PO can also be used to verify current-state opacity of a system.

Corollary 4.3: Given a system G, a secret S and the sets of events E_I and E_S, let $\mathcal{P} = (Q, E_o, \delta_p, q_0)$ be the parallel observer. G is not current-state opaque wrt S and E_I if and only if there exists a state $q = (C_I, C_S) \in Q$ such that $C_I \subseteq S$.

Based on Corollary 4.3 and supervisory control theory, Algorithm 2 is proposed to derive the maximally permissive supervisor. The inputs of the algorithm are the parallel observer $\mathcal{P} = (Q, E_o, \delta_p, q_0)$ of the system and the secret S. First, all states such that $C_I \subseteq S$ are computed. We call $\hat{Q} \subseteq Q$ such a set of states. Since there may be events in the PO that cannot be observed by the supervisor, the observer of the PO wrt E_S is first constructed. We denote $\mathcal{P}_s = (Y, E_S, \delta_m, m_0)$ the obtained observer. Clearly, if $E_I \subseteq E_S$, then $\mathcal{P}$ and $\mathcal{P}_s$ are identical. A state m of $\mathcal{P}_s$ is a subset of Q. Initially, the supervisor Sup is identical to $\mathcal{P}_s$. Then selfloops of events in E_o that are not defined at some states of Sup are added. At this point, we compute the set $\hat{Y}_1$ of states that contain a state in $\hat{Q}$. In other words, states in $\hat{Y}_1$ are states needed to be unreachable in $\mathcal{P}_s$, since they contain states in $\hat{Q}$ that will leak the secret. Secondly, to ensure controllability, states (i.e., states in $\hat{Y}_2$) that lead to states in $\hat{Y}_1$ by firing uncontrollable events are computed. Finally, we remove all states in $\hat{Y}_1$ and $\hat{Y}_2$ and the related transitions from Sup. The obtained DFA is the supervisor.

Algorithm 2 Computation of the Maximally Permissive Supervisor

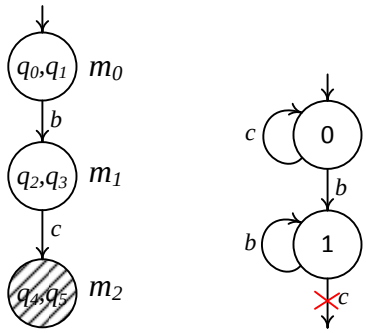
Input: The PO $\mathcal{P} = (Q, E_o, \delta_p, q_0)$, and the secret S.

Output: The supervisor $Sup = (Y, E, \delta_s, y_0)$.

1: $\hat{Q} := \{q = (C_I, C_S) \in Q | C_I \subseteq S\}$;
2: Compute the observer $\mathcal{P}_s = (Y, E_S, \delta_m, m_0)$ of $\mathcal{P}$ for the supervisor.
3: $Sup := \mathcal{P}_s$.
4: **for all** $y \in Y$, **do**
5: **for all** $e \in E_o$ that is not defined at y, **do**
6: Add $\delta_s(y, e) = y$;
7: **end for**
8: **end for**
9: $\hat{Y}_1 := \{y \in Y | \exists q \in y : q \in \hat{Q}\}$;
10: $\hat{Y}_2 := \{y \in Y | \exists \sigma \in E^*_{uc} : \delta^*_s(y, \sigma) \in \hat{Y}_1\}$;
11: Remove all states and related transitions in $\hat{Y}_1 \cup \hat{Y}_2$ from Sup;
12: Output Sup.

Theorem 4.4: Given a system $G = (X, E, \delta, x_0)$, a secret $S \subseteq X$, the set of events observable by the intruder E_I, the set of events E_S observable by the supervisor, and the set of controllable events E_c. Let Assumptions A1 and A2 be satisfied. The supervisor obtained using Algorithm 2 is maximally permissive and the controlled system is current-state opaque wrt S and E_I. $\diamond$

Proof: We first prove that the controlled system is current-state opaque wrt S and E_I. By Algorithm 2, the supervisor obtained disables all events whose occurrence will lead to states in $\hat{Q}$. In other words, in the parallel observer there would be no state $q = (C_I, C_S)$ such that $C_I \subseteq S$. Therefore, by Corollary 4.3, the controlled system is current-state opaque wrt S and E_I.



(a) (b)

Fig. 5. (a) Observer of the PO in Fig. 4 for the supervisor; (b) Optimal supervisor enforcing current-state opacity.

Next we show that the supervisor is maximally permissive. Suppose that the supervisor Sup is not maximally permissive. Namely, there exists at least a state x and an controllable event $e \in E_c$ such that e is disabled by the supervisor but the occurrence of e at x does not violate opacity. Let $\sigma_x \in L(G)$ be such that $\delta(x_0, \sigma_x) = x$. Considering that the supervisor has only partial observation, for any $\sigma \in L(G)$ such that $P_S(\sigma) = P_S(\sigma_x)$, the following conditions hold:

- $\mathcal{C}_I(w_i e) \not\subseteq S$, where $w_i = P_I(\sigma)$, and
- $\forall \sigma_u \in E_{uc}^*$ and $w_i' = P_I(\sigma) e P_I(\sigma_u)$, $\mathcal{C}_I(w_i') \not\subseteq S$.

Since e is disabled at x by the supervisor, $\delta_s(y, e)$ is not defined in Sup, where $y = \delta_s(y_0, w_s)$ and $w_s = P_S(\sigma)$. Let $\delta_m(m_0, w_s) = m$ and $\delta_m(m, e) = m'$, where δ_m is the transition function of the observer $\mathcal{P}_s = (M, E_S, \delta_m, m_0)$. Since $\delta_s(y, e)$ is not defined in Sup, by Algorithm 2, $m' \in \hat{Y}_1 \cup \hat{Y}_2$. By Algorithm 1, $\forall q = (C_I, C_S) \in m$, the following conditions hold:

- sets C_S are equal to $\mathcal{C}_S(w_s)$, and
- set $C_I = \mathcal{C}_I(w_{is}) \not\subseteq S$, where $w_{is} = P_I(\sigma_s)$ and $\sigma_s \in L(G)$ is a sequence such that $P_S(\sigma_s) = w_s$.

By the assumption that the occurrence of e at x does not violate opacity, we have $\forall q' = (C_I', C_S') \in m'$, $C_I' = \mathcal{C}_I(w_{is} e) \not\subseteq S$, (i.e., $q' \notin \hat{Q}$), and $\forall \sigma_u \in E_{uc}^*, \mathcal{C}_I(w_{is} e P_I(\sigma_u)) \not\subseteq S$. Namely, $m' \notin \hat{Y}_1 \cup \hat{Y}_2$, which leads to a contradiction. ■

Example 4.5: Consider again the problem in Example 4.1. According to the result in Example 3.4, $\hat{Q} = \{q_5\}$. The observer $\mathcal{P}_s$ for the supervisor is shown in Fig. 5(a). Therefore, $\hat{Y}_1 = \{m_2\}$ and $\hat{Y}_2 = \emptyset$. Using Algorithm 2, the obtained supervisor is shown in Fig. 5(b). If the supervisor observes nothing, all events are enabled by the supervisor. However, when the supervisor observes $w_s = b$, the set of events allowed by the supervisor is $\{a, b, d\}$, i.e., c is disabled by the supervisor. ◇

C. Computational complexity analysis

According to the previous analysis, in the worst case the number of states of the PO is $2^{2|X|}$, therefore, due to the determinization of the PO (namely the construction of the observer), the complexity of Algorithm 2 is $\mathcal{O}(2^{2^{2|X|}})$. If

$E_I \subseteq E_S$ or $E_S \subseteq E_I$ holds, the proposed approach has the same complexity as the one in [13], [14] has. On the contrary, if neither $E_I \subseteq E_S$ nor $E_S \subseteq E_I$ holds, as discussed at the beginning of Section IV-B, the supervisory synthesis problem considered in the paper cannot be solved using the approaches in [13], [14].

V. Conclusions and Future Work

In this paper, we proposed a novel approach to solve the problem of current-state opacity enforcement in discrete event systems using finite automata. By constructing the parallel observer (PO), the states that will violate current-state opacity were characterized. Based on the PO, current-state opacity can be checked and a synthesis algorithm was provided to design the maximally permissive supervisor, without assuming the existence of containment relationships between the sets of events observable by the intruder and the supervisor. We also point out that the proposed approach can be extended to Petri nets, a model that is more powerful than finite automata. Moreover, some structural properties of Petri nets may be used to further reduce the computational complexity, which leads our future research.

References

[1] J. A. Goguen and J. Meseguer. Security policies and security models. In *2012 IEEE Symposium on Security and Privacy*, pages 11–11, 1982.

[2] N. Busi and R. Gorrieri. A survey on non-interference with Petri nets. In *Lectures on Concurrency and Petri Nets*, pages 328–344. Springer, 2004.

[3] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[4] V. Shmatikov. Probabilistic analysis of an anonymity system. *Journal of Computer Security*, 12(3):355–377, 2004.

[5] N. B. Hadj-Alouane, S. Lafrance, F. Lin, J. Mullins, and M. M. Yeddes. On the verification of intransitive noninterference in mulitlevel security. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 35(5):948–958, 2005.

[6] J. W. Bryans, M. Koutny, and P. Y. Ryan. Modelling opacity using Petri nets. *Electronic Notes in Theoretical Computer Science*, 121:101–115, 2005.

[7] A. Saboori and C. N. Hadjicostis. Verification of initial-state opacity in security applications of DES. In *9th International Workshop on Discrete Event Systems*, pages 328–333, 2008.

[8] F. Cassez, J. Dubreil, and H. Marchand. Dynamic observers for the synthesis of opaque systems. In *Automated Technology for Verification and Analysis*, pages 352–367. Springer, 2009.

[9] Y. C. Wu and S. Lafortune. Comparative analysis of related notions of opacity in centralized and coordinated architectures. *Discrete Event Dynamic Systems*, 23(3):307–339, 2013.

[10] Y. Tong, Z. W. Li, C. Seatzu, and A. Giua. Verification of current-state opacity using Petri nets. In *American Control Conference, 2015*, pages 1935–1940, Chicago, US, 2015. IEEE.

[11] P. J. G. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, 1989.

[12] E. Badouel, M. Bednarczyk, A. Borzyszkowski, B. Caillaud, and P. Darondeau. Concurrent secrets. *Discrete Event Dynamic Systems*, 17(4):425–446, 2007.

[13] J. Dubreil, P. Darondeau, and H. Marchand. Opacity enforcing control synthesis. In *9th International Workshop on Discrete Event Systems, 2008.*, pages 28–35. IEEE, 2008.

[14] J. Dubreil, P. Darondeau, and H. Marchand. Supervisory control for opacity. *IEEE Transactions on Automatic Control,*, 55(5):1089–1100, 2010.

[15] A. Saboori and C. N. Hadjicostis. Opacity-enforcing supervisory strategies via state estimator constructions. *Automatic Control, IEEE Transactions on*, 57(5):1155–1165, 2012.

[16] X. Yin and S. Lafortune. A new approach for synthesizing opacity-enforcing supervisors for partially-observed discrete-event systems. In *American Control Conference, 2015*, pages 377–383, Chicago, US, 2015. IEEE.

[17] C. G. Cassandras and S. Lafortune. *Introduction to discrete event systems*. Springer, 2008.

[18] A. Saboori and C.N. Hadjicostis. Notions of security and opacity in discrete event systems. In *Decision and Control, 2007 46th IEEE Conference on*, pages 5056–5061. IEEE, 2007.