Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

Opacity-Enforcing Supervisory Strategies for Secure Discrete Event Systems

Anooshiravan Saboori and Christoforos N. Hadjicostis

Abstract— **Initial-state opacity emerges as a key property in numerous security applications of discrete event systems including key-stream generators for cryptographic protocols. Specifically, a system is *initial-state opaque* if the membership of its true initial state to a set of *secret states* remains uncertain (opaque) to an outside intruder who observes system activity through a given projection map. In this paper, we consider the problem of constructing a minimally restrictive opacity-enforcing supervisor (MOES) which limits the system's behavior within some pre-specified legal behavior while enforcing the initial-state opacity requirement. To tackle this problem, we extend the state-based definition of initial-state opacity to languages and characterize the solution to MOES in terms of the supremal element of certain *controllable*, *observable* and *opaque* languages. We also derive conditions under which this supremal element exists and show how the *initial-state estimator*, which was introduced in our earlier work for verifying initial-state opacity, can be used to implement the solution to MOES.**

I. Introduction

The exchange of vital information over shared cyber-infrastructures has increased concerns about the vulnerability of such systems to intruders and other malicious entities. As a result, various notions of *security and privacy* have received considerable attention from researchers. *Opacity* is a security notion that aims at determining whether a given system's *secret* behavior (i.e., a subset of the behavior of the system that is considered critical and is usually represented by a predicate) is kept opaque to outsiders [1], [2]. More specifically, this requires that an intruder (modeled as an observer of the system's behavior) is never able to establish (with absolute certainty) the truth of the predicate.

In our earlier work [1], [3], we considered opacity with respect to predicates that are state-based. More specifically, given a discrete event system (DES) that can be modeled as a (possibly non-determinstic) finite automaton with partial observation on its transitions, we considered a scenario where the intruder's observability power can be modeled through the set of observable transitions (which is the subset of system transitions that are observable to outsiders). By partitioning the set of system states into *secret* and *non-secret* states, we considered, analyzed and characterized the notion of *initial-state opacity* in [3]. Initial-state opacity requires that, regardless of the underlying activity in the system, the sequence of (observable) transitions seen by the intruder never allows him/her to unambiguously determine that the initial state of the system belonged to the set of secret states. In [3], we developed a method for verifying initial-state opacity using an *initial-state estimator*.

In this paper, we consider the problem of designing a supervisor which can: (i) limit the system's behavior within some pre-specified legal behavior, and (ii) enforce initial-state opacity requirements by disabling at any given time (and based on the partial observations seen so far) the least number of possible events (i.e., the supervisor is minimally restrictive). We show that the solution to our problem is the supremal element (if one exists) of a set of languages which can be characterized as the intersection of controllable, observable, and *initial-state opaque* languages. We argue that, under certain conditions, the supremal element exists and derive a formulation for it. Moreover, assuming that the given legal behavior is *regular* (i.e., it can be described via a finite state machine), we show that the supremal element is also regular. In addition we propose a procedure that uses the initial-state estimator construction of [3] to enforce this supremal element, and effectively integrating the verification and control problems.

There has been some related work on the design of supervisors to enforce various types of security properties in DES [4], [5], [6], [7]. Following a *language-based* approach, the authors of [4] consider multiple observers with different observation capabilities (modeled through natural projection maps); opacity in this setting requires that no observer is able to determine whether the actual trajectory of the system belongs to the secret language assigned to that observer. The control problem seeks a supervisor that enforces such opacity for all observers of the system. The case of one particular observer is later considered in [5]. The authors of [6] model the intruder as an entity which can override the decision of the supervisor to disable certain events. The authors of [7] consider the problem of non-interference for timed automata. They partition the event set into *public* and *private* events and define non-interference as the property under which the system's public behavior is not affected by its private behavior.

In contrast to [4] and [5], opacity in our framework relies on the partitioning of system states into *secret* and *non-*

This material is based upon work supported in part by the National Science Foundation (USA), under NSF Career Award 0092696 and NSF ITR Award 0426831. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of NSF. The research leading to these results has also received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements INFSO-ICT-223844 and PIRG02-GA-2007-224877.

The first author is with the Coordinated Science Laboratory, and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign. The second author is with the Department of Electrical and Computer Engineering, University of Cyprus, and also with the Coordinated Science Laboratory, and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign. Corresponding author: C. N. Hadjicostis, 110 Green Park, 75 Kallipoleos Avenue, P.O. Box 20537, 1678 Nicosia, Cyprus (e-mail: chadjic@ucy.ac.cy).

secret ones; this state-based formulation enables us to use a state estimator to synthesize the supervisor. Compared to [6], the intruder in our framework is passive (modeled as an observer) and cannot change the system configuration or model. Also, in this paper, we are not seeking to *avoid* states as long as we can guarantee that entrance to these states retains the property of initial-state opaqueness. Our system model (untimed automaton) and our model of the intruder's capability (in terms of observability power) is different from [7] which makes the two frameworks rather incomparable.

II. Preliminaries and Notations

Let Σ be an alphabet of symbols (also called elements or events) and denote by Σ^* the set of all finite-length strings of elements of Σ, including the empty string ϵ (of length zero). A language $L \subseteq \Sigma^*$ is a subset of finite-length strings from Σ^*. For a string ω, $\overline{\omega}$ denotes the *prefix-closure* of ω and is defined as $\overline{\omega} = \{t \in \Sigma^* | \ \exists s \in \Sigma^* : ts = \omega\}$. The prefix closure $\overline{L}$ of language L is the set of all prefix closures of all strings in L. A language is prefix-closed if $L = \overline{L}$ [8].

A DES is modeled in this paper as a (possibly non-determinstic) finite automaton $G = (X, \Sigma, \delta, X_0)$, where $X = \{0, 1, \ldots, N-1\}$ is the set of states, Σ is the set of events, $\delta : X \times \Sigma \rightarrow 2^X$ is the (possibly partial) state transition function and $X_0 \subseteq X$ is the set of possible initial states. The function δ can be extended from the domain $X \times \Sigma$ to the domain $X \times \Sigma^*$ in the routine recursive manner $\delta(i, ts) := \bigcup_{j \in \delta(i,t)} \delta(j, s)$, for $t \in \Sigma$ and $s \in \Sigma^*$ with $\delta(i, \epsilon) := i$. We use $L(G, i)$ to denote the set of all traces that originate from state i of G (so that $L(G) = \bigcup_{i \in X_0} L(G, i)$). The prefix-closed language E is regular if there exists a finite automaton G such that $L(G) = E$.

The product of two automata $G_1 = (X_1, \Sigma_1, \delta_1, X_{01})$ and $G_2 = (X_2, \Sigma_2, \delta_2, X_{02})$ is the automaton $G_1 \times G_2 := AC(X_1 \times X_2, \Sigma_1 \cap \Sigma_2, \delta_{1\times 2}, X_{01} \times X_{02})$ where $\delta_{1\times 2}((i_1, i_2), \alpha) := \delta_1(i_1, \alpha) \times \delta_2(i_2, \alpha)$ if $\alpha \in \Sigma_1 \cap \Sigma_2$ and is undefined otherwise, and AC denotes the accessible part of the automaton (i.e., the set of states reachable from the set of initial states via some string $s \in \Sigma^*$). The construction of the product automaton implies that $L(G_1 \times G_2) = L(G_1) \cap L(G_2)$ [8].

We assume that only a subset Σ_{obs} of the events in Σ can be observed and monitored, and adopt the common assumption that Σ can be partitioned into two sets, Σ_{obs} and Σ_{uo}. The natural projection $P_{\Sigma_{obs}} : \Sigma^* \rightarrow \Sigma_{obs}^*$ can be used to map any trace executed in the system to the sequence of observations associated with it. This projection is defined recursively as $P_{\Sigma_{obs}}(\sigma s) = P_{\Sigma_{obs}}(\sigma) P_{\Sigma_{obs}}(s)$, for $\sigma \in \Sigma$ and $s \in \Sigma^*$, with $P_{\Sigma_{obs}}(\sigma) = \sigma$ if $\sigma \in \Sigma_{obs}$ and $P_{\Sigma_{obs}}(\sigma) = \epsilon$ if $\sigma \in \Sigma_{uo} \cup \{\epsilon\}$ (where ϵ represents the empty string [8]). In the sequel, the index Σ_{obs} in $P_{\Sigma_{obs}}$ will be dropped if it is clear from context. For any string s, $P^{-1}(s)$ denotes all of the strings in Σ^* that have projection s.

In the Ramadge and Wonham framework introduced in [9], it is further assumed that the event set Σ can be partitioned into the sets of controllable events (Σ_c) and uncontrollable events (Σ_{uc}), and control is achieved by means of a *supervisor* which at any given time can enable or disable one or more controllable events. Formally, given a system G, a *feasible* supervisor ν_o (subscript o denotes the partial observation) for G is a map $\nu_o : P(L(G)) \rightarrow \{\Sigma' \subseteq \Sigma | \Sigma_{uc} \subseteq \Sigma'\}$ which defines the set of events Σ' that remain enabled after observing a particular string from the system (note that Σ' necessarily includes all uncontrollable events). If we denote the closed-loop system by ν_o/G, the *minimally restrictive feasible supervisor problem* (*MS*) is defined as the design of a feasible supervisor ν_o such that: (i) $L(\nu_o/G) \subseteq E$ for a given (prefix-closed) language E that describes desirable behavior, and (ii) $L(\nu_o/G)$ is as large as possible (i.e., for any other feasible supervisor ν_o' such that $L(\nu_o'/G) \subseteq E$, we have $L(\nu_o'/G) \subseteq L(\nu_o/G)$).

The solution to the MS problem can be expressed in terms of certain *controllable* and *observable* languages. A language $K \subseteq L(G)$ is controllable [8] with respect to $L(G)$ and Σ_{uc} if $\overline{K}\Sigma_{uc} \cap L(G) \subseteq \overline{K}$. Also, K is said to be $(L(G), P)$–observable [8] if for all $s \in \Sigma^*$ and $\sigma \in \Sigma$, $s\sigma \notin \overline{K}$ and $s\sigma \in L(G)$ implies $P^{-1}(P(s))\sigma \cap \overline{K} = \emptyset$.

If we exclude requirement (ii) (that the supervisor is minimally restrictive) the following theorem from [8] characterizes all solutions to the supervisory control problem for certain prefix-closed languages.

Theorem 1 ([8]): Given a prefix-closed language $K \subseteq L(G)$, $K \neq \emptyset$, there exists a feasible supervisor ν_o for G such that $L(\nu_o/G) = K$ if and only if: (i) K is controllable with respect to $L(G)$ and Σ_{uc}, and (ii) K is observable with respect to $(L(G), P)$.

The existence of a minimally restrictive feasible supervisor is not guaranteed in general. However, if we choose to implement only *normal* sublanguages of E, then the minimally restrictive supervisor exists [10]. Formally, K is said to be $(L(G), P)$-normal if $K = L(G) \cap P^{-1}(P(K))$. Note that in the special case when $\Sigma_c \subseteq \Sigma_{obs}$, the notions of observability and normality become equivalent for a controllable language [8]. For any $E \subseteq L(G)$, we define $\mathcal{N}(E)(\mathcal{C}(E))$ to be the set of all prefix-closed sublanguages of E that are normal (controllable). The set $\mathcal{CN}(E) \equiv \mathcal{C}(E) \cap \mathcal{N}(E)$ is closed under union and hence there exists a unique supremal element $sup\mathcal{CN}(E)$ under the partial order of set inclusion for this set. We denote $sup\mathcal{CN}(E)$ by $E^{\uparrow CN}$. Using this, we can formulate the solution $\nu_o^{\uparrow CN}$ to MS, when limited to normal sublanguages of E, as $E^{\uparrow CN}$. The following lemma (taken from [11]) characterizes this solution. In the sequel, the superscript $^{\uparrow C_o}$ denotes the supremal controllable and prefix-closed sublanguage with respect to $P(L(G))$ and Σ_{uc}, and the superscript $^{\uparrow N}$ denotes the supremal prefix-closed and normal sublanguage with respect to $(L(G), P)$.

Lemma 1 ([11]): For any prefix-closed language $E \subseteq L(G)$, we have $E^{\uparrow CN} = L(G) \cap P^{-1}((P(E^{\uparrow N}))^{\uparrow C_o})$.

Another approach for defining supervisory control problems is the state-based approach where, instead of specifying the legal behavior as a prefix-closed language E, a set of forbidden states is provided via some predicate $R : X \rightarrow \{0, 1\}$ with $R(x) = 0$ capturing the fact that x is a forbidden

state. This set of forbidden states needs to be avoided via a *state-feedback* supervisor $\nu_s : X \rightarrow \{\Sigma' \subseteq \Sigma | \Sigma_{uc} \subseteq \Sigma'\}$ (Chapter 7 of [8]). The controller determines what controllable events to disable given the state the system is in. It can be shown that there exists a state-feedback supervisor such that all states x for which $R(x) = 1$ can be visited under supervision, if and only if R is *controllable* [8], i.e., if and only if it satisfies the following: (i) if state m satisfies R then m is reachable from the initial state of G via a string of states satisfying R, and (ii) at any of the visited states, uncontrollable events take the system to states which again satisfy R. If R is not controllable, we can seek a controllable predicate that best approximates R from below. Specifically, we say the predicate R_1 refines R_2 if for all $x \in X$, $R_1(x) = 1$ implies $R_2(x) = 1$. Now define $\mathcal{CR}(R)$ to be the set of all predicates that are controllable and refine R. Then $\mathcal{CR}(R)$ is closed under union and hence has supremal element $sup\mathcal{CR}(R)$ (denoted by $R^{\uparrow CR}$). Let $\nu_s^{\uparrow CR}$ be the state-feedback supervisor that synthesizes the predicate $sup\mathcal{CR}(R)$. Also denote by R/G the accessible part of automaton G when all the states that do not satisfy R are removed. Then, for any predicate R, we have [8]

$$L(\nu_s^{\uparrow CR}/G) = L^{\uparrow C}(R/G). \tag{1}$$

In other words, to find the state-feedback supervisor to synthesize the predicate $sup\mathcal{CR}(R)$, one can first remove all states that do not satisfy R and then find the supremal controllable sublanguage of the closed-behavior of the remaining graph.

III. Problem Formulation

In this section we recall the definition of initial-state opacity from [3] and the construction of the initial-state estimator (introduced in that paper) to verify this property.

Definition 1 (Initial-State Opacity): Given a (possibly non-determinstic) finite automaton $G = (X, \Sigma, \delta, X_0)$, a projection map P with respect to the set of observable events Σ_{obs}, and a set of secret states $S \subseteq X$, automaton G is initial-state opaque with respect to S and P (or (S, P, ∞) initial-state opaque) if for all $i \in X_0 \cap S$:

$$\forall t \in L(G, i), \exists j \in X_0 - S, \exists s \in L(G, j), P(s) = P(t).$$

According to Definition 1, system G is (S, P, ∞) initial-state opaque if for every string t that originates from an initial state in the set of secret states S there exists a string s that originates from an initial state outside S and has the same projection as t.

To verify initial-state opacity we can model the intruder with a state estimator, called the *initial-state estimator* (ISE), which uses the notion of a *state mapping* to reconstruct the states from which the sequence of observations seen up to this point could have been initiated [3]. A state mapping is a set whose elements are pairs of states: the first component of each element (pair) is the starting state and the second component is the ending state. The composition operation on state mappings is defined in [3] as follows: each element in the first state mapping can be combined with an element in the second mapping if the ending state of the former one is the same as the starting state of the latter one; the result is a new pair with the starting state taken to be the starting state of the first state mapping and the ending state taken to be the ending state of the second state mapping. The ISE in [3] utilizes state mappings as follows: each state of the ISE is associated with a unique state mapping and, since the initial state assumes nothing about the state of the system, the mapping associated with this initial state is the mapping $\{(x_i, x_i) | x_i \in X_0\}$ where starting and ending states are identical for all states in X_0. When the first observation is made, the *induced state mapping* corresponding to that observation is composed with the initial state mapping; the resulting state mapping becomes the next state of the state estimator. The induced state mapping for any observation s is defined in [3] as the state mapping whose pairs consist of starting and ending states such that there exists a sequence of events that starts from the starting state and ends at the ending state, while producing observation s. For subsequent observations, the current state of the ISE transitions into the state associated with the mapping that can be obtained via the composition of the previous state mapping and the mapping induced by the new observation. In this way we can build a structure which, at any time and based on the observations seen so far, gives information about possible pairs of starting (initial) states and ending (current) states through the state mappings associated with each of its states. Note that this structure is guaranteed to be finite and has at most 2^{N^2} states where N is the number of states of the DES G (and thus N^2 is the number of different pairs of starting and ending states).

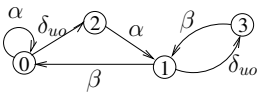

Fig. 1. DES G discussed in Example 1.

As mentioned before, the ISE can be used to verify initial-state opacity. In [3], it was shown that DES G is initial-state opaque if and only if for each of the state mappings associated with reachable states in its ISE, the set of starting states contains at least one element outside S. The following example demonstrates how the ISE can be used to verify initial-state opacity.

Example 1: Consider the automaton G of Figure 1 with $\Sigma_{obs} = \{\alpha, \beta\}$. Figure 2 shows the ISE for this system. The initial uncertainty is assumed to be equal to the state space and hence the initial state of the ISE m_0 is the state mapping $\{(0,0), (1,1), (2,2), (3,3)\}$. Upon observing α, the next state of the ISE becomes $m' = \{(0,0), (0,1), (0,2), (0,3), (2,1), (2,3)\} \equiv m_1$. Observe that α can be observed only from states 0 and 2; moreover, if the initial state was 0, the current state can be any of the states in $\{0,1,2,3\}$ but if the initial state was 2, the current state could only be $\{1,3\}$. Mapping m_1 summarizes this information with its pairs (on the right of Figure 2 we use a graphical way to describe the pairs associated with the ISE). Using this approach for all possible observations

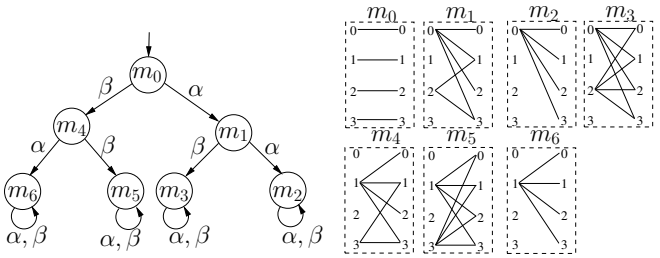

Fig. 2. ISE corresponding to G discussed in Example 1.

(from each state), the ISE construction can be completed (by composing the state mapping associated with the given state with the mapping induced by each possible observation) as shown in Figure 2. This system is not $(\{1\}, P, \infty)$ initial-state opaque due to the existence in the ISE of state $m_6 = \{(1,0),(1,1),(1,2),(1,3)\}$ whose set of starting states ($\{1\}$) is strictly within S. In other words, observing $\beta\alpha(\alpha+\beta)^*$ completely determines the initial state as state 1 which is within the set of secret states (and, hence, violates initial-state opacity). □

In this paper, using the verification method introduced in [3], we construct a feasible supervisor to limit the behavior of the system within a pre-specified legal behavior while enforcing initial-state opacity and while disabling the least possible number of controllable events. Before defining this problem formally, we need to clarify one issue. Opacity is defined to be a property of the states of the given finite automaton; however, the application of supervisory control to the system modifies the original structure of the automaton and hence its states. The remedy to this problem is to find a way to map states of the supervised system to states of the original system and, hence, re-define the set of secret states for the system under supervision to include all those states that are mapped to secret states in the original system. The following definition uses the product operator to map the states and hence extends the definition of opacity to the supervised system.

Definition 2 (Initial-State Opacity for Supervised System): Given a (possibly non-deterministic) finite automaton $G = (X, \Sigma, \delta, X_0)$, a projection map P with respect to the set of observable events Σ_{obs}, and a set of secret states $S \subseteq X$, we say that (possibly non-determinstic) automaton $G' = (X', \Sigma', \delta', X_0')$ is (S, P, ∞) initial-state opaque with respect to G if $G_p = G' \times G = (X_p, \Sigma_p, \delta_p, X_{0p})$ is (S_p, P, ∞) initial-state opaque where $X_{0p} = \{(x_o', x_o) | x_o' \in X_o', x_o \in X_o\}$ and $S_p = \{(x,y) \in X_{0p} | x \in X_0', y \in S\}$.

Using Definition 2, initial-state opacity is enforced under supervision if ν_o/G (the supervised system) is (S, P, ∞) initial-state opaque with respect to G. Note that this definition requires ν_o/G to be regular. A feasible supervisor that achieves this property is called an *opacity-enforcing* feasible supervisor for the system and is denoted by ν_{op}. Next, we define the minimally restrictive opacity-enforcing feasible supervisor (MOES) problem.

Definition 3 (MOES): Given a (possibly non-determinstic) finite automaton $G = (X, \Sigma, \delta, X_0)$, a set of controllable events Σ_c, $\Sigma_c \subseteq \Sigma_{obs}$, a projection map P with respect to the set of observable events Σ_{obs}, a set of secret states $S \subseteq X$, and a prefix-closed and regular language $E \subseteq L(G)$, find an opacity-enforcing feasible supervisor ν_{op} for G such that (i) $L(\nu_{op}/G) \subseteq E$, and (ii) $L(\nu_{op}/G)$ is as large as possible.

IV. Solution to MOES

A. Characterizing the Solution to MOES

In order to characterize the solutions to MOES using machinery that already exists in the literature on supervisory control (e.g., [8], [9]) we need to bring in a language-based formulation of the state-based notions of initial-state opacity.

Definition 4 (Language-Based Definition of Opacity): Given a (possibly non-determinstic) finite automaton $G = (X, \Sigma, \delta, X_0)$, a projection map P with respect to the set of observable events Σ_{obs}, and a set of secret states $S \subseteq X$, we say that language $K \subseteq L(G)$ is (S, P, ∞) initial-state opaque with respect to G if for all $i \in X_0 \cap S$, and $t \in L(G, i) \cap \bar{K}$

$$\exists j \in X_0 - S, s \in L(G, j) \cap \bar{K}, P(s) = P(t). \qquad (2)$$

Based on Definition 4, language K is (S, P, ∞) initial-state opaque with respect to G if for each string t in $\overline{K}$ that can originate from a secret state in G, there exists at least another string s in $\overline{K}$ that has the same projection as t and originates from a non-secret state in G. The following lemma relates Definition 4 to Definition 2 for a regular language K.

Lemma 2: Given a prefix-closed and regular language $K \subseteq L(G)$ and the (possibly non-determinstic) finite automaton $G_K = (X_K, \Sigma_K, \delta_K, X_{0K})$ such that $L(G_K) = K$, K is (S, P, ∞) initial-state opaque with respect to G if and only if G_K is (S, P, ∞) initial-state opaque with respect to G.

Using the notion of initial-state opacity for languages, we now characterize all opacity-enforcing feasible supervisors and derive the solution to MOES.

Theorem 2: Given a prefix-closed and regular language $K \subseteq L(G)$, $K \neq \emptyset$, there exists an opacity-enforcing feasible supervisor ν_{op} for G such that $L(\nu_{op}/G) = K$, if and only if: (i) K is controllable with respect to $L(G)$ and Σ_{uc}; (ii) K is observable with respect to $(L(G), P)$; (iii) K is (S, P, ∞) initial-state opaque with respect to G.

Proof: Follows from Theorem 1 and Lemma 2. ■

For any $E \subseteq L(G)$, define $\mathcal{O}(E)$ ($\mathcal{P}(E)$) to be the set of prefix-closed sublanguages of E that are observable (initial-state opaque). Then, using Theorem 2, for any supervisor ν_{op} that enforces initial-state opacity we have $L(\nu_{op}/G) \subseteq \mathcal{COP}(E) := \mathcal{C}(E) \cap \mathcal{O}(E) \cap \mathcal{P}(E)$. MOES assumes that $\Sigma_c \subseteq \Sigma_{obs}$ which implies that $\mathcal{CO}(E) = \mathcal{CN}(E)$, hence, $L(\nu_{op}/G) \subseteq \mathcal{CNP}(E)$. Since MOES requires the minimally restrictive opacity-enforcing feasible supervisor, the solution to MOES could be the supervisor $\nu_{op}^{\uparrow CNP}$ such that $L(\nu_{op}^{\uparrow CNP}/G) = sup\mathcal{CNP}(E) \equiv E^{\uparrow CNP}$. In the next section, we prove that such supremal element exists and provide a formulation for it.

B. Properties of Initial-State Opaque Languages

In this section, through various lemmas, we characterize the language $E^{\uparrow CNP}$ and hence obtain the solution to MOES. The first lemma characterizes the set $\mathcal{P}(E)$.

Lemma 3: For any language $E \subseteq L(G)$, we have $\mathcal{P}(E) = \{K \subseteq E |\ K = \overline{K}, K \subseteq L(G) \cap P^{-1}(P(K \cap L(G, X_0 - S)))\}$.

Proof: Define $V \equiv K \cap L(G, X_0 - S)$ and $W \equiv K \cap L(G, X_0 \cap S)$. From the definition of $\mathcal{P}(E)$ it follows that $K \in \mathcal{P}(E)$ implies that $P(W) \subseteq P(V)$. Moreover, we have $K = V \cup W$ which implies that $P(K) = P(V) \cup P(W)$ and since $P(W) \subseteq P(V)$, we have $P(K) = P(V)$ and, thus $K \subseteq P^{-1}(P(V))$. Moreover, $K \subseteq E \subseteq L(G)$, and hence $K \subseteq (L(G) \cap P^{-1}(P(V)))$ which concludes the proof. ■

The next lemma states that the set of initial-state opaque and prefix-closed sublanguages of E has a supremal element.

Lemma 4: $\mathcal{P}(E)$ is nonempty and closed under arbitrary unions; in particular, the supremal element $sup\mathcal{P}(E)$ exists in $\mathcal{P}(E)$.

Assuming that E is prefix closed, the following theorem derives a formulation for the supremal element $E^{\uparrow P}$ of $\mathcal{P}(E)$.

Theorem 3: For any prefix-closed language $E \subseteq L(G)$, we have $E^{\uparrow P} = E \cap P^{-1}(P(E \cap L(G, X_0 - S)))$.

Proof: Due to space limitation, we only provide a sketch of the proof. Define $H \equiv E \cap P^{-1}(P(E \cap L(G, X_0 - S)))$. To prove the theorem, we show that: (a) every $K \in \mathcal{P}(E)$ is a subset of H, which follows from Lemma 3, and (b) $H \in \mathcal{P}(E)$ which can be established by showing that: (i) H is initial-state opaque, and (ii) H is prefix-closed. ■

Corollary 1: Normality is preserved under $^{\uparrow P}$ operator for a prefix-closed and normal language $E \subseteq L(G)$.

Proof: The proof is omitted due to space limitation. ■

Next we characterize the set $\mathcal{NP}(E)$ using Lemma 3.

Lemma 5: For any $E \subseteq L(G)$, we have $\mathcal{NP}(E) = \{K \subseteq E |\ K = \overline{K}, K = L(G) \cap P^{-1}(P(K \cap L(G, X_0 - S)))\}$.

We complete the analysis on the properties of initial-state opaque languages by considering the controllability condition.

Lemma 6: Initial-state opacity is preserved under $^{\uparrow CN}$ operator for any prefix-closed and normal language $E \subseteq L(G)$.

The following theorem derives a formulation for $E^{\uparrow CNP}$.

Theorem 4: Given a prefix-closed and normal language $E \subseteq L(G)$, we have $E^{\uparrow CNP} = L(G) \cap P^{-1}((P((E^{\uparrow N})^{\uparrow P}))^{\uparrow C_o})$.

Proof: By Lemma 6, initial-state opacity is preserved under $^{\uparrow CN}$ operator for normal languages; hence $E^{\uparrow CNP} = (E^{\uparrow NP})^{\uparrow CN}$. Also by Corollary 1, normality is preserved under the $^{\uparrow P}$ operator for normal languages; therefore $E^{\uparrow NP} = (E^{\uparrow N})^{\uparrow P}$. Putting these two together, we have $E^{\uparrow CNP} = ((E^{\uparrow N})^{\uparrow P})^{\uparrow CN}$. Using Lemma 1, the proof is complete. ■

C. Implementing the Solution to MOES using the Initial-State Estimator

MOES requires the minimally restrictive opacity-enforcing feasible supervisor ν_{op} that can enforce the legal behavior described via the prefix-closed language E. Theorem 2 states that the solution to MOES boils down to $E^{\uparrow CNP}$ and Theorem 4 (assuming that E is normal) characterizes this solution as $E^{\uparrow CNP} = L(G) \cap P^{-1}((P(E^{\uparrow P}))^{\uparrow C_o})$. Observe that Theorem 3 characterizes $E^{\uparrow P} = E \cap P^{-1}(P(E \cap L(G, X_0 - S)))$ which implies that $E^{\uparrow P}$ can be implemented using projection and intersection operations on languages. Also [11] gives a formulation for the supremal controllable sublanguage $E^{\uparrow C}$ (assuming that E is prefix-closed) using *concatenation*, intersection and *complement* operations on languages. For regular languages, all these operations can be implemented using operations on finite state machines [8]. Hence, since in MOES E is assumed to be regular, $E^{\uparrow CNP}$ can be obtained (and hence the solution to MOES can be implemented) via certain operations on automata describing E and G. This implies that $E^{\uparrow CNP}$ is regular. In this section, we take a different approach for implementing the solution to MOES and introduce an algorithm which uses the ISE structure to obtain the solution. Observe that the ISE construction can be used for verifying initial-state opacity and, hence, the proposed algorithm integrates the verification and control problems. In the sequel we assume, without loss of generality, that E is normal (if this assumption is not satisfied, we can always first compute $E^{\uparrow N}$ using the results in [11] and then proceed with the following).

Algorithm A: Given a (possibly non-determinstic) finite automaton $G = (X, \Sigma, \delta, X_0)$, a set of controllable events Σ_c, $\Sigma_c \subseteq \Sigma_{obs}$, a set of secret states $S \subseteq X$, and a prefix-closed, normal and regular language $E \subseteq L(G)$, $E \neq \emptyset$, describing the legal behavior, we can obtain a minimally restrictive supervisor $\nu_s^{\uparrow CR}$ for G via the following steps: (i) Construct automaton $G_E = (X_E, \Sigma, \delta_E, X_{0E})$ such that $L(G_E) = E$. (ii) Construct $G_p = G_E \times G = (X_p, \Sigma, \delta_p, X_{0p})$; define $S_p \equiv \{(x, y) \in X_{0p} | x \in X_{0E}, y \in S\}$. (iii) Construct the ISE $G^p_{\infty,obs}$ corresponding to G_p. (iv) Construct the state-feedback supervisor $\nu_s^{\uparrow CR}$ for $G^p_{\infty,obs}$ that avoids all states in $G^p_{\infty,obs}$ for which the set of starting states of the associated state mapping contains no state outside S_p; for this, first define the predicate R to be true on such states, and then construct the accessible part $R/G^p_{\infty,obs}$ of the ISE $G^p_{\infty,obs}$ when all the states that do not satisfy R are removed. (v) Check for controllability condition on the remaining graph.

Theorem 5: Given a prefix-closed, normal and regular language $E \subseteq L(G)$, the control action of the solution $\nu_{op}^{\uparrow CNP}$ to MOES after observing s is the same as the control action of the state-feedback supervisor $\nu_s^{\uparrow CR}$ (as synthesized by Algorithm A) at the state reached in $G^p_{\infty,obs}$ via s.

Proof: We first show that steps (i)-(iv) implement $P(E^{\uparrow P})$. In [3], we showed that in the ISE $G_{\infty,obs}$ for G, $P(L(G, X_0 - S))$ is characterized via the set of strings in $G_{\infty,obs}$ that start from the initial state and reach a state in $G_{\infty,obs}$ for which the associated state mapping contains at least one state outside the set of secret states S in its set of starting states. Using this result, we can argue that $P(E \cap L(G, X_0 - S))$ is characterized via the set of

strings in $G^p_{\infty,obs}$ (as constructed in Algorithm A) that start from initial state $X^p_{0,obs}$ and reach a state in $G^p_{\infty,obs}$ for which the associated state mapping contains at least one state outside S_p in its set of starting states. Theorem 3 states that $E^{\uparrow P} = E \cap P^{-1}(P(E \cap L(G, X_0 - S)))$ which implies that $P(E^{\uparrow P}) = P(E \cap L(G, X_0 - S))$. This means that one can implement $P(E^{\uparrow P})$ by removing in the ISE the states that do not satisfy R, i.e., $L(R/G^p_{\infty,obs}) = P(E^{\uparrow P})$. Removing states and the associated labels from ISE $G_{\infty,obs}$ (i.e., evaluating $R/G^p_{\infty,obs}$) might violate the controllability condition, hence in step (v) the algorithm implements $(P(E^{\uparrow P}))^{\uparrow C_o}$; therefore, the state-feedback supervisor $\nu_s^{\uparrow CR}$ synthesizes the predicate $sup\mathcal{CR}(R)$. By (1), $L(\nu_s^{\uparrow CR}/G^p_{\infty,obs}) = (L(R/G^p_{\infty,obs}))^{\uparrow C_o} = (P(E^{\uparrow P}))^{\uparrow C_o}$.

Finally, to prove the theorem we need to show that $\nu_{op}(s) \equiv \nu_s^{\uparrow CR}(\delta^p_{\infty,obs}(X^p_{0,obs}, s))$ is the solution to MOES where $\delta^p_{\infty,obs}(X^p_{0,obs}, s)$ denotes the state in $G^p_{\infty,obs}$ reached via s. For this, we can equivalently prove that the transition structure of the automaton $\tilde{G} \equiv \nu_s^{\uparrow CR}/G^p_{\infty,obs}$ can be used to implement the solution to MOES as follows: a string s can be executed in the closed loop system ν_{op}/G if its projection $P(s)$ belongs to $L(\tilde{G})$. For this, we show that: (i) $\tilde{G}$ is an opacity-enforcing feasible supervisor, and (ii) $L(\tilde{G}/G) = E^{\uparrow CNP}$. Note that automaton $\tilde{G}$ is a feasible supervisor for G since $L(\tilde{G})$ is controllable with respect to $P(L(G))$ [8]. Moreover, $L(\tilde{G})$ is initial-state opaque by construction. Hence, $\tilde{G}$ is an opacity-enforcing feasible supervisor. To show (ii), note that the supervisor $\tilde{G}$ observes the behavior through the projection map P; therefore, the behavior of $\tilde{G}$ can be described via $L(\tilde{G}/G) = L(G) \cap P^{-1}(L(\tilde{G}))$. Moreover, $L(G) \cap P^{-1}(L(\tilde{G})) = L(G) \cap P^{-1}((P(E^{\uparrow P}))^{\uparrow C_o}) = E^{\uparrow CNP}$ which results in $L(\tilde{G}/G) = E^{\uparrow CNP}$. ■

Example 2: Consider the DES G in Figure 1. Assume that $\Sigma_c = \{\beta\}$. Also suppose that $E = L(G)$ meaning that the only requirement for the supervisor is to enforce initial-state opacity. This implies that steps (i)-(iii) of Algorithm A can be replaced by the construction of the ISE for G. Figure 2 depicts this ISE for G ($G_{\infty,obs}$). As mentioned in Example 1, this system is not $(\{1\}, P, \infty)$ initial-state opaque due to the existence of state m_6 in the ISE. To obtain the minimally-restrictive feasible supervisor that enforces $(\{1\}, P, \infty)$ initial-state opacity, following step (iv) of Algorithm A, we first remove from $G_{\infty,obs}$ the state that violates initial-state opacity, i.e., m_6; Figure 3-a depicts the accessible part of the remaining automaton $R/G_{\infty,obs}$. Next, following step (v) of Algorithm A, we check for controllability condition. At state m_4, α is disabled which is an uncontrollable event. Hence, access to state m_4 should be rejected earlier, which is accomplished by disabling β at state m_0. Figure 3-b depicts the automaton associated with the supremal controllable sublanguage of the automaton in Figure 3-a. Based on this, the supervisor does not allow observing β as the first observation. Indeed, observing $\beta\alpha$ determines the initial state to be state 1 which is a secret set and hence violates initial-state opacity. □

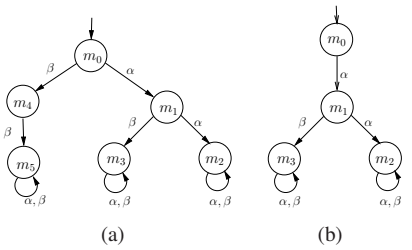

Fig. 3. (a) $R/G_{\infty,obs}$ (b) $\tilde{G} \equiv \nu_s^{\uparrow CR}/G_{\infty,obs}$ as well as the minimally restrictive opacity-enforcing supervisor $\nu_{op}^{\uparrow CNP}$ for G in Figure 1.

V. Conclusions

In this paper, we consider the problem of designing feasible supervisors that enforce initial-state opacity while limiting the behavior of the system to a subset behavior, called legal behavior and described by a prefix-closed language E. We show that there always exists a solution to this problem and characterize the set of solutions as the set of sublanguages of E that are controllable, observable, and initial-state opaque. We show that under the assumption that $\Sigma_c \subseteq \Sigma_{obs}$, there always exists a minimally restrictive solution to this problem and propose a method to find the supremal of such languages (which is the solution of our minimally restrictive supervisory control problem).

In the future, we are interested in introducing probability metrics to this framework. The control problem can then be connected to the design of control policies for stochastic systems under suitable optimality criteria for probabilistic opacity.

References

[1] A. Saboori and C. N. Hadjicostis, "Notions of security and opacity in discrete event systems," in *Proc. of the 46th IEEE Conference on Decision and Control*, December 2007, pp. 5056–5061.

[2] J. Bryans, M. Koutny, L. Mazare, and P. Ryan, "Opacity generalised to transition systems," in *Proc. of the 3rd International Workshop on Formal Aspects in Security and Trust*, July 2005, pp. 81–95.

[3] A. Saboori and C. N. Hadjicostis, "Initial–state estimation and its application to security problems," in *Proc. of the 9th International Workshop on Discrete Event Systems*, May 2008, pp. 328–333.

[4] E. Badouel, M. Bednarczyk, A. Borzyszkowski, B. Caillaud, and Ph. Darondeau, "Concurrent secrets," in *Proc. of the 8th International Workshop on Discrete Event Systems*, July 2006, pp. 51–57.

[5] J. Dubreil, Ph. Darondeau, and H. Marchand, "Opacity enforcing control synthesis," in *Proc. of the 9th International Workshop on Discrete Event Systems*, May 2008, pp. 28–35.

[6] D. Thorsley and D. Teneketzis, "Intrusion detection in controlled discrete event systems," in *Proc. of the 45th IEEE Conference on Decision and Control*, December 2006, pp. 6047–6054.

[7] G. Gardey, J. Mullins, and O. H. Roux, "Non–interference control synthesis for security timed automata," *Electronic Notes in Theoretical Computer Science*, vol. 180, pp. 35–53, June 2007.

[8] W. Wonham, *Supervisory Control of Discrete–Event Systems*, Systems and Control Group, Department of Electrical and Computer Engineering, University of Toronto. Available at `www.utoronto.ca/DES`, 2005.

[9] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM Journal on Control and Optimization*, vol. 25, pp. 206–230, January 1987.

[10] F. Lin and W. Wonham, "On observability of discrete event systems," *Information Sciences*, vol. 44, no. 3, pp. 173–198, April 1988.

[11] R. D. Brandt, V. Garg, R. Kumar, F. Lin, S. I. Marcus, and W. M. Wonham, "Formulas for calculating supremal controllable and normal sublanguages," *System and Control Letters*, vol. 15, no. 2, pp. 111–117, 1990.