2019 IEEE 58th Conference on Decision and Control (CDC)
Palais des Congrès et des Expositions Nice Acropolis
Nice, France, December 11-13, 2019

Opacity of Networked Supervisory Control Systems over Insecure Multiple Channel Networks

Xiang Yin and Shaoyuan Li

***Abstract*— In this paper, we investigate the security issue in networked supervisory control systems over multiple channel networks. We consider a networked discrete-event system controlled by a supervisor that receives information from sensors and sends control decisions to actuators via observations channels and control channels, respectively. The security problem is studied for the scenario where some of the communication channels are insecure in the sense there exists a passive intruder (eavesdropper) that can access the information-flow in those insecure communication channels. We adopt the concept of opacity, an information flow security property, to characterize the security status of the supervisory control system. Specifically, we assume that system has a secret and the system is said to be opaque if the intruder can never determine the secret of the system unambiguously based on the information-flow in the insecure channels. A new network observer is proposed to estimate the state of the system with two-side incomparable channel information. We show that the opacity verification problem for the networked setting can be effectively solved using the proposed network observer.**

I. Introduction

Supervisory control theory is a formal approach for controller synthesis of Discrete-Event Systems (DES) with provable correctness guarantees. In the supervisory control theory, the system/plant, which is modeled as a DES, is controlled by a *supervisor* that disables/enables the occurrences of events dynamically based on its observation such that the closed-loop system under control meets some desired design specification.

In essence, a supervisor is a decision making module. In many modern applications, the supervisor and the plant are connected via *communications networks*, where the supervisor receives sensor readings via observation channels and sends commands via control channels. Control systems with such networked information architecture are referred to as *networked control systems* (NCSs). Compared with traditional control architectures, NCS provide a more flexible way for controlling a system, e.g., we can implement the controller in the cloud utilizing more powerful computation resources. Therefore, supervisory control of *networked discrete-event systems* has also drawn considerable attention in the past few years; see, e.g., [1]–[4].

Although networked control systems have many advantages, it also brings new research challenges. One of the major challenges in NCSs is the security/privacy issue. In particular, communication channels in NCSs may be *insecure* in the sense that the information transmission may be "listened" by an intruder (eavesdropper) that is potentially malicious. In other words, the networked architecture may cause information leak which may further reveal some "secret" of the system.

This work was supported by the National Natural Science Foundation of China (61803259, 61833012).

Xiang Yin and Shaoyuan Li are with Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China. yinxiang@sjtu.edu.cn.

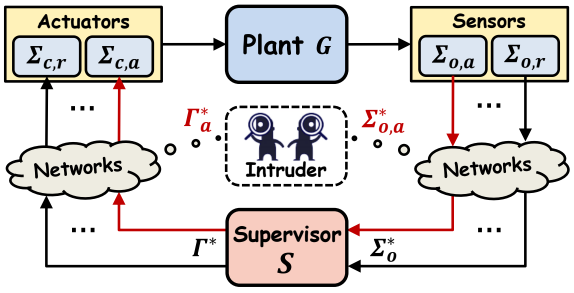


Fig. 1. A networked supervisory control system with insecure multiple channel networks. Sensors and actuators in the plant are modeled as observable events Σ_o and controllable events Σ_c, respectively. Sets $\Sigma_{o,a} \subseteq \Sigma_o$ and $\Sigma_{c,a} \subseteq \Sigma_c$ denote sensors and actuators whose communication channels are insecure, respectively.

In this paper, we propose a new framework for investigating the security issue in networked supervisory control systems over multiple channel networks. The information structure of the networked supervisory control system investigated in this paper is depicted in Figure 1. We consider a plant G modeled as a discrete-event system. We assume that the supervisor receives observable events from sensors via observation channels and sends control decisions to actuators via control channels. We consider the general scenario where information is transmitted via *multiple channel networks*, i.e., different actuators and sensors may transmit information using different channels. We further assume that some of the observation/control channels are *insecure* in the sense that there exists a passive intruder knowing the information transmitted in those insecure channels.

To characterize the security status of the networked supervisory control system, we adopt the concept of an information-flow security property called *opacity*. Specifically, we assume that the networked supervisory control system has a "secret" that does not want to be revealed to the intruder. We say that the system is *opaque* if the intruder can never determine that the system is at a secret state unambiguously by "listening" those insecure observation and control channels. We then investigate the verification of opacity. In particular, we show that the verification problem can be effectively solved by constructing a new information structure called the *network observer* that estimates the state of the system by correctly fusing the incomparable information in the observation and control channels.

In the DES literature, the notion of opacity has also been studied very extensively in the past few years; see, e.g., [5]–[19]. However, most of the existing works assume that the intruder observes a set of events of the system; this essentially corresponds to our setting of insecure observation channels. In [20], [21], more general observation models of the intruder are considered. However, their models do not explicitly capture the information-flow in networked supervisory control systems. In our recent work [22], we consider opacity in networked supervisory control systems for the case of a single insecure control channel. In this paper, we consider a more general case of *multiple channel networks*, where both control channels and observation channels can be insecure. This general setting is fundamentally more difficult than the one-side and single-channel case studied in [22] as in our multiple-channel setting, information in control channels and information in observation channels are *incomparable*. Hence, a new state estimation technique is needed to handle this general case.

II. Preliminary

A. System Model

Let Σ be a finite set of events. We call a finite sequence of events a *string* and we denote by Σ^* the set of all strings over Σ including the empty string ϵ. We define $\Sigma^\epsilon = \Sigma \cup \{\epsilon\}$. For any string $s \in \Sigma^*$, we denote by $|s|$ its length, i.e., the number of event occurrences in it, with $|\epsilon| = 0$. A *language* is a set of strings. For any language $L \subseteq \Sigma^*$, we denote by $\overline{L}$ its *prefix-closure*, i.e., $\overline{L} = \{t \in \Sigma^* : \exists w \in \Sigma^* \text{ s.t. } tw \in L\}$.

We consider a DES modeled as a deterministic finite-state automaton (DFA) $G = (X, \Sigma, \delta, x_0)$, where X is the finite set of state, Σ is the finite set of events, $\delta : X \times \Sigma \to X$ is the partial deterministic transition function, and $x_0 \in X$ is the initial state. For any $x, x' \in X$ and $\sigma \in \Sigma$, $\delta(x, \sigma) = x'$ means that there exists a transition from x to x' with event label σ. We define $E_G(x)$ as the set of events defined at state $x \in X$, i.e., $E_G(x) = \{\sigma \in \Sigma : \delta(x, \sigma)!\}$, where "!" means "is defined". The transition function is also extended to $\delta : X \times \Sigma^* \to X$ recursively in the usual manner; see, e.g., [23]. For the sake of simplicity, we write $\delta(x, s)$ as $\delta(s)$ when $x = x_0$. The language generated by G is $\mathcal{L}(G) = \{s \in \Sigma^* : \delta(s)!\}$.

Let $\hat{\Sigma} \subseteq \Sigma$ be a set of events. The natural projection from Σ to $\hat{\Sigma}$ is a mapping $P_{\hat{\Sigma}} : \Sigma^* \to \hat{\Sigma}^*$ defined recursively by: for any $s \in \Sigma^*, \sigma \in \Sigma$, we have

$$P_{\hat{\Sigma}}(\epsilon) = \epsilon \;\text{ and }\; P_{\hat{\Sigma}}(s\sigma) = \begin{cases} P_{\hat{\Sigma}}(s)\sigma & \text{if } \sigma \in \hat{\Sigma} \\ P_{\hat{\Sigma}}(s) & \text{if } \sigma \notin \hat{\Sigma} \end{cases} \tag{1}$$

The natural projection is also extended to $P_{\hat{\Sigma}} : 2^{\Sigma^*} \to 2^{\hat{\Sigma}^*}$ by: for any $L \subseteq \Sigma^*$, $P_{\hat{\Sigma}}(L) = \{P_{\hat{\Sigma}}(s) \in \hat{\Sigma}^* : s \in L\}$.

B. Supervisory Control Systems

In the supervisory control framework, system G is controlled by a *supervisor* that restricts the behavior of the system dynamically such that some desired closed-loop requirement is fulfilled. We assume that the event set is partitioned as

$$\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc} = \Sigma_o \dot{\cup} \Sigma_{uo}$$

where $\Sigma_c, \Sigma_{uc}, \Sigma_o$ and Σ_{uo} are the sets of controllable events, uncontrollable events, observable events and unobservable events, respectively.

A supervisor is a mechanism that disables/enables controllable events dynamically based on its observation. That is, supervisor cannot disable the occurrence of an uncontrollable event or observe the occurrence of an unobservable event. More formally, a supervisor can be modeled as a new DFA

$$S = (Z, \Sigma, \xi, z_0)$$

such that the following two conditions hold:

- $(\forall z \in Z)[\Sigma_{uc} \subseteq E_S(z)]$; and
- $(\forall z, z' \in Z, \sigma \in \Sigma : \xi(z, \sigma) = z')[z \neq z' \Rightarrow \sigma \in \Sigma_o]$.

Intuitively, supervisor S works as follows. When string $s \in \mathcal{L}(G)$ is generated by the system (if allowed), the supervisor reaches state $z = \xi(s) \in Z$. Then it decides to enable events $E_S(z)$ currently. We refer to $E_S(\xi(s))$ as the *control decision* upon the occurrence of s. For the sake of simplicity, hereafter, we also define $S(s) := E_S(\xi(s))$. Therefore, the first condition essentially requires that uncontrollable events should always be enabled by the supervisor. Also, we note that the supervisor can only update its control decision upon the occurrence of an observable event; this is captured by the second condition. By the second condition, we know that $S(s) = S(P_{\Sigma_o}(s))$ for any $s \in \Sigma^*$).

The closed-loop system under control is

$$G \times S = (X \times Z, \Sigma, \eta, (x_0, z_0))$$

where for each $(x, z) \in X \times Z$ and each $\sigma \in \Sigma$, we have

$$\eta((x, z), \sigma) = \begin{cases} (x', z') & \text{if } \delta(x, \sigma) = x' \wedge \xi(z, \sigma) = z' \\ \text{undefined} & \text{otherwise} \end{cases}$$

III. Modeling of Network Information Flow and Its Opacity

The supervisory control framework described in the previous section provides a mathematical model of how a supervisor works. Note that, from the implementation point of view, each controllable event essentially represents a corresponding *actuator* that controls the plant physically and each observable event represents a corresponding *sensor* that reads the occurrence of event. Therefore, in essence, the supervisor is a *decision making module* and it needs to interact with the system physically via sensors and actuators in the plant in order to control the system. As depicted in Figure 1, when implementing a supervisor in the networked environment, each sensor needs to send its reading (occurrence of observable event) to the supervisor via its corresponding observation channel and the supervisor needs to send the enable/disable decision for each controllable event to the corresponding actuator via its control channel.

In the networked environment, however, the communication channels may not always be secure and some information transmitted between the supervisor and the plant may be "listened" by an intruder that is potential malicious. The question naturally arises, therefore, whether or not the system is still secure in the presence of such insecure communication networks. In this paper, we propose a framework to analyze security of networked supervisory control systems using the concept of *opacity*.

To formulate the information security problem, first, we consider the information-flow in observation channels from sensors to supervisor. When the system generates an observable event, its occurrence can be detected by the associated sensor, which then sends this information to the supervisor via its observation channel. To model insecure observation channels, we assume that observable events Σ_o are further partitioned as

$$\Sigma_o = \Sigma_{o,r} \dot{\cup} \Sigma_{o,a}$$

where $\Sigma_{o,r}$ denotes the set of events whose observation channels are secure ("r" stands for "reliable") and $\Sigma_{o,a}$ denotes the set of events whose observation channels are insecure ("a" stands for "attackable"). Therefore, the intruder can only observe event transmission in $\Sigma_{o,a}$.

Note that, at each instant, there is only one sensor sending information to the supervisor since the system cannot generate two events simultaneously. However, the information-flow in control channels is more involved. Although upon the observation of $P_{\Sigma_o}(s)$, the control decision made by the supervisor is $S(P_{\Sigma_o}(s))$. As we discussed above, in multi-channel networks, this control decision is not sent to the plant directly as a "package". Instead, the supervisor needs to send disable/enable command to each actuator that corresponds to each controllable event individually. In this regard, although mathematically equivalent, it is more meaningful to interpreted the supervisor as a mapping

$$S : P_{\Sigma_o}(\mathcal{L}(G)) \times \Sigma_c \to \{Disable, Enable\}$$

To model insecure control channels, similar to the case of observation channel, we also assume that controllable event set Σ_c is further partitioned as

$$\Sigma_c = \Sigma_{c,r} \dot{\cup} \Sigma_{c,a}$$

where $\Sigma_{c,r}$ denotes the set of controllable events whose control channels are secure and $\Sigma_{c,a}$ denotes the set of controllable events whose control channels are insecure. Unlike the case of observation channels, where only one sensor will send information to the supervisor at each instant, the supervisor needs to send control decisions to *all* actuators simultaneously. Therefore, when the supervisor sends control decision $\gamma \in 2^{\Sigma_c}$ to each actuator, the intruder can only obtain information $[\gamma]_{\Sigma_{c,a}} := \gamma \cap \Sigma_{c,a}$, which is a *projected control decision*. We denote by $\Gamma_a := 2^{\Sigma_{c,a}}$ the set of all projected control decisions. Note that $[\gamma]_{\Sigma_{c,a}} = \emptyset$ does not imply that the intruder observes nothing in control channels; it means that the intruder knows that the supervisor is disabling all events in $\Sigma_{c,a}$.

Let $s \in \mathcal{L}(G \times S)$ be a string generated by the closed-loop system and suppose $P_{\Sigma_o}(s) = \sigma_1\sigma_2\cdots\sigma_n$. Then the information-flow released in the communication channels from *the intruder's point of view* is the following sequence

$$\begin{aligned} I_S(s) = & P_{\Sigma_{o,a}}(\sigma_1)[S(\sigma_1)]_{\Sigma_{c,a}} P_{\Sigma_{o,a}}(\sigma_1)[S(\sigma_1\sigma_2)]_{\Sigma_{c,a}} \\ & \cdots P_{\Sigma_{o,a}}(\sigma_n)[S(\sigma_1\sigma_2\cdots\sigma_n)]_{\Sigma_{c,a}} \in (\Sigma_{o,a}^{\epsilon}\Gamma_a)^*. \end{aligned} \quad (2)$$

We denote by

$$\mathcal{I}_S = \{I_S(s) \in (\Sigma_{o,a}^{\epsilon}\Gamma_a)^* : s \in \mathcal{L}(G \times S)\}$$

the set of all information-flows available to the intruder.

To summarize, we consider an information security problem of a networked supervisory control system in the presence of an intruder having the following capabilities:

- The intruder knows both the system model and the functionality of the supervisor;
- The observation channels and the control channels are partially secure in the sense that the intruder knows information-flow $\mathcal{I}_S(s)$ when string s is executed.

Remark 1: We assume that $\Sigma_{c,a} \neq \emptyset$. Otherwise, the intruder will only observe the projected behavior of the system w.r.t. event set $\Sigma_{o,a}$, which boils down to the standard opacity analysis problem. Therefore, the intruder always has an observation in control channels when the supervisor sends a control decision. However, it may not be able to distinguish two control decisions $\gamma_1, \gamma_2 \in 2^{\Sigma}$ such that $[S(\gamma_1)]_{\Sigma_{c,a}} = [S(\gamma_2)]_{\Sigma_{c,a}}$.

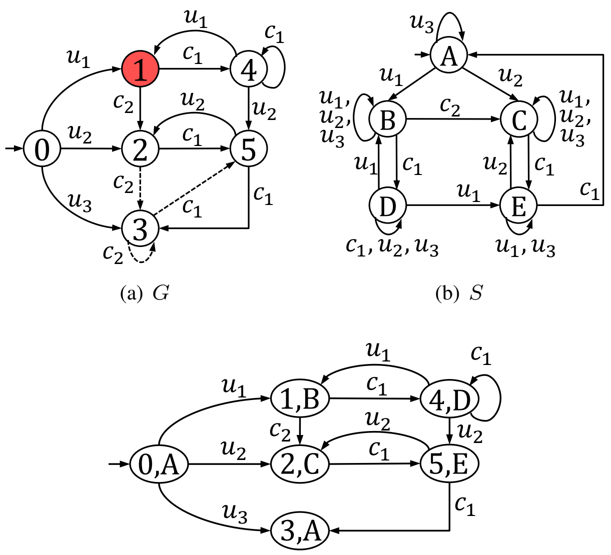
Remark 2: The reason why the information-flow defined in Equation (2) starts from an observable event and ends up with a control decision is as follows. We do not consider the initial control decision $S(\epsilon)$ in the information flow since any string generated by the closed-loop system will start with the same control decision. In other words, the initial control decision does not carry any information about the state of the system when the functionality of the supervisor is known. Also, we assume implicitly that the supervisor will send a control decision *immediately* after it receives a new observable event. This is why the information flow ends up with a control decision rather than an observable event.

To characterize the security status of the supervisory control system, we adopt the concept of *opacity*. Specifically, we assume that the system has a "secret", which is modeled as a set of secret states $X_{secret} \subset X$. We say that the overall networked control system is *opaque* if the intruder can never know for sure that the system is currently at a secret state based on the information released in the communication channels. This is formalized by the following definition.

Definition 1: Supervisory control system $G \times S$ is said to be *opaque* w.r.t. insecure observation channels $\Sigma_{o,a}$, insecure control channels $\Sigma_{c,a}$ and secret states X_{secret} if

$$\begin{aligned} & (\forall s \in \mathcal{L}(G \times S) : \delta(s) \in X_{secret}) \\ & (\exists t \in \mathcal{L}(G \times S) : \delta(t) \notin X_{secret})[\mathcal{I}_S(s) = \mathcal{I}_S(t)] \end{aligned} \quad (3)$$

Intuitively, opacity in the above definition requires that, for any string that goes to a secret state, there exists another



(a) G (b) S

(c) $G \times S$

Fig. 2. A supervisory control system (G, S).

string that goes to a non-secret state, such that they will generate the same information-flow, from the intruder's point of view, in the communication networks.

Example 1: Let us consider system G shown in Figure 2(a) with $\Sigma_c = \{c_1, c_2\}$ and $\Sigma_o = \Sigma$. We further assume that $\Sigma_{c,a} = \{c_1\}$ and $\Sigma_{o,a} = \{c_1, c_2\}$, i.e., the intruder can only observe the occurrence of events c_1 and c_2 and knows the control decision for c_1. Suppose that the system is controlled by supervisor S shown in Figure 2(b) and the closed-loop system $G \times S$ is then shown in Figure 2(c). The control objective is simply to avoid the occurrence of dashed transitions in Figure 2(a). We assume that $X_{Secret} = \{1\}$, i.e., we do not want the intruder to know for sure that the system is at secret state 1.

Let us consider the occurrence of string $u_1c_1u_1 \in \mathcal{L}(G \times S)$ that leads to secret state 1. Since supervisor S can observe all events, it will issue a control decision sequence $S(u_1)S(u_1c_1)S(u_1c_1u_1) = \{c_1, c_2\}\{c_1\}\{c_1, c_2\}$. Then from the intruder's point of view, the information-flow is

$$\begin{aligned} I_S(u_1c_1u_1) =& P_{\Sigma_{o,a}}(u_1)[\{c_1, c_2\}]_{\Sigma_{c,a}} P_{\Sigma_{o,a}}(c_1)[\{c_1\}]_{\Sigma_{c,a}} \\ & P_{\Sigma_{o,a}}(u_1)[\{c_1, c_2\}]_{\Sigma_{c,a}} = \{c_1\}c_1\{c_1\}\{c_1\} \end{aligned}$$

However, for string $u_2c_1u_2 \in \mathcal{L}(G \times S)$, which leads to non-secret state 2, we also have

$$\begin{aligned} I_S(u_2c_1u_2) =& P_{\Sigma_{o,a}}(u_2)[\{c_1\}]_{\Sigma_{c,a}} P_{\Sigma_{o,a}}(c_1)[\{c_1\}]_{\Sigma_{c,a}} \\ & P_{\Sigma_{o,a}}(u_2)[\{c_1\}]_{\Sigma_{c,a}} = \{c_1\}c_1\{c_1\}\{c_1\} \end{aligned}$$

That is, the occurrence of secret string $u_1c_1u_1$ does not violate opacity. Note that, although $S(u_1)$ requires to enable both c_1 and c_2 while $S(u_2)$ only requires to enable c_1, these two control decisions are identical for the intruder since the control channel for c_2 is assumed to be secure.

IV. Verification of Opacity using Network Observer

In this section, we discuss how to formally verify opacity for networked supervisory control systems with insecure communication channels. To this end, we first define the *state estimate* from the intruder's point of view.

Let $s \in \mathcal{L}(G \times S)$ be a string generated by the closed-loop system. Then the state estimate of the intruder is defined by

$$\hat{X}(s) = \{x \in X : \exists t \in \mathcal{L}(G \times S) \text{ s.t. } I_S(t) = I_S(s) \wedge \delta(t) = x\}$$

According to Definition 1, it is clear that the system is opaque if and only if for any $s \in \mathcal{L}(G \times S)$, we have $\hat{X}(s) \not\subseteq X_{secret}$. Therefore, how to compute all possible state estimates becomes the key of verifying opacity.

In the event-based observation setting, such state estimate can be computed by constructing the observer automaton; see, e.g., [23]. However, our setting faces the following main challenge: the information in control channels and the information in observation channels are *incomparable* in the sense that knowing the information in one side cannot recover the information in the other side (even if the system model and the functionality of the supervisor are known). For example, when the intruder observes a new event $\sigma \in \Sigma_{o,a}$, it cannot perfectly assert the control decision that will be issued by the supervisor, since it cannot perfectly track the state of the supervisor due to those secure observable events $\Sigma_{o,r}$. On the other hand, when the intruder observes a projected control decision $\gamma \in \Gamma_a$, it also cannot infer what event is received by the supervisor from observation channels, since (i) the control information is projected; and (ii) the supervisor may issue the same control decision upon the occurrences of different events. Therefore, we need an information fusion mechanism for this incomparable information in control channels and observation channels.

To estimate the state of the system, we need to consider the following two situations of the intruder's observation at each instant

- the intruder first observes a new observable event in observation channels and then (immediately) observes a (projected) control decision in control channels; or
- the intruder just observes a (projected) control decision in control channels directly without seeing anything in observation channels.

For the first case, the intruder will know that the last observable event must be in set $\Sigma_{o,a}$ and the projected control decision observed can further help the intruder to eliminate uncertainty of the system. The second case is more complicated and three levels of inference are involved. First, the intruder needs to infer all feasible control decisions based on the projected control decision obtained. Then for each possible control decision, it needs to further infer all possible observations that make the supervisor to issue such a decision. Finally, it will use the inferred observation to further infer the actual strings generated by the system to update the state estimate.

To this end, we define a new structure called the *network observer* that utilizes the above discussed information. Let G be a system and S be a supervisor. Then the network observer is defined as a new DFA

$$Obs(G,S) = (Q, \Sigma_{obs}, f, q_0) \tag{4}$$

where

- $Q \subseteq 2^{X\times Z} \times \{O,C\}$ is the set of states, where O and C are two symbols standing for "observation" and "control", respectively. We denote by Q_O the set of states whose second components are O and by Q_C the set of states whose second components are C.
- $\Sigma_{obs} = \Sigma_{o,a} \cup \Gamma_a$ is the set of events, which is the set of possible observations of the intruder;
- $f : Q \times \Sigma_{obs} \to Q$ is the transition function defined as follows:
  - for any $(q_1,C) \in Q_C, (q_2,O) \in Q_O$ and $\sigma \in \Sigma_{o,a}$, we have $f((q_1,C),\sigma) = (q_2,O)$ if
$$q_2 = \left\{ (x',z') \in X\times Z : \begin{array}{c} \exists (x,z) \in q_1 \text{ s.t.} \\ (x',z') = \eta((x,z),\sigma) \end{array} \right\} \tag{5}$$
  - for any $(q_1,O) \in Q_O, (q_2,C) \in Q_C$ and $\gamma \in \Gamma_a$, we have $f((q_1,O),\gamma) = (q_2,C)$ if
$$q_2 = \left\{ (x',z') \in X\times Z : \begin{array}{c} \exists (x,z) \in q_1, w \in \Sigma_{uo}^* \text{ s.t.} \\ [E_S(z)]_{\Sigma_{c,a}} = \gamma \text{ and} \\ (x',z') = \eta((x,z),w) \end{array} \right\} \tag{6}$$
  - for any $(q_1,C) \in Q_C, (q_2,C) \in Q_C$ and $\gamma \in \Gamma_a$, we have $f((q_1,C),\gamma) = (q_2,C)$ if
$$q_2 = \left\{ (x',z') \in X\times Z : \begin{array}{c} \exists (x,z) \in q_1, \sigma \in \Sigma_{o,r}, w \in \Sigma_{uo}^* \\ \text{s.t. } (x',z') = \eta((x,z),\sigma w) \\ \text{and } [E_S(z')]_{\Sigma_{c,a}} = \gamma \end{array} \right\} \tag{7}$$
- $q_0 = (\{(x,z) \in X\times Z : \exists w \in \Sigma_{uo}^* \text{ s.t. } (x,z) = \eta((x_0,z_0),w)\}, C)$ is the initial state.

Note that, we will only consider reachable states in $Obs(G,S)$.

Intuitively, a C-state represents the intruder's knowledge of the system immediately after observing a control decision in control channels and an O-state represents the intruder's knowledge of the system immediately after observing an event in observation channels. For the sake of clarity, we write the transition function f as f_{CO} if it is from a C-state to a O-state; the same for f_{OZ} and f_{CC}. More specifically, the intuition of each transition function is as follows:

- Function f_{CO} as defined in Equation (5) simply updates each system state and each supervisor state in the state estimate based on the new observable event $\sigma \in \Sigma_{o,a}$. Note that we do not consider any unobservable tile of this observable event in f_{CO} as the supervisor will response to this event immediately before the occurrence of a new event;
- Function f_{OC} as defined in Equation (6) has the following two roles: (i) it first eliminates states in the state estimate whose projected control decisions are not consistent with the observation in control channels; and (ii) then it adds all states that can be reached unobservably in the closed-loop system from states remained in the state estimate.
- Function f_{CC} as defined in Equation (7) captures the case that the intruder observes two projected control decisions consecutively in control channels without seeing an event in observation channels in between. For this case, we know that the supervisor must have received an secure observable event $\sigma \in \Sigma_{o,r}$, which is silent for the intruder, and upon the occurrence of which the supervisor makes a control decision with projection $\gamma \in \Gamma_a$.

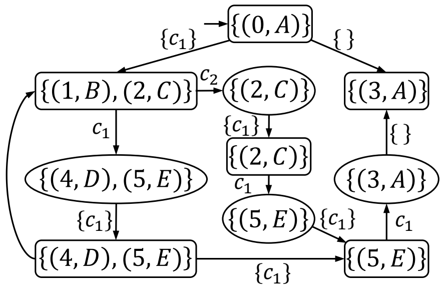


Fig. 3. The network observer $Obs(G,S)$. Rectangular states and oval states denote C-states and O-states, respectively.

Note that, however, there is no transition function f_{OO} as the intruder cannot observe two events in observation channels consecutively without seeing a control decision in between.

Before we formally show the properties of the network observer, we first illustrate this structure by the following example.

Example 2: Let us still consider system G and supervisor S shown in Figures 2(a) and 2(b), respectively. Its network observer $Obs(G,S)$ is shown in Figure 3. For the sake of simplicity, symbols C and O are omitted for each state and we use rectangular states and oval states to denote C-states and O-states, respectively. Initially, the state starts from C-state $\{(0,A)\}$. Note that all observable events that can occur from this state are secure, i.e., the intruder cannot see their occurrences. Therefore, only function f_{CC} is defined at this C-state. If the intruder observes $\{c_1\}$, i.e., c_1 is enabled, then it moves to a new C-state $\{(1,B),(2,C)\}$ in which both state requires to enable c_1. If the intruder observes $\{\ \}$, i.e., c_1 is disabled, then it moves to C-state $\{(3,A)\}$, which contains the only possible state consistent with this control decision. From C-state $\{(1,B),(2,C)\}$, the intruder can first observe event c_1 in observation channels and then observe control decision $\{c_1\}$ in control channels. For this case, the observer will first move to O-state $\{(4,D),(5,E)\}$ and then move to C-state $\{(4,D),(5,E)\}$.

The following result reveals that the proposed network observer indeed tracks the state estimate of the intruder.

Proposition 1: For any $\alpha \in \mathcal{I}_S$, we have $\alpha \in$

$\mathcal{L}(Obs(G,S))$. Moreover, we have

$$q = \left\{(x,z) \in X \times Z : \begin{array}{c} \exists t \in \mathcal{L}(G \times S) \text{ s.t. } I_S(t) = \alpha \\ \text{and } (x,z) = \eta((x_0,z_0),t) \end{array}\right\} \quad (8)$$

where $(q,C) = f(q_0,\alpha)$ is the C-state reached by α in $Obs(G,S)$. (Recall that, by the definition of information-flow, α always ends up with a projected control decision.)

For any state $q \in 2^{X \times Z}$, we denote by $X(q)$ the set of the first component of each element, i.e,

$$X(q) = \{x \in X : \exists z \in Z \text{ s.t. } (x,z) \in q\}$$

Then we also have the following immediate corollary of Proposition 1.

Corollary 1: For any $s \in \mathcal{L}(G \times S)$, we have $\hat{X}(s) = X(f(I_S(s)))$.

Proposition 1 says any information-flow available to the intruder is contained in the network observer, i.e., $\mathcal{I}_S \subseteq \mathcal{L}(Obs(G,S))$. The following result shows that the networked observer will only generate valid information-flow.

Proposition 2: $\mathcal{L}_C(Obs(G,S)) = \mathcal{I}_S$, where

$$\mathcal{L}_C(Obs(G,S)) = \{\alpha \in (\Sigma_{o,a}^{\epsilon}\Gamma_a)^* : f(q_0,\alpha) \in Q_C\}.$$

With Corollary 1 and Proposition 2, we obtain the following main theorem, which shows that we can indeed use the networked observer to verify opacity.

Theorem 1: Let $G \times S$ be a supervisory control system with insecure observation channels $\Sigma_{o,a}$, insecure control channels $\Sigma_{c,a}$ and secret states X_{secret}. Let $Obs(G,S) = (Q, \Sigma_{obs}, f, q_0)$ be its network observer. Then $G \times S$ is opaque if and only if for any C-state $(q,C) \in Q_C$, we have $X(q) \not\subseteq X_{secret}$.

We illustrate Theorem 1 by the following example.

Example 3: Again, we consider system G and supervisor S shown in Figures 2(a) and 2(b), respectively. In Example 1, we have analyzed that secret string $u_1c_1u_1$ does not violate opacity. To check whether or not there exists a secret revealing string, we consider its network observer $Obs(G,S)$ is shown in Figure 3. Clearly, the only C-state that contains a secret state in $\{(1,B),(2,C)\}$ and we have $X(\{(1,B),(2,C)\}) = \{1,2\} \not\subseteq X_{secret} = \{1\}$. Therefore, the networked supervisory control system $G \times S$ is opaque.

Finally, we discuss the complexity of verifying opacity in our setting. The network observer contains $2^{|X|\times|Z|+1}$ states and $|\Sigma_{o,a}| \times 2^{|X|\times|Z|} + 2^{|\Sigma_{c,a}|} \times 2^{|X|\times|Z|}$ transitions. Therefore, the complexity of checking opacity is exponential.

V. Conclusion

In this paper, we proposed a framework for analyzing information-flow security of networked supervisory control systems over multi-channel networks. We provided a model to describe the information leakage in communication networks and adopted the notion of opacity to evaluate the security status. An effective approach was proposed to verify opacity for networked supervisory control systems. Our results provide a generalized framework for the analysis of opacity for networked supervisory control systems by considering both insecure control channels and insecure observation channels simultaneously. In the future, we would like to investigate how to verify other notions of opacity. Also, we are interested in investigating how to synthesize a networked supervisor that is "opaque-by-construction".

References

[1] F. Lin, "Control of networked discrete event systems: dealing with communication delays and losses," *SIAM Journal on Control and Optimization*, vol. 52, no. 2, pp. 1276–1298, 2014.

[2] J. Komenda and F. Lin, "Modular supervisory control of networked discrete-event systems," in *13th International Workshop on Discrete Event Systems*, pp. 85–90, 2016.

[3] S. Shu and F. Lin, "Predictive networked control of discrete event systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4698–4705, 2017.

[4] M. Alves, L. Carvalho, and J. Basilio, "Supervisory control of timed networked discrete event systems," in *56th IEEE Conference on Decision and Control*, pp. 4859–4865, 2017.

[5] F. Lin, "Opacity of discrete event systems and its applications," *Automatica*, vol. 47, no. 3, pp. 496–503, 2011.

[6] A. Saboori and C. Hadjicostis, "Verification of infinite-step opacity and complexity considerations," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1265–1269, 2012.

[7] C. Keroglou and C. Hadjicostis, "Probabilistic system opacity in discrete event systems," *Discrete Event Dyn. Sys.: Theory & Apl.*, pp. 1–26, 2017.

[8] Y. Ji, X. Yin, and S. Lafortune, "Opacity enforcement using non-deterministic publicly-known edit functions," *IEEE Transactions on Automatic Control*, 2019.

[9] X. Yin and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 8, pp. 2140–2154, 2016.

[10] K. Zhang, X. Yin, and M. Zamani, "Opacity of nondeterministic transition systems: A (bi) simulation relation approach," *IEEE Transactions on Automatic Control*, 2019.

[11] J. Chen, M. Ibrahim, and R. Kumar, "Quantification of secrecy in partially observed stochastic discrete event systems," *IEEE Transactions on Automation Sci. Eng.*, vol. 14, no. 1, pp. 185–195, 2017.

[12] X. Yin and S. Lafortune, "A new approach for the verification of infinite-step and K-step opacity using two-way observers," *Automatica*, vol. 80, pp. 162–171, 2017.

[13] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Verification of state-based opacity using Petri nets," *IEEE Transactions on Automatic Control*, vol. 62, no. 6, pp. 2823–2837, 2017.

[14] X. Yin, Z. Li, W. Wang, and S. Li, "Infinite-step opacity and k-step opacity of stochastic discrete-event systems," *Automatica*, vol. 99, pp. 266–274, 2019.

[15] Y. Ji, X. Yin, and S. Lafortune, "Enforcing opacity by insertion functions under multiple energy constraints," *Automatica*, vol. 108, p. 108476, 2019.

[16] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Current-state opacity enforcement in discrete event systems under incomparable observations," *Discrete Event Dynamic Systems*, vol. 28, no. 2, pp. 161–182, 2018.

[17] X. Yin and S. Li, "Synthesis of dynamic masks for infinite-step opacity," *IEEE Trans. Automatic Control*, 2019.

[18] R. Jacob, J.-J. Lesage, and J.-M. Faure, "Overview of discrete event systems opacity: Models, validation, and quantification," *Annual Reviews in Control*, vol. 41, pp. 135–146, 2016.

[19] S. Takai and Y. Oka, "A formula for the supremal controllable and opaque sublanguage arising in supervisory control," *SICE J. Control, Measu. & Syst. Integration*, vol. 1, no. 4, pp. 307–311, 2008.

[20] J. Mullins and M. Yeddes, "Opacity with orwellian observers and intransitive non-interference," in *12th Int. Workshop on Discrete Event Systems*, pp. 344–349, 2014.

[21] L. Hélouët, H. Marchand, and L. Ricker, "Opacity with powerful attackers," in *14th International Workshop on Discrete Event Systems*, pp. 464–471, 2018.

[22] X. Yin and S. Li, "Verification of opacity in networked supervisory control systems with insecure control channels," in *IEEE Conference on Decision and Control*, pp. 4851–4856, 2018.

[23] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Springer, 2nd ed., 2008.