Opacity of Networked Supervisory Control Systems Over Insecure Communication Channels

Shuo Yang, Junyao Hou, *Student Member, IEEE*, Xiang Yin, *Member, IEEE*, and Shaoyuan Li, *Senior Member, IEEE*

***Abstract*—In this article, we investigate security and privacy issues 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 observation 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 the 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. We consider the notions of current-state opacity, K-step opacity, and infinite-step opacity in the networked control setting. New network observers are proposed to estimate the state of the system with two-side incomparable channel information. We show that the opacity verification problems for the networked setting can be effectively solved using the proposed network observers.**

***Index Terms*—Discrete event systems, networked control systems, opacity, security.**

I. Introduction

A. Motivation

SUPERVISORY control theory is a formal approach for controller synthesis of discrete-event systems (DES) with provable correctness guarantees. Since the seminal work of Ramadge and Wonham in the late 1980s, the supervisory control theory has been developed extensively and has been successfully applied to many engineering systems [5]. 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, which receives information from sensors in the plant (modeled as observable events) and sends control decisions to actuators in the plant (modeled as controllable events). In many modern applications, the supervisor and the plant are connected via communication networks, where the supervisor receives sensor readings via observation channels and sends commands via control channels. Control systems with such networked information architectures are referred to as networked control systems (NCSs). Compared with the traditional control architectures, NCSs provide a more flexible way for controlling a system; for example, we can implement the controller in the cloud utilizing more powerful computation resources. Due to the advantages of NCSs, supervisory control of networked discrete-event systems has also drawn considerable attention in the DES literature in the past few years; see, e.g., [1], [11], [14], [16], [19], [21]–[23], and [35].

Although networked control systems have many advantages, they also bring new research challenges. One of the major challenges in NCSs is the information 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 an information leak, which may further reveal some "secret" of the system. Therefore, developing security analysis methodologies is becoming progressively more important for NCSs.

B. Related Works

Networked supervisory control systems have drawn many attention recently in the DES literature; see, e.g., [1], [11], [14], [18], [19], and [21]–[23]. However, most of them focus on how to design a networked supervisor that can handle information delays/losses in the control and observation channels [4], [23], [24], [27], [34], [37]. For example, Lin [14] proposes the notion of network controllability and network observability as necessary and sufficient conditions for the existence of a networked

Manuscript received July 30, 2020; accepted December 18, 2020. Date of publication January 8, 2021; date of current version August 24, 2021. This work was supported by the National Natural Science Foundation of China under Grant 62061136004, Grant 61803259, and Grant 61833012. *(Shuo Yang and Junyao Hou contributed equally to this work.)* Recommended by Associate Editor G. Como. *(Corresponding author: Xiang Yin.)*

The authors are with the Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: xiang-yang@sjtu.edu.cn; houjunyao@sjtu.edu.cn; yinxiang@sjtu.edu.cn; syli@sjtu.edu.cn).

Digital Object Identifier 10.1109/TCNS.2021.3050131

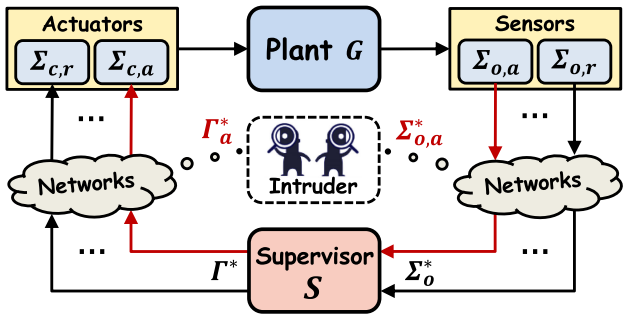

Fig. 1. Networked supervisory control system with insecure multiple channel networks. Sensors and actuators in the plant are modeled as observable events Σ_o and controllable events Σ_c, respectively. Sets $\Sigma_{o,a} \subseteq \Sigma_o$ and $\Sigma_{c,a} \subseteq \Sigma_c$ denote sensors and actuators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

supervisor. To our knowledge, information-flow security issue has not been investigated systematically for supervisory control systems in the context of DES.

In [36], the supervisory control problem over insecure control channels is considered. In particular, the authors consider a "control command eavesdropping actuator enablement attacker." This setting of insecure communication channel is very similar to the setting considered in our work. However, [36] considers the *behavior-preserving* problem of the controlled system, while we investigate the *information-security* problem under insecure channels, which are clearly different.

In the DES literature, the notion of opacity has also been studied very extensively in the past few years; see, e.g., [2], [3], [6], [8]–[10], [13], [20], [25], [26], [29], and [33]. 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 [7] and [17], more general observation models of the intruder are considered. However, their models do not explicitly capture the information-flow in networked supervisory control systems.

The study of opacity of networked DES has recently been reported in [15] and [28]. Specifically, these works assume that the observation of the intruder may be subject to delays or losses, which is similar to Lin's model proposed in [14] for networked DES. However, our work considers the case that the supervisor is implemented in a networked environment in which the communication channels may be insecure. Instead of investigating the effects of delays and losses, our work focuses on investigating information leakage in the feedback channels, which is clearly different from the network opacity problem considered in [15] and [28].

C. Our Contributions

In this article, 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 article is depicted in Fig. 1. Specifically, 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/was at a secret state unambiguously by "listening" those insecure observation and control channels. In particular, we consider three specific types of opacity, namely: 1) current-state opacity; 2) K-step opacity; and 3) infinite-step opacity. In current-state opacity, only the current-state-secret is considered while in K-step and infinite-step opacity, one is also interested in whether or not the visit of a secret state at some instant can be revealed in the future. We then investigate the verification problems of notions of opacity in the networked setting. In particular, we show that the verification of current-state opacity 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. Furthermore, we generalize the network observer to the *two-way network observer* capturing the smoothed information to verify infinite-step and K-step opacity.

Note that our recent work [30] formulates a similar opacity problem in networked supervisory control systems but only for the case of a single insecure control channel. In this article, 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 [30]. In particular, in the single-channel setting, the information leakage in the observation channel is strictly more than that in the control channel. However, 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.

D. Organization

The rest of this article is organized as follows. In Section II, we introduce some necessary preliminaries. Then, we propose the framework of information-flow opacity in multiple channel networks in Section III. In Section IV, we discuss how to verify current-state opacity. Then, we further extend our framework to the infinite-step and K-step opacity in Section V. Finally, Section VI concludes this article. Preliminary and partial versions of some of the results in this article are presented in [31] without proofs. Compared with [31], this article contains i) detailed literature review, explanations, and examples; ii) all technique

proofs; and iii) the simplification of network observer and new results for infinite-step and K-step opacity.

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., [5]. For the sake of simplicity, we write $\delta(x, s)$ as $\delta(s)$ when $x = x_0$. The language generated by G is $\mathcal{L}(G) = \{s \in \Sigma^* : \delta(s)!\}$. Furthermore, the language generated by G from state $x \in X$ is defined by $\mathcal{L}(G, x) = \{s \in \Sigma^* : \delta(x, s)!\}$.

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

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

The natural projection is also extended to $P_{\hat{\Sigma}} : 2^{\Sigma^*} \to 2^{\hat{\Sigma}^*}$ as follows: 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, and "$\dot{\cup}$" denotes the disjoint union. In general, there is no relationship between Σ_c and Σ_o.

A supervisor is a mechanism that disables/enables controllable events dynamically based on its observation. That is, a 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:

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

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 Fig. 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 potentially malicious. The question naturally arises, therefore, whether or not the system is still secure in the presence of such insecure communication networks. In this article, 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. However, 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 multichannel 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 interpret the supervisor as a mapping

$$S : P_{\Sigma_o}(\mathcal{L}(G)) \times \Sigma_c \to \{Disable, Enable\}. \tag{2}$$

This interpretation is usually adopted in the decentralized control problem to distinguish the fusion capability of each controllable event; see, e.g., [12], [32]. 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 $\Sigma_{c,a}$ as it will observe "disable" in each channel of $\Sigma_{c,a}$.

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:

$$I_S(s) = P_{\Sigma_{o,a}}(\sigma_1)[S(\sigma_1)]_{\Sigma_{c,a}} P_{\Sigma_{o,a}}(\sigma_2)[S(\sigma_1\sigma_2)]_{\Sigma_{c,a}} \cdots P_{\Sigma_{o,a}}(\sigma_n)[S(\sigma_1\sigma_2\cdots\sigma_n)]_{\Sigma_{c,a}} \in (\Sigma_{o,a}^{\epsilon}\Gamma_a)^*. \tag{3}$$

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.

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

Remark 1: Hereafter, we assume, without loss of generality, that $\Sigma_{c,a} \neq \emptyset$. Otherwise, the intruder will only observe the projected behavior of the system with respect to event set $\Sigma_{o,a}$, which boils down to the standard opacity analysis problem. Therefore, under this assumption, 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 (3) 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 publicly 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*, which is widely used in the information-flow security analysis. Specifically, we assume that the system has a "secret," which is modeled as a set of secret states $X_S \subset X$. We say that the overall networked control system is *current-state 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 *current-state opaque* with respect to insecure observation channels $\Sigma_{o,a}$, insecure control channels $\Sigma_{c,a}$, and secret states X_S if

$$(\forall s \in \mathcal{L}(G \times S) : \delta(s) \in X_S)$$
$$(\exists t \in \mathcal{L}(G \times S) : \delta(t) \notin X_S)[I_S(s) = I_S(t)]. \tag{4}$$

We illustrate the proposed framework and the notion of opacity by the following example.

Example 1: Let us consider system G shown in Fig. 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 Fig. 2(b) and the closed-loop system $G \times S$ is then shown in Fig. 2(c). The control objective is simply to avoid the occurrence of dashed transitions in Fig. 2(a). We assume that $X_S = \{1\}$, i.e., we do not want the intruder knows for sure that the system is at secret state 1.

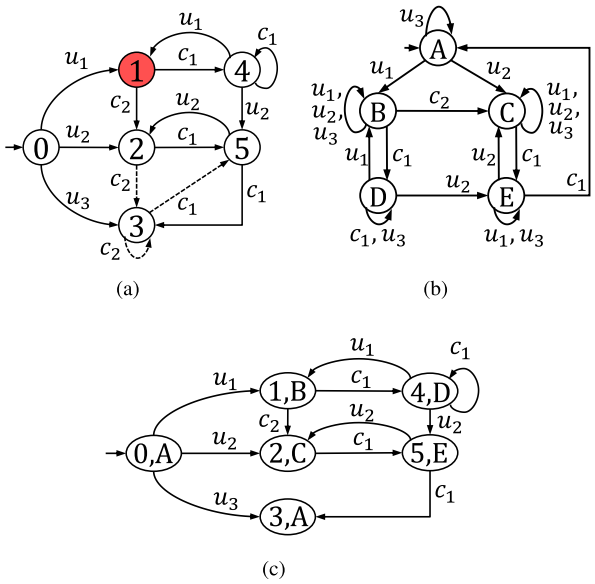

Fig. 2. Supervisory control system (G, S). (a) G. (b) S. (c) $G \times S$.

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

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

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

$$I_S(u_2c_2u_2) = P_{\Sigma_{o,a}}(u_2)[\{c_1\}]_{\Sigma_{c,a}} P_{\Sigma_{o,a}}(c_1)[\{c_1\}]_{\Sigma_{c,a}}$$
$$P_{\Sigma_{o,a}}(u_2)[\{c_1\}]_{\Sigma_{c,a}} = \{c_1\}c_1\{c_1\}\{c_1\}.$$

That is, the occurrence of secret string $u_2c_1u_2$ does not violate current-state 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. However, if $\Sigma_{c,a} = \{c_1, c_2\}$, we know that $G \times S$ is not current-state opaque, since the intruder knows for sure that the system is at secret state 1 when $\{c_1, c_2\}c_1\{c_1\}\{c_1, c_2\}$ is observed.

IV. Verification of Current-State Opacity Using Network Observer

In this section, we discuss how to formally verify current-state opacity for networked supervisory control systems with insecure communication channels.

A. Network Observer

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 current-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, clearly, the system is current-state opaque if and only if for any $s \in \mathcal{L}(G \times S)$, we have $\hat{X}(s) \not\subseteq X_S$. Therefore, how to compute all possible current-state-estimates becomes the key of verifying current-state opacity.

In the event-based observation setting, such current-state-estimate can be computed by constructing the observer automaton; see, e.g., [5]. 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 on one side cannot recover the information on 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.

1) The intruder first observes a new observable event in observation channels and then (immediately) observes a (projected) control decision in control channels; or
2) 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

$$Obs(G, S) = (Q, \Sigma_{obs}, f, q_0) \tag{5}$$

where

1) $Q \subseteq (2^{X \times Z} \setminus \emptyset) \times \{O, C\}$ 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.

2) $\Sigma_{obs} = \Sigma_{o,a} \cup \Gamma_a$ is the set of events, which is the set of possible observations of the intruder.
3) $f : Q \times \Sigma_{obs} \to Q$ is the transition function defined as follows:
   a) for any $(q_1, C) \in Q_C, (q_2, O) \in Q_O$ and $\sigma \in \Sigma_{o,a}$, $f((q_1, C), \sigma) = (q_2, O)$ if

$$q_2 = \left\{ (x,' z') \in X \times Z : \begin{array}{c} \exists (x, z) \in q_1 \text{ s.t.} \\ (x', z') = \eta((x, z), \sigma) \end{array} \right\} \tag{6}$$

   b) for any $(q_1, O) \in Q_O, (q_2, C) \in Q_C$ and $\gamma \in \Gamma_a$, $f((q_1, O), \gamma) = (q_2, C)$ if

$$q_2 = \left\{ (x', z') \in X \times Z : \begin{array}{r} \exists (x, z) \in q_1, w \in \Sigma_{uo}^* \text{ s.t.} \\ [E_S(z)]_{\Sigma_{c,a}} = \gamma \text{ and} \\ (x', z') = \eta((x, z), w) \end{array} \right\} \tag{7}$$

   c) for any $(q_1, C) \in Q_C, (q_2, C) \in Q_C$ and $\gamma \in \Gamma_a$, $f((q_1, C), \gamma) = (q_2, C)$ if

$$q_2 = \left\{ (x', z') \in X \times Z : \begin{array}{r} \exists (x, z) \in q_1, \sigma \in \Sigma_{o,r}, w \in \Sigma_{uo}^* \\ \text{s.t. } (x', z') = \eta((x, z), \sigma w) \\ \text{and } [E_S(z')]_{\Sigma_{c,a}} = \gamma \end{array} \right\} \tag{8}$$

   d) $q_0 = (\{(x, z) \in X \times Z : \exists w \in \Sigma_{uo}^* \text{ s.t. } (x, z) = \eta((x_0, z_0), w)\}, C)$ 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 an O-state; the same for f_{OC} and f_{CC}. More specifically, the intuition of each transition function is as follows:

1) Function f_{CO}, as defined in Equation (6), simply updates each system state and each supervisor state in the state estimate based on the new observable event $\sigma \in \Sigma_{o,a}$. We do not consider any unobservable tail of this observable event in f_{CO} as the supervisor will respond to this event immediately before the occurrence of a new event.
2) Function f_{OC}, as defined in (7), 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.
3) Function f_{CC}, as defined in (14), 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 a 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 $\gamma \in \Gamma_a$.

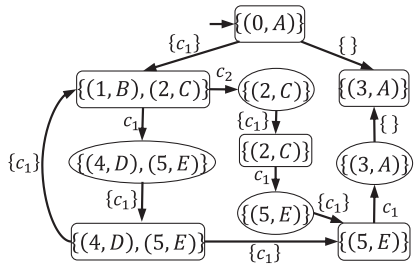

Fig. 3. Network observer $Obs(G, S)$. Rectangular states and oval states denote C-states and O-states, respectively.

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 by the following example.

Example 2: Let us still consider system G and supervisor S shown in Fig. 2(a) and (b), respectively. Its network observer $Obs(G, S)$ is shown in Fig. 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 $\{(4, D), (5, E)\}$.

B. Verification of Current-State Opacity

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, the state reached by α can be characterized by

$$q = \left\{ (x, z) \in X \times Z : \begin{array}{c} \exists t \in \mathcal{L}(G \times S) \text{ s.t. } I_S(t) = \alpha \\ \text{and } (x, z) = \eta((x_0, z_0), t) \end{array} \right\} \tag{9}$$

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.)

Proof: See the Appendix. ■

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.,

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

We have the following result from Proposition 1.

Corollary 1: For any $s \in \mathcal{L}(G \times S)$, the current-state-estimate is the first component of the state reached by $I_S(s)$ in $Obs(G, S)$, i.e., we have $\hat{X}(s) = X(f(I_S(s)))$.

Proposition 1 says that 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 further shows that the network observer will only generate valid information-flow.

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

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

Proof: See the Appendix. ■

With Corollary 1 and Proposition 2, we obtain the following main theorem, which shows that we can indeed use the network 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_S. Let $Obs(G,S) = (Q, \Sigma_{obs}, f, q_0)$ be its network observer. Then, $G \times S$ is current-state opaque if and only if for any C-state $(q,C) \in Q_C$, we have $X(q) \not\subseteq X_S$.

Proof: See the Appendix. ■

We illustrate Theorem 1 by the following example.

Example 3: Again, we consider G and S shown in Fig. 2(a) and (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 $Obs(G,S)$, which is shown in Fig. 3. Clearly, the only C-state that contains a secret state in $\{(1,B),(2,C)\}$ and we have $X(\{(1,B),(2,C)\}) = \{1,2\} \not\subseteq X_S = \{1\}$. Therefore, we conclude that $G \times S$ is current-state opaque.

Remark 3: The complexity of checking current-state opacity using Theorem 1 is exponential in both the number of system/supervisor states and the number of insecure controllable events since the network observer contains at most $2^{|X|\times|Z|+1}$ states and $|\Sigma_{o,a}| \times 2^{|X|\times|Z|} + 2^{|\Sigma_{c,a}|} \times 2^{|X|\times|Z|}$ transitions.

C. Simplified Network Observer

Note that, in the construction of the network observer, states are partitioned as C-states and O-states in order to precisely track different status in the information-flow. However, as shown in Theorem 1, only C-states are considered for the purpose of verification. To simplify our future developments, we can simplify the network observer by hiding O-states. Specifically, we write the projected decision in the control channels and the event in the observation channels as a pair. If a projected control decision $\gamma \in \Gamma_a$ is observed directly without seeing an observable event, then we write this information as (ϵ,γ). If the intruder first observes an event $\sigma \in \Sigma_{o,a}$ followed immediately by a projected control decision $\gamma \in \Gamma_a$, then we write this information as (σ,γ). Then, we can define the simplified network observer, by omitting O-states, as a new DFA

$$\widetilde{Obs}(G \times S) = (\tilde{Q}, \tilde{\Sigma}_{\text{obs}}, \tilde{f}, \tilde{q}_0) \tag{10}$$

where

1) $\tilde{Q} \subset 2^{X\times Z}$ is the set of states.
2) $\tilde{\Sigma}_{\text{obs}} = \Sigma_{o,a}^{\epsilon} \times \Gamma_a$ is the set of events.
3) $\tilde{f} : \tilde{Q} \times \tilde{\Sigma}_{\text{obs}} \to \tilde{Q}$ is the transition function defined by: for any $q \in \tilde{Q}$ and $(\sigma,\gamma) \in \tilde{\Sigma}_{\text{obs}}$, we have

$$(\tilde{f}(q,(\sigma,\gamma)),C) = f((q,C),\sigma\gamma).$$

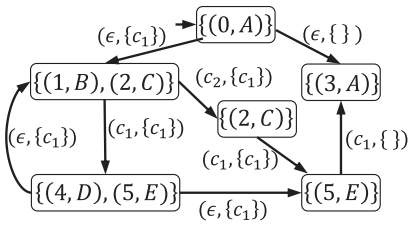

Fig. 4. Simplified network observer $\widetilde{Obs}(G,S)$.

Algorithm 1: Simp-Net-Obs Construction.

$\tilde{q}_0 \leftarrow \{\eta((x_0,z_0),w) \in X \times Z : w \in \Sigma_{uo}^*\}$ and $\tilde{Q} \leftarrow \{\tilde{q}_0\}$
DFS($\widetilde{Obs}$, $\tilde{q}_0$)
return $\widetilde{Obs} = (\tilde{Q}, \tilde{\Sigma}_{\text{obs}}, \tilde{f}, \tilde{q}_0)$
procedure DFS($\widetilde{Obs}$, $\tilde{q}$)
  for all $(\sigma,\gamma) \in \Sigma_{o,a}^{\epsilon} \times \Gamma_a : \tilde{f}(\tilde{q},(\sigma,\gamma))!$ **do**
    $\tilde{q}' \leftarrow \tilde{f}(\tilde{q},(\sigma,\gamma))$
    Add transition $\tilde{q} \xrightarrow{(\sigma,\gamma)} \tilde{q}'$ to $\tilde{f}$
    if $\tilde{q}' \notin \tilde{Q}$ **then**
      $\tilde{Q} \leftarrow \tilde{Q} \cup \{\tilde{q}'\}$
      DFS($\widetilde{Obs}$, $\tilde{q}'$)
    end if
  end for
end procedure

4) $\tilde{q}_0 = \{(x,z) \in X \times Z : \exists w \in \Sigma_{uo}^*$ s.t. $(x,z) = \eta((x_0,z_0),w)\}$ is the initial state, which is the same as the state component of the initial C-state in $Obs(G \times S)$.

The simplified network observer can be constructed by Algorithm 1 via a depth-first-search. The network observer can be constructed analogously, which is omitted due to space constraints.

For example, for system G and supervisor S shown in Fig. 2(a) and (b), respectively, the simplified network observer $\widetilde{Obs}(G \times S)$ is shown in Fig. 4. Compared with $Obs(G,S)$ in Fig. 3, all O-states are merged with C-states.

V. Verification of Infinite and K-Step Opacity Using Two-Way Network Observer

A. Smoothed Information and Reversed Observer

Current-state opacity only guarantees that the intruder does not know that the system is currently at a secret state. However, it may use future observation to infer that the system was at a secret state for some previous instant; such a secret requirement can be captured by infinite-step opacity and K-step opacity. In this section, we investigate how to verify these two notions in the networked setting, which are formally defined as follows.

Definition 2: Given networked supervisory control system $G \times S$ with $\Sigma_{o,a}$, $\Sigma_{c,a}$ and X_S, we say that $G \times S$ is

1) infinite-step opaque if

$$(\forall st \in \mathcal{L}(G \times S) : \delta(s) \in X_S)$$

$$[\exists s't' \in \mathcal{L}(G \times S) : \delta(s') \notin X_S]$$
$$[I_S(s) = I_S(s') \wedge I_S(st) = I_S(s't')]. \quad (11)$$

2) K-step opaque if

$$(\forall st \in \mathcal{L}(G \times S) : \delta(s) \in X_S \wedge |P_{\Sigma_o}(t)| \leq K)$$
$$[\exists s't' \in \mathcal{L}(G \times S) : \delta(s') \notin X_S]$$
$$[I_S(s) = I_S(s') \wedge I_S(st) = I_S(s't')]. \quad (12)$$

Let $st \in \mathcal{L}(G \times S)$ be a string in the controlled system and let $P_{\Sigma_o}(st) = \sigma_1 \dots \sigma_n \sigma_{n+1} \dots \sigma_{n+m}$, where $|P_{\Sigma_o}(s)| = n$ and $|P_{\Sigma_o}(t)| = m$. Then, the information-flow $I_S(st)$ can be written as $I_S(st) = \imath_s \imath_t$, where

$$\imath_s = P_{\Sigma_{o,a}}(\sigma_1)[S(\sigma_1)]_{\Sigma_{c,a}} \cdots P_{\Sigma_{o,a}}(\sigma_n)[S(\sigma_1 \cdots \sigma_n)]_{\Sigma_{c,a}}$$
$$\imath_t = P_{\Sigma_{o,a}}(\sigma_{n+1})[S(\sigma_1 \cdots \sigma_{n+1})]_{\Sigma_{c,a}} \cdots P_{\Sigma_{o,a}}(\sigma_{n+m})[S(\sigma_1 \cdots \sigma_{n+m})]_{\Sigma_{c,a}}.$$

To check infinite-step and K-step opacity, the main difficulty is to handle the smoothed information $\imath_t$. To this end, we utilize the *reversed dynamic* of the system as follows. We read the smoothed information $\imath_t$ in a *reversed* manner so that a projected event is read first before reading its predecessor observable event. In this way, we are able to track where the system may start from generating this smoothed information. This idea is formalized by the *reversed network observer*, which is a new DFA

$$\widetilde{Obs}_R(G \times S) = (\tilde{Q}_R, \tilde{\Sigma}_{obs,R}, \tilde{f}_R, \tilde{q}_{0,R}) \quad (13)$$

where

1) $\tilde{Q}_R \subset 2^{X \times Z}$ is the set of states.
2) $\tilde{\Sigma}_{obs,R} \subseteq \Gamma_a \times \Sigma^{\epsilon}_{o,a}$ is the set of events.
3) $\tilde{f}_R : \tilde{Q}_R \times \tilde{\Sigma}_{obs,R} \to \tilde{Q}_R$ is the transition function defined as follows: for any $q_1, q_2 \in \tilde{Q}_R$, and $(\gamma, \sigma) \in \tilde{\Sigma}_{obs,R}$, we have $\tilde{f}_R(q_1, (\gamma, \sigma)) = q_2$ if

$$q_2 = \left\{ (x', z') \in X \times Z : \begin{array}{c} \exists (x, z) \in q_1, w \in \Sigma^*_{uo}, \sigma' \in \Sigma_o \\ \text{s.t. } (x, z) = \eta((x', z'), \sigma' w) \\ \text{and } P_{\Sigma_{o,a}}(\sigma') = \sigma \\ \text{and } [E_S(z)]_{\Sigma_{c,a}} = \gamma \end{array} \right\} \quad (14)$$

4) $\tilde{q}_{0,R} = X \times Z$ is the initial state.

Let $(x, z) \in X \times Z$ be a pair of plant state and supervisor state and $s \in \mathcal{L}(G \times S, (x, z))$ be a string feasible from (x, z) in the controlled string such that $P_{\Sigma_o}(s) = \sigma_1 \sigma_2 \cdots \sigma_n$. Then, we define the information-flow of s from (x, z) by

$$I_S(s, (x, z)) = P_{\Sigma_{o,a}}(\sigma_1)[\xi(z, \sigma_1)]_{\Sigma_{c,a}} P_{\Sigma_{o,a}}(\sigma_2)[\xi(z, \sigma_1\sigma_2)]_{\Sigma_{c,a}} \cdots P_{\Sigma_{o,a}}(\sigma_n)[\xi(z, \sigma_1\sigma_2 \cdots \sigma_n)]_{\Sigma_{c,a}} \in (\Sigma^{\epsilon}_{o,a}\Gamma_a)^*.$$

Clearly, we have $I_S(s, (x, z)) = I_S(s)$ when $(x, z) = (x_0, z_0)$. Then, the following result reveals that the proposed reversed network observer indeed computes all possible state pairs that are consistent with the information-flow in a reversed manner.

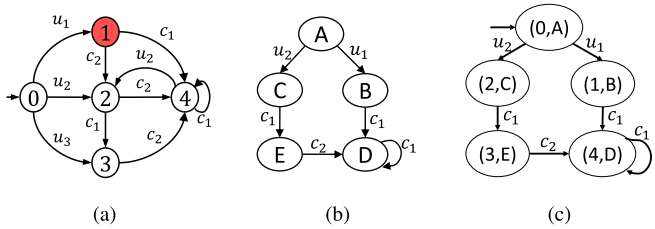

Fig. 5. For system G_1, we have $\Sigma_c = \{c_1, c_2\}$ and $\Sigma_o = \Sigma$, $\Sigma_{c,a} = \{c_1\}$ and $\Sigma_{o,a} = \{c_1, c_2\}$. (a) G_1. (b) S_1. (c) $G_1 \times S_1$.

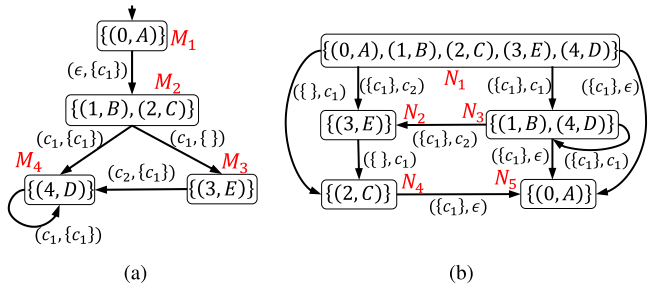

Fig. 6. Example of the (simplified) network observer and the reversed network observer for G_1 and S_1. (a) $\widetilde{Obs}(G_1 \times S_1)$. (b) $\widetilde{Obs}_R(G_1 \times S_1)$.

Proposition 3: For any string $(\gamma_1, \sigma_1) \dots (\gamma_n, \sigma_n) \in \mathcal{L}(\widetilde{Obs}_R(G \times S))$, we have

$$\tilde{f}_R(\tilde{q}_{0,R}, (\gamma_1, \sigma_1) \dots (\gamma_n, \sigma_n)) = \left\{ (x, z) \in X \times Z : \begin{array}{c} \exists s \in \mathcal{L}(G \times S, (x, z)) \\ \text{s.t. } I_S(s, (x, z)) = \sigma_n \gamma_n \dots \sigma_1 \gamma_1 \end{array} \right\}. \quad (15)$$

Proof: See the Appendix.

Based on Proposition 3, we can further obtain the following result, which shows that the reversed network observer can be used together with the network observer to compute the smoothed-state-estimate of the system.

Theorem 2: For any information-flow $\sigma_1\gamma_1 \dots \sigma_n\gamma_n \in \mathcal{I}_S$ and any integer $0 \leq k \leq n$, we have

$$\tilde{f}((\sigma_1, \gamma_1) \cdots (\sigma_k, \gamma_k)) \cap \tilde{f}_R((\gamma_n, \sigma_n) \cdots (\gamma_{k+1}, \sigma_{k+1})) = \left\{ (x, z) \in X \times Z : \begin{array}{c} \exists st \in \mathcal{L}(G \times S)) \text{ s.t. } \eta(s) = (x, z) \\ \text{and } I_S(s) = \sigma_1\gamma_1 \dots \sigma_k\gamma_k \\ \text{and } I_S(st) = \sigma_1\gamma_1 \dots \sigma_n\gamma_n \end{array} \right\}. \quad (16)$$

Proof: See the Appendix.

The reversed network observer and its properties are illustrated by the following example.

Example 4: Let us consider system G_1 and supervisor S_1 shown in Fig. 5 (a) and (b), respectively. We assume that $\Sigma_c = \{c_1, c_2\}$, $\Sigma_o = \Sigma$, $\Sigma_{c,a} = \{c_1\}$, and $\Sigma_{o,a} = \{c_1, c_2\}$. Then, the closed-loop system $G_1 \times S_1$ is shown in Fig. 5(c). The (simplified) network observer and the reversed network observer of $G_1 \times S_1$ are shown in Fig. 5(a) and (b), respectively. For the reversed network observer $\widetilde{Obs}_R(G_1 \times S_1)$, its initial state is all possible states in $G_1 \times S_1$. If event $(\{c_1\}, c_2)$ occurs,

then we move to state $\{(3, E)\}$ since for state $(4, D) \in \tilde{q}_{0,R}$, $[E_S(D)]_{\Sigma_{c,a}} = \{c_1\}$ and $\eta((3, E), c_2) = (4, D)$.

B. Two-Way Network Observer

Theorem 2 essentially says that all possible smoothed-state-estimates can be captured by combining states in the network observer and states in the reversed network observer. In order to verify infinite-step opacity and K-step opacity, we compose $\widetilde{Obs}(G \times S)$ and $\widetilde{Obs}_R(G \times S)$ together, where the former tracks all states that are consistent with the current information-flow while the latter tracks all states that are consistent with the future information-flow. We call the composed structure the *two-way network observer*, which is a DFA

$$Obs_{TW}(G \times S) = (Q_{TW}, \Sigma_{TW}, f_{TW}, q_{TW,0}) \tag{17}$$

where

1) $Q_{TW} \subseteq \tilde{Q} \times \tilde{Q}_R$ is the set of states.
2) $\Sigma_{TW} = (\tilde{\Sigma}_{obs} \times \{\epsilon\}) \cup (\{\epsilon\} \times \tilde{\Sigma}_{obs,R})$ is the set of events.
3) $f_{TW} : Q_{TW} \times \Sigma_{TW} \to Q_{TW}$ is the transition function defined as follows: for any state $(q_1, q_2) \in Q_{TW}$ and event $\sigma \in \tilde{\Sigma}_{obs} \cup \tilde{\Sigma}_{obs,R}$, the following transitions are defined whenever they are feasible

$$f_{TW}((q_1, q_2), (\sigma, \epsilon)) = (\tilde{f}(q_1, \sigma), q_2) \tag{18}$$

$$f_{TW}((q_1, q_2), (\epsilon, \sigma)) = (q_1, \tilde{f}_R(q_2, \sigma)) \tag{19}$$

4) $q_{TW,0} = (\tilde{q}_0, \tilde{q}_{0,R})$ is the initial state.

Still, we only consider the accessible part of $Obs_{TW}(G \times S)$.

The two-way network observer has the following properties. First, we show that, if the intersection of the first and the second components of a state in Q_{TW} is non-empty, then information-flows leading to each component can be "connected": one as the current information-flow and the other as the future information-flow.

Proposition 4: For any state $(q_1, q_2) \in Q_{TW}$ such that $q_1 \cap q_2 \neq \emptyset$, there exist strings $s = (\sigma_1, \gamma_1) \dots (\sigma_n, \gamma_n) \in \mathcal{L}(\widetilde{Obs}(G \times S))$ and $t = (\gamma'_1, \sigma'_1) \dots (\gamma'_m, \sigma'_m) \in \mathcal{L}(\widetilde{Obs}_R(G \times S))$ such that $\tilde{f}(\tilde{q}_0, s) = q_1, \tilde{f}_R(\tilde{q}_{0,R}, t) = q_2$ and

$$(\sigma_1, \gamma_1) \dots (\sigma_n, \gamma_n)(\sigma'_m, \gamma'_m) \dots (\sigma'_1, \gamma'_1) \in \mathcal{L}(\widetilde{Obs}(G \times S)).$$

Proof: See the Appendix. ■

Example 5: Consider again G_1 in Fig. 5(a) and S_1 in Fig. 5(b). The two-way network observer $Obs_{TW}(G_1 \times S_1)$ is shown in Fig. 7 , which essentially composes $\widetilde{Obs}(G \times S)$ and $\widetilde{Obs}_R(G \times S)$ asynchronously. For the sake of simplicity, each state in $Obs_{TW}(G_1 \times S_1)$ is renamed according to the state names in Fig. 6(a) and (b). For instance, state (M_2, N_4) represents state $(\{(1, B), (2, C)\}, \{(2, C)\})$, which can be reached by string $((\epsilon, \{c_1\}), \epsilon)(\epsilon, (\{\}, c_1))$. Since $\{(1, B), (2, C)\} \cap \{(2, C)\} \neq \emptyset$, by Proposition 4, we can find string $(\epsilon, \{c_1\})(c_1, \{\}) \in \mathcal{L}(\widetilde{Obs}(G \times S))$.

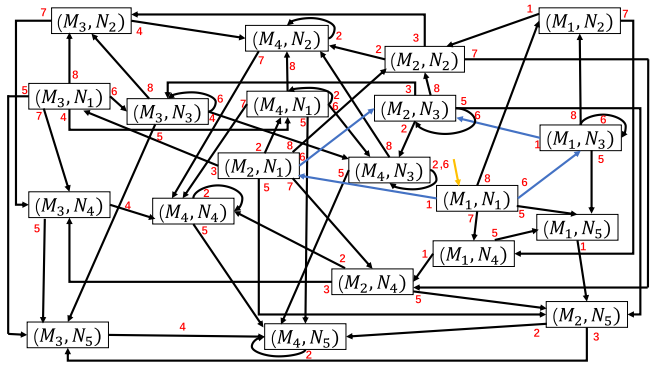

Fig. 7. Two-way network observer $Obs_{TW}(G_1 \times S_1)$ for the networked system $G_1 \times S_1$ in Fig. 5(c) and events in $\{1, 2, 3, 4, 5, 6, 7, 8\}$ represent events in $\{((\epsilon, \{c_1\}), \epsilon), ((c_1, \{c_1\}), \epsilon), ((c_1, \{\}), \epsilon), ((c_2, \{c_1\}), \epsilon), (\epsilon, (\{c_1\}, \epsilon)), (\epsilon, (\{c_1\}, c_1)), (\epsilon, (\{\}, c_1)), (\epsilon, (\{c_1\}, c_2))\}$, respectively.

C. Verification of Infinite-Step and K-Step Opacity

The following two theorems show how the two-way network observer can be used to verify infinite-step opacity and K-step opacity. First, we show how to check infinite-step opacity.

Theorem 3: 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_S. Let $Obs_{TW}(G \times S) = (Q_{TW}, E_{TW}, f_{TW}, q_{TW,0})$ be its two-way network observer. Then, $G \times S$ is infinite-step opaque if and only if

$$\forall (q_1, q_2) \in Q_{TW} : q_1 \cap q_2 \not\subseteq X_S \text{ or } q_1 \cap q_2 = \emptyset. \tag{20}$$

Proof: See the Appendix.

For any string $s \in \mathcal{L}(Obs_{TW}(G \times S))$, we denote by $|\tau(s)|$ the number of events in s in the form of (ϵ, σ). Then, we have the following theorem for the verification of K-step opacity. Its proof is omitted as it is similar to the case of infinite-step opacity. Specifically, the main difference is that we are only interested in searching states in the two-way network observer that can be reached within K steps counted according to the second component of the transitions.

Theorem 4: Let $G \times S$ be a supervisory control system with $\Sigma_{o,a}, \Sigma_{c,a}$, and X_S. Let $Obs_{TW}(G \times S)$ be its two-way network observer. Then, $G \times S$ is K-step opaque, if and only if, for any string $s \in \mathcal{L}(Obs_{TW}(G \times S))$ such that $f_{TW}(q_{TW,0}, s) = (q_1, q_2)$, we have

$$[q_1 \cap q_2 \subseteq X_S \wedge q_1 \cap q_2 \neq \emptyset] \Rightarrow |\tau(s)| > K. \tag{21}$$

We illustrate how to verify infinite-step opacity and K-step opacity by the following example.

Example 6: Consider again the system G_1 in Fig. 5(a) and the supervisory S_1 in Fig. 5(b), then we verify infinite-step opacity by using two-way network observer shown in Fig. 7. We notice that state (M_2, N_3), which denotes state $(\{(1, B), (2, C)\}, \{(1, B), (4, D)\})$, is reached by string $((\epsilon, \{c_1\}), \epsilon)(\epsilon, (\{c_1\}, c_1))$ or string $(\epsilon, (\{c_1\}, c_1))((\epsilon, \{c_1\}), \epsilon)$ and these two strings are highlighted by blue transition lines in Fig. 7. Since $\{(1, B), (2, C)\} \cap \{(1, B), (4, D)\} =$

$(1, B) \subseteq X_S = \{(1, B)\}$. Therefore, we know that the networked supervisory control system $G_1 \times S_1$ is not infinite-step opaque. Specifically, when the intruder observes $\{c_1\}c_1\{c_1\}$, it knows with certainty that $G_1 \times S_1$ was in a secret state 1-step earlier. In fact, $G_1 \times S_1$ is not 1-step opaque as $|\tau((\epsilon, (\{c_1\}, c_1))((\epsilon, \{c_1\}), \epsilon))| = 1$.

Remark 4: Note that the reversed network observer is neither the reverse of the network observer nor the network observer of the reversed plant model. Specifically, the intruder observes both communication channels and control channels; this issue cannot be captured by the two-way observer in [29]. Therefore, although our approach is motivated by the two-way observer in [29], the original construction cannot be directly adopted for our networked setting. We need the proposed new construction for the reversed observer to track where the system may start from to generate the smoothed information in the multichannel network setting.

Remark 5: Let us analyze the complexity of the verification of infinite-step opacity and K-step opacity in networked system. The network two-way observer contains at most $2^{|X|\times|Z|} \times 2^{|X|\times|Z|}$ states and $|\Sigma_{o,a}| \times 2^{|\Sigma_{c,a}|} \times 2^{|X|\times|Z|} \times 2^{|X|\times|Z|}$ transitions. To check infinite-step opacity, it suffices to preform a reachability search within the two-way observer, and to check K-step opacity, it suffices to preform a K-step (in terms of the moves in the second component) depth-first search within the two-way observer. Therefore, both properties in the networked setting can be checked in exponential-time.

VI. Conclusion

In this article, we proposed a framework for analyzing information-flow security of networked supervisory control systems over multichannel networks. We provided a model to describe the information leakage in communication networks and adopted three opacity notions (current-state, infinite-step, and K-step opacity) to evaluate the security status. Effective approaches were proposed to verify current-state opacity, infinite-step opacity, and K-step opacity for networked supervisory control systems. Our results provided a generalized framework for the analysis of opacity for networked supervisory control systems by considering both insecure control channels and insecure observation channels simultaneously. Note that this work focused on the verification problem, i.e., to check whether or not a given networked supervisory control system is opaque. For future work, we are also interested in investigating how to synthesize a networked supervisor that is "opaque-by-construction."

Appendix
Proof of Proposition 1

Proof: We prove by induction on the length of α.

Induction Basis: When $|\alpha| = 0$, clearly $\alpha = \epsilon \in \mathcal{L}(Obs(G, S))$ and (q, C) is the initial state in $Obs(G, S)$. Therefore, we have

$$\begin{aligned} q &= \{(x, z) \in X \times Z : \exists t \in \Sigma_{uo}^* \text{ s.t. } (x, z) = \eta((x_0, z_0), t)\} \\ &= \left\{(x, z) \in X \times Z : \begin{array}{c} \exists t \in \mathcal{L}(G \times S) \text{ s.t. } P_{\Sigma_o}(t) = \epsilon \\ \text{and } (x, z) = \eta((x_0, z_0), t) \end{array}\right\} \\ &= \left\{(x, z) \in X \times Z : \begin{array}{c} \exists t \in \mathcal{L}(G \times S) \text{ s.t. } I_S(t) = \alpha \\ \text{and } (x, z) = \eta((x_0, z_0), t) \end{array}\right\}. \end{aligned}$$

Note that, the last equality comes from the fact that, for any $t \in \mathcal{L}(G \times S)$, $I_S(t) = \epsilon$ if and only if $P_{\Sigma_o}(t) = \epsilon$. Therefore, the induction basis holds.

Induction Hypothesis: Assume that $\alpha \in \mathcal{L}(Obs(G, S))$ and (9) holds for $|\alpha| = k$. To proceed the induction step, we need to consider the following two cases.

Case 1: The next observation of the intruder is a projected control decision $\gamma \in \Gamma_a$.

For the sake of clarity, we write $f(q_0, \alpha) = (q_k, C)$ and $f(q_0, \alpha\gamma) = (q_{k+1}, C)$. That is, the induction hypothesis is

$$q_k = \left\{(x, z) \in X \times Z : \begin{array}{c} \exists t \in \mathcal{L}(G \times S) \text{ s.t. } I_S(t) = \alpha \\ \text{and } (x, z) = \eta((x_0, z_0), t) \end{array}\right\}. \quad (22)$$

First, we show that information-flow $\alpha\gamma$ is well defined in $Obs(G, S)$. Since $\alpha\gamma \in \mathcal{I}_S$, we know that there exists $s \in \mathcal{L}(G \times S)$ such that $I_S(s) = \alpha\gamma$. We write s in the form of $s = s'\sigma w$, where $\sigma \in \Sigma_o$ and $w \in \Sigma_{uo}^*$. Furthermore, we have $\sigma \in \Sigma_{o,r}$; otherwise, σ should be observed before γ. Then, we have $I_S(s') = \alpha$ and $[E_S(\xi(s'\sigma w))]_{\Sigma_{c,a}} = \gamma$. By the induction hypothesis, we have that $\eta(s') \in q_k$. Therefore, by the definition of f, we know that $f(q_k, \gamma)!$, i.e., $\alpha\gamma \in \mathcal{L}(Obs(G, S))$.

Then, by the definition of f, we have $f_{CC}((q_k, C), \gamma) = (q_{k+1}, C)$, i.e.,

$$\begin{aligned} q_{k+1} &= \left\{(x', z') \in X \times Z : \text{s.t. } \begin{array}{c} \exists (x, z) \in q_k, \sigma \in \Sigma_{o,r}, w \in \Sigma_{uo}^* \\ (x', z') = \eta((x, z), \sigma w) \\ \text{and } [E_S(z')]_{\Sigma_{c,a}} = \gamma \end{array}\right\} \\ &= \left\{(x, z) \in X \times Z : \text{s.t. } \begin{array}{c} \exists t\sigma w \in \mathcal{L}(G \times S), \sigma \in \Sigma_{o,r}, w \in \Sigma_{uo}^* \\ (x, z) = \eta((x_0, z_0), t\sigma w), \\ I_S(t) = \alpha \text{ and } [E_S(z)]_{\Sigma_{c,a}} = \gamma \end{array}\right\} \\ &= \left\{(x, z) \in X \times Z : \begin{array}{c} \exists s \in \mathcal{L}(G \times S) \text{ s.t. } I_S(s) = \alpha\gamma \\ \text{and } (x, z) = \eta((x_0, z_0), t) \end{array}\right\}. \end{aligned}$$

Case 2: After α, the intruder first observes an observable event $\sigma \in \Sigma_{o,a}$ and then observes the next projected control decision $\gamma \in \Gamma_a$.

For this case, we write $f(q_0, \alpha) = (q_k, C)$, $f(q_0, \alpha\sigma) = (q_{k+1}, O)$, and $f(q_0, \alpha\sigma\gamma) = (q_{k+2}, C)$. The induction hypothesis is still

$$q_k = \left\{(x, z) \in X \times Z : \begin{array}{c} \exists t \in \mathcal{L}(G \times S) \text{ s.t. } I_S(t) = \alpha \\ \text{and } (x, z) = \eta((x_0, z_0), t) \end{array}\right\}. \quad (23)$$

To proceed the induction step, first, we show that information-flow $\alpha\sigma\gamma$ is well-defined in $Obs(G, S)$. Since $\alpha\sigma\gamma \in \mathcal{I}_S$, we know that there exists $s \in \mathcal{L}(G \times S)$ such that $I_S(s) = \alpha\sigma\gamma$. We can write s in the form of $s = s'\sigma w$, where $w \in \Sigma_{uo}^*$ and $I_S(s') = \alpha$. By the induction hypothesis, we have that $\eta(s') \in q_k$. Moreover, since $\sigma \in \Sigma_{o,a}$, by (6), we have $f_{CO}(q_k, \sigma)!$ and $\eta(s'\sigma) \in q_{k+1}$. Then, since $[E_S(\xi(s'\sigma))]_{\Sigma_{c,a}} = [E_S(\xi(s'\sigma w))]_{\Sigma_{c,a}} = \gamma$, by (7), we have $\eta(s'\sigma w) \in q_{k+2}$. That is, $\alpha\sigma\gamma \in \mathcal{L}(Obs(G, S))$.

Then, by the definition of f, we have $f_{CO}((q_k, C), \sigma) = (q_{k+1}, O)$ and $f_{OC}((q_{k+1}, C), \gamma) = (q_{k+2}, C)$. Therefore,

$$\begin{aligned} q_{k+2} &= \left\{ \begin{matrix} (x', z') \\ \in X \times Z \end{matrix} : \begin{matrix} \exists (x, z) \in q_{k+1}, w \in \Sigma_{uo}^* \text{ s.t.} \\ [E_S(z)]_{\Sigma_{c,a}} = \gamma \text{ and} \\ (x', z') = \eta((x, z), w) \end{matrix} \right\} \\ &= \left\{ \begin{matrix} (x', z') \\ \in X \times Z \end{matrix} : \begin{matrix} \exists (x, z) \in q_k, w \in \Sigma_{uo}^* \text{ s.t.} \\ [E_S(\xi(z, \sigma))]_{\Sigma_{c,a}} = \gamma \text{ and} \\ (x', z') = \eta((x, z), \sigma w) \end{matrix} \right\} \\ &= \left\{ \begin{matrix} (x, z) \\ \in X \times Z \end{matrix} : \begin{matrix} \exists t \in \mathcal{L}(G \times S), w \in \Sigma_{uo}^* \\ \text{:s.t. } I_S(t) = \alpha, (x, z) = \eta(t\sigma w) \\ \text{and } [E_S(\xi(t\sigma))]_{\Sigma_{c,a}} = \gamma \end{matrix} \right\} \\ &= \left\{ \begin{matrix} (x, z) \\ \in X \times Z \end{matrix} : \begin{matrix} \exists s \in \mathcal{L}(G \times S) \text{ s.t. } I_S(s) = \alpha\sigma\gamma \\ \text{and } (x, z) = \eta((x_0, z_0), s) \end{matrix} \right\}. \end{aligned}$$

Since the induction step holds for both cases, we complete the inductive proof. ■

Proof of Proposition 2

Proof: By Proposition 1, we already have $\mathcal{I}_S \subseteq \mathcal{L}_C(Obs(G, S))$. It remains to show that $\mathcal{L}_C(Obs(G, S)) \subseteq \mathcal{I}_S$. Let us consider an arbitrary string $\alpha \in \mathcal{L}_C(Obs(G, S))$. Next, we show that $\alpha \in \mathcal{I}_S$ by induction on the length of α. When $\alpha = \epsilon$, we know immediately that $\alpha \in \mathcal{I}_S$. Now we assume that, $\alpha \in \mathcal{I}_S$ when $|\alpha| = k$ and we consider the following two cases:

Case 1: $\alpha\gamma \in \mathcal{L}_C(Obs(G, S))$, where $\gamma \in \Gamma_a$. By the induction hypothesis, we know that there exists a string $s \in \mathcal{L}(G \times S)$ such that $I_S(s) = \alpha$. Since $\alpha\gamma \in \mathcal{L}(Obs(G, S))$, by the definition of f_{CC}, we know that there exist $\sigma \in \Sigma_{o,r}, w \in \Sigma_{uo}^*$ such that $[S(t\sigma w)]_{\Sigma_{c,a}} = \gamma$, where t is a string such that $I_S(t) = I_S(s) = \alpha$. Therefore, we know that $I_S(t\sigma w) = \alpha P_{\Sigma_{o,a}}(\sigma)[S(t\sigma w)]_{\Sigma_{c,a}} = \alpha\gamma \in \mathcal{I}_S$.

Case 2: $\alpha\sigma\gamma \in \mathcal{L}_C(Obs(G, S))$, where $\sigma \in \Sigma_{o,a}$ and $\gamma \in \Gamma_a$. By the induction hypothesis, we still know that there exists a string $s \in \mathcal{L}(G \times S)$ such that $I_S(s) = \alpha$. Since $\alpha\sigma sw \in \mathcal{L}(Obs(G, S))$, by the definition of f_{CO} and f_{OC}, we know that there exists $w \in \Sigma_{uo}^*$ such that $[S(t\sigma w)]_{\Sigma_{c,a}} = \gamma$, where t is a string such that $I_S(t) = I_S(s) = \alpha$. Therefore, $I_S(t\sigma w) = \alpha P_{\Sigma_{o,a}}(\sigma)[S(t\sigma w)]_{\Sigma_{c,a}} = \alpha\sigma\gamma \in \mathcal{I}_S$.

Since the induction step holds for both cases, we complete the proof. ■

Proof of Theorem 1

Proof: According to Corollary 1, $G \times S$ is current-state opaque if and only if $\forall s \in \mathcal{L}(G \times S) : X(f(I_S(s))) = \hat{X}(s) \not\subseteq X_S$. Based on Proposition 2, this is further equivalent to $\forall (q, C) \in Q_C : X(q) \not\subseteq X_S$ holds as any C-state is reached and only reached by strings in $\mathcal{I}_S$. This ends the proof.

Proof of Proposition 3

Proof: We prove it by induction on n. When $n = 0$, it is clear that $\tilde{f}_R(\tilde{q}_{0,R}, \epsilon) = X \times Z$ since $I_S(\epsilon, (x, z)) = \epsilon$ for any $(x, z) \in X \times Z$. Now, let us assume that Proposition 3 holds for $n = k$. For the sake of simplicity, we denote $\tilde{f}_R(\tilde{q}_{0,R}, (\gamma_1, \sigma_1) \dots (\gamma_k, \sigma_k))$ by S_k. Then, for $n = k + 1$, we have that

$$\begin{aligned} &\tilde{f}_R(\tilde{q}_{0,R}, (\gamma_1, \sigma_1) \dots (\gamma_k, \sigma_k)(\gamma_{k+1}, \sigma_{k+1})) \\ &= \tilde{f}_R(S_k, (\gamma_{k+1}, \sigma_{k+1})) \\ &= \left\{ \begin{matrix} (x, z) \\ \in X \times Z \end{matrix} : \begin{matrix} \exists (x', z') \in S_k, w \in \Sigma_{uo}^*, \sigma' \in \Sigma_o \text{ s.t.} \\ (x', z') = \eta((x, z), \sigma' w), P_{\Sigma_{o,a}}(\sigma') = \sigma_{k+1} \\ \text{and } [E_S(z')]_{\Sigma_{c,a}} = \gamma_{k+1} \end{matrix} \right\} \\ &= \left\{ (x, z) \in X \times Z : \begin{matrix} \exists \sigma' w s \in \mathcal{L}(G \times S, (x, z)) \\ \text{s.t. } I_S(s, (x', z')) = \sigma_k\gamma_k \dots \sigma_1\gamma_1 \\ \text{and } (x', z') = \eta((x, z), \sigma' w) \\ \text{and } P_{\Sigma_{o,a}}(\sigma') = \sigma_{k+1} \\ \text{and } [E_S(z')]_{\Sigma_{c,a}} = \gamma_{k+1} \end{matrix} \right\} \\ &= \left\{ (x, z) \in X \times Z : \begin{matrix} \exists \sigma' w s \in \mathcal{L}(G \times S, (x, z)) \\ \text{s.t. } I_S(\sigma' w s, (x, z)) \\ = \sigma_{k+1}\gamma_{k+1}\sigma_k\gamma_k \dots \sigma_1\gamma_1 \end{matrix} \right\}. \end{aligned}$$

Note that the third equivalence follows from the induction hypothesis, and the last equation holds since $I_S(\sigma' w s, (x, z)) = I_S(\sigma' w, (x, z)) I_S(s, (x', z')) = P_{\Sigma_{o,a}}(\sigma')[\xi(z, \sigma')]_{\Sigma_{c,a}} \sigma_k\gamma_k \dots \sigma_1\gamma_1 = \sigma_{k+1}\gamma_{k+1} \dots \sigma_1\gamma_1$. ■

Proof of Proposition 4

Proof: By the construction of $Obs_{TW}(G \times S)$, there exist strings $s \in \mathcal{L}(\widetilde{Obs}(G \times S))$ and $t \in \mathcal{L}(\widetilde{Obs}_R(G \times S))$ such that $\tilde{f}(\tilde{q}_0, s) = q_1$ and $\tilde{f}_R(\tilde{q}_{0,R}, t) = q_2$. Let $s = (\sigma_1, \gamma_1) \dots (\sigma_n, \gamma_n)$ and $t = (\gamma'_1, \sigma'_1) \dots (\gamma'_m, \sigma'_m)$. Let $(x, z) \in q_1 \cap q_2$ be a state in $X \times Z$. Then, we have

$$\begin{aligned} &i) \quad \exists (x_0, z_0) \in \tilde{q}_0, \exists s_1 \in \mathcal{L}(G \times S, (x_0, z_0)) : \\ &\quad I_S(s_1, (x_0, z_0)) = \sigma_1\gamma_1 \dots \sigma_n\gamma_n \wedge \eta((x_0, z_0), s_1) = (x, z) \\ &ii) \quad \exists (x,' z') \in \tilde{q}_{0,R}, \exists t_1 \in \mathcal{L}(G \times S, (x, z)) : \\ &\quad I_S(t_1, (x, z)) = \sigma'_m\gamma'_m \dots \sigma'_1\gamma'_1 \wedge \eta((x, z), t_1) = (x,' z'). \end{aligned}$$

Note that i) is from Proposition 1, while ii) comes from Proposition 3. Therefore, we know that $\eta((x_0, z_0), s_1 t_1) = (x', z')$ and $I_S(s_1 t_1, (x_0, z_0)) = \sigma_1\gamma_1 \dots \sigma_n\gamma_n\sigma'_m\gamma'_m \dots \sigma'_1\gamma'_1$. This implies that $(\sigma_1, \gamma_1) \dots (\sigma_n, \gamma_n)(\sigma'_m, \gamma'_m) \dots (\sigma'_1, \gamma'_1) \in \mathcal{L}(\widetilde{Obs}(G \times S))$. ■

Proof of Theorem 2

Proof: According to the definition of $\tilde{f}$, we have that

$$\tilde{f}((\sigma_1, \gamma_1) \cdots (\sigma_k, \gamma_k)) = \left\{ (x, z) \in X \times Z : \begin{matrix} \exists s \in \mathcal{L}(G \times S)) \text{ s.t. } \eta(s) = (x, z) \\ \text{and } I_S(s) = \sigma_1\gamma_1 \dots \sigma_k\gamma_k \end{matrix} \right\}$$

Based on Proposition 3, we obtain that

$$\tilde{f}_R((\gamma_n, \sigma_n) \cdots (\gamma_{k+1}, \sigma_{k+1})) = \left\{ (x, z) \in X \times Z : \begin{matrix} \exists t \in \mathcal{L}(G \times S, (x, z)) \\ \text{s.t. } I_S(t, (x, z)) = \sigma_{k+1}\gamma_{k+1} \dots \sigma_n\gamma_n \end{matrix} \right\}.$$

Then, by intersecting these two sets, we obtain Theorem 2. ■

Proof of Theorem 3

Proof: ($\Rightarrow$) By contraposition. Suppose that there exists a state $(q_1, q_2) \in Q_{TW}$ such that $q_1 \cap q_2 \neq \emptyset$ and $q_1 \cap q_2 \subseteq X_S$. According to Proposition 4, there exist

strings $\alpha = (\sigma_1, \gamma_1)\dots(\sigma_n, \gamma_n) \in \mathcal{L}(\widetilde{Obs}(G \times S))$ and $\beta = (\gamma'_1, \sigma'_1)\dots(\gamma'_m, \sigma'_m) \in \mathcal{L}(\widetilde{Obs}_R(G \times S))$ such that $\tilde{f}(\tilde{q}_0, \alpha) = q_1, \tilde{f}_R(\tilde{q}_{0,R}, \beta) = q_2$ and

$$\sigma_1\gamma_1\dots\sigma_n\gamma_n\sigma'_m\gamma'_m\dots\sigma'_1\gamma'_1 \in \mathcal{I}_S.$$

Let $st \in \mathcal{L}(G \times S)$ be a string such that $I_S(s) = \sigma_1\gamma_1\dots\sigma_n\gamma_n$ and $I_S(st) = \sigma_1\gamma_1\dots\sigma_n\gamma_n\sigma'_m\gamma'_m\dots\sigma'_1\gamma'_1$. Then, by Theorem 2, we have that

$$\left\{\eta(s') \;\middle|\; \begin{array}{c}\exists s't' \in \mathcal{L}(G \times S)) \text{ s.t.} \\ I_S(s') = I_S(s) \text{ and } I_S(s't') = I_S(st)\end{array}\right\} \subseteq X_S.$$

This means that $G \times S$ is not infinite-step opaque.

($\Leftarrow$) Also by contraposition. Suppose that $G \times S$ is not infinite-step opaque w.r.t. $\Sigma_{o,a}$, $\Sigma_{c,a}$ and X_S. This means that there exists a string $st \in \mathcal{L}(G \times S)$ such that

$$\left\{\eta(s') \;\middle|\; \begin{array}{c}\exists s't' \in \mathcal{L}(G \times S) \text{ s.t.} \\ I_S(s') = I_S(s) \text{ and } I_S(s't') = I_S(st)\end{array}\right\} \subseteq X_S.$$

Then, let

$$I_S(st) = \sigma_1\gamma_1\dots\sigma_n\gamma_n\sigma_{n+1}\gamma_{n+1}\dots\sigma_{n+m}\gamma_{n+m} \in \mathcal{I}_S$$

such that $I_S(s) = \sigma_1\gamma_1\dots\sigma_n\gamma_n$. This implies that $\alpha := (\sigma_1, \gamma_1)\dots(\sigma_n, \gamma_n) \in \mathcal{L}(\widetilde{Obs}(G \times S))$ and $\beta := (\gamma_{n+m}, \sigma_{n+m})\dots(\gamma_{n+1}, \sigma_{n+1}) \in \mathcal{L}(\widetilde{Obs}_R(G \times S))$. Then, by the construction of $Obs_{TW}(G \times S)$, we have string

$$\begin{aligned}\varphi = &((\sigma_1, \gamma_1), \epsilon)\dots((\sigma_n, \gamma_n), \epsilon)(\epsilon, (\gamma_{n+m}, \sigma_{n+m})) \\ &\dots(\epsilon, (\gamma_{n+1}, \sigma_{n+1})) \in \mathcal{L}(Obs_{TW}(G \times S))\end{aligned} \tag{24}$$

and string φ leads to state $(q_1, q_2) := (\tilde{f}(\tilde{q}_0, \alpha), \tilde{f}_R(\tilde{q}_{0,R}, \beta))$. According to Theorem 2, we have

$$q_1 \cap q_2 = \left\{\eta(s') \;\middle|\; \begin{array}{c}\exists s't' \in \mathcal{L}(G \times S)) \text{ s.t.} \\ I_S(s') = I_S(s) \text{ and } I_S(s't') = I_S(st)\end{array}\right\}.$$

Therefore, $q_1 \cap q_2 \subseteq X_S$. Furthermore, $q_1 \cap q_2 \neq \emptyset$ as $\eta(s) \in q_1 \cap q_2$. This completes the contrapositive proof. ■

REFERENCES

[1] M. V. S. Alves, L. K. Carvalho, and J. C. Basilio, "Supervisory control of timed networked discrete event systems," in *Proc. 56th IEEE Conf. Decis. Control*, 2017, pp. 4859–4865.

[2] E. Badouel, M. Bednarczyk, A. Borzyszkowski, B. Caillaud, and P. Darondeau, "Concurrent Secrets," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 17, no. 4, pp. 425–446, 2007.

[3] J. W. Bryans, M. Koutny, L. Mazaré, and P. Ryan, "Opacity generalised to transition systems," *Int. J. Inf. Secur.*, vol. 7, no. 6, pp. 421–435, 2008.

[4] L. K. Carvalho, J. C. Basilio, and M. V. Moreira, "Robust diagnosis of discrete event systems against intermittent loss of observations," *Automatica*, vol. 48, no. 9, pp. 2068–2078, 2012.

[5] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. Berlin, Germany: Springer, 2008.

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

[7] L. Hélouët, H. Marchand, and L. Ricker, "Opacity with powerful attackers," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 464–471.

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

[9] Y. Ji, Y.-C. Wu, and S. Lafortune. "Enforcement of opacity by public and private insertion functions," *Automatica*, vol. 93, pp. 369–378, 2018.

[10] C. Keroglou and C. N. Hadjicostis, "Probabilistic system opacity in discrete event systems," *Discrete Event Dyn. Sys.: Theory Appl.*, vol. 28, no. 2, pp. 289–314, 2018.

[11] J. Komenda and F. Lin, "Modular supervisory control of networked discrete-event systems," in *Proc. 13th Int. Workshop Discrete Event Syst.*, 2016, pp. 85–90.

[12] R. Kumar and S. Takai, "Inference-based ambiguity management in decentralized decision-making: Decentralized control of discrete event systems," *IEEE Trans. Autom. Control*, vol. 52, no. 10, pp. 1783–1794, Oct. 2007.

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

[14] F. Lin, "Control of networked discrete event systems: Dealing with communication delays and losses," *SIAM J. Control Optimiz.*, vol. 52, no. 2, pp. 1276–1298, 2014.

[15] F. Lin *et al.*, "Information control in networked discrete event systems and its application to battery management systems," *Discrete Event Dyn. Syst.*, vol. 30, no. 2, pp. 243–268, 2020.

[16] F. Lin, W. Wang, L. Han, and B. Shen, "State estimation of multi-channel networked discrete event systems," *IEEE Trans. Control Netw. Syst.*, vol. 7, no. 1, pp. 53–63, Mar. 2020.

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

[18] C. Nunes, M. V. Moreira, M. Alves, L. K. Carvalho, and J. C. Basilio, "Codiagnosability of networked discrete event systems subject to communication delays and intermittent loss of observation," *Discrete Event Dyn. Syst.*," vol. 28, no. 2, pp. 215–246, 2018.

[19] A. Rashidinejad, M. Reniers, and L. Feng, "Supervisory control of timed discrete-event systems subject to communication delays and non-FIFO observations," in *Proc. 14th Int. Workshop Discrete Event Syst.*, 2018, pp. 456–463.

[20] A. Saboori and C. N. Hadjicostis, "Verification of infinite-step opacity and complexity considerations," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1265–1269, May. 2012.

[21] S. Shu and F. Lin, "Supervisor synthesis for networked discrete event systems with communication delays," *IEEE Trans. Autom. Control*, vol. 60, no. 8, pp. 2183–2188, Aug. 2015.

[22] S. Shu and F. Lin, "Deterministic networked control of discrete event systems with nondeterministic communication delays," *IEEE Trans. Autom. Control*, vol. 62, no. 1, pp. 190–205, Jan. 2017.

[23] S. Shu and F. Lin, "Predictive networked control of discrete event systems," *IEEE Trans. Autom. Control*, vol. 62, no. 9, pp. 4698–4705, Sep. 2017.

[24] S. Takai and T. Ushio, "Verification of codiagnosability for discrete event systems modeled by mealy automata with nondeterministic output functions," *IEEE Trans. Autom. Control*, vol. 57, no. 3, pp. 798–804, Mar. 2012.

[25] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Decidability of opacity verification problems in labeled petri net systems," *Automatica*, vol. 80, pp. 48–53, 2017.

[26] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Verification of state-based opacity using petri nets," *IEEE Trans. Autom. Control*, vol. 62, no. 6, pp. 2823–2837, Jun. 2017.

[27] T. Ushio and S. Takai, "Nonblocking supervisory control of discrete event systems modeled by mealy automata with nondeterministic output functions," *IEEE Trans. Autom. Control*, vol. 61, no. 3, pp. 799–804, Mar. 2016.

[28] J. Yang, W. Deng, C. Jiang, and D. Qiu, "Opacity of networked discrete event systems," in *Proc. 58th IEEE Conf. Decis. Control*, 2019, pp. 6736–6741.

[29] 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.

[30] X. Yin and S. Li, "Verification of opacity in networked supervisory control systems with insecure control channels," in *Proc. IEEE Conf Decis. Control*, 2018, pp. 4851–4856.

[31] X. Yin and S. Li, "Opacity of networked supervisory control systems over insecure multiple channel networks," in *Proc. 58th IEEE Conf. Decis. Control*, 2019, pp. 7641–7646.

[32] T.-S. Yoo and S. Lafortune, "A general architecture for decentralized supervisory control of discrete-event systems," *Discrete Event Dyn. Sys.: Theory Appl.*, vol. 12, no. 3, pp. 335–377, 2002.

[33] K. Zhang, X. Yin, and M. Zamani, "Opacity of nondeterministic transition systems: A (bi)simulation relation approach," *IEEE Trans. Autom. Control*, vol. 64, no. 12, pp. 5116–5123, Dec. 2019.

[34] R. Zhang, K. Cai, Y. Gan, and W. M. Wonham, "Distributed supervisory control of discrete-event systems with communication delay," *Discrete Event Dyn. Syst.*, vol. 26, no. 2, pp. 263–293, 2016.
[35] B. Zhao, F. Lin, C. Wang, X. Zhang, M. P. Polis, and L. Y. Wang, "Supervisory control of networked timed discrete event systems and its applications to power distribution networks," *IEEE Trans. Control Netw. Syst.*, vol. 4, no. 2, pp. 146–158, Jun. 2017.
[36] Y. Zhu, L. Lin, and R. Su, "Supervisor obfuscation against actuator enablement attack," in *Proc. 18th Eur. Control Conf.*, 2019, pp. 1760–1765.
[37] Y. Zhu, L. Lin, S. Ware, and R. Su, "Supervisor synthesis for networked discrete event systems with communication delays and lossy channels," in *Proc. 58th IEEE Conf. Decis. Control*, 2019, pp. 6730–6735.

Shuo Yang was born in Hunan, China, in 2000. He has been working toward the B.Eng. degree with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, since 2017.

His research interests include control theory, cyber-physical systems, and discrete-event systems.

Junyao Hou (Student, Member IEEE) was born in Hunan, China, in 1993. He received the B.S. degree from Southwest University, Chongqing, China, in 2015, and the M.S. degree from Xidian University, Xi'an, China, in 2018, all in automation. He is currently working toward the Ph.D. degree with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China.

His research interests include the security of cyber-physical systems and abstraction-based synthesis in formal methods.

Xiang Yin (Member, IEEE) was born in Anhui, China, in 1991. He received the B.Eng. degree from Zhejiang University, Hangzhou, China, in 2012, the M.S. and Ph.D. degrees from the University of Michigan, Ann Arbor, MI, USA, in 2013 and 2017, respectively, all in electrical engineering.

Since 2017, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is currently an Associate Professor. His research interests include formal methods, discrete-event systems, and cyber-physical systems.

Dr. Yin is serving as the Co-chair of the IEEE CSS Technical Committee on Discrete Event Systems, an Associate Editor for the *Journal of Discrete Event Dynamic Systems: Theory & Applications*, and a member of the IEEE CSS Conference Editorial Board. He was the finalist in IEEE Conference on Decision and Control (CDC) Best Student Paper Award, 2016.

Shaoyuan Li (Senior Member, IEEE) received the Ph.D. degree in control science and engineering from Nankai University, Tianjin, China, in 1997.

Since 1997, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is currently a Professor. His research interests include model predictive control, dynamic system optimization, and cyber-physical systems.

Dr. Li is the Vice President of the Chinese Association of Automation.