CONFIDENTIALITY OF CYBER-PHYSICAL SYSTEMS USING EVENT-BASED CRYPTOGRAPHY

Públio M. Lima, Lilian K. Carvalho, Marcos V. Moreira

COPPE - Electrical Engineering Program, Universidade Federal do Rio de Janeiro, Cidade Universitária, Ilha do Fundão, Rio de Janeiro, 21.945-970, RJ, Brazil,
Emails: publio@poli.ufrj.br, lilian.carvalho@poli.ufrj.br, moreira.mv@poli.ufrj.br

Abstract: One of the most important challenges for the application of cyber-physical systems (CPS) in smart industries is ensuring its security against cyber attacks. In this paper, we consider that the CPS is abstracted as a Discrete-Event System (DES), and we consider cyber attacks where the intruder eavesdrops the sensor communication channel to detect the occurrence of a sequence in the secret behavior of the system. In order to prevent the attacker from getting information from the sensor channel, we introduce a new cryptographic scheme based on events called event-based cryptography. We also define the property of confidentiality of DES, present a necessary and sufficient condition for ensuring this property, and propose a verification test.

Keywords: Cyber-security, Cyber-physical systems, Network communication, Discrete-event systems, Automata, Confidentiality, Encryption.

1. INTRODUCTION

Cyber attacks can be classified as active or passive (Stallings, 2006). In the former, the intruder attempts to alter system resources or affect its operation, whereas in the latter, the attacker tries to learn or make use of the information obtained from the system, but is not capable of affecting its operation. In this work, we consider only passive attacks in the sensor channel of Cyber-Physical Systems (CPS), as depicted in Figure 1, *i.e.*, we consider that the communication channel between sensors and supervisor can be invaded by an attacker that eavesdrops the transmitted messages in order to identify the occurrence of a secret behavior of the system. This secret behavior may represent, for instance, a sequence of operations that the intruder cannot know, or the reach of a specific state in which the system is more vulnerable to an active attack.

At some level of abstraction, considering the logical behavior of the system, CPS can be modeled as Discrete-Event Systems (DES) (Goes et al., 2017; Nunes et al., 2018; Lima et al., 2018). In this paper, we focus on the logical behavior of the system described by the sequences of events that it can execute, *i.e.*, the system is abstracted as a DES.

In the context of Discrete-Event Systems (DES), the term security has been used as the capability of detecting network attacks and preventing damages caused by an active attack (Thorsley and Teneketzis, 2006; Goes et al., 2017; Lima et al., 2018; Carvalho et al., 2018; Su, 2018), and also as the capability of misleading an outside observer in order to hide a secret system behavior, referred in the literature to as opacity (Saboori and Hadjicostis, 2007; Jacob et al., 2016; Lafortune et al., 2018; Lin, 2011). A system is said to be opaque with respect to a secret language and a non-secret language, if it is possible to ensure that the secret behavior is kept hidden from the attacker, *i.e.*, for any sequence t in the secret language, there exists another sequence t' in the non-secret language, with the same observation.

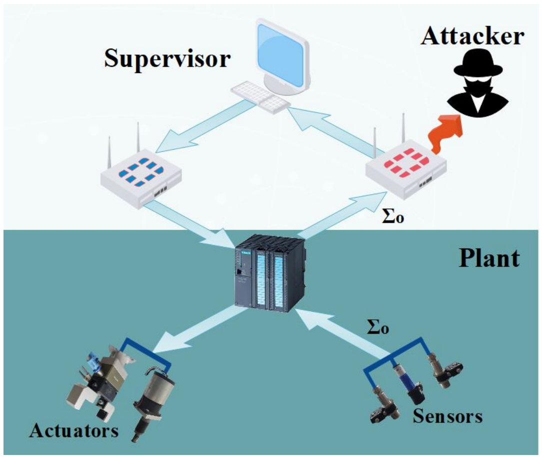

Figure 1. Cyber attack in the sensor channel.

Opacity enforcers have been proposed in the literature to guarantee that the intruder is not able to discover the occurrence of a secret behavior (Barcelos and Basilio, 2018; Yin and Lafortune, 2015; Tong et al., 2018; Wu

[1] This work has been partially supported by the National Council for Scientific and Technological Development - Brasil (CNPq) - under grants 305267/2018-3, 431307/2018-0, and 436672/2018-9, FAPERJ, and the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

and Lafortune, 2014). The main drawback of enforcing opacity is that the intended receiver of the message is also not capable of distinguishing the occurrence of the secret sequence from the occurrence of the non-secret sequence that has the same observation. Since, in our case, the intended receiver is a supervisor, and different observations may lead to different control actions, then, if two different sequences are transmitted with the same observation, the supervisor will not be able to distinguish them, which may change the expected closed-loop behavior.

In the context of Information Technology, the problem of ensuring security of information in the communication channel between sender and receiver is called confidentiality. Confidentiality requires that only the sender and the intended receiver should be able to understand the contents of the transmitted message (Kurose and Ross, 2011). Thus, in order to ensure confidentiality, it is necessary to encrypt the transmitted message. Encryption is the process of transforming a message, plain text, into something illegible, cipher text, and by doing so, preventing the intruder from understanding the signals transmitted through the communication channel (Stallings, 2006).

Only few works consider the use of cryptography to ensure security in communication networks of CPS modeled as DES (Fritz and Zhang, 2018; Fritz et al., 2019). In Fritz et al. (2019), the authors adapt known public-key cryptography for the transmitted data of CPS modeled by Petri nets. In order to ensure that the public-key is strong enough, large prime numbers must be used. In this case, the transmission of a bit or an integer must be replaced with the transmission of a large number, increasing the amount of transmitted data, and slowing down the communication time rate.

In this paper, we introduce a new defense strategy based on cryptography that does not change the structure of the transmitted messages, avoiding the increase in the amount of transmitted data through the communication channels. In order to do so, the observation of the events by the sensors of the plant are encrypted before transmission, and then decrypted in the supervisor's site. Since the encryption is carried out in the event level, we call this type of cryptography as event-based cryptography. We also introduce the property of confidentiality of DES with respect to a secret language and an encryption function, and present a necessary and sufficient condition for this property. We also propose a test to verify it.

This paper is organized as follows. In Section 2, we present some preliminary concepts. In Section 3, we propose a defense strategy against cyber attacks in the sensor communication channel of a CPS. In Section 4, we formalize the property of confidentiality of DESs. In Section 5, we introduce transition-based encryption functions, and in Section 6, we present a necessary and sufficient condition for confidentiality and show a method for verifying this property of DESs. Finally, in Section 7, the conclusions are drawn.

2. PRELIMINARIES

Let $G = (X, \Sigma, f, x_0)$ be a deterministic automaton, where X is the set of states, Σ is the finite set of events, $f : X \times \Sigma \to X$ is the transition function, and $x_0 \in X$ is the initial state of the system. Let $\Gamma_G : X \to 2^\Sigma$ be the active event function, where $\Gamma_G(x) = \{\sigma \in \Sigma : f(x, \sigma) \text{ is defined}\}$, for all $x \in X$. The domain of the transition function f can be extended to $X \times \Sigma^*$, where Σ^* denotes the Kleene-closure of Σ, as usual: $f(x, \varepsilon) = x$, and $f(x, s\sigma) = f(f(x, s), \sigma)$, for all $s \in \Sigma^*$, and $\sigma \in \Sigma$, where ε denotes the empty sequence. The language generated by G is defined as $L(G) = \{s \in \Sigma^* : f(x, s) \text{ is defined}\}$.

A nondeterministic automaton is a four-tuple $\mathcal{G} = (X, \Sigma \cup \{\varepsilon\}, f_{nd}, x_0)$, where the elements of $\mathcal{G}$ have the same interpretation as in the deterministic automaton G, with the exception that the transition function can be nondeterministic, $f_{nd} : X \times \Sigma \cup \{\varepsilon\} \to 2^X$, and the initial state can be defined as a set $x_0 \subseteq X$.

The set of events Σ can be partitioned as $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, where Σ_o and Σ_{uo} denote, respectively, the sets of observable and unobservable events of the system. The projection $P_o : \Sigma^* \to \Sigma_o^*$, where $\Sigma_o \subset \Sigma$, is defined as usual. The projection operation can be extended to languages by applying them to all sequences in the language.

It is important to remark that it is always possible to compute a deterministic automaton whose generated language is equal to $P_o(L(G))$. We adopt, in this paper, the observer automaton presented in Cassandras and Lafortune (2008), denoted as $Obs(G, \Sigma_o) = (X_o, \Sigma_o, f_o, x_{0,o})$.

Let G_1 and G_2 be two automata, then $G_1 \parallel G_2$ and $G_1 \times G_2$, denote, respectively, the parallel composition and the product of G_1 and G_2 (Cassandras and Lafortune, 2008).

3. DEFENSE STRATEGY

In this paper, we consider a CPS composed of a plant and a supervisor, as shown in Figure 1, where the communication between plant and supervisor is carried out using a wired or wireless network. The channel that is used to send information, gathered by sensors, from the plant to the supervisor, is called sensor channel, and the channel where control actions are transmitted from the supervisor to the plant, enabling actuators, is called supervisory control channel. Let G denote the automaton model of the plant and H a realization of the supervisor. Then, the closed-loop system model T is obtained by making the parallel composition $T = G \| H$.

Consider now that the sensor channel between plant and supervisor is vulnerable to attacks, as shown in Figure 1, and that the attacker can observe all events transmitted from the plant to the supervisor. In this paper, we consider that the supervisory control channel is secure, *i.e.,* cannot be attacked. Since the attacker observes only events transmitted from the plant to the supervisor, then the controlled plant from the attacker's point of view is represented by $T_o = Obs(T, \Sigma_o) = (X_o, \Sigma_o, f_o, x_{0,o})$. The language generated by T_o is denoted as L_o.

The objective of the attacker is to estimate that a sequence in the secret language, denoted as $L_S \subset L_o$, has occurred based on the observation of the events gathered by the sensors of the plant. We assume that L_S is a regular language, and make the following assumptions regarding

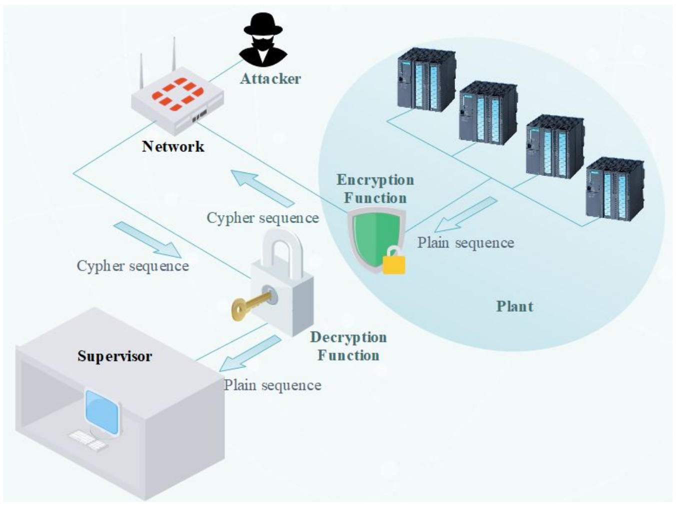

Figure 2. Defense Strategy.

the attacker capabilities: *(i)* the attacker can read all signals transmitted from the plant to the supervisor; *(ii)* the attacker knows the closed-loop system model T_o; and *(iii)* the current state of the system is unknown when the attacker starts to observe it.

In this paper, we propose a defense strategy based on cryptography to guarantee that, given that a secret sequence $s \in L_S$ has been executed by the system, the attacker is not able to estimate the occurrence of any sequence in the secret language L_S. We denote by plain event any event not modified by an encryption function, and by cipher event those modified by encryption functions. We also use this terminology for sequences and languages.

In Figure 2, we present the closed-loop system with the defense strategy that encrypts an observable event $\sigma \in \Sigma_o$, generating the cipher event σ_c before it is transmitted to the supervisor, and the application of its inverse encryption at the supervisor site in order to recover σ from σ_c. Notice that the encryption function cannot mislead the supervisor. Thus, the plain sequence s must be recoverable from the cipher sequence s_c, *i.e.*, the cryptography function must be invertible in order to the supervisor be able to read and actuate correctly on the system. In addition, since the supervisory control must be able to act just after the observation of an event executed by the system, it is important that one cipher event be transmitted to the supervisor after every occurrence of an observable event in the plant. This procedure is guaranteed in this paper by considering only stream encryption functions, *i.e.*, the encryption function considered in this paper encrypts one observable event at a time.

It is important to remark that the defense strategy proposed in this paper does not increase the amount of data communicated in the sensor channel, since, after encryption of the observed plain event, a cipher event is transmitted through the communication channel.

4. CONFIDENTIALITY OF DES

In this section, we formally present the definition of confidentiality of DES. In order to do so, it is first necessary to define function $Suf : \Sigma^* \rightarrow 2^{\Sigma^*}$, which returns the set of all suffixes of a sequence $s \in \Sigma^*$ given by:

$$Suf(s) = \{s_2 \in \Sigma^* : (\exists s_1 \in \Sigma^*)[s = s_1 s_2]\}.$$

The suffix operation can be extended to a language $L \subseteq \Sigma^*$ as $Suf(L) = \{s_2 \in \Sigma^* : (\exists s \in L)(\exists s_1 \in \Sigma^*)[s = s_1 s_2]\}$.

Since, by assumption, the attacker may start the observation of the sequence at any time, then the attacker observes a suffix of the cipher sequence s_c, $s' \in Suf(s_c)$. Based on the observation of s', and the knowledge of the closed-loop system model T_o, the attacker estimates the sequences of L_o that may have occurred. If s' cannot be a suffix of any sequence in L_o, then the estimated language is equal to the empty set. On the other hand, the attacker estimates all sequences of L_o that have a suffix equal to s'. Let us denote the set of all estimated sequences from the observation of s' by $L_{e,s'} = \{s \in L_o : s' \in Suf(s)\}$. Notice that, if $L_{e,s'} \subseteq L_S$ and $L_{e,s'} \neq \emptyset$, the attacker estimates that a secret sequence has occurred. Therefore, after the occurrence of a secret sequence, it is important to mislead the attacker by either making $L_{e,s'} = \emptyset$ or $L_{e,s'} \cap (L_o \setminus L_S) \neq \emptyset$.

Example 1. Let us consider that the language generated by the closed-loop system is $L_o = \{\varepsilon, a, ab, c, ca\}$, and that the secret language is $L_S = \{ab, ca\}$. Let us also consider that the system executes and transmits to the supervisor sequence $s = ca$. Thus, depending on when the attacker starts to eavesdrop the system, any suffix of $Suf(s) = \{\varepsilon, a, ca\}$ can be observed. Consider that the observed sequence is $s' = a$. Then, since the attacker knows language L_o, the attacker estimates the occurrence of the sequences in $L_{e,a} = \{a, ca\}$. Since $a \notin L_S$, then the attacker is not certain about the occurrence of a secret sequence. However, if the observed sequence is $s' = ca$, then $L_{e,ca} = \{ca\}$, and the attacker is certain about the occurrence of a secret sequence. □

An encryption function $F_E : L_o \rightarrow \Sigma_o^*$ transforms a plain sequence $s \in L_o$ into a cipher sequence $s_c \in \Sigma_o^*$. This encryption function must be invertible in order to ensure confidentiality, *i.e.*, there must exist a decryption function $F_E^{-1} : \Sigma_o^* \rightarrow \Sigma_o^*$ such that $F_E^{-1}(F_E(s)) = s, \forall s \in L_o$. If the encryption function is applied to all sequences of L_o, then we obtain the cipher language $L_c = \{F_E(s) : s \in L_o\}$. The automaton that generates L_c is denoted as T_c. In the sequel, we formally define the property of confidentiality of DES.

Definition 1. A language L_o is said to be confidential with respect to the secret languages L_S and encryption function F_E if:

$$[\forall s \in L_S][\forall s' \in Suf(F_E(s))] \Rightarrow (s' \notin Suf(L_S)) \vee (s' \in Suf(L_o \setminus L_S)).$$ □

According to Definition 1, a system is said to be confidential if after the occurrence of a sequence $s \in L_S$: *(i)* the attacker does not estimate that a sequence in L_S has occurred, *i.e.*, $s' \notin Suf(L_S)$ for all $s' \in Suf(F_E(s))$; or *(ii)* there exists $s' \in Suf(F_E(s))$ such that $s' \in Suf(L_S)$, but there also exists a non secret sequence in the estimated language $L_{e,s'}$, *i.e.*, $s' \in Suf(L_o \setminus L_S)$. Therefore, if either the attacker cannot estimate a secret sequence by observing s', or if the attacker is uncertain about the occurrence of a secret sequence, the system is said to be

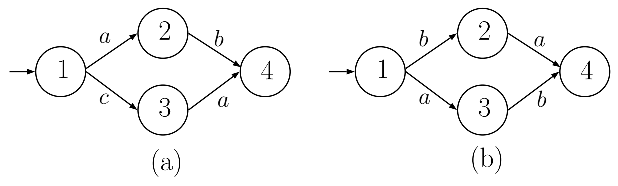

Figure 3. Automata T_o and T_c of Example 2.

confidential with respect to the secret language L_S and encryption function F_E.

Example 2. Let T_o be the automaton of the closed-loop system depicted in Figure 3(a), and consider the secret language $L_S = \{ab, ca\}$. Let us consider an encryption function F_E which changes the observation of sequences in T_o such that a is encrypted as b, ab is encrypted as ba, c is encrypted as a, and ca is encrypted as ab. Automaton T_c, that models the cipher language L_c, is depicted in Figure 3(b). In order to verify confidentiality we need to consider both secret sequences, *(i)* ab, and *(ii)* ca. Let us first consider that the system executes sequence ab and that the attacker has started to observe the system when the plant was in the initial state 1. Thus, the attacker observes the transmitted sequence ba. After observing ba the attacker is not able to estimate the current state of the system since there is no sequence in L whose suffix is ba, *i.e.*, $L_{e,ba} = \emptyset$. Let us now suppose that the attacker has started to observe the system only after b has been transmitted, and therefore, the attacker observes only event a. In this case, $L_{e,a} = \{a, ca\}$, and the attacker does not know if the plant was in the initial state 1, and after the occurrence of event a reached state 2, or if the plant was in state 3, and is now in state 4. Thus the attacker is not certain about the occurrence of the secret sequence ca. In this case, the occurrence of sequence ab is kept confidential from the attacker by using encryption function F_E.

Let us consider now that plant has generated sequence ca, observed by the attacker as ab. After observing ab the attacker would estimate $L_{e,ab} = \{ab\}$, and is certain that a secret sequence has been generated by T_o. Although ab is not the sequence generated by T_o, it is a secret sequence that leads to the same state of the plant as sequence ca. Thus, according to Definition 1, L_o is not confidential with respect to the secret language L_S and encryption function F_E. □

5. TRANSITION-BASED ENCRYPTION FUNCTIONS

An encryption function F_E is classified as transition-based if for all $s, s' \in L_o$ such that $f_o(x_{0,o}, s) = f_o(x_{0,o}, s')$, then $F_E(se) = F_E(s'e)$, *i.e.*, the events labeling the transitions of T_o are encrypted in the same way independently of the sequence of events that has been executed before. As a consequence, automaton T_c, that generates the cipher language L_c, can be obtained from T_o following the steps of Algorithm 1.

Algorithm 1. (Computation of T_c).

Inputs: $T_o = (X_o, \Sigma_o, f_o, x_{0,o})$ and F_E.

Output: $T_c = (X_c, \Sigma_c, f_c, x_{0,c})$.

```
1: Define $X_c = X_o$, $\Sigma_c = \Sigma_o$, $x_{0,c} = x_{0,o}$.
2: Define $f_c(x, e_c) = f_o(x, e)$, for all $x \in X_o$ and $e \in \Sigma_o$,
   where $e_c$ denotes the cipher event associated with
   the plain event $e$, obtained from the transition-based
   encryption function $F_E$.
```

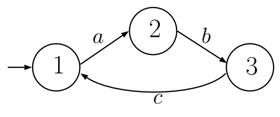

Figure 4. Plant model T_o of Example 3.

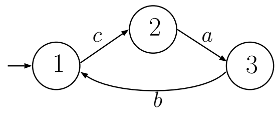

Figure 5. Automaton T_c obtained using the transition-based encryption function applied to automaton T_o of Figure 4.

Notice, according to Algorithm 1, that the transition-based encryption function does not change the structure of the automaton model T_o, *i.e.*, T_c and T_o have the same states and transition structure. The following result provides a necessary and sufficient condition for a transition-based encryption function be invertible.

Theorem 1. A transition-based encryption function F_E is invertible if, and only if, T_c, obtained according to Algorithm 1, is deterministic.

Proof: The proof is straightforward and will be ommitted due to lack of space. ■

Example 3. Let $T_o = (X_o, \Sigma_o, f_o, x_{0,o})$, depicted in Figure 4, be the automaton of the closed-loop system, and consider that the secret language is formed of all sequences that reach state 3, *i.e.*, $L_S = (abc)^*ab$. A transition-based encryption function F_E can be defined such that for every observation of event a in T_o, a is encrypted and transmitted as c in T_c, every observation of event b is encrypted as a, and every observation of event c is encrypted as b. Let us now consider that the attacker observes the cipher sequence ca, transmitted by the system. Since the attacker knows the closed-loop system model T_o, the attacker estimates that the system has reached state 2. However, in fact, the system has executed sequence ab, reaching the secret state 3. The model of the encrypted system T_c, obtained according to Algorithm 1, is depicted in Figure 5. □

Note that, since encryption is carried out after each observation of an event generated by the plant, then two different cipher sequences s_c, s'_c, obtained from two different plain sequences $s, s' \in L_o$, must have the same prefix of length n, if s and s' have the same prefix of length n.

Remark 1. It is also possible to define a language-based encryption function. A language-based encryption function can encrypt the event of the same transition of the system model as two completely different cipher events, depending on the sequence of events that has been executed before, *i.e.*, $F_E(se)$ can be different from $F_E(s'e)$, even if $f_o(x_{0,o}, s) = f_o(x_{0,o}, s')$. Notice that, in order to obtain an invertible language-based encryption function, then two different plain sequences cannot be transmitted as the same cipher sequence. Language-based encryption functions are not addressed in this paper, and will be considered in future works. □

In the following section, we present an algorithm to verify the confidentiality of DES.

6. CONFIDENTIALITY VERIFICATION

In order to obtain a method for the verification of confidentiality of DES, it is first necessary to associate the secret sequences of L_S with the states reached in the closed-loop system model after the execution of these sequences. Let $X_S = \{x \in X_o : (\exists s \in L_S)[x = f_o(x_{0,o}, s)]\}$ denote the set of secret states of T_o, and assume that there does not exist a sequence $s' \in L_o \setminus L_S$ such that $x = f_o(x_{0,o}, s')$ and $x \in X_S$. It is important to remark that if this assumption is not true, then, since it is assumed that L_S is a regular language, it is always possible to obtain a modified automaton T'_o for which the assumption is valid (Wu and Lafortune, 2013). Thus, the set of states of T_o can be partitioned as $X_o = X_S \dot\cup X_{NS}$, where $X_{NS} = \{x \in X_o : (\exists s \in L_o \setminus L_S)[x = f_o(x_{0,o}, s)]\}$ denotes the set of non-secret states of T_o.

In this paper, we obtain a verifier for the property of confidentiality of DES based on the verifier for codiagnosability proposed in Moreira et al. (2011). Since we assume that the current-state of the system is unknown when the attacker starts to eavesdrop the sensor communication channel, we need to present the current-state estimator of an automaton G, denoted as $\mathcal{E}(G) = (X_\mathcal{E}, \Sigma, f_\mathcal{E}, x_{0,\mathcal{E}})$ (Saboori and Hadjicostis, 2011). The current-state estimator of $G = (X, \Sigma, f, x_0)$ is computed in two steps: (i) obtain the nondeterministic automaton $\mathcal{G} = (X, \Sigma, f, X)$ from G, by defining the initial state of $\mathcal{G}$ as the set X; and (ii) compute $\mathcal{E}(G) = Obs(\mathcal{G}, \Sigma)$.

The set of states of the current-state estimator of the plant T_o, $\mathcal{E}(T_o) = (X_{o\mathcal{E}}, \Sigma_o, f_{o\mathcal{E}}, x_{0,o\mathcal{E}})$, can be partitioned as $X_{o\mathcal{E}} = X_{o\mathcal{E},S} \dot\cup X_{o\mathcal{E},NS} \dot\cup X_{o\mathcal{E},U}$, where $X_{o\mathcal{E},S}$ is the set formed only of secret states of X_S, $X_{o\mathcal{E},NS}$ is formed only of non-secret states of X_{NS}, and $X_{o\mathcal{E},U}$ is formed of states of X_S and X_{NS}. The language generated by $\mathcal{E}(T_o)$ is formed of all suffixes of the sequences of L_o, *i.e.*, $L(\mathcal{E}(T_o)) = Suf(L_o)$.

It is also possible to define the current-state estimator of the cipher automaton T_c as $\mathcal{E}(T_c) = (X_{c\mathcal{E}}, \Sigma_o, f_{c\mathcal{E}}, x_{0,c\mathcal{E}})$. The set of states of $\mathcal{E}(T_c)$ can also be partitioned as $X_{c\mathcal{E}} = X_{c\mathcal{E},S} \dot\cup X_{c\mathcal{E},NS} \dot\cup X_{c\mathcal{E},U}$, where $X_{c\mathcal{E},S}$ is formed only of secret states of X_S, $X_{c\mathcal{E},NS}$ is formed only of non-secret states of X_{NS}, and $X_{c\mathcal{E},U}$ is formed of states of X_S and X_{NS}. It is not difficult to see that $x_{0,c\mathcal{E}} = x_{0,o\mathcal{E}}$. The language generated by $\mathcal{E}(T_c)$ is formed of all suffixes of the cipher sequences of L_c, *i.e.*, $L(\mathcal{E}(T_c)) = Suf(L_c)$.

In the sequel we present the algorithm for the verification of confidentiality of DES.

Algorithm 2. (Confidentiality verifier).

Inputs: $T_o = (X_o, \Sigma_o, f_o, x_{0,o})$ and $T_c = (X_c, \Sigma_o, f_c, x_{0,c})$.

Output: Verifier $V = (X_V, \Sigma_o, f_V, x_{0,V})$.

```
1: Compute $\mathcal{E}(T_o)$.
2: Compute $\mathcal{E}(T_c)$.
3: Compute $V = \mathcal{E}(T_o) \parallel \mathcal{E}(T_c)$.
```

In Algorithm 2, the verifier automaton V is computed from the current-state estimators of the closed-loop system T_o and the cipher system T_c. The state estimate of the cipher system T_c represents the actual system state estimate after the execution of a sequence $s \in Suf(L_o)$, observed as the cipher sequence $s_c \in Suf(F_E(s))$. The state estimate of T_o, on the other hand, represents what the attacker estimates from the observed cipher sequence s_c. Thus, the parallel composition between the current-state estimators compares, after each new observed event, the system states that the attacker estimates from T_o with the actual system state estimate. In the sequel we present a necessary and sufficient condition for the confidentiality of a DES based on the verifier automaton computed in Algorithm 2.

Theorem 2. Let V be computed according to Algorithm 2. Then, L_o is confidential with respect to L_S and F_E if, and only if, for all $x_V = (x_{o\mathcal{E}}, x_{c\mathcal{E}}) \in X_V$ such that $x_{o\mathcal{E}} \in X_{o\mathcal{E},S}$, we have that $x_{c\mathcal{E}} \in X_{c\mathcal{E},NS}$.

Proof: Let us suppose that there is a state $(x_{o\mathcal{E}}, x_{c\mathcal{E}}) \in X_V$ such that $x_{o\mathcal{E}} \in X_{o\mathcal{E},S}$. Then, the observed sequence is a suffix $s' \in Suf(L_S)$ such that $s' \notin Suf(L_o \setminus L_S)$, *i.e.*, the attacker estimates that a secret sequence has been executed by the system. If, in this case, $x_{c\mathcal{E}}$ belongs to $X_{c\mathcal{E},S}$ or $X_{c\mathcal{E},U}$, then there exists a sequence $s \in L_S$ such that $s' \in Suf(F_E(s))$, which violates Definition 1 of confidentiality of the DES. If, on the other hand, $x_{c\mathcal{E}} \in X_{c\mathcal{E},NS}$, then, there does not exist $s \in L_S$ such that $s' \in Suf(F_E(s))$, *i.e.*, the plant has not executed a secret sequence, and the attacker would estimate wrongly that a secret behavior has occurred. ■

According to Theorem 2, the confidentiality of a DES can be easily verified by searching in the verifier V a state $x_V = (x_{o\mathcal{E}}, x_{c\mathcal{E}})$ such that $x_{o\mathcal{E}}$ is formed only of secret states in X_S, which means that the attacker is certain that a secret sequence has been executed, and $x_{c\mathcal{E}}$ has at least one secret state of X_S, which means that it is possible that the system has indeed generated a secret sequence. If there exists a state satisfying these conditions, then the language of the system L_o is not confidential with respect to L_S and F_E. Otherwise, L_o is confidential.

Example 4. Let us consider again automaton T_o and the cipher automaton T_c of Example 3, presented in Figures 4 and 5, respectively. The secret language is given by $L_S = (abc)^*ab$, and, consequently, $X_S = \{3\}$ and $X_{NS} = \{1, 2\}$. Following Steps 1 and 2 of Algorithm 2, we compute the current-state estimators $\mathcal{E}(T_o)$ and $\mathcal{E}(T_c)$, depicted in Figures 6(a) and 6(b), respectively. In this case, we have that $X_{o\mathcal{E},S} = X_{c\mathcal{E},S} = \{\{3\}\}$, $X_{o\mathcal{E},NS} = X_{c\mathcal{E},NS} = \{\{1\}, \{2\}\}$, and $X_{o\mathcal{E},U} = X_{c\mathcal{E},U} = \{\{1, 2, 3\}\}$.

Verifier $V = \mathcal{E}(T_o) \parallel \mathcal{E}(T_c)$, is presented in Figure 7. Since the unique state $x_V = (x_{o\mathcal{E}}, x_{c\mathcal{E}})$, where $x_{o\mathcal{E}} = \{3\} \in X_{o\mathcal{E},S}$ is state $(\{3\}, \{1\})$, and $\{1\} \in X_{c\mathcal{E},NS}$, then L_o is confidential with respect to the encryption function F_E and L_S. □

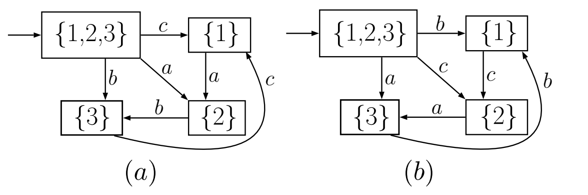

Figure 6. $\mathcal{E}(T_o)$ (a) and $\mathcal{E}(T_c)$ (b) of Example 4.

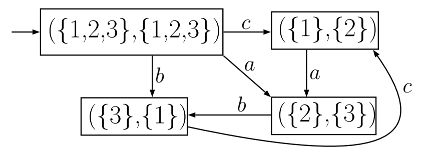

Figure 7. Verifier $V = \mathcal{E}(T_o) \parallel \mathcal{E}(T_c)$ of Example 4.

7. CONCLUSIONS

In this paper, we propose a defense strategy, based on an event-based cryptography, to prevent attackers from getting important information from the sensor communication channel between plant and supervisor of CPS. We define transition-based encryption functions, which changes the transmitted events in order to prevent the attacker from correctly estimating that a secret sequence has been executed by the system. This encryption function must be invertible in order to the supervisor be capable of recovering the sequence generated by the plant. We also introduce the notion of confidentiality of DES, associated with the capability of the encrypted system to hide a secret from the attacker, and we propose a method to verify this property.

We are currently studying language-based encryption functions, and how to obtain the cipher automaton T_c for this type of encryption. We are also investigating the application of known cryptography methods from Information Technology to the event-based cryptography proposed in this paper.

REFERENCES

Barcelos, R.J. and Basilio, J.C. (2018). Enforcing current-state opacity through shuffle in event observations. *IFAC-PapersOnLine*, 51(7), 100–105.

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

Cassandras, C.G. and Lafortune, S. (2008). *Introduction to discrete event systems.* Springer, Secaucus, NJ.

Fritz, R., Fauser, M., and Zhang, P. (2019). Controller encryption for discrete event systems. In *2019 American Control Conference (ACC)*, 5633–5638. PHILADELPHIA, CA, USA.

Fritz, R. and Zhang, P. (2018). Modeling and detection of cyber attacks on discrete event systems. *IFAC-PapersOnLine*, 51(7), 285–290.

Goes, R.M., Kang, E., Kwong, R.H., and Lafortune, S. (2017). Stealthy deception attacks for cyber-physical systems. In *Proceedings of the 56th IEEE Conference on Decision and Control*, 4224–4230. Melbourne, Australia.

Jacob, R., Lesage, J.J., and Faure, J.M. (2016). Overview of discrete event systems opacity: Models, validation, and quantification. *Annual Reviews in Control*, 41, 135–146.

Kurose, J.F. and Ross, K.W. (2011). *Computer networking: a top-down approach.* Addison Wesley.

Lafortune, S., Lin, F., and Hadjicostis, C.N. (2018). On the history of diagnosability and opacity in discrete event systems. *Annual Reviews in Control*, 45, 257–266.

Lima, P.M., Alves, M.V.S., Carvalho, L.K., and Moreira, M.V. (2018). Security against communication network attacks of cyber-physical systems. *Journal of Control, Automation and Electrical Systems*, 1(30), 125–135.

Lin, F. (2011). Opacity of discrete event systems and its applications. *Automatica*, 47(3), 496–503.

Moreira, M.V., Jesus, T.C., and Basilio, J.C. (2011). Polynomial time verification of decentralized diagnosability of discrete event systems. *IEEE Transactions on Automatic Control*, 56(7), 1679–1684.

Nunes, C.E., Moreira, M.V., Alves, M.V., Carvalho, L.K., and Basilio, J.C. (2018). Codiagnosability of networked discrete event systems subject to communication delays and intermittent loss of observation. *Discrete Event Dynamic Systems*, 28(2), 215–246.

Ramadge, P.J. and Wonham, W.M. (1987). Supervisory control of a class of discrete event processes. *SIAM J. on Control and Optimization*, 25(1), 206–230.

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

Saboori, A. and Hadjicostis, C.N. (2011). Verification of infinite-step opacity and complexity considerations. *IEEE Transactions on Automatic Control*, 57(5), 1265–1269.

Stallings, W. (2006). *Cryptography and network security, 4/E.* Pearson Education India.

Su, R. (2018). Supervisor synthesis to thwart cyber attack with bounded sensor reading alterations. *Automatica*, 94, 35–44.

Thorsley, D. and Teneketzis, D. (2006). Intrusion detection in controlled discrete event systems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, 6047–6054. San Diego, CA, USA.

Tong, Y., Cai, K., and Giua, A. (2018). Decentralized opacity enforcement in discrete event systems using supervisory control. In *2018 57th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, 1053–1058. IEEE.

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

Wu, Y.C. and Lafortune, S. (2014). Synthesis of insertion functions for enforcement of opacity security properties. *Automatica*, 50(5), 1336–1348.

Yin, X. and Lafortune, S. (2015). A new approach for synthesizing opacity-enforcing supervisors for partially-observed discrete-event systems. In *2015 American Control Conference (ACC)*, 377–383. IEEE.