Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 53-4 (2020) 95–100

Probabilistic verification of attack detection using logical observer [★]

Dimitri Lefebvre [*] Carla Seatzu [**] Christoforos N. Hadjicostis [***] Alessandro Giua [**]

[*] GREAH, Le Havre Normandy University, 75 rue Bellot, 76600 Le Havre, France, (e-mail: dimitri.lefebvre@univ-lehavre.fr)
[**] DIEE, University of Cagliari, Cagliari 09124, Italy,(e-mail: {seatzu, giua}@diee.unica.it)
[***] Department of Electrical and Computer Engineering, University of Cyprus, 75 Kallipoleos Av., Nicosia 1678, Cyprus, (e-mail: chadjic@ucy.ac.cy)

Abstract: This paper focuses on the detection of cyber-attacks in a timed probabilistic setting. The plant and the possible attacks are described in terms of a labeled continuous time Markov model that includes both observable and unobservable events, and where each attack corresponds to a particular subset of states. Consequently, attack detection is reformulated as a state estimation problem. A verification methodology is described using a parallel-like composition of the Markov model and its logical observer. The construction of this parallel composition allows us to (i) concisely characterize the set of attacks that can be detected based on the sequences of observations they generate, and (ii) compute performance indicators of interest, such as the *a priori* probability of an undetectable attack, the average detectability, and the mean delay to detection.

Keywords: Cyber-security, continuous time Markov model, average attack detectability

1. INTRODUCTION

Due to their heterogenous and often distributed nature, cyber-physical systems are exposed to attacks from malicious intruders. Therefore, there is an emerging need for developing tools that evaluate the risk of attacks in a quantitative way. Cyber-security in dynamical systems has been studied in the framework of continuous time systems and discrete event systems (Teixeira et al., 2012; Lun et al., 2019; Ding et al., 2018). As long as continuous time models are used, attack scenarios depend mainly on replay, zero dynamics, and bias injection (Teixeira et al., 2012). When discrete event models are considered, attacks depend on: (i) the communication channel where the attack happens, (ii) the attack impact on the transmitted data, and (iii) the mechanism to prevent damage (Rashidinejad et al., 2019).

Attacks may occur in the observation channel (known as a sensor attack), in the control channel (known as an actuator attack) or in both the observation and control channels (as in most realistic cases) (Carvalho et al., 2018). For example, vulnerabilities in the address resolution protocol may be exploited by malicious hosts in a local area network to implement attacks (Hubballi et al., 2011). Deletion, insertion, and replacement are typical examples of the ways an attack can alter the transmitted information (Cardenas et al., 2008). With the purpose of preventing the consequences of cyber-attacks, two main approaches have been presented in the literature: detection and prevention on the one hand, and synthesis of a resilient supervisor on the other hand. Detection approaches place an intrusion detection module in the system to detect an attack and prevent it before it causes damage to the system (Carvalho et al., 2018; Gao et al., 2019). For supervisory control approaches, a supervisor is synthesized which is resilient to attacks (Rashidinejad et al., 2019).

The problem considered in this paper concerns the first approach where attack detection is formulated in a timed probabilistic setting. The main advantages obtained by introducing probabilistic/timing information are associated with the ability to refine the detection decision. This refinement can appear in two distinct ways as described below.

- **Case 1: Logical Decision and Probabilistic Verification.** One can maintain a logical decision formulation (in other words, one can insist on obtaining a binary decision regarding the occurrence of an attack when that is absolutely certain) and use the probabilistic/timing information provided by the model to assess performance indicators of interest. Such performance indicators include, for instance, the *a priori* probability of (logically) detecting an attack or the *a priori* average delay involved in (logically) detecting an attack.
- **Case 2: Probabilistic Decision and Probabilistic Verification.** One can also relax the logical requirement on the decision (that an attack is detected with absolute certainty) by using the probabilistic information to determine the posterior likelihood of an attack, conditioned on the specific sequence of observations. Such relaxations can offer advantages in cases where the attacks of a given system may be undetectable in a logical setting whereas

[★] This paper is partially supported by the Visiting Professor/Scientist Program, Call 2018, funded by Region Sardinia, Italy. It is also partially supported by Project RASSR05871 MOSIMA funded by Region Sardinia, Italy, FSC 2014-2020, 2017, Area 3, Action Line 3.1.

10.1016/j.ifacol.2021.04.010

they may have probability of detection near unity (close to $1 - \epsilon$ for a small ϵ that is a design choice). Such systems might be considered ϵ-safe; in fact, when there is some system behavior that leads to violations of ϵ-safety, we can determine (as in Case 1) the *a priori* probability that the system will generate behavior that does not violate ϵ-safety.

It is worth pointing out that, in the context of fault diagnosis for stochastic DES that can be modeled as probabilistic finite automata under partial observation, Case 1 above appeared under the name of A-diagnosability whereas Case 2 appeared under the name of AA-diagnosability (Thorsley and Teneketzis, 2005). Similarly, in the context of observability analysis (again for probabilistic finite automata), Case 1 above appeared under the name of A-detectability (Keroglou and Hadjicostis, 2015) whereas Case 2 appeared under the name of AA-detectability (Keroglou and Hadjicostis, 2017).

In this paper we focus on the analysis of cyber attacks in a timed probabilistic setting assuming logical decisions and probabilistic verification (Case 1), though we plan to discuss the setting of probabilistic decisions and probabilistic verification (Case 2) in future extensions of this work. The considered systems are modeled with Markov models. A Markov chain is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained after the previous event, and the next state is a function of the current state and the event that is selected (Karlin and Taylor, 2012; Gagniuc, 2017). In continuous-time, a Markov chain is known as a continuous-time Markov model (CTMM), or as a Markov process, and is suitable for describing stochastic processes where the interarrival times between events are distributed exponentially (Norris, 1997; Gagniuc, 2017). Such processes are used in various domains. In particular, Poisson processes, birth and death processes, and parallel queuing systems are examples of uses of continuous-time Markov models in the domain of computer science and networked systems where cyber-attacks may occur (Gagniuc, 2017). With usual CTMM, the events from one state to the next one are assumed to be unknown and the best one can do is to compute the probability of the states. In this work, on the contrary, we are interested in the case where some (but not necessarily all) events are observable (Thorsley, 2010). For this purpose, labeled CTMM (LCTMM) are considered. Each time an observable event occurs, the label of the event is collected as well as the time when the event occurs. The observation of these timed events is helpful for refining the computation of the probabilities of the various states or to even disqualify certain states.

In this paper, LCTMM are used to model plants and their possible cyber-attacks. The first step is to design a logical observer by abstracting from the timing aspects in order to capture the logical structure of successive observations. Then, a parallel-like composition, similar to the one detailed in (Lefebvre and Hadjicostis, 2019), of the continuous-time Markov model with its logical observer is proposed to compute a probabilistic verifier. Given a sequence of past observations and an initial state, this probabilistic verifier can be used to estimate the probability that the system is attacked. The average attack detectability (as well as the mean detection delay) can be computed from its steady state distribution. Compared to previous works by some of the authors (Lefebvre and Hadjicostis, 2019,b), the contributions of this paper are twofold. From a methodological point of view, the paper extends the design of a probabilistic verification methodology for LCTMM and focuses on detection properties based on particular sequences of observations. From a practical point of view, it extends the opacity analysis (that does not include any model of the intruder actions) to cyber-attacks analysis where intruder actions are detailed.

The paper is organized as follows. Section II introduces probabilistic models of cyber attack with LCTMM. Section III details the design of probabilistic verifiers for LCTMM. Then, in Section IV, the probabilistic verifier is used for attack detection purposes and mean detectability analysis. Section V concludes the paper.

2. PROBABILISTIC MODELS OF ATTACKS

In this section we first provide some background on continuous-time Markov models. Then, we introduce labeled continuous-time Markov models. Finally, we show how such a model can be used to describe the behaviour of a plant affected by a certain number of attacks in the communication channel, which may alter the observations.

2.1 Continuous-time Markov models

A continuous-time Markov model is a continuous-time stochastic process $\{X(t), t \geq 0\}$ that has a countable number of states in its state space $S_P = \{1, 2, ...\}$ and that possesses the Markov property

$$prob(X(t) = j | X(s) = i, X(t_{n-1}) = i_{n-1}, ..., X(t_1) = i_1) \\ = prob(X(t) = j | X(s) = i),$$

where $0 \leq t_1 \leq ... \leq t_{n-1} \leq s \leq t$ is any nondecreasing sequence of times and $i_1, ..., i_{n-1}, i, j$ are states in the state space S_P (Karlin and Taylor, 2012; Gagniuc, 2017; Norris, 1997). Given the state of the process at time s, the probability distribution of the process at any time after s, namely t, is independent of the entire past of the process before a time s. The Markov property is a "forgetting" property, suggesting memorylessness in the distribution of the time a continuous-time Markov model spends at any state. A continuous-time stochastic process with a set of discrete states $\{X(t), t \geq 0\}$ is a Continuous-Time Markov Model (CTMM) if it satisfies the Markov property.

In addition, an CTMM with state space S_p is time homogeneous if the transition probabilities only depends on the difference $t - s$ between s and t and not on the actual times s and t. Then, for any $s \leq t$ and any states $i, j \in S_P$,

$$prob(X(t) = j | X(s) = i) = prob(X(t - s) = j | X(0) = i).$$

In the following, time homogeneous CTMM will be considered with a finite number $|S_P|$ of states. We further assume that each state $i \in S_P$ is associated with a set of n_i independent, exponential alarm clocks with rates $\mu_{i,j_1}, ..., \mu_{i,j_{n_i}}$, and $Post(i) = \{j_1, ..., j_{n_i}\}$ is the set of possible states the process may jump to when it leaves state i. The rates $\mu_{i,j_1}, ..., \mu_{i,j_{n_i}}$ are input parameters assumed to be known. When the process enters state i, the time t it spends at state i has an average value $d_i = (\mu_{i,j_1} + ... + \mu_{i,j_{n_i}})^{-1}$ and is exponentially distributed with the probability density function $(1/d_i) \times \exp(-t/d_i)$. The probabilities of going to state $j_k \in \{j_1, ..., j_{n_i}\}$ are given by $\mu_{i,j_k} \times d_i$.

The state probability functions, $\pi_{P,i}(t, \Pi_P(0))$ that the system is in state $i \in S_P$ at time t, when the vector of initial probabilities is $\Pi_P(0)$, form an $1 \times |S_P|$ vector $\Pi_P(t, \Pi_P(0))$ whose ith entry is $\pi_{P,i}(t, \Pi_P(0))$. Vector $\Pi_P(t)$ is defined by Eq. (1) as

$$\Pi_P(t, \Pi_P(0)) = \Pi_P(0) \times \exp(G_P \times t). \qquad (1)$$

The matrix G_P in Eq. (1) is the CTMM generator where the ith row has $(-d_i)^{-1}$ at the ith column (diagonal entry) and μ_{i,j_k} at the j_kth column. A CTMM can be formally defined as follows.

Definition 1. A CTMM is defined as $M_P = (S_P, G_P, \Pi_P(0))$ where S_P is a set of states, G_P is the generator matrix, and $\Pi_P(0)$ is the initial probability vector.

In general, it is impossible to know precisely the state of a given CTMM at a given time t, but one can evaluate the probability $\pi_{P,i}(t, \Pi_P(0))$ that the system is in state $i \in S_P$ at time t. The time $d_{i,j_k}, j_k \in Post(i)$, required to jump from i to j_k is exponentially distributed and satisfies Eq. (2) below

$$prob(d_{i,j_k} \leq t) = 1 - \exp(-\mu_{i,j_k} \times t) \qquad (2)$$

where t is any value of the time. When simulating an CTMM, the successor j^* of a given state i is randomly selected by computing first the times $d_{i,j_k}, j_k \in Post(i)$ with exponential probability density functions of the form (2) and then by searching for the successor j^* reached after a duration d_{i,j^*} that takes the minimal value over the set of times $d_{i,j_k}, j_k \in Post(i)$

$$j^* = \underset{j_k \in Post(i)}{\operatorname{argmin}} \{d_{i,j_k}\}.$$

2.2 Labeled CTMM

In this work, CTMM with partial observations of the process jumps are considered (namely labeled CTMM). To define formally such a labeled CTMM, let us introduce E as a finite set of events such that each jump from state $i \in S_P$ to state $j \in S_P$ is associated to the event $e_{i,j} \in E$. Then, consider the partition of E as $E = E_o \cup E_{uo}$, where E_o is the set of observable events and E_{uo} is the set of unobservable events, and introduce an output alphabet Q (i.e., a set of observed labels). The labeling function $Obs : E \to Q \cup \{\epsilon\}$ is defined such that for each $e \in E_o$, $Obs(e) \in Q$ and for each $e \in E_{uo}$, $Obs(e) = \epsilon$, where ϵ stands for the empty string. Essentially, Obs acts as an observation filter that associates at most one label $q \in Q$ to each event $e \in E$. The advantage of this model is to separate the events that drive the jump within the process states and the labels that result from observation.

Definition 2. A Labeled Continuous-Time Markov Model (LCT MM) is a pair (M_P, Obs) where $M_P = (S_P, G_P, \Pi_P(0))$ is an CTMM, E_o and E_{uo} are respectively the sets of observable and unobservable jumps, Q is a set of output labels and $Obs : E_o \cup E_{uo} \to Q \cup \{\epsilon\}$ is a labeling function.

2.3 Application to cyber security

In this section we consider a plant that is partially observed through a labeling function. The following assumptions are considered for simplicity:

(1) the attacks are not permanent,
(2) only one attack affects the system at a time,
(3) the plant is not attacked at time 0 and the initial state is assumed to be known.

We assume that the nominal behaviour of the plant when no attack is present is described by an LCTMM (M_N, Obs_N) with $M_N = (S_N, G_N, \Pi_N(0)))$. The plant can be subject to K different types of attack. The behaviour of the system under attack of type A_i (for $i = 1, \ldots, K$) can also be described by LCTMM (M_{A_i}, Obs_{A_i}) with $M_{A_i} = (S_{A_i}, G_{A_i}, \Pi_{A_i}(0))$ with $\Pi_{A_i}(0) = 0$ (Assumption 2). The LCTMM of the nominal mode and of the altered modes have similar generators (wih the exception of the diagonal terms that depend on the time parameters of the possible jumps to other modes) but the observation functions Obs_N and Obs_{A_i} are different. This model corresponds to the situation where the intruder just changes the observations without affecting the process itself. Assuming the switching between the nominal mode and the attack modes is also a Markovian process, the plant under attack can be described by an LCTMM (M_P, Obs) with $M_P = (S_P, G_P, \Pi_P(0))$, $S_P = S_N \cup S_{A_1} \cup ... \cup S_{A_K}$ and

$$G_P = \begin{pmatrix} G_N & G_{N,A_1} & \cdots & G_{N,A_K} \\ G_{A_1,N} & G_{A_1} & \ddots & G_{A_1,A_K} \\ \vdots & \ddots & \ddots & \vdots \\ G_{A_K,N} & G_{A_K,A_1} & \cdots & G_{A_K} \end{pmatrix}. \qquad (3)$$

The underlying graph described by G_P is strongly connected (Assumption 1). The submatrix G_{N,A_k} describes how the attack A_k starts from normal behaviour. On the contrary, the submatrix $G_{A_k,N}$ describes how the attack A_k ends, when the system returns to the normal behavior. The submatrix G_{A_k,A_m} describes how attack A_k switches to attack A_m. The attack starting, ending and switching correspond obviously to silent events. Other events may be silent or observable.

Example 1: Consider the example of a plant affected by two different attacks A_1, and A_2, depicted in Fig. 1. The set of states is $S_P = S_N \cup S_{A_1} \cup S_{A_2}$ with $S_N = \{1, 2, 3\}$, $S_{A_1} = \{4, 5, 6\}$ and $S_{A_2} = \{7, 8, 9\}$. The events that correspond to the starting or end of an attack are unobservable. The other events generate symbols in the alphabet $Q = \{a, b, c\}$. Attack A_1 permutes the labels and transforms a into b, b into c and c into a. Attack A_1 starts from state 1 and ends from state 5. Attack A_2 replaces a and c by b except the label a from 3 to state 2 that is replaced by c. Attack A_2 also starts from state 1 and ends from state 8. For simplicity, all events in this system are assumed to be exponentially distributed with time parameters that are equal to 1 (more generally, one can imagine that different rates are associated with different events). The initial state is assumed to be state 1. In Fig. 1, the labels resulting from the observation function and the time parameters are reported.

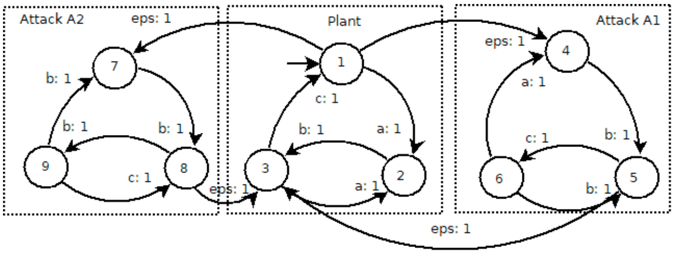

Fig. 1. LCTMM model of a plant and two attacks.

3. DESIGN OF OBSERVERS FOR LCTMM

In this section we present the logical observer for a given LCTMM.

3.1 Logical observer of an LCTMM

Making abstraction of timing aspects, an LCTMM can be viewed as a Labeled Deterministic Finite Automaton.

Definition 3. A Labeled Deterministic Finite Automaton (LD FA) is a pair (G, Obs). $G = (X, E, \delta, x_0)$ is a Deterministic Finite Automaton (DFA) where X is a finite set of states, E is an alphabet (finite set of events), $\delta : X \times E \to X$ is a transition function and $x_0 \in X$ is the initial state. $Obs : E \to Q \cup \{\varepsilon\}$ is a labeling function, where Q is a set of observable labels and ε represents the empty string.

Such an automaton is based on two primitives, namely states and transitions, and describes in a natural way the behavior of a dynamical system that evolves from state to state upon the occurrence of discrete events (Cassandras and Lafortune, 2008; Giua, 2017). Compared with the DFA, there are two nondeterministic primitives that are possible in an LDFA: (i) some transitions correspond to the occurrence of silent events; (ii) two or more transitions outgoing from one state can produce the same label.

To obtain a logical observer (LOB) of a given LCTMM, we consider the LCTMM as a LDFAwith a set of states S_P, a set of events E, a transition function δ defined for any $i, j \in S_P$, $e_{i,j} \in E$ by $\delta(i, e_{i,j}) = j$, and an initial state that corresponds to the LCTMM state i such that $\pi_{P,i}(0) = 1$ (see Assumption 3). The labeling functions of the LCTMM and the LDFA are the same. Consequently, ignoring the timing aspects of an LCTMM, a state observer $LOB = (X, Q, \delta_o, x_0)$ is obtained using standard methods (Giua, 2017): X is the set of observer states (which are subsets of S_P), Q is the alphabet of observed symbols, δ_o is the transition function of LOB, and x_0 is the observer initial state. Each state $x \in X$ of the observer is the subset of plant states (i.e., we can regard state x as a subset of S_P) that are consistent with the sequence of untimed observations seen thus far.

3.2 Product of an LCTMM with its logical observer

For probabilistic verification purposes, we are interested in the parallel product of the LCTMM generator G_P and its logical state observer LOB. The PV of a given LCTMM (M_P, Obs) with $M_P = (S_P, G_P, \Pi_P(0))$ is defined below.

Definition 4. Let (M_P, Obs) with $M_P = (S_P, G_P, \Pi_P(0))$ be an LCTMM. The Probabilistic Verifier (PV) of (M_P, Obs) is defined as the triplet $M = (S, G, \Pi(0))$ with

- $S = \{(i, x), i \in S_P, x \in X \text{ such that } i \in x\}$,
- $G(s, s') = G_P(i, i')$ for $s = (i, \bullet) \in S$ and $s' = (i', \bullet) \in S$, $s \neq s'$ and $G(s, s) = \sum_{s' \neq s} -G(s, s')$,
- $\Pi(0)$ is defined by $\pi_s(0) = \pi_{P,i}(0)$ for $s = (i, \bullet) \in S$.

Algorithm 1 is similar to the algorithm detailed in (Lefebvre and Hadjicostis, 2019), and can be used to design the PV. Each state $s \in S$ of the PV is a pair $s = (i, x)$ composed by the LCTMM state $i \in S_P$, and the observer state $x \in X$. If the PV is in state s it means that the state of the plant is i and the set of states that are consistent with the logical observation that led to i is x. S is composed of N states. Only states $s = (i, x_0)$ with $\pi_{P,i}(0) > 0$ have a non zero initial probability $\pi_s(0) = \pi_{P,i}(0)$.

Algorithm 1: PV design for LCTMM

```
Require: : $(M_P, Obs)$, $M_P = (S_P, G_P, \Pi_P(0))$, $E$
Ensure: : $M = (S, G, \Pi(0))$
 1: $UNXPL \leftarrow \emptyset$, $S \leftarrow \emptyset$
 2: compute $LOB = (X, Q, \delta_o, x_0)$ from $(M_P, Obs)$
 3: $s \leftarrow (1, x_0)$, $\pi_s(0) \leftarrow 1$
 4: $S \leftarrow S \cup \{s\}$, $UNXPL \leftarrow UNXPL \cup \{s\}$
 5: while $UNXPL \neq \emptyset$ do
 6:     select a state $s = (i, x)$ in $UNXPL$
 7:     for each $e_{i,i'}$ in $E$ do
 8:         if $Obs(e_{i,i'}) \neq \epsilon$ then
 9:             $x' \leftarrow \delta(x, Obs(e_{i,i'}))$
10:         else
11:             $x' \leftarrow x$
12:         end if
13:         $s' \leftarrow (i', x')$
14:         if $s'$ does not already exist in $S$ then
15:             $S \leftarrow S \cup \{s'\}$
16:             $UNXPL \leftarrow UNXPL \cup \{s'\}$
17:         end if
18:         $G(s, s') \leftarrow G_P(i, i')$
19:     end for
20:     $G(s, s) \leftarrow \sum_{s' \neq s} -G(s, s')$
21:     $UNXPL \leftarrow UNXPL \setminus \{s\}$
22: end while
```

The sets $UNXPL$ and S are initialized at lines 1-8. The main cycle (lines 9-26) explores the successive states in $UNXPL$. Each time a new state is found, it is added in sets S and $UNXPL$ (lines 19-20) and the generator matrix G is updated (lines 22 and 24). Once, all possible successors of a given state s have been found (lines 11-23), s is removed from set $UNXPL$ (line 25) so that $UNXPL$ tends to the empty set. Proposition 1 proves that the PV is a CTMM, and also characterizes the evolution of the state probability vector associated with the PV. In particular, it illustrates the relationship with the state probability vector of the plant.

Proposition 1: The PV $M = (S, G, \Pi(0))$ of a given LCTMM (M_P, Obs), with $M_P = (S_P, G_P, \Pi_P(0))$ is a CTMM and for all $i \in S_P$, it holds

$$\pi_{P,i}(t, \Pi_P(0)) = \sum\nolimits_{s=(i,\bullet)} \pi_s(t, \Pi(0)). \qquad (4)$$

Proof : $M = (S, G, \Pi(0))$ is an CTMM by construction: (i) the initial probability vector $\Pi(0)$ satisfies $\pi_s(0) = \pi_{P,i}(0)$ for $s = (i, x_0)$ and $\pi_s(0) = 0$ otherwise. Obviously, $\sum_s \pi_s(0) = 1$ and $\pi_{P,i}(0) = \sum_{s=(i,\bullet)} \pi_s(0)$; (ii) the matrix G is a generator matrix because $G(s,s) = \sum_{s' \neq s} -G(s,s')$. In addition, to obtain Eq. (4), let $\alpha_i(t, \Pi_P(0)) = \sum_{s=(i,\bullet)} \pi_s(t, \Pi(0))$, and compute its derivative

$$\begin{aligned}
\frac{d\alpha_i(t, \Pi_P(0))}{dt} &= \sum\nolimits_{s=(i,\bullet)} \frac{d\pi_s(t, \Pi(0))}{dt} \\
&= \sum\nolimits_{s=(i,\bullet)} \left(\sum\nolimits_{s' \in S} (\pi_{s'}(t, \Pi(0)) \times G(s', s)) \right) \\
&= \sum\nolimits_{j \in S_P} \left(\sum\nolimits_{s'=(j,\bullet)} \pi_{s'}(t, \Pi(0)) \right) \times G_P(j, i) \\
&= \sum\nolimits_{j \in S_P} (\alpha_j(t, \Pi_P(0)) \times G_P(j, i)).
\end{aligned}$$

Since $\alpha_i(t, \Pi_P(0))$ and $\pi_{P,i}(t, \Pi_P(0))$ have the same initial value and the same derivative, one can conclude that they are equal at any time t, thus Eq. (4) holds. □

To conclude, for any PV state $s = (i, x)$, $\pi_s(t, \Pi_P(0))$ is the probability that the plant is in state i and the LOB is in state x at time t. The advantage of the PV is to encode the dynamics and the observations of the LCTMM (M_P, Obs) in a single CTMM.

Example 2: Consider the LCTMM in Fig. 1. The LOB has 10 states and is reported in Fig. 2. Its initial state is $x_1 = \{1, 4, 7\}$. The PV, computed with Algorithm 1 has 20 states and is reported in Fig. 3. Its initial state is $(1, x_1)$ (with a probability that equals 1). State $(1, x_1)$ has three ouput arcs. The first one leads to state $(2, x_2)$. It corresponds to event a that in the model leads from the initial state 1 to state 2, while in the LOB leads from state x_1 to state x_2. The second output arc from state $(1, x_1)$ to state $(7, x_1)$ corresponds to the ε-transition going from state 1 to state 7 in the plant. Finally, the third output arc from state $(1, x_1)$ to state $(4, x_1)$ corresponds to the ε-transition going from state 1 to state 4 in the plant. Similarily, the other states of the POB can be explained.

4. PROBABILISTIC VERIFICATION

The PV is useful to evaluate the probability of a given plant property $\mathcal{A}$ that depends on the plant states. We assume that each state $i \in S_P$ may or may not satisfy the property $\mathcal{A}$ and this decision (to satisfy or not the property $\mathcal{A}$) is a logical decision as mentioned in the introduction. Thus, $\mathcal{A}$ may be defined as the subset A of states in S_P that satisfy the property.

4.1 Average detectability and mean detection delay

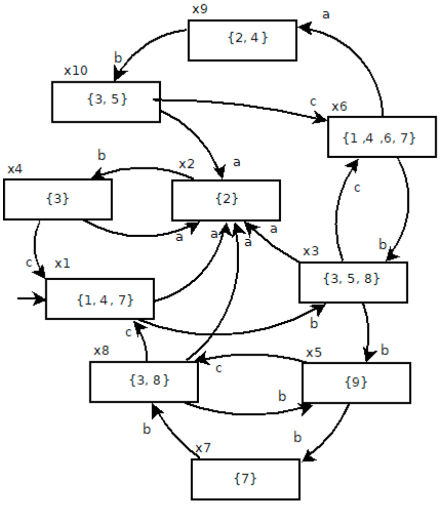

Fig. 2. LOB for the system in Fig. 1.

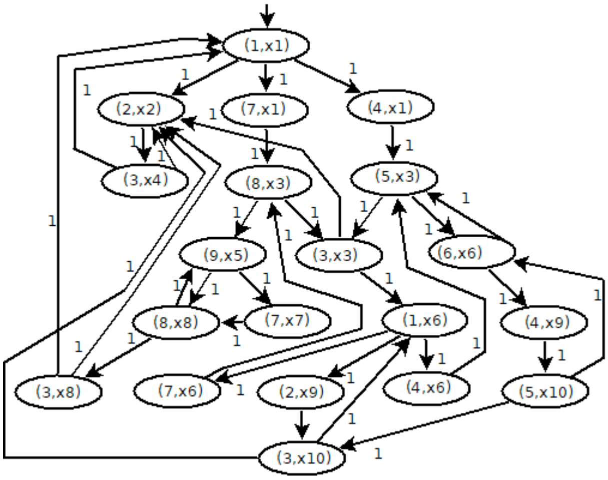

Fig. 3. PV for the system in Fig. 1.

Independently from any sequence of observations, the PV can be used to compute the *a priori* mean detectability of a given property $\mathcal{A}$. As long as the PV is composed by a transient and a single strongly connected component, the PV steady state $\Pi(\infty)$ does not depend on $\Pi(0)$ and is the unique solution of the following equations (Norris, 1997)

$$\begin{aligned}
&\Pi(\infty) \times G = (0)_{1 \times |S|}, \\
&\Pi(\infty) \times (1)_{|S|} = 1,
\end{aligned} \qquad (5)$$

where $(0)_{1 \times |\bullet|}$ is the row vector of size $|\bullet|$ with all entries equal to 0. The PV may be used to compute the probability that a given property $\mathcal{A}$ is detected on the average. For this purpose, let us introduce the sets $E(A)$ and $F(A)$

$$\begin{aligned}
E(A) &= \{s \in S \text{ such that } s = (i, x) \text{ and } i \in A\}, \\
F(A) &= \{s \in S \text{ such that } s = (i, x) \text{ and } x \subseteq A\}.
\end{aligned} \qquad (6)$$

Looking at the PV one can evaluate the average probability $prob(F(A)|\infty)$ that the property $\mathcal{A}$ is detected in the long run under the assumption that $\mathcal{A}$ is satisfied, namely the average detectability of $\mathcal{A}$,

$$prob(F(A)|\infty) = \frac{\Pi_{F(A)}(\infty) \times (1)_{|F(A)|}}{\Pi_{E(A)}(\infty) \times (1)_{|E(A)|}}, \qquad (7)$$

where $\Pi_{F(A)}(\infty)$ refers to the steady state probability vector of the states in the set $F(A)$.

In addition, the analysis of PV leads to the Mean Detection Delay of $\mathcal{A}$ ($MDDA$) that characterizes the time interval that is needed, on the average, to detect that property $\mathcal{A}$ is satisfied. Such a time interval is computed in a method similar to the one detailed in (Lefebvre and Hadjicostis, 2019b) and captured by

$$MDDA = \Pi(A) \times (-G(H(A), H(A)))^{-1} \times (1)_{|H(A)|}, \quad (8)$$

where $G(X, Y)$ denotes the square submatrix with rows X and columns Y extracted from G, $H(A) = \{s \in E(A) | s \notin F(A)\}$, $E(\bar{A}) = \{s \in S | s \notin E(A)\}$, and $\Pi(A)$ is the mean distribution when the PV enters a state in $E(A)$

$$\Pi(A) = \Pi_{E(\bar{A})}(\infty) \times \left(-G(E(\bar{A}), E(\bar{A}))\right)^{-1} \\ \times G(E(\bar{A}), E(A)) \times (\Pi_{E(\bar{A})}(\infty) \times (1)_{|E(\bar{A})|})^{-1}.$$

4.2 Probabilistic verification of attack detection

In this section we propose to evaluate the probability that the system is under attack based on the sequence of observations that is recorded, and to compute the average detectability of the attacks. For each attack A_k, $k = 1, ..., K$, we may define three sets:

- the set of states of S corresponding to the occurrence of attack A_k: $E(A_k) = \{s \in S \mid s = (i, x)$ and $i \in S_{A_k}\}$, where S_{A_k} is the set of states of the LCTMM relative to attack A_k,
- the set of states of S corresponding to the detection of attack A_k: $F(A_k) = \{s \in S \mid s = (i, x)$ and $x \subseteq S_{A_k}\}$,
- the set of states of S corresponding to the non-detection of attack A_k: $H(A_k) = \{s \in E(A_k) \mid s \notin F(A_k)\}$.

There is no difficulty to interpret $MDDA_k$ as the mean detection delay for attack A_k and compute it using Eq. (8).

Example 3: Continuing the previous example, we compute that the mean probability to detect A_1 is 0 whereas the probability to detect A_2 (under the assumption that the system is under attack A_2) is 0.33. The performance can be explained according to the observer that is capable of detecting A_2 but not A_1. The mean delay to detect A_2 is about 1.5 time units and is ∞ for A_1.

5. CONCLUSION AND FUTURE WORK

This paper has proposed a probabilistic verifier for LCTMM. This verifier benefits from the timed observations generated by the plant and from the exponential dynamics of the LCTMM probabilities. It is helpful to verify plant properties that are defined according to some subsets of plant states. The method has been applied to evaluate the *a priori* probability that an cyber-attack in a given system will be detected with certainty.

Future research directions for our work are twofold. From a practical point of view, we will be interested in establishing additional average indicators for cyber-security performance characterization. From a methodological point of view, we will study at first probabilistic decisions and probabilistic verification. For that purpose, a probabilistic observer that uses the timed observations to estimate the probability of states for labeled continuous-time Markov models will be designed in a formal way including a more general setting where the jump between two states may deliver several different labels.

REFERENCES

Cardenas, A. A., Amin, S., and Sastry, S. (2008). Secure control: Towards survivable cyber-physical systems, Proc. IEEE Int. Conf. on Distributed Computing Systems, pp. 495–500.

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

Cassandras, C. G. and Lafortune, S. (2008). *Introduction to Discrete Event Systems*, Springer.

Ding, D., Han, Q. H., Xiang, Y., Ge, X., and Zhang, X. M. (2018). A survey on security control and attack detection for industrial cyber-physical systems, *Neurocomputing*, vol. 275 pp.1674–1683.

Gagniuc, P. A. (2017). *Markov Chains: From Theory to Implementation and Experimentation*, USA, NJ: John Wiley and Sons.

Gao, C., Seatzu, C., Li, Z., and Giua, A. (2019). Multiple attacks detection on discrete event systems, Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, Bari, Italy.

Giua, A. (2017). Automata and supervisory control, Internal Report, Aix-Marseille University.

Hubballi, N., Biswas, S., Roopa, S., Ratti, R., and Nandi, S. (2011). LAN attack detection using discrete event systems, *ISA Transactions*, vol. 50, pp. 119–130.

Karlin, S. and Taylor, H. E. (2012). *A First Course in Stochastic Processes*, Academic Press.

Keroglou, C. and Hadjicostis, C. N. (2015). Detectability in stochastic discrete event systems, *Systems & Control Letters*, vol. 84, pp. 21–26.

Keroglou, C. and Hadjicostis, C. N. (2017). Verification of detectability in probabilistic finite automata, *Automatica*, vol. 86, pp. 192–198.

Lefebvre, D. and Hadjicostis, C. N. (2019). Exposure time as a measure of opacity in timed discrete event systems, Proc. IEEE European Control Conf., Naples, Italy.

Lefebvre, D. and Hadjicostis, C. N. (2019b). Revelation time for initial-state opacity measurement in timed discrete event systems, Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, Bari, Italy.

Lun, Y. Z., D'Innocenzo, A., Smarra, F., Malavolta, I., and Di Benedetto, M. D. (2019). State of the art of cyber-physical systems security: An automatic control perspective, *The Journal of Systems and Software*, vol. 149, pp. 174–216.

Norris, J. R. (1997). *Markov Chains*, Cambridge Press, pp. 60–125.

Rashidinejad, A., Lin, L. Y., Wetzels, B., Zhu, Y., Reniers, M., and Su, R. (2019). Supervisory control of discrete-Event systems under attacks: An overview and outlook, Proc. IEEE European Control Conf., Naples, Italy.

Teixeira, A., Pérez, D., Sandberg, H., and Johansson, K. H. (2012). Attack models and scenarios for networked control systems, Proc. Int. Conf. on High Confidence Networked Systems, Beijing, China.

Thorsley, D. and Teneketzis, D. (2005). Diagnosability of stochastic discrete-event systems, *IEEE Transactions on Automatic Control*, vol. 50, no. 4, pp. 476–492.

Thorsley, D. (2010). Diagnosability of stochastic chemical kinetic systems: A discrete event systems approach, Proc. American Control Conference, Baltimore, Maryland, USA.