Malicious Origin of Deadlocks in Flexible Manufacturing Systems

Amaury Beaudet, Éric Zamaï, Cédric Escudero, Emil Dumitrescu

Malicious Origin of Deadlocks in Flexible Manufacturing Systems

Amaury Beaudet * **Éric Zamaï** * **Cédric Escudero** *
Emil Dumitrescu *

* *Univ Lyon, INSA Lyon, Université Claude Bernard Lyon 1, Ecole Centrale de Lyon, CNRS, Ampère, UMR5005, 69621 Villeurbanne, France (email: amaury.beaudet@insa-lyon.fr)*

Abstract: Flexible Manufacturing Systems (FMSs) are designed to execute parallel processes simultaneously by using flexible resources (e.g. robots, re-configurable machines) alongside a supervisor allocating the resources to the different processes. In these systems, deadlocks are defined as blocking states where no resource allocation decision can be taken. They originate from resources characteristics (i.e. mutual exclusion condition, non-preemptive), processes interactions (i.e. shared resources, circular wait of resources) and an inappropriate sequence of allocation decisions. In modern FMSs, the use of open and interconnected control components for resource allocation control and productivity enhancement makes FMSs vulnerable to cyber-attacks. Hence, although FMSs are designed to deal with known natural deadlocks, new malicious ones can originate from sophisticated cyber-attacks. In this paper, malicious deadlocks are defined and contextualized regarding the existing literature on deadlocks management and on cyber-attacks targeting Discrete-Event Systems (DES). Then, a model representative of deadlock attacks is proposed based on the S^3PR theory and DES attacks modeling. Finally, the deadlock attack model is simulated and new malicious deadlocks are exhibited and discussed.

Keywords: Flexible Manufacturing Systems, Cyber-attacks, Deadlock, Petri-Net

1. INTRODUCTION

In the past decades, manufacturing systems became highly automated to increase their productivity and reliability (Kusiak (2018)). In these systems, each production cell is controlled by a Programmable Logic Controller (PLC) and cell machines by local controllers and regulators. All those components are interconnected together and with the production supervision, scheduling and management software through a dense industrial network. This increase in networked components and the connectivity of industrial networks to business networks made manufacturing systems vulnerable to cyber-attacks (Sicard et al. (2018); Beaudet et al. (2021)).

Among them, new attacks targeting the specific properties of control systems have emerged. In the literature (Escudero et al. (2018); Carvalho et al. (2018)), the introduced attacks target primarily the controlled physical system and have the objectives to degrade its behavior (Sicard et al. (2018)) or deteriorate the health of its physical components (Escudero et al. (2022)). These works have either a focus on production machines controlled with continuous signals including side-channel signals (Elhabashy et al. (2021)), and control signals (Escudero et al. (2022); Cárdenas et al. (2011)) or on production cells controlled through discrete events of actuators and sensors (Carvalho et al. (2018); Khoumsi (2019)). In this paper, a new attack objective is covered : the stoppage of the physical system, i.e. the production line. To reach this objective, the attack is assumed to target concurrently multiple cells and machines. If such an attack is successful, the consequences on the manufacturing system would be catastrophic. For instance, it could cause a decrease of the system productivity, a non-quality of ephemeral products (e.g. food, wafers), or a non-respect of clients delivery deadlines. In this work, the stoppage of the production system is achieved by the attacker through the malicious occurrence of deadlocks.

In an attack-free manufacturing system, deadlock states are reached when no resource allocation decision is eligible due to a circular wait condition between the different running processes (Li et al. (2012)). This condition appears in presence of flexible resources (e.g. robots, machines) able to execute a large diversity of processes by cooperating with each others. Manufacturing systems equipped with such resources are called Flexible Manufacturing Systems (FMSs). In Fig. 1 an example of FMS is illustrated (Kaid et al. (2020)). This system owns 6 resources (4 machines and 2 robots) and can execute 2 different processes, namely part A and part B. FMSs own by design natural deadlocks states. In the FMS from Fig. 1, a deadlock occurs for instance when Part A holds resource M2 and is requiring resource R1 while Part B holds R1 and requires M2. Both processes are waiting for a resource held by the other, creating a circular wait condition.

In the literature, natural deadlocks are coped with thanks to prevention, avoidance, and detection & recovery methods (Li et al. (2012)). These methods are implemented in a centralized manner into a supervisor connected through the industrial network to all the PLCs and controllers controlling the different resources (Uzam (2002); Du et al.

(2020)). Few references also propose a distributed implementation of these methods for avoidance (Du et al. (2020)) and prevention methods (Yang and Hu (2020)). Deadlock management methods are mainly static methods based either on a complete knowledge of natural deadlocks for the design of robust supervisors (prevention), or on a model of the FMS normal behavior (avoidance). Few authors propose also dynamic methods to deal with changes intrinsic to FMS such as resource failures (Du et al. (2020)), uncontrollable/ unobservable transitions (Huang et al. (2021)), process reconfiguration or addition/removal of machines (Kaid et al. (2020)). However, these methods seem to be unable to cope with cyber-attacks regarding two main limits. First, their centralized nature makes command and observation events transiting on the industrial network between PLCs and the supervision (hosting the methods) vulnerable to cyber-attacks. Furthermore, as the supervision, contains its owns monitoring model of the FMS, an attacker can misslead the supervision regarding the physical system real state and desynchronize them. In the case of distributed methods, events shared between the distributed supervisors are also vulnerable even if their number is reduced and limited to fewer control components. Second, static models used by the existing methods are not aware of malicious deadlocks during their design, whereas an attacker is able to manipulate and modify both the set of existing deadlocks and the FMS normal behavior. As for methods using dynamical models, they do not consider attacks as a source of behavioral changes in the FMS. To first answer these two limits, a definition of malicious deadlocks and a model on how such attacks can manipulate the behavior of the FMS physical system, communication and supervision to reach deadlock states are needed. To our best knowledge, no work from the literature proposes such contributions.

In this paper, we propose a theoretical model for deadlock attacks based on Petri net (PN) and an attack analysis using PN reachability graphs applied on vulnerable S^3PR. In Section 2, PN and S^3PR models will be recalled. In Section 3, deadlock attacks will be defined and the deadlock attack model will be detailed and discussed. In Section 4, simulation results of the attack model will be presented. Finally, a conclusion and future research perspectives will end this paper.

2. THEORETICAL BACKGROUND

In the literature, Petri nets are used to deal with deadlock management for both prevention and avoidance methods (Li et al. (2012)). In particular, a specific PN subclass has been designed to model System of Simple Sequential Processes with Resources, called S^3PR systems (Ezpeleta et al. (1995)). In this Section, mathematical expressions of PNs and S3PR sub-class will first be introduced.

2.1 Petri net

A Petri net is a five-tuple $PN = (P, T, F, W, M_0)$, where $P = \{p_1, p_2, p_3, \ldots, p_n\}$ is a finite set of places, with $n \in \mathbb{N}_+^*$, $T = \{t_1, t_2, t_3, \ldots, t_m\}$ is a finite set of transitions, with $m \in \mathbb{N}_+^*$, and with $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$. Also, $F \subseteq (P \times T) \cup (T \times P)$ is the set of all directed arcs, where arcs belonging to $(P \times T)$ connect a place to a transition and arcs in $(T \times P)$ connect a transition to a place. In addition, let $W : F \to \mathbb{N}$ be the weight function defined for every arc of F and $M_0 : P \to \mathbb{N}$ be the initial marking of N, associating every place $p_i \in P$, $\forall i \in \{1, n\}$, with an initial number of tokens $M_0(p_i)$. In the remaining of this paper, Petri nets with no specific initial marking will be noted as $N = (P, T, F, W)$. When associated with an initial marking M_0, a Petri net N will be referenced as (N, M_0). More generally, a marking M of a Petri net N is a function $M : P \to \mathbb{N}$ associating every place $p_i \in P$, $i \in \{1, n\}$, with its current marking $M(p_i) \in \mathbb{N}$. $M(p_i)$ represents the number of tokens included in p_i and the set of all $M(p_i)$, namely the marking M, depicts the current state of the system modeled with N.

For an element $x \in (P \cup T)$, the preset of x is denoted ${}^\bullet x$, and its postset $x^\bullet$, where ${}^\bullet x = \{y \in (P \cup T) | (y, x) \in F\}$ and $x^\bullet = \{y \in (P \cup T) | (x, y) \in F\}$. For a place p_i, $i \in \{1, n\}$, the set of input transitions is denoted ${}^\bullet p_i$, and the set of output transitions $p_i^\bullet$. In the same way, for a transition t_j, $j \in \{1, m\}$, the set of input places is denoted ${}^\bullet t_j$, and the set of output places $t_j^\bullet$. A transition t_j is said to be firable or enabled if each input place $p_i \in {}^\bullet t_j$ is marked with at least $W(p_i, t_j)$ tokens. The firing of a transition t_j follows two steps. First, its removes $W(p_i, t_j)$ tokens from each input place $p_i \in {}^\bullet t_j$ and then it adds $W(t_j, p_i)$ tokens to each output places $p_i \in t_j^\bullet$.This firing process drives the Petri net N from a marking M to a new marking M' and is noted $M[t_j > M'$. If a marking M'' can be reached from a marking M by firing a sequence of transitions $\sigma = [t_1, t_2, ..., tk]$, with $k \in \{1, m\}$, this firing process is noted $M[\sigma > M''$; marking M'' is said to be reachable from marking M. Based on the definition of place marking, if $M[t_j > M'$ with $t_j \in T$, we have, for every places $p_i \in N$, $M'(p_i) = M(p_i) + W(t_j, p_i) - W(p_i, t_j)$. Given a Petri net N, the set of all reachable markings from a marking M is denoted as $\mathcal{R}(N, M) = \{M' \mid \exists\ \sigma = \{t_0, t_1, t_2, ..., t_k\},\ k \in \{1, m\}$ and $M[\sigma > M'\}$. From $\mathcal{R}(N, M)$, the reachability graph $\mathcal{RG}(N, M)$ is defined as follow. $\mathcal{RG}$ states are the markings of $\mathcal{R}$ and every $\mathcal{RG}$ directed arc between two states M and M' represents the firing of $t_j \in T$ such that $M[t_j > M'$. We denote an arc as (M, M')

Petri nets own specific properties that will be used in the next Sections. A Petri net is said to be pure if it has no self-loops, where a selfloop is a couple (p, t) such that $t \in {}^\bullet p$ and $t \in p^\bullet$. In a pure Petri net, we define the Incidence

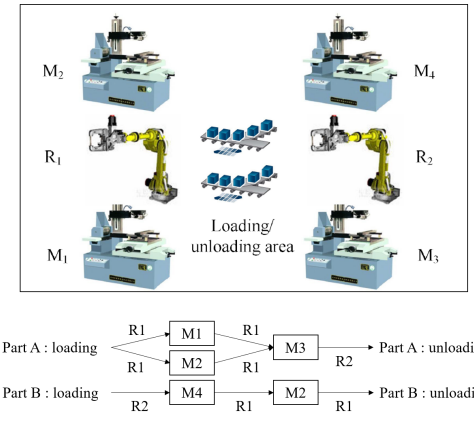

Fig. 1. FMS with 4 machines, 2 robots and 2 processes

matrix I of dimensions $(n \times m)$ as $I(i,j) = W(t_j, p_i) - W(p_i, t_j), \forall i \in \{1,n\}, \forall j \in \{1,m\}$. In a Petri net (N, M_0), a place p is called k-bounded if for every $M \in \mathcal{R}(N, M_0)$, $M(p) \leqslant k$. A Petri net (N, M_0) is said to be k-bounded if for every $p_i \in P$, p_i is k-bounded. A transition t_j is said to be live if for any $M \in \mathcal{R}(N, M_0)$, there exists a sequence σ of firable transitions from M that contains t_i. A Petri net N is considered to be live if all its transitions are live. A deadlock or dead marking is a marking $M \in \mathcal{R}(N, M_0)$ from which no transition can be fired. We define as pre-deadlock or pre-dead marking, a markings that leads irremediably to deadlocks. Petri net liveness ensures the absence of deadlocks and pre-deadlocks since from every $M \in \mathcal{R}(N, M_0)$, it is always possible to fire any transition t_i following some firing sequence.

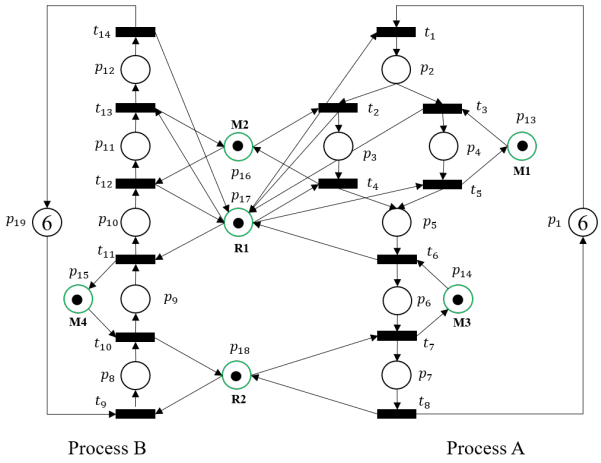

Fig. 2. S^{3}PR model of example from Fig. 1

2.2 S^{3}PR models

S^{3}PR models are a sub-class of Petri nets used to model a wide range of manufacturing systems. In this section, S^{3}PRs will be introduced step by step, by first defining two sub-classes of PN S^{3}PRs (Ezpeleta et al. (1995)).

A Simple Sequential Process (S^{2}P) is a Petri net $N = (P_A \cup p^0, T, F)$ where P_A are the activity or operation places, $p^0 \notin P_A$ is the process idle place, T is a set of transitions and F is the set of directed arcs. N is a strongly connected state machine (every two markings of N can be connected through a sequence of transitions) and every circuit of N contains P^0 (a circuit is a simple cycle of places in N). A S^{2}P model depicts a manufacturing process and its successive operations.

An S^{2}P with resources (S^{2}PR) is a Petri net $N = (P_A \cup P^0 \cup P_R, T, F, W)$, where P_A are the activity or operation places and $p^0 \notin P_A$ is the process idle place. P_R are the resource places symbolizing the resources availability where $P_R \neq \emptyset$ and $P_R \cap (P_A \cup P^0) = \emptyset$. An S^{2}PR has the following properties :

(1) The subnet created from $X = P_A \cup P^0 \cup T$ is a S^{2}P ;
(2) $\forall p \in P_A, \forall t \in {}^\bullet p,\ \forall t' \in p^\bullet, \exists\ r_p \in P_R$ such that ${}^\bullet p \cap P_R = p^\bullet \cap P_R = r_p$;
(3) $\forall r \in P_R, {}^{\bullet\bullet}r \cap P_A = r^{\bullet\bullet} \cap P_A \neq \emptyset$;
(4) $\forall r \in P_R, {}^\bullet r \cap r^\bullet = \emptyset$;
(5) ${}^{\bullet\bullet}(p^0) \cap P_R = (p^0)^{\bullet\bullet} \cap P_R = \emptyset$;

An S^{2}PR models a production sequence by pairing each operation of a process S^{2}P with the required resources. Each operation place is now bound with at least one resource place. In an S^{2}PR $N = (P_A \cup P^0 \cup P_R, T, F, W)$, an initial marking is called an acceptable marking iff :

(1) $M_0(p^0) \geqslant 1$; The process is idle and can be launched once or more times simultaneously.
(2) $M_0(p) = 0, \forall p \in P_A$; No operation is running initially.
(3) $M_r(p) \geqslant 1, \forall r \in P_R$; All resources are available initially and have a capacity superior or equal to 1.

An S^{3}PR refers to a "System of S^{2}PR" and can therefore be built recursively from two or more S^{2}PR models. An S^{3}PR is defined as follow :

(1) An S^{2}PR is an S^{3}PR;
(2) Let $N_i = (P_{A_i} \cup P_i^0 \cup P_R, T_i, F_i, W_i)$, $i \in 1,2$, be two S^{3}PR, satisfying $P_{R_1} \cap P_{R_2} = P_C \neq \emptyset$, $(P_{A_1} \cup p_1^0) \cap (P_{A_2} \cup p_2^0) = \emptyset$, $T_1 \cap T_2 = \emptyset$. N_1 and N_2 can be combined into $N = (P_A \cap p^0, T, F, W)$ through the shared resources P_C, and N is still an S^{3}PR model, defined as $P_A = P_{A_1} \cup P_{A_2}$, $p^0 = p_1^0 \cup p_2^0$, $T = T_1 \cup T_2$, $F = F_1 \cup F_2$, and $W = W_1 \cup W_2$.

This means two distinct S^{3}PRs are composable if they share a set of common resources and the resulting is a combination of the two original ones. In an S^{3}PR $N = (P_A \cup P^0 \cup P_R, T, F, W)$, an initial marking M_0 is called an acceptable marking iff :

(1) (N, M_0) is an acceptable marking regarding S^{2}PR conditions.
(2) $N = N_1 \circ N_2$, so that (N_i, M_{0_i}) is an acceptably marked S^{3}PR and :
   (a) $\forall i \in \{1,2\}, \forall p \in P_i \cup P_i^0,\ M_0(p) = M_{0_i}(p)$
   (b) $\forall i \in \{1,2\}, \forall r \in P_{R_i} \setminus P_C,\ M_0(r) = M_{0_i}(r)$
   (c) $\forall r \in P_C,\ M_0(r) = max\{M_{0_1}(r), M_{0_2}(r)\}$

Fig. 2 exhibits the S^{3}PR model of the example from Fig. 1.

In this section, S^{3}PRs were presented theoretically based on the PN theory and the FMS characteristics. In the next section, deadlock attacks will be defined and a model will be proposed based on PN and S^{3}PR theory.

3. DEADLOCK ATTACKS

3.1 Context and Definition

A cyber-attack reaches its objective by exploiting vulnerabilities in the targeted control system. A vulnerability is defined as a vulnerable component or communication whose digital behavior can be modified by an attacker to reach its objective. In FMSs, vulnerabilities exploitable for deadlock attacks can be identified by analyzing jointly the control architecture and the attack objective, namely the malicious occurrence of deadlocks. The closed-loop architecture for resource allocation control in FMSs is schematized in Fig. 3 and runs as follow. A centralized supervisor S is in charge of the resource allocation control for a physical system G (a production line) comprising all the production resources and processes. S receives discrete observation events from the PLCs regarding the evolution of G, updates its control model from these events, and takes consequently a resource allocation decision for the

running processes that translates into discrete command events sent to the resources through the PLCs. We denote by PL the PLCs Layer of the architecture. In the literature, an S^{3}PR modeling an FMS represents with its marking the state of G by giving the states of the running processes and the resources availability. Furthermore, in this model, the eligible allocation decisions are modeled with the firable transitions. This model is denoted N_G. Observation and command events are called the control events of S. We denote as E_{in}, the set of inputs events e_{in} of S and as E_{out} the set of outputs events e_{out} of S.

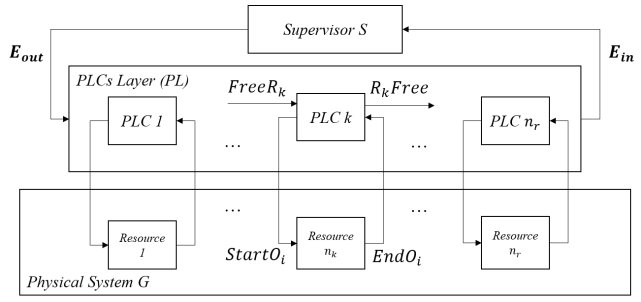

Fig. 3. Closed-loop control in FMS

In this paper, the resources, their local controllers and the closed-loop communication between the PLCs and the controllers are considered non-vulnerable. Indeed, these lower layers of the FMS architecture are outside the closed-loop dedicated to resource allocation control and responsible for the occurrence of deadlocks. The vulnerabilities of these components are assessed in other references (Escudero et al. (2022); Elhabashy et al. (2021); Cárdenas et al. (2011)). As for the PLCs, the supervisor S and the closed-loop communication between them, they are considered as vulnerable for the execution of deadlock attacks.

Within the FMS closed-loop control, the vulnerabilities materialize themselves through the vulnerable control events an attacker can act on by inserting or deleting them ((Carvalho et al. (2018); Khoumsi (2019)). Therefore, FMS resource allocation control events, responsible for the occurrence of deadlocks, need to be further characterized to define what a deadlock attack is. First, supervisor inputs events in E_{in} are the events $R_{k+1}Free$, with $k \in \{1, |P_R|\}$ indicating a required resource R_{k+1} is free, and $EndO_j$, with $j \in \{1, |P_A|\}$, denoting an operation O_j executed by resource R_k ends, and meaning the corresponding process requires now R_{k+1} to realize O_{j+1}. Second, S output events in E_{out} are the events $FreeR_k$, allocating the free status R_kFree to resource R_k as O_j ends, and $StartO_{j+1}$, commanding the start of the operation O_{j+1} by R_{k+1}. We have for resource allocation control in FMSs : $E_{in} = \{R_kFree, \forall\ p_k \in P_R\} \cup \{EndO_j, \forall\ p_j \in P_A\}$ and $E_{out} = \{FreeR_k, \forall\ p_k \in P_R\} \cup \{StartO_j, \forall\ p_j \in P_A\}$. In N_G, the different events can be schematized by the input/output arcs of $t \in T$ since they are the inputs/outputs of the decision represented by the firing of t.

In the work of (Carvalho et al. (2018)), four types of attacks are considered against observation and command events in Discrete Events Systems (DESs). These four attack types can be summarized as follows. First, when an attacker intercepts an event on a communication between the supervisor and the physical system, he has the choice to delete it or let it go through. We call it an Event Deletion Attack (**EDA**). Second, whenever an attacker has access to an existing event via the legitimate communication or storage of this event, it can insert it freely. We call it an Event Insertion Attack (**EIA**). In FMSs, the attacker has the potential to launch EDA and EIA on each of the four events. The singularity of our work is the exploitation of EDA and EIA by the attacker to drive an FMS into deadlock states rather than, as seen in the litterature, to make the physical system reach critical states.

Finally, as exposed in the literature (Carvalho et al. (2018); Khoumsi (2019)), EIA on command events and command events emitted by a compromised supervisor will affect the state of G, whereas EDA and EIA on events monitored by S will desynchronize S state monitoring with G true state. In consequence, two different entities, G and S, might be compromised by the attack. In FMSs, this paradigm stays also true. Therefore, two types of malicious deadlocks can be defined depending on their origin :

(1) **Physical Deadlocks**. They originate from G. They are the outcome of inconsistent resource allocation commands and can be natural.
(2) **Supervision Deadlocks**. They originate from S. They are either consequences of missing events for S to take decisions or of a desynchronization with G creating a circular wait of events between S and G.

In this section, the context of deadlocks attacks on FMSs has been defined regarding the related works from the literature. This allows us to give the following definition.

Definition 1. (Deadlock Attack). A deadlock attack is a sequence of malicious EDAs and EIAs on resource allocation control events that brings either the physical system or the supervisor into a deadlock marking. Let e^a be an attacked event, where $e \in E_{in} \cup E_{out}$. A deadlock attack is a sequence $\sigma = \{e^a_1, e^a_2, ..., e^a_{K_{atck}}\}$ of K_{atck} malicious events driving the FMS modeled with an S^{3}PR from a live marking M to a deadlock marking M_{dead}.

3.2 Deadlock attack model

In our work, we propose to model deadlock attacks with S^{3}PR and PNs models for 3 reasons. First, it brings a coherence between the FMS model used for resource allocation control and the attack model. Then, as an attacker can modify simultaneously different control events, PNs are able to model this parallelism. Finally, PNs own liveness properties relevant for the analysis of deadlocks.

An attack targeting an FMS will modify maliciously events related to the resource allocation control. In N_G, the S^{3}PR model of the physical system G, we showcased in section 3.1 that these events are related to transitions inputs and outputs arcs. Thus, we define an attacked S^{3}PR as a Petri net $N_{atck} = (P_A \cup P^0 \cup P_R, T_{free} \cup T_{atck}, F, W)$, where T_{atck} is the set of attacked transitions and T_{free} the set of attack-free transitions, with $T = T_{free} \cup\ T_{atck}$. A transition t is said to be attacked if at least one of its related events is vulnerable to an EIA or an EDA. However, in S^{3}PR models, the vulnerable events are not clearly represented. In this paper, we propose a PN submodel for each $t \in T_{atck}$ to answer this concern.

Let $N_t = (P_t, T_t, F_t, W_t)$ be the PN model representing the attack free transition t. This PN aims at modeling the control loop and the events between G and S responsible for G change of state through the firing of transition t in N_G, i.e. the decision making from S. Thereby, we define by $P_{in} = {}^\bullet t$ the set of entry places of t, $P_{out} = t^\bullet$ the set of output places of t, and P_{cl} the set of places modeling the closed-loop communication of the different control events between S, the PLCs and G following the control architecture in Fig. 3. We have $P_t = P_{in} \cup P_{out} \cup P_{cl}$.

The proposed model N_t includes the following steps of the closed-loop control :

(1) S is in idle mode and waits for the events in E_{in}.
(2) Occurrence in G and PL of the events in E_{in}.
(3) Transmission of events in E_{in} from G and PL to S.
(4) All events in E_{in} have reached S. An allocation decision is taken and events in E_{out} are generated.
(5) Transmission of events in E_{out} from S to PL and G.
(6) The events in E_{out} are executed by G and PL.

In N_t, each closed-loop control step is represented either by a transition $t \in T_t$ when an event occurs or is executed by G, PL or S and by a place $p \in P_{cl}$ when it is an event passive step (idle mode, communication). In Fig. 4, N_t is illustrated based on the definition given above. The steps $(1), (3), (5)$ are defined as places $\{p_3, p_4, p_5, p_6, p_7, p_8\}$ and the steps $(2), (4), (6)$ as transitions $\{t_1, t_2, t_3, t_4, t_5\}$.

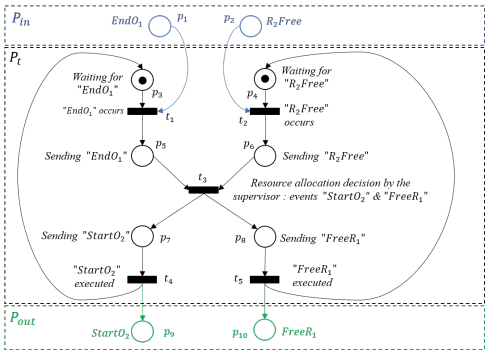

Fig. 4. PN model N_t of an attack-free transition in a S^{3}PR

For a transition $t_j \in T$, N_{t_j} is integrated into N_G by merging places from N_t and N_G as follows : $EndO_1 = p_1 \in P_{in}$ is merged with $p \in {}^\bullet t_j \cap P_A$, $R_2Free = p_2 \in P_{in}$ with $p \in {}^\bullet t_j \cap P_R$, $StartO_2 = p_9 \in P_{out}$ with $p \in t_j^\bullet \cap P_A$ and $FreeR_1 = p_{10} \in P_{out}$ with $p \in t_j^\bullet \cap P_R$.

Let $N_{t_{atck}} = (P_{t_{atck}}, T_{t_{atck}}, F_{t_{atck}}, W_{t_{atck}})$ be the PN model of t_{atck}. $N_{t_{atck}}$ is a modified version of N_t with new places and transitions to model both EDA and EIA. As EDA and EIA can target the four events within t, they are modeled in a general case, then replicated in $N_{t_{atck}}$ for each event. An EDA results from the attacker *choice* to delete the event rather than letting it through (Khoumsi (2019)). In $N_{t_{atck}}$, a place modeling the attacker *choice* is added after the transition representing the event occurrence. From this *choice* place, two paths, deleting or letting the event through, are modeled with two transitions. In the deletion path, the token is redirected to the initial *wait* place as the event e never occurred for S when $e \in E_{in}$ and is considered as executed by G when $e \in E_{out}$. In the second path, the token is directed to the normal *sending* place.

An EIA aims at inserting an unexpected event into the closed-loop communications. In N_t, an event occurrence is modeled by the firing of the transition preceding a *sending* place. Inserting an event can be modeled by faking the firing of this transition, i.e. inserting a token into the following *sending* place. To do so, an "attack" place p_{att} is added and is connected to the *sending* place through a new transition. From the definitions above, replicated to all events, $N_{t_{atck}}$ is displayed in Fig. 5.

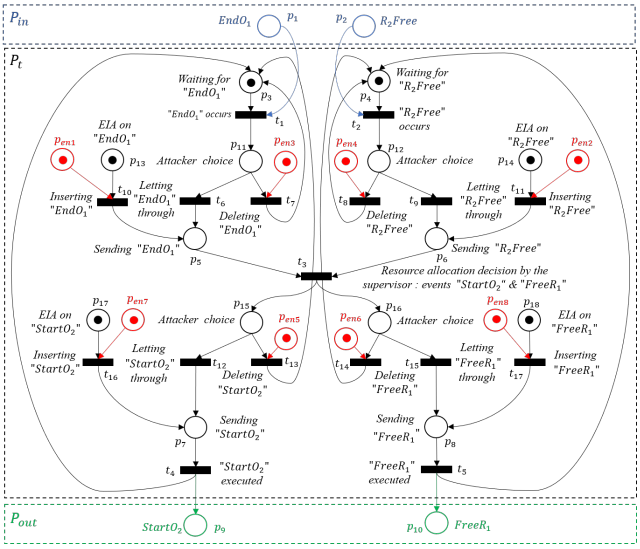

Fig. 5. PN model $N_{t_{atck}}$ of an attacked transition in a S^{3}PR

In Fig. 5, p_{en_i} with $i \in \{1, 8\}$ are places enabling or disabling the corresponding EIA or EDA. An attack is enabled when $M(p_{en_i}) = 1$, meaning the corresponding event is vulnerable to such attack. We denote by P_{en} the set of all p_{en_i}. The attacker capacity is then described by the vector $C_{atck} = [M(p_{en_1}), ..., M(p_{en_8})]$. The places and transitions added to model EIA and EDA on the four events are gathered into P_{att} and T_{att} respectively. In consequence, we have $P_{t_{atck}} = P_t \cup P_{att} \cup P_{en}$ and $T_{t_{atck}} = T_t \cup T_{att}$. The initial marking of $N_{t_{atck}}$ is $M_0(p_3) = M_0(p_4) = 1$ (*waiting* places) and $M_0(p_{13}) = M_0(p_{14}) = M_0(p_{17}) = M_0(p_{18}) = 1$ (EIA places). The initial marking of C_{atck} depends on the skills given to the attacker.

Once the transition t is fired, $N_{t_{atck}}$ is reinitialized. Indeed, in S^{3}PR , t can not be fired twice simultaneously and after being fired, t returns into idle mode. Furthermore, if an attacker is able to realize an EIA or EDA on t once, it can repeat it indefinitely. EIA/EDA and P_{en_i} places own a single token since events have only one occurrence in $N_{t_{atck}}$ and thereby, might be deleted or inserted only once.

In the next section of this paper, an exploration of N_{atck} reachable markings will be conducted using $N_{t_{atck}}$ when a vulnerable transition is fired. This exploration will help analyze the malicious deadlocks states and the malicious routes to deadlocks generated by EIAs and EDAs.

4. SIMULATION AND RESULTS

In this section, the model $N_{G_{atck}}$ will be simulated to study deadlocks resulting from EIA and EDA. As $N_{G_{atck}}$ is the attacked model from N_G, the simulation will exclusively showcase attacks driving the physical system G into

Table 1. Simulation Results of Attacks 1 and 2

Process	Vulnerable transitions	Number of markings	Attack 1			Number of markings	Attack 2		
			DDs	Pre-DDs	Pre-DDs of natural DDs		DDs	Pre-DDs	Pre-DDs of natural DDs
Attack-free		282	16	61	61	282	16	61	61
A	t2	470	37 (21)	159 (98)	0	5656	95 (79)	277 (227)	11
	t3	467	39 (23)	162 (101)	0	4278	82 (66)	209 (169)	4
	t7	710	80 (64)	255 (194)	0	1475	98 (82)	326 (270)	210
	t2/ t3	799	73 (57)	352 (291)	0	11706	183 (167)	360 (331)	4
	t2/t3/t7	1999	339 (323)	1056 (995)	0	59918	1074 (1058)	3137 (3109)	107
B	t10	601	89 (73)	237 (176)	0	1464	111 (95)	324 (263)	112
	t12	464	37 (21)	153(92)	0	5672	94 (78)	193 (155)	0
	t10/t12	991	200 (184)	421 (360)	0	29065	624 (608)	1732 (1694)	74
A +B	t7/t10	1209	158 (142)	492 (431)	0	3027	194 (178)	634 (578)	520
	t12/t2/t3	854	83 (67)	397 (336)	0	11858	187 (171)	170 (151)	1

Table 2. Simulation Results of Attack 3

Vulnerable resource	Number of markings	Attack 3		
		DDs	Pre-DDs	Pre-DDs of natural DDs
Attack-free	282	16	61	61
M2	985	68(52)	12 (12)	6
R2	1713	130 (114)	64 (58)	6
R1	9985	330 (314)	66 (66)	0

deadlocks state, i.e. into physical deadlocks. Supervision deadlocks are not examined in this paper.

The simulation of our attack model relies on the construction of the reachability graph $\mathcal{RG}(N_{G_{atck}}, M_0)$ where FMS deadlocks are identified from $\mathcal{RG}$ dead markings. In this paper, N_G markings represent the states of G and therefore, the exploration of $\mathcal{RG}(N_{G_{atck}}, M_0)$ needs to respect physical constraints inherent to G. First, an activity cannot be launched twice simultaneously and a resource has a fixed capacity of 1; i.e. $\forall p \in (P_A \cup P_R) \cap P_{atck},\ M(p) < 2$. Second, a resource $r \in P_R$ is able to realize a single operation at a time, meaning $\forall r \in (P_R \cap P_{atck}), \forall M \in \mathcal{RG}$, for $P_{A_r} = {}^{\bullet\bullet}r \cap r^{\bullet\bullet} \cap P_A,\ M(\{r, P_{A_r}\}) \leq M_0(r) = 1$. Finally, the number of allowed concurrent processes must be bounded and the full production capacity of a process must be reachable, i.e. $\forall p \in p^0, \forall M \in \mathcal{RG}, M(p) + M(\{p_1, ..., p_k\}) = M_0(p)$, with $\{p_1, ..., p_k\}$ the activity places of the process related to p. The exploration algorithm of $\mathcal{RG}(N_{G_{atck}}, M_0)$ is described in Algorithm 1 and has a complexity $O(|\mathcal{R}(N_{G_{atck}})|^*|T|^*|\mathcal{R}(N_{t_{atck}})|)$.

Require. The attacker capacities $C_{atck}(t)$ are defined for each transition t. The values in $C_{atck}(t)$ are chosen regarding the criteria assigned to the attacker (e.g. the types of attacks (EDA or EIA) or the targeted processes). **Running**. The exploration of $\mathcal{RG}(N_{G_{atck}}, M_0)$ starts from M_0 and ends when all reachable markings are visited. For a marking $M \in \mathcal{RG}$ that has not been explored yet, every firable transition $t \in T$ aims to be fired. If $t \in T_{free}$, I is used to obtain M'. Else, if $t \in T_{atck}$, $N_{t_{atck}}$ is used by calling the function $FireT_{atck}$. This function needs as inputs $C_{atck}(t)$ and the marking of P_{in} and P_{out} and returns as output ${\mathcal{R}_t}^{red} = \{M_t(P_{in} \cup P_{out}), \forall M_t \in \mathcal{R}_t\}$, with $\mathcal{R}_t$ the reachable markings set of $N_{t_{atck}}$, after exploring $N_{t_{atck}}$. A new marking is further explored if no convergence condition is encountered. **Convergence conditions** ($CvCond$). Three main constraints guarantee the convergence of the algorithm. First, the exploration is interrupted when a reached marking already belongs to $\mathcal{RG}(N_{G_{atck}}, M_0)$. Second,the physical constraints of G provides $N_{G_{atck}}$ the propriety of boundness. Third, the exploration is interrupted when a marking $M \in \mathcal{RG}$ is a dead marking for N_G. M is then considered as a dead marking for $N_{G_{atck}}$ since the attacker reached its objective of creating a deadlock within the existing FMS. **Ensure**. The outputs of the algorithm aim at helping identifying deadlock states, routes to reach these deadlocks and if malicious, the corresponding deadlock attacks. Consequently, the algorithm returns $\mathcal{RG}(N_{G_{atck}})$. This algorithm is ran on the example from Fig. 1 and Fig. 2. The following attacks were simulated.

Attack 1 targets the physical system with EIAs on $StartO_i$ events. It translates into $C_{atck}(t) = [00000010]$. We simulated this attack for $\{t_2, t_3, t_7, t_{10}, t_{12}\}$.

Attack 2 targets the events in E_{in} to misguide the supervisor control. It translates into $C_{atck}(t) = [11110000]$. We simulated this attack for $\{t_2, t_3, t_7, t_{10}, t_{12}\}$.

Attack 3 targets a specific resource with EIAs and EDAs on all events related to this resource. For instance for M_2, we define $C_{atck}(t_2) = C_{atck}(t_{12}) = [01011010]$ and $C_{atck}(t_4) = C_{atck}(t_{13}) = [10100101]$, meaning all events related to the control of M_2 are vulnerable. We also simulated this attack for R_1 and R_2.

Algorithm 1 Exploration algorithm of $\mathcal{RG}(N_{G_{atck}}, M_0)$

Require: $\forall t, C_{atck}(t)$, I of N_G, $M_0(N_G)$
Ensure: $\mathcal{RG}(N_{G_{atck}})$
  Add M_0 *to* $\mathcal{RG}(N_{G_{atck}})$
  $S_{ToExplo} \leftarrow M_0$
  $CV_{cond} \leftarrow \{DeadMarking \cup PhysicalConstraints\}$
  while $S_{ToExplo} \neq \emptyset$ **do**
    $s_{explo} \leftarrow S_{ToExplo}(1)$
    for $t_j \in T$ **do**
      if $C_{atck}(t) \neq 0^8$ **then**
        $NewStates \leftarrow FireT_{atck}(t_j)$
      else if $N = 0^8$ **then**
        $NewStates \leftarrow M' = M + I(j)$ $(M[t_j > M')$
      end if
      if $(NewStates \notin \mathcal{RG}(N_{G_{atck}})) \wedge (CV_{cond}$ are respected) **then**
        Add $NewStates$ *to* $\mathcal{RG}(N_{G_{atck}})$
        Add $NewStates$ *to* $S_{ToExplo}$
      end if
    end for
    Remove $S_{ToExplo}$ *from* s_{explo}
  end while

The results of the simulation of the 3 attacks are gathered in Tab. 1 and Tab. 2. For each attack, the number of generated deadlocks (DDs) and pre-deadlocks (Pre-DDs) markings is given. The value under parenthesis gives the number of new markings compared to the attack-free case. Finally, the last column gives the number of new pre-deadlocks markings leading to existing natural deadlocks.

From the simulation results, the following conclusions can be drawn. First, every attack makes G reach new physical deadlocks and pre-deadlocks markings (Tab. 1 and 2). This confirms the existence of malicious deadlocks as the values under parenthesis indicates the number of new reachable dead markings. Then, attacks 2 and 3 are able to generate new pre-deadlocks markings for existing natural deadlocks as shown by the columns "Pre-DDs of natural DDs" from Tab. 1 and 2. Therefore, even if natural deadlocks are coped with thanks to deadlocks management methods, they can still be reached using malicious routes through new pre-deadlocks states. Third, with attack 2, it was shown that an attacker can create physical deadlocks by deceiving the supervisor S. Only events monitored by S are manipulated with the aim to generate unfortunate command events driving G into a deadlock state. This attack does not cause supervisor deadlocks, yet it showcases the possibility to create deadlocks from the supervisor manipulation. However, these results and their analysis still have some limits. First, the different attacks have not been compared between each other yet. It would allow to identify deadlocks specific to one or more types of attack. Second, not all attacks have been tested due to an expensive computation time. Third, the different routes to a deadlock marking and the corresponding malicious events sequences have not been studied. Finally, supervision deadlocks are not taken into account explicitly in the attack model and the simulation. These limits will be answered in our future works.

5. CONCLUSION

In this paper, we have introduced a new type of attack called deadlock attacks. The latter aims at blocking an FMS by creating circular wait conditions. We have defined two types of malicious deadlocks: physical and supervision deadlocks. For the first type, a model has been proposed based on the S^3PR and PN theory. The simulation of this model using PN reachability graphs has provided answers to the limits exposed in introduction. First, the existence of malicious physical deadlocks has been proven. In addition, new malicious pre-deadlocks states leading to existing natural deadlocks have been highlighted. Second, the vulnerability of existing deadlock management methods to cyber-attacks has been revealed. Indeed, the attacker, by manipulating resource allocation control events between these methods (hosted by the supervisor) and the physical system, is able to create new malicious deadlocks. Third, attacks able to create deadlocks by only deceiving the supervisor have been demonstrated to be effective. However, this attack model does not consider yet supervision deadlocks into its design. In our future works, this limit will be addressed and a new attack model integrating both physical and supervision deadlocks will be proposed. To that end, this model will integer directly EDAs and EIAs into an S^3PR extension containing both a model of the FMS true state and a model of the FMS state monitored by the supervisor. In this new model, prevention methods based on implementing deadlocks monitors will be added and simulated. The feature of stealthiness will also be included. Finally, our last research axis will be the development of a cyber-security solution able to detect and characterize the malevolence of a deadlock.

REFERENCES

Beaudet, A., Escudero, C., and Éric Zamaï (2021). Malicious anomaly detection approaches robustness in manufacturing icss. *IFAC-PapersOnLine*, 54(1), 146–151. 17th IFAC Symposium INCOM 2021.

Cárdenas, A.A., Amin, S., Lin, Z.S., Huang, Y.L., Huang, C.Y., and Sastry, S. (2011). Attacks against process control systems: Risk assessment, detection, and response. In *Proc. of the 6th ASIACCS*, 355–366.

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

Du, N., Hu, H., and Zhou, M. (2020). Robust deadlock avoidance and control of automated manufacturing systems with assembly operations using petri nets. *IEEE Trans. Autom. Sci. Eng.*, 17(4), 1961–1975.

Elhabashy, A.E., Dastoorian, R., Wells, L.J., and Camelio, J.A. (2021). Random sampling strategies for multivariate statistical process control to detect cyber-physical manufacturing attacks. *Qual. Eng.*, 33(2), 300–317.

Escudero, C., Massioni, P., Zamaï, E., and Raison, B. (2022). Analysis, prevention, and feasibility assessment of stealthy ageing attacks on dynamical systems. *IET Control. Theory Appl.*, 16(4), 381–397.

Escudero, C., Sicard, F., and Zamai, E. (2018). Process-aware model based idss for industrial control systems cybersecurity: Approaches, limits and further research. In *IEEE 23rd ETFA*, volume 1, 605–612.

Ezpeleta, J., Colom, J., and Martinez, J. (1995). A petri net based deadlock prevention policy for flexible manufacturing systems. *IEEE Trans. Rob. Autom.*, 11(2), 173–184.

Huang, B., Zhou, M., Wang, C., Abusorrah, A., and Al-Turki, Y. (2021). Deadlock-free supervisor design for robotic manufacturing cells with uncontrollable and unobservable events. *IEEE/CAA J. Autom. Sin.*, 8(3).

Kaid, H., Al-Ahmari, A., Li, Z., and Davidrajuh, R. (2020). Automatic supervisory controller for deadlock control in reconfigurable manufacturing systems with dynamic changes. *Appl. Sci.*, 10(15).

Khoumsi, A. (2019). Sensor and actuator attacks of cyber-physical systems: A study based on supervisory control of discrete event systems. In *2019 8th ICSC*, 176–182.

Kusiak, A. (2018). Smart manufacturing. *Int. J. Prod. Res.*, 56(1-2), 508–517.

Li, Z., Wu, N., and Zhou, M. (2012). Deadlock control of automated manufacturing systems based on petri nets—a literature review. *IEEE Trans. Syst. Man Cybern.: Syst.*, 42(4), 437–462.

Sicard, F., Escudero, C., Zamaï, E., and Flaus, J.M. (2018). From ics attacks' analysis to the s.a.f.e. approach: Implementation of filters based on behavioral models and critical state distance for ics cybersecurity. In *2nd CSNet*, 1–8.

Uzam, M. (2002). An Optimal Deadlock Prevention Policy for Flexible Manufacturing Systems Using Petri Net Models with Resources and the Theory of Regions. *Int. J. Adv. Manuf*, 19(3), 192–208.

Yang, Y. and Hu, H. (2020). A distributed control approach to automated manufacturing systems with complex routes and operations using petri nets. *IEEE Trans. Syst. Man Cybern.: Syst.*, 50(10), 3670–3684.