



Prise en compte des risques de cyber-attaques dans le domaine de la sécurité des systèmes cyber-physiques : proposition de mécanismes de détection à base de modèles comportementaux

Franck Sicard

► To cite this version:

Franck Sicard. Prise en compte des risques de cyber-attaques dans le domaine de la sécurité des systèmes cyber-physiques : proposition de mécanismes de détection à base de modèles comportementaux. Informatique. Université Grenoble Alpes, 2018. Français. NNT : 2018GREAT080 . tel-02090972

HAL Id: tel-02090972

<https://theses.hal.science/tel-02090972>

Submitted on 5 Apr 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

Pour obtenir le grade de

**DOCTEUR DE LA COMMUNAUTE UNIVERSITE
GRENOBLE ALPES**

Spécialité : **AUTOMATIQUE PRODUCTIQUE**

Arrêté ministériel : 25 mai 2016

Présentée par

Franck SICARD

Thèse dirigée par **Éric ZAMAI** et
codirigée par **Jean-Marie FLAUS**

préparée au sein du **Laboratoire G-SCOP**
dans l'**École Doctorale EEATS**

Prise en compte des risques de cyber-attaques dans le domaine de la sécurité des systèmes cyber-physiques : Proposition de mécanismes de détection à base de modèles comportementaux

Thèse soutenue publiquement le **11 Octobre 2018**
devant le jury composé de :

M. Nouredine HADJSAID

Professeur à l'Univ. Grenoble Alpes (Grenoble INP), Président du jury

M. Benoit IUNG

Professeur à l'Université de Lorraine, Rapporteur

M. François PERES

Professeur à l'Université de Toulouse (Toulouse INP ENIT), Rapporteur

M. Florent FRIZON DE LAMOTTE

Maître de Conférences à l'Université de Bretagne Sud (ENSIBS), Examinateur

M. François PEBAY-PEYROULA

Directeur de programme R&D à ATOS Worldgrid, Examinateur

M. Michel CUCIUC

Ingénieur à DGA Maîtrise de l'Information, Co-Encadrant

M. Jean-Marie FLAUS

Professeur à l'Univ. Grenoble Alpes (UGA), Co-Directeur de thèse

M. Éric ZAMAI

Maître de Conférences à l'Univ. Grenoble Alpes (Grenoble INP),
Directeur de thèse



Remerciements

Difficile exercice qu'est celui de l'écriture des remerciements d'un manuscrit de thèse. Premiers (parfois les seuls) lus et pourtant derniers rédigés, ils marquent vraiment la fin d'une aventure. Merci à tous d'y avoir contribué, en m'ayant supporté, encouragé et soutenu durant ces trois années de travail. Je tâcherai de n'oublier personne...

Le travail présenté dans ce mémoire a été effectué au Laboratoire des Sciences pour la Conception, l'Optimisation et la Production (G-SCOP) dans l'équipe Gestion et Conduite des Systèmes de Production (GCSP). A ce titre, je tiens à remercier tout d'abord Monsieur François Villeneuve, directeur du Laboratoire G-SCOP, de m'avoir accueilli et permis d'effectuer mes travaux de recherche tout au long de ces trois années de thèse.

J'exprime également toute ma gratitude à la Direction Générale de l'Armement (DGA) Maîtrise de l'Information, financeur de ces travaux de thèse. En particulier, je remercie Monsieur Éric Bornette, qui a permis de rendre possible cette collaboration, Monsieur Michel Cuciuc, pour ses disponibilités et son aide dans l'accompagnement de ces travaux et Monsieur Frédéric Majorczyk pour ses conseils utiles dans le déroulement de ces travaux.

Je tiens à adresser mes remerciements à mon directeur de thèse, Monsieur Éric Zamaï, Maître de Conférences HDR à l'Institut National Polytechnique de Grenoble, et co-directeur de thèse Monsieur Jean-Marie Flaus, Professeur à l'Université de Grenoble, pour avoir accepté de m'encadrer et de diriger ces travaux de recherche. Leurs soutiens et leurs confiances sans failles ainsi que leurs disponibilités ont contribué non seulement à la qualité de ce travail mais également à mon évolution personnelle et professionnelle. Cela a été un plaisir d'avoir pu travailler avec vous et pour cela je vous dis « merci ».

Je continuerai en remerciant les membres de mon jury. Tout d'abord, Monsieur Nouredine Hadjsaid, Professeur à l'Institut National Polytechnique de Grenoble pour avoir accepté de présider mon jury. J'exprime également ma gratitude à Messieurs Benoit Iung, Professeur à l'Université de Lorraine, et François Peres, Professeur à l'Université de Toulouse, pour m'avoir fait le privilège d'être rapporteurs de ma thèse et surtout pour leurs critiques constructives. Enfin, je souhaiterais aussi remercier Messieurs Florent Frizon De Lamotte, Maître de Conférences à l'Université de Bretagne Sud, et François Pebay-Peyroula, Directeur de programme R&D à Atos Worldgrid, pour avoir évalué scientifiquement mon travail de recherche et pour avoir pris part au jury. Vos remarques et vos conseils ont contribué à parfaire la version finale de ce manuscrit.

Je souhaite également exprimer mes remerciements à l'ensemble du personnel des services informatique et administratif du laboratoire G-SCOP pour avoir participé à mon intégration au sein du laboratoire, pour la gestion des missions et pour l'organisation de ma soutenance de thèse. Pour tout cela, merci à Myriam, Marie-Josèphe, Fadila, Amandine, Christine, Souad, Kévin, Olivier... Je remercie également l'AIP PRIMECA Dauphiné Savoie pour la mise à disposition de la plateforme ainsi que Monsieur Duong Quoc Bao responsable technique de cette plateforme pour ses conseils et sa disponibilité.

Je voudrais aussi remercier mes collègues, qui au fil du temps, sont devenus des amis pour leurs aides, leurs amitiés, leurs précieux conseils et pour les discussions qui ont largement contribué à l'écriture de ce mémoire. Je souhaiterais notamment dire merci à Alexandre, Matthieu, Parisa et Tom qui m'ont accompagné à l'ADOC, Lucie, Grigori, Nicolas, Clément, Cédric, Yahya pour les bons moments passés à la Kfet. Merci également à mes collègues du label Recherche, Entreprise et Innovation (REI) et d'InnovDoc en particulier ceux avec qui j'ai pu travailler au pôle web comme Alexandre, Dominik, Hélène ou encore Corado et Stéphanie.

Je poursuivrai avec un grand merci à tous ceux avec qui j'ai eu la chance de travailler au cours de ces travaux Michel Tollenaere (Label REI), Franck Pourroy (Colloque AIP) et Gülgün Alpan (Fête de la sciences).

Je souhaite également remercier mes amis et coéquipiers de l'équipe de volley (Sam, Cédric, Tovo, Loïc, Etienne, Valentin, Benjamin, Florian) qui ont eu à supporter les hauts et les bas, mes amis présents depuis l'école d'ingénieur (Thomas, Chelsea, Nicolas, Sophie, Charles, Solenne, Valentin, Marion, François, Gautier, Florent, Elliot, Guillaume, Sophie, ...). Aussi, au moment de conclure cette thèse, je souhaite également remercier mes anciens collègues de chez Automatique & Industrie, chez qui j'ai travaillé en la qualité d'ingénieur développement, un grand merci à eux. Enfin, merci à tous ceux qui ont par leurs présences, leurs aides et leurs soutiens permis la réalisation de ces travaux.

Enfin, je dédie ce paragraphe à ma famille qui a su me supporter et me réconforter dans cette épreuve. Merci à Gaëlla qui a su me comprendre, me conseiller et m'encourager pour me permettre de réaliser et de finaliser ce travail, ma belle-famille pour leurs soutiens et leurs confiances, mes parents, ma tante, ma sœur et ma famille pour leurs amours et leurs dévouements indéfectibles.

Vous m'avez apporté force et courage pour achever cette thèse et poursuivre vers de nouvelles aventures. Que ce travail et ce manuscrit soient le gage de mon affection.

Franck

Les systèmes de contrôle-commande industriels (*Industrial Control System*, ICS) sont des infrastructures constituées par un ensemble de calculateurs industriels reliés en réseau et permettant de contrôler un système physique. Ils assurent le pilotage de réseaux électriques (Smart Grid), de systèmes de production, de transports, de santé ou encore de systèmes d'armes. Pensés avant tout pour assurer productivité et respect de la mission dans un environnement non malveillant, les ICSs sont, depuis le 21^{ème} siècle, de plus en plus vulnérables aux attaques (Stuxnet, Industroyer, Triton, ...) notamment avec l'arrivée de l'industrie 4.0. De nombreuses études ont contribué à sécuriser les ICSs avec des approches issues du domaine de la sécurité (cryptographie, IDS, etc...) mais celles-ci ne tiennent pas compte du comportement du système physique et donc des conséquences de l'acte de malveillance en lui-même. Ainsi, une sécurisation se limitant exclusivement à l'analyse des informations qui transitent sur un réseau industriel n'est pas suffisante. Notre approche amène un changement de paradigme dans les mécanismes de détection en y intégrant la modélisation du comportement du système cyber-physique.

Cette thèse propose des mécanismes de détection d'attaques en se positionnant au plus proche du système physique. Ils analysent les données échangées entre le système de contrôle-commande et le système physique, et filtrent les échanges au travers de modèles déterministes qui représentent le comportement du système physique soumis à des lois de commande. A cet effet, une méthodologie de conception est proposée dans laquelle l'ensemble des ordres est identifié afin de détecter les attaques brutales. Pour faire face aux autres attaques, en particulier celles plus sournoises, comme les attaques par séquences, nous proposons une stratégie de détection complémentaire permettant d'estimer l'occurrence d'une attaque avant que ses conséquences ne soient destructives. A cet effet, nous avons développé des concepts de distance d'un état caractérisé comme critique auquel nous avons adjoint un second mécanisme dit de trajectoire dans le temps permettant de caractériser une intention de nuire.

L'approche proposée hybride ainsi deux techniques orientées sécurité (sonde IDS) et sûreté (approche filtre) pour proposer une stratégie de détection basée sur quatre mécanismes liés :

- A la détection de contexte : basé sur l'état courant de l'ICS, un ordre émis par l'API peut être bloqué s'il conduit vers un état critique (attaque brutale).
- Aux contraintes combinatoires (attaque par séquences) : vérifiées par les concepts de distance et de trajectoire (évolution de la distance).
- Aux contraintes temporelles (attaque temporelle) : vérifiées par des fenêtres temporelles sur l'apparition d'événements et d'indicateurs surveillant la durée moyenne d'exécution.
- Aux sur-sollicitations basées sur un indicateur surveillant les commandes envoyées afin de prévenir un vieillissement prématuré (attaque sur les équipements).

L'approche proposée a été appliquée sur différents exemples de simulation et sur une plateforme industrielle réelle où la stratégie de détection a montré son efficacité face à différents profils d'attaquants.

Mots-clés : Détection à base de modèles, Surveillance, Contrôle-Commande, Systèmes à événements discrets, Cybersécurité, Systèmes Cyber-Physiques

Industrial Control Systems (ICSs) are infrastructures composed by several industrial devices connected to a network and used to control a physical system. They control electrical power grid (Smart Grid), production systems (e.g. chemical and manufacturing industries), transport (e.g. trains, aircrafts and autonomous vehicles), health and weapon systems. Designed to ensure productivity and respect safety in a non-malicious environment, the ICSs are, since the 21st century, increasingly vulnerable to attacks (e.g. Stuxnet, Industroyer, Triton) especially with the emergence of the industry 4.0. Several studies contributed to secure the ICS with approaches from the security field (e.g. cryptography, IDS) which do not take into account the behavior of the physical system and therefore the consequences of the malicious act. Thus, a security approach limited exclusively to the analysis of information exchanged by industrial network is not sufficient. Our approach creates a paradigm shift in detection mechanisms by integrating the behavioral modeling of the cyber-physical system.

This thesis proposes detection mechanisms of attacks by locating detection closer to physical system. They analyze the data exchanged between the control system and the physical system, and filter the exchanges through deterministic models that represent the behavior of the physical system controlled by control laws. For this purpose, a design methodology has been proposed in which all actions are identified in order to instantly detect brutal attacks. To deal with other attacks, especially the more sneaky, such as sequential attacks, we propose a complementary detection strategy to estimate the occurrence of an attack before its consequences are destructive. To this end, we have developed the concepts of distance of a state identified as critical to which we have added a second mechanism called trajectory which leads to a temporal notion that characterize an intention to harm.

As part of this thesis, the proposed approach combines two techniques oriented security (IDS probe) and safety (filter approach) to propose a detection strategy based on four mechanisms related to:

- Context detection: based on the current state of the system, an order sent by the PLC can be blocked by the control filter if it leads to a critical state (brutal attack).
- Combinatorial constraints (sequential attack): verified by the concepts of distance (risk indicator for the current state) and trajectory (indicator of the intention to harm by studying the evolution of the distance on a sequence).
- Temporal constraints (temporal attack): verified by time windows on the appearance of events and an indicator monitoring the average duration of execution.
- Over-solicitation monitoring mechanism: based on an indicator monitoring orders sent to the actuators to prevent premature ageing of the production equipment (attack on the equipment).

The proposed approach has been applied to various simulation examples and an industrial platform where the detection strategy has shown its effectiveness against different scenarios corresponding to attacker profiles.

Keywords: Model-based Detection, System Monitoring, Industrial Control Systems, Discrete-Event Systems, Cybersecurity, Cyber-Physical Systems.

Table des matières

Introduction générale.....	9
 Partie 1 – Cadre général de l'étude.....	 13
Chapitre 1 – Contexte de l'étude.....	15
1. Malveillance dans les environnements industriels.....	17
2. Cadre d'application de la cybersécurité : la Loi de Programmation Militaire (LPM) ..	28
3. Les systèmes industriels.....	30
4. Environnement d'un Industrial Control System (ICS).....	37
5. Étude des vulnérabilités d'un ICS	39
6. Bilan du chapitre.....	44
 Chapitre 2 – Etat de l'art des approches de surveillance et de détection d'attaques dans les ICSs	 45
1. Cybersécurité dans la littérature	46
2. Intrusion Detection System (IDS) dans les ICSs.....	48
3. Étude des IDS orientés connaissance du procédé	52
4. Étude des approches orientées sûreté de fonctionnement.....	57
5. Bilan des approches Sûreté/Sécurité	62
6. Bilan du chapitre.....	63
 Chapitre 3 – Problématique, positionnement et principe de l'approche proposée	 65
1. Spécificités des ICSs par rapport au SI « traditionnel ».....	66
2. Positionnement de l'approche proposée.....	68
3. Types d'attaques considérés	69
4. Approche S.A.F.E.: Security Approach based on Filter Execution	71
5. Hypothèses de travail et verrous scientifiques.....	75
6. Bilan du chapitre.....	77
 Partie 2 – Approche S.A.F.E. : Modélisation et Exploitation.....	 79
Chapitre 4 – Méthodologie de conception des filtres de commande.....	81
1. Méthodologie générale de l'approche S.A.F.E.	83
2. Étape 1 : Identification des paramètres du système	84
3. Étape 2 : Génération du modèle de filtre de commande.....	90
4. Étape 3 : Exploitation (en ligne)	97
5. Bilan du chapitre.....	97

Chapitre 5 – Indicateurs de la sécurité d’un ICS.....	99
1. Concept de distance au sens de Carcano.....	100
2. Adaptation pour les SED : calcul du plus court chemin aux états critiques	102
3. Indicateur de distance : révélateur d’une présomption de risque.....	105
4. Indicateur de trajectoire : révélateur de l’intention de nuire.....	106
5. Indicateurs de temporalité.....	109
6. Indicateur de sur-sollicitation des équipements	110
7. Bilan du chapitre.....	112
 Chapitre 6 – Protection des ICSs : présentation des mécanismes de détection d’anomalies .	113
1. Détection de couples Etat/Action	115
2. Détection d’anomalies par contraintes combinatoires.....	116
3. Détection d’anomalies par contraintes temporelles.....	124
4. Détection de la dégradation des équipements de production.....	127
5. Récapitulatif en fonction des attaques.....	129
6. Bilan du chapitre.....	130
 Partie 3 – Application de l’approche S.A.F.E.	131
Chapitre 7 – Exemples de simulation	133
1. Généralités.....	135
2. Présentation des systèmes et Application de l’approche S.A.F.E.....	138
3. Analyse des scénarii d’attaques	147
4. Tableau récapitulatif	155
5. Bilan du chapitre.....	156
 Chapitre 8 – Expérimentations sur plateforme industrielle.....	157
1. Présentation de la plateforme de test	158
2. Mise en œuvre de l’approche sur la plateforme	160
3. Analyse des résultats	162
4. Bilan du chapitre.....	166
 Conclusion générale.....	167
Références.....	170
Glossaire	188
Table des figures et des tableaux	189
Liste des publications.....	193
Annexes.....	195

Introduction Générale

Depuis les années 2000, les systèmes de contrôle-commande industriels (*Industrial Control System*, ICS) sont au cœur d'une révolution technologique numérique centrée essentiellement sur les nouveaux moyens de communication permettant d'améliorer encore la productivité des entreprises. Il s'agit de l'*Industrie 4.0*. Cette révolution s'est appuyée sur de nouveaux catalogues industriels, proposant des gammes de calculateurs de plus en plus performants, permettant d'intégrer des fonctions logicielles toujours plus importantes et présentant nativement des capacités d'interconnexions avec l'ensemble des calculateurs de l'usine. Cependant, cette introduction massive et rapide de telles nouvelles technologies issues de l'informatique (Information Technology) s'est malheureusement accompagnée d'un ensemble de vulnérabilités qui n'ont pas été prises en compte assez tôt.

Ainsi, les systèmes industriels intégrant ces nouvelles technologies numériques ont commencé à se voir attaqués : d'abord par ricochet, puis progressivement par des attaques les ciblant spécifiquement. Les conséquences de telles malveillances sur des systèmes industriels sont importantes, voire catastrophiques. En effet, contrairement à un système informatique, un système industriel relie l'informatique à la physique ; autrement dit, toute prise de contrôle de l'informatique peut avoir un effet sur le système physique. De nombreuses attaques sont à déplorer depuis une dizaine d'années, et leurs conséquences sont parfois critiques comme pour Stuxnet (attaque sur des centrifugeuses), BlackEnergy et Industroyer (attaques sur un réseau électrique) ou encore Triton (attaque sur une gamme d'automates de sécurité).

Une réelle prise de conscience de ces nouveaux risques industriels s'est opérée tant au niveau des états par la création d'organismes tels que l'ANSSI (Agence Nationale de la Sécurité des Systèmes d'Information, France), la NIST (National Institute of Standards and Technology, USA) ou encore l'ENISA (European Union Agency for Network and Information Security, UE) ou la mise en place d'un cadre législatif (LPM), que de la communauté scientifique, notamment informatique.

Ainsi, de nombreuses approches ont contribué à traiter ce problème avec une vision IT. Elles ont permis de sécuriser les niveaux informatiques des ICSs, avec des moyens et des technologies informatiques tels que le chiffrement, le développement de pare-feu spécifiques, des techniques d'intelligence artificielle pour apprendre des profils de trames réseaux, etc.

Cependant, force est de constater qu'elles ne vont pas étudier le sens même des informations qui sont échangées, informations qui ne peuvent être interprétées qu'à la lumière du comportement physique qu'elles représentent. Pour cela, un point de vue automatique doit être développé, s'appuyant sur des mécanismes de détection à base de modèles de commande et de parties opératives.

Le travail que nous présentons dans ce manuscrit propose d'apporter sa contribution au domaine de la Surveillance d'actes de malveillance dans les systèmes de contrôle-commande à événements discrets. L'étude de la bibliographie nous a permis d'identifier et de localiser les

vulnérabilités affectant les parties commande et opérative afin de proposer un placement original d'un système de détection à base de modèles déterministes entre les organes de commandes (automates programmables industriels) et les capteurs et actionneurs. Cette localisation au plus proche de la physique permet non seulement de comprendre le sens même de l'attaque mais aussi d'agir en tant que dernier rempart face à une attaque ayant traversé l'ensemble des barrières informatiques implantées dans l'architecture numérique de contrôle-commande. Cette approche fonde sa proposition sur les approches filtres développées à l'origine par les laboratoires CRAN et LAGIS, respectivement par Pascal Lhoste (Lhoste 1991) et Didier Cruette (Cruette, Bourey, et Gentina 1991). Notre apport principal se situe dans la proposition d'un nouveau modèle de filtre de commande adapté à la problématique d'analyse de l'intention de nuire de l'ordre émis par un automate programmable industriel vers des actionneurs. Ce modèle s'appuie sur une modélisation comportementale de la commande et de la partie opérative. Sur la base du modèle de filtre de commande proposé, que nous générons automatiquement à partir des modèles de commandes et du modèle de la partie opérative, le mécanisme de détection évalue en temps réel plusieurs indicateurs permettant de caractériser une attaque. Le premier de ces indicateurs représente la distance qui sépare l'état courant du système de chacun des états critiques identifiés. Le deuxième permet d'étudier l'évolution de cette distance dans le temps, nous l'appelons « trajectoire ». Le troisième indicateur permet d'évaluer le risque d'attaque en fonction du temps, et le dernier cible plus particulièrement les attaques dites de sur-sollicitations des actionneurs. L'ensemble de ces indicateurs permet d'identifier les principaux types d'attaques décrits dans la littérature. Notre approche a été évaluée sur des exemples académiques et sur une plateforme représentant un système de production réel.

Ce manuscrit s'articulera autour de 3 parties sur la base des thèmes suivants :

La première partie présente de manière générale le cadre de l'étude ainsi que la problématique. Après avoir décrit et défini les concepts de malveillance dans le domaine des systèmes industriels, nous proposons au lecteur une présentation des attaques principales qui ont été menées au cours de la dernière décennie à l'encontre des systèmes de contrôle-commande industriels (ICSs en anglais) en veillant d'une part à les caractériser, et d'autre part à mettre en lumière les vulnérabilités exploitées. Sur cette base, nous décrivons ensuite les organismes que l'État Français a créé ainsi que les lois auxquelles les entreprises sont désormais soumises pour faire face aux nouveaux enjeux de la cyber-criminalité visant les sites industriels. Clairement positionnés au cœur des mécanismes de détection d'attaques cyber de ces systèmes industriels, nous présentons ensuite un état de l'art du domaine afin de mettre en exergue les approches et techniques à retenir pour orienter notre contribution. Sur cette base, nous exposons notre approche générale et proposons une hybridation de solutions orientées sécurité (NIDS avec connaissance du procédé) et sûreté (approche filtre) afin de tirer avantage de ces deux visions.

La deuxième partie est entièrement consacrée à la présentation de notre approche tant sur le plan méthodologique qu'algorithmique. Ainsi, dans un premier temps nous proposons notre

méthode pour générer un nouveau modèle de filtre de commande servant de base pour les mécanismes de détection. La démarche proposée consiste, à partir des zones critiques à protéger et des paramètres relatifs, à produire un modèle comportemental sur la base des modèles de commande et de parties opératives. Ce modèle, nouveau et spécifique à la problématique de la cybersécurité des ICSs, s'appuie sur cinq types d'états à partir desquels quatre indicateurs d'attaques sont proposés tels que celui de la distance ou encore de trajectoire vers un état critique. Sur cette base, plusieurs algorithmes sont proposés, qu'ils soient de génération de modèles ou d'exploitation à des fins de détection.

Dans la troisième et dernière partie de ce mémoire, nous développons plusieurs exemples d'applications basés sur des cas d'étude académiques et sur une plateforme réelle représentative d'un système de production industriel. Après avoir présenté ces différents cas d'étude et en quoi ils permettent de valider les concepts proposés, nous présentons les différents scénarii d'attaques qui seront réalisés sur ces différents systèmes. Une fois appliqués, nous présentons les comportements de nos algorithmes afin de tester et d'évaluer les capacités de détection des mécanismes développés.

Partie 1 :

Cadre général de l'étude

Chapitre 1

Contexte de l'étude

Ce premier chapitre a pour objectif de positionner le contexte général dans lequel se déroule notre étude. Ainsi, au sein de la première section nous définirons ce qu'est une malveillance et nous étudierons les principales attaques ayant affecté des systèmes industriels. Nous positionnerons ensuite nos travaux par rapport au guide des bonnes pratiques édité par l'Agence Nationale de Sécurité des Systèmes d'Information (ANSSI). La deuxième section replace le cadre actuel d'application de la cybersécurité en s'appuyant notamment sur la Loi de Programmation Militaire (LPM) et son périmètre d'action qui définissent, depuis 2006, les acteurs et les dispositions relatives à la sécurité nationale. Par la suite, nous présentons les systèmes qui sont au cœur de cette thèse : les systèmes cyber-physiques (CPS) et plus particulièrement les systèmes de contrôle-commande industriels (ICSs). Fort de la présentation de ces systèmes, nous nous intéresserons à l'environnement qui les caractérise. Nous ferons notamment référence au contexte multi-acteurs, incertain, variable et complexe qui définissent de tels systèmes. Les techniques permettant d'assurer la productivité et la sûreté de fonctionnement de ces systèmes dans ce contexte incertain sont présentées dans cette section. Sur cette base, la dernière section permet d'énoncer les vulnérabilités existantes sur des systèmes de contrôle-commande industriels et de les classer en fonction de leur localisation dans l'architecture ICS et du composant ciblé.

Table des matières

1.	Malveillance dans les environnements industriels.....	17
1.1.	Définitions.....	17
1.2.	Historique des attaques.....	19
1.3.	Description des attaques de référence	22
1.3.1.	Stuxnet	22
1.3.2.	BlackEnergy et Industroyer : intrusion dans les réseaux électriques	24
1.3.3.	Triton : un nouveau type d'attaque	25
1.4.	Définition d'une attaque contre un système industriel.....	25
1.5.	Guides de bonnes pratiques, normes et positionnement.....	27
2.	Cadre d'application de la cybersécurité : la Loi de Programmation Militaire (LPM)	28
2.1.	La Loi de Programmation Militaire	28

2.2.	Périmètre d'action de la LPM.....	29
3.	Les systèmes industriels.....	30
3.1.	Les Systèmes Cyber-Physiques (CPSs)	31
3.2.	Les Systèmes de contrôle-commande industriels (ICSs).....	32
3.2.1.	Niveau 2 : Supervision	34
3.2.2.	Niveau 1 : Partie Commande	34
3.2.3.	Niveau 0 : Partie Opérative	35
3.2.4.	Communication entre les niveaux	36
4.	Environnement d'un Industrial Control System (ICS).....	37
5.	Étude des vulnérabilités d'un ICS	39
5.1.	Logiciel.....	40
5.2.	Firmware.....	41
5.3.	Hardware.....	41
5.4.	Réseau.....	42
5.5.	Récapitulatif des vulnérabilités ICS	42
6.	Bilan du chapitre.....	44

1. Malveillance dans les environnements industriels

Dans ce paragraphe, nous présentons les cyberattaques qui peuvent affecter un système industriel. Dans un premier temps, nous définissons ce qu'est une cyberattaque en particulier à l'encontre d'un environnement industriel. Ensuite, nous dressons une liste des attaques commises contre ces systèmes puis, nous nous intéressons aux scénarii de trois des principales références dans le domaine. Par la suite, nous définissons ce qui constitue une attaque contre de tels systèmes, puis nous faisons le lien entre nos travaux et les guides de bonnes pratiques édités par les agences gouvernementales telles que l'ANSSI (Agence Nationale de la Sécurité des Systèmes d'Information).

1.1. Définitions

L'ANSSI définit le comportement malveillant comme un ensemble d'attaques, informatiques et informationnelles (exploitation des réseaux sociaux) dont les objectifs principaux sont la déstabilisation, l'espionnage, le sabotage et la cybercriminalité (ANSSI 2018a). Les acteurs de cette malveillance, qui vont de l'individu isolé aux organisations étatiques, prennent pour cible aussi bien les particuliers que les entreprises et les administrations. Les conséquences de ces attaques sont extrêmement pénalisantes pour les victimes, allant de pertes financières au remplacement matériel en passant par la perte de crédibilité. Concernant les systèmes industriels, (ANSSI 2012; Stouffer et al. 2015) regroupent ces conséquences en trois catégories :

- Le système physique : temps d'arrêt, temps de réparation et temps de redémarrage de l'installation.
- L'environnement du système : impact négatif sur l'Homme, la santé, l'écologie, le social.
- L'entreprise : perte financière, dégradation de l'image de marque.

Une cyberattaque repose sur l'utilisation de techniques dans le but de parvenir à une malveillance à l'encontre d'un système.

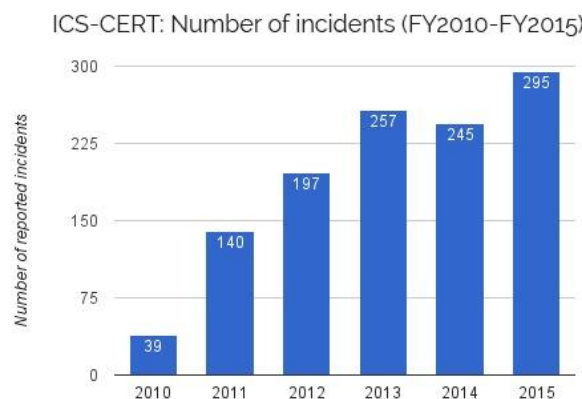


Figure 1 : Évolution du nombre d'incidents répertoriés dans les systèmes industriels entre 2010 et 2015

Dans le cadre de nos travaux, nous nous sommes focalisés sur la prise en compte des cyberattaques dans les systèmes industriels. Afin de comprendre les enjeux liés à ces derniers, la

Figure 1 représente l'évolution des attaques contre des systèmes industriels. Toutefois, la répartition des incidents n'est pas uniforme sur l'ensemble des secteurs d'applications (ICS-CERT 2016a) comme l'indique la Figure 2. Ainsi, les secteurs de l'énergie, de la production (manufacturière, chimique, etc.) et de la gestion de l'eau sont plus impactés par les incidents de sécurité que ceux du nucléaire et de la défense. Cette différence dans le choix des cibles de la part des hackers s'explique par l'ouverture de ces systèmes (smartgrids, systèmes de production...) se traduisant par l'intervention de plusieurs acteurs pour en assurer le fonctionnement, l'étendue géographique offrant des surfaces d'attaques plus importantes, ainsi que l'interopérabilité de ces systèmes. *A contrario*, les systèmes liés au secteur de la défense et au nucléaire, sont moins ouverts, avec moins d'acteurs exogènes ainsi que des interopérabilités plus faibles.

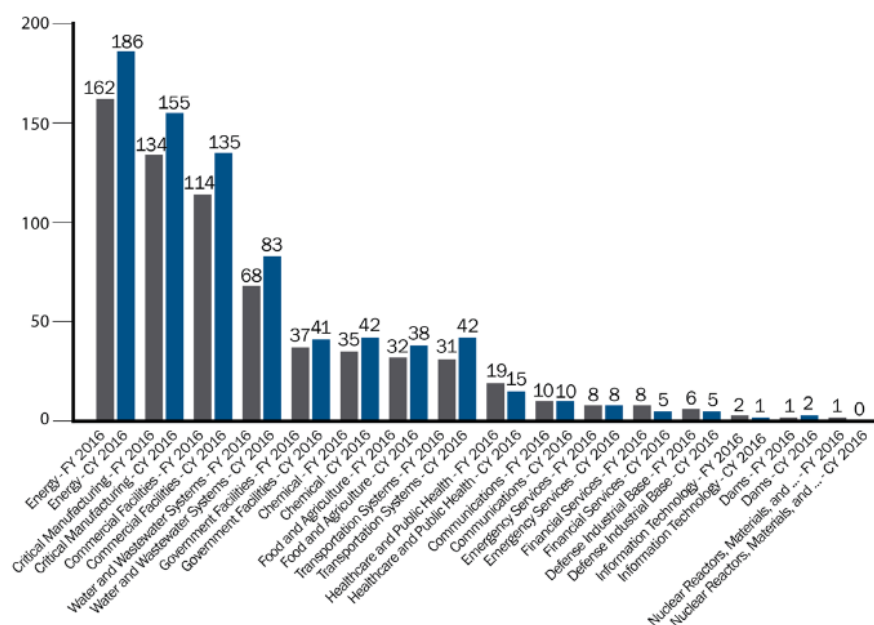


Figure 2 : Répartition des secteurs industriels impactés pour l'année 2016

Enfin, depuis quelques années, des études comme (Schwab et Poujol 2018) représentées en Figure 3, évaluent l'impact des cyberattaques sur les environnements industriels sur différents critères comme la production, l'image de l'entreprise ou encore les pertes financières.

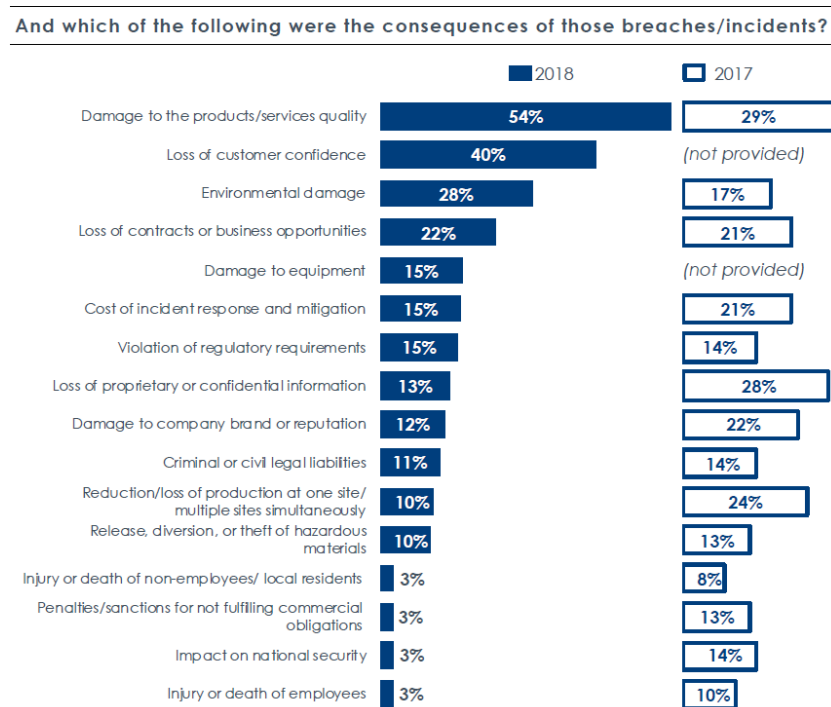


Figure 3 : Répartition des conséquences à la suite d'un incident de sécurité en 2017 et 2018 (Schwab et Poujol 2018)

A la vue de ces informations, la prise en compte du risque cyber dans les systèmes industriels devient essentielle. Dans le paragraphe suivant, nous étudions les attaques majeures ayant été perpétrées à l'encontre de ces systèmes.

1.2. Historique des attaques

Depuis le début du 21^{ème} siècle, les systèmes industriels sont devenus la cible de pirates informatiques qui exploitent les vulnérabilités de leurs composants et de leurs architectures afin d'exécuter des cyberattaques (Fourastier et Pietre-Cambacedes 2015; McLaughlin et al. 2016). L'intérêt des attaquants pour ces systèmes vient de l'impact engendré en cas de réussite et qui est renforcé par la simplicité d'accès à ces systèmes. En effet, ces derniers ont été conçus pour répondre à des problématiques de production (productivité) sans prendre en considération le risque de malveillance dont la possibilité a été largement simplifiée depuis les années 2000 par l'introduction des nouvelles technologies de communication (Berruet et al. 2007). Contrairement à un acte de malveillance perpétré dans le domaine de l'informatique (vol de données, arrêt d'un système de gestion, etc.), dans le domaine des systèmes industriels, les attaques peuvent provoquer d'importants dommages sur la partie opérative (arrêt, casse, dégradation, redémarrage, etc.), sur l'environnement (pollutions, arrêt de service, etc.) et sur l'entreprise (perte financière, image de marque) (Stouffer et al. 2015; Fourastier et Pietre-Cambacedes 2015).

Plusieurs travaux témoignent des grandes attaques commises contre des systèmes industriels tels que ceux développés par (McLaughlin et al. 2016; Khorrami, Krishnamurthy, et Karri 2016) ou encore les bases de données collectées par les CERT telles que (« RISI - The Repository of Industrial Security Incidents » 2016; ICS-CERT 2016c). A des fins de synthèse, nous pouvons classer ces attaques comme nous l'avons fait dans la Figure 4.

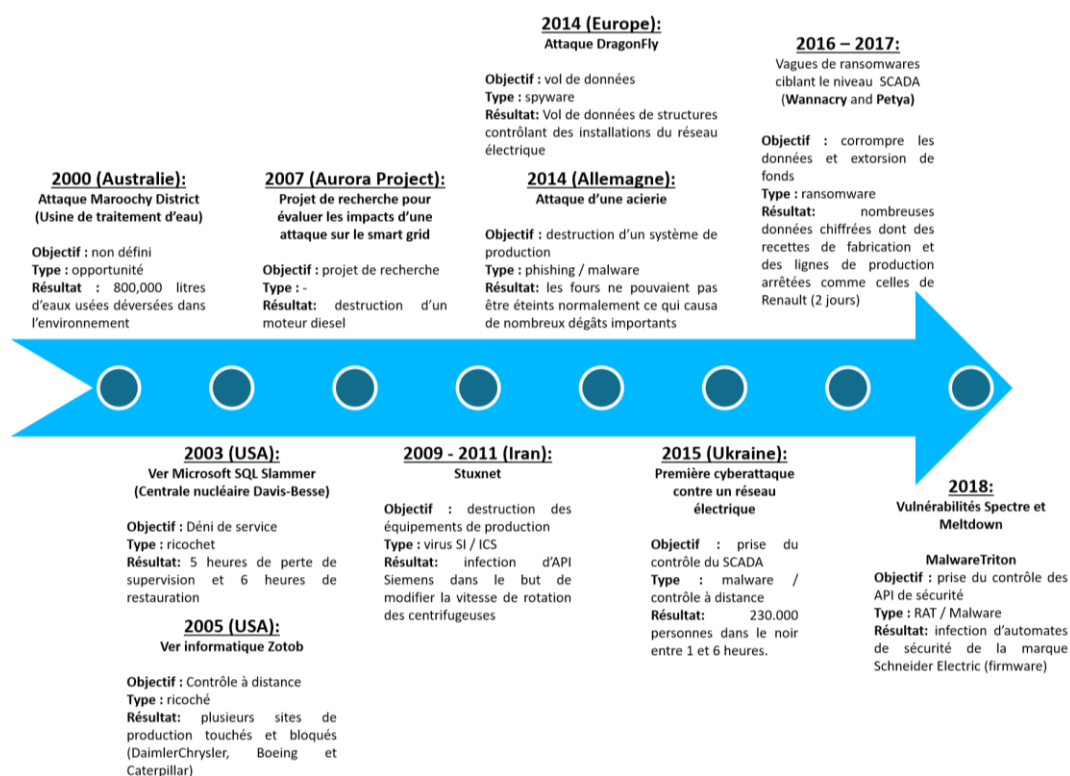


Figure 4 : Historique des principales attaques commises contre des environnements industriels au 21^{ème} siècle

Même si une grande partie de ces attaques ont eu lieu au cours du 21^{ème} siècle, la littérature fait référence à une attaque contre un pipeline russe appelée bombe logique (Thomas Reed 2005) dans les années 80. Ainsi, en 1982, en pleine Guerre Froide, la CIA aurait introduit un code malveillant (bombe logique ou trojan) dans un logiciel contrôlant le pipeline et développé par une entreprise canadienne (rconnon12 2014). La charge virale aurait ensuite réinitialisé les vitesses des pompes et les réglages des vannes afin d'augmenter la pression à l'intérieur du conduit jusqu'à son explosion. Toutefois, ce sabotage n'a pas été officiellement reconnu au niveau international (Markoff 2009).

La première agression directe reconnue et répertoriée sur un système industriel remonte au début du siècle et est une attaque d'opportunité : il s'agit de celle de Maroochy District (Abrams et Weiss 2008). Lors de cette intrusion, un ancien employé du site industriel, mécontent de son licenciement, a utilisé ses anciens identifiants pour se connecter au logiciel de supervision. Il a ensuite envoyé un ordre d'ouverture de vanne pour déverser plus de 800 000 litres d'eaux usées dans l'environnement (Slay et Miller 2007). En 2003, puis en 2005, les systèmes industriels sont les victimes collatérales d'attaques de grande ampleur contre des SI traditionnels alors qu'ils n'étaient pas la cible principale de l'attaque. Ainsi, le ver informatique Microsoft SQL Slammer réalise une attaque par déni de service (DOS) en exploitant une vulnérabilité de Microsoft SQL Server de dépassement de tampon (buffer overflow). Le ver commence alors à générer des adresses IP pour infecter une nouvelle machine, ce qui a notamment entraîné un ralentissement du trafic internet (Vamosi 2003). Par ricochet, Slammer a infecté des ordinateurs de la couche supervision de la centrale nucléaire de Davis-Besse équipée de Microsoft SQL Server où le correctif de sécurité n'était pas appliqué. Il en résulte un arrêt de 5 heures du système de supervision et de plus de 6 heures pour avoir une restauration complète du système (Poulsen 2003). En 2005, c'est au tour

du ver Zotob d'infecter des systèmes industriels, sans que ces derniers ne soient à nouveau l'objectif principal. Comme pour Slammer, Zotob utilise une vulnérabilité de Windows 2000 pour réaliser un dépassement de tampon et prendre le contrôle de la machine infectée (Symantec 2005; Roberts 2005a). Le virus a infecté plusieurs usines comme DaimlerChrysler entraînant des arrêts de production (Roberts 2005b) générant des pertes financières considérables.

L'année 2007 marque un tournant avec le projet Aurora dont l'objectif était de prouver qu'une attaque par vecteur numérique pouvait détruire physiquement un composant dans le cadre du smart grid. Le but de l'expérience était de démontrer qu'en venant contrôler l'ouverture et la fermeture rapides des disjoncteurs reliant l'alternateur entraîné par un moteur diesel au réseau électrique, il était possible de détruire le moteur (Zeller 2011). Une vidéo réalisée par l'Idaho National Laboratory et le Homeland National Security est disponible en libre accès sur plusieurs plateformes dont YouTube (Troelsen 2007). En 2008, en Pologne, un jeune garçon a utilisé une télécommande de télévision pour prendre le contrôle de l'aiguillage du tramway à Lodz (Leyden 2008). Ce geste a eu des conséquences tant sur le plan humain (12 blessés) que sur le plan matériel (4 tramways ont déraillé avec des dégâts matériels importants). Même si l'intention n'était pas de nuire, un appareil basique et des connaissances du système à contrôler ont permis de réaliser cet acte de malveillance.

A partir des années 2010, les systèmes industriels deviennent réellement et pleinement la cible des attaques. Dans la majorité des scénarii que nous détaillons ultérieurement, l'objectif des hackers est de s'introduire dans le système industriel pour en prendre le contrôle afin de réaliser des actions malveillantes ou de voler des données sur le fonctionnement du procédé contrôlé (i.e. recettes). Il en est ainsi, en 2010, du ver informatique Stuxnet. La particularité de ce dernier est qu'il a été conçu pour ne s'attaquer qu'aux environnements industriels utilisant les produits de marque Siemens, grand acteur dans le domaine des automatismes industriels. Il a notamment infecté, entre 2009 et 2011, la centrale d'enrichissement en uranium de Natanz en Iran, et plus particulièrement les automates Siemens entraînant la destruction des équipements de production (Kushner 2013). Stuxnet est considéré comme la première « cyber arme » puisqu'il a été conçu par un état (programme de la NSA et du Mosad) afin d'attaquer un autre état (Iran) en sabotant un organe précis du système industriel (centrifugeuse) faisant partie du programme nucléaire iranien (Langner 2011). Cette attaque est une référence dans le domaine de la cybersécurité des systèmes industriels, c'est pourquoi elle sera analysée dans la suite de cette section (paragraphe 1.3.1).

En 2014, un groupe de hackers lance une vague d'attaques en Europe qui vise à mener des dénis de service, de la récupération de données et de la destruction d'informations sur des entités travaillant notamment dans le domaine de l'Énergie. Cette vague d'attaque est appelée DragonFly (Symantec Security Response 2014). Tout d'abord, une campagne de phishing est menée afin que la victime exécute des macros Excel visant à infecter un réseau d'ordinateurs. Lors de cette vague d'attaques, les organes de contrôle-commande, localisés en Ukraine, sont ciblés (Kaspersky Lab 2014). Cette même année 2014, des hackers ont également réussi à prendre le contrôle d'une aciérie en Allemagne afin d'infliger de lourds dégâts matériels. Le scénario de l'attaque est « classique » : dans un premier temps, les hackers se sont introduits sur le réseau

bureautique du site industriel via une campagne de phishing, puis, les attaquants ont infecté le logiciel de supervision où ils ont pu avoir accès à la partie pilotant le système industriel. Ils ont ensuite ciblé certains automates afin de dégrader le système physique (haut fourneau) sans que les sécurités ne se déclenchent pour protéger les équipements (R. M. Lee, Assante, et Conway 2014).

A la fin de l'année 2015 (17-18 Décembre) s'est déroulée la première attaque contre un réseau de distribution électrique. Contrairement à Stuxnet, le code est moins abouti puisque les hackers se sont introduits dans le système de supervision (SCADA) afin de réaliser eux-mêmes leurs attaques (Bock et al. 2017) là où Stuxnet était autonome. Cette intrusion, connue sous le nom de BlackEnergy est détaillée dans le paragraphe 1.3.2. Elle a abouti à un blackout de la région de Kiev allant de 1 à 6 heures privant d'électricité 230 000 personnes (Zetter 2016). En 2016, le réseau ukrainien a une nouvelle fois été attaqué par un malware nommé Industroyer (détaillé en 1.3.2) dont les dégâts ont été similaires au précédent black-out causé (Polityuk, Vukmanovic, et Jewkes 2017). Les années 2016 et 2017 sont marquées par des vagues d'attaques par rançongiciel (ou ransomware) contre des systèmes industriels. Ces logiciels malveillants, comme WannaCry ou Petya, chiffrent les données de l'utilisateur présentes sur la machine infectée pour lui soutirer une rançon en échange de la clé de déchiffrement (ANSSI s. d.). Les effets de tels logiciels sur des systèmes de production sont dramatiques (IIoT World 2017) tant au niveau du blocage de la production, comme pour Renault (D. Lee 2017), qu'au niveau des données perdues, comme pour Saint-Gobain (L'Usine Nouvelle 2018).

En 2018, la menace principale se nomme Triton qui est un malware affectant les automates de sécurité Schneider Electric (Stoler 2018). Bien que l'attaquant n'ait pas réussi à prendre le contrôle du système, l'exploitation de cette vulnérabilité peut avoir des conséquences importantes si l'on considère le pouvoir de coupure de ces systèmes de sécurité. Triton sera développé dans le paragraphe 1.3.3.

Dans cette revue des attaques, de nombreuses attaques n'ont pas été évoquées comme celles en lien avec la Santé (C. Li, Raghunathan, et Jha 2011) ou encore l'automobile (Checkoway et al. 2011; Koscher et al. 2010), l'oil & gas (Bencsáth 2012). Des informations complémentaires peuvent être trouvées dans (Ani, He, et Tiwari 2017).

1.3. Description des attaques de référence

Dans ce paragraphe, nous détaillons les trois attaques de référence évoquées précédemment : Stuxnet, Industroyer et Triton. L'objectif est d'en dégager le mode opératoire ainsi que des pistes permettant de détecter l'apparition d'une attaque.

1.3.1. Stuxnet

Stuxnet est un ver informatique qui cible les systèmes industriels équipés d'automates Siemens. Il a été conçu à l'origine par la NSA pour saboter les infrastructures du programme d'enrichissement en uranium iranien, notamment le site de Natanz entre 2009 et 2011 (Kushner 2013). Stuxnet est un ver complexe puisqu'il a nécessité le vol de deux certificats différents et

des dizaines d'années de développement (Falliere, Murchu, et Chien 2011). Il repose notamment sur trois points qui seront détaillés dans l'ordre chronologique de l'attaque :

- Des techniques de social engineering pour introduire le code malveillant dans le site industriel qui n'était pas connecté à internet. Les hackers ont ciblé des employés de l'usine afin qu'ils connectent un support de stockage (clé USB) avec un fichier contenant le ver (Kushner 2013).
- Du code permettant de se répliquer sur un réseau et de cibler le matériel Siemens. Lorsque Stuxnet arrive sur le réseau bureautique du site industriel, il cherche en premier lieu à se répliquer en utilisant les connexions entre les différents éléments connectés. Le ver s'installe sur une machine uniquement si elle est équipée d'un logiciel Siemens (WinCC pour la supervision et Step7 pour la programmation d'automate) ; dans le cas contraire, le ver cherche un nouvel hôte à infecter (Falliere, Murchu, et Chien 2011). Avec cette stratégie, Stuxnet a pu s'introduire sur le réseau industriel du site pour infecter les machines et les IHM de supervision ainsi que les consoles de programmation. Ces dernières sont des ordinateurs dédiés à la programmation d'automate. Lorsque ces derniers ont été infectés, Stuxnet a pu accéder aux API contrôlant le procédé (Karnouskos 2011).
- Du code attaquant spécifiquement des automates Siemens S7-300. Comme énoncé au début, l'objectif principal de ce ver informatique est de saboter un système de production. Ainsi, lorsque Stuxnet a atteint cette couche de commande du procédé, il suit un mode opératoire très strict détaillé ci-dessous et illustré en Figure 5 (Falliere, Murchu, et Chien 2011; Karnouskos 2011) :
 - 1) Apprentissage de la partie opérative. Deux fournisseurs ont équipé les centrifugeuses du site de Natanz : le finlandais Vacon et l'iranien Fararo Paya. Stuxnet identifie le type de matériel utilisé ainsi que la vitesse de rotation appliquée aux centrifugeuses.
 - 2) Stuxnet prend ensuite le contrôle des commandes envoyées à la partie opérative via la communication Profibus (Karnouskos 2011). Cette action est facilitée par le fait qu'il s'agit d'un protocole propriétaire sans chiffrement.
 - 3) Plusieurs blocs fonctions dans l'API sont corrompus dont les OB1 et OB35 ce qui aboutit à l'envoi de séquences dégradant les équipements et le produit final. Les séquences varient en fonction du matériel utilisé mais Stuxnet alterne les vitesses de rotation rapides et lentes sur une période de temps variable puis augmente la fréquence au maximum de ses capacités (Falliere, Murchu, et Chien 2011). Dans le même temps, les IHM locaux et la supervision ne « perçoivent » pas les attaques car Stuxnet corrompt la remontée d'informations entre l'API et le niveau 2 (supervision). Enfin, les séquences d'attaques sont programmées pendant les heures de rotation des opérateurs du site industriel afin qu'ils ne détectent pas l'attaque à l'oreille.
 - 4) Stuxnet entre ensuite dans une phase de sommeil pour une durée variable avant de relancer une séquence d'attaques. Ce mécanisme rend difficile sa détection.

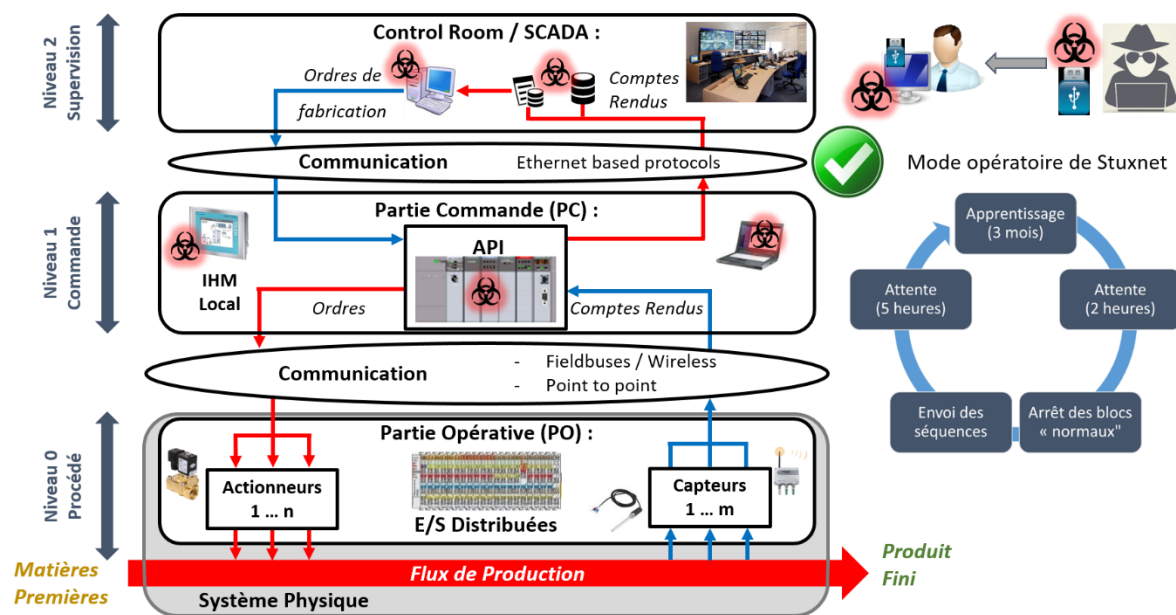


Figure 5 : Illustration de l'attaque Stuxnet sur un système industriel

1.3.2. BlackEnergy et Industroyer : intrusion dans les réseaux électriques

L'attaque BlackEnergy a eu lieu le 23 Décembre 2015, elle a été préparée pendant plusieurs mois par un groupe de hackers et a conduit à un blackout sur le réseau électrique ukrainien pendant plusieurs heures privant d'électricité 230 000 personnes (Zetter 2016; Lipovsky et Cherepanov 2016). Le scénario d'attaque s'est déroulé en trois grandes étapes détaillées dans (Bock et al. 2017). Dans un premier temps, les pirates se sont introduits sur le réseau bureautique via une campagne de spear phishing. Des mails contenant le malware étaient envoyés aux employés du site industriel afin que l'un d'eux ouvre la pièce jointe. Une fois cette étape réalisée, le logiciel malveillant BlackEnergy a envoyé des données issues du système industriel aux hackers. Pendant plusieurs mois, ils ont pu apprendre le fonctionnement du système, repérer les vulnérabilités et préparer leur attaque afin qu'elle puisse faire le plus de dégâts en un minimum de temps. La dernière étape, la plus importante a également été la plus courte puisqu'elle n'a duré que 10 minutes. Les hackers ont utilisé le malware présent dans le SCADA pour prendre le contrôle de la supervision et ouvrir les disjoncteurs du réseau électrique. Dans le même temps, une attaque DOS sur le centre de coordination a rendu impossible la transmission de l'alerte. Avant de terminer leur attaque, les pirates ont supprimé les sauvegardes afin de ralentir la remise en service du système.

Industroyer est un autre malware, qui a une nouvelle fois affecté le réseau électrique Ukrainien, dans la région de Kiev en décembre 2016. Il a eu les mêmes conséquences que l'attaque BlackEnergy une année auparavant (Polityuk, Vukmanovic, et Jewkes 2017). Le logiciel malveillant a été conçu pour s'attaquer aux systèmes industriels pilotant les sous-stations du réseau électrique. Pour cela, il s'est appuyé sur une connaissance et une compréhension du système ainsi que des protocoles de communication utilisés et de leurs vulnérabilités (Cherepanov 2017). Ces connaissances ont permis aux attaquants de prendre le contrôle d'une partie du réseau

électrique. Des informations complémentaires peuvent également être trouvées dans (Lipovsky et Cherepanov 2017).

1.3.3. Triton : un nouveau type d'attaque

Triton est un malware qui s'attaque au fonctionnement des automates de sécurité dans le but de provoquer des dégâts physiques importants sur une installation industrielle. Comme pour Stuxnet, un constructeur spécifique est ciblé par cette vulnérabilité (ICS-CERT 2018b) : Schneider Electric et la gamme d'automates de sécurité Triconex (plus précisément le modèle 3008, version de firmware 10.0-10.4) déployés dans plusieurs dizaines de milliers de systèmes à travers le monde dans des secteurs aussi variés que la pétrochimie, l'énergie ou encore la production. Ces systèmes de sécurité ont pour missions de surveiller les décisions et les données reçues, via un réseau de capteurs identiques et redondants, par le système industriel pour surveiller les fonctions critiques, et en cas de dysfonctionnement, de protéger le système (Haight 2013; Stoler 2018). Il s'agit donc bien d'automates supplémentaires, disposés en parallèle du système contrôlant la partie opérative, et disposant d'un pouvoir de coupure important, qui permet de mettre la partie opérative en sécurité en cas d'anomalie (en coupant l'alimentation électrique par exemple). Toutefois, il s'agit d'automates vulnérables eux aussi au risque de cyberattaques.

Triton agit pour se substituer au logiciel permettant de configurer les SIS Triconex (Safety Instrumented Systems) et, ainsi, pouvoir modifier leurs logiques de contrôle. En effet, ces automates particuliers sont aussi vulnérables que les autres infectés par Stuxnet car ils doivent rester configurables et adaptables (voir paragraphe 5 - Étude des vulnérabilités d'un ICS). Des détails sur les mécaniques d'infection peuvent être trouvés dans (Newman 2018), toutefois Triton devait cibler un automate ayant le bon modèle (3008), la bonne version de firmware et dont le mode « programmation » était enclenché (Stoler 2018). Une infection par le code malveillant permet de modifier la logique à l'intérieur de l'API permettant au mieux d'interrompre le flux de production en entrant dans un mode « fail safe » (incohérence entre les différents processeurs composant l'automate de sécurité) et au pire d'autoriser des événements qui ne devraient pas l'être.

1.4. Définition d'une attaque contre un système industriel

Comme nous avons pu le voir dans le paragraphe 1, les systèmes industriels sont vulnérables à de nombreux types d'attaques. Les vulnérabilités exploitées par les hackers seront détaillées dans le paragraphe 5, toutefois, en se basant sur (ICS-CERT 2016c), nous pouvons distinguer deux catégories d'attaques : celles orientées IT traditionnelles (Information Technology) et celles spécifiques à l'OT (Operational Technology).

Les systèmes industriels sont organisés en couches afin de définir les missions de chaque niveau et de réduire la complexité. Les plus hauts niveaux de cette architecture (niveau 2 et supérieurs) peuvent s'apparenter à des systèmes d'informations classiques et sont donc soumis aux mêmes types d'attaques (ENISA 2017; Dzung et al. 2005) :

- Dénî de service (*DOS*) où le but est de rendre indisponible un service ou d'empêcher des utilisateurs d'y accéder (Y. Yang et al. 2011),
- Dénî de service distribué (*DDOS*), similaire aux attaques DOS mais la technique utilisée diffère (Y. Yang et al. 2011),
- Attaque de l'homme du milieu (*Man-in-the-middle*, *MITM*) dont l'objectif est d'intercepter les communications entre deux niveaux sans qu'aucun des deux ne puisse se douter que le canal a été compromis. Cette attaque permet la récupération de données et/ou d'en injecter (Dzung et al. 2005; Yuan et al. 2016),
- Attaque par rejeu (*replay attack*) où un attaquant renvoie de manière « malicieuse » une communication déjà envoyée (Knapp 2014).

Différents travaux traitent des attaques IT qu'il est possible de réaliser sur des systèmes industriels comme ceux de (Bartman et Carson 2016) qui détaillent les vulnérabilités sur plusieurs niveaux d'architecture ou encore ceux de (Han et al. 2014) où les menaces sont détaillées couche par couche. Enfin, (ICS-CERT 2018a) explique et présente plusieurs exemples pour prendre le contrôle de la partie commande, pilotant le système industriel (MITM, Corruption des IHMs, modification des bases de données) à partir d'une attaque sur le réseau de communication industriel.

Toutefois, les systèmes de contrôle-commande des systèmes industriels ont un fonctionnement qui diffère des SI classiques ne serait-ce que par les contraintes temporelles qui doivent être garanties (temps de réponse) ou encore la séquentialité des actions requise pour aboutir à un objectif de production. Dans ce cadre, d'autres types d'attaques peuvent alors apparaître comme, par exemple :

- Aléatoire (*random attack*), l'attaquant envoie des informations aléatoires (ordre ou compte rendu) sans prendre en compte le système contrôlé, par manque de connaissance sur le fonctionnement d'un ICS ou par manque de préparation,
- Brutale (*direct attack*), l'objectif est d'endommager au maximum le système en un minimum de temps et sans que les exploitants du site puissent réagir à temps. Généralement, ce type d'attaque requiert une bonne préparation afin d'identifier l'information à envoyer au système de contrôle-commande afin de l'endommager. Toutefois, une approche à base de modèles comportementaux permet la détection immédiate de ce type d'attaques comme dans (Sicard, Zamaï, et Flaus 2017a),
- Séquentielle (*sequential attack*), la logique de contrôle imposée par la loi de commande est cassée par l'attaque. Contrairement au type d'attaque précédent, l'objectif est d'infliger des dégâts importants au système sur la durée (Caselli, Zambon, et Kargl 2015; W. Li et al. 2016),
- Par injection de fausses données (*False Data Injection*, *FDI*), l'attaquant intercepte des données et les corrompt avant de les renvoyer. Nous distinguons :
 - Injection de fausses commandes, où les ordres sont corrompus dans la chaîne de commande (PC vers PO) (Gao et al. 2010; Sicard, Zamaï, et Flaus 2017a),

- Injection de faux comptes rendus, où les données issues du réseau de capteurs sont corrompues avant d'être transmises à la partie commande (PO vers PC) afin d'induire une action erronée (Wang et al. 2014).

Les grandes catégories d'attaques possibles dans ce domaine, qui ont été présentées ci-dessus, sont reprises sur un exemple de feu de circulation dans (Z. Li et al. 2016). Elles s'appuient sur des vulnérabilités qui seront détaillées dans le paragraphe 5. Pour conclure ce paragraphe, nous définirons une attaque comme un acte malveillant, dont l'intention est de nuire, ayant pour objectif de dégrader le service fourni par une installation (destruction du système et/ou de son environnement, dégradation de la qualité, interruption du service). Une cyberattaque est l'équivalent de la précédente définition mais dans le cyberspace où les moyens utilisés sont numériques. Elle fait référence au risque cyber qui peut affecter, directement ou non, les particuliers, les administrations et les entreprises sur les aspects de cybercriminalité, d'atteinte à l'image, d'espionnage et de sabotage.

1.5. Guides de bonnes pratiques, normes et positionnement

Comme nous avons pu le détailler dans les paragraphes précédents, dans le cadre des systèmes de contrôle-commande industriels, les effets d'une attaque peuvent être importants (Luallen et Harp 2014) et avec de nombreuses conséquences (ANSSI 2012; Stouffer et al. 2015).

Les états ne pouvaient ignorer plus longtemps ce risque cyber, ce qui a conduit à la création de plusieurs instituts gouvernementaux comme l'ANSSI (France, 2009), CPNI (Center for the Protection of National Infrastructure, Grande Bretagne, 2007) ou encore l'ENISA (European Union Agency for Network and Information Security, Europe, 2004). Nous rajouterons également l'institut américain NIST (National Institute of Standards and Technology, USA, 1901) qui a débuté ses travaux sur le risque de cyberattaques au début des années 2000. Une des missions principales de ces instituts a été d'appréhender le risque pour les systèmes industriels et également d'orienter les choix technologiques, d'organiser la formation et la prévention vis-à-vis du risque cyber des acteurs opérant sur les systèmes industriels ou encore d'assurer la certification des produits. Nous pouvons également citer d'autres organismes travaillant sur l'obtention d'une norme ou d'un standard pour la cybersécurité comme l'IEC 61850 pour les smart grids (Kabir-Querrec et al. 2016) ou encore l'ISA 99 pour l'organisation d'un standard d'architecture de défense contre les cyberattaques et l'amélioration globale de la sécurité des systèmes industriels. Enfin, (Fourastier et Pietre-Cambacedes 2015) (CLUSIF 2014) font le constat que plusieurs organismes dans des secteurs spécialisés ont également créé leurs propres référentiels en tenant compte des spécificités de leurs activités. Il en a résulté plusieurs guides de bonnes pratiques qui servent de référence, tels que :

- Guide to Industrial Control Systems (ICS) Security (NIST 800-82, r2) (Stouffer et al. 2015),
- Protecting Industrial Control Systems (ENISA 2011),
- Maîtriser la SSI pour les systèmes industriels (ANSSI 2012) et cas pratique d'un tunnel routier (ANSSI 2014).

Au-delà de ces guides de bonnes pratiques, en France, un cadre légal est fourni à la cybersécurité dans le cadre de l'application de la Loi de Programmation Militaire (LPM). Cette dernière définit notamment un ensemble d'acteurs indispensables à la sécurité de la nation qui doivent mettre en œuvre les recommandations de l'ANSSI (ANSSI 2012). Cette thèse se positionne dans le quatrième item de ces recommandations : **surveillance des installations et détection des incidents**. Les aspects liés à la LPM sont détaillés dans le paragraphe suivant.

2. Cadre d'application de la cybersécurité : la Loi de Programmation Militaire (LPM)

2.1. La Loi de Programmation Militaire

Depuis 1997, la Loi de Programmation Militaire (LPM) donne les grandes orientations stratégiques contenues dans le Livre blanc sur la défense et la sécurité nationale. Elle est rédigée par une commission composée de parlementaires, de responsables d'administrations et d'acteurs de la sécurité nationale. Elle définit le cadre légal d'application à ces recommandations, dépend du ministre de la Défense, impose un certain nombre de règles pour garantir la sécurité nationale et fixe le budget alloué pour les appliquer sur une période de cinq ans. En 2006, l'État français prend conscience de la menace du risque « cyber » ce qui se traduit par la rédaction deux ans plus tard du Livre blanc sur la défense et la sécurité nationale (Commission du Livre blanc 2008). L'Agence Nationale de la Sécurité des Systèmes d'Information, l'ANSSI, est alors créée en 2009 pour accompagner les transformations et adaptations des acteurs de la cybersécurité ainsi que renforcer la cohérence des actions de l'État (Secrétariat Général de la Défense Nationale 2009). La création de l'ANSSI est une étape forte dans la protection des systèmes d'information sensibles et l'agence se voit confier quatre missions essentielles : détecter et réagir en cas d'attaques informatiques, prévenir la menace, conseiller les administrations et les acteurs stratégiques et informer les entreprises et les particuliers de la menace cyber. L'ANSSI est sous la direction du premier ministre français et permet de faire le lien afin que le cadre d'action de la LPM soit appliqué.

Le Livre blanc sur la défense et la sécurité nationale de 2013 (Commission du Livre blanc 2013) fait basculer la cybersécurité dans une autre dimension, notamment dans le chapitre IV : dispositions relatives à la protection des infrastructures vitales contre la cybermenace, avec la création de plusieurs termes (ANSSI 2018b) définis ci-dessous et illustrés dans la Figure 6 :

- Opérateur d'Importance Vitale (OIV) : en France, ce sont des organisations identifiées par l'État comme ayant des activités indispensables ou potentiellement dangereuses pour la population. Les dommages, l'indisponibilité ou la destruction d'un OIV nuirait au potentiel de guerre ou économique, à la sécurité ou à la capacité de survie de la Nation. A ce jour, plus de 200 opérateurs publics et privés dont les activités sont indispensables au bon fonctionnement et à la survie de la Nation ont été identifiés. Pour des questions de sécurité nationale, cette liste est gardée confidentielle et définie par arrêté ministériel.

- **Système d'Information d'Importance Vitale (SIIV) :** il s'agit des systèmes d'information (SI) des OIV. Ces SIIV peuvent être situés sur des PIV (Points d'Importance Vitale) différents et répartis sur le territoire national dans des ZIV (Zones d'Importance Vitale). Dans le cas où un SIIV a des liens avec d'éventuels sous-traitants ou fournisseurs, les SI (systèmes d'information) correspondants de ces derniers sont également impactés.
- **Secteur d'Activités d'Importance Vitale (SAIV) :** « un ensemble d'activités, essentielles et difficilement substituables ou remplaçables, concourant à un même objectif ou visant à produire et à distribuer des biens ou des services indispensables ». Douze secteurs d'activité d'importance vitale pour la Nation ont été définis en France, au sein desquels ont été identifiés des Opérateurs d'Importance Vitale (OIV) chargés de la protection de leur Point d'Importance Vitale (PIV).

La LPM en vigueur (décembre 2013) dispose également d'un nouveau périmètre d'action détaillé dans la section 2.2.

2.2. Périmètre d'action de la LPM

Le périmètre de la LPM est corrélé aux notions d'OIV, de SAIV et de SIIV expliquées dans la section 2.1. Les OIV, ainsi que les sous-traitants, identifiés dans l'un des douze SAIV se doivent d'appliquer à leurs frais les contre-mesures imposées par la LPM afin de renforcer leurs SIIV. Ces exigences sont formalisées par les Directives Nationales de Sécurité (DNS). Il s'agit d'une contrainte forte puisque les OIV sont obligés, par la loi, d'appliquer les DNS de leur secteur d'activité sous peine d'exposer leur(s) dirigeant(s) à une amende pouvant atteindre 150 000€ par fait et 750 000€ pour une personne morale. Les articles 21, 22, 23 et 24 de la LPM formalisent au niveau juridique ces obligations (Commission du Livre blanc 2013). Le Tableau 1 regroupe les secteurs d'activités concernés ainsi que leurs ministères coordinateurs.

Tableau 1 : Secteurs d'Activités d'Importance Vitale et Ministères associés

Secteur d'Activité	Ministère coordinateur
Activité Civile de l'État	Ministère de l'intérieur
Activité Militaire de l'État	Ministère des armées
Activités Judiciaires	Ministère de la justice
Alimentation	Ministère de l'agriculture
Communications électroniques, audiovisuel et information	Ministère des communications électroniques
Espace et Recherche	Ministère de la recherche
Énergie	Ministère de l'énergie
Finances	Ministère de l'économie et des finances
Gestion de l'eau	Ministère de l'écologie

Santé	Ministère de la santé
Industrie	Ministère de l'Economie, de l'Industrie et du Numérique
Transport	Ministère des transports

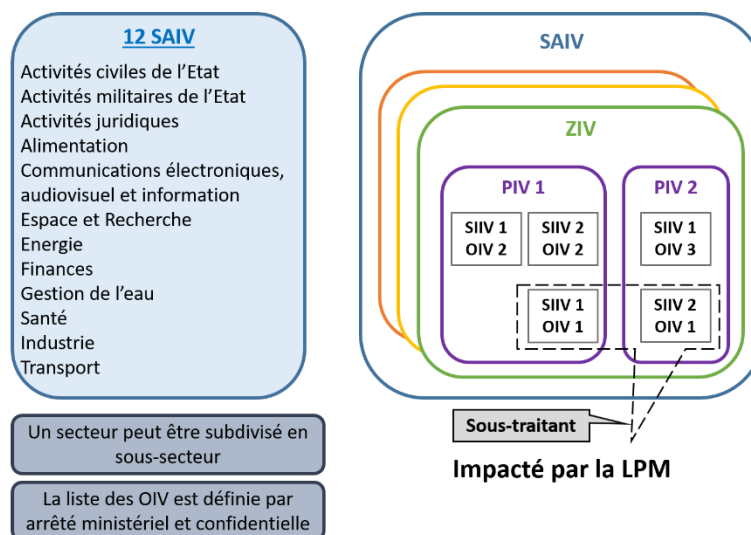


Figure 6 : Synthèse du périmètre d'action de la LPM

Cette sécurisation sectorielle passe notamment par l'application d'un certain nombre de règles de sécurité obligatoires pour les OIV et leurs sous-traitants au niveau législatif. La France est le premier pays à être passé par la réglementation pour mettre en place un dispositif efficace et obligatoire de cybersécurité de ces infrastructures critiques. La Figure 6 synthétise le contenu et le principe de fonctionnement de la LPM (Commission du Livre blanc 2013).

Nous pouvons remarquer que le SAIV « activités militaires de l'état » est concerné par l'exploitation de systèmes de contrôle commande industriels, au travers de ceux déployés dans le cadre des systèmes d'armes et/ou de leurs servitudes. Il apparaît ainsi ici l'importance de protéger des attaques de type « cyber » ces systèmes numériques interagissant avec un processus physique : il s'agit de la cybersécurité pour les systèmes industriels. Ces derniers ont d'ailleurs un dénominateur commun puisqu'il s'agit de systèmes répartis qui ont pour mission principale commune la production d'un objectif. Autrement dit, il s'agit de systèmes physiques interconnectés qui partagent un système de pilotage numérique, ce dernier étant vulnérable à des attaques par vecteur numérique.

3. Les systèmes industriels

Dans la section 2, nous avons présenté le périmètre d'action de la LPM qui met en avant la nécessité de prendre en considération le risque cyber pour les SI des OIV nécessaires à la sécurité de la Nation. Cependant, certains de ces SIIV pilotent des systèmes physiques. Les notions d'infrastructures critiques et de systèmes cyber-physiques sont alors avancées. Dans cette

section 3, nous nous intéressons à la nature de ces systèmes ainsi qu'à leur représentation et leur principe de fonctionnement.

3.1. Les Systèmes Cyber-Physiques (CPSs)

Le terme « système cyber-physique », ou encore « Cyber-Physical System » (CPS) en anglais, est apparu pour la première fois en 2006 dans (E. A. Lee 2006). Un CPS désigne l'intégration de composants numériques et physiques qui interagissent au travers d'un réseau de communication. (Olivier Cardin 2016) a répertorié les différentes définitions qui ont pu être données au CPS au fur et à mesure des développements. Toutefois, un système cyber-physique, représenté en Figure 7, se structure autour de trois couches : physique, communication et cyber (Leitão, Colombo, et Karnouskos 2016). La première couche, dite physique, permet d'interagir avec l'environnement du CPS. Les capteurs permettent d'acquérir l'information à remonter à la couche numérique et les actionneurs permettent d'agir sur le système physique. La couche communication permet de transmettre les informations entre les couches cyber et physique. Plusieurs techniques peuvent être utilisées en fonction de la quantité d'informations qui doit transiter, de la vitesse de transmission ou encore du niveau de sécurité (wifi, bluetooth, réseau IoT (Internet of Things) (ZigBee, SigFox ; LoRaWAN), 4G/5G, internet). Enfin, la couche cyber permet d'interpréter les données reçues afin de contrôler le système physique en garantissant sa stabilité et sa performance.

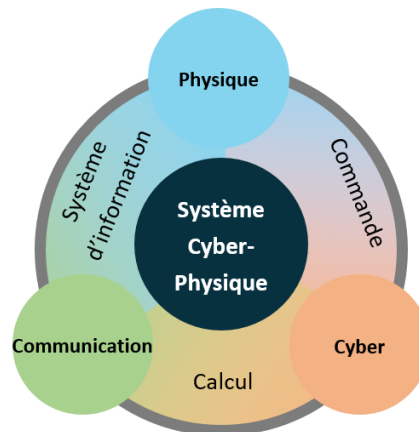


Figure 7 : Représentation d'un système cyber-physique (Leitão, Colombo, et Karnouskos 2016; Ashibani et Mahmoud 2017)

Les CPS sont des systèmes décentralisés, qui communiquent entre eux et aboutissent à la construction de sous-systèmes de systèmes. Une classification par fonctionnalités et caractéristiques a été proposée par (J. Lee, Bagheri, et Kao 2015) et permet de distinguer cinq catégories (5C) :

- Connexion (utilisation des données d'un réseau de capteurs),
- Conversion (traitement et retranscription de l'information),
- Cyber (connaissance des autres CPS de l'environnement et interaction avec eux),
- Cognition (diagnostic basé sur des simulations de son propre comportement),
- Configuration (adaptation et reconfiguration en cas de défaillance).

Cette architecture 5C a également été pensée pour guider la construction d'un CPS, les normes (IEC, ISO, DIN) guidant ce développement sont détaillées dans (Ahmadi et al. 2017). Cependant, il apparaît également pertinent de classer les CPS en fonction de leurs domaines d'utilisation. Ces derniers sont extrêmement vastes, (Klötzer, Weißenborn, et Pflaum 2017) et (Zhong et al. 2017) en donnent une description exhaustive, mais peuvent être réduits selon (Leitão, Colombo, et Karnouskos 2016) à :

- IoT et communication : conception de systèmes de communication innovants et dont l'objectif est le déploiement d'objets connectés auparavant isolés,
- Robotique : application pour la production, le service ou encore la conception d'humanoïdes,
- Énergie : application sur la production et la distribution d'énergie mais également les concepts de smart city et les infrastructures attenantes (smart home et building),
- Transport : modes de transports individuels ou collectifs de personnes et de marchandises,
- Défense : application dans le domaine militaire,
- Santé : application dans le smart health,
- Industrie : application au domaine de la production (smart industry).

Dans notre étude, nous nous limitons aux applications des CPS dans le domaine de la production, aussi appelées Industrial CPS (ICPS) (Leitão, Colombo, et Karnouskos 2016) ou encore Cyber-Physical Production System (CPPS). Au sein de ces environnements, nous nous intéressons aux structures de pilotage d'installations existantes, dont la pérennité est de l'ordre de 20 à 30 ans. Dans le cadre de la malveillance, ces installations doivent pouvoir, sur ce transitoire, être sécurisées. Bien entendu, l'approche que nous développerons plus loin dans ce cadre, devra à terme être étendue aux nouvelles structures de pilotages préconisées dans l'industrie 4.0, plus collaboratives (reconfigurables et répartis géographiquement) (O. Cardin et al. 2017) (Trentesaux, Borangiu, et Thomas 2016) qu'intégratives (Monostori 2014). Ces systèmes de pilotage sont nommés systèmes de contrôle-commande industriels, (*Industrial Control System*, ICS), développés dans le paragraphe 3.2.

3.2. Les Systèmes de contrôle-commande industriels (ICSs)

Les systèmes de contrôle-commande industriels déployés à ce jour, ou Industrial Control System (ICSs) en anglais, mettent en œuvre un ensemble de couches numériques et physiques qui interagissent afin de réaliser un objectif dans un environnement industriel. Cette définition est très proche de celle donnée pour un système cyber-physique dans le paragraphe 3.1 (Ashibani et Mahmoud 2017). De ce fait, les CPPS représentent l'avenir des ICSs avec une intégration plus forte des outils numériques afin de renforcer l'objectif de production industrielle : il s'agit de l'industrie 4.0 (Monostori 2014).

De nos jours, les systèmes de contrôle-commande sont utilisés dans de nombreuses applications (Alphonsus et Abdullah 2016) notamment dans des infrastructures soumises au périmètre d'action de la LPM telles que détaillées dans (Stouffer et al. 2015) et (Fourastier et Pietre-Cambacedes 2015) :

- Énergie : production et distribution d'énergie (eau, gaz, électricité, oil & gas),
- Production : systèmes industriels (manufacturier, chimique, pharmaceutique),
- Transports : trains, voitures, avions,
- Santé,
- Défense.

Ainsi, nous remarquons ici que les secteurs, présentés précédemment et dans le paragraphe 3.1, sont une évolution intégrant une couche de calcul plus importante définie par le terme « smart ». De par leurs utilisations dans ces différents secteurs, l'implémentation physique de ces systèmes est très variable (Colla, Leidi, et Semo 2009). Durant les années 1970-1980, le modèle de référence CIM (Computer Integrated Manufacturing) a été proposé afin de hiérarchiser les échanges, les rôles et fonctions au sein d'un ICS. Cette organisation est également appelée Purdue Reference Model (Purdue Research Foundation 1991) et sert de base pour de nombreuses normes telles que l'ISA 95 et l'ISA 99. Elle se compose de 6 niveaux (Ani, He, et Tiwari 2017) :

- Niveau 5 : Information Stratégique (Direction générale de l'entreprise), production de l'entreprise, planification.
- Niveau 4 : Production de l'usine, ordonnancement (Direction opérationnelle usine).
- Niveau 3 : Coordination inter-unités, gestion de l'approvisionnement.
- Niveau 2 : Contrôle de l'unité (contrôle et supervision).
- Niveau 1 : Contrôle du procédé (contrôle local).
- Niveau 0 : Partie opérative (transformation physique).

Le modèle CIM permet d'interconnecter le système de production avec les tâches de plus hauts niveaux comme la logistique ou l'ordonnancement. Par la suite, ce modèle est également utilisé pour diminuer la complexité des tâches à automatiser au sein du contrôle-commande en cloisonnant les éléments à contrôler (Henry 2005). La Figure 8 permet de représenter un ICS avec le découpage proposé pour le modèle CIM.

Chaque niveau est composé d'éléments caractéristiques et échange des informations avec les niveaux supérieur et inférieur. Nous présentons dans cette étude les 3 premiers niveaux d'un ICS également détaillés dans (Sicard, Zamaï, et Flaus 2018a). Une présentation plus détaillée est fournie dans (Berruet et al. 2007).

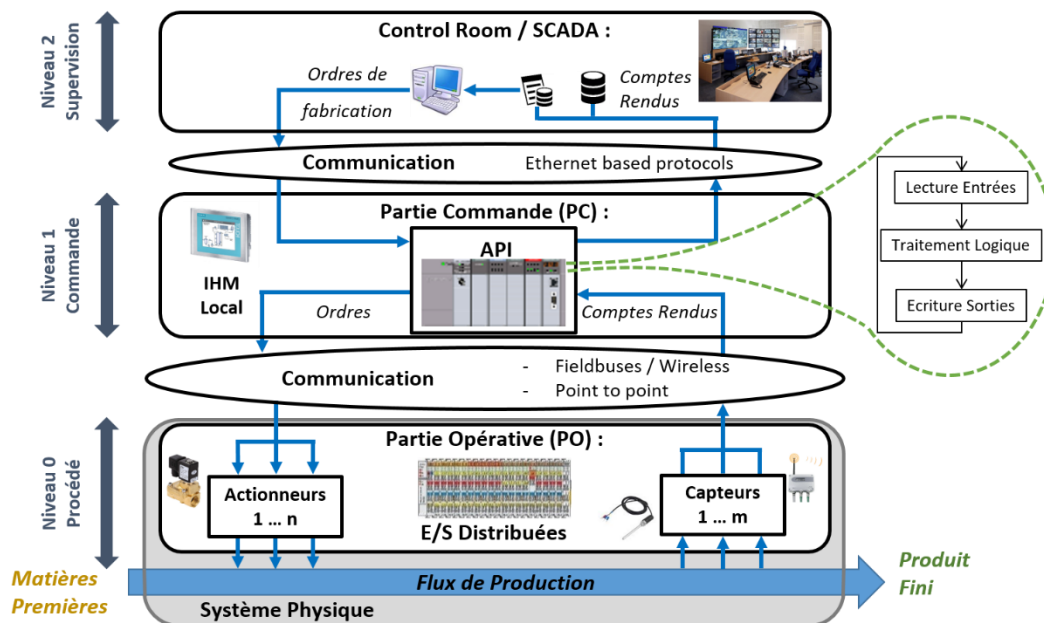


Figure 8 : Représentation d'un ICS sous forme fonctionnelle avec la norme CIM

3.2.1. Niveau 2 : Supervision

Ce niveau remplit plusieurs missions dans un ICS. Tout d'abord, il permet de collecter les données des couches inférieures (Niveaux 1 et 0) dans le but de fournir aux opérateurs une image représentative du système à un instant donné. Ainsi, au travers de cette couche de supervision, les différents acteurs d'un ICS, des opérateurs aux managers, sont capables de suivre l'évolution du flux de production. Les ordres de fabrication sont transmis au niveau 1 (Commande) dans le but de modifier la loi de commande (qui intègre la recette) ou d'ajuster une consigne afin d'éviter ou de limiter les dommages sur un ICS. Enfin, ce niveau communique également avec les couches supérieures de la pyramide CIM. Cette couche est appelée supervision, salle de contrôle ou également SCADA pour Supervisory Control And Data Acquisition.

Ce terme peut être source de confusion puisqu'il est utilisé dans la littérature pour désigner l'ensemble de l'architecture d'un système de contrôle-commande présenté en Figure 8 ou pour faire référence à ce niveau particulier (Sicard, Zamaï, et Flaus 2018a). Dans le premier cas, SCADA devient un synonyme d'ICS et nous parlerons d'architecture SCADA pour définir l'ensemble de la chaîne de contrôle et de monitoring du poste de supervision à la partie opérative. Dans le second cas, SCADA désigne uniquement la couche de supervision. Il désigne alors les éléments permettant de monitorer le procédé contrôlé.

3.2.2. Niveau 1 : Partie Commande

Cette couche interagit avec le niveau 0 (Partie Opérative) en faisant l'acquisition des données des capteurs et en contrôlant les actionneurs, avec le niveau 2 (Partie Commande) en communiquant avec les autres éléments (autres automates, IHM locaux) et avec le niveau 2 (Supervision) en échangeant des informations avec les opérateurs d'exploitation. L'élément principal de ce niveau est l'API (Automate Programmable Industriel, aussi appelé PLC pour Programmable Logic Controller en anglais) qui contrôle l'ICS en temps-réel. Le niveau 1 est

également appelé partie commande (PC). Un historique très complet sur les API peut être trouvé dans (Kamel et Kamel 2013) notamment de leurs créations dans les années 60 par Modicon Corporation (Modicon 084) jusqu'aux années 2010. Leurs applications sont variées comme détaillées dans (Alphonsus et Abdullah 2016) et dans le début de ce paragraphe.

Les ouvrages comme (Bolton 2009; Hanssen 2015) permettent d'appréhender l'architecture d'un API, son intégration dans un ICS ainsi que son fonctionnement. En effet, en fonctionnement normal, ce dernier a un fonctionnement cyclique lui permettant de contrôler le procédé en temps-réel : (1) acquisition des données mesurées par les capteurs (lecture des entrées), (2) calcul des actions à mener sur le système en fonction des données capteurs et du programme automate (traitement logique), (3) envoi des ordres aux actionneurs (écriture des sorties). Ce cycle est représenté sur la Figure 8 et peut être complété par une étape de vérification interne de l'API aussi bien au niveau hardware (cohérence des I/Os) qu'au niveau logiciel (watchdog) (Hanssen 2015). Cette étape est réalisée au début de chaque cycle si les contraintes temps-réel le permettent. Enfin, la Figure 9 représente l'architecture d'un API (Bolton 2009) qui sera notamment utile pour comprendre en quoi ces éléments sont vulnérables à des cyberattaques.

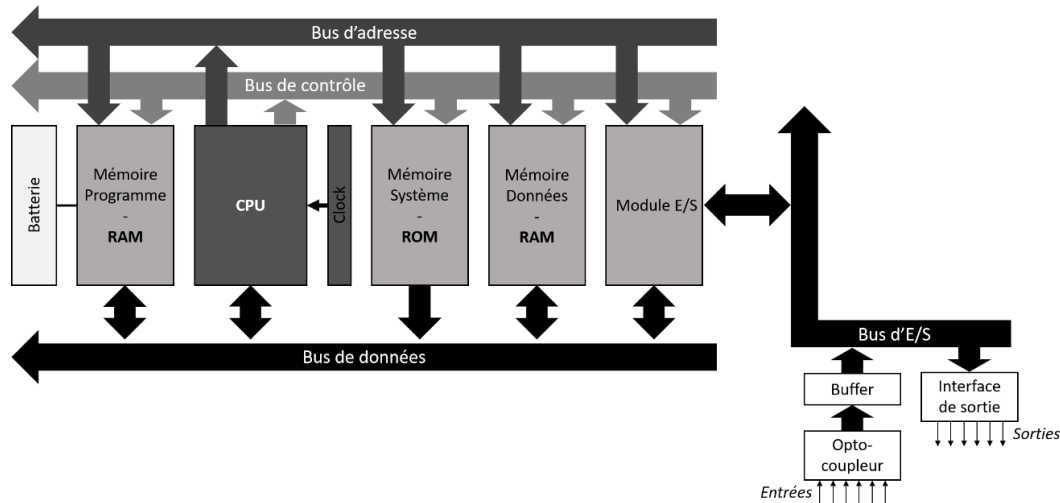


Figure 9 : Architecture détaillée d'un Automate Programmable Industriel (Bolton 2009)

3.2.3. Niveau 0 : Partie Opérative

La mission de cette couche est de transformer les matières premières en produits finis en interagissant avec le flux de production. En d'autres termes, le système de contrôle-commande doit faire évoluer le produit d'un état initial à un état final en réalisant des actions sur le flux de production tout en assurant la productivité et la fiabilité du système.

Ce niveau se compose de capteurs et d'actionneurs qui sont à l'interface entre les couches physiques et matérielles. Les capteurs appartiennent à la chaîne d'acquisition qui regroupe l'ensemble des composants permettant d'acquérir, d'élaborer et de transmettre les informations exploitables par l'ICS (Henry 2005). Ainsi, une mesure physique est convertie en signal électrique afin d'être transmise au Niveau 1 : Partie Commande. Les actionneurs, eux, appartiennent à la chaîne d'action qui se compose de trois éléments : le pré-actionneur, l'actionneur et l'effecteur (Gentil et Zamaï 2003). L'objectif de cet ensemble est d'appliquer les ordres envoyés par la partie

commande afin d'agir sur le flux de produits. Pour cela, la chaîne d'action convertit un signal électrique en énergie transmise au système. Ainsi, le pré-actionneur assure la distribution de l'énergie électrique (signal de la partie commande) en énergie électrique, hydraulique ou pneumatique. Cette dernière est ensuite convertie par l'actionneur en énergie mécanique. Enfin, l'effecteur l'utilise pour agir sur le flux de produits.

Le système physique est l'ensemble comprenant la partie opérative et le flux de production.

3.2.4. Communication entre les niveaux

Le réseau de communication a pour mission de connecter les différentes couches présentées précédemment afin de leur permettre d'échanger des informations. Il est important de distinguer la communication entre les couches 1-0 et 2-1. En effet, les exigences concernant les temps de réponse et la quantité de données transportées sont très différentes. Ainsi, des communications point à point ou utilisant des protocoles propriétaires sont utilisées pour connecter les couches temps-réel tandis que des protocoles souvent basés sur Ethernet TCP/IP sont employés pour connecter le système de contrôle à la supervision. Toutefois, depuis le début du 21^{ème} siècle, nous pouvons remarquer une tendance à développer des protocoles s'appuyant sur TCP/IP (dédié IT, à garantie de performance d'acheminement) pour connecter toute l'architecture de la supervision à la partie opérative (EtherNet/IP, EtherCAT) ou encore l'utilisation de plus en plus importante de wireless protocole avec l'émergence de l'industrie 4.0 (Wireless HART, ISA100).

La communication entre la partie opérative et la partie commande transporte peu de données (bits, bytes) avec des temps de réponses extrêmement courts (hard real-time), de l'ordre de la milliseconde (Clarhaut et al. 2015; Knapp 2014). Historiquement, cette communication utilise des connexions point à point ou des bus terrains s'appuyant sur des protocoles temps réels. Nous pouvons notamment citer des protocoles généraux comme le Profibus (Siemens), le Fipway (Schneider Electric), des plus spécifiques comme l'AS-i (production), le CAN (automobile) ou des protocoles ouverts comme le Modbus. Cependant, des protocoles « sans-fil » remplacent peu à peu dans les catalogues industriels les communications précédemment évoquées pour transmettre des données capteurs (Ashibani et Mahmoud 2017; Berruet et al. 2007), notamment dans des secteurs en plein développement (Smart Grid, Industrie 4.0).

En revanche, la communication entre la supervision et la partie commande requiert de transporter plus de données (kbytes) avec un temps de réponse proche du « soft real-time », de l'ordre de la minute / seconde. Ces protocoles (Clarhaut et al. 2015; Knapp 2014), comme le Profinet (Siemens), sont basés sur de l'Ethernet TCP/IP. De nos jours, les constructeurs ont tendance à développer des protocoles permettant leur intégration tout le long de l'architecture ICS, du niveau 2 (supervision) au niveau 0 (partie opérative). Nous pouvons notamment citer EtherNet/IP (ODVA 2016) qui se base sur le protocole Ethernet TCP/IP et respecte les normes récentes telles que l'IEEE 802.3, ou encore EtherCAT (EtherCAT Technology Group 2012) qui se présente comme un bus de terrain Ethernet (à garantie de performance temporelle) permettant de relier les différents niveaux de la pyramide CIM en respectant les normes IEC et ISO.

L'introduction du protocole TCP/IP dans les ICSs a été extrêmement positive en augmentant notamment la facilité de communication et donc l'interopérabilité des systèmes logiciels.

Cependant, cette intégration n'a pas pris en considération l'aspect sécurité. Nous verrons par la suite en quoi ces protocoles de communication renforcent ces problématiques de cybersécurité pour les systèmes de contrôle-commande industriels.

4. Environnement d'un Industrial Control System (ICS)

Comme introduit dans la section 1, les ICSs doivent dorénavant faire face aux actes de malveillance. Dans la section 3, nous avons souligné le fait que ces derniers sont rendus en partie possibles par l'introduction de technologies héritées du monde de l'IT (Information Technology) et des SI (Systèmes d'information) comme le protocole Ethernet TCP/IP ou la technologie sans-fil. Toutefois, avant d'introduire la notion de cybersécurité pour les ICSs, une bonne compréhension de l'environnement dans lequel ces systèmes évoluent, est essentielle. Celui-ci est caractérisé par le type d'ICS, l'incertitude, la variabilité, la complexité et un contexte multi-acteurs.

Tout d'abord, en fonction du secteur d'application de l'ICS, le contrôle-commande est de type (Gentil et Zamaï 2003) :

- Discret, les parties commandes et opératives échangent alors des variables discrètes (TOR),
- Continu, dans ce cas il s'agit de variables analogiques (0-10V, 4-20 mA, ...),
- Hybride, cas le plus fréquent dans lequel l'ICS utilise à la fois des variables discrètes et continues pour contrôler le flux de production.

De plus, les systèmes de contrôle-commande industriels sont soumis à des aléas de fonctionnement caractérisant l'incertitude de leur environnement. L'incertitude du contexte de production renvoie au concept de défaillance qui, d'après (AFNOR 1991), est l'incapacité à réaliser un service. L'apparition d'une défaillance peut conduire à un défaut sur le produit ou le composant. Ainsi, le défaut est défini par l'écart entre les spécifications souhaitées et les caractéristiques observées (Villemeur 1992). Une défaillance peut apparaître à la suite d'une autre défaillance mais la première est qualifiée de défaillance source. Dans un ICS, 4 sources de défaillances possibles (Nguyen 2015) sont généralement mises en exergue :

- Facteur humain : en dépit du fait que l'être humain est indispensable au bon fonctionnement d'un ICS, il est une source de perturbation soit parce qu'il n'a pas une vision globale du problème, soit parce qu'il est faillible volontairement ou non,
- Équipements : ces derniers sont au cœur du fonctionnement des ICSs. Ils peuvent faire l'objet de fatigue ou de phénomènes d'usure. De plus, une mauvaise fabrication, un mauvais montage ou une mauvaise utilisation réduit également leurs durées de vie,
- Ordre de fabrication (recette) : ils permettent de séquencer la production en demandant l'application des recettes contenant les lois de commande qui seront par la suite appliquées par les différents API. Cependant, ces recettes sont développées par la R&D et qualifiées sur des équipements dédiés avant la mise en production.

Ainsi, lors du passage en production, ces recettes peuvent ne pas correspondre parfaitement au contexte de production (Ben Said et al. 2015),

- Produit : comme présenté dans la section 3, l'ICS permet de transformer les matériaux initiaux en produit final en agissant sur le flux de production. Cependant, une mauvaise spécification des matières premières peut entraîner des défaillances ou dégrader les équipements de production.

Au-delà du contexte incertain, l'environnement est également variable. En effet, avec l'avènement de l'industrie 4.0, les ICSs tendent à être plus flexibles et réactifs au changement de la demande client. Afin d'assurer de faibles volumes avec une large variété de produits (Abu-Samah, Shahzad, et Zamaï 2017), les systèmes de contrôle-commande se composent d'équipements de production reconfigurables au niveau hardware et firmware (S. Henry, Zamaï, et Jacomino 2004). Nous pouvons notamment citer les travaux de (Dequeant 2017) sur ces problématiques de variabilité dans le cas de la production de produits dans le domaine du semi-conducteur.

Les systèmes de contrôle-commande industriels sont le plus souvent complexes de par les éléments évoqués précédemment (variabilité, incertitude), le nombre de variables à contrôler et l'architecture matérielle déployée pour assurer la production de l'ICS (Berruet et al. 2007). En effet, cette architecture se décompose en plusieurs niveaux dont les chaînes fonctionnelles peuvent être multiples afin de réduire cette complexité (Henry 2005). Au-delà de l'architecture présentée en Figure 8 et précisée précédemment, les ICSs possèdent également des SIS surveillant le respect des règles de sécurité. Ainsi, ces systèmes augmentent également la complexité des systèmes de contrôle-commande industriels. Enfin, cette dernière est accrue par l'hétérogénéité présente dans les ICSs comme les technologies utilisées (protocole, architecture, techniques) et les différents composants connectés, de constructeurs et de gammes différents. Cette hétérogénéité est accentuée par le fait que ces systèmes sont construits au fur et à mesure, chaque partie bénéficiant lors de sa mise en service des avancées technologiques et des normes proposées. Plusieurs éléments mettant en avant cette complexité sont repris et détaillés dans (Efthymiou et al. 2012).

Enfin, le bon fonctionnement d'un ICS repose sur la collaboration entre plusieurs acteurs tels que :

- Les opérateurs et les ingénieurs opérationnels dont l'objectif est d'assurer et de maintenir la productivité du système (qualité, surveillance),
- La Recherche & Développement dont la principale mission est d'améliorer le contrôle du procédé et qui travaille sur des problématiques amont,
- La maintenance, intervenant sur la réparation et la remise en production de l'ICS à la suite d'une défaillance, d'une casse ou d'un plan de maintenance,
- Les intégrateurs, car certaines fonctionnalités de l'ICS sont assurées par des prestataires extérieurs à l'entreprise comme la mise en service d'une partie du système de production, la télémaintenance des équipements de production ou d'autres fonctions au sein de l'ICS.

Face aux aléas de fonctionnement et à la complexité de l'environnement d'un ICS, différentes fonctions ont été créées et réparties dans chaque niveau du CIM permettant de surveiller, superviser et commander la partie opérative (Berruet et al. 2007). De nombreux travaux ont été proposés pour améliorer la détection, le suivi, le diagnostic et le pronostic, notamment ceux de (Pérès et Noyes 2003), (Iung et al. 2009) et (Iung et al. 2008) sur la maintenance des ICSs. Une analyse plus complète des enjeux liés à cette problématique peut être trouvée chez (Said 2016). L'objectif de ces fonctions est de garantir la fiabilité de ces systèmes telle que définie dans (Villemeur 1992), à savoir assurer la production coûte que coûte. Un état de l'art plus complet des approches de surveillance sera proposé par la suite. L'environnement dans lequel se trouvent les systèmes de contrôle-commande industriels est soumis à des perturbations qui peuvent avoir des conséquences négatives sur la productivité et/ou la sûreté de fonctionnement.

Toutefois, l'introduction de technologies héritées du milieu de l'IT a profondément modifié l'environnement des ICSs tel que décrit précédemment. Par exemple, le fonctionnement des API a été changé depuis les années 2000 passant de l'exécution de code interprété au compilé. La modification la plus importante concerne l'introduction du protocole Ethernet TCP/IP (CPNI 2011). Cette implémentation a permis d'améliorer les quantités de données transportées dans l'ICS. De plus, cette intégration a modifié l'environnement des systèmes de contrôle-commande en rendant possible les actions distantes comme la télémaintenance ou l'envoi d'ordre de fabrication. Toutefois, cette implémentation s'est faite sans prendre en considération la sécurité des systèmes équipés, introduisant une nouvelle cause de perturbation. Comme pour les défaillances, cette dernière a des conséquences sur les ICSs qui seront explicitées dans le paragraphe 5. Une solution originale pour protéger les systèmes de contrôle-commande industriels sera proposée dans la partie 2.

5. Étude des vulnérabilités d'un ICS

Dans le premier paragraphe, nous avons vu que les hackers ciblaient les ICSs depuis le début du siècle. De telles attaques sont possibles par les vulnérabilités présentes dans les composants des ICSs (hardware ou software) ainsi que dans l'architecture elle-même (Sicard, Zamaï, et Flaus 2018a) et qui sont répertoriées dans (ICS-CERT 2016c). Une vulnérabilité, également appelée menace dans ce manuscrit, est une faiblesse d'un système dans sa création ou sa configuration et qui permet de dégager des surfaces d'attaque pour s'introduire dans un système. Dans le cadre de cette thèse, nous nous focaliserons sur les vulnérabilités softwares et considérerons que les composants matériels sont exempts de vulnérabilités (absence de fonctions cachées par exemple). Afin d'évaluer les impacts, nous distinguons trois critères rangés par ordre d'importance : la disponibilité, l'intégrité et la confidentialité (ANSSI 2012). La Figure 10 et le Tableau 2 regroupent les menaces et leurs impacts qui sont détaillés dans (Sridhar, Hahn, et Govindarasu 2012), (McLaughlin et al. 2016) et (Graham, Hieb, et Naber 2016). De plus, des ouvrages permettent de compléter ces menaces comme (Fourastier et Pietre-Cambacedes 2015; Clarhaut et al. 2015; Knapp 2014). Bien entendu, cette hypothèse forte devra être à l'évidence levée. Des travaux relatifs aux techniques de protection basées sur des composants « secured by design » constituera une première piste de réflexion adossée à des techniques d'analyses de risques.

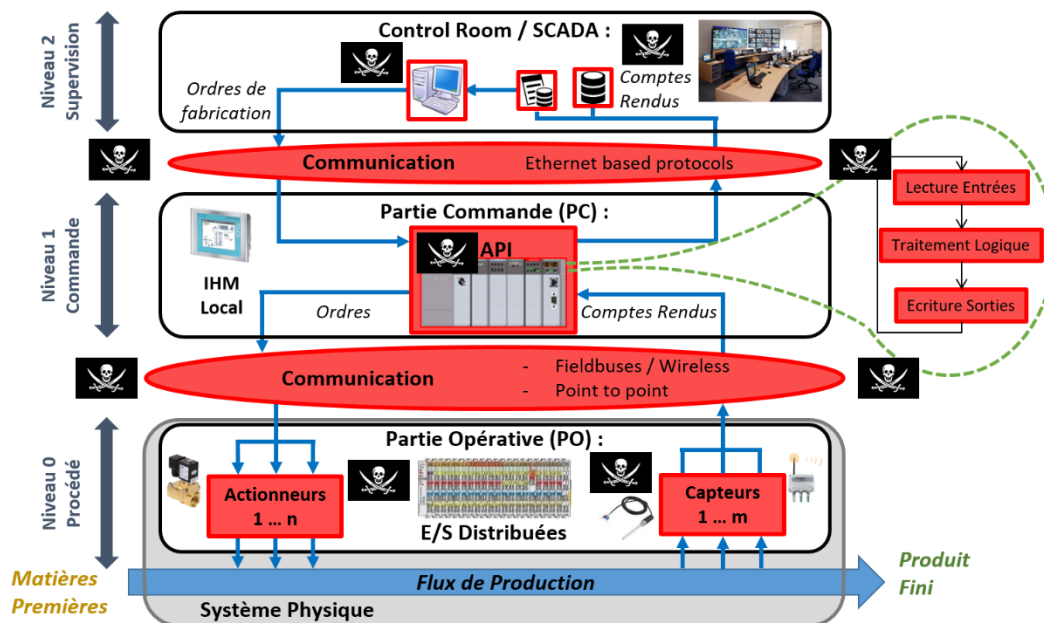


Figure 10 : Illustration des principales menaces et surfaces d'attaque (en rouge) dans un ICS (Niveau 2 - Supervision, Niveau 1 - Commande et Niveau 0 - Procédé)

En reprenant l'architecture d'un automate programmable industriel, décrit en Figure 9 dans le paragraphe 3.2.2, nous remarquons qu'aucun module n'est conçu pour assurer la sécurité de l'appareil. Ainsi, un attaquant peut corrompre un coupleur Ethernet pour accéder à l'API (W. Yang et Zhao 2014) ou alors infecter une console de programmation en attendant qu'elle se connecte à l'automate pour modifier le firmware (Ahmed et al. 2017), cette connexion se faisant sans authentification. Les travaux de (Milinković et Lazić 2012) reprennent les principales vulnérabilités d'un automate programmable industriel. De plus, (Cárdenas, Amin, et Sastry 2008) résument les principales vulnérabilités présentées précédemment en y ajoutant les conséquences sur un ICS. Ainsi, il apparaît que les menaces affectent des niveaux et des natures de composants différents. Les travaux de (McLaughlin et al. 2016) permettent de les identifier et d'indiquer les domaines d'investigations qui sont étudiés pour sécuriser les systèmes de contrôle-commande industriels. Ces pistes de recherche peuvent être classées en quatre catégories : logiciel, firmware, hardware et réseau. Les attaques exploitant la connaissance du procédé relèvent plus d'une philosophie d'attaques et représentent à elles seules un ensemble de menace pouvant être exploitées. Le chapitre 2 traitera des investigations menées sur les attaques utilisant la connaissance du procédé. Enfin, les travaux de (Rubio-Hernan, De Cicco, et Garcia-Alfaro 2016; Fourastier et Pietre-Cambacedes 2015) fournissent des informations complémentaires sur ces vulnérabilités et la manière de les utiliser.

5.1. Logiciel

Omniprésents dans les ICSs, les logiciels permettent de configurer différents éléments de ces systèmes tels que les équipements, les API, les IHM (locaux ou globaux) ou encore des serveurs de communication. Les vulnérabilités constatées vont de la simple erreur de code jusqu'à l'accès aux mécanismes de contrôle du logiciel (McLaughlin et al. 2016). Les vulnérabilités de ces composants conduisent le plus souvent à des alertes de type dépassement de tampon (buffer

overflow) dans (ICS-CERT 2016c). (Y. Yang et al. 2011) permet d'avoir un aperçu de ces vulnérabilités et de leurs impacts sur un ICS, notamment dans le cas des smart grids.

5.2. Firmware

Cette couche est à l'interface entre la partie logicielle et physique (hardware). Elle contient les données et instructions nécessaires pour piloter le comportement du composant (bootloader, fonctionnalités, ...). Cette couche gère également l'interface avec la couche logicielle permettant l'utilisation pour une application donnée (Basnight et al. 2013).

Dans le domaine d'application des ICSs, plusieurs travaux proposent des approches visant à sécuriser cette couche. Dans (Basnight et al. 2013), les vulnérabilités du firmware présentes dans les API sont identifiées notamment à l'aide d'une nouvelle méthodologie de reverse engineering, des contre-mesures sont ensuite mises en œuvre et implémentées au niveau du contrôleur. (W. Yang et Zhao 2014) mettent en évidence d'autres vulnérabilités et également leurs impacts en cas de modifications du firmware. L'identification d'attaques génériques utilisant le firmware fait l'objet des recherches de (Schuett, Butts, et Dunlap 2014). Les travaux de (Abbasi et Hashemi 2016) donnent un aperçu à la fois des attaques affectant le firmware, de la maturité de la littérature sur le sujet et des pistes en cours de développement. L'impact d'une attaque affectant le firmware d'un équipement est dévastateur pour l'ICS et son environnement comme décrit dans (Milinković et Lazić 2012). (Peck et Peterson 2009) montrent notamment comment prendre le contrôle d'un coupleur Ethernet, permettant la communication entre le PLC et le reste de l'ICS via le protocole Ethernet, en téléchargeant un firmware malicieux.

5.3. Hardware

Cette couche contient les composants électroniques embarqués, tels que les API ou les IHM, permettant d'exécuter physiquement ce qui est programmé dans la couche logicielle. Des attaques par injection de fautes peuvent être conduites et des portes cachées (backdoors) peuvent être introduites dans les composants. L'exploitation de ces vulnérabilités peut permettre à des attaquants de prendre le contrôle de l'équipement, de voler des informations ou d'interrompre le service fourni.

La littérature recense de nombreux travaux traitant de cette problématique regroupés dans (McLaughlin et al. 2016). (Papp, Ma, et Buttyan 2015) traitent des vulnérabilités et des menaces pesant sur ces systèmes notamment au niveau hardware. (Mohan et al. 2013), dont les travaux « S3A » portant sur la sécurisation des systèmes cyber-physiques, accordent une grande importance au niveau hardware. Enfin, (Ren, Blanton, et Tavares 2016) proposent une approche basée sur de l'apprentissage pour sécuriser le port JTAG. Cette interface est essentielle pour la programmation des composants électroniques et leurs tests. Toutefois comme ce port donne accès au firmware et à la mémoire embarquée dans la puce, ces vulnérabilités sont largement exploitées pour attaquer des composants. D'autres travaux traitant d'autres ports de communication sont disponibles dans (McLaughlin et al. 2016).

5.4. Réseau

Le rôle du réseau est d'assurer la communication entre les différents niveaux de l'architecture ICS comme défini dans chapitre 1 - 3.2.4 « Communication entre les niveaux ». Cependant, l'utilisation d'un réseau introduit de multiples vulnérabilités pouvant être exploitées comme décrit dans (Han et al. 2014) et (Gupta et Mo-Yuen Chow 2010). Les éléments de sécurité (firewall, etc.) (Bartman et Carson 2016), les protocoles réseaux (terrain, Ethernet) (Amoah, Camtepe, et Foo 2016), les moyens d'accès (modem, routeur, etc.) (Bonney et al. 2015) sont autant de failles potentielles. Les publications de (Slay et Miller 2006) et (Knapp 2014) permettent d'avoir également un aperçu de ces différentes vulnérabilités et de leurs répercussions. Des approches basées sur l'utilisation de systèmes de détection d'intrusion sont largement explorées dans la littérature (Sicard, Zamaï, et Flaus 2017b). Les axes de recherche sur cet aspect de la cybersécurité seront développés dans la partie suivante « Intrusion Detection System (IDS) dans les ICSs ».

5.5. Récapitulatif des vulnérabilités ICS

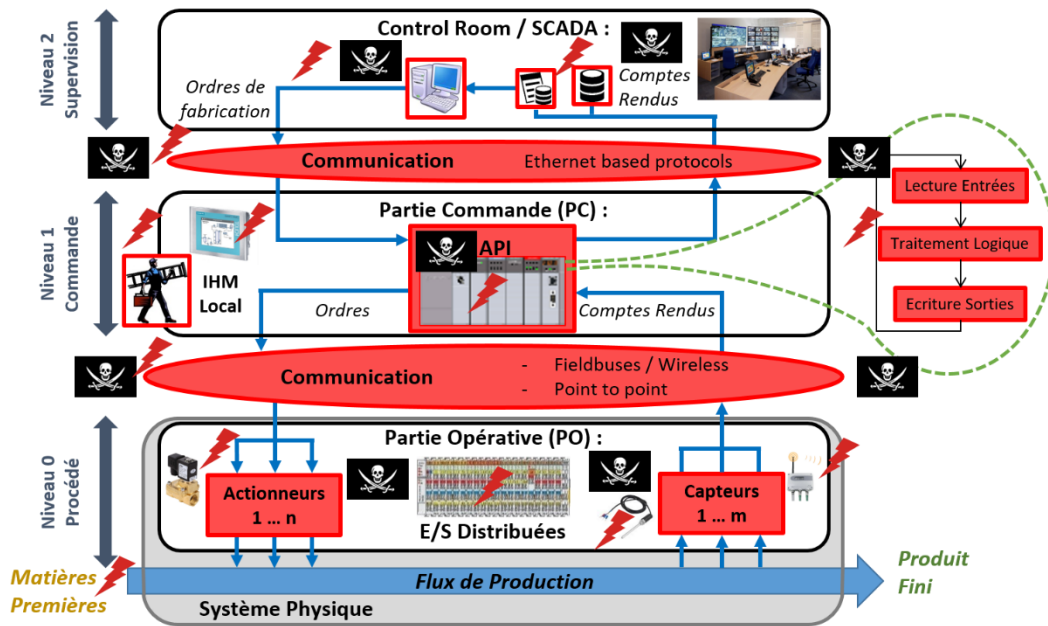


Figure 11 : Superposition des principales menaces et sources de défaillances dans un ICS (éclair rouge : défaillance, drapeau pirate : menace)

Cependant, en superposant les sources des menaces et des défaillances, nous remarquons des similitudes dans leurs localisations illustrées par la Figure 11. Nous mettons ici en évidence le fait que les conséquences d'une attaque (suivant son type) peuvent être identiques à celles d'une défaillance, à savoir la perte du service demandé. En effet, une défaillance de la partie opérative peut avoir plusieurs conséquences comme la non réalisation du service demandé (arrêt, maintenance), l'impact sur l'environnement du système industriel ou encore la propagation de la défaillance vers l'aval de la production. Une malveillance sur un ordre envoyé de la partie commande vers la partie opérative aura en ce sens le même effet puisqu'il ne correspondra pas la commande optimale. Toutefois, la différence fondamentale réside dans l'intentionnalité de cet

événement. En conclusion, au-delà de la protection de chaque élément composant les différents niveaux de l'ICS, l'environnement de programmation et de configuration doit être sécurisé d'un point de vue IT (console de programmation, ordinateur d'un intégrateur ou d'un opérateur de maintenance). Cependant, les ICSs présentent des spécificités par rapport aux systèmes d'information « traditionnels » et demandent une culture de la sûreté spécifique pour traiter la problématique imposée par les cyberattaques.

Tableau 2 : Principales menaces dans un système de contrôle-commande industriel avec leurs localisations dans l'architecture ICS et les références correspondantes

Localisation	Menace	Effets	Références
Niveau 2 SCADA	Injection de Code	Intégrité	(Bartman et Carson 2016) (Dzung et al. 2005) (Babu et al. 2017)
	DOS / DDOS	Disponibilité	
	Vols de données	Confidentialité	
	Buffer Overflow	Disponibilité	
Niveau 2 Supervision / IHM	Diverses attaques (virus, ver informatique, malware, ransomware...)	Intégrité Disponibilité Confidentialité	(Falliere, Murchu, et Chien 2011) (ICS-CERT 2017) (Bencsáth 2012) (ICS-CERT 2013)
Communication Niveaux 2 - 1	Modification de données	Intégrité	(Zhu et Sastry 2010)
	Interruption de la communication	Disponibilité	
Niveau 1 Automate Programmable Industriel	Modification du firmware	Intégrité	(McLaughlin et al. 2016) (Wardak, Zhioua, et Almulhem 2016) (Basnight et al. 2013)
	Buffer Overflow	Disponibilité	(ICS-CERT 2015) (ICS-CERT 2016b)
	Modification de la configuration (logiciel, coupleur)	Intégrité	(McLaughlin et al. 2016) (Wardak, Zhioua, et Almulhem 2016) (Milinković et Lazić 2012) (Abbasi et Hashemi 2016) (W. Yang et Zhao 2014)
	Altération de la loi de commande	Disponibilité	(McLaughlin et al. 2016) (Falliere, Murchu, et Chien 2011) (Wardak, Zhioua, et

			Almulhem 2016) (McLaughlin 2015)
	Attaque de la mémoire	Intégrité	(McLaughlin et al. 2016) (Ahmed et al. 2017) (Abbasi et Hashemi 2016)
Communication Niveaux 1 – 0	Modification des ordres / comptes rendus	Intégrité	(Wardak, Zhioua, et Almulhem 2016) (McLaughlin 2013)
	Interruption de la communication	Disponibilité	
Niveau 0 Capteurs / Actionneurs	Modification et altération de données	Intégrité	(Ashibani et Mahmoud 2017) (He et Yan 2016) (Lu et al. 2015) (Sandberg et Hunter 2017)

6. Bilan du chapitre

Ce premier chapitre a permis de mettre en évidence le risque cyber qui s'est accentué depuis le début du 21^{ème} siècle et affecte désormais les systèmes de contrôle-commande industriels. De par leur conception et leur environnement, ces systèmes présentent des vulnérabilités qui offrent des surfaces d'attaques à des hackers. L'État Français a réagi en créant une agence dédiée à la cybersécurité des OIV opérant dans des SAIV. La nouvelle LPM durcit les conditions pour ces acteurs vitaux à la nation en les forçant législativement à se mettre aux normes par rapport aux préconisations de l'ANSSI. Cette thèse se place dans le cadre de la surveillance des installations et de la détection des actes de malveillance. L'étude des vulnérabilités a mis en évidence la nécessité de sécuriser les différents niveaux de l'architecture ICS. Toutefois, des approches privilégiant l'utilisation de la connaissance autour de ces systèmes seront présentées dans le prochain chapitre consacré aux techniques de détection existantes dans la littérature pour identifier des intrusions ou des défaillances.

Chapitre 2

Etat de l’art des approches de surveillance et de détection d’attaques dans les ICSs

Ce chapitre est consacré à l’étude bibliographique de la cybersécurité dans les systèmes de contrôle-commande industriels. Dans un premier temps, nous identifions les différents champs d’investigations présents dans la littérature pour sécuriser ces systèmes vis-à-vis de la menace cyber. Par la suite, nous nous focalisons sur une technique de détection d’intrusion en particulier : les Intrusion Detection Systems (IDS). A l’origine créés pour des systèmes d’information, plusieurs travaux ont permis de les adapter aux ICSs. Enfin, nous terminons cette première partie de l’état de l’art avec des approches de sécurité à base de modèles de la partie opérative. Afin de compléter l’étude bibliographique, nous analysons ensuite les approches orientées sûreté de fonctionnement. En effet, le problème de la détection d’intrusions se rapproche de celui rencontré dans les approches de surveillance dans les ICSs.

Les différents apports bibliographiques sont localisés dans l’architecture ICS afin d’identifier les bénéfices mais également les lacunes des approches actuelles. Enfin, nous discutons de la complémentarité entre les concepts de sûreté et de sécurité pour les systèmes de contrôle-commande industriels afin d’hybrider ces approches.

Table des matières

1.	Cybersécurité dans la littérature	46
2.	Intrusion Detection System (IDS) dans les ICSs	48
2.1.	Généralité sur les IDSs	48
2.2.	Utilisation des IDS pour la sécurisation des ICSs	49
2.2.1.	Méthode de détection	49
2.2.2.	Source de données	50
2.3.	État de l’art des IDS dans les systèmes industriels	50
3.	Étude des IDS orientés connaissance du procédé	52
4.	Étude des approches orientées sûreté de fonctionnement	57
5.	Bilan des approches Sûreté/Sécurité	62
6.	Bilan du chapitre	63

1. Cybersécurité dans la littérature

La cybersécurité est un domaine qui, aujourd'hui, est riche de propositions pour protéger les systèmes d'information dont les techniques ont commencé à être élaborées depuis la fin du 20^{ème} siècle. L'utilisation de ces approches pour sécuriser les ICSs a été expérimentée comme le montrent (Slay et Miller 2006). Nous pouvons retenir les solutions suivantes représentées en Figure 12 :

- Pare-feu (firewall). Il s'agit d'un moyen de protection physique et/ou logiciel pour sécuriser la connexion d'un réseau interne à un réseau externe (généralement internet). Ce séparateur permet un filtrage des paquets entrants en se basant sur des règles garantissant le respect de la politique réseau (Maheshwari et Dagale 2018). Différentes analyses des trames existent (Nurika et al. 2012), elles peuvent être statiques basées sur des règles simples comme une liste d'adresses autorisées ou de ports de communication (couche réseau), dynamiques basées sur l'étude des séquences de paquets (couche réseau), ou applicatives, au sens du modèle OSI (couche applicative) fondées sur la compréhension des données transportées pour garantir leur intégrité (protocole, ports, ...).
- DMZ (Demilitarized Zone). Une zone démilitarisée est constituée d'un ou deux pare-feux permettant d'isoler un sous-réseau (zone en pointillés) des réseaux dont elle dépend (réseau A et B). Une DMZ est une zone tampon entre le réseau à protéger (réseau B) et le réseau externe (réseau A) (Barker et Cheese 2012; Hu, Hu, et Chen 2015).
- Diode de données (data diode). Cette technologie permet de créer un canal de communication unidirectionnel entre deux réseaux qui est garanti physiquement (Barker et Cheese 2012). Le flux de données ne peut se transférer dans le sens contraire ce qui permet de sécuriser la communication d'un réseau à un autre et d'assurer l'intégrité des données (Jeon et Na 2016).
- Antivirus. Les antivirus sont des solutions exploitant les ressources de l'hôte (machine) sur lesquelles ils sont déployés (McIntyre et al. 2012). Le principe de fonctionnement est basé sur l'analyse de signatures virales alimentées par une base de données et, plus récemment, sur les méthodes heuristiques qui utilisent des algorithmes recherchant des comportements définis comme malveillants et en analysant le code du programme inconnu.
- Chiffrement. Cette technologie consiste à rendre impossible la compréhension d'une communication par le biais d'une clé de chiffrement. La cryptographie symétrique permet de chiffrer et déchiffrer un message là où le chiffrement asymétrique nécessite une clé publique et privée ainsi qu'une authentification de l'expéditeur (Fauri et al. 2017).
- IDS (Intrusion Detection System). Cette technologie est basée sur des sondes déployées dans le réseau d'un SI. Ces dernières captent et analysent des informations échangées avec des règles garantissant une bonne politique réseau (Denning 1987).

Tout d’abord utilisée pour sécuriser les architectures IT, cette technique a été appliquée aux ICSs (Zhu et Sastry 2010) et sera développée dans le paragraphe Chapitre 3 - 2 Intrusion Detection System (IDS) dans les ICSs.

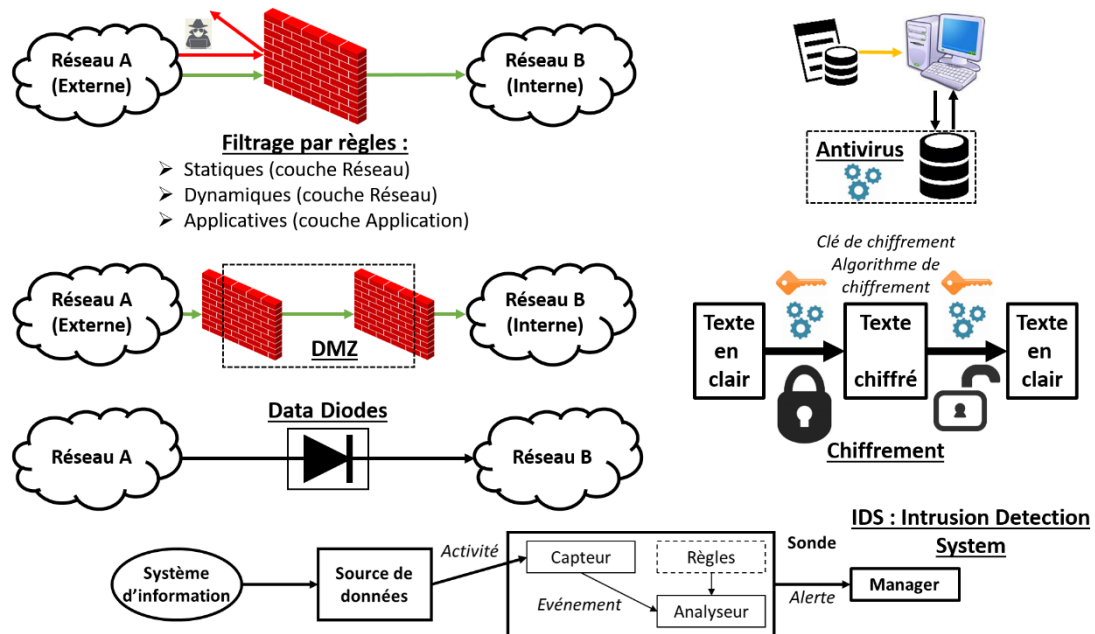


Figure 12 : Illustration du principe de fonctionnement des principales solutions de sécurité utilisées dans l'IT

Toutes ces solutions ne sont pas applicables pour sécuriser les niveaux 1 – 0 des ICSs. Ainsi le chiffrement des communications n’est pas applicable dans toutes les couches temps-réel à cause de la capacité de calcul limitée et du besoin de disponibilité. Les solutions antivirus nécessitent, quant à elles, une mise à jour régulière de la base virale pour conserver leur efficacité ce qui s’avère pénalisant dans un contexte de production. De plus, elles utilisent des algorithmes complexes basés sur des heuristiques nécessitant des puissances de calculs élevés et non disponibles dans les calculateurs implantés dans les ICSs. Les DMZ et les diodes de données sont utilisées pour sécuriser les niveaux 2 et supérieurs des ICSs, toutefois, leurs conceptions ont été pensées pour sécuriser les réseaux informatiques et non la protection des parties opératives et commandes (notamment vis-à-vis de la disponibilité). Les pare-feux semblent efficaces mais cette technologie est basée fortement sur une analyse du réseau et l’utilisation de règles applicables sur un réseau SI non adaptées aux contraintes d’un réseau industriel (bande passante, disponibilité, intégrabilité dans l’architecture matérielle). La formation des opérateurs, les plans de sauvegarde et la sécurisation des accès physiques ont besoin de temps pour devenir efficaces. Ainsi, les solutions précédentes ne sont pas retenues puisqu’elles n’apportent pas de solutions pour sécuriser les niveaux 1-0 (DMZ, diode de données) ou leurs utilisations sont limitées par les capacités de calculs (chiffrement, antivirus). Les pare-feux sont écartés car leur fonctionnement repose sur des règles pouvant être contournées, et sur le filtrage du réseau (au niveau de la couche réseau ou applicative) ne permettant pas l’utilisation de modèles comportementaux. La technologie IDS, elle, est intéressante et adaptable à l’environnement complexe des systèmes de contrôle-commande industriels. Il s’agit d’un système complémentaire aux pare-feux et DMZ signalant l’intrusion dans un système connecté en réseau. De plus, les IDSs permettent d’utiliser la

compréhension du fonctionnement d'un ICS, la connaissance « métier », pour proposer des solutions de sécurisation innovantes. Ces axes, développés dans le chapitre 3 - 3, concernent la prévention par l'analyse conjointe de la sécurité et de la sûreté (Siwar Kriaa et al. 2015), les outils d'aide à la sécurisation (Igor Nai Fovino, Masera, et De Cian 2009) et la réaction face aux attaques en développant des approches hybrides (Sicard, Zamaï, et Flaus 2018a). Le paragraphe suivant est consacré à l'étude de la littérature sur l'utilisation de cette technique dans les ICSs.

2. Intrusion Detection System (IDS) dans les ICSs

Le principe de fonctionnement des systèmes de détection d'intrusions (IDS) est intéressant pour la sécurisation des ICSs comme conclu dans le paragraphe précédent Chapitre 2 -1. Cybersécurité dans la littérature. Dans un premier temps, nous détaillons cette technologie, puis nous identifions les travaux utilisant cette technique pour développer des approches sécurisant les ICSs. Ces dernières sont fortement basées sur l'analyse du réseau correspondant au quatrième axe d'investigation énoncé par (McLaughlin et al. 2016).

2.1. Généralité sur les IDSs

Dans le domaine de recherche de la sécurité informatique, les systèmes de détection d'intrusions (IDS) sont largement référencés pour la détection d'attaques contre les systèmes d'information et les réseaux de communication. Le cadre d'utilisation de cette technique a été pour la première fois énoncé par (Denning 1987) et les IDSs sont encore employés de nos jours pour sécuriser les SI. L'IDS est une mesure de sécurité a posteriori puisque la détection a lieu après une intrusion. Le système identifie automatiquement les violations de la politique réseau en se basant sur les critères classiques : confidentialité, intégrité et disponibilité des données. Pour ce faire, les IDSs utilisent des sondes qui sont déployées au travers du réseau à protéger pour acquérir des données. Les données sont ensuite analysées en fonction des règles garantissant le respect de la politique réseau. En cas de détection, une alerte est envoyée aux opérateurs du système. Ainsi, une réflexion sur la localisation des sondes dans le réseau est essentielle pour assurer un bon taux de détection en minimisant le nombre de sondes. La Figure 13 illustre le principe de fonctionnement d'un IDS.

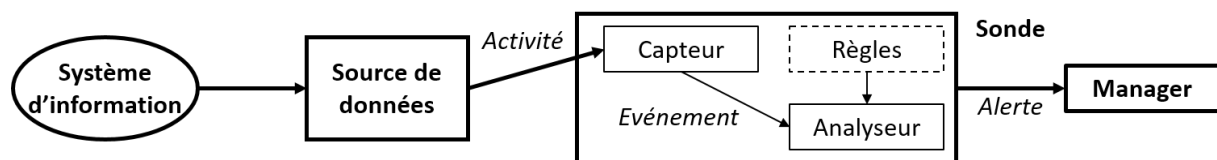


Figure 13 : Principe de fonctionnement d'une sonde IDS

L'évaluation des performances d'un IDS se base sur quatre indicateurs (Kabir-Querrec 2017) :

- Le nombre de faux positifs (FP), le nombre d'événements normaux identifiés comme anormaux, il s'agit de fausses alarmes.
- Le nombre de faux négatifs (FN), le nombre d'événements anormaux classifiés comme normaux.

- Le nombre de vrais positifs (VP), le nombre d’événements anormaux vrais par rapport au nombre d’intrusions détectées.
- Le nombre de vrais négatifs (VN), le nombre d’événements normaux vrais par rapport au nombre de non intrusions détectées.

Pour chaque situation, normale ou malicieuse, l’IDS peut identifier correctement ou non l’événement. Ainsi, dans le cas d’une situation normale où l’IDS détecte un événement :

- Anormal, nous parlerons de faux positifs.
- Normal, nous parlerons de vrais négatifs.

Dans le cas d’actions malicieuses, lorsque l’IDS détecte un événement :

- Malicieux, nous parlerons de vrais positifs.
- Normal, nous parlerons de faux négatifs.

L’efficacité de l’IDS se mesure sur le calcul de deux indicateurs :

- Taux de vrais positifs (TVP) indiquant la fiabilité de l’IDS obtenu par le rapport $\frac{VP}{VP+FN}$
- Taux de faux positifs (TFP) indiquant la pertinence de l’IDS obtenu par le rapport $\frac{FP}{FP+VN}$

Les performances de l’IDS sont jugées par la comparaison du TVP avec le TFP qui correspond à la courbe de sensibilité par rapport à la spécificité.

2.2. Utilisation des IDS pour la sécurisation des ICSs

Le principe de détection des IDS est propice à une adaptation aux systèmes de contrôle-commande industriels. En effet, les sondes, à partir du moment où elles intègrent les protocoles requis, peuvent être déployées indifféremment sur un réseau bureautique (IT) ou industriel (OT) puisque ces systèmes se basent sur les données acquises de l’environnement à protéger. Cet intérêt pour les IDSs a été formalisé dans (Mitchell et Chen 2014) où deux caractéristiques sont à distinguer : la méthode de détection et la source de données.

Les diverses approches qui sont présentées dans les parties suivantes sont évaluées sur les critères suivants : couverture face aux attaques affectant les niveaux 1-0 des ICSs, intégrabilité dans l’architecture actuelle des systèmes industriels et possibilité de construction de modèles déterministes pour analyser les informations à l’aide de modèles comportementaux.

2.2.1. Méthode de détection

Deux grandes approches peuvent être identifiées pour détecter des intrusions dans l’environnement protégé (Evangelista 2004). D’une part, certains IDS repèrent des déviations à partir d’un modèle de comportement normal du système surveillé, il s’agit des approches comportementales. D’autre part, les autres IDS s’appuient sur une base de connaissances de comportements anormaux, nous parlerons alors d’approches par signature d’attaques. Les approches comportementales peuvent être réalisées à l’aide de techniques :

- Probabilistes basées sur des modèles supervisés par un expert (modèles statistiques) ou obtenus à partir d'algorithmes d'apprentissage,
- Déterministes basées sur la spécification du comportement normal du système.

Indéniablement, l'avantage de l'approche comportementale fondée sur des modèles du comportement normal réside dans sa capacité à détecter des attaques inconnues (dites attaques *zero-day*). Toutefois, plusieurs inconvénients existent dans cette approche notamment le taux important de fausses alertes, dû à la difficulté de représenter parfaitement le comportement normal exact du système (le modèle n'étant qu'une abstraction de la réalité). De plus, le comportement du système peut évoluer avec le temps ce qui introduit un biais dans cette approche pouvant être utilisé par l'attaquant pour contourner les protections. Un autre inconvénient réside dans le peu d'adéquation avec la phase de diagnostic, nécessitant généralement une prise de relais par un opérateur humain. A l'opposé, l'approche par signature génère beaucoup moins de faux positifs puisque l'IDS recherche un motif (pattern) pour identifier les actions suspectes. De même, le diagnostic sera immédiat puisqu'une détection est caractérisée par une signature particulière reliée à une attaque unique. En revanche, cela suppose que les signatures soient pertinentes (compromis spécificité/détectabilité dans l'écriture des règles) et que la base de connaissances soit mise à jour régulièrement (détection de nouvelles attaques).

2.2.2. Source de données

Pour assurer leur mission de détection, les IDSs peuvent être de deux types :

- NIDS (Network Intrusion Detection System), l'IDS se base sur l'étude des données du réseau comme les trames, les en-têtes de paquet ou encore les données transportées,
- HIDS (Host Intrusion Detection System), l'IDS utilise les données de l'hôte rendues disponibles par les applications ou le système d'application.

Les inconvénients majeurs des HIDS sont l'impact sur les ressources mémoires et le manque de généralités tandis que ceux des NIDS résident dans la capacité à analyser le trafic (chiffrement, niveau d'analyse, temps de réponse exigé). Toutefois, un des avantages de l'approche NIDS est la passivité et l'intégrabilité dans une architecture existante sans perturber le fonctionnement de cette dernière.

2.3. État de l'art des IDS dans les systèmes industriels

Selon (Mitchell et Chen 2014), un IDS capable de s'adapter aux ICSs doit pouvoir : surveiller le processus physique et les lois de commande, mettre à profit la régularité et la prédictibilité des ICSs, faire face à des attaques complexes et gérer les technologies existantes. Dans la littérature, plusieurs approches se distinguent pour utiliser les IDSs orientés ICS.

Certains travaux se concentrent sur l'utilisation des composants dotés de capacités de calculs dans les ICSs afin d'utiliser des HIDS pour détecter des anomalies. Ces approches restent néanmoins très minoritaires par rapport à l'utilisation de NIDS. Ceci s'explique notamment par la spécificité du matériel utilisé dans les ICSs, la disponibilité et la capacité de calcul de ces ressources ainsi que de la vulnérabilité intrinsèque de ces approches (IHM, API, ...). Nous pouvons néanmoins citer les travaux de (Zimmer et al. 2010) qui développent une méthode de détection

d'intrusions basée sur l'analyse du temps d'exécution des tâches dans les composants critiques des ICSs soumis aux contraintes temps-réel. Trois techniques ont été mises en œuvre dans cette publication se basant sur des appels synchrones et asynchrones afin de comparer les temps d'exécution du code. L'objectif principal de cette approche est de détecter des injections de code de la part d'un attaquant. Ainsi, l'IDS surveille le temps d'exécution de la tâche en cours. L'inconvénient principal de cette approche est la compatibilité avec les contraintes temps-réel puisque du code doit être implémenté dans les applications afin de générer des appels. (Rrushy et Kang 2009) modélisent l'évolution des variables mémoires de l'API en fonction des paramètres du processus physique contrôlé. L'IDS tire ainsi parti de la régularité pouvant être présente dans les ICSs pour représenter les interactions entre l'automate et le processus physique. L'objectif de cette méthode est de classer les paquets reçus en fonction de l'effet induit sur la mémoire de l'API. Une approche statistique est utilisée dans cette approche qui montre des limites lorsque le système contrôlé ne présente plus de périodicité. (Reeves et al. 2012) présentent un outil capable de détecter un type particulier de détournement de pointeurs pouvant faire ainsi appel à une fonction malicieuse. Enfin, (Bellettini et Rrushy 2008) ont étudié un mécanisme de détection d'accès incorrect à la mémoire de l'hôte protégé. L'approche se base notamment sur l'étude du code utilisé pour implémenter les protocoles de type Modbus et OPC. Les approches IDS basées hôtes ne sont pas retenues puisqu'elles s'axent principalement autour d'algorithmes heuristiques analysant les données contenues dans l'hôte. Comme nous le verrons plus tard dans le manuscrit, bien que les approches HIDS soient intéressantes pour une extension des travaux développés, notre axe de recherche se situe sur les approches comportementales.

D'autres approches se basent sur l'analyse du réseau pour développer des mécanismes de détection et représentent la grande majorité des publications de la littérature. L'étude de la structure des trames échangées sur le réseau industriel fait l'objet de travaux comme ceux de (Cheung et al. 2007) où des spécifications sont formulées sur les codes fonctions, exceptions ou identificateurs pouvant être utilisés. Ainsi, une partie ou l'ensemble d'un réseau peut ne bénéficier que d'un sous-ensemble de fonctions offertes par le protocole utilisé. Ces travaux se sont basés sur le protocole Modbus mais l'équivalent a été développé par (Lin et al. 2013) sur le protocole DNP3. Les travaux de (Goldenberg et Wool 2013; Kleinmann et Wool 2014) proposent de modéliser le trafic entre l'API et l'IHM sans prendre en compte les données transportées dans les trames (sémantiques). Ainsi, un automate fini permet de représenter l'évolution des échanges en se basant sur l'hypothèse de forte périodicité. Toutefois, ces approches restent vulnérables aux attaques respectant le formalisme des échanges. De plus, cette approche n'est pas applicable lorsque l'on cherche à protéger des systèmes ne présentant pas des cyclicités explicites. Enfin, (Morris, Vaughn, et Dandass 2012) proposent de convertir le protocole terrain en trafic TCP afin de l'analyser avec des sondes SNORT configurées avec des règles respectant la politique réseau.

Certains travaux exploitent les données caractérisant une communication pour des calculs statistiques dans le but de détecter des intrusions sur le réseau industriel telles que le temps de réponse à une requête, la quantité de paquets échangés et perdus ou encore l'intervalle temporel entre deux trames. Ainsi, les travaux de (Ponomarev et Atkison 2016; D. Yang, Usynin, et Hines 2006) cherchent à caractériser le comportement normal de l'ICS à l'aide de l'apprentissage des échanges réseaux afin d'établir un profil de trafic. Ces approches perdent en pertinence sur la

protection des ICSs puisqu'elles ne tiennent pas compte de la signification des données transportées. De plus, seules les attaques par déni de service sont considérées dans ces travaux. (Barbosa, Sadre, et Pras 2012, 2016) ont développé des algorithmes de détection en s'inspirant de techniques utilisées en traitement du signal. En exploitant l'hypothèse de périodicité des ICSs, les auteurs caractérisent la fréquence d'échange et le nombre de trames échangées afin d'identifier des intrusions. Une analyse spectrale à l'aide de transformée de Fourier permet de détecter des changements suspects dans l'évolution du trafic. Enfin, (Linda, Vollmer, et Manic 2009) ont proposé une approche par apprentissage en utilisant des réseaux de neurones afin de caractériser la structure des échanges (adresse IP, nombre de paquets échangés, intervalle entre deux trames, protocoles, ...). Toutefois, les paramètres devant être choisis pour assurer une bonne détection ne sont pas génériques.

Enfin, un dernier axe de recherche met à profit les informations issues de la communication comme les adresses sources et de destinations du flux, les ports utilisés ou encore le protocole support. Les travaux de (Barbosa, Sadre, et Pras 2013) vont dans ce sens puisqu'ils proposent une approche basée sur une liste blanche issue de la modélisation du comportement normal du réseau caractérisé par les adresses du client et du serveur, le port de communication et le protocole utilisé. Cette modélisation se fait par apprentissage. La limite principale réside dans la gestion de l'allocation dynamique des ports, largement démocratisée, et qui fait exploser la taille de la liste. (Hadeli et al. 2009) proposent d'exploiter les fichiers de configuration présents dans les ICSs pour identifier les informations nécessaires à la construction de règles. Enfin, (Cheung et al. 2007) proposent un algorithme d'apprentissage bayésien afin d'identifier les différents composants qui communiquent ainsi que les codes fonctions utilisés. Ainsi, toute connexion non validée par le modèle est considérée comme suspecte.

Nous noterons également que plusieurs des travaux cités mixent les axes de recherche afin d'améliorer les mécanismes de détection. La force des NIDS réside dans leurs intégrabilités dans l'architecture ICS. Toutefois, nous remarquerons que l'hypothèse fondamentale de ces travaux réside en la forte périodicité des systèmes contrôlés qui ne peut pas être toujours assurée dans les ICSs. De plus, les approches NIDS détaillées précédemment ne tiennent pas compte de la signification des données transportées sur le processus physique. Or, notre approche souhaite se baser sur des modèles comportementaux ce qui n'est pas compatible avec les approches IDS orientées réseau. Force est de constater que l'analyse « traditionnelle » du trafic n'apporte pas de réponses satisfaisantes, nous détaillerons dans la suite des approches NIDS intégrant la compréhension comportementale du procédé.

3. Étude des IDS orientés connaissance du procédé

Sur l'ensemble des travaux répertoriés dans la littérature, plusieurs axes de recherche ont pu être dégagés. Nous retiendrons les grandes catégories d'études portant sur :

- L'identification des variables du système à partir des échanges,
- La protection par la spécification d'états critiques pouvant être obtenue par apprentissage ou connaissances expertes,

- La réaction face à des attaques prédéterminées obtenues à partir de séquences d’observation,
- La sécurisation des ICSs face aux attaques par injection de données,

L’ensemble des approches énoncées par la suite repose sur une modélisation du système (approche comportementale) à l’aide d’un ensemble de règles, de représentations quantitatives (équation, représentation d’état, fonction de transfert) ou qualitative (graphe) ou encore d’un apprentissage.

(Hadziosmanovic et al. 2013) cherchent à identifier les variables du processus physique à partir des valeurs extraites des paquets analysés. La caractérisation est réalisée à partir d’une méthode heuristique qui permet d’identifier les variables dont la valeur : change en permanence (capteurs), est comprise dans un ensemble fini (variable interne à l’API) ou constante. Deux mécanismes sont ensuite construits en se basant sur les bornes maximales et minimales des variations observées ainsi que sur un modèle autorégressif prédisant les prochaines valeurs. Cette approche souffre de la pertinence de l’identification des variables puisque l’algorithme d’apprentissage ne permet pas une caractérisation exacte. Dans le même axe de recherche se trouvent les travaux de (Erez et Wool 2015) qui cherchent à modéliser l’évolution des registres de l’API. L’apprentissage se base sur une approche statistique sur des fenêtres temporelles. Là encore, un registre est caractérisé selon plusieurs catégories (variable de mesure, de compteurs ou constante). La détection s’effectue par comparaison des valeurs de registre entre deux fenêtres temporelles plus importantes (24h). Ces travaux se basent sur l’hypothèse fondamentale de forte périodicité des ICSs.

D’autres travaux introduisent la notion d’analyse d’états critiques pour développer des algorithmes de détection. Cette méthode s’appuie sur la spécification des états critiques possiblement atteignables par le système et aboutit à définir des intervalles de valeurs dans lesquels la variable peut évoluer. Contrairement à d’autres approches comme (Cardenas, Roosta, et Sastry 2009) ou (Papa, Casper, et Nair 2012), la dynamique du système n’est pas modélisée. Ainsi, dans les travaux de (Andrea Carcano et al. 2009) et (I. N. Fovino et al. 2010), le système est préalablement analysé pour identifier les états critiques. Un état est caractérisé par les valeurs des capteurs et des actionneurs connectés à l’automate. Lorsque le système se rapproche d’une transition vers un état critique ou de l’occurrence d’un tel état alors l’IDS lève une alerte. Il s’agit bien d’une approche préventive puisque l’IDS est placé en surveillance passive entre la supervision et l’API ou alors lance des requêtes à destination de l’API. Ces travaux prennent pour hypothèse que l’ensemble des états critiques est connu et qu’il reste de taille raisonnable, que l’attaquant cherche à provoquer des dommages sur le système physique, que ce dernier utilise les actionneurs en sortie de l’API (*commande « légitime »*) pour réaliser son attaque et qu’aucune confusion n’est possible entre un état critique provoqué par une attaque et les défaillances ou les attaques physiques. Le mécanisme de détection développé s’appuie sur des règles, écrites à l’aide du langage ISML, qui garantissent le bon fonctionnement du système. Les apports de ces travaux sont importants notamment sur la notion d’états critiques et sur la prise en compte du fonctionnement d’un ICS. Toutefois, la détection à base de règles est limitée par leur écriture (oubli, rapport spécificité/détection). Enfin, le placement de l’approche au niveau communication

entre les couches 2 et 1 laisse encore une surface d'attaque importante aux hackers pour s'introduire dans le système comme en corrompant l'API par exemple. Cet inconvénient est compensé par le fait que l'approche de (Andrea Carcano et al. 2009) et (I. N. Fovino et al. 2010) cherche à corréler les différentes alertes et nécessite donc une vision globale des données. Avec une localisation similaire, nous citerons les travaux de (Koucham et al. 2016) qui analysent les échanges réseaux afin de vérifier des spécifications représentées par des ensembles de propriétés de sécurité temporelle sur les états et les événements correspondant aux capteurs et actionneurs. Les travaux de (Toublanc et al. 2017) portent sur une approche similaire mais sur les niveaux 1 – 0. Les propriétés sont ensuite synthétisées sous forme de moniteurs qui signalent les violations sur les traces d'exécution. Enfin, (Mitchell et Chen 2013) ont développé un IDS avec des règles comportementales sur les capteurs et les actionneurs. L'analyse des états du système est plus poussée dans le sens où un état peut être de type : sécurisé, d'avertissement ou non sécurisé. Le comportement du système est décrit sous forme d'un automate fini probabiliste où les probabilités de transition dépendent du type d'attaque. Un degré de conformité permet d'identifier un comportement normal d'un scénario d'attaque. L'inconvénient majeur de cette approche demeure le seuillage définissant le « normal » et qui est défini de manière empirique. En complément des travaux précédemment cités, (Xiao et al. 2007) ont proposé de construire un simulateur afin d'évaluer la dangerosité d'une commande selon des modèles d'attaque prédéfinis.

Les travaux de (Caselli, Zambon, et Kargl 2015) s'intéressent aux séquences d'observation du système contrôlé dans le but d'identifier la connaissance métier pour contrer les attaques de séquence. Comme défini dans le « Chapitre 1 - 1.4 Définition d'une attaque », ces attaques ne cherchent pas à endommager le système en premier lieu mais plutôt à modifier l'évolution des actions, individuellement légitimes, à mener sur le processus physique pour le dégrader progressivement : modification de l'ordre des actions ou modification du timing de la commande. L'approche se base sur les chaînes de Markov pour modéliser l'ICS avec une méthode d'apprentissage. En phase de détection, un autre modèle est construit en ligne et comparé avec le modèle de référence. Des anomalies traduisent des attaques pouvant affecter le séquençement ou les contraintes temporelles de la commande. En complément, les travaux de (Pan et al. 2013) s'intéressent à la détection d'intrusions dans les systèmes de distribution d'énergie en se basant sur des scénarii d'attaques identifiés en amont (approche par signature). L'applicabilité de cette approche est remise en question lorsque des signatures ressemblent fortement à des comportements normaux ce qui génère une grande proportion de faux positifs obtenus dans l'expérimentation.

Les travaux présentés dans la suite se sont focalisés sur la détection d'attaques injectant des fausses données dans le réseau de communication de l'ICS. (Wang et al. 2014) proposent d'analyser les relations entre variables, qui seront modélisées à l'aide d'un graphe, sans connaissance explicite sur le processus physique. Le graphe permet d'avoir une corrélation entre les différentes variables à l'issue d'une phase d'apprentissage et de détecter des transitions ou des états inconnus. Toutefois, cette approche a été expérimentée uniquement sur une plateforme spécifique aux injections de fausses données ce qui ne permet pas d'appréhender le comportement de l'approche sur un cas plus général. Les travaux de (Cardenas, Roosta, et Sastry 2009) puis de (Urbina, Giraldo, Cardenas, Tippenhauer, et al. 2016) vont dans le sens contraire puisqu'ils

mettent en évidence la nécessité de prendre en compte les interactions entre le système de commande et le système physique. La détection est axée sur le déni de service et la corruption des mesures envoyées par les capteurs notamment, en comparant les mesures reçues avec celles prédites obtenues avec un modèle précis du système. Ces travaux ont notamment été à la base d’une publication sur les perspectives de développement des approches « *physic-based* » dans les ICSs (Urbina, Giraldo, Cardenas, Valente, et al. 2016). Similairement, les travaux de (Sridhar et Govindarasu 2014) portent sur la détection d’intrusions comportementales dans les systèmes de régulation électriques. La détection se base sur l’étude des mesures que l’attaquant cherchera à modifier pour dégrader le système. L’IDS calcule donc, pour chaque système de régulation, la fréquence mesurée et la puissance et les compare avec des seuils d’alerte tenant compte d’une erreur de contrôle. L’approche permet de caractériser des déviations dans la prédiction des erreurs de contrôle (obtenues statistiquement) et des erreurs temporelles. Un des apports majeurs de ces travaux se situe dans la variété des modèles d’attaques pris en compte (brusque, impulsion, modification de la pente, aléatoire). L’inconvénient de cet algorithme de détection est l’absence de formalisme pour fixer la valeur des seuils. Enfin, (Papa, Casper, et Nair 2012) proposent un IDS basé sur une fonction de transfert. Cette dernière permet de représenter les relations entre les signaux d’entrées et de sorties. Cette approche permet de modéliser les relations linéaires et la logique combinatoire afin de détecter des actions malveillantes en comparant l’ordre émis avec la sortie de la fonction de transfert. Cet IDS est notamment évalué dans le cadre des attaques MITM. Cependant, cette approche requiert de pouvoir décrire sous forme d’équation le système protégé et, de plus, elle évalue uniquement quantitativement l’ordre émis.

(Koutsandria et al. 2014) proposent dans leurs approches un IDS basé sur un ensemble de règles extraites d’une représentation d’un smart grid. L’outil de modélisation est un automate hybride et permet de caractériser le comportement des variables continues et discrètes à l’aide des trames échangées entre automates. A la vue de l’application, les variables continues décrivent l’évolution de l’intensité de courant tandis que celles discrètes reflètent l’état des disjoncteurs. L’IDS est testé sur une plateforme hardware in the loop avec un API et le protocole Modbus TCP. Plusieurs scénarii d’attaques sont mis en évidence pour détecter des incohérences dans l’envoi des messages ou des intervalles de temps trop importants.

(Gao et al. 2010) proposent, quant à eux, une approche par réseaux de neurones pour construire une approche de détection comportementale. L’ICS correspond à une cuve d’eau où l’on souhaite maintenir un certain niveau de remplissage. L’approche comprend deux modes de fonctionnement (automatique et manuel) et se base sur des caractéristiques du système physique (état de la pompe, niveau d’eau) et de la communication (fréquence d’envoi des trames). Encore une fois, cette approche se base sur la cyclicité très forte de ces systèmes pour construire le modèle par apprentissage. Toutefois, en s’appuyant à la fois sur l’analyse du réseau et du processus physique, et comme certaines approches précédemment citées, la tâche de l’attaquant pour atteindre son but est complexifiée.

Un dernier axe de développement est la sécurisation au niveau de l’automate en prenant en compte la connaissance de la commande ou du système physique. Nous pouvons citer les travaux de (McLaughlin et al. 2014) où le code automate est vérifié hors ligne, avant qu’il ne soit

téléchargé dans l'API, pour vérifier s'il respecte bien les contraintes de sécurité du système. Dans (Garcia et al. 2016), un hyperviseur est implanté dans l'API pour vérifier si les entrées reçues par ce dernier ne conduisent pas à un état de dégradation du système. Dans la même philosophie, (Mohan et al. 2013) proposent une architecture capable de monitorer en temps-réel si les règles de sûreté du système sont respectées. Toutefois, cette supervision est essentiellement temporelle sans prendre en compte les contraintes séquentielles ou combinatoires. Nous pouvons mettre en avant les travaux de (McLaughlin et Zonouz 2014) qui permettent de détecter des attaques par injection de données en se basant sur un modèle formel du code automate ; ou encore ceux de (Hailesellasi et Hasan 2017) qui se basent également sur une représentation du programme contenu dans le PLC pour générer des modèles graphiques permettant de comparer l'exécution courante du code avec un point de référence. Dans la lignée des travaux de (McLaughlin et al. 2014) qui proposent une sécurisation en pré-exécutant le code automate, les travaux de (McLaughlin 2013) proposent de sécuriser le système en analysant les ordres émis par l'API. Le respect d'une action par rapport à la politique du réseau est évalué en fonction du code automate et sur trois critères : sûreté, sécurité et performance. Toutefois, seules les attaques sur la chaîne d'action sont considérées. Plusieurs modes de replis sont proposés en cas de détection d'une attaque allant de la simple notification à l'opérateur jusqu'au blocage de l'ordre. Enfin, (McLaughlin 2015) propose une synthèse des deux approches vues précédemment en développant des *Guard Bloc* garantissant la bonne exécution du système. Cependant, cette approche, comme les précédentes, prend pour hypothèse forte que la loi de commande est correctement programmée dans l'API ce qui n'est pas toujours le cas dans le milieu industriel. De plus, la modification de la politique contenue dans les *Guard Bloc* se fait à chaque modification de la loi de commande ce qui pose un double problème de fréquence de modification (dans le cas d'une production flexible) et d'authentification pour effectuer le changement (garantir une modification légitime). Ces approches proposent des modèles déterministes aboutis de la partie commande et du système physique mais comme pour les HIDS, ces approches ne permettent pas de couvrir l'ensemble des attaques possibles pour les niveaux 1-0 de l'ICS.

La majorité des approches énoncées ci-dessus traite de la détection comportementale sur la base de modèles issus de l'apprentissage requérant des jeux de données conséquents, représentatifs du comportement normal du système et sains (sans corruption de la part d'un attaquant). Ces approches permettent de construire des outils simulant des attaques ou des techniques permettant de détecter des anomalies avec une certaine probabilité. Or, le placement dans les niveaux 1-0 de l'ICS présente l'avantage de diminuer la complexité du système en rendant possible l'écriture de modèles déterministes permettant de garantir la précision de la détection. En ce sens, ces approches ne sont pas retenues pour la suite de notre étude. En revanche, les travaux de (A. Carcano et al. 2011; I. Nai Fovino et al. 2012), qui se basent sur une modélisation déterministe du système à protéger en analysant notamment les états critiques, serviront de base pour l'approche développée. Les auteurs évoquent la notion de distance aux états critiques qui sera reprise et adaptée aux SED dans notre approche (partie 3). Dans (A. Carcano et al. 2011), cette notion permet de comparer l'état courant avec un ensemble d'états interdits dans le but d'indiquer la proximité d'un état interdit. Toutefois, cette approche utilise actuellement les données de l'automate afin de construire le vecteur d'états témoignant de l'évolution du processus

physique. Il s’agit ici d’une faille pouvant être utilisée par un attaquant pour contourner la détection. Dans la suite de ce chapitre, nous analyserons les approches utilisées en sûreté de fonctionnement pour obtenir des modèles déterministes en veillant à leur adaptation à des fins de détection dans un contexte cyber.

4. Étude des approches orientées sûreté de fonctionnement

Dans le paragraphe précédent, nous avons identifié des manques dans les approches orientées sécurité pour construire des modèles comportementaux. Ainsi, l’étude d’approches du domaine de la sûreté permet de compléter celles orientées sécurité. De plus, au début de ce chapitre, nous avons identifié une corrélation entre la problématique de détection d’une cyberattaque et celle relative à la détection de défaillances issues de la partie opérative. En effet, depuis des dizaines d’années, de nombreuses méthodes de détection et de diagnostic ont permis d’appréhender ce type de défaillances. Similairement aux différentes approches utilisées dans les IDSs, la littérature regroupe trois méthodes (Nguyen 2015) basées sur :

- La connaissance. Cette catégorie d’approches canalise la connaissance issue de l’observation du système pour mettre en relation, de manière causale, les symptômes et les défaillances. Nous pouvons citer les méthodes de type AMDEC, systèmes experts et arbres de défaillances,
- Le traitement des données. Ces approches utilisent les informations mises à disposition par le système pour modéliser le fonctionnement du système. Les méthodes d’apprentissage par machine learning et réseau de neurones et de reconnaissance de formes structurent cette catégorie,
- Les modèles. Ces approches s’appuient sur une représentation explicite du système. L’avantage de ces approches comportementales est de pouvoir comparer le comportement du système par rapport à une référence (M. Combacau et Courvoisier 1990). Nous distinguons notamment les approches quantitatives, qui mettent en relation les entrées et les sorties avec une équation, des approches qualitatives, qui permettent de formaliser de manière abstraite le fonctionnement du système.

Dans la suite de ce paragraphe, nous nous focaliserons sur les approches comportementales à base de modèles qualitatifs puisque nous souhaitons modéliser le comportement de la partie opérative tout en intégrant la connaissance « métier » inhérente aux ICSs. En effet, l’étude de la bibliographie pour les IDSs (paragraphe Chapitre 2 - 2 et Chapitre 2 - 3) nous a orientés vers une approche NIDS utilisant des modèles déterministes pour modéliser le comportement de l’ICS. L’intérêt principal de s’inspirer d’approches utilisées en sûreté de fonctionnement pour détecter des défaillances est la robustesse acquise au fil des développements.

Dans (Zamaï 1997), trois approches de surveillance qui s’inscrivent dans le cadre de travaux permettant d’obtenir une commande sûre ont attiré notre attention. Trois axes ont été dégagés dans les travaux de (Marangé 2009) qui découlent de ceux de (Pétin 2007) :

- Les approches par validation/vérification : ces approches ont pour objectif de garantir la bonne exécution de la commande en se basant sur les spécifications du cahier des charges. Il s'agit de méthodes de prévention qui interviennent avant la mise en ligne des systèmes et ne permettent pas de réagir. Quatre types de travaux sont regroupés dans ces approches : simulation, analyse d'atteignabilité, théorème de Proving et model-checking. Les difficultés majeures de ces approches demeurent l'explosion combinatoire et la définition correcte des spécifications. Ainsi, ces approches préventives ne sont pas retenues pour ces travaux de recherche.
- L'approche par synthèse : ces méthodes s'appuient sur les travaux de (Ramadge et Wonham 1987). Ces derniers permettent de spécifier la commande en se basant sur une modélisation du système contrôlé. Comme pour les précédentes approches, l'approche par synthèse souffre du phénomène d'explosion combinatoire lorsque le système est complexe et n'est pas retenue pour assurer la modélisation de l'ICS à protéger.
- Les approches par surveillance. Ces méthodes permettent d'assurer la sécurité de la commande tout en garantissant la qualité de la production. Ces approches permettent également de réagir face à une défaillance en se basant sur des modèles.

Les méthodes retenues permettent d'analyser les ordres émis par la partie commande à l'aide d'un modèle avant qu'ils ne soient appliqués à la partie opérative :

- Modèle de Référence (M. Combacau et Courvoisier 1990) - Figure 14 : dans cette approche, le modèle du processus physique est la référence pour la partie commande et définit le comportement normal de ce système physique. Avant chaque émission d'un ordre de la PC vers la PO, l'action est évaluée par le modèle de référence pour vérifier si toutes les conditions sont remplies pour l'appliquer sur le système physique. Une défaillance sur la partie opérative peut être détectée en comparant le compte rendu reçu de l'état du système dans le modèle de commande avec celui de référence. Cette approche séparant les modèles de commandes purs de ceux de la physique permet de détecter différents types de symptômes de défaillance pertinents pour envisager une reprise. Cette approche est pertinente dans la construction et l'utilisation des modèles, toutefois elle est centrée sur le modèle du procédé qui définit le comportement normal du système sans prendre en compte le cahier des charges spécifiant la commande.

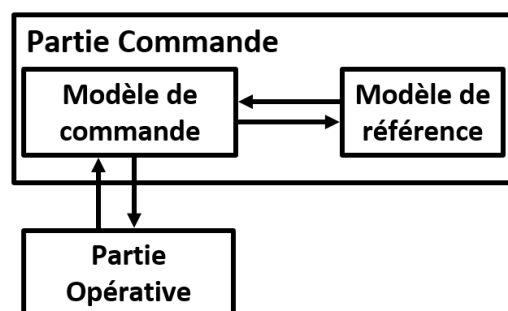


Figure 14 : Principe de fonctionnement de l'approche par modèle de référence (M. Combacau et Courvoisier 1990)

- Approche par Émulation (Holloway et Krogh 1991) - Figure 15 : dans ces travaux, l’analyse s’effectue sur la remontée d’informations de la part des capteurs. L’ordre de commande est envoyé à la fois à la PO et à l’émulateur, ce qui permet de comparer la sortie de l’émulateur (modèle du fonctionnement normal) avec les valeurs de capteurs (processus physique). Toute incohérence dans la comparaison mettra en avant une défaillance au niveau du procédé alors qu’elle peut provenir d’une simple désynchronisation entre le modèle et la commande. Cette approche ne permettant pas de surveiller la commande, elle ne sera pas retenue également pour notre approche.

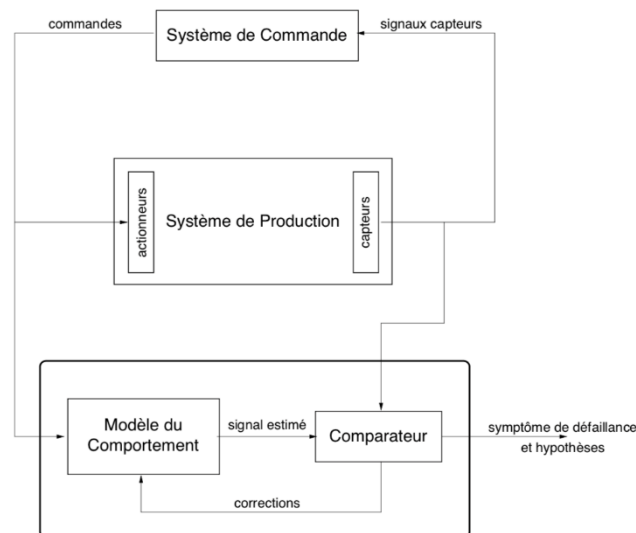


Figure 15 : Approche par Émulation (Holloway et Krogh 1991)

- Approche Filtre (Lhoste 1991), (Cruette, Bourey, et Gentina 1991) - Figure 16 : cette approche se base sur deux blocs de vérification placés entre la partie commande et la partie opérative comme illustré en Figure 16. Chacun de ces blocs analyse en permanence l’information qui lui est envoyée et valide son envoi ou la bloque. Le bloc vérifiant les ordres appliqués de la PC à la PO est appelé filtre de commande, il vérifie que l’action demandée ne dégradera pas le processus physique. A l’opposé, le bloc qui analyse les données remontées de la partie opérative à la partie commande est appelé filtre de comptes rendus et ce dernier vérifie si les CR sont cohérents avec l’ordre précédemment envoyé. Les deux filtres communiquent entre eux afin d’échanger respectivement le dernier ordre/CR validé et la liste des CR/ordres attendus. Au cours de ces différentes implémentations, cette approche a mis en œuvre différentes techniques de modélisation (règles, modèles déterministes, ...). Les modèles des parties commande et opérative peuvent être implémentés afin d’apporter une connaissance « métier » à la détection de défaillances et, dans le cas de notre étude, à la détection d’intrusions. Enfin, l’intégration de cette approche dans l’architecture ICS n’est pas sans conséquence puisque tout ordre ne respectant pas le modèle peut être bloqué. Nous noterons que ces filtres sont proposés comme intégrés aux API.

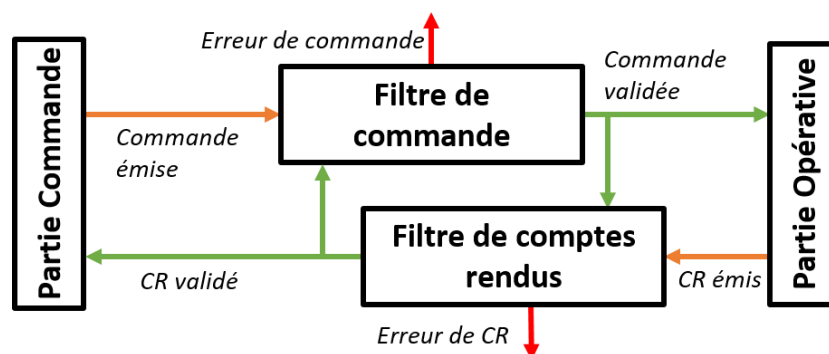


Figure 16 : Principe de fonctionnement de l'approche Filtre au sens de (Cruette, Bourey, et Gentina 1991) et (Lhoste 1991)

La lecture de (Sicard, Flaus, et Zamaï 2017) permet de détailler l'approche Filtre et ses différentes évolutions. Dans le paragraphe précédent, nous avons pu étudier différentes approches présentes dans la littérature pour détecter des défaillances. Les critères de décision ont guidé notre choix sur l'approche par modèle de référence et l'approche Filtre de Cruette et de Lhoste. Les deux approches sont similaires et détectent des désynchronisations avec la PO. Toutefois, l'approche par modèle de référence se focalise sur le modèle de la PO pour décider si un ordre est acceptable ou non. Or, dans le cadre de la cybersécurité, nous cherchons à détecter l'intentionnalité de la commande. Ainsi, l'approche Filtre est choisie puisqu'elle se concentre sur la commande en intégrant un modèle de cette dernière.

L'hybridation de cette approche avec les IDSs, localisée dans l'architecture à sécuriser, permet de sécuriser les ICSs face à la menace cyber sur les couches 1 – 0. Ainsi, comme dans l'approche originelle de Cruette et de Lhoste, les filtres sont considérés comme des blocs de vérification communiquant entre eux. Le filtre de commande analyse les ordres émis par l'API en se basant sur les informations communiquées par le filtre de CR et le modèle de la commande. Le filtre de comptes rendus corrèle les informations reçues des capteurs avec le modèle de la partie opérative et les informations fournies par le filtre de commande. Le fonctionnement de chaque filtre est détaillé ci-dessous et se base sur le fonctionnement global détaillé en Figure 16.

Le filtre de commande, dont le fonctionnement est représenté en Figure 17, analyse les actions émises par la partie commande à destination des actionneurs qui agissent sur le processus physique. L'objectif principal de ce filtre est de répondre au problème de désynchronisation de la partie opérative avec l'API. En effet, si désynchronisation il y a, un ordre peut cependant être émis et provoquer des comportements anormaux. En ce sens, ce filtre reçoit en entrée l'ordre, décidé par l'automate, qui doit être vérifié. La liste des ordres admissibles, c'est-à-dire les ordres attendus en fonction de l'état du système physique, et le dernier compte rendu validé sont envoyés par le filtre de comptes rendus. Sur la base de ces informations, le filtre de commande exécutera dans notre approche des algorithmes de détection (explicités dans la partie 3). Ces derniers vérifieront notamment le respect de contraintes issues des modèles générés et garantiront le bon fonctionnement du système. Fort de ces algorithmes de détection, l'ordre peut être :

- Validé par le filtre de commande. L'action est alors transmise aux actionneurs et au filtre de comptes rendus ainsi que la liste des états admissibles,

- Validé avec une alerte par le filtre de commande. Comme précédemment l’ordre est transmis à la PO et au filtre de CR ; toutefois, une alerte est émise à destination des opérateurs pour les informer d’une anomalie (défaillance ou attaque), c’est-à-dire un éloignement entre la trajectoire prévue et celle prise,
- Bloqué. Le filtre de commande stoppe l’émission de l’ordre vers les actionneurs pour qu’il ne soit pas appliqué sur le système physique. Une alerte est également envoyée aux opérateurs de supervision pour agir sur l’ICS. Suivant le mode d’implantation des filtres, un mode de repli peut être engagé et les filtres de commande et de comptes rendus pilotent le processus physique pour l’amener dans un état défini comme sûr.

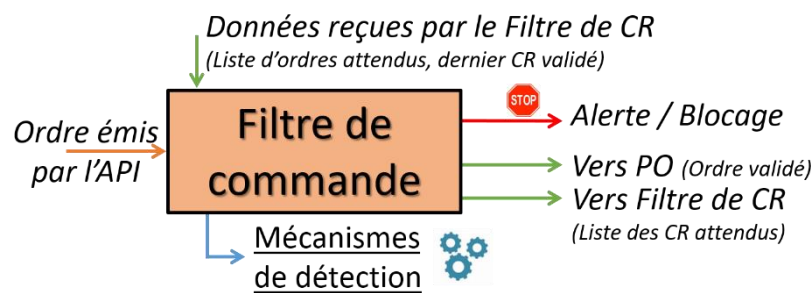


Figure 17 : Principe de fonctionnement du filtre de commande

Le filtre de comptes rendus, représenté en Figure 18, a pour rôle d’intercepter la remontée d’informations des capteurs vers la partie commande afin de vérifier d’une part la bonne exécution de l’action et, d’autre part, la cohérence entre l’état courant du système remonté par les capteurs et la trajectoire du système. Pour ce faire, le filtre de CR prend en entrée le compte rendu émis par les capteurs ainsi que les informations fournies par le filtre de commande, à savoir le dernier ordre validé et la liste des comptes rendus admissibles. Sur la base de ces informations, le filtre de CR effectue des calculs pour les mécanismes de détection. A la suite de ces calculs, le compte rendu est transmis à la partie commande ainsi qu’au filtre de commande avec la liste des ordres admissibles. Le filtre de CR ne dispose pas de la fonctionnalité de blocage des informations. En effet, dans le cas où l’attaquant corromprait les données des capteurs pour induire une action malveillante de l’API, le filtre de commande se chargerait du blocage. Le filtre de comptes rendus permet de vérifier la bonne exécution des actions et la cohérence des données des capteurs.

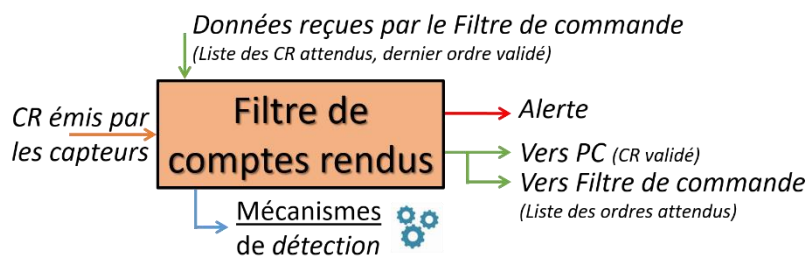


Figure 18 : Principe de fonctionnement du filtre de comptes rendus

Ainsi, chaque filtre est le *Trust Anchor* de l’autre. En considérant que la communication entre les deux filtres est sécurisée, le principe de fonctionnement décrit ci-dessus permet de protéger la partie opérative d’ordres incompatibles avec l’état de cette dernière. L’insertion de

modèles comportementaux représentatifs des parties commande et opérative et la localisation des filtres sont deux éléments indispensables et non dissociables pour rendre possible la détection d'attaque. Dans la suite de ce chapitre, nous discuterons des notions de sûreté et de sécurité.

5. Bilan des approches Sûreté/Sécurité

Les notions de sécurité et de sûreté sont à la fois très distinctes et relativement communes (Fourastier et Pietre-Cambacedes 2015). La sûreté de fonctionnement est l'étude du risque fortuit (défaillance, faute, accident) pouvant affecter un système, son environnement et la sécurité des biens et des personnes (Flaus 2013). La sécurité, quant à elle, se concentre sur la protection des systèmes contre des actes malicieux. La cybersécurité est l'équivalent avec l'utilisation du vecteur numérique. Ces deux concepts ont tout d'abord la même fonction sur les systèmes de contrôle-commande : assurer la non-perte du service devant être fourni. Cependant, la nature de l'événement est différente : volontaire (sécurité) ou fortuite (sûreté). (Fourastier et Pietre-Cambacedes 2015) ont défini quatre types de relations entre ces deux concepts dans le cadre de la cybersécurité des ICSs :

- Dépendance conditionnelle, le respect des conditions de sûreté détermine le niveau de sécurité et vice versa,
- Renforcement mutuel, les mesures de sécurité améliorent la sûreté du système et inversement,
- Antagoniste : les besoins en sûreté nuisent à la sécurité du système et vice versa,
- Indépendance : aucune interaction entre ces concepts.

De nombreuses recherches tendent à un *renforcement mutuel* des approches pour sécuriser les systèmes de contrôle-commande industriels (Piètre-Cambacédès et Bouissou 2013). Ainsi, les travaux de (Siwar Kriaa et al. 2015; Ellis 2015) ont identifié les points de convergence entre les analyses réalisées dans le cadre de la sûreté et de la sécurité. Des méthodologies pour combiner ces deux concepts ont été proposées dans ces travaux. De nombreux travaux proposent également de concevoir des scénarii d'attaques ou de défaillances dans le but d'évaluer les approches comme dans (S. Kriaa, Bouissou, et Laarouchi 2015) où une approche basée sur des modèles du système physique est utilisée pour générer des scénarii. La publication de (Abdo et al. 2018) va dans le même sens en utilisant les arbres d'attaques avec l'analyse bowtie. (Cárdenas et al. 2011) ont développé une approche pour détecter des attaques informatiques affectant le comportement de l'ICS en s'appuyant sur le fonctionnement du système physique. Cette connaissance est modélisée avec une méthode quantitative à base d'équations. Enfin, dans (Friedberg et al. 2017), les auteurs ont développé une nouvelle méthodologie d'analyse pour prendre en compte à la fois les aspects sûreté et sécurité dans des systèmes cyber-physiques en se basant sur des indicateurs de vulnérabilité et de risques de défaillance. Par la suite, la meilleure stratégie de mitigation est identifiée pour le système.

Bien que la littérature prenne en considération de plus en plus ce problème, peu de travaux ont testé les approches énoncées précédemment sur des systèmes industriels. Toutefois, nous nous appuierons sur ces analyses et résultats pour la suite de notre approche. En effet, la prévention

et l’identification sont les conditions *sine qua non* pour développer des algorithmes de détection efficaces. Toutefois, nous pourrions remarquer que dans le cas de la sûreté de fonctionnement, l’ordre n’est jamais remis en cause : l’actionneur, le produit ou le capteur est alors en défaut. La question est *quel élément est à l’origine de la non-réalisation du service relatif à l’ordre envoyé ?* Cependant, dans le contexte de cybersécurité, cette analyse s’inverse pour que la question devienne : ***l’ordre envoyé est-il malveillant ?***

Dans notre approche, nous mettons à profit le concept de renforcement mutuel énoncé dans (Fourastier et Pietre-Cambacedes 2015). Pour ce faire, nous avons sélectionné une liste de critères :

- Couverture face aux attaques affectant les niveaux 1-0 des ICSs.
- Intégrabilité dans l’architecture actuelle des systèmes industriels.
- Possibilité de construction de modèles déterministes pour analyser les informations à l’aide de modèles comportementaux.

Les approches IDS orientées réseaux permettent une localisation des sondes sur toute l’architecture ICS afin d’étudier le réseau. Ce point permet de couvrir l’ensemble des surfaces d’attaques entre les niveaux 1-0. De plus, l’étude bibliographique a montré que les NIDS sont facilement intégrables dans une architecture connectée en réseau puisqu’ils ne font que capter l’information sans perturber le système. Les NIDS orientés connaissance du procédé permettent d’axer la détection d’intrusions sur des modèles comportementaux obtenus dans la majorité des cas par apprentissage et dans l’approche de (A. Carcano et al. 2011) par un modèle déterministe. L’utilisation de ce dernier permet de garantir l’évolution du système et donc une détection. Toutefois, la construction des modèles déterministes n’est pas aboutie dans la littérature des approches orientées sécurité. En cela, l’approche Filtre de (Cruette, Bourey, et Gentina 1991) et (Lhoste 1991) permet d’apporter des solutions de modélisation pour le troisième critère de sélection. En revanche, dans l’approche originelle, les filtres sont situés à l’intérieur de l’API. En se projetant dans le contexte de la cybersécurité, la localisation des filtres devra se faire à l’extérieur des automates qui sont considérés comme des éléments vulnérables (Chapitre 1 - 5). Cette réflexion dans le placement des filtres est compensée par les approches NIDS. Ainsi, les faiblesses de chacune des approches sont compensées par les forces de l’autre.

6. Bilan du chapitre

Dans ce chapitre, un panorama de la cybersécurité des systèmes de contrôle-commande industriels a été présenté. Il a permis d’explorer les différentes techniques permettant de sécuriser les systèmes d’information. Nous avons ensuite ciblé l’étude bibliographique sur les IDSs dont l’adaptation aux ICSs est très présente dans la littérature. Plusieurs catégories d’IDS ont été présentées en détail ainsi que leur application aux systèmes de contrôle-commande industriels. Enfin, nous avons considéré la littérature traitant de l’utilisation de la connaissance des ICSs pour créer des mécanismes de détection d’intrusions. Les travaux de (A. Carcano et al. 2011) et de (I. Nai Fovino et al. 2012) ont permis d’établir une notion intéressante pour évaluer le

rapprochement d'un état critique : la distance. Cette dernière sera détaillée dans le chapitre 3 et sera à la base des mécanismes de détection développés.

Par la suite, nous avons mis en évidence le lien entre la problématique de détection dans la détection d'intrusions et les approches par surveillance. Après avoir replacé ces approches dans les méthodes développées pour la sûreté de fonctionnement, nous avons détaillé plusieurs travaux sur ce sujet. L'approche filtre proposée par (Cruette, Bourey, et Gentina 1991; Lhoste 1991) a retenu notre attention pour être hybridée avec un IDS basé sur des modèles comportementaux.

Ce chapitre s'est conclu sur une discussion autour de la notion de sûreté et de sécurité. De nombreux travaux tentant d'introduire les deux notions dans des analyses conjointes ont été présentés. Toutefois ces analyses restent préventives mais sont nécessaires pour appuyer les approches réactives. En hybridant une approche de sûreté et une de sécurité, notre approche s'appuiera sur les bénéfices des deux pour développer de nouveaux mécanismes de détection orientés sur la connaissance de l'ICS.

Chapitre 3

Problématique, positionnement et principe de l’approche proposée

Dans ce chapitre, nous allons définir la problématique de nos travaux ainsi que le positionnement de l’approche qui sera développée. Le premier chapitre a permis d’appréhender la menace cyber inhérente aux systèmes de contrôle-commande ainsi que la nécessité de les protéger ; cette protection étant obligatoire pour les opérateurs d’importance vitale. Dans le chapitre qui a suivi, les approches existantes en sécurité et en sûreté de fonctionnement ont été énoncées. Toutefois, ces techniques ne garantissent pas une protection exhaustive contre les risques d’intrusion dans un ICS. Dans ce chapitre, nous allons spécifier les particularités de ces systèmes qui rendent les solutions existantes seulement partiellement satisfaisantes. Ensuite, nous positionnerons notre approche, le type d’attaques considéré et le principe général de détection de l’approche proposée.

Table des matières

1.	Spécificités des ICSs par rapport au SI « traditionnel ».....	66
2.	Positionnement de l’approche proposée.....	68
3.	Types d’attaques considérés	69
3.1.	Filtrage assurant la sécurité des biens et des personnes.....	70
3.2.	Filtrage assurant la qualité du produit	71
3.3.	Filtrage assurant la protection des équipements	71
4.	Approche S.A.F.E.: Security Approach based on Filter Execution	71
4.1.	Localisation de l’approche.....	72
4.2.	Principe de détection.....	73
5.	Hypothèses de travail et verrous scientifiques	75
6.	Bilan du chapitre.....	77

1. Spécificités des ICSs par rapport au SI « traditionnel »

Dans le chapitre 2, nous avons noté que de nombreuses solutions existent pour prendre en considération la cybersécurité des systèmes d'information tant au niveau software qu'au niveau hardware (firewall, sondes IDS, diode, antivirus, clé de chiffrement...). Cependant, ces solutions n'apportent pas de solutions satisfaisantes lorsqu'on les applique sur des systèmes de contrôle-commande industriels. Cela vient du fait que ces solutions ne prennent pas en compte les spécificités des ICSs. L'objectif de ces systèmes, l'environnement dans lequel ils évoluent, le matériel utilisé et les utilisateurs sont autant de paramètres qui varient entre un système d'information « classique » et un système industriel (ANSSI 2012; Cárdenas, Amin, et Sastry 2008).

Une des premières distinctions est évoquée dans (Ani, He, et Tiwari 2017), les systèmes d'information appartiennent au monde de l'IT traditionnelle et manipulent des données. Les ICSs font partie de la catégorie OT (Operational Technology) où **l'objectif** est de piloter des infrastructures, d'acquérir et de traiter des données (ANSSI 2012). La **priorité** dans l'IT est la confidentialité des données, puis l'intégrité et enfin la disponibilité de ces dernières (modèle CIA). En revanche dans le cas de l'OT, l'ordre des priorités est complètement inversé, à savoir disponibilité, intégrité et confidentialité des données (Cheminod, Durante, et Valenzano 2013; Kabir-Querrec 2017). En se basant sur (Sicard, Zamaï, et Flaus 2018a; Weiss 2016; Stouffer et al. 2015), plusieurs autres points peuvent être soulevés :

- **Contraintes temps-réels.** Comme énoncé précédemment, la disponibilité des données dans un SI n'est pas la première priorité. Cela provient du fait que ces systèmes doivent assurer la confidentialité des données en s'adaptant aux contraintes métiers de l'application dont ils assurent le fonctionnement. A l'inverse, les ICSs doivent s'adapter aux contraintes métiers tout en assurant les fonctions de sécurité et un fonctionnement 24h/7j. De plus, comme évoqué dans le chapitre 1 - 4 Environnement d'un Industrial Control System (ICS), les niveaux 1 et 0 sont soumis à des contraintes temps-réels fortes (de l'ordre de la milliseconde) et le niveau 2 est contraint par des obligations moins fortes (de l'ordre de la seconde).
- **Protocoles de communication.** Pour les systèmes d'information, les protocoles utilisés sont très standardisés (TCP/IP, UDP) et universels (Cheminod, Durante, et Valenzano 2013). En revanche, les ICSs reposent beaucoup sur des protocoles propriétaires (W. Yang et Zhao 2014) comme décrit dans le chapitre 1 – 3.2.4 Communication entre les niveaux.
- **Hétérogénéité des technologies.** Dans la lignée de ce qui a été évoqué au point précédent, la compatibilité des composants entre eux est normalisée dans l'IT (homogénéité et interopérabilité). En revanche, les technologies utilisées dans les ICSs progressent en fonction de l'évolution du système industriel et de sa durée de vie (cf. chapitre 1 - 4 Environnement d'un Industrial Control System (ICS)). Il en résulte une superposition de technologies (communication, matériel, logiciel) intergénérationnelles souvent complexes à manager.

- **Environnement.** Dans les SI, les différents éléments sont protégés et opèrent dans un environnement sain (salle climatisée, bureau, domicile). En revanche, les différents éléments qui composent un ICS (automate, IHM, capteurs, actionneurs, ...) sont soumis à un environnement dur : température, pression, poussières. De plus, et contrairement au SI, il y a une nécessité dans les ICSs de pouvoir se connecter aux API (via le réseau en Ethernet ou en physique via une liaison série) pour la mise en service et la maintenance d’une installation ou le contrôle de la qualité.
- **Intervenants.** Si dans les systèmes d’information, seul les informaticiens interviennent, dans les ICSs (cf. chapitre 1 – 4 Environnement d’un Industrial Control System (ICS)), de multiples acteurs prennent part au bon fonctionnement (automaticien, technicien, maintenance, instrumentiste, spécialiste du procédé contrôlé (chimiste, physicien, mécanicien, ...)).
- **Contraintes matérielles.** Dans les SI, le matériel est conçu et optimisé pour traiter des données et faire face au risque cyber. En revanche, dans les ICSs, la priorité est donnée à la robustesse par rapport aux conditions opératoires et la prise en compte des contraintes temps-réels. Un API doit être facilement configurable, adaptable et flexible. Ces différences se traduisent notamment par des limitations du nombre de calculs (mémoire) dans les API par rapport à des machines appartenant aux SI.
- **Maintenance.** Dans l’IT, la sécurité est intégrée dès la conception et tout au long de la durée de vie du produit. Ainsi, des patches et des mises à jour sont régulièrement publiés pour corriger des problèmes logiciels, firmware, d’architecture. En revanche, cette culture est différente pour les ICSs. Bien que depuis les années 2010, les éditeurs publient de plus en plus de mises à jour, ces dernières ne sont pas forcément appliquées sur les systèmes puisque cela implique des arrêts de production, des temps de redémarrage parfois importants et surtout, une non-garantie de retrouver le même fonctionnement que celui précédant l’arrêt. De plus, comme les ICSs évoluent dans des environnements complexes, les maintenances sont planifiées longtemps en avance et réduites au strict minimum pour assurer les missions de production et de fiabilité respectant le mythe « *Le système tombe en marche* » (§1.1.2 - Quelques mythes concernant les systèmes d’information industriels (ANSSI 2012)).
- **Durée de vie.** La durée de vie des éléments d’un SI est en moyenne de 5 ans et dépasse rarement les 10 ans pour les plus vieilles infrastructures. En ce qui concerne les ICSs, les équipements ont une durée de vie de 15 ans en moyenne et cette dernière peut monter sur certains sites industriels à 20 ou 30 ans.
- **Contraintes de fonctionnement.** Les ICSs doivent assurer une production en continu (24h/24, 7j/7) tout en garantissant productivité et fiabilité (Villemeur 1992). Dans les SI, le fonctionnement peut être interrompu notamment pour la maintenance de l’infrastructure.

Ces spécificités, regroupées dans le Tableau 3, permettent de comprendre en quoi les approches actuelles n’apportent pas de solutions satisfaisantes. L’adaptation de techniques utilisées dans l’IT, bien qu’elles soient efficaces et certifiées, ne sécurise pas les ICSs. Dans la suite de ce chapitre, nous détaillerons en quoi la prise en compte de la connaissance « métier »

inhérente aux systèmes de contrôle-commande industriels est une composante essentielle pour protéger ces systèmes. Cette originalité sera au cœur de l'approche proposée dans la suite de ces travaux.

Tableau 3 : Récapitulatif des spécificités de l'OT par rapport à l'IT

	Système d'information (IT)	Industrial Control Systems (OT)
Objectif	Traiter des données	Piloter un processus physique, acquérir des données et interagir avec des opérateurs
Contraintes	Confidentialité, Intégrité, Disponibilité	Disponibilité (système temps-réel) et sûreté de fonctionnement
Environnement	Doux (Bureau, data center)	Dur (ligne de production, poussières, vibrations, température, ...)
Communication	Protocoles standards	Plusieurs protocoles propriétaires
Maintenance	Mises à jour régulières	Mises à jour (logiciel, firmware, patch de sécurité) n'existent pas ou ne sont pas déployées (arrêt de la production, redémarrage, ...)
Durée de vie	3-5 ans	15-20 ans

2. Positionnement de l'approche proposée

Comme discuté au début de ce chapitre, les solutions existantes en termes de cybersécurité des systèmes d'information n'apportent pas de réponses entièrement satisfaisantes lorsqu'elles sont utilisées telles quelles dans les ICSs. En effet, elles permettent de sécuriser les plus hauts niveaux de l'architecture CIM (du niveau 5 au niveau 2) qui se rapprochent le plus de la typologie des SI classiques. Cependant, lorsqu'il s'agit du réseau industriel (niveaux 1 et 0), nous avons vu que les spécificités des ICSs diminuent leur pertinence et leur efficacité (Chapitre 3 - 1. Spécificités des ICSs par rapport au SI « traditionnel »). De ce fait, en considérant l'intégration sur un système industriel de l'ensemble des approches et préconisations énoncées précédemment, tout en garantissant une bonne hygiène « informatique » sur les postes abritant les logiciels de configuration ainsi qu'en terme d'accès physique aux installations, de maintenance (télémaintenance) et de chiffrement, pouvons-nous supposer que l'ensemble des surfaces d'attaques ont été comblées ? Tout d'abord, l'hypothèse de départ est erronée de par le coût de l'installation de ces solutions. D'autre part, la complexité des ICSs rendrait difficile la mise en place et l'efficacité de toutes ces solutions et préconisations. En effet, les protections « traditionnelles » ne sécuriseraient pas le système si l'automate était directement ciblé, à l'occasion d'une maintenance par exemple réalisée par une tierce personne n'appartenant pas au site industriel (un intégrateur par exemple).

Il apparaît indispensable de développer une approche complémentaire se concentrant sur la protection des couches « basses » de l’architecture CIM, aussi appelées temps-réels. Ces couches sont en lien direct avec le système physique et en cas d’attaque, il s’agit de la dernière chance de stopper une action malveillante. L’approche développée dans cette étude se situe entre la partie commande (niveau 1) et la partie opérative (niveau 0). Ce choix est motivé par l’étude de vulnérabilité menée dans le chapitre 1 - 5. Étude des vulnérabilités d’un ICS, qui conclut que l’architecture ICS en entier est vulnérable. L’intérêt majeur de ce positionnement est de pouvoir utiliser la connaissance « métier » issue de ces systèmes. Cette dernière comprend le fonctionnement d’un ICS dans les couches temps-réels, les spécificités de ces systèmes et la complexité de l’environnement dans lequel ils interviennent. Cette connaissance « métier » sera modélisée par le biais de modèles comportementaux prenant en compte le fonctionnement de la partie commande et de la partie opérative. La localisation de cette approche, au plus proche de la partie opérative, permet de bénéficier d’une protection adaptée aux systèmes de contrôle-commande et surtout de minimiser la complexité. En effet, quant au niveau 2, la supervision gère une quantité importante (milliers) de variables, la partie commande (niveau 1), elle, n’en contrôle qu’au maximum une centaine, et plus classiquement une trentaine. Ce placement est primordial pour garantir la convergence des algorithmes et mécanismes de détection qui seront proposés.

Toutefois, cette approche requiert une méthodologie pour modéliser la connaissance « métier », et donc générer des modèles des parties commande et opérative, piliers du travail développé par la suite. Par ailleurs, des limites inhérentes au placement de l’approche apparaissent :

- Le temps d’exécution des mécanismes sera lié aux couches considérées et donc proche du temps réel. Cette limite sera nuancée dans la suite de nos travaux notamment par les mécanismes détectant des attaques de séquence,
- Le manque de vision globale lié à la localisation de l’approche. Comme expliqué précédemment, une cellule d’automate (niveau 1) a accès à moins de variables qu’une supervision globale (niveau 2) puisqu’elle ne gère qu’un sous-ensemble du système physique.

Cette approche est bien évidemment complémentaire aux solutions IT tant au niveau de la localisation que des mécanismes de détection mis en œuvre (analyse réseau / connaissance « métier ») (Sicard, Zamaï, et Flaus 2018d).

3. Types d’attaques considérés

Notre approche se localise entre la partie commande et la partie opérative ; nous considérons les attaques pouvant être exécutées ou pouvant avoir des répercussions à ces niveaux-là. Nous proposerons des mécanismes de détection pour les attaques détaillées dans le chapitre 1 - 1.4. Définition d’une attaque comme les MITM, Replay, DDOS, FDI ou encore les attaques aléatoires. Notre approche surveillera les attaques ayant pour cible les éléments suivants :

- *Control channel*, canal liant des sorties de l'API jusqu'aux pré-actionneurs et qu'un hacker peut corrompre pour commettre des actions malveillantes sur le système physique,
- *Report channel* qui remonte les données du réseau de capteurs jusqu'à la partie commande. Un attaquant peut corrompre les informations envoyées afin d'induire des ordres malveillants sur l'ICS,
- API ou supervision, où en attaquant sa configuration, sa mémoire ou son intégrité logicielle (firmware, logiciel de programmation), le hacker cherche à endommager la partie opérative en induisant une action malveillante ou en inhibant un ordre protégeant le système.

Dans le cadre de ces travaux, quatre natures d'attaques seront considérées, issues de l'analyse de la littérature effectuée au Chapitre 2 :

- Directes : amènent rapidement le système dans un état critique afin de l'endommager.
- Séquentielles : dégradent le comportement de l'ICS et s'effectuent sur une séquence ordre/CR qui rapproche peu à peu le système d'un état critique.
- Temporelles : attaques visant à transmettre des ordres/comptes rendus en violant la périodicité.
- Sur-sollicitatrices : ciblent les équipements de production afin de les rendre inutilisables pour la production en les usant prématurément, en provoquant une maintenance ou encore en cassant le matériel.

Les attaques considérées viennent dégrader la sécurité des biens et des personnes, la qualité du produit et la protection des équipements. Afin de faire face à ces menaces, différents modes de protection doivent être définis.

3.1. Filtrage assurant la sécurité des biens et des personnes

Dans cet objectif de protection, les filtres doivent garantir un ensemble d'états dans lesquels le système peut évoluer sans danger. Ainsi, les filtres de commande et de comptes rendus analysent les informations échangées entre les parties commande et opérationnelle dans l'objectif de vérifier que le système n'atteigne pas un état interdit. Dans ce niveau de sécurité, seul le modèle de la partie opérative (niveau 0) est nécessaire et suffisant. Les séquences d'ordres ne sont pas analysées puisque le modèle de commande n'est pas pris en compte ce qui implique que la bonne exécution de la loi de commande n'est pas vérifiée. Dans notre approche, nous viserons à identifier des éloignements par rapport au modèle de la loi de commande. Ainsi, le fait de ne pas avoir accès à cette dernière est pénalisant et ne permet pas d'utiliser la totalité des mécanismes de détection développés. Toutefois, ce niveau de protection a besoin d'être modifié uniquement en cas de changement de la partie opérative.

Ainsi, dans le cas où on s'intéresse à la protection des systèmes avec une forte variabilité de la commande (changement de recette fréquent) alors l'approche se basant uniquement sur le modèle de la partie opérative peut être pertinente. En contrepartie, les indicateurs développés durant ces travaux de thèse ne sont pas utilisés dans cet objectif de filtrage puisque les spécifications de la commande ne sont pas connues des filtres.

3.2. Filtrage assurant la qualité du produit

Cet objectif de protection garantit la bonne exécution de la loi de commande au-delà de la surveillance de l’évolution des états du système. Ainsi, la trajectoire pour évoluer de l’état initial à l’état final est connue par les filtres et comparée avec l’évolution courante. Les modèles de commande et du système physique doivent être connus pour garantir ce niveau de protection, autrement dit la connaissance des niveaux 1 et 0 est nécessaire et suffisante. Ainsi, les filtres s’assurent que le système respecte les contraintes imposées par l’analyse de risques (évolutions temporelles et combinatoires) et la partie commande (séquence d’ordres). Le concept de trajectoire peut être utilisé pour détecter des anomalies dans la séquence d’ordres envoyée aux systèmes ou dans la séquence d’états obtenue pouvant altérer la qualité du produit final. Cependant, pour ce niveau de protection, le modèle de loi de commande, qui est l’élément primordial pour avoir accès aux différents types d’états et au calcul de la distance, doit être changé à chaque modification de la loi de commande.

Ainsi, ce mécanisme de protection est efficace pour les systèmes industriels à faibles variabilités. Bien que sa mise en place soit lourde (méthodologie de conception des filtres) et son fonctionnement strict (une seule trajectoire optimale est définie), elle permet d’utiliser les indicateurs de distance et de trajectoire développés dans ces travaux de thèse.

3.3. Filtrage assurant la protection des équipements

Ce niveau de protection surveille la sollicitation des actionneurs vis-à-vis des ordres émis par l’API afin d’éviter une sur-sollicitation de ces derniers. Ces sur-sollicitations peuvent être en fréquence ou en amplitude. L’objectif est de protéger les équipements de la fatigue, la maintenance ou les pannes. Ainsi, pour chaque actionneur, les filtres surveillent l’évolution des ordres envoyés. Cet objectif de protection permet de prévenir des attaques par séquence ciblant un équipement en particulier pour le mettre hors de fonction pour arrêter la production de l’ICS, dégrader la qualité du produit final ou encore mettre en danger la sécurité des biens et des personnes.

En conclusion, les attaques considérées dans ce travail de recherche sont celles qui *in fine* ont l’objectif à plus ou moins long terme de dégrader le flux de production, les équipements ou la qualité du flux de production. Les protections des niveaux supérieurs sont nécessairement primordiales pour diminuer le nombre et le type d’attaques à traiter. Ces travaux s’inscrivent dans le concept de défense en profondeur (Fourestier et Pietre-Cambacedes 2015).

4. Approche S.A.F.E.: Security Approach based on Filter Execution

L’approche S.A.F.E. proposée dans ces travaux de thèse se focalise sur la détection d’intrusions dans les niveaux 1 – 0 de l’ICS. Elle s’intègre à l’architecture existante des systèmes de contrôle-commande afin de vérifier l’intégrité des échanges entre la partie commande et la partie opérative. La détection s’appuiera non pas sur l’analyse classique des trames réseaux mais

sur la connaissance « métier » de ces systèmes. Ainsi, une approche comportementale basée sur une modélisation à la fois de la partie commande et de la partie opérative sera développée. Ces modèles analysent d'une part, si les ordres envoyés à partir de l'API aux actionneurs sont conformes aux contraintes de sécurité et, d'autre part, si les comptes rendus transmis des capteurs à la partie commande sont cohérents avec les actions décidées.

4.1. Localisation de l'approche

Les filtres, synthétisés à partir de l'approche S.A.F.E., se basent sur une hybridation des techniques d'IDS et de l'approche filtre (cf. Chapitre 3). Nous avons pu notamment détailler les points positifs et négatifs de chacune de ces approches et discuter du potentiel pour les adapter à la problématique « cyber ». A la vue de la localisation des vulnérabilités dans un ICS, les filtres issus de l'approche développée analyseront les échanges entre les parties commande et compte rendu. En pratique, ces derniers seront situés en dehors de l'API ou de tout autre composant relié au réseau industriel. De plus, ils devront être localisés au plus proche de la partie opérative comme illustré sur la Figure 19.

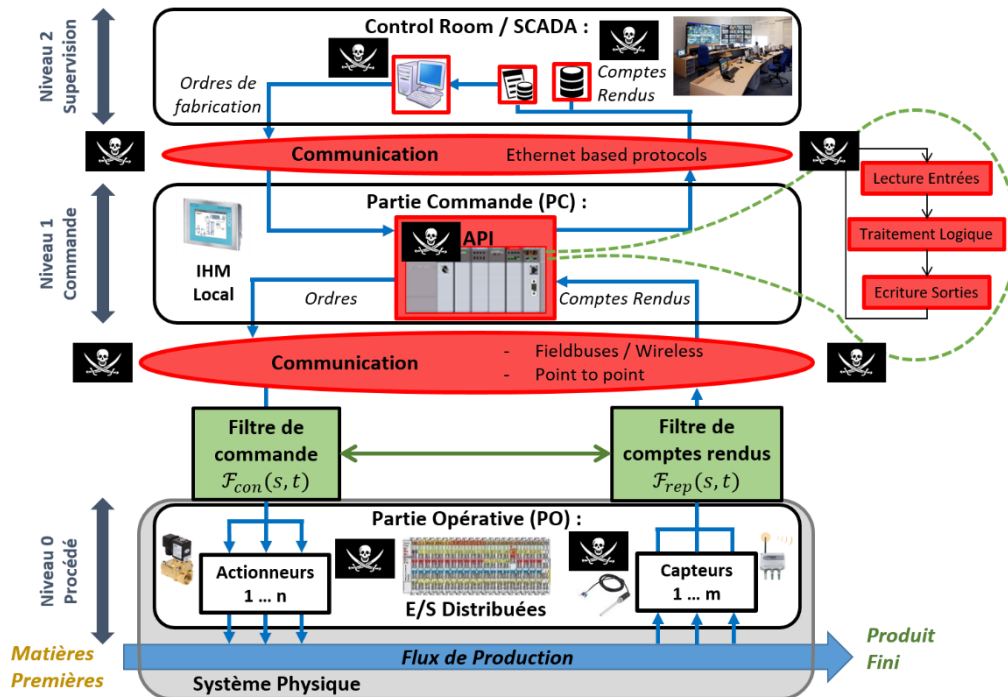


Figure 19 : Localisation des filtres de l'approche S.A.F.E. dans l'architecture ICS

La connexion entre les filtres et les organes de commande et de remontées d'informations se fait soit en liaison directe (point à point, analogique) soit par un réseau de terrain (capteurs / actionneurs). L'objectif est d'isoler la partie opérative du reste de l'ICS qui peut être corrompu. Les filtres agissent tel un disjoncteur électrique qui, lorsqu'un défaut est présent sur un réseau électrique, s'ouvre pour protéger l'installation. De plus, la localisation des filtres au plus proche de la PO permet ainsi d'analyser les flux de données dans les deux sens pour garantir les données analysées. En effet, comme l'ICS dans son ensemble est considéré comme vulnérable, les ordres émis par l'API et les données remontées à ce dernier peuvent être corrompus. Les filtres de

commande et de compte rendu permettent d’agir comme un *Trust Anchor* garant du bon fonctionnement du système sécurisé (Sicard, Zamaï, et Flaus 2018a).

Le filtre de commande a pour objectif d’analyser les ordres émis par l’API avant qu’ils ne soient appliqués par les actionneurs au système physique. Ce filtre se positionne au plus près des organes de commande, c’est-à-dire :

- A l’entrée des actionneurs dans le cas d’une liaison directe (point-à-point, analogique, protocole terrain),
- Après le dernier module de communication dans le cas d’une communication sans-fil,
- Après le dernier module d’entrée/sortie distribué dans les autres cas.

Ce placement dans l’architecture ICS permet de garantir l’intégrité de l’action envoyée à la partie opérative même si cette dernière est d’origine malveillante par le biais d’un composant corrompu. De la même manière, le filtre de comptes rendus doit être placé au plus proche des capteurs, à savoir :

- Dans le cas de capteurs non-connectés, si la communication est :
 - Directe (point-à-point ou analogique), le filtre de CR est placé en entrée de l’automate,
 - Via un réseau (protocole terrain ou sans-fil), le filtre de CR doit se situer avant le module de communication.
- Dans le cas de capteurs « connectés » (smart sensors), des fonctions de calcul de communication sont embarquées dans ces systèmes de mesures. De plus, l’attaquant peut directement compromettre le capteur, contrairement à des capteurs « classiques » qui ne transmettent qu’une mesure envoyée à la PC sous forme de signal électrique ou du réseau terrain. Le filtre de CR devrait être placé dans chaque capteur ou à défaut au plus tôt dans la chaîne permettant de remonter l’information.

Ce placement ne permet pas de protéger la partie commande ou même la communication sur le réseau industriel entre les niveaux 1 et 0 mais permet de protéger la partie opérative.

4.2. Principe de détection

Cette approche se base sur des filtres afin de surveiller les échanges entre la PO et la PC et les éventuelles dérives naturelles ou intentionnelles. Cette construction sera formalisée par le biais d’une méthodologie de conception représentée en Figure 20. De plus, elle sera orientée en fonction du filtrage souhaité pour la partie de l’ICS sécurisée : sécurité des biens et des personnes, qualité du produit et sécurité des équipements de production. Ces notions seront développées dans la partie 2 de ce manuscrit. Ainsi, les objectifs de détection de ces filtres seront :

- Le blocage des actions conduisant immédiatement à des états critiques pour le système, c’est-à-dire des états caractérisant des dommages, légers ou sévères, sur l’ICS,
- L’anticipation des déviations en se basant sur les modèles des couches basses des ICSs tant au niveau combinatoire que temporel. La philosophie sous-jacente est là encore

la connaissance « métier » afin de détecter des attaques plus complexes, agissant sur des séquences plus longues.

- L'aide au pré-diagnostic des anomalies constatées sur le système afin d'améliorer la vigilance et la prévention des opérateurs vis-à-vis du risque cyber.

Pour cela, l'approche développée se propose de créer des indicateurs sur lesquels se basent des mécanismes de détection. L'ensemble compose la stratégie de détection. Le premier indicateur signale l'entrée du système dans une zone critique et utilise la notion de distance aux états critiques. Cette dernière a été introduite dans (Sicard, Zamaï, et Flaus 2017a) et étend les travaux de (A. Carcano et al. 2011). Il s'agit d'un indicateur de risque pour le système. A partir de la modélisation de l'ICS, les algorithmes développés sont en mesure de calculer le nombre minimal d'actions à appliquer au système pour qu'il atteigne un état critique. Une comparaison est possible avec un randonneur se promenant au bord d'une falaise. Lorsque ce dernier est sur un chemin, le risque de chute est relativement faible, en revanche s'il marche en direction du vide, le risque augmente considérablement. Nous remarquons pour autant que cet indicateur ne fournit pas une probabilité d'occurrence. En effet, le chemin pris par le randonneur peut lui faire emprunter une arête pour atteindre le sommet. De la même manière, la loi de commande peut amener le système aux limites acceptables sans pour autant que ce comportement soit anormal. Enfin, dans le cas d'un fonctionnement en mode dégradé, cet indicateur peut permettre d'identifier l'intentionnalité de l'anomalie. Cependant, comme nous avons pu le voir précédemment, les hackers peuvent attaquer séquentiellement le système. Pour cela, un indicateur dédié à l'analyse du comportement du système au cours du temps sera mis en œuvre : la trajectoire. Cette dernière, définie dans (Sicard, Zamaï, et Flaus 2017b), correspond à l'évolution de la distance au cours d'une séquence événementielle ou temporelle. Elle traduit ainsi l'intention de nuire et observe le comportement du système avec un point de vue plus étendu que celui de la distance, qui est calculée à un moment donné. Ces deux derniers indicateurs veillent au respect des contraintes combinatoires du système. D'autres, comme celui d'appartenance aux fenêtres temporelles ou celui de durée moyenne d'activation, permettent de vérifier les spécifications temporelles du système. Enfin, un indicateur est dédié à la surveillance des commandes envoyées aux équipements afin d'éviter les sur-sollicitations. L'ensemble de ces indicateurs est utilisé pour détecter des attaques et aider les opérateurs dans l'analyse de la situation.

La Figure 20 illustre l'ensemble de l'approche S.A.F.E. développée au cours de ces travaux avec le point d'entrée principal, la méthodologie de conception des filtres et la phase d'exploitation qui couvrent la partie détection, la post-détection et l'amélioration du process. Dans le cadre de nos propositions, notre approche se focalisera sur la détection d'attaques en sortie des automates programmables industriels à destination des actionneurs. Nos apports se localisent donc au niveau des filtres de commandes et en particulier de leurs modèles. Concernant la partie filtres de comptes rendus, nous reprendrons à l'identique les modèles proposés par D. Cruette et P.Lhoste que nous compléterons d'indicateurs.

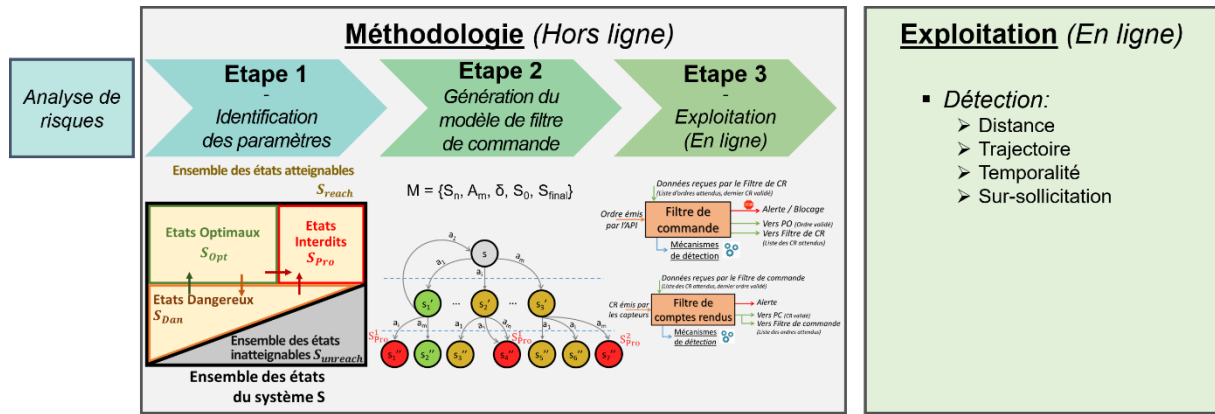


Figure 20 : Méthodologie et mise en application de l'approche S.A.F.E.

5. Hypothèses de travail et verrous scientifiques

Ce chapitre a permis de présenter l'approche S.A.F.E. qui est développée dans les parties suivantes. Nous allons proposer une série d'hypothèses afin de définir le cadre d'application de la méthodologie de conception des filtres et de la stratégie de détection. Ainsi, et comme défini dans le chapitre 3 – 3 Types d'attaques considérés, nous ne considérerons uniquement les attaques ayant l'intention de dégrader l'un des éléments suivants (hypothèse d'unicité de l'attaque) : les équipements de production, la qualité du produit ou la sécurité des biens et des personnes. En effet, l'approche S.A.F.E. a pour vocation de s'appuyer sur des modèles comportementaux de l'ICS pour détecter la dangerosité d'une séquence d'événements (actions/comptes rendus). Les attaques par déni de service ou de vol de données sont donc exclues de cette étude. En revanche, les attaques d'injection de données sur la *control channel*, la *report channel*, la corruption de l'automate ou la prise de commande de la supervision sont prises en compte dans notre approche, sous l'hypothèse qu'elles dégradent l'un des éléments dégradant l'ICS cités ci-dessus. En revanche, nous excluons les attaques sur le réseau de capteurs du périmètre des attaques considérées. Nous poserons également l'hypothèse que les capteurs sont considérés comme fiables au sens de (Villemeur 1992).

L'approche S.A.F.E. se positionne en complément des solutions et préconisations actuelles (ANSSI 2012). Nous considérons dans ces travaux que les défenses précédentes ont été franchies par l'attaquant, par voie informatique (spam et infiltration sur un réseau bureautique puis industriel) et/ou en local (intrusion dans la supervision ou sur un automate). Les niveaux supérieurs étant considérés comme partiellement ou totalement corrompus, notre approche se place donc comme le dernier rempart face à l'attaquant pour protéger le système physique et son environnement. Afin de développer notre approche, l'analyse de risques sera présentée comme connue et à disposition au début de la méthodologie (Pérès et al. 2007; González-Prida et al. 2016). Il s'agit bien ici d'un point d'entrée de notre méthode. De la même manière, le nombre d'états critiques est considéré comme connu et fini ainsi que la possibilité de les spécifier sur la base de la connaissance de la partie opérative du système. Il en est de même pour la modélisation des parties commande et opérative. Elle est considérée comme possible au sens où un état stable (les phénomènes transitoires ne sont pas pris en compte dans les modèles) peut être caractérisé

avec un ensemble de capteurs/actionneurs (sous réserve d'un nombre de paramètres nécessaires et suffisants). A l'issue de l'étape de modélisation, seul un modèle du filtre de commande est implanté dans le filtre de commande, le filtre de CR étant composé uniquement du modèle de la partie opérative. De plus, dans le cadre de ces travaux, seule la détection durant la phase de production est considérée puisque les étapes d'initialisations ou d'arrêts ne sont pas prises en compte dans les modèles générés. L'hypothèse de synchronisation entre les modèles contenus dans les filtres et le système physique est également émise. Enfin, ces travaux se limiteront uniquement à l'application de l'approche S.A.F.E. sur des systèmes de commande à événements discrets.

Enfin, le fonctionnement des filtres issus de cette méthodologie, est considéré comme non compromis en prenant notamment les hypothèses suivantes :

- Inattaquable sur le plan hardware (*secured by design* : conçus pour assurer les fonctions de prévention, de détection et de réaction face au risque cyber notamment par un durcissement face aux vulnérabilités découvertes (Aftabjahani et Das 2017)).
- Communication entre les filtres sur un réseau sécurisé différent du réseau industriel pouvant faire appel à des techniques de chiffrement et PKI (Rao, Srivastava, et Sreekanth 2017).
- Programmation des filtres dans un environnement « sain et sécurisé » au sens IT du terme.

Les travaux menés dans le cadre de cette thèse s'intéressent à la détection d'intrusions dans les systèmes de contrôle-commande industriels en proposant d'hybrider des approches IDS et Filtres en y incluant des modèles comportementaux. Cette approche s'axe autour des deux problématiques suivantes :

- Détection d'attaques brutales. Ces dernières agissent dans un intervalle de temps court où la détection doit être immédiate. Cette rapidité impose, à la fois, la localisation des filtres (proche de la PO, au niveau de la cellule pour réduire la complexité) et la méthodologie de conception des filtres. Ces derniers analysent les échanges au travers de modèles déterministes qui représentent le comportement du système physique soumis à des lois de commande. A cet effet, une méthodologie de conception a été proposée dans laquelle l'ensemble des ordres est identifié afin de détecter les attaques brutales immédiatement.
- Détection d'attaques plus sournoises (*Stealthy attack*). *A contrario* des précédentes, ces attaques agissent sur un intervalle de temps plus important comme les attaques par séquences, temporelles ou par sur-sollicitation. Ces modes opératoires empêchent leurs détections avec le mécanisme développé précédemment. Une stratégie de détection complémentaire permettant d'estimer l'occurrence d'une attaque avant que ses conséquences ne soient destructives est proposée. A cet effet, l'approche propose des concepts de distance d'un état caractérisé comme critique auxquels nous avons adjoint un second mécanisme dit de trajectoire dans le temps permettant de caractériser une intention de nuire au cours d'une séquence événementielle ou temporelle. Cette trajectoire permet par la suite la détection d'anomalies liées au non-

respect des contraintes combinatoires et temporelles spécifiées dans les modèles générés.

Ainsi, les verrous scientifiques concernent les modèles déterministes qui doivent être utilisés pour assurer la fonction de détection dans un contexte d'actes de malveillance cyber. Le premier verrou se situe dans la méthodologie de conception permettant d'obtenir le filtre de commande. Cette dernière doit aboutir à une généricité dans la synthèse des filtres. Enfin, l'établissement et l'utilisation d'indicateurs permettant de détecter des attaques, de complexités variables, affectant le comportement des ICSs représentent le deuxième verrou.

6. Bilan du chapitre

Ce chapitre a permis de souligner les spécificités des ICSs par rapport à des systèmes d'information « traditionnels » où la cybersécurité est prise en compte depuis plus longtemps. Ainsi, il apparaît qu'une approche sécurisant les couches basses des ICSs est nécessaire et complémentaire aux approches existantes. Ce positionnement définit l'originalité de ces travaux puisque l'utilisation de la connaissance « métier » est à la base de l'approche S.A.F.E. qui a été présentée, de la localisation et de la méthodologie de conception des filtres. La stratégie de détection repose sur des indicateurs capables de détecter des déviations pouvant conduire le système à des états critiques si aucune action correctrice n'est appliquée. Enfin, les verrous scientifiques qui font l'objet de ces travaux ont été présentés.

Partie 2 : Approche S.A.F.E. : Modélisation et Exploitation

Mécanismes de détection d'attaques ICS
et méthodologie de conception des filtres

Chapitre 4

Méthodologie de conception des filtres de commande

Ce chapitre est consacré à l'exposé de la méthodologie de conception des filtres de commande qui permettent d'implanter des mécanismes de détection d'ordres malveillants envoyés aux actionneurs. Le point d'entrée de cette démarche est l'analyse de risques qui identifie les zones critiques pour le système à protéger. La première étape consiste à définir le périmètre de l'ICS à protéger. Comme notre approche n'a pas vocation à sécuriser le système dans sa totalité, un ensemble d'entrées/sorties de l'automate contrôlant le sous-système est sélectionné. L'état du système se base sur les valeurs prises par les capteurs/actionneurs considérés. Les contraintes combinatoires (relations entre les états) et temporelles (fenêtres d'apparition de ces événements) du système sont alors spécifiées. Au cours de la deuxième étape, les modèles de la partie commande et de la partie opérative sont construits. Le premier utilise l'outil de modélisation réseaux de Petri afin de retranscrire l'objectif de production tout en vérifiant l'exactitude de la commande. La partie opérative est, quant à elle, modélisée à partir d'un automate fini déterministe. Un algorithme d'exploration des états du système permet de calculer toutes les évolutions possibles du système en appliquant toutes les actions possibles qui peuvent être exécutées par la PC. A la suite de l'exécution de ce dernier et après le report des types d'états qui ont été atteints (optimal, dangereux et interdit), le modèle du filtre de commande est obtenu. Enfin, la troisième et dernière étape consiste à la mise en ligne des filtres afin que les modèles alimentent les mécanismes de détection développés.

Table des matières

1.	Méthodologie générale de l'approche S.A.F.E.	83
2.	Étape 1 : Identification des paramètres du système	84
2.1.	Spécification des états du système	85
2.2.	Spécification des contraintes temporelles du système.....	87
3.	Étape 2 : Génération du modèle de filtre de commande.....	90
3.1.	Modèle de commande.....	90
3.2.	Modèle de la partie opérative.....	91
3.3.	Algorithme de génération du modèle.....	93

4.	Étape 3 : Exploitation (en ligne)	97
5.	Bilan du chapitre.....	97

1. Méthodologie générale de l'approche S.A.F.E.

Comme évoqué dans le chapitre précédent, la méthodologie développée en parallèle de l'approche S.A.F.E. permet de concevoir des filtres dotés de modèles comportementaux à partir de l'analyse de risques. La Figure 21 représente cette méthodologie qui est découpée en 3 étapes.

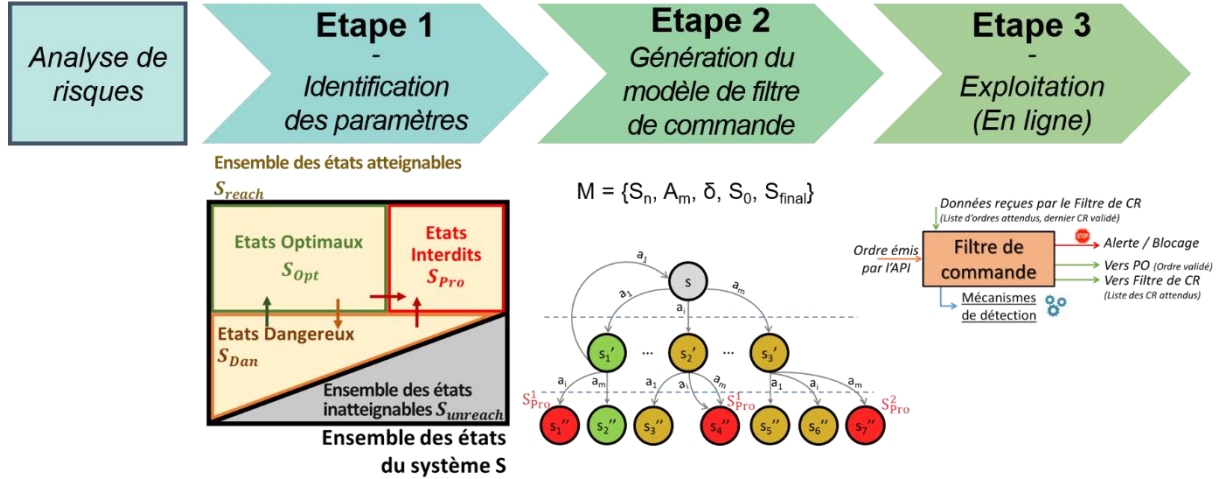


Figure 21 : Méthodologie de conception des filtres de commande

Notre approche prend pour hypothèse de départ qu'une analyse de risques (de type EBIOS par exemple, préconisé par l'ANSSI) a été préalablement menée sur le système de contrôle-commande considéré. Cette analyse constitue donc un prérequis pour la suite de la méthodologie que nous proposons. Nous invitons le lecteur à lire des travaux tels que (Villemeur 1992; Cherdantseva et al. 2016) pour plus de détails sur ces analyses de risques. Cette analyse devra mettre en exergue les événements redoutés pour le système considéré. Ces derniers induisent donc des parties de l'ICS qui doivent être protégées en priorité.

Sur cette base, la première étape de la méthodologie proposée identifie les paramètres à prendre en compte pour assurer la modélisation du système. Ces derniers correspondent aux entrées-sorties de l'API et définissent donc le périmètre du système à protéger. En effet, cette approche n'a pas vocation à sécuriser toute l'architecture de contrôle-commande mais uniquement les parties les plus critiques aboutissant aux événements redoutés les plus sévères. Les paramètres retenus dans cette première étape permettent alors de définir les états qui sont, par la suite, utilisés dans les modèles comportementaux. Un expert caractérise les états appartenant à la zone critique identifiée par l'analyse de risques. En parallèle, une caractérisation temporelle du système est faite. Celle-ci permet de spécifier les fenêtres temporelles des séquences d'ordres et de comptes rendus les uns par rapport aux autres afin de les utiliser ultérieurement dans les mécanismes de détection temporels.

L'étape 2 est définie par la modélisation de la loi de commande, de la partie opérative et de l'association des deux qui aboutit au modèle du filtre de commande. Le modèle de la loi de commande représente ce que nous souhaitons faire de l'ICS. Cette partie s'appuie sur l'outil de modélisation réseaux de Petri afin de bénéficier de leurs propriétés d'expression et de vérification.

En effet, dans notre approche le modèle est construit à partir des spécifications du cahier des charges et non pas du programme utilisé dans l'API. Le modèle de la partie opérative représente ce que le système physique peut faire. La modélisation se base sur un automate fini déterministe, identique à celui utilisé dans les travaux de (A. Carcano et al. 2011) et dont la description mathématique permet de construire un algorithme d'exploration des états. La mécanique d'exploration et les conditions d'arrêts utilisées permettent de converger rapidement vers un résultat. L'obtention de tous les états atteignables du système et les modèles de commande et de partie opérative permettent de les classer en trois catégories : optimaux, dangereux ou interdits. Ainsi, le modèle du filtre de commande est construit.

La dernière étape de la démarche s'effectue en ligne et correspond à l'exploitation des données remontées par les filtres de commande et de comptes rendus. Des indicateurs génériques permettent d'analyser les échanges entre la PO et la PC afin de détecter des anomalies. Les mécanismes de détection s'appuyant sur ces indicateurs identifient des attaques ne respectant pas les contraintes combinatoires, temporelles ou provoquant des sur-sollicitations et qui dégradent *in fine* la partie opérative.

Cette méthodologie a deux objectifs principaux :

- Intégrabilité en indiquant clairement les points d'entrée et de sortie de chacune des étapes de la méthodologie ainsi que les outils à utiliser.
- Reproductibilité sur différents systèmes de contrôle-commande industriels à protéger.

Bien évidemment, la méthodologie diffère en fonction des objectifs de détection détaillés dans le chapitre 3. Ainsi, dans le cas où l'objectif de protection recherche uniquement la sécurité des biens et des personnes, l'étape de modélisation de la loi de commande peut être sautée ainsi que celle de catégorisation. Dans ce chapitre et dans la suite de ce manuscrit, nous nous plaçons dans le cas le plus complet, à savoir, celui où le mode de protection exige la sécurité des biens et des personnes et la qualité du flux de produit avec une modélisation des parties commande et opérative ainsi que des contraintes temporelles. Enfin, tout au long de cette démarche, nous considérons les environnements d'analyse et de calcul comme sains et sécurisés vis-à-vis de menaces extérieures.

2. Étape 1 : Identification des paramètres du système

Fort de l'analyse de risques, les zones critiques du système ont été identifiées. Ces dernières peuvent être atteintes par le non-respect de contraintes combinatoires (combinaison d'une ou plusieurs actions demandées par l'API dans un ou plusieurs états particuliers du système) ou temporelles (timing des séquences d'apparition des commandes et des comptes rendus). Cette première étape de la méthodologie a le double objectif d'identifier les paramètres importants pour assurer la modélisation du système (contraintes combinatoires) et d'analyser les spécifications temporelles du système à protéger (contraintes temporelles).

Dans la suite de ce manuscrit, nous adopterons la convention suivante : un caractère en majuscule désignera un ensemble tandis qu'une minuscule représentera un élément de cet ensemble. Le Tableau 4 résume les principales notations utilisées.

2.1. Spécification des états du système

Considérons qu'une analyse de risques a été menée et a permis de définir les zones physiques susceptibles d'être dangereuses pour l'ICS. Cette étude conduit à définir un périmètre à protéger pour le système qui induit les entrées-sorties à prendre en compte pour caractériser l'état du système. En effet, en se basant sur (Mitchell et Chen 2014), il est possible de décrire le comportement d'un ICS en utilisant ses états stables comme illustré en Figure 22. Il s'agit d'une continuité des travaux de (Michel Combacau 1991). Sur la base de ces travaux, nous utilisons la notation employée dans (Mitchell et Chen 2014) et définissons 5 types d'états différents avec diverses propriétés.

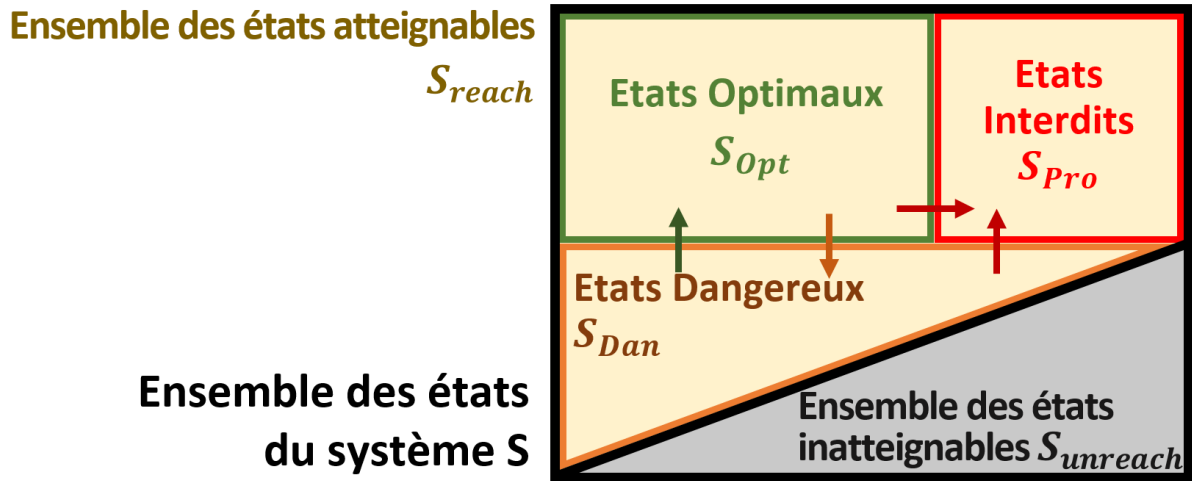


Figure 22 : Répartition des états possibles dans un système de contrôle-commande

En reprenant à notre compte les définitions de système de (Cassandras et Lafortune 2008) et de (Kumar, Cheung, et Marcus 1998), un ICS est décrit par un état unique correspondant à la combinaison des valeurs des capteurs, qui sont considérés comme non défaillants, et actionneurs comme indiqué dans l'équation (1).

$$s \in S_i \subset S, s = [\text{Capteurs}^i; \text{Actionneurs}^i] \quad (1)$$

Ainsi, les catégories suivantes peuvent être distinguées :

- État optimal $s_{opt} \in S_{opt} \subset S$. Cet état respecte à la fois les contraintes imposées par la loi de commande et par la partie opérative. La partie commande et les opérateurs cherchent autant que faire se peut à contenir l'évolution de l'ICS dans ce sous-ensemble d'états pour garantir l'efficacité de la production du produit initial au produit final. Par exemple, pour un système à vérin, si l'ordre de sortie est donné, alors l'état optimal correspond au compte rendu indiquant que le capteur de fin de course est activé.

- État dangereux $s_{Dan} \in S_{Dan} \subset S$. Les états de ce sous-ensemble respectent uniquement les contraintes liées à la partie opérative. En revanche, le système viole les contraintes imposées par la loi de commande. Des dommages peuvent être infligés à l'ICS, puisque ce dernier fonctionne en mode dégradé, mais ils restent négligeables. Toutefois, puisque la loi de commande n'est pas respectée, la qualité du flux de production peut être impactée, comme le temps de production ou encore l'usure des équipements. Par exemple, pour un système de cuves, si la loi de commande impose un remplissage par deux vannes successivement alors un état dangereux est obtenu si le remplissage s'effectue par les deux vannes simultanément. Le système n'est pas dans un état critique (débordement par exemple) mais les spécifications de la commande n'ont pas été respectées pour autant.
- État interdit $s_{Pro} \in S_{Pro} \subset S$. Les états appartenant à cet ensemble représentent la zone critique identifiée par l'analyse de risques pour le système de contrôle-commande. En effet, les contraintes de la loi de commande et de la partie opérative ne sont pas respectées ce qui entraînent des dégâts importants. La différence entre ce sous-ensemble S_{Pro} et celui des états dangereux S_{Dan} tient dans la dangerosité des dégâts infligés à l'ICS. Par exemple, pour un système de vérins en L, cet état correspond à la collision entre les deux vérins lorsqu'ils sont sortis. Cet état peut être atteint lorsque deux ordres de sortie de vérins sont envoyés.
- État atteignable $S_{Reach} \subset S$. Cet ensemble fait référence à tous les états qu'il est possible d'atteindre en utilisant les actions disponibles contenues dans l'ensemble A. Les sous-ensembles S_{Opt} , S_{Dan} et S_{Pro} sont inclus dans S_{Reach} .
- État inatteignable $S_{Unreach} = S \setminus S_{Reach}$. Cet ensemble est le complément de S_{Reach} et regroupe les états qui ne sont pas atteignables avec les actions de l'ensemble A. Par exemple pour le système de cuves, un état inatteignable est l'activation du capteur haut de la cuve (remplissage total) alors que le capteur de fond de cuve est inactif (cuve vide).

La Figure 22 illustre donc les différents états possibles dans un système de contrôle-commande industriel. Nous notons également A_{Opt} l'ensemble des ordres qui respectent les contraintes de la loi de commande et respectivement S_{Pre} et A_{Pre} l'ensemble des états et ordres prédits par l'un des filtres et attendus par l'autre.

Tableau 4 : Principales notations utilisées dans le manuscrit

S	Ensemble des états possibles du système
S_{Reach}	Ensemble des états atteignables, $S_{Reach} \subset S$
$S_{Unreach}$	Ensemble des états inatteignables, $S_{Unreach} = S \setminus S_{Reach}$
S_{Opt}	Ensemble des états optimaux, $S_{Opt} \subset S_{Reach}$
S_{Pro}	Ensemble des états interdits, $S_{Pro} \subset S_{Reach}$
S_{Dan}	Ensemble des états dangereux, $S_{Dan} = S_{Reach} \setminus (S_{Opt} \cup S_{Pro})$
S^i	Ensemble des états de la $i^{ème}$ itération
s_i	$i^{ème}$ état du système
S_0	Ensemble des états initiaux

S_{final}	Ensemble des états finaux (Automate Fini Déterministe M)
S_{Prev}	Ensemble des états déjà atteints durant une précédente itération de l'AFD M
S_{Pre}	Ensemble des états prédits par un filtre et attendus par l'autre
A	Ensemble des actions possibles
a_j	$J^{\text{ème}}$ action de l'ensemble des actions possibles
A_{Opt}	Ensemble des ordres respectant la loi de commande
A_{Pre}	Ensemble des actions prédites par un filtre et attendues par l'autre
δ	Fonction de transition appliquée aux états et aux actions
Δ	Fonction de transition appliquée aux ensembles d'états et d'actions

Afin de définir exactement les états $s \in S$ de notre système, il est important de déterminer le nombre de paramètres nécessaires et suffisants pour correctement modéliser le système. En effet, l'approche S.A.F.E. n'a pas vocation à protéger l'ensemble de l'ICS mais seulement les parties jugées les plus critiques définies par l'analyse de risques. Ainsi, la détermination du nombre correct de paramètres est une problématique essentielle. Les modèles établis ne considèrent qu'une seule cellule composée d'un unique API. L'évolution vers une approche intégrant plusieurs cellules sera abordée dans la partie Perspectives. La caractérisation des états du système permet de mettre en évidence le comportement de l'ICS et est à la base des mécanismes de détection des attaques faisant tendre le système vers un état interdit.

2.2. Spécification des contraintes temporelles du système

L'analyse temporelle est à la base des mécanismes de détection permettant d'identifier les attaques temporelles. Ces attaques respectent les contraintes liées à la combinatoire et à la séquentialité des échanges entre la partie commande et la partie opérative mais en modifiant les contraintes temporelles (timing). Une contrainte temporelle est un ensemble de propriétés caractérisant un événement (action ou compte rendu) dans le temps par rapport à un déclenchement, une durée d'activation (ou de désactivation) et un autre événement (retard, marge). L'ensemble des contraintes temporelles d'un système définit des fenêtres temporelles qui spécifient le comportement temporel du système pour chaque action et compte rendu possibles. La Figure 23 illustre les éléments caractéristiques de ces fenêtres temporelles. De manière explicite, si une action $a_j \in A$ doit être appliquée à une date t_a durant une certaine période τ et qu'un attaquant envoie un compte rendu conduisant l'arrêt de l'action a_j à une date t_b telle que $t_b < t_a + \tau$ alors la qualité du flux de production en est affectée. Par exemple, en considérant un vérin ayant reçu l'ordre de sortie à l'instant t_a , le compte rendu attestant de la bonne exécution de l'action arrivera à l'instant $t_a + \tau$, τ représentant le temps nécessaire au vérin pour arriver au capteur de fin de course. Une attaque temporelle consisterait à envoyer un compte rendu attestant la sortie du vérin avant la date $t_a + \tau$.

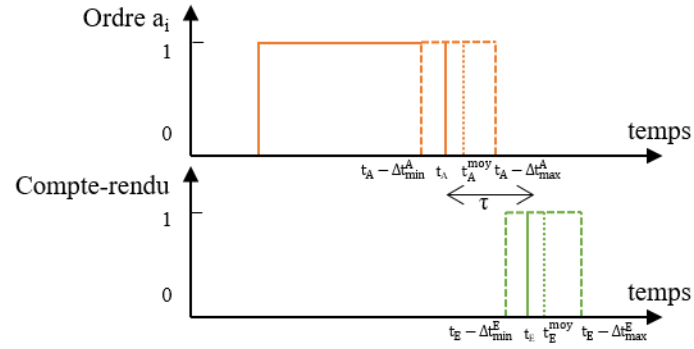


Figure 23 : Illustration des spécifications temporelles

Ainsi, nous mettons en évidence que les contraintes combinatoires ne permettent pas de sécuriser complètement le système. En effet, dans l'exemple précédent, l'enchaînement des événements (ordre/CR) est normal au niveau combinatoire mais ne l'est pas au niveau temporel. Ainsi, ces contraintes combinatoires se doivent d'être complétées par des spécifications temporelles qui bornent les actions et les comptes rendus.

Plusieurs travaux ont exploré le domaine de la détection d'intrusions basée sur l'étude des spécifications temporelles. (Taylor, Japkowicz, et Leblanc 2015) ont travaillé sur des approches défensives sur le bus CAN détectant l'insertion de paquets sur le réseau de communication. L'algorithme développé calcule le temps moyen entre chaque paquet et le compare à un historique afin de détecter des anomalies. La principale limitation de ces travaux se situe dans l'hypothèse forte de périodicité de paquets échangés. Le mécanisme devient sensible aux aléas pouvant apparaître sur un ICS. Les travaux de (Kleinmann et Wool 2017) et (Pan et al. 2013) prennent également le temps en considération dans leurs approches respectives pour construire des modèles et évaluer la sécurité des systèmes sur des scénarii d'attaques. Dans (Kleinmann et Wool 2017), les auteurs modélisent le trafic de communication entre le PLC et l'IHM. Cette approche met en œuvre des connaissances sur les états atteignables de l'API après une requête ou une réponse. L'algorithme, qui a été testé sur le protocole Siemens S7-0x72, donne un faible taux de faux positifs mais nécessite une longue période d'apprentissage puisque la méthode est basée sur la reconnaissance de formes cycliques. Dans (Pan et al. 2013), le temps est utilisé pour dérouler le scénario d'attaque sans pour autant forcément prendre en compte les spécifications temporelles du système évalué. (Mohan et al. 2013) ont développé une approche basée sur la caractérisation de l'exécution temporelle sur des systèmes temps-réel. Ainsi, pour chaque tâche devant être effectuée, une durée d'activation caractéristique a été définie. Une détection intervient lorsque la tâche active est trop longue ou trop courte par rapport au temps de référence. Des propriétés annexes ont par la suite été rajoutées par les auteurs afin d'améliorer la robustesse du mécanisme comme le temps moyen d'exécution, le meilleur et le pire temps d'exécution. L'approche développée dans les précédents travaux s'apparente aux fenêtres temporelles développées dans ces travaux de recherche, toutefois, les deux mécanismes introduisent une vulnérabilité pouvant être exploitée par un attaquant. En effet, les limites sont arbitrairement définies par un système expert ou par apprentissage ce qui introduit une incertitude dans leur établissement. Enfin, dans (Zimmer et al. 2010), l'objectif est de détecter des injections de code malveillant. Pour ce faire, des appels synchrones et asynchrones sont utilisés pour comparer l'exécution temporelle du code.

L'inconvénient majeur de cette approche est la compatibilité avec les contraintes temps-réel ainsi que la gestion des interruptions présentes dans de tels systèmes.

Des approches dans le domaine de la sûreté apportent également des résultats pertinents pour la surveillance et le diagnostic. Les travaux de (Loures 2006) regroupent plusieurs de ces approches dont les chroniques temporelles de (Valette et al. 1996) qui permettent de définir les dates d'arrivée d'un événement attendu.

La majorité des approches du domaine de la sécurité sont basées sur l'apprentissage de la temporalité de ces systèmes. Or, notre étude est localisée à bas niveau de l'architecture ICS, en particulier entre un automate programmable industriel et une partie opérative. Ce positionnement réduit à l'évidence la complexité du système considéré et donne de fait du sens à une approche à base de modèles déterministes. Ainsi, des approches évaluent et comparent les durées d'exécution des tâches pour déterminer une durée de référence et détecter une anomalie par comparaison avec la durée de la tâche actuelle. Bien que ces techniques soient actuellement mises en place sur des composants de la partie commande ayant des capacités de calcul supérieures à celles des filtres de notre approche, le principe de détection est retenu. Enfin, l'approche par chroniques temporelles de (Valette et al. 1996) qui permet de définir les spécifications temporelles du système autour de fenêtres d'apparitions d'événements est reprise dans notre approche.

Fort de ces approches, nous chercherons à baser notre mécanisme sur la caractérisation du temps d'exécution de chaque action ainsi que de celui de chaque compte rendu. Pour ce faire, nous noterons t_{Aj} le temps caractéristique du déclenchement de l'action $a_j \in A$ et t_{Ej} celui correspondant à l'apparition du compte rendu attestant de son exécution. Afin de prendre en compte les aléas pouvant intervenir sur un ICS, une marge Δt est définie pour chacun de ces temps d'apparition. Ainsi, la fenêtre temporelle FT_{Ordre} caractérisant les ordres d'un ICS s'écrit comme dans l'équation (2).

$$FT_{Ordre} = [t_A - \Delta t_{min}^A; t_A + \Delta t_{max}^A] \quad (2)$$

De la même manière, la fenêtre temporelle FT_{CR} spécifiant les comptes rendus d'un ICS s'écrit comme dans l'équation (3) et permet de définir le temps et la durée d'activation.

$$FT_{CR} = [t_E - \Delta t_{min}^E; t_E + \Delta t_{max}^E] \quad (3)$$

T^{moy} (cf. Figure 23) représente le temps moyen d'apparition de l'événement considéré et il évolue au cours du temps. A l'opposé, Δt_{min} et Δt_{max} sont constants et fixés lors de cette phase d'identification des paramètres du système. Ils définissent les extremums temporels d'activation d'un événement. Pour finir, un retard τ est utilisé pour définir un écart entre la fin d'un événement (une action par exemple) et l'activation d'un autre (compte rendu attestant de sa réalisation).

Cette étape d'étude des spécifications temporelles est indispensable à la réalisation de l'algorithme de détection temporel développé dans la suite des travaux. Son écriture ajoute une dimension dans la caractérisation du comportement du système et permet également de rajouter un niveau de protection vis-à-vis d'un attaquant pour l'ICS.

3. Étape 2 : Génération du modèle de filtre de commande

Dans cette étape, le système de contrôle-commande à protéger est modélisé (partie commande et partie opérative) et un algorithme d'exploration des états du système permet d'identifier tous les états atteignables. Ainsi, le modèle du filtre de commande, produit à partir des deux précédents modèles, est implanté dans le filtre de commande. Le modèle de la partie opérative est implanté dans le filtre de comptes rendus, comme dans l'approche Filtre originelle. La caractérisation qui a été faite dans l'étape 1 a permis de catégoriser ces états selon trois types. Le modèle de commande, après avoir vérifié la cohérence de la logique programmée, fournit les états optimaux dans lesquels le système devra évoluer. L'analyse de risques, obtenue en prérequis de cette méthodologie, a identifié les zones critiques pour l'ICS et qui ont été traduites sous la forme d'états interdits. Les états dangereux en sont déduits puisqu'ils forment le troisième sous-ensemble des états atteignables.

La construction de chaque modèle ainsi que celle de l'algorithme d'exploration des états du système sont détaillées dans les paragraphes suivants.

3.1. Modèle de commande

Le modèle de commande représente l'objectif de production (ce que doit faire le système physique) afin de satisfaire le besoin client. Pour cela, sa construction nécessite l'accès aux spécifications de la logique de contrôle programmée dans l'API. Nous rappelons que, dans le cas où seul le mode de protection de la sécurité des biens et des personnes est requis, la construction de ce modèle n'est pas nécessaire. Une solution aurait pu consister à récupérer directement le code automate pour en extraire la partie loi de commande. Cependant, dans ce cas, rien ne nous garantit contre d'éventuelles erreurs de programmation de la part de l'intégrateur, de présence de codes morts suite à des reprises multiples ou encore de compromission du code liée à un acte malveillant comme dans Stuxnet (Kushner 2013). Dans ce dernier cas, notre approche se verrait à l'évidence inefficace.

Pour ces raisons, nous avons choisi de repartir des spécifications clients pour développer des lois de commande à partir d'un outil de modélisation permettant aussi la vérification des bonnes propriétés (pas de blocages, absence de conflits, réinitialisation, etc...).

L'outil de modélisation satisfaisant ces contraintes pour établir le modèle de la loi de commande est naturellement le Réseau de Petri (RdP) (Petri 1966; Peterson 1977). Leurs propriétés, à la fois graphiques et mathématiques, permettent de fournir un environnement uniforme pour modéliser, concevoir et analyser des systèmes industriels (Zurawski et Zhou 1994; Zhang et al. 2018; Chaillet, Combacau, et Courvoisier 1993). Les RdP sont un moyen d'expression de la commande qui permettent de représenter des contraintes de séquentialité, de parallélisme, de synchronisation et d'exclusion mutuelle (Zurawski et Zhou 1994; Murata 1989; Michel Combacau, Esteban, et Nketsa 2005). La Figure 24 illustre les différentes propriétés citées ci-

dessus sous forme graphique ainsi que la définition d'une activité avec les RdP, indispensables à la modélisation de la commande (Michel Combacau, Esteban, et Nketsa 2005).

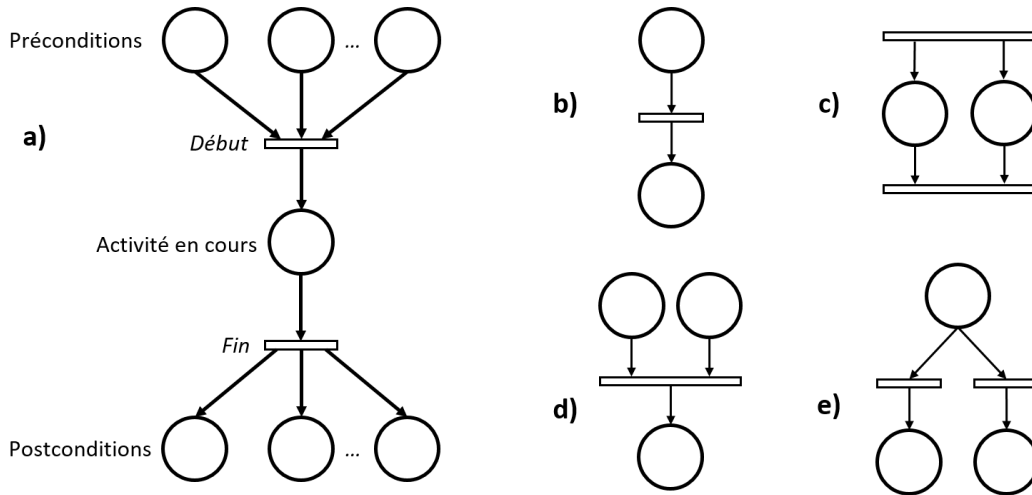


Figure 24 : Illustration des propriétés graphiques des RdP – a) Modélisation d'une activité, b) Séquentialité, c) Parallélisme, d) Synchronisation, e) Exclusion mutuelle

Des informations complémentaires sur les Réseaux de Petri peuvent être trouvées en Annexe 1. Les RdP sont également un outil de vérification qui permet de s'assurer du bon fonctionnement (RdP borné, vivant et réinitialisable) de la logique de programmation embarquée dans l'API.

3.2. Modèle de la partie opérative

Le modèle de la partie opérative représente les possibilités offertes par le système physique. L'objectif est de détailler toutes les évolutions possibles du système, c'est-à-dire l'ensemble des états atteignables, sur la base de la description des actions possibles et de leurs effets sur le système. L'ensemble des actions possibles, noté A , représente les ordres pouvant être émis par l'automate à savoir l'activation seule ou combinée d'un ou plusieurs actionneurs. Dans un système de contrôle-commande, le nombre d'actions possibles pouvant être exécutées par la partie commande sur la partie opérative est fini. Ainsi, le nombre d'états atteignables S_{Reach} l'est également. L'outil de modélisation choisi pour représenter le comportement de la PO est l'automate fini déterministe (AFD), dans la continuité des travaux de (A. Carcano et al. 2011) dont nous adapterons la notion de distance. Il permet la description du système en ensembles et sous-ensembles d'éléments (état, action) permettant, par la suite, le développement d'un algorithme d'exploration du système. De plus, les AFD utilisent la fonction de transition pour décrire les évolutions du système modélisé dont le formalisme permet de réutiliser les travaux de (Sébastien Henry, Zamaï, et Jacomino 2012) qui nous seront utiles afin de modéliser les effets des actions sur la partie opérative. Il s'agit d'un graphe orienté évoluant d'un état à un autre, à partir d'un état initial et jusqu'à des états finaux, en fonction des mots, composés de lettres appartenant à un alphabet, qui lui sont appliqués (Pin 2006). La Figure 25 illustre un exemple d'automate fini déterministe. En effet, le comportement est clairement défini pour chaque lettre de l'alphabet et le nombre d'états possibles est fixé à trois. Chaque lettre de l'alphabet propre à

l'automate permet de faire évoluer les états de ce dernier. Enfin, nous remarquerons que les états 2 et 3 sont des états finaux comme l'indiquent les flèches sortantes.

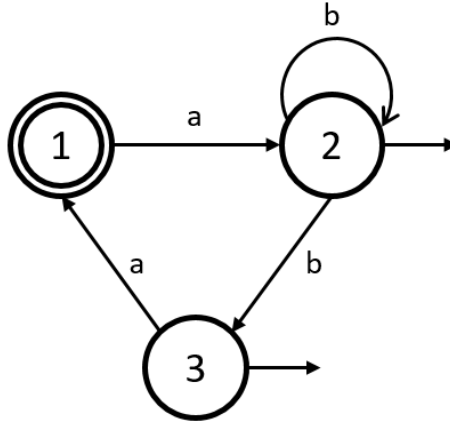


Figure 25 : Représentation d'un automate fini déterministe.

Cette représentation en automate déterministe a été utilisée dans la littérature pour alimenter des mécanismes de détection orientés cybersécurité ou orientés défaillances comme dans (Sayed-Mouchaweh 2018). De plus, les travaux de (Sébastien Henry, Zamaï, et Jacomino 2012) sur lesquels nous nous appuierons dans la suite de ce paragraphe, utilisent également des automates afin de modéliser le système pour reconfigurer la loi de commande. Enfin, dans le cas de systèmes hybrides, intégrant des variables continues et discrètes, des automates hybrides peuvent être utilisés comme dans (Zhou, Biswas, et Feng 2015) et (Belkacem et al. 2016). Dans ces systèmes, l'automate comporte des composantes supplémentaires pour modéliser le comportement continu qui peut varier en fonction de l'état du système.

L'équation (9) définit l'automate fini déterministe M utilisé pour modéliser la partie opérative avec le quintuplé suivant :

$$M = \{S_n, A_m, \delta, S_0, S_{final}\} \quad (9)$$

- S_n est l'ensemble fini des états possibles pouvant être atteints par l'ICS depuis l'état initial, en d'autres termes S_{Reach} , avec n représentant le nombre total d'états s_i . Le vecteur s_i modélise l'état du système et peut appartenir à l'un des trois sous-ensembles suivants : états optimaux S_{Opt} , états dangereux S_{Dan} et états interdits S_{Pro} ,
- A_m est l'ensemble fini des ordres qui peuvent être appliqués par l'ICS. Il correspond à l'ensemble A défini dans le paragraphe 2.1 avec l'indice m qui indique le nombre total d'actions a_j ,
- δ est la fonction de transition de l'automate M qui représente la relation entre les états et les ordres du système. Cette fonction permet de modifier l'état de l'automate M en simulant l'évolution du système modélisé. Dans ce but, chacune des actions doit être caractérisée dans le modèle afin de pouvoir restituer son effet sur le processus physique,
- $S_0 \subset S_{Reach}$ est l'ensemble des états initiaux du système,
- S_{final} est l'ensemble des états finaux de l'automate. Lorsque l'un de ces états est atteint, l'évolution de l'automate est stoppée. Ainsi, les éléments de cet ensemble sont les

conditions d'arrêt de l'algorithme d'exploration des systèmes développé dans le prochain paragraphe 3.3.

En reprenant l'exemple illustré en Figure 25, l'équation (10) permet de définir l'automate par le quintuplé suivant en reprenant l'équation (9) :

$$M = \{\{1, 2, 3\}, \{a, b\}, \delta, \{1\}, \{2, 3\}\} \quad (10)$$

Afin de compléter la modélisation de la partie opérative, il est essentiel de spécifier la fonction de transition δ de l'automate. Cette fonction permet de modéliser les effets, les conditions et les contraintes associées pour chaque actionneur sur le flux de production. Pour ce faire, nous nous appuyons sur les travaux de (Sébastien Henry, Zamaï, et Jacomino 2012). Pour chaque opération du système, une fiche spécifiant les contraintes s'appliquant à l'action est réalisée. Des caractéristiques s'appliquant à la ressource ou au produit comme les précontraintes, les contraintes, les états initiaux, intermédiaires et finaux sont détaillées. Ces informations peuvent être mises en œuvre afin de caractériser le comportement de la partie opérative. Ainsi, le comportement du système vis-à-vis de chaque action possible peut être spécifié. L'équation (11) permet de définir la fonction de transition δ qui associe un état s' à un couple {état s ; action a }. Nous noterons $S_i \subset S_n$ l'ensemble auquel appartient l'état avant d'appliquer la fonction de transition δ et $S_{i+1} \subset S_n$ l'ensemble auquel appartient l'état après son évolution.

$$\forall s \in S_i \subset S_n, \forall a \in A_m \text{ tel que } s' = \delta(s, a) \in S_{i+1} \subset S_n \quad (11)$$

Afin de conclure ce paragraphe, nous noterons que si le modèle de commande comporte plusieurs ordres en parallèle, notre modèle de la partie opérative devra alors :

- Modéliser toutes les combinaisons de chaque ordre possible.
- Faire appel à d'autres automates M décrivant les autres évolutions possibles.

Dans le cadre de ces travaux de recherche, nous nous sommes orientés vers la première solution puisque les systèmes modélisés se limitaient à la taille d'une seule cellule automate. En revanche, si l'approche venait à sécuriser plusieurs cellules, un changement d'outil de modélisation serait à envisager (RdP par exemple), ce qui sera discuté en perspective de ces travaux.

3.3. Algorithme de génération du modèle

Sur les bases des modélisations des parties commande et opérative, l'algorithme développé a pour objectif principal d'explorer les évolutions possibles et atteignables de la partie opérative en utilisant l'ensemble des actions pouvant être exécutées par la PC sur la PO. Cet algorithme se rapproche d'une approche intégrée où le comportement normal et, en partie, anormal est modélisé. En effet, à l'aide du modèle de la loi de commande, il est possible d'identifier les états appartenant à l'ensemble des états optimaux S_{Opt} . L'étape 1 a permis de caractériser les états interdits appartenant à l'ensemble S_{Pro} . Les états dangereux, complément des deux derniers sous-ensembles cités de l'ensemble des états atteignables S_{Reach} , en sont facilement déduits. Les comportements dangereux pour le système peuvent ainsi être mis en évidence. Cependant, comme les défaillances matérielles ne sont pas modélisées, les états afférents ne sont pas modélisés non

plus et ne sont donc pas pris en compte par notre algorithme (états non-atteignables avec les actions possibles) ce qui induit un comportement anormal non exhaustif.

Toutefois, comme indiqué en Figure 22, le système peut évoluer d'un type d'état à un autre dans le sous-ensemble des états atteignables S_{Reach} . Ainsi, si nous appliquons une action $a_j \in A$, notre modèle permet de faire évoluer la partie opérative d'un état :

- Optimal $s_{Opt} \in S_{Opt}$ à un état dangereux $s_{Dan} \in S_{Dan}$ ou interdit $s_{Pro} \in S_{Pro}$,
- Dangereux $s_{Dan} \in S_{Dan}$ vers un état optimal $s_{Opt} \in S_{Opt}$ ou interdit $s_{Pro} \in S_{Pro}$.

Pour ce faire, l'algorithme 1 se base sur la fonction de transition δ définie dans l'équation (11) et qui associe un nouvel état $s' \in S^{i+1}$ en appliquant une action $a \in A$ à un état $s \in S^i$. Cette relation est généralisée et une nouvelle fonction de transition Δ est définie sur les ensembles d'états (Sicard, Zamaï, et Flaus 2018a). Ainsi, l'ensemble des états fils appartenant à l'ensemble $S^{i+1} \subset S_n$ est obtenu en appliquant l'ensemble des actions possibles A_i aux états appartenant à l'ensemble S^i tel que défini dans l'équation (12) :

$$S^{i+1} = \Delta(S^i, A) = \{ s' = \delta(s, a) | s \in S^i, a \in A \} \subset S_{Reach} \quad (12)$$

L'ensemble des états interdits S_{Pro} a été identifié dans le paragraphe 2.1. en combinant les zones critiques caractérisées par l'analyse de risques et les paramètres définissant un état. Un des objectifs de l'algorithme 1 est de détecter des contextes qui conduisent le système dans des états interdits. Un contexte est défini comme un couple état/action qui résulte d'un état critique $s_{Pro} \in S_{Pro}$ lorsque l'action est appliquée sur le système dans cet état (Sicard, Flaus, et Zamaï 2017; Sicard, Zamaï, et Flaus 2017b). L'équation (13) définit mathématiquement un contexte en utilisant la fonction de transition δ :

$$\exists s \in S^i \subset S_{Reach}, \exists a \in A \text{ tel que } s' = \delta(s, a) \in S^{i+1} \subset S_{Pro} \quad (13)$$

Ainsi, l'algorithme 1 calcule l'image des états de l'itération S^i au travers de la fonction de transition Δ à l'aide de l'ensemble des actions possibles A dans le but d'identifier tous les contextes menant à des états critiques. Le deuxième objectif de l'algorithme 1 est d'explorer toutes les évolutions possibles du système et de catégoriser les différents états atteints. Pour ce faire, toutes les actions possibles doivent être appliquées à chacun des états atteignables, en commençant par l'ensemble des états initiaux S_0 , dans le but de réaliser les deux objectifs. Cependant, afin de contenir les problématiques de convergence (le temps de calcul important, lié au nombre de paramètres du modèle, combiné à l'exploration de branches déjà rencontrées peut aboutir à une non convergence de l'algorithme), l'algorithme 1 dispose de plusieurs conditions d'arrêts (Sicard, Zamaï, et Flaus 2018a). Ces dernières stoppent l'exploration de la branche en cours comme un état appartenant à l'un des ensembles suivants :

- États initiaux S_0 : $S^{i+1} \cap S_0 \neq \{\emptyset\} \leftrightarrow \exists s' \in S^{i+1} \text{ tel que } s' \in S_0$.
- États interdits S_{Pro} : $s' \in S_{Pro} \subset S_{Reach}$. Lorsque cette condition est rencontrée, l'algorithme 1 n'a plus d'évolution possible et le système industriel est gravement impacté : $S^{i+1} \cap S_{Pro} \neq \{\emptyset\} \leftrightarrow \exists s' \in S^{i+1} \text{ tel que } s' \in S_{Pro}$. Dans l'algorithme 1, l'ensemble F désigne l'intersection entre les ensembles S^{i+1} et S_{Pro} .

- États précédemment rencontrés S_{prev} . Lorsque l'algorithme 1 calcule un état déjà rencontré lors d'une précédente itération, alors l'exploration de la branche est immédiatement stoppée puisque l'on connaît déjà son évolution : $S^{i+1} \subset \cup S^i \leftrightarrow \exists s' \in S^{i+1} \text{ tel que } s' \in \cup S^i$. Dans l'algorithme 1, l'ensemble P désigne l'intersection entre les ensembles S^{i+1} et $\cup S^i$.

Cet algorithme garantit une exploration de tous les états atteignables en appliquant toutes les combinaisons d'actions et en établissant des conditions d'arrêts interdisant l'exploration de branches d'états déjà rencontrés. A la fin de l'exploration, la séquence d'ordres issue du modèle de la loi de commande est appliquée au modèle de la partie opérative afin d'obtenir l'ensemble des états optimaux S_{Opt} . Par déduction, et avec l'hypothèse selon laquelle l'ensemble des états atteignables est exploré, l'ensemble des états dangereux S_{Dan} est établi. L'algorithme 1 est détaillé ci-dessous, ainsi qu'un exemple d'exploration d'états en Figure 26, Figure 27 et Figure 28.

Fonction Trouver tous les états possibles atteignables

```

 $S^i = S_0$ 
WHILE ( $S^i \neq \{\emptyset\}$ )
     $S^{i+1} = \Delta(S^i, A)$ 
    IF ( $S^{i+1} \cap S_{\text{Pro}} \neq \{\emptyset\}$ ) THEN
         $S^{i+1} \cap S_{\text{Pro}} = F$ 
        FOR ( $\exists s' \in S^{i+1}, s' \in S_{\text{Pro}}$ )
            Context = {Context, F}
        ENDFOR
    ENDIF
    IF ( $S^{i+1} \cap S_0 \neq \{\emptyset\}$ ) THEN
         $S^{i+1} \cap S_0 = G$ 
         $\exists s' \in S^{i+1}, s' \in S_0$ 
    ENDIF
    IF ( $S^{i+1} \cap S_{\text{Prev}} \neq \{\emptyset\}$ ) THEN
         $S^{i+1} \cap S_{\text{Prev}} = H$ 
        FOR ( $\exists s' \in S^{i+1} \text{ s.t. } s' \notin S_{\text{Prev}}$ )
             $S_{\text{Prev}} = \{S_{\text{Prev}}, s'\}$ 
        ENDFOR
    ELSE
         $S_{\text{Prev}} = S_{\text{Prev}} \cap S^{i+1}$ 
    ENDIF
     $S^i = S^{i+1} \setminus (F \cup G \cup H)$ 
DONE
End
    
```

Algorithme 1. Algorithme d'exploration de tous les états possibles atteignables du système

La Figure 26 représente le mécanisme d'exploration des états du système de l'algorithme 1 ainsi que l'utilisation des conditions d'arrêts sur un exemple illustratif. En effet, l'exploration des états n'est pas finie puisque les évolutions possibles des états obtenus lors de la deuxième itération S^2 n'ont pas été recherchées. Cependant, nous pouvons remarquer que :

- L'action a_I appliquée à l'état s_I conduit à l'état initial $s \in S_0$,

- L'action a_m appliquée à l'état s_3' mène à l'état interdit $s_{Pro}^1 \in S_{Pro}^1 \in S_{Pro}$. La paire $\{s_3', a_m\}$ est identifiée comme contexte et l'exploration de la branche est arrêtée,
- L'état s_3'' a déjà été rencontré lors d'une précédente itération avec l'état s_1' tel que $s_3'' = \mathfrak{A}(s_3', a_i) = s_1'$. Là encore, l'exploration de la branche est interrompue et l'état s_3'' est retiré de la liste des états à explorer lors de la prochaine itération comme l'indique la Figure 27.

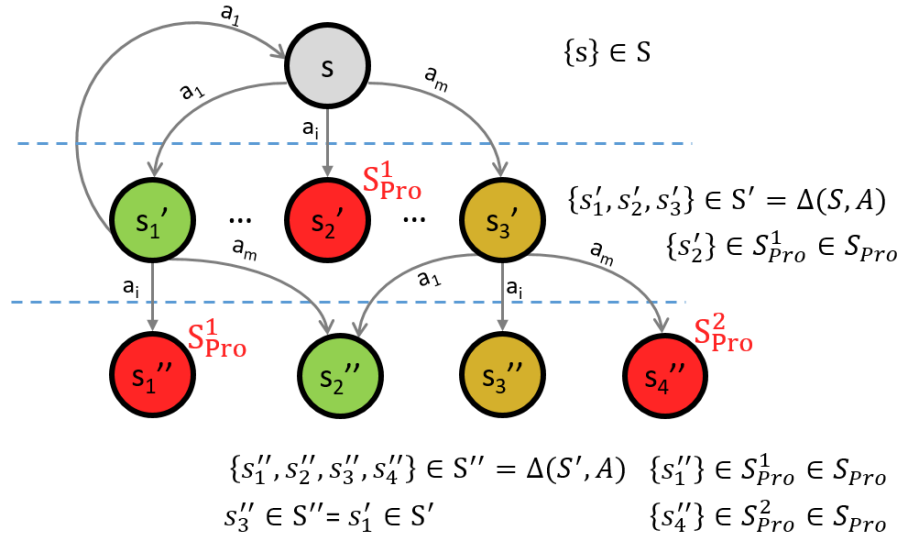


Figure 26 : Illustration des conditions d'arrêts de l'algorithme 1 (exploration des états en cours) [Gris : état initial, vert : état optimal, jaune : état dangereux et rouge : état interdit]

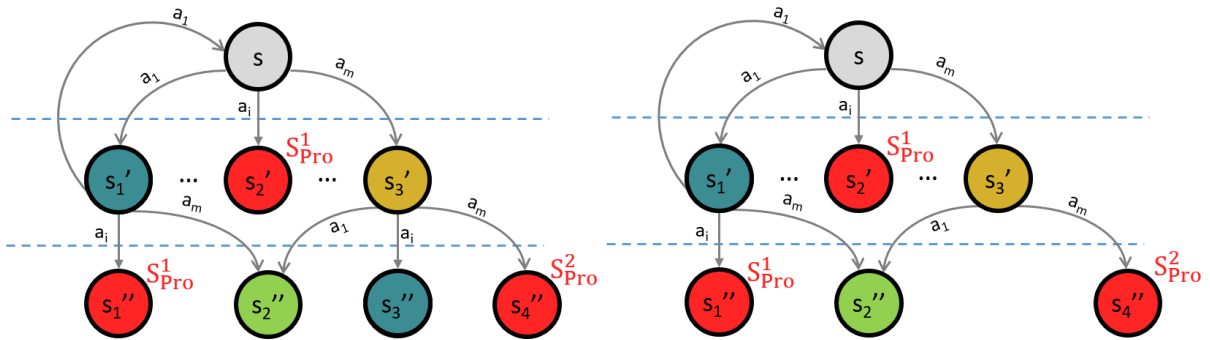


Figure 27 : Illustration de la suppression d'un état déjà rencontré de la liste des états à explorer lors de la prochaine itération (état s_3'') [Bleu : états identiques]

Ainsi, l'algorithme 1 ne s'arrête que lorsque l'ensemble des états à explorer est vide, ce qui signifie que toutes les évolutions possibles ont été évaluées. Ainsi, une unique séquence état/action (branche) est calculée à chaque itération. L'exploration complète du système pris pour exemple est disponible en Figure 28. Il s'agit du modèle du filtre de commande.

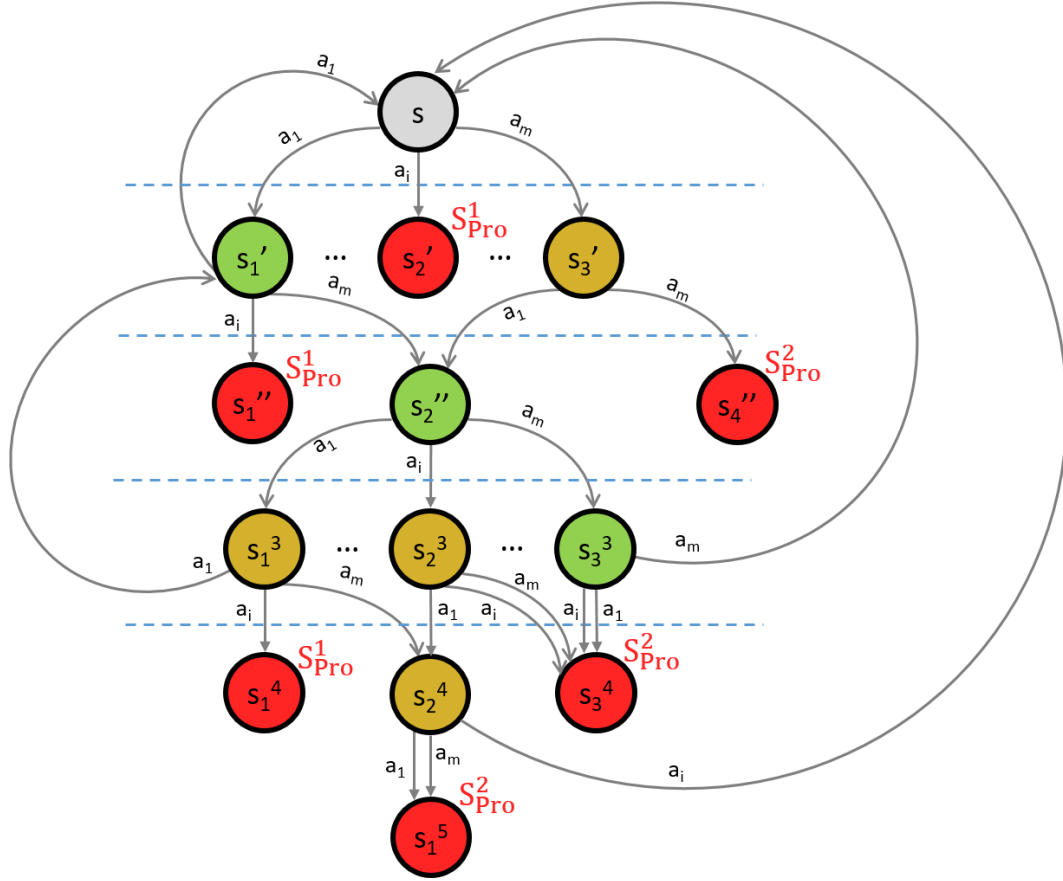


Figure 28 : Exemple d'illustration complet (l'état s_4^4 est similaire à l'état s_2^3 et n'est donc pas représenté)

4. Étape 3 : Exploitation (en ligne)

Cette étape correspond à la mise en ligne des filtres de commande et de comptes rendus sur le système industriel. Les modèles du filtre de commande (filtre de commande) et de la partie opérative (filtre de CR) servent alors d'appui pour les mécanismes de détection développés dans ces travaux. Le chapitre 5 met en place les indicateurs relatifs à la sécurité de l'ICS vis-à-vis du risque de cyberattaques tandis que le chapitre 6 détaille les mécanismes de détection qui se basent sur les informations des indicateurs et caractérise cette étape d'exploitation.

5. Bilan du chapitre

Dans ce chapitre, les aspects modélisation de l'approche et méthodologie de conception du filtre de commande ont été présentés. En effet, la stratégie de détection mise en place dans l'approche S.A.F.E. se base sur une connaissance approfondie du système de contrôle-commande à sécuriser.

Dans un premier temps, la méthodologie de conception des filtres supportant ces mécanismes a été explicitée. Elle s'appuie notamment sur l'analyse de risques qui définit les zones critiques du système et induit le périmètre devant être protégé. Dans un premier temps, les paramètres à utiliser pour modéliser les états de l'ICS sont identifiés et correspondent aux entrées/sorties de

l'API. Ce choix du type et du nombre de paramètres est crucial pour assurer une modélisation du système correcte et des mécanismes de détection efficaces. Les spécifications temporelles sont également détaillées afin d'alimenter certaines stratégies de détection.

La seconde étape consiste à modéliser la loi de commande de la partie commande et le système physique. Un algorithme d'exploration des états du système calcule ensuite toutes les évolutions possibles. Une convergence rapide est assurée par, d'une part, le placement de l'approche dans l'architecture ICS (réduction de la complexité) et, d'autre part, les conditions d'arrêts de l'exploration. Le modèle obtenu à l'issue de cet algorithme est celui du filtre de commande. Il est implanté dans le filtre de commande tandis que le modèle de la PO est mis dans le filtre de CR.

Enfin, la troisième étape correspond à la mise en ligne des filtres sur le système industriel et à l'exploitation des modèles construits par des indicateurs relatifs à la sécurité de l'ICS (Chapitre 5) et aux mécanismes de détection correspondants (Chapitre 6).

Cette partie a permis de mettre en place la stratégie de détection de l'approche S.A.F.E. ainsi que la phase de modélisation nécessaire pour assurer la pertinence des mécanismes développés dans le chapitre 6 de ce manuscrit. L'exposé d'une méthodologie de conception des filtres a permis de comprendre les points d'entrée et les apports de chacune des étapes de la méthode.

Chapitre 5

Indicateurs de la sécurité d'un ICS

Ce chapitre est consacré à l'enrichissement des modèles à des fins de détection d'attaques sur la commande. Ainsi, les modèles, créés à l'aide de cette méthodologie de conception, sont mis à disposition de la stratégie de détection. Dans un premier temps, le concept de distance issu des travaux de (A. Carcano et al. 2011) est énoncé. Il permet de quantifier un écart entre l'état courant du système et une zone définie comme critique. Cette notion de distance s'applique principalement sur des variables continues, c'est pourquoi une adaptation a été proposée au cours de ces travaux de recherche pour permettre son usage dans le domaine des systèmes à événements discrets (SED). A partir de cet indicateur, d'autres seront proposés pour améliorer encore les performances de la fonction détection.

Le premier indicateur mis en place est la distance qui mesure l'écart entre l'état courant et l'état interdit le plus proche. Il révèle la notion de risque dans lequel le système se trouve. Dans la même veine, l'indicateur de trajectoire, qui fait référence à l'évolution de la distance, permet de mettre en évidence l'intention de nuire. Les spécifications temporelles sont également mises à profit pour détecter des erreurs liées à des attaques temporelles et définissent un indicateur de temporalité. Enfin, l'indicateur de surveillance des équipements permet d'identifier des sur-sollicitations sur les actionneurs, caractéristiques d'attaques cherchant à interrompre le flux de production.

Table des matières

1.	Concept de distance au sens de Carcano.....	100
2.	Adaptation pour les SED : calcul du plus court chemin aux états critiques	102
3.	Indicateur de distance : révélateur d'une présomption de risque.....	105
4.	Indicateur de trajectoire : révélateur de l'intention de nuire.....	106
5.	Indicateurs de temporalité.....	109
6.	Indicateur de sur-sollicitation des équipements	110
7.	Bilan du chapitre.....	112

1. Concept de distance au sens de Carcano

La stratégie de détection suivie par ces travaux vise à détecter des attaques sur la commande ciblant l'intégrité de l'ICS, son environnement, la qualité du produit et des équipements de production. A cette étape du manuscrit, la détection comportementale qui peut être réalisée sur la base du filtre de commande permet de détecter des attaques dites brutales comme un ordre provoquant le passage d'un état optimal à un état interdit. Cependant, dans le cycle de vie très perturbé d'un ICS (i.e. modes dégradés), les trajectoires suivies ne sont pas toujours optimales, et ceci ne témoigne pas forcément d'une attaque. Pour compléter les mécanismes de détection qui font l'objet du chapitre 6 de cette partie, nous nous proposons ici d'enrichir notre modèle de filtre de commande par des indicateurs caractérisant le bon fonctionnement ou, *a contrario*, la dégradation d'un des éléments cités ci-dessus. La notion de distance est la clé de voûte des mécanismes de détection présentés dans la suite de ce chapitre et est au centre de la stratégie de détection.

(A. Carcano et al. 2011), et par la suite (I. Nai Fovino et al. 2012), ont été les premiers à développer ce concept de distance construit sur une modélisation sous forme d'états de l'ICS. Le principe de fonctionnement est basé sur la mesure de l'écart entre les états courants, définis par les valeurs prises par les différents composants de l'ICS et les états critiques. Une alerte est levée par l'IDS lorsqu'un état se rapproche trop d'une zone critique pour le système. Nous remarquerons que l'IDS est localisé sur le réseau industriel entre l'API (RTU, *Remote Terminal Unit*) et la supervision, globale ou locale (MTU, *Master Terminal Unit*). Dans le détail, l'approche de Carcano nécessite le suivi et l'analyse de l'évolution du système et donc des éléments suivants : (i) une représentation formelle du système, (ii) une identification des états considérés comme critiques, (iii) un *monitor* suivant l'évolution du système, (iv) une détection lorsque le système entre dans une zone définie comme critique à l'aide notamment de métriques de distance pour vérifier la proximité entre l'état courant et cette zone.

La description du système considéré (i) dans l'approche de (A. Carcano et al. 2011), est réalisée à l'aide du langage ISML (« Industrial State Modeling Language »). Il s'agit d'un langage à base de règles, spécifié pour le protocole Modbus, qui permet de modéliser le comportement du système sous la forme condition ➔ action. Un état est défini par les valeurs pouvant être prises par les différents éléments du système. Il est ainsi spécifié par l'élément considéré (PLC, IHM, ...), son adresse sur le réseau et le registre dans lequel se trouve la variable considérée. Un état critique (ii) est défini comme une classe spéciale d'état avec un certain niveau de risque, entre 1 (peu élevé) et 5 (important). Lorsqu'un état ne respecte plus les conditions de sécurité fixées, une alerte est envoyée par l'IDS avec le niveau de risque correspondant. Le « monitor », élément (iii), se situe au niveau logiciel et utilise les trames échangées entre l'API et le MTU du système pour retranscrire l'état de la mémoire de ses composants. Pour garantir une synchronisation entre le monitor et le système, les requêtes pour connaître l'état de chaque API sont directement faites par le logiciel (émulation du MTU). Ce mode opératoire est extrêmement consommateur de mémoire pour faire l'acquisition et la sauvegarde de tous les registres et bobines des API du système. De plus, le vecteur d'état est construit sur la base de la mémoire de l'automate qui peut

être corrompue puisqu'il s'agit d'un élément vulnérable (Partie 1, Chapitre 1 – 5.). Il s'agit là d'une des limites de l'approche de Carcano. Enfin, l'IDS doit pouvoir identifier lorsque le système entre dans une zone critique en se basant sur la représentation de l'état de l'ICS (iv). Comme expliqué ci-dessus, un état « s » est défini par les valeurs prises par les composants n du système contenus dans la mémoire de l'API telles que $s \in \mathbb{R}^n$. L'IDS surveille à chaque instant t , si l'état $s(t)$ satisfait les conditions définissant l'ensemble des états critiques du système $CS \subseteq \mathbb{R}$ tel que $s(t) \in CS$. Cet ensemble CS est constitué par des formules critiques Φ déterminant les critères d'entrée dans une zone critique pour le système. La complexité pour définir le vecteur de contraintes Φ est linéaire avec le nombre de paramètres considérés.

L'originalité de cette approche se situe dans l'apport de la notion de distance. Elle permet de quantifier l'écart entre l'état courant du système et les zones définies comme critiques. Pour ce faire, la distance d est définie comme le minimum entre deux types de distances particulières et spécifiées en (14).

$$d: \mathfrak{R}^n \times \mathfrak{R}^n \rightarrow \mathfrak{R}^+, s \in \mathfrak{R}^n$$

$$d(s, \Phi) = \min_j d_{min}(s, C_j) \text{ avec } d_{min} = d_1 = \sum_1^n |s_i - c_i| \text{ ou } d_{min} = d_v = \#\{i | s_i \neq c_i\} \quad (14)$$

La distance d_1 est définie comme la *distance de Manhattan* et calcule l'écart entre l'état du système et les contraintes C_j du vecteur Φ composante par composante. En revanche, la distance d_v compte le nombre de composantes différentes entre les deux vecteurs. La distance est caractérisée par le minimum entre d_1 et d_v et permet de mesurer la proximité d'une zone définie comme critique.

De l'approche développée par (A. Carcano et al. 2011), nous retiendrons pour nos travaux la notion de distance car elle permet de donner une indication aux opérateurs sur la proximité d'une zone critique pour le système ; zone dans laquelle l'attaquant peut plus facilement provoquer un passage dans un état interdit (attaque brutale). Toutefois, l'analyse de l'approche de Carcano met en avant qu'elle s'applique uniquement aux variables continues comme la consigne de vitesse de rotation d'un moteur ou la consigne de température. En l'état, l'information issue de la distance dans des systèmes à événements discrets, avec des capteurs et actionneurs tout-ou-rien, est inutilisable. En effet, le système peut directement dévier vers un état dangereux ou interdit lorsque les actions appliquées ne respectent pas la loi de commande. Ainsi, le calcul de la distance doit être adapté pour les systèmes à événements discrets. Il se fera notamment sur les états définis pour le système. En effet, sur la base du modèle de la loi de commande et de l'identification des états optimaux, la notion de distance peut être modifiée pour permettre aux opérateurs de surveiller que l'ICS suit la « trajectoire » optimale. En conclusion, il apparaît que cette notion de distance, combinée aux modèles développés dans le Chapitre 4, fournit des informations pertinentes dans une stratégie de détection sur les couches temps-réels des ICSs. Une adaptation de ce concept de distance au SED est proposée dans le paragraphe suivant. Elle aura pour objectif d'intégrer les modèles des parties commande et opérative afin de construire un indicateur caractérisant la notion de risque associé à la proximité de l'état courant par rapport à un état critique.

2. Adaptation pour les SED : calcul du plus court chemin aux états critiques

Le paragraphe précédent a permis d'introduire le concept de distance aux états critiques d'un système. Il permet de quantifier l'écart entre l'état courant du système et un ensemble de contraintes Φ , qui définit une zone critique, afin de détecter des comportements malveillants. Toutefois, la distance au sens de Carcano, (i) n'a de sens que pour les variables continues (vitesse de rotation, consigne de température) et (ii) ne tient pas compte de la commande spécifiée pour le système. En effet, nous pouvons nous en rendre compte au travers des exemples suivants :

- (i) Soit un système de contrôle de moteur où l'API commande la vitesse de rotation. Une zone critique est définie en prenant différents paramètres critiques tels que l'échauffement maximal du moteur. La distance au sens de Carcano permet de détecter des « glissements » en cas d'attaque sur la consigne envoyée à la chaîne d'action. En revanche, si l'on considère un système SED contrôlé par un API déclenchant le fonctionnement d'un convoyeur, il apparaît que la distance n'apporte aucune information. En effet, elle indique uniquement le passage dans la zone critique.
- (ii) Soit un système de pompage-turbinage dans un barrage hydroélectrique, lorsque l'électricité est excédentaire sur un réseau, le système peut actionner des pompes pour déplacer l'eau du niveau inférieur au niveau supérieur. Ainsi, quand la demande en électricité augmentera, le stockage d'énergie pourra être utilisé. Lors de cette phase de pompage, les moteurs peuvent être amenés à fonctionner dans des zones critiques mais durant une courte période temporelle. De la même manière, un système de production peut être amené à se rapprocher d'un état critique, et donc d'une distance nulle, sans pour autant être victime d'une attaque. Dans ces deux exemples, il apparaît nécessaire de prendre en compte la commande conçue pour piloter le système.

Afin d'améliorer le concept de distance, nous utiliserons le filtre de commande proposé dans le Chapitre 4. Ainsi, la distance est définie comme le nombre minimal d'actions à appliquer à un système à partir de l'état courant avant d'arriver dans un état interdit (Sicard, Zamaï, et Flaus 2018a). Cette séquence d'ordres est appelée le plus court chemin vers un état interdit (*Shortest Path to Critical State*, SPCS). A l'aide du modèle filtre de commande, un algorithme a été développé pour calculer le SPCS à partir de n'importe quel état atteignable du système et quel que soit l'état interdit considéré. Cette étape de calcul est réalisée hors-ligne afin de s'affranchir des contraintes temps-réels et est la base de la stratégie de détection développée par la suite. Elle s'intègre à la suite de l'algorithme d'exploration des états de l'étape 2 (Chapitre 4 - 3.) de la méthodologie de conception des filtres.

L'algorithme 2 retourne le nombre d'itérations nécessaires pour aboutir à un état interdit à partir d'un état quelconque, la séquence d'ordres conduisant à cet état ainsi que la séquence d'états suivie par le système. Pour ce faire, cet algorithme se base sur l'automate fini déterministe M défini par l'équation (9). Il représente l'évolution de la partie opérative, avec les états atteignables $s \in S_n$ pouvant être pris, en fonction des ordres $a \in A_m$ au travers de la fonction

de transition δ définie par l'équation (11). L'état initial $s_0 \in S_0 \subset S_n$ est le point de départ de l'algorithme et l'ensemble des états finaux $S_{final} \subset S_n$ représente les conditions d'arrêts. Par construction, l'automate M garantit les conditions d'atteignabilité des états calculés par le système physique. En effet, dans le Chapitre 4, nous avons expliqué le principe de l'algorithme d'exploration des états (algorithme 1) qui permet d'étudier toutes les évolutions possibles de la partie opérative à partir de l'ensemble des états initiaux S_0 . La distance avec tous les états critiques S_n est calculée pour chaque état atteignable $s \in S_n$. Le plus court chemin est la distance la plus faible avec un état interdit. Comme pour décrire l'algorithme d'exploration des états du système, nous raisonnerons avec la fonction de transition sur les ensembles Δ explicités dans l'équation (12). Ainsi, la distance D est définie dans l'équation (15) comme le nombre minimum d'itérations n à effectuer à partir d'un état $s \in S^i \subset S_n$ pour atteindre un état interdit $s_{Pro}^j \in S_{Pro}$ tel que :

$$D(s|S_{Pro}) = \min_n \Delta^n S^i, a \in S_{Pro} \forall s \in S^i \subset S_n, \forall a \in A_m \quad (15)$$

L'algorithme 2 fournit des informations complémentaires sur le calcul de la distance aux états critiques. La Figure 29 illustre le calcul de distance qui est réalisé par l'algorithme 2. A l'issue de ce dernier, pour chaque état atteignable $s \in S_n$ du système, le plus court chemin aux états interdits est connu ce qui implique le calcul de la distance $D(s|S_{Pro})$ ainsi que les séquences d'ordres et d'états conduisant à ces états critiques.

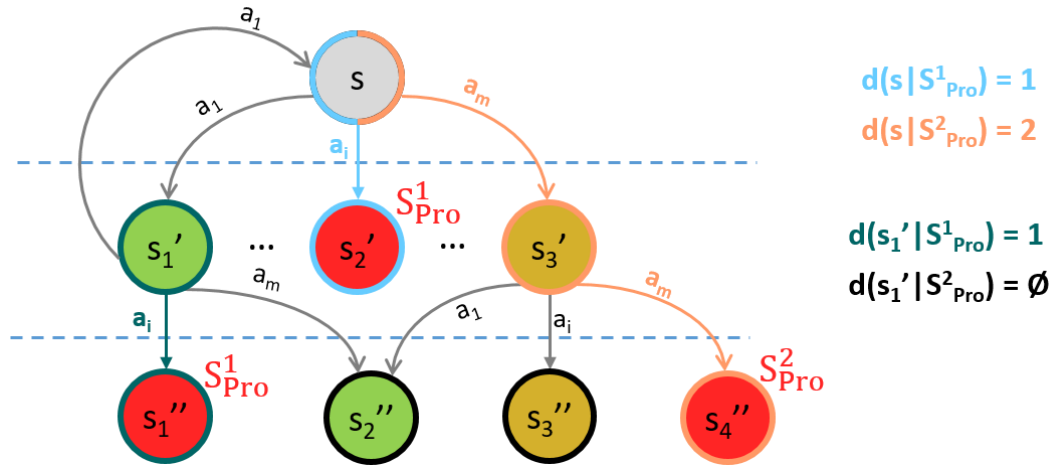


Figure 29 : Illustration du calcul de la distance (contour) entre l'état courant et les états critiques pour un système à événements discrets (algorithme 2)

Function Find nearest Prohibited State

Iteration = 0; $S_i = S$

WHILE (Stop $\neq$ 0)

 Iteration = Iteration + 1

$S' = \Delta(S_i, A)$

 IF ($S' \cap S_{Prev} \neq \{\emptyset\}$) THEN

$S' = S' \setminus P$

$S_{Prev} = S_{Prev} \cap S'$

 ELSE

$S_{Prev} = S_{Prev} \cap S'$

 ENDIF

```

    IF (S' ∩ SPro ≠ {∅}) THEN
        FOR (j = 1 TO SIZEOF(SPro))
            IF (si ∈ S' = sProj ∈ SPro)
                D(s|sProj) = iteration
                SPro = SPro \ sProj
            ENDIF
            IF (SPro = {∅})
                Stop = 0
            ENDIF
        ENDFOR
        S' = S' \ F
    ENDIF
    Si = S'
    DONE
    End
    
```

Algorithme 2. *Algorithme pour trouver l'état interdit le plus proche de l'état courant $s \in S^i \subset S_n$ (plus court chemin)*

Ainsi, comme l'illustre la Figure 29, l'algorithme 2 utilise la modélisation de la partie opérative pour calculer les différentes évolutions possibles à partir d'un état $s \in S_n$ pour aboutir à un état interdit. Le nombre minimum d'actions à appliquer pour chacun des états critiques du système définit la distance. Celle qui est la plus faible est appelée plus court chemin vers un état interdit, noté $D(s|S_{Pro})$ et est définie en (16), elle indique la présomption du risque le plus proche.

$$D(s|S_{Pro}) = \min d(s|S_{Pro}^i) \text{ avec } S_{Pro}^i \subset S_{Pro} \text{ et } \forall i \in \mathbb{N}, 1 \leq i \leq \text{Card}(S_{Pro}) \quad (16)$$

Sur l'exemple illustré ci-dessus, deux états interdits ont été identifiés ($S_{Pro}^1 \subset S_{Pro}$ et $S_{Pro}^2 \subset S_{Pro}$). Afin d'évaluer la distance de l'état $s \in S_n$ aux états interdits, les différentes actions possibles sont appliquées à cet état jusqu'à atteindre chaque état interdit. Finalement, nous obtenons les résultats suivants :

- $D(s|S_{Pro}^1) = 1$ puisque lorsque le système est dans l'état « s » alors l'action a_m conduit à l'état interdit $s'_2 \in S_{Pro}^1$ tel que : $s'_2 = \delta(s, a_m) \in S_{Pro}^1$. Le chemin critique pour atteindre l'état interdit $S_{Pro}^1 \subset S_{Pro}$ est donc constitué de $\{a_m\}$ pour les ordres et $\{s'_2\}$ pour les états.
- $D(s|S_{Pro}^2) = 2$ puisque lorsque le système est dans l'état « s », au minimum deux actions consécutives a_m et a_m doivent être appliquées pour atteindre l'état interdit $s''_4 \in S_{Pro}^2$ tel que : $s''_4 = \delta(\delta(s, a_m), a_m) \in S_{Pro}^2$. Le chemin critique pour atteindre l'état interdit $S_{Pro}^2 \subset S_{Pro}$ est constitué de $\{a_m, a_m\}$ pour les ordres et $\{s'_3, s''_4\}$ pour les états.

Nous remarquons que l'état s''_1 , appartenant au sous-ensemble des états interdits $S_{Pro}^1 \subset S_{Pro}$, est atteint après la séquence d'actions $\{a_1, a_i\}$. Il n'est pas pris en compte puisqu'un chemin plus court pour atteindre l'état interdit S_{Pro}^1 a déjà été trouvé. Enfin, sur l'exemple illustré en Figure 29, le plus court chemin vers un état critique à partir de l'état s pour le système est $D(s|S_{Pro}^1) = 1$ puisqu'il s'agit de l'état interdit le plus proche. L'ensemble des résultats de l'algorithme 2 sur l'exemple illustré en Figure 29 est disponible dans le Tableau 5.

Tableau 5 : Résultats de l'algorithme 2 sur l'exemple développé sur la Figure 29

États de départ	Distance aux états interdits	Séquence d'actions	Séquence d'états
s	$D\ s S_{Pro}^1 = 1$ $D\ s S_{Pro}^2 = 2$	$\{a_m\}$ $\{a_m, a_m\}$	$\{s'_2\}$ $\{s'_3, s'_4\}$
s'_1	$D\ s'_1 S_{Pro}^1 = 1$ $D\ s'_1 S_{Pro}^2 = 3$	$\{a_i\}$ $\{a_m, a_m, a_1\}$ ou $\{a_1, a_m, a_m\}$	$\{s''_1\}$ $\{s''_2, s'_3, s'_4\}$ ou $\{s, s'_3, s'_4\}$
s'_3	$D\ s'_3 S_{Pro}^1 = 3$ $D\ s'_3 S_{Pro}^2 = 1$	$\{a_1, a_1, a_i\}$ $\{a_m\}$	$\{s''_2, s'_3, s'_4\}$ $\{s'_4\}$
s''_2	$D\ s''_2 S_{Pro}^1 = 2$ $D\ s''_2 S_{Pro}^2 = 2$	$\{a_1, a_i\}$ $\{a_m, a_1\}$ ou $\{a_m, a_i\}$	$\{s'_3, s'_4\}$ $\{s'_3, s'_4\}$
s^3_1	$D\ s^3_1 S_{Pro}^1 = 1$ $D\ s^3_1 S_{Pro}^2 = 2$	$\{a_i\}$ $\{a_m, a_m\}$ ou $\{a_m, a_1\}$	$\{s^4_1\}$ $\{s^4_2, s^5_1\}$
s^3_2	$D\ s^3_2 S_{Pro}^1 = 3$ $D\ s^3_2 S_{Pro}^2 = 1$	$\{a_1, a_i, a_i\}$ $\{a_i\}$ ou $\{a_m\}$	$\{s^4_2, s, s'_2\}$ $\{s^4_3\}$
s^3_3	$D\ s^3_3 S_{Pro}^1 = 2$ $D\ s^3_3 S_{Pro}^2 = 1$	$\{a_m, a_i\}$ $\{a_1\}$ ou $\{a_i\}$	$\{s, s'_2\}$ $\{s^4_3\}$
s^4_2	$D\ s^4_2 S_{Pro}^1 = 2$ $D\ s^4_2 S_{Pro}^2 = 1$	$\{a_i, a_i\}$ $\{a_1\}$ ou $\{a_m\}$	$\{s, s'_2\}$ $\{s^5_1\}$

3. Indicateur de distance : révélateur d'une présomption de risque

Telle que définie dans l'approche proposée, la distance quantifie l'écart entre l'état courant et chaque type d'états interdits les plus proches. Ainsi, par construction avec l'algorithme 2, il est possible de connaître pour chaque état du système, la présomption de risque associée à ce dernier. Tout d'abord, cet indicateur renseigne l'opérateur de l'entrée du système dans une zone critique (proximité d'un ou de plusieurs états interdits) indépendamment du fait que la loi de commande soit respectée ou non. De plus, la distance identifie uniquement le plus court chemin vers un état critique sans prendre en compte la probabilité d'occurrence de ce dernier. Ainsi, cet indicateur doit être pondéré par rapport au nombre d'actions $a_i \in A_{S_{Pro}} \subseteq A_m$ menant effectivement à l'état interdit considéré, comme indiqué dans l'équation (17) ci-après :

$$\forall a_i \in A_{S_{Pro}} \subseteq A_m, \forall a_j \in A_m, \exists R \in [0, 1] \text{ tel que } R = \frac{\sum_i a_i}{\sum_j a_j} \quad (17)$$

Un niveau d'occurrence R , associé à la distance d'un état critique, est calculé en fonction du rapport entre le nombre d'actions menant effectivement vers un état interdit $\sum_i a_i \in A_{S_{Pro}} \subseteq A_m$ et le nombre d'actions possibles sur le système $\sum_j a_j \in A_m$. Comme dans une partie d'échec,

il indique aux opérateurs les actions permettant de s'éloigner d'un état critique ou, au contraire, celles qui le rapprochent. Ainsi, le calcul de R permet de compléter l'indicateur de distance D qui est révélateur de la présomption de risque associée à l'état dans lequel se trouve le système.

Le concept de distance, depuis son introduction dans cette approche à partir de la notion de Carcano, a été construit autour des états interdits ce qui a conduit à l'élaboration de l'algorithme 2. Toutefois, dans le but d'identifier le risque associé à l'état actuel du système, les modèles des parties commande et opérative peuvent être utilisés. Ainsi, en complément de la distance aux états critiques $D(s|S_{Pro})$, d'autres distances peuvent être calculées (Sicard, Zamaï, et Flaus 2017b, 2018c, 2018a) comme :

- $D(s|S_{Opt})$ qui mesure l'écart entre l'état $s \in S_n$ actuel du système et l'état optimal $s_{Opt} \in S_{Opt} \subset S_n$ dans lequel il devrait être si la loi de commande était respectée. Ces états optimaux sont connus de par l'analyse de risques réalisée préalablement à la méthodologie de conception des filtres.
- $D(a|A_{Opt})$ qui quantifie l'écart entre l'action $a_i \in A_m$ analysée par le filtre et l'ordre attendu $a_{Opt} \in A_{Opt}$ (obtenu à partir du modèle de filtre).

Lorsque le système suit la loi de commande alors les distances $D(s|S_{Opt})$ et $D(a|A_{Opt})$ sont nulles. En revanche, si le système dévie de la séquence optimale, respectant les contraintes de la commande, alors la distance augmentera d'une unité jusqu'à retrouver respectivement l'état ou l'action optimale. Nous précisons que seules les distances $D(s|S_{Opt})$ et $D(s|S_{Pro})$ sont implémentées dans le filtre de comptes rendus repris de (Lhoste 1991; Cruette, Bourey, et Gentina 1991). Elles permettent de vérifier si l'état atteint par le système correspond à l'état attendu et l'évolution par rapport aux états critiques. En revanche, le filtre de commande, qui analyse les ordres émanant de la partie commande, calcule en plus la distance $D(a|A_{Opt})$ afin de détecter une dérive sur les ordres envoyés.

En conclusion, le concept de distance permet de situer précisément la présomption de risque associée à un état en quantifiant l'écart avec un état critique $D(s|S_{Pro})$ mais également vis-à-vis de l'état optimal attendu $D(s|S_{Opt})$. L'utilisation du modèle de la partie commande au sein du filtre de commande permet également d'analyser l'écart entre l'ordre reçu et celui attendu avec $D(a|A_{Opt})$. Enfin, le calcul du risque R caractérise la facilité pour un attaquant de provoquer le passage du système dans un état interdit.

4. Indicateur de trajectoire : révélateur de l'intention de nuire

Introduit dans le paragraphe précédent, le concept de distance donne des indications sur le risque associé à l'état actuel. Cependant, comme nous l'avons évoqué précédemment, R peut être grand sans témoigner d'une attaque. Pour aller plus loin, nous proposons d'étudier l'évolution de R au cours du temps afin de pouvoir évaluer s'il y a ou non intention de nuire, et répondre ainsi à la problématique des attaques « par séquence » qui dégradent le système sur plusieurs

enchaînements états/actions. Le concept de trajectoire est défini comme l'évolution de la distance sur une séquence qui peut être basée sur les états ou sur le temps. Dans le premier cas, la distance est calculée pour chaque état stable atteint par le système et pour chaque émission d'ordre de la part de l'API. Dans l'autre cas, l'évolution de la distance est calculée et surveillée en fonction d'une séquence temporelle. En fonction des situations, les mécanismes de détection auxquels les filtres font appel ne sont pas les mêmes. Lorsque la séquence d'états est considérée, les filtres utilisent la stratégie de détection basée sur les contraintes combinatoires (Chapitre 6 - 2. Détection d'anomalies par contraintes combinatoires). De la même manière, les mécanismes de détection basés sur les contraintes temporelles sont mis en œuvre lorsque les filtres surveillent le système sur des séquences temporelles (Chapitre 6 - 3. Détection d'anomalies par contraintes temporelles). L'équation (18) représente ce concept de trajectoire d'un point de vue mathématique :

$$\forall s \in S_n, \forall t, \begin{cases} T s = \frac{\partial d(s)}{\partial s} \\ T t = \frac{\partial d(t)}{\partial t} \end{cases} \quad (18)$$

La trajectoire permet de représenter l'évolution du système sur une séquence à la différence de la distance qui ne la décrit qu'à un seul instant. Ainsi, la trajectoire permet la détection d'attaques par séquence et est un indicateur de l'intentionnalité d'une anomalie. En effet, certains schémas d'attaques demandent à être étudiés sur de nombreuses itérations pour être détectés, tout comme la vitesse d'évolution et la répétition des trajectoires, comme dans le cas de Stuxnet où la vitesse était modifiée aléatoirement avant de revenir à la normale pour une durée déterminée. La vision offerte par cette notion de trajectoire permet de prendre en compte l'évolution du système à différentes échelles qui sont mises en œuvre dans le chapitre 7.

Afin de compléter l'explication de ce concept, une comparaison avec la vitesse d'une voiture peut être faite. Lorsque la vitesse du véhicule est importante (en respectant la limitation imposée ou non) mais qu'il se situe sur une ligne droite alors le danger peut être considéré comme faible. En revanche, si nous observons que la vitesse augmente à l'amorce d'un virage, un comportement suspect peut être identifié. De plus, si ce dernier se répète dans le temps alors un comportement malveillant ou suicidaire peut être mis en évidence ; c'est en cela que le concept de trajectoire est révélateur de l'intentionnalité d'un comportement.

Sur la base de l'équation (18) et de la même manière que pour la distance, il est possible de calculer différents types de trajectoires en fonction de l'élément considéré :

- $T_{order}(a|A_{Opt})$ vérifie l'évolution des ordres envoyés aux actionneurs afin qu'ils correspondent au modèle de commande (ordres optimaux). Cet indicateur se base sur la distance $D(a|A_{Opt})$ et est défini dans (19).

$$\forall a \in A_m, \forall t, \begin{cases} T_{order}(a|A_{Opt}) = \frac{\partial d(a|A_{Opt})}{\partial A} \\ T_{order} t = \frac{\partial d(a|A_{Opt})}{\partial t} \end{cases} \quad (19)$$

- $T_{etat}(s|S_{Opt})$ surveille l'évolution de la distance $D(s|S_{Opt})$ de l'état courant avec les états optimaux en fonction d'une séquence d'états ou de temps telle que définie dans (20). La trajectoire optimale du système est obtenue lorsque tous les états atteints

par le système sont optimaux indiquant le respect des contraintes imposées par la loi de commande.

$$\forall s \in S_n, \forall t, \begin{cases} T_{\text{état}}(s|S_{\text{Opt}}) = \frac{\partial d(s|S_{\text{Opt}})}{\partial S} \\ T_{\text{état}} t = \frac{\partial d(s|S_{\text{Opt}})}{\partial t} \end{cases} \quad (20)$$

- $T_{\text{état}}(s|S_{\text{Pro}})$, défini dans l'équation (21), représente l'évolution de la distance $D s|S_{\text{Pro}}$ par rapport aux états interdits du système. Comme pour le concept de distance, cette trajectoire est calculée pour chaque état critique, celle qui est la plus proche d'un état interdit est appelée trajectoire critique :

$$\forall s \in S_n, \forall t, \begin{cases} T_{\text{état}} s|S_{\text{Pro}} = \frac{\partial d(s|S_{\text{Pro}})}{\partial S} \\ T_{\text{état}} t = \frac{\partial d(s|S_{\text{Pro}})}{\partial t} \end{cases} \quad (21)$$

La Figure 30 représente les trajectoires calculées par le filtre de commande sur le système illustré en Figure 28. Comme le chemin respectant la loi de commande et passant par les états optimaux a été choisi, les trajectoires du filtre de comptes rendus ne seront pas tracées. En effet, ces dernières sont identiques à celles ($T_{\text{état}}(s|S_{\text{Opt}})$ et $T_{\text{état}}(s|S_{\text{Pro}})$) représentées sur la Figure 30 dans ce cas précis.

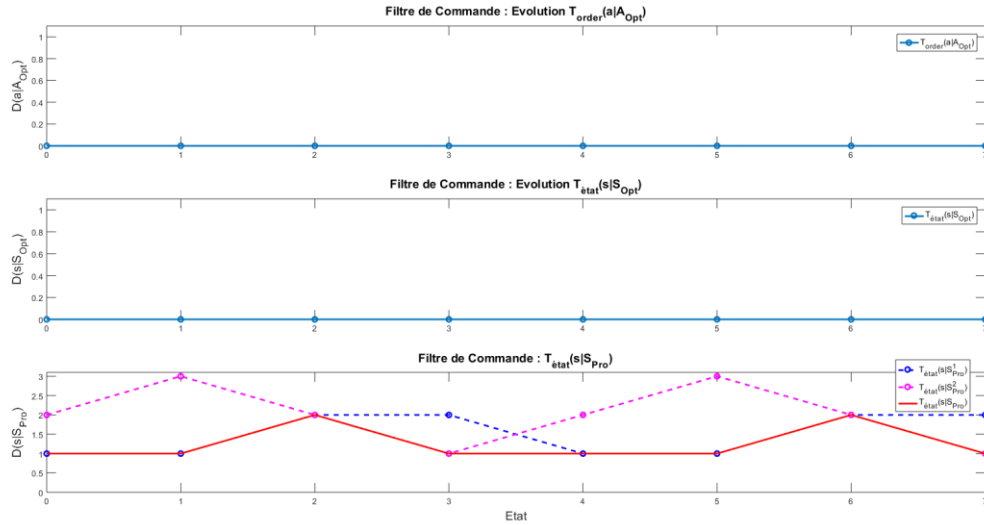


Figure 30 : Illustration de l'indicateur de trajectoire sur le système représenté en Figure 28 (ici, représentation de la trajectoire optimale)

Les trajectoires représentant l'évolution de l'écart entre l'état courant et l'état optimal $T_{\text{order}}(a|A_{\text{Opt}})$ (en haut sur Figure 30) et entre l'ordre à analyser et l'ordre attendu $T_{\text{état}}(s|S_{\text{Opt}})$ (au milieu sur Figure 30) sont nulles sur la séquence considérée sur la Figure 30. En effet, nous considérons que la loi de commande est appliquée au système sans l'apparition de défaillances. $T_{\text{état}}(s|S_{\text{Pro}})$ est représentée en bas de la Figure 30. Les courbes en pointillés illustrent l'évolution de la distance par rapport à chacun des états interdits du système (S_{Pro}^1 et S_{Pro}^2). La courbe pleine est la trajectoire critique reprenant le plus court chemin vers un état critique et dont la distance est la plus faible $D s|S_{\text{Pro}} = \min(d s|S_{\text{Pro}}^1, d s|S_{\text{Pro}}^2)$. Ainsi, en s'appuyant sur le concept de distance, qui est définie comme le nombre minimum d'actions à appliquer à l'état

courant pour atteindre un état interdit, la trajectoire $T_{état}(s|S_{Pro})$ représente toujours le pire scénario possible pour le système.

La Figure 30 propose l'exécution de la loi de commande sur le système illustré en Figure 28. Des scénarii d'attaques faisant intervenir les mécanismes de détection développés à l'aide des contraintes combinatoires sont détaillés dans le chapitre suivant.

5. Indicateurs de temporalité

Dans le chapitre 4 traitant de la méthodologie de conception des filtres, une analyse temporelle du système a été menée aboutissant à la construction de fenêtres temporelles définies à l'aide des équations (2) pour les FT_{Ordre} et (3) pour les FT_{CR} . Ces dernières veillent au respect des contraintes temporelles inhérentes à l'ICS protégé (Sicard, Zamaï, et Flaus 2018b).

Afin de vérifier si les échanges entre les parties commande et opérative sont compris dans les gabarits définis par les fenêtres temporelles, plusieurs indicateurs sont mis en place. Le premier t_{FT} est détaillé dans l'équation (22) et détermine si l'événement observé $e \in (S_n \cup A_m)$ (action ou compte rendu) apparaît dans la fenêtre temporelle respectant les contraintes de temps du système.

$$\forall e \in S_n \cup A_m, t_{FT} = \begin{cases} 1 & \text{si } e \in FT \\ 0 & \text{sinon} \end{cases} \quad (22)$$

Ainsi, l'indicateur t_{FT} garantit que l'action ou le compte rendu soit respectivement envoyé/reçu après la date au plus tôt et avant la date au plus tard. La vérification de cet indicateur tient également compte du retard τ pouvant être déterminé entre deux événements ce qui permet de respecter les contraintes temporelles spécifiées pour le système. En effet, si un événement apparaît en dehors de sa fenêtre temporelle alors les filtres le détecteront puisqu'il n'appartiendra pas au gabarit prédéfini. A l'inverse, si un événement ne se produit pas lors de sa fenêtre temporelle alors une alerte est envoyée par les filtres. Bien entendu, ces gabarits sont déterminés à l'aide de la date d'apparition de l'événement, et peuvent déclencher une alerte s'ils ne sont pas respectés.

Le deuxième indicateur Δt^{moy} surveille l'évolution de la durée moyenne d'activation des actions t_A^{moy} ou des comptes rendus t_E^{moy} . Il est défini dans l'équation (23) ci-dessous :

$$\Delta t^{moy} = \begin{cases} \sum_k t_{a_k}^{moy} \\ \sum_l t_{e_l}^{moy} \end{cases} \quad (23)$$

De la même manière que la trajectoire surveille l'évolution de la distance au cours d'une séquence pour détecter des comportements anormaux sur le long terme, l'indicateur Δt^{moy} surveille les durées d'exécutions moyennes dans le même souci. Ainsi, les filtres sont en mesure de détecter des signes avant-coureurs comme le décalage anormal de la durée d'activation vers un extremum ou encore une rupture dans l'évolution de Δt^{moy} (par exemple, une durée moyenne qui se raccourcit ou qui varie fortement). Cet aspect est détaillé dans le chapitre suivant mais deux éléments doivent être pris en considération : les spécifications des contraintes temporelles sont fortement dépendantes du système considéré (usure, vieillissement, ...) et les ICSs sont des

environnements très variables comme expliqué dans la Partie 1. En effet, les indicateurs construits ne se basent pas uniquement sur une hypothèse de forte périodicité, comme dans les travaux de (Zimmer et al. 2010) ou (Kleinmann et Wool 2017). Cette dernière peut toutefois affecter les performances des informations fournies par Δt^{moy} . En considérant un système dont le comportement est similaire d'une période à une autre, une variation soudaine de Δt^{moy} suggère une anomalie dans l'ICS. A l'inverse, un système dont la dynamique est plus aléatoire peut avoir de fortes variations de cet indicateur sans relever un comportement suspect.

D'autres indicateurs auraient pu être choisis pour alimenter les mécanismes de détections liés aux contraintes temporelles, comme la surveillance de l'évolution des extremums. Toutefois, la complexité et la variabilité dans un système de contrôle-commande compliquent la mise en place de tels indicateurs sur les contraintes temporelles. En effet, l'attaquant pourrait alors utiliser à son avantage le déclenchement d'alertes intempestives. Ainsi, l'évolution de la durée d'activation Δt^{moy} permet d'une part de suivre les variations dans le temps de cette variable et, d'autre part, de la resituer par rapport à des extremums définis lors de l'analyse du système. Il s'agit là du meilleur compromis détection par rapport à la menace potentielle.

6. Indicateur de sur-sollicitation des équipements

Les indicateurs présentés précédemment se basent sur la connaissance des modèles comportementaux des ICSs pour détecter des événements ou des séquences d'ordres suspects. Dans ce paragraphe, nous traitons d'un indicateur qui se focalise sur la sécurité des actionneurs vis-à-vis de la sur-sollicitation. Cet indicateur est construit pour assurer le troisième mode de protection présenté dans la Partie 1 - Chapitre 3 – 3.3. : la protection des équipements de production.

En effet, les filtres surveillent et analysent les échanges entre la partie commande et la partie opérative du système. Toutefois, un certain type d'attaques peut viser l'intégrité des équipements de production afin qu'ils n'assurent plus leurs services sur le flux de production. Il se caractérise par des sur-sollicitations de la chaîne d'actions entraînant des problématiques de maintenance et, à terme, des dégâts sur les équipements eux-mêmes. Ce type d'attaque se manifeste dans la plupart des cas sur le long terme, comme dans Stuxnet où les moteurs des centrifugeuses ont cessé de fonctionner plusieurs mois après le début des séquences ayant pour objectif d'user le matériel utilisé.

Pour détecter ce type de malveillance que nous noterons « O », les filtres surveillent l'occurrence des ordres par rapport à l'évolution de la partie opérative comme définie dans l'équation (24)

$$\forall a \in A_m, \forall s \in S_n, O = \frac{\sum \Delta a^2}{\sum \Delta s^2} \quad (24)$$

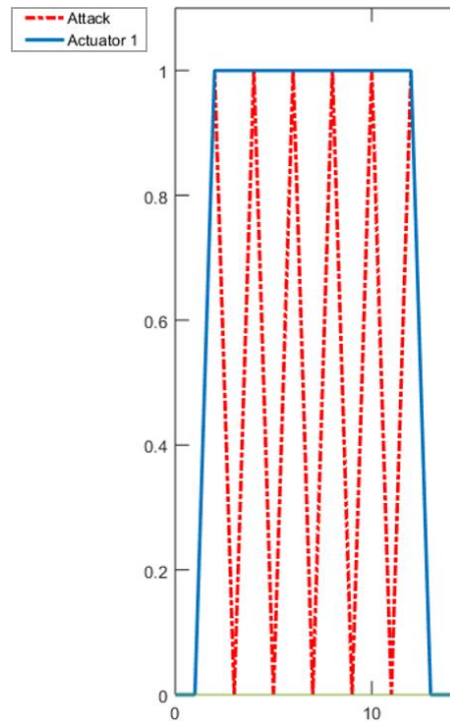


Figure 31 : Illustration d'une sur-sollicitation sur un actionneur

Pour illustrer cet indicateur, un exemple est détaillé dans (Sicard, Zamaï, et Flaus 2018d) sur la base de cuves gérées par des vannes tout-ou-rien. D'un point de vue combinatoire, l'attaquant respecte les contraintes de la loi de commande en termes de séquentialité, parallélisme, synchronisation et exclusion mutuelle, puisque la « bonne » action est bien appliquée sur la partie opérative. En revanche, la contrainte sur l'action en cours est incorrecte, l'ordre étant corrompu afin de dégrader la partie opérative. La Figure 31 illustre l'une de ces attaques sur un exemple de vannes. L'actionneur 1 (courbe pleine) déclenche l'ouverture d'une vanne de remplissage sur le système de cuves. Dans les spécifications de la loi de commande, l'action doit être maintenue jusqu'au déclenchement d'un capteur de niveau. L'attaque (pointillés rouges) consiste à ouvrir/fermer la vanne afin de provoquer une usure prématurée de l'équipement. En revanche, les spécifications de la loi de commande sont respectées puisque le système atteint au final le capteur de niveau. L'indicateur O indique dans ce cas une anomalie par rapport au comportement normal puisque l'actionneur 1 est sur-sollicité pour réaliser l'action.

Nous noterons qu'une telle attaque ne respecte pas non plus les contraintes temporelles du système, puisque le système ne maintient pas l'action suffisamment longtemps dans l'exemple développé plus haut. Les indicateurs temporels sont également mis en œuvre lors d'une attaque de sur-sollicitation des équipements.

7. Bilan du chapitre

Ce chapitre a permis la présentation des indicateurs qui alimentent les mécanismes de détections développés dans le chapitre suivant. Chacun d'entre eux rend possible l'identification d'une menace spécifiée dans la Partie 1 de ce manuscrit.

Le concept de distance donne une information sur le risque associé à l'état courant dans lequel se trouve le système. Il est issu de la notion de distance au sens de (A. Carcano et al. 2011). Cependant, comme elle n'était pas adaptée aux systèmes à événements directs, une adaptation a été proposée dans ce chapitre. La distance représente le nombre minimum d'actions à appliquer à l'état courant pour atteindre un état interdit. L'algorithme permettant le calcul de cet écart ainsi que la probabilité d'occurrence du chemin critique ont également été présentés. De la distance naît le concept de trajectoire qui indique l'intention de nuire d'une séquence sur le système protégé. La trajectoire représente ainsi l'évolution de la distance sur une séquence pouvant être événementielle (sur plusieurs changements d'états) ou temporelle. L'étude de l'évolution du risque non plus sur un état donné mais sur une séquence permet ainsi de mettre en évidence des malveillances sur le long terme appelées attaques par séquences.

Deux indicateurs sont garants des contraintes temporelles spécifiées lors de l'analyse temporelle du système réalisée lors de la première étape de la méthodologie de conception des filtres. t_{TT} vérifie l'apparition d'un événement (action ou compte rendu) durant la fenêtre temporelle qui lui est associée. Il active une détection lorsque le gabarit n'est pas respecté ou en cas de non-apparition. Δt^{moy} permet de suivre l'évolution de la durée d'activation de chacun des événements spécifiés dans les modèles. L'étude intrinsèque de cet indicateur et la comparaison avec les extremums, définis lors de l'analyse temporelle, permettent de mettre en avant des malveillances.

Enfin, un indicateur a été établi pour détecter des sur-sollicitations d'équipements. Ce type de malveillance ayant pour objectif de provoquer une usure prématurée ou la destruction du matériel, l'indicateur se focalise sur l'étude de l'évolution de l'occurrence des ordres de commande par rapport à l'état de la partie opérative.

Le chapitre suivant se concentre sur la mise en œuvre de ces indicateurs dans les mécanismes de détection.

Chapitre 6

Protection des ICSs : présentation des mécanismes de détection d'anomalies

Ce chapitre est dédié à la présentation des mécanismes de détection mis en œuvre dans les filtres. Il s'agit du dernier maillon de la stratégie de détection de l'approche S.A.F.E. et la dernière étape de la démarche. Dans un premier temps, la détection de couples état/action est présentée. Elle permet d'identifier des menaces immédiates appelées contextes. Ces derniers sont caractérisés par le fait que l'application de l'ordre dans l'état courant du système mène le système vers un état interdit. Il s'agit d'un mécanisme de protection conduisant à un blocage de l'ordre. Dans le but d'éviter autant que faire se peut ces situations, l'indicateur de trajectoire est utilisé pour prévenir les blocages. Ce mécanisme de détection permet d'anticiper les dérives en se basant sur le respect des contraintes combinatoires. De la même manière, les comportements violant les spécifications temporelles sont détectés en utilisant les indicateurs de temporalité (appartenance à la fenêtre temporelle et évolution de la durée d'activation d'un événement). Enfin, un dernier mécanisme permet au filtre de détecter des attaques visant les équipements de production en identifiant des sur-sollicitations.

Ce chapitre se conclut sur un tableau récapitulatif de la stratégie de détection mise en place dans l'approche S.A.F.E. présentée dans ce manuscrit. Chacun des mécanismes présentés permet de détecter un type d'attaques identifié dans la Partie 1 : attaque brutale, attaque par séquence, attaque temporelle et attaque par sur-sollicitation.

Table des matières

1.	Détection de couples Etat/Action	115
2.	Détection d'anomalies par contraintes combinatoires.....	116
2.1.	Cas de la trajectoire optimale	117
2.2.	Cas d'un ordre non optimal	118
2.3.	Cas d'un compte rendu non conforme.....	120
2.4.	Cas d'une attaque par séquence.....	123
3.	Détection d'anomalies par contraintes temporelles.....	124

3.1.	Mécanisme basé sur l'indicateur de respect des fenêtres temporelles t_{FT}	124
3.2.	Mécanisme basé sur l'indicateur de durée d'activation Δt^{moy}	126
4.	Détection de la dégradation des équipements de production.....	127
5.	Récapitulatif en fonction des attaques.....	129
6.	Bilan du chapitre.....	130

1. Détection de couples État/Action

Ce mécanisme s'appuie sur l'indicateur de distance, développé dans l'algorithme 2 et sur l'algorithme d'exploration des états du système (algorithme 1). A chaque fois que le filtre de commande détecte un couple état/action, également appelé contexte (Sicard, Zamaï, et Flaus 2017b), qui conduit à un état interdit, alors l'ordre est bloqué par le filtre. Un contexte est défini par un état $s \in S_n$ où l'application d'une action $a \in A_m$ conduit à un état interdit $s' \in S_{Pro}$. Ce mécanisme est similaire à une approche par signature basée sur un modèle. En fait, il est utilisé lorsque tous les autres mécanismes, présentés par la suite, ont échoué dans leurs détections. Les règles C permettant de détecter les contextes sont calculées à partir de l'équation (25) et correspondent aux états dont la distance $D(s|S_{Pro})$ est égale à 1 et où l'action du chemin critique est appliquée :

$$C = True \leftrightarrow \exists s' \in S^{i+1} \subset S_n, \exists s \in S^i \subset S_n, \exists a \in A_m \\ \text{tel que } s' = \delta(s, a) \in S_{Pro} \quad (25)$$

Ce mécanisme est exécuté à chaque fois qu'un ordre est analysé par le filtre de commande (Sicard, Zamaï, et Flaus 2017a; Sicard, Flaus, et Zamaï 2017). Lorsque la règle est vraie ($C = True$) alors l'action $a \in A_m$ est stoppée par le filtre et n'est donc pas transmise aux actionneurs et *in fine* appliquée sur la partie opérative. Il s'agit d'un mécanisme de protection similaire à un arrêt d'urgence électrique qui protège contre les attaques directes. Toutefois, le blocage présente de nombreux désavantages pour l'ICS protégé. En effet, le flux de production est interrompu ce qui induit des procédures de redémarrage et donc des temps de redémarrage et des pertes financières. De plus, l'attaquant peut chercher à envoyer un ordre pour bloquer volontairement l'ICS et induire des pertes financières.

Concernant la première limite et en fonction du procédé contrôlé, l'arrêt d'un système industriel par le blocage d'un ordre émanant de la partie commande peut provoquer une destruction matérielle des équipements de production et/ou un risque pour le redémarrage. En ce sens, des algorithmes de mise en repli de l'installation sont donnés à la fin de ce manuscrit, dans la partie perspectives. En cas de détection d'un contexte, le filtre de commande bloque l'ordre analysé, et pilote le système, à l'aide du filtre de comptes rendus, vers un état défini comme sûr (état initial ou prédéfini lors de la conception des filtres). Ainsi, le système peut être protégé d'une anomalie menant à un état interdit (attaque) et mis en sécurité (arrêt du système).

En ce qui concerne les pertes financières liées à l'arrêt du flux de production, ce mécanisme, qui est à la disposition de l'intégrateur, ne s'active qu'en cas de danger imminent (couple état/action particulier). Il s'agit ici de la mise en avant de la caractéristique de « dernier rempart » exprimée dans le cahier des charges de l'approche S.A.F.E. et il se justifie par le fait que sa non-application aurait des conséquences bien plus graves sur l'ICS et sur l'aspect financier.

Afin d'illustrer ce mécanisme de détection de contextes, le système développé en Figure 28 est repris. Dans un tel système, le filtre de commande bloque, lors de l'analyse, les ordres suivants :

- a_i quand le système est dans l'état s puisque l'état $s'_2 \in S_{Pro}^1$ est atteint,

- a_m quand le système est dans l'état s'_3 puisque l'état $s''_4 \in S_{Pro}^2$ est atteint.

Le Tableau 6, situé ci-dessous, regroupe tous les contextes identifiés dans le système considéré en Figure 28.

Tableau 6 : Regroupement des contextes menant à des états interdits du système illustré en Figure 28

État du système (par le filtre de CR)	Action analysée (filtre de commande)	État interdit atteint
s	a_i	S_{Pro}^1
s'_1	a_i	S_{Pro}^1
s'_3	a_m	S_{Pro}^2
s_1^3	a_i	S_{Pro}^1
s_2^3	a_i	S_{Pro}^2
s_2^3	a_m	S_{Pro}^2
s_3^3	a_1	S_{Pro}^2
s_3^3	a_i	S_{Pro}^2
s_2^4	a_i	S_{Pro}^2
s_2^4	a_m	S_{Pro}^2

Le mécanisme à base de règles présenté dans ce paragraphe est une mesure de sécurité dite « statique » (Sicard, Zamaï, et Flaus 2018a) qui ne protège le système que vis-à-vis d'un danger immédiat en bloquant l'ordre analysé. Dans la suite de ce chapitre, les mécanismes construits cherchent à anticiper les dérives afin d'éviter les blocages de l'installation.

2. Détection d'anomalies par contraintes combinatoires

Ce mécanisme de détection se base sur les indicateurs de distance et de trajectoire construits dans le chapitre précédent. Ces derniers sont eux-mêmes issus des modèles comportementaux du système industriel protégé. Ainsi, de par leurs constructions, les indicateurs de distance et de trajectoire sont les garants du respect des contraintes combinatoires.

Ces concepts ont été définis dans le Chapitre 6 – paragraphes 3 et 4 comme ci-après :

- La distance représente le nombre minimal d'actions à appliquer à partir de l'état courant pour atteindre un état interdit.
- La trajectoire est l'évolution de la distance au cours d'une séquence d'états ou temporelle.

A l'aide de ces indicateurs, les filtres sont capables de détecter des déviations lorsque la trajectoire s'éloigne de celle optimale pour tendre vers un état interdit. Le cas échéant, la distance

fournit une information sur le niveau de risque, représenté par l'éloignement, et la criticité, modélisée par la probabilité d'occurrence. Toutefois, la trajectoire se base sur la distance qui calcule pour chaque état uniquement le plus court chemin vers un état critique. Ainsi, la trajectoire du système vis-à-vis des états interdits se doit d'être corrélée avec la distance aux états et aux ordres optimaux. De la même manière, les filtres de commande et de compte rendu échangent leurs informations et travaillent ensemble pour surveiller l'évolution du système, la cohérence des informations échangées et la pertinence des actions menées. Ainsi, les mécanismes de détection embarqués dans les filtres corréleront les informations issues des deux filtres pour identifier des déviations. Nous détaillerons dans la suite de ce paragraphe les différentes situations de détection auxquelles font face les algorithmes de détection par contraintes combinatoires. Le système représenté en Figure 28 est utilisé pour illustrer les différents exemples.

Nous noterons que des cas particuliers ont été choisis pour illustrer les mécanismes de détection mis en jeu ; cependant, ces derniers sont bien évidemment généralisables à tout le système. Ils ont été testés de manière exhaustive (à savoir toutes les combinaisons d'actions possibles) sur des exemples de simulations détaillés dans la Partie 3.

2.1. Cas de la trajectoire optimale

Ainsi, lorsque le système se rapproche d'un état interdit en respectant la loi de commande alors les indicateurs traduisent un comportement normal du système. En effet, les trajectoires $T_{Order}(a|A_{Opt})$ (pour le filtre de commande) et $T_{état}(s|S_{Opt})$ (pour le filtre de commande et de CR) sont nulles ce qui implique que l'ordre analysé correspond à l'ordre attendu par le modèle de commande et que l'état qui sera atteint est similaire à l'état optimal. Dans ces conditions, le rapprochement avec des états interdits est normal et pris en compte dans la loi de commande. Ce raisonnement est conforté par le tracé des trajectoires vues par le filtre de comptes rendus : $T_{état}(s|S_{Opt})$ (nulle ce qui indique un suivi de la trajectoire optimale de l'ICS) et $T_{état}(s|S_{Pro})$ (identique à celle du filtre de commande).

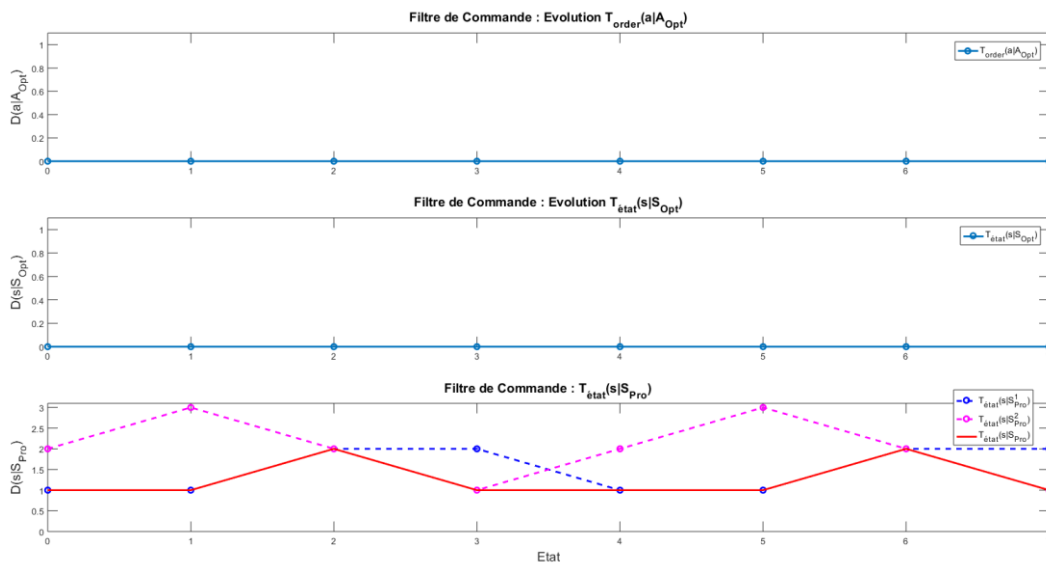


Figure 32 : Trajectoires d'un système respectant la loi de commande (Filtre de commande)

Toutefois, il est encore possible qu'un attaquant lance une attaque directe sur l'ICS qui déclenchera le mécanisme de détection de contexte qui bloquera l'action avant qu'elle ne soit transmise à la PO. Dans ce paragraphe, nous nous intéressons uniquement aux attaques par séquences. La Figure 32 illustre le cas d'un système suivant sa trajectoire optimale (similaire à la Figure 30) avec le filtre de commande et la Figure 33 représente la vue du filtre de comptes rendus.

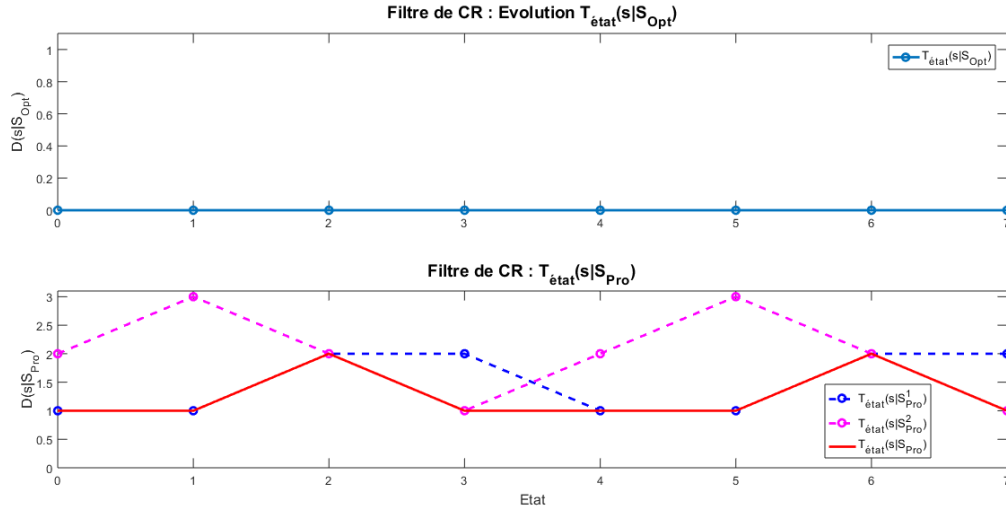


Figure 33 : Trajectoires d'un système respectant la loi de commande (filtre de CR)

2.2. Cas d'un ordre non optimal

Dans cette situation, nous considérons le cas de l'émission d'un ordre anormal qui n'est pas induit par un compte rendu corrompu. La modification de l'action s'est faite via le niveau 2, par infection de l'automate ou via le réseau de communication (attaque MITM entre l'API et le filtre de commande). Ainsi, deux conséquences doivent être considérées suivant si l'ordre conduit le système dans un état interdit ou dangereux. Dans le premier cas, le mécanisme de détection de contexte prendra le relais tandis que, dans le second cas, une alerte est émise pour les opérateurs signalant la dérive par rapport au comportement normal sans pour autant stopper l'ordre analysé.

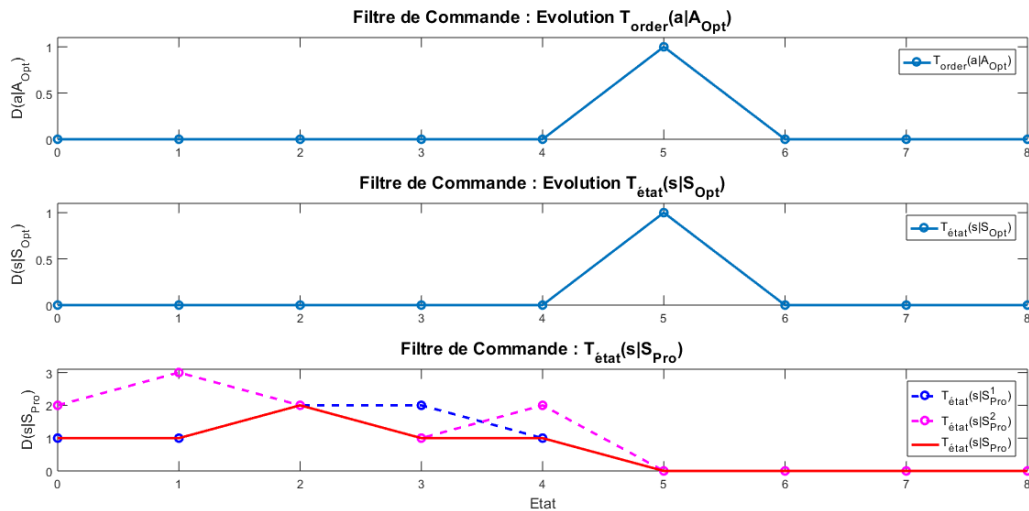


Figure 34 : Trajectoires d'un système avec un ordre menant à un état interdit (filtre de commande)

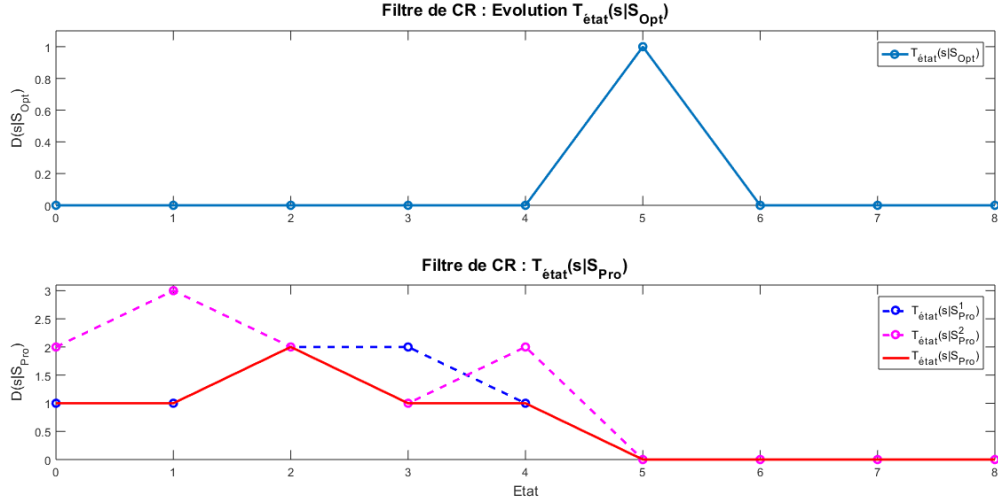


Figure 35 : Trajectoires d'un système avec un ordre menant à un état interdit (filtre de CR)

La Figure 34 et la Figure 35 représentent le cas d'un ordre menant à un état interdit respectivement pour le filtre de commande et de comptes rendus. Sur la Figure 34, et contrairement au cas précédent (2.1. Cas de la trajectoire optimale), le système ne respecte pas la trajectoire optimale lors du passage à l'état 5. L'ordre a_i a été envoyé au système au lieu de l'ordre a_1 . L'état atteint s_2 ne correspond donc pas à l'état attendu s_1 . De plus, sur la trajectoire $T_{état}(s|S_{Pro})$, la distance aux états interdits est nulle ce qui est caractéristique d'un contexte puisque l'état atteint est également un état interdit $s_2 \in S_{Pro}^1$. Le mécanisme de détection de contexte est alors déclenché pour bloquer l'ordre analysé.

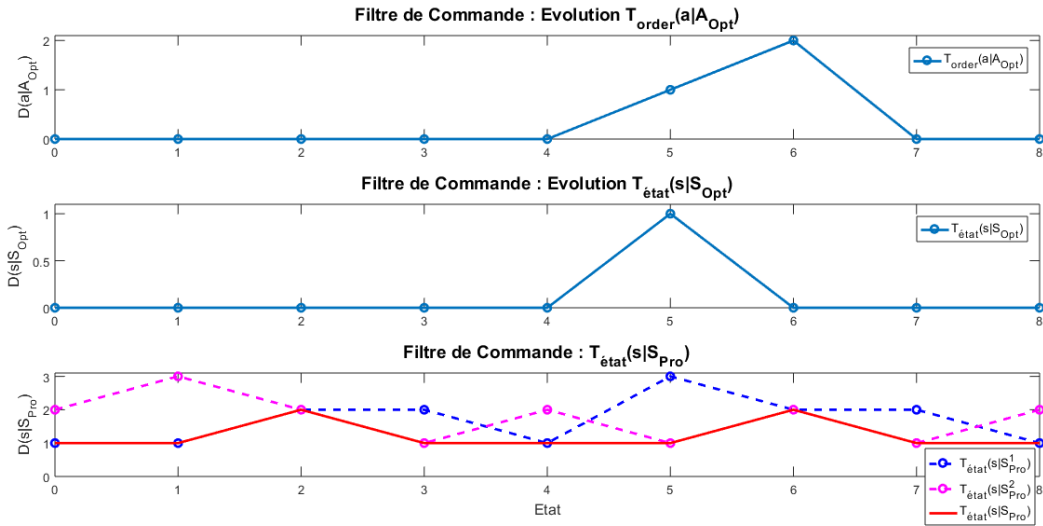


Figure 36 : Trajectoires d'un système avec un ordre menant à un état dangereux (filtre de commande)

La Figure 36 pour le filtre de commande et la Figure 37 pour le filtre de CR illustrent le cas où l'ordre analysé conduit le système dans un état dangereux. Cette situation n'aboutit pas à un blocage du système puisque la trajectoire $T_{état}(s|S_{Pro})$ n'est jamais nulle ce qui traduit le fait qu'aucun contexte n'est rencontré. Ainsi, le système a quitté la trajectoire optimale pour atteindre un état dangereux via un ordre ne respectant pas la loi de commande avant de retrouver des

états optimaux. Cette déduction se fait de l'analyse conjointe des trajectoires $T_{\text{état}}(s|S_{\text{Opt}})$ et $T_{\text{Order}}(a|A_{\text{Opt}})$. En effet le passage de $T_{\text{état}}(s|S_{\text{Opt}})$ à la valeur 1 traduit le fait que l'état s'_3 est atteint au lieu de l'état s'_1 . Similairement, l'ordre a_m a été envoyé au système (au lieu de a_1), expliquant cette dérive. L'envoi de l'ordre a_1 ne respecte également pas la loi de commande mais permet au système de retrouver la trajectoire optimale, signifiée par les valeurs nulles sur $T_{\text{état}}(s|S_{\text{Opt}})$.

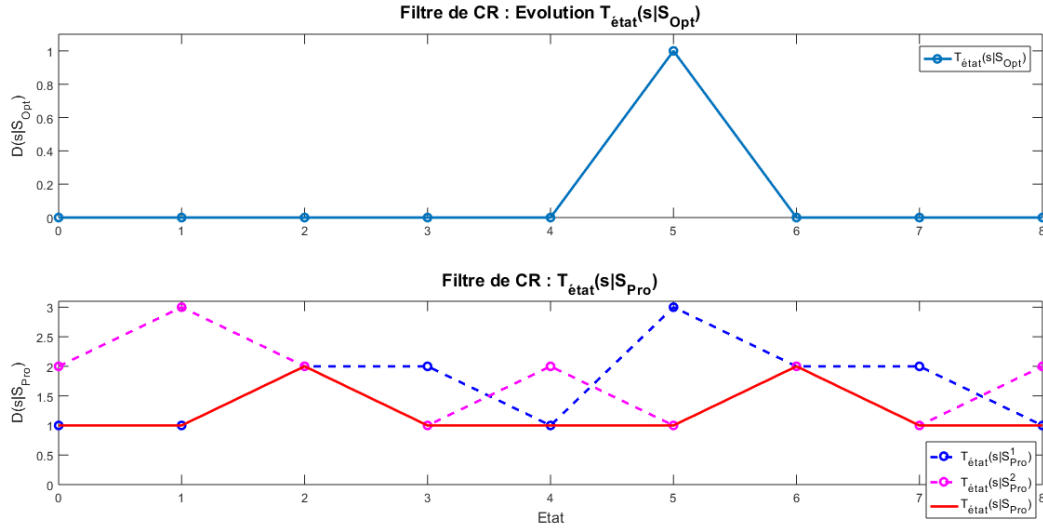


Figure 37 : Trajectoires d'un système avec un ordre menant à un état dangereux (filtre de CR)

Dans les exemples présentés ci-dessus, les résultats fournis par le filtre de comptes rendus et représentés en Figure 35 et Figure 37, ne sont pas prépondérants dans l'analyse alimentant le mécanisme de détection. Le filtre CR valide le fait que le système s'éloigne de la trajectoire optimale et ainsi conforte la détection autour d'un ordre anormal contrairement à la situation présentée dans le prochain paragraphe.

2.3. Cas d'un compte rendu non conforme

Dans ce cas, nous considérons l'émission d'un CR anormal qui n'est pas dû à un ordre malveillant. La modification des données émises par les capteurs a été faite sur le réseau de communication (attaque MITM entre le filtre et l'API). Là encore, deux conséquences sont à distinguer : la modification qui a pour but d'entraîner un ordre malveillant de la part de l'automate et celle qui a pour effet de donner un état ne correspondant pas à la réalité de la partie opérative. Pour ces deux situations, nous retrouvons les figures présentées dans le paragraphe précédent (Figure 34, Figure 35, Figure 36, Figure 37). En effet, comme le filtre de CR est placé au plus proche de la partie opérative, il ne peut pas détecter des attaques sur la communication (réseau industriel, point à point, etc.) remontant les informations de la PO entre le filtre et l'API. Si cette situation se produit, le filtre de commande détectera une action déviant du comportement normal (cf. paragraphe précédent).

Dans la première proposition d'implantation de l'approche, le filtre de CR était localisé au plus proche de l'automate. Il pouvait ainsi détecter des modifications de l'état du système via le réseau de communication (data spoofing). La Figure 38 illustre le placement des filtres différents

(Sicard, Flaus, et Zamaï 2017; Sicard, Zamaï, et Flaus 2017a, 2017b). Dans cette localisation, une modification de l'état par un attaquant serait détectée par le filtre de comptes rendus puisque le CR fourni par la PO et le CR attendu ne seraient pas identiques.

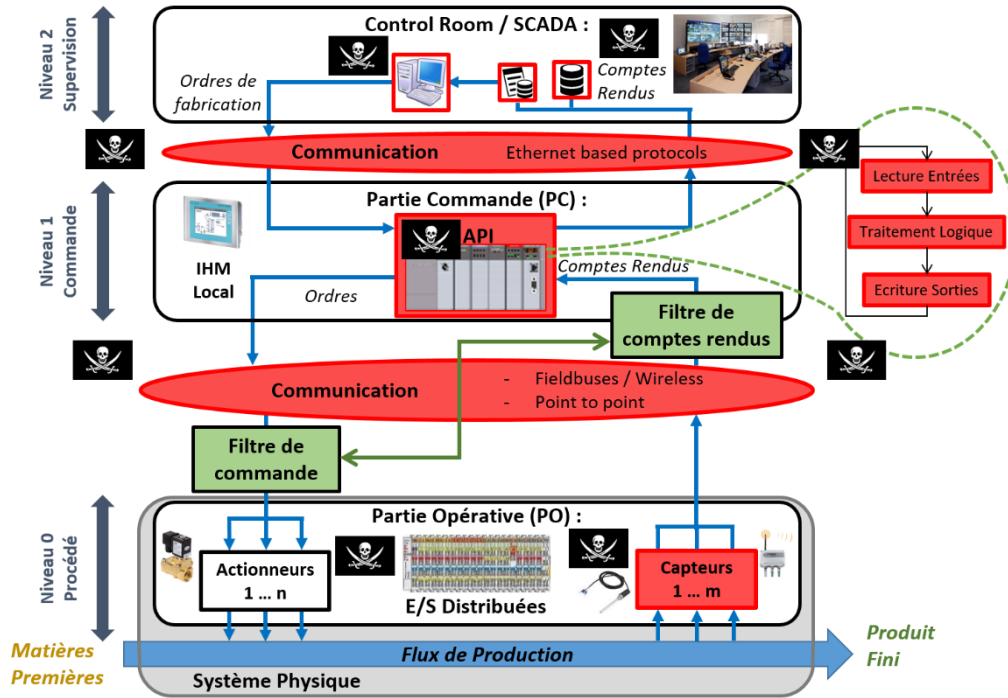


Figure 38 : Localisation des filtres dans la première version de l'approche S.A.F.E.

La précédente localisation permet d'améliorer la détection d'anomalies sur la remontée d'informations (*report channel*) en détectant des attaques par data spoofing par exemple. L'amélioration se situe au niveau de la localisation de l'attaque et de l'élément corrompu (API, *control channel*, *report channel*). Au contraire, l'approche présentée dans ce manuscrit et détaillée dans (Sicard, Zamaï, et Flaus 2018b, 2018d), ne permet pas cette localisation puisque les filtres sont situés au plus proche de la partie opérative. Ainsi, l'attaquant peut corrompre la remontée d'informations (réseau de communication ou l'API), la logique de contrôle (API) ou les actions envoyées à la PO (API ou réseau de communication). En revanche, le gain se situe sur la qualité de l'information reçue pour vérifier si l'action envoyée aux actionneurs est correctement exécutée et pour faciliter le pilotage du système dans le cas du déclenchement d'un mode de repli, notion que nous reprendrons dans les perspectives à ces travaux.

Ce placement relève d'un compromis entre localisation de l'attaque potentielle et la confiance accordée aux données reçues et alimentant les algorithmes de détection. Dans notre approche, si l'action n'est pas correctement exécutée (état reçu différent de l'état attendu) alors le filtre de comptes rendus détecte une incohérence comme dans les approches dites Filtre de (Cruette, Bourey, et Gentina 1991; Lhoste 1991). Pour compléter ce raisonnement, l'ajout d'un troisième filtre sur la communication remontant les informations et avant l'API permettrait un encore meilleur compromis. Toutefois, dans le cadre de ces travaux de recherche, nous nous sommes limités à seulement deux filtres pour analyser les échanges entre la partie opérative et la partie commande.

Dans le cas où l'ICS protégé est composé de capteurs intelligents, pouvant être victimes d'attaques corrompant l'information transmise, le filtre de comptes rendus permet de détecter ces incohérences à l'aide du modèle de la PO comme dans les approches dites Filtre de (Cruette, Bourey, et Gentina 1991; Lhoste 1991). Les Figure 39 et Figure 40 illustrent la détection de ce type d'attaque puisque l'ordre envoyé aux actionneurs et validé par le filtre de commande ne conduit pas à l'état attendu.

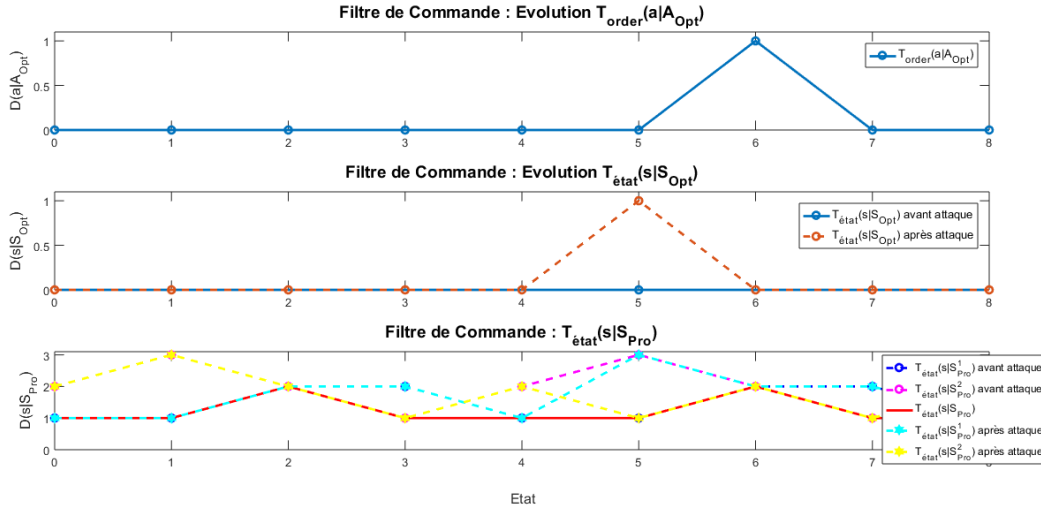


Figure 39 : Trajectories d'un système avec un CR menant à un état dangereux (filtre de commande)

Sur la Figure 39, les courbes de $T_{état} s|S_{Pro}$ en bleu et magenta indiquent les distances aux états critiques calculées pour l'ordre optimal conduisant à l'état optimal attendu tandis que les courbes cyan et jaune indiquent ces mêmes distances pour l'état réellement atteint. Cette modification d'état est également visible sur la trajectoire $T_{état}(s|S_{Opt})$ où la courbe en pointillés souligne ce changement détecté par les filtres concernant l'état n°5 de la séquence. Enfin, nous remarquerons la variation de $T_{Order}(a|A_{Opt})$ correspondant à l'action correctrice pour rejoindre la trajectoire optimale.

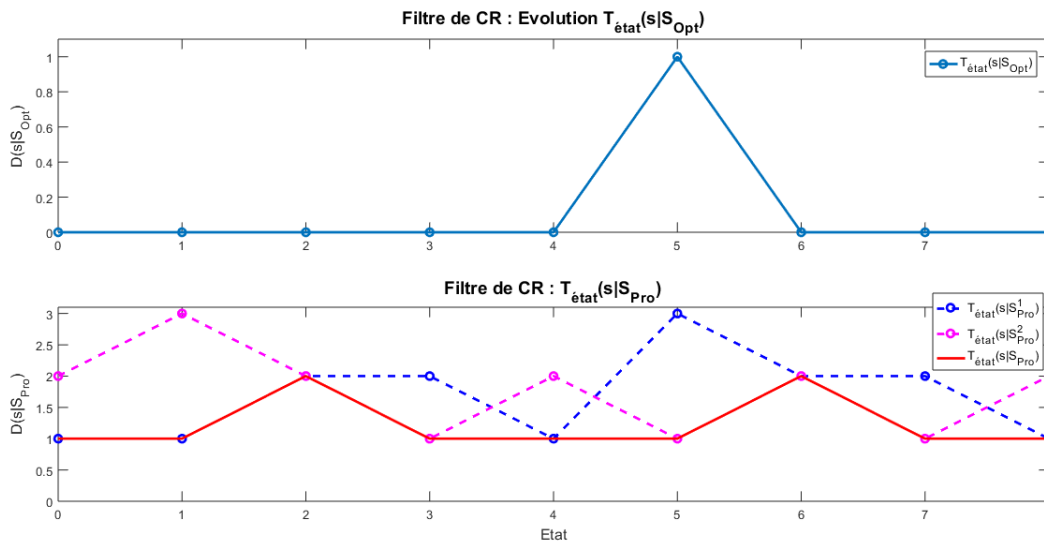


Figure 40 : Trajectories d'un système avec un CR menant à un état dangereux (filtre de CR)

Cette détection est en fait réalisée par le filtre de comptes rendus comme le montre la Figure 40. L'état reçu par le filtre de CR ne correspond pas à celui attendu, c'est-à-dire l'état résultant de l'exécution de l'ordre envoyé. Cette information est envoyée au filtre de commande qui la prend en compte (les courbes en pointillé indiquent cette modification). Dans le même temps, les distances aux états critiques $D(s|S_{Pro})$ sont recalculées dans le filtre de comptes rendus.

2.4. Cas d'une attaque par séquence

Dans les paragraphes précédents, les exemples ont permis de mettre en avant différents cas de figures où le concept de trajectoire est utilisé pour détecter des anomalies comportementales et avertir les opérateurs. Toutefois, pour faciliter la compréhension, les exemples faisaient référence à des états dangereux et interdits simples.

Dans l'exemple ci-dessous, la situation se dégrade peu à peu et dérive de la trajectoire optimale vers un état interdit ($s_2'' \in S_{Opt} \rightarrow s_2^3 \in S_{Dan} \rightarrow s_2^4 \in S_{Dan} \rightarrow s_1^5 \in S_{Pro}$). Ici, le concept de trajectoire prend tout son sens puisqu'il permet d'indiquer aux opérateurs cette déviation. Ils peuvent ainsi agir pour corriger la tendance prise par l'ICS avant d'arriver dans une situation de blocage.

La Figure 41 représente le comportement perçu par le filtre de commande au travers des trajectoires $T_{Order}(a|A_{Opt})$, $T_{état}(s|S_{Opt})$ et $T_{état}(s|S_{Pro})$. La séquence de commande dérive à partir de l'état n°7 jusqu'à atteindre un état interdit, cela correspond à l'application de l'ordre a_i au lieu de a_m . Par la suite, les ordres a_1 et a_m sont envoyés à la partie opérative pour atteindre l'état $s_1^5 \in S_{Pro}^1 \subset S_{Pro}$. La Figure 42 confirme cette déviation de la trajectoire optimale au travers du filtre de comptes rendus. Toutefois, nous remarquons que l'API n'envoie pas d'ordre corrigeant cette déviation ce qui laisse penser que la partie commande et/ou le réseau de communication ont été corrompus. Le concept de trajectoire, basé sur la prise d'information des filtres dans l'architecture ICS, permet d'informer les opérateurs de cette déviation avant d'atteindre un état critique. La détection des signaux de comportements anormaux se fait en étudiant les séquences et l'évolution sur le long terme des échanges entre la PC et la PO.

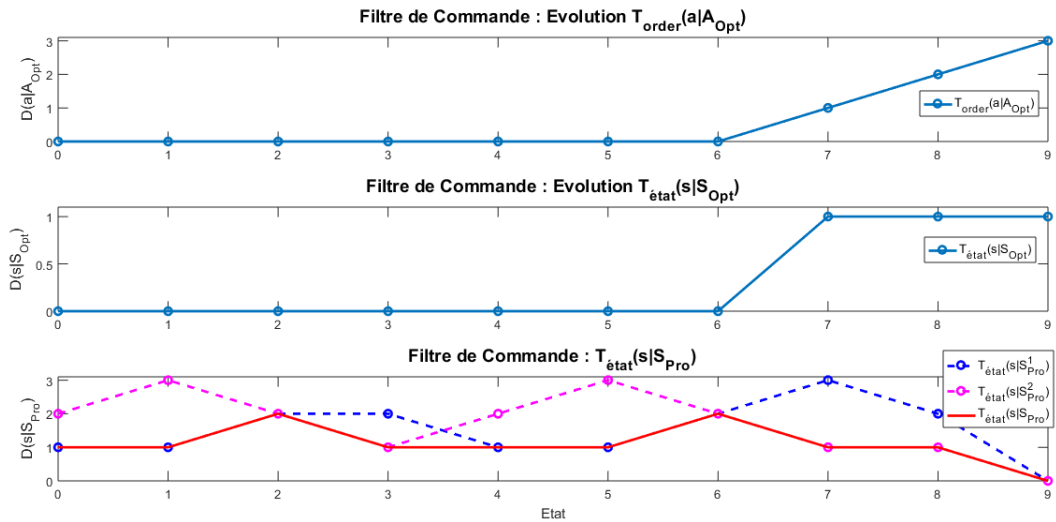


Figure 41 : Trajectoires d'un système avec une séquence menant à un état interdit (filtre de commande)

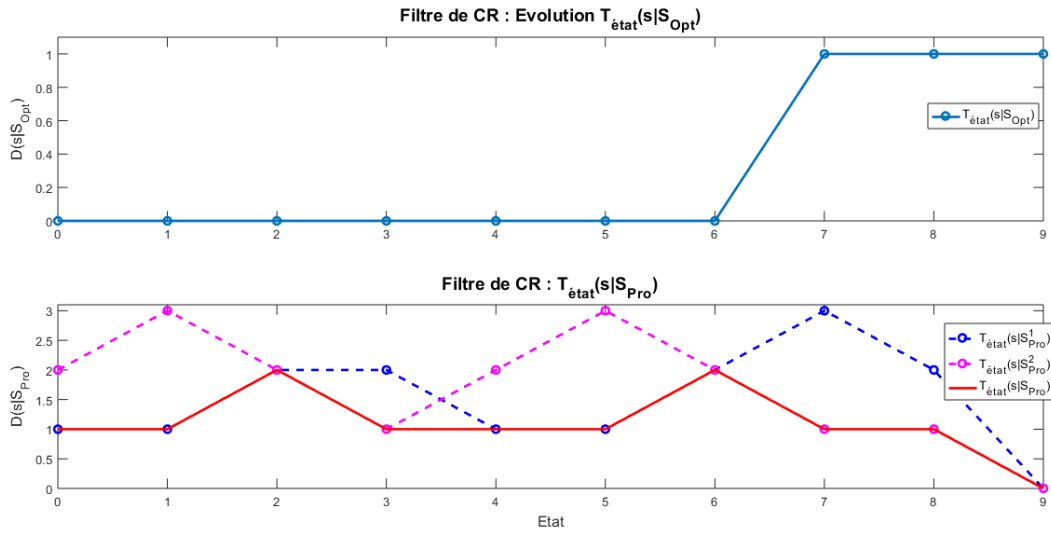


Figure 42 : Trajectoires d'un système avec une séquence menant à un état interdit (filtre de CR)

Pour conclure cet exemple, le filtre de commande bloque l'action a_m envoyée à la PO en l'état n°9 puisqu'elle conduit à un état interdit. L'ensemble des mécanismes présentés dans ce paragraphe utilise les contraintes combinatoires définies dans l'étape 1 de la méthodologie de conception (Chapitre 4 - 2.1. Spécification des états du système). Dans les prochains paragraphes, les mécanismes liés aux contraintes combinatoires et aux sur-sollicitations sont explicités.

3. Détection d'anomalies par contraintes temporelles

Ce mécanisme de détection s'assure que les contraintes temporelles spécifiées dans l'étape 1 de la méthodologie de conception des filtres (Chapitre 4 - 2.2. Spécification des contraintes temporelles du système) soient respectées. Il s'appuie sur deux indicateurs expliqués dans le chapitre précédent (Chapitre 5 - 5. Indicateurs de temporalité). Nous précisons que ces mécanismes sont exécutés en parallèle de la vérification des contraintes combinatoires et reposent sur la vérification des indicateurs suivants : le respect des fenêtres temporelles t_{FT} et l'évolution de la durée moyenne d'activation Δt^{moy} .

3.1. Mécanisme basé sur l'indicateur de respect des fenêtres temporelles t_{FT}

Ce mécanisme se base sur le respect des gabarits spécifiés lors de l'analyse temporelle du système sur les ordres et les comptes rendus échangés par l'ICS. Ces fenêtres temporelles sont définies par une date de début, une date de fin et une période d'exécution qui sont représentées sur la Figure 43 par les courbes en pointillés avec des cercles (date de début et de fin) et des signes plus (période d'exécution). Sur l'exemple illustré ci-dessous, les actions envoyées au système respectent les fenêtres temporelles spécifiées. Ainsi, l'indicateur $t_{FTOrder}$ n'est pas nul ce qui valide le respect des contraintes temporelles sur les ordres envoyés à la partie opérative selon l'équation (22). Cette séquence de commande correspond à la trajectoire optimale pour le système illustré en Figure 28.

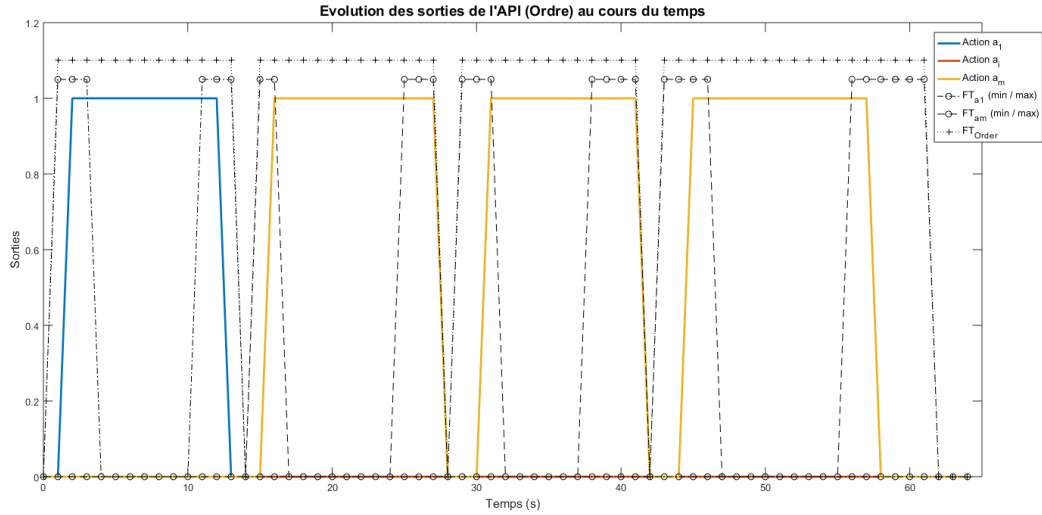


Figure 43 : Évolution des ordres envoyés à la partie opérative au cours du temps avec les FT correspondantes

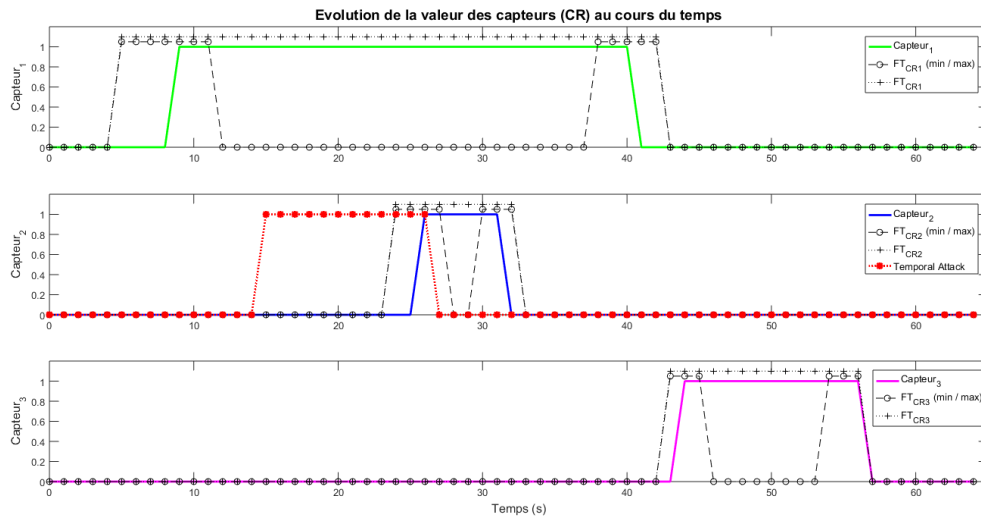


Figure 44 : Évolution de la valeur des capteurs de la partie opérative au cours du temps avec les FT

La Figure 44 représente l'évolution des valeurs prises par les capteurs du système protégé (courbes pleines) avec les fenêtres temporelles correspondantes (pointillés avec cercles (apparition et fin de l'événement) et avec signes plus (durée d'activation de l'événement)). Les comptes rendus envoyés par la PO respectent les gabarits temporels spécifiés pour le système pour les capteurs 1 et 3. Le capteur 1 s'active à la fin de l'action a_1 et se désactive à la fin de la deuxième action a_m . De la même manière, le capteur 3 est actif en même temps que la troisième action a_m . En revanche, un attaquant perturbe l'activation du capteur 2. Ce dernier doit se déclencher à la fin du premier ordre a_m et se désactiver au début de la deuxième action a_m . Sur la courbe représentant l'évolution du capteur 2 sur la Figure 44, l'attaquant fait croire que le capteur est actif dès le début de la première action a_m . Ainsi, le gabarit n'est pas respecté ce qui entraîne une valeur nulle pour l'indicateur $t_{FT_{CR2}}$. Dans cet exemple, le non-respect des contraintes temporelles n'entraîne pas de risques pour la sécurité des biens et des personnes, seulement un risque vis-à-vis de la qualité du produit final. Ainsi une alerte est envoyée par les filtres aux

opérateurs pour signaler l'anomalie. Nous notons que la réaction face à cette détection doit pouvoir être modulée en fonction du procédé contrôlé. Ainsi, dans le cas où l'ICS pilote un système dont la qualité du produit est primordiale d'autres réactions suite à la détection doivent être mises en œuvre comme dans le cas du secteur du nucléaire ou de l'industrie du luxe par exemple.

Nous remarquons au travers de cet exemple que ce mécanisme de détection permet d'identifier des attaques qui respectent les contraintes combinatoires du système. Toutefois, suivant le type d'ICS à protéger, la spécification des contraintes peut être complexe (gabarits trop stricts ou au contraire trop larges) pouvant conduire à des faux positifs ou à des non-détections.

3.2. Mécanisme basé sur l'indicateur de durée d'activation Δt^{moy}

Un deuxième mécanisme a été présenté dans le chapitre précédent pour détecter des attaques temporelles. Il se base sur l'étude de l'évolution de la durée d'activation des événements (actions ou comptes rendus). Cet indicateur Δt^{moy} vient en complément des fenêtres temporelles pour s'adapter à la variabilité inhérente aux systèmes industriels.

Pour illustrer ce mécanisme, nous utiliserons les ordres représentés sur la Figure 43 ci-dessus. La Figure 45 présente l'évolution de la durée d'activation de chacun des ordres activés lors de la trajectoire optimale de l'exemple illustré en Figure 28. La courbe en pointillés représente la moyenne de la durée d'activation de l'action considérée, celles en pointillés avec des cercles signalent les durées maximales et minimales autorisées par les contraintes temporelles et les courbes pleines illustrent les variations de durées d'exécution sur un exemple particulier permettant de mettre en avant les différentes détections possibles.

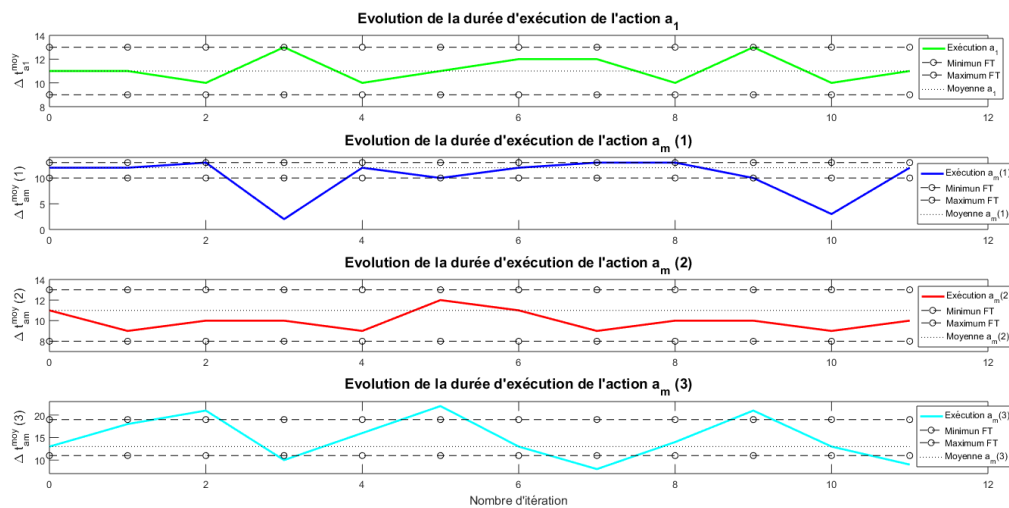


Figure 45 : Évolution de la durée d'activation des ordres

Ainsi, sur la première courbe montrant l'évolution de $\Delta t_{a_1}^{moy}$, nous remarquons que les variations d'exécution restent comprises dans les bornes. Le comportement de l'action a_1 est donc normal. En revanche, l'évolution de $\Delta t_{a_m}^{moy}(1)$ qui représente l'évolution de la durée d'activation

de l'action a_m lors de son premier appel (passage de l'état s'_1 à l'état s'_2), présente des irrégularités. En effet, deux pics aux itérations 3 et 10 signalent une durée d'activation ne correspondant pas aux gabarits fixés par les contraintes temporelles. Les filtres envoient donc une alerte aux opérateurs. Les variations de $\Delta t_{a_m}^{moy}(2)$ sont normales et comprises dans l'écart admissible. Enfin, l'évolution de $\Delta t_{a_m}^{moy}(3)$ est suspecte puisque régulièrement la durée d'activation ne respecte pas le gabarit fixé (à l'itération 2, 3, 5, 9 et 11). De plus, les variations d'un cycle d'exécution à l'autre sont très importantes et ne suivent pas les mêmes tendances. Cette dernière situation soulève la question de la définition des seuils lors de l'analyse temporelle puisque les dépassements sont très faibles. Toutefois, une alerte est envoyée aux opérateurs à cause du comportement de la durée d'activation qui ne cesse de varier de façon importante.

4. Détection de la dégradation des équipements de production

Ce mécanisme de détection a pour objectif d'identifier des séquences d'ordres qui respectent les contraintes combinatoires spécifiées pour le système mais dont l'application conduit à une sur-sollicitation des actionneurs. Les conséquences vont alors de l'usure prématurée, qui amène des maintenances préventives sur les équipements, à la casse du matériel.

Dans l'objectif de prévenir ces comportements, un indicateur « O » a été proposé dans le chapitre précédent. Il surveille l'occurrence des ordres par rapport à l'évolution de la partie opérative comme défini dans l'équation (24).

En prenant l'exemple d'un système composé de cinq actionneurs, le principe de fonctionnement de ce mécanisme de détection s'explique ainsi : le gabarit d'exécution de chacune des actions est connu par l'analyse du système et représenté en Figure 46. Le scénario d'attaque, qui consiste à sur-solliciter les actionneurs 1 et 5 quand ils sont activés, est également représenté en pointillés.

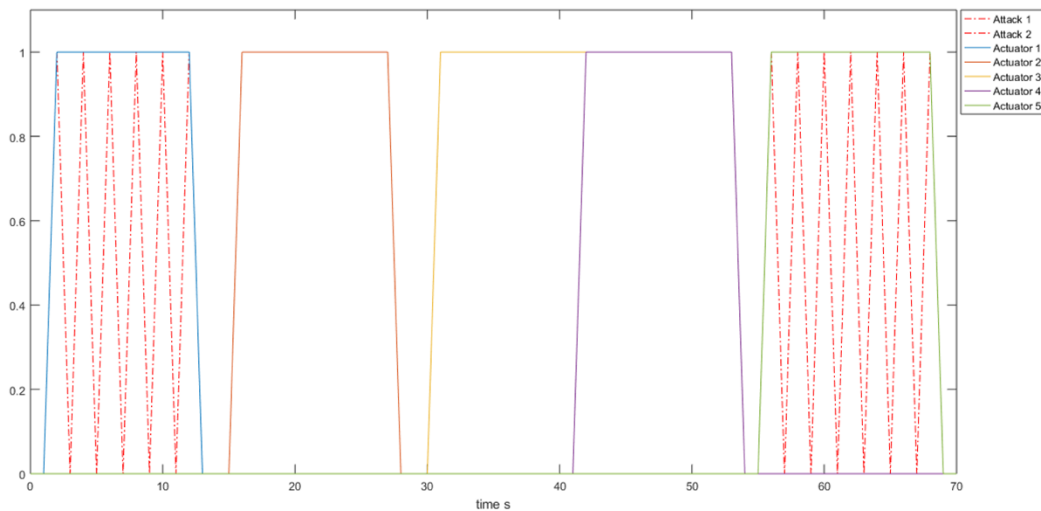


Figure 46 : Étude des sur-sollicitations sur un système à 5 actionneurs (actionneurs 1 et 5 affectés)

Lors de l'exécution de l'ordre, le filtre de commande constate que les actions 1 et 5 ne sont pas maintenues durant toutes leurs activations mais oscillent entre la valeur ON (1) et OFF (0). Cela découle d'un comportement anormal sur le système protégé et déclenche une alerte envoyée aux opérateurs. Ce mécanisme de détection permet de prévenir l'usure prématurée des équipements de protection. Ainsi, il assure la qualité du flux de production et non la sécurité des biens et des personnes. Nous noterons également que les contraintes temporelles ne sont pas respectées durant cette attaque. La corrélation entre ces deux informations permet de cibler une sur-sollicitation plutôt qu'une attaque temporelle.

Nous notons qu'ici l'attaquant marque « correctement » son attaque en faisant osciller la valeur entre 0 et 1. Si la variation était moins marquée, la détection de ces comportements deviendrait plus complexe. En effet, lorsque la valeur de commande envoyée aux actionneurs est 0,6 elle est arrondie à 1. Ainsi, tel qu'est implanté ce mécanisme, il ne permet pas de détecter des attaques complexes sur la partie opérative requérant une connaissance plus fine.

5. Récapitulatif en fonction des attaques

Tableau 7 : Récapitulatif des attaques considérées pour notre approche

Mécanisme de détection	Attaque ciblée	Indicateur utilisé	Principe de fonctionnement	Commentaire
Détection de contexte	Brutale	Distance aux états interdits $D(s S_{Pro})$	Détection de couple État/Action (contexte) où l'exécution de l'action dans l'état courant du système conduit à un état interdit L'action est bloquée par le filtre de commande	Protection du système contre un danger immédiat. Risque de dégradation liée au blocage de l'ordre
Détection par contraintes combinatoires	Séquentielle	Concept de distance Concept de trajectoire	Anticiper les déviations du système pour détecter au plus tôt les comportements anormaux et prévenir les opérateurs	Anticipation possible mais localisation précise de l'élément compromis impossible
Détection par contraintes temporelles	Temporelle	Respect des fenêtres temporelles t_{FT} Évolution de la durée d'activation Δt^{moy}	Vérifications des spécifications temporelles du système réalisées lors de l'analyse temporelle (Étape 1 de la méthodologie). Les filtres vérifient le respect des fenêtres temporelles et l'évolution des durées moyennes d'exécution	Problème lié au choix du seuil des contraintes temporelles. Le dépassement et/ou les variations doivent être importants pour entraîner une détection
Détection de sur-sollicitation	Sur-sollicitatrice	Sur-sollicitation $O = \frac{\sum \Delta a^2}{\sum \Delta s^2}$	Détection des actions pouvant détériorer les équipements de production et vérifiant leurs exécutions correctes	La détection est assurée par l'étude des variations du rapport O

6. Bilan du chapitre

Dans de ce chapitre, les différents mécanismes de détection mis en œuvre dans les filtres de commande et de comptes rendus ont pu être détaillés. Ces derniers s'appuient sur les indicateurs mis en place dans le chapitre précédent. Ainsi, la stratégie de détection dans les systèmes industriels s'appuie sur les mécanismes suivants :

- Détection de contexte. Lorsque le filtre de commande identifie un couple état / action menant vers un état interdit, il bloque l'action avant qu'elle ne soit appliquée au système.
- Détection d'anomalies par contraintes combinatoires. Les concepts de distance et de trajectoire sont mis en œuvre dans ce mécanisme pour anticiper des dérives vers des états interdits avant d'arriver à une situation de blocage (détection de contexte). Ce mécanisme vérifie le respect des contraintes combinatoires du système.
- Détection d'anomalies par contraintes temporelles. Ce mécanisme assure l'identification d'événements (action ou CR) ne respectant pas les spécifications temporelles du système. Les indicateurs se basent sur l'appartenance à la fenêtre temporelle et l'évolution de la durée moyenne d'activation. Cette détection cible les attaques temporelles sur les ICSs.
- Détection de sur-sollicitations des équipements de production. L'objectif de ce mécanisme est d'identifier des variations dans la commande des actionneurs afin de prévenir des sur-sollicitations qui entraînent usures, maintenances et casses du matériel. Des attaques de ce type entraînent également des alertes de la part des mécanismes de détection d'anomalies par contraintes temporelles.

L'ensemble de ces mécanismes de détection permet d'alimenter la stratégie de détection de l'approche qui est basée à la fois sur les modèles comportementaux et sur la localisation des filtres dans l'architecture ICS.

Partie 3 :

Application de l’approche

S.A.F.E.

Chapitre 7

Exemples de simulation

Ce chapitre est consacré à la mise en application de l'approche S.A.F.E. sur des exemples de simulation inspirés de systèmes industriels. Les objectifs sont multiples : mettre en avant l'aspect généraliste de la méthodologie proposée, montrer l'applicabilité de l'approche sur des systèmes de différentes complexités et vérifier le bon fonctionnement de la stratégie de détection mise en œuvre. Dans un premier temps, le cadre général dans lequel les simulations sont exécutées est défini ainsi que le type d'attaques pris en considération pour tester notre approche. Des profils types d'attaquants sont ainsi dégagés correspondant aux menaces que l'approche S.A.F.E. se doit de détecter. Ensuite, les différents exemples inspirés de systèmes de contrôle-commande industriels sont présentés. Pour chacun de ces systèmes, la méthodologie de conception est appliquée. Ainsi, les états interdits et les paramètres devant être pris en compte sont identifiés, l'analyse temporelle du système est réalisée, les modèles de commande et de la partie opérative sont construits et l'algorithme de génération du modèle de filtre de commande est présenté. Enfin, les scénarii construits sur la base des profils d'attaquants sont appliqués à chacun de ces systèmes. Les résultats de détection des filtres sont analysés et critiqués. En conclusion, un tableau récapitulatif permettra d'avoir une vue d'ensemble sur les différents éléments de ce chapitre.

Table des matières

1.	Généralités.....	135
1.1.	Cadre de modélisation et d'attaques.....	135
1.2.	Profils d'attaquants.....	136
1.3.	Déroulement des tests.....	137
2.	Présentation des systèmes et Application de l'approche S.A.F.E.....	138
2.1.	Système à 3 cuves.....	138
2.2.	Système à 5 cuves.....	143
3.	Analyse des scénarii d'attaques.....	147
3.1.	Comportement normal du système.....	148
3.2.	Scénario 1 : Attaque sur l'ICS et son environnement.....	150
3.3.	Scénario 2 : Attaque sur la qualité.....	151
3.4.	Scénario 3 : Attaque discrète.....	153

3.5.	Scénario 4 : Attaque sur les équipements.....	154
4.	Tableau récapitulatif	155
5.	Bilan du chapitre.....	156

1. Généralités

1.1. Cadre de modélisation et d'attaques

Afin de valider notre approche, nous commencerons par définir le périmètre de l'étude expérimentale en reprenant les hypothèses émises au début de ce manuscrit. Les filtres de commande et de comptes rendus sont placés au plus proche de la partie opérative afin d'analyser les échanges entre la PO et la PC. La stratégie de détection est basée sur les modèles comportementaux construits à l'aide de la méthodologie proposée dans l'approche S.A.F.E. ainsi que sur les mécanismes de détection développés à cet effet. Ainsi, dans cette partie application, nous considérons les filtres comme non-attaquables sur le plan hardware et *secured by design*. De la même manière, nous supposons que les filtres communiquent sur un réseau qui leur est propre, isolé du réseau de l'ICS garantissant ainsi sa non-compromission. Enfin, nous prenons l'hypothèse selon laquelle les modèles contenus dans les filtres et la partie opérative sont synchronisés et le restent durant l'intégralité des tests.

Sur les différents systèmes considérés, l'analyse de risques est présumée comme réalisée ce qui induit que les parties de l'ICS à protéger sont connues. Les états interdits ainsi que les paramètres à prendre en compte ont été étudiés et sont disponibles pour appliquer la démarche. De manière similaire, nous prenons l'hypothèse selon laquelle la loi de commande ne présente pas d'erreur tout comme le modèle de filtre de commande généré. Enfin, nous admettons que les systèmes industriels simulés dans la suite de ce chapitre prennent en compte la sûreté de fonctionnement au niveau de la logique de commande et des équipements et que les opérateurs de l'ICS ne sont pas hostiles. Enfin, les capteurs sont considérés comme sûrs et supposés non défaillants au sens de (Villemeur 1992).

Toute attaque, quel qu'en soit le type (FDI, DDOS, MITM, ...), qui cherche à impacter directement la partie opérative en nuisant à la sécurité des biens et des personnes, ou en dégradant la qualité du produit ou en endommageant les équipements de production, est prise en compte. Nous retenons particulièrement *les attaques aléatoires, directes, séquentielles, temporelles ou entre les injections de fausses commandes* (communication de la PC à la PO) *et de fausses données* (remontée d'informations de la PO vers la PC). Nous rajouterons aussi dans le périmètre les attaques affectant les équipements de production via des *attaques par sur-sollicitation*. Ainsi, les vecteurs pouvant être utilisés sont :

- Les attaques *Man-In-The-Middle* (MITM) sur la communication entre la partie commande (ou les niveaux supérieurs) et la partie opérative où la personne malveillante vient agir en direct sur l'ICS.
- La compromission de la partie commande par l'attaquant telle que l'infection physique ou au cours d'une intervention des équipes de maintenance (maintenance, télémaintenance).

Enfin, nous ajoutons l’hypothèse d’unicité de l’attaque. Ainsi, le cas où l’attaquant exerce en parallèle plusieurs types d’attaques n’entre pas dans le cadre de ces travaux de recherche. L’ensemble de ces éléments forme le périmètre dans lequel est testée l’approche proposée.

1.2. Profils d’attaquants

Au vu des attaques considérées dans le périmètre défini dans le paragraphe précédent, quatre profils d’attaquants ont été sélectionnés pour tester l’approche. Ces derniers sont construits autour de trois points clés déterminant l’attaque qui sera exécutée :

- L’objectif ou les objectifs recherchés par l’attaquant : destruction de la partie opérative et de son environnement, détérioration de la qualité du produit, dégradation des équipements de production.
- Les connaissances de l’ICS. Ce point regroupe à la fois le savoir général en automatisme industriel de l’attaquant (organisation et fonctionnement d’un ICS) ainsi que les informations qu’il a pu réunir sur la cible de son attaque telles que le procédé contrôlé, les automates utilisés (marque, firmware, configuration), le mode de communication (réseau terrain, Ethernet ou point à point) ou encore la surveillance du site (accès physique, opérateur de supervision, opérateur de maintenance ...). L’ensemble de ces éléments va déterminer en partie la discrétion de l’attaque.
- Les moyens utilisés durant l’attaque comme le type d’attaque choisi, le ou les éléments à corrompre. Ce troisième point définit également le niveau de discrétion du scénario d’attaque.

L’ensemble de ces éléments permet de définir des scénarii d’attaques qui sont ensuite intégrés aux exemples de simulation pour vérifier les capacités de détection des filtres. Les profils retenus sont représentés en Figure 47 et détaillés ci-après :

- Profil n°1 (Dégradation de l’ICS par attaque brutale) :
Cet attaquant ne possède pas de connaissances poussées sur le système industriel attaqué. Même s’il connaît le fonctionnement d’un système industriel, il ne possède pas d’informations sur le système ciblé. En revanche, il a identifié la partie de l’ICS à attaquer. Dans ce scénario, nous considérerons que l’intrusion a un niveau de discrétion très faible. L’objectif est la destruction de la partie opérative de l’ICS et de son environnement. Ce scénario est similaire à celui d’une attaque brutale.
- Profil n°2 (Dégradation de la qualité des produits) :
L’attaquant a un niveau de connaissance sur l’ICS ciblé supérieur au profil précédent. Il a notamment des informations sur le procédé contrôlé par le système ciblé. Pour mener à bien son objectif, l’attaquant cherche à modifier des paramètres impactant la qualité du produit en sortie de l’ICS. Le niveau de discrétion de ce scénario est intermédiaire puisque l’impact sur la qualité est permanent et facilement identifiable sur le flux de production par les opérateurs de production lors de phase de contrôle du produit. Ce scénario fait appel à la fois à des attaques par séquences et temporelles.
- Profil n°3 (Dégradation de l’ICS et/ou de la qualité mais avec un haut niveau de discrétion) :

Le niveau de connaissance de l'attaquant dans ce scénario est très élevé et nous considérerons qu'il possède toutes les informations nécessaires sur le système industriel ciblé ainsi que sur le fonctionnement théorique des ICSs. L'objectif principal est la discrétion des malveillances visant à dégrader la qualité de la production ou la sécurité des biens et des personnes. Ce scénario utilise des attaques par séquences et temporelles mais sont plus complexes à détecter que dans le scénario précédent.

- Profil n°4 (Dégradation des équipements de production) :

De la même manière que pour le profil précédent, les connaissances de l'attaquant sont totales sur l'ICS ciblé ainsi que sur son fonctionnement théorique. Ce scénario considérera que les informations acquises ont permis d'identifier un ensemble d'équipements à cibler pour mettre à mal la production. Ici, l'attaquant ne cherche pas à être discret puisque le degré de précision du mécanisme associé à ce type d'attaque ne permet pas l'identification de malveillances discrètes, au sens de la détection. Les attaques utilisées dans ce scénario s'inspireront d'attaques sur les équipements de production.

<u>Profil n°1</u>	    	<i>Faibles connaissances sur l'ICS ciblé</i> Objectif : Détruire le système et son environnement
<u>Profil n°2</u>	    	<i>Connaissances sur le procédé contrôlé par l'ICS</i> Objectif : Affecter la qualité du produit sans vouloir être discret
<u>Profil n°3</u>	    	<i>Fortes connaissances sur l'ICS ciblé,</i> Objectif : Dégrader la qualité et l'intégrité physique du système en étant le plus discret possible
<u>Profil n°4</u>	    	<i>Fortes connaissances sur l'ICS ciblé et des équipements de production</i> Objectif : Endommager les équipements de production

Figure 47 : Récapitulatif des différents profils d'attaquants sélectionnés pour tester l'approche proposée

1.3. Déroulement des tests

Dans les paragraphes suivants, les exemples de simulation inspirés de systèmes industriels sont décrits tant au niveau des spécifications de la loi de commande que de la partie opérative. Par la suite, la méthodologie proposée avec l'approche S.A.F.E. est appliquée et permet d'identifier les paramètres à prendre en compte et la construction des modèles de filtres de commande et de la partie opérative sur la base de la description qui en aura été faite. Puis, les mécanismes de détection sont implantés dans les filtres de commande et de comptes rendus. Ainsi, pour chacun des systèmes introduits, la méthodologie est appliquée pour construire les modèles comportementaux associés aux exemples. Enfin, les résultats des filtres pour chacun des profils d'attaquants sont expliqués et critiqués. Dans ce manuscrit, l'analyse se limitera à un certain nombre d'attaques significatives pour démontrer le fonctionnement des mécanismes développés, ou intéressantes pour établir les limites de l'approche. Toutefois, la robustesse des filtres a été testée en appliquant toutes les combinaisons d'attaques possibles en utilisant les paramètres d'entrées / sorties du système.

2. Présentation des systèmes et Application de l’approche S.A.F.E.

Les exemples utilisés dans ce chapitre sont inspirés de systèmes industriels référencés dans la littérature (systèmes à cuves) et d’attaques réelles (système à chariot, attaque du tramway à Lodz). Ils permettent de tester l’approche S.A.F.E. sur des systèmes en adéquation avec la réalité, tout en disposant d’informations sur la manière de modéliser et tout en maîtrisant la complexité. En effet, pour ce dernier point, l’exemple du système à 3 cuves représente une sous-partie du système à 5 et 6 cuves.

Chacun de ces exemples permet de tester un point de notre approche, ainsi le :

- Système à 3 cuves permet de vérifier le mécanisme détectant les attaques directes.
- Système à 5 cuves est adapté pour s’assurer du bon fonctionnement des indicateurs de distance et de trajectoire sur des attaques séquentielles.
- Système à 6 cuves permet d’évaluer l’approche sur un cas plus complexe par le nombre de paramètres à gérer.
- Système à chariot permet de vérifier la genericité de la démarche.
- Et l’évolution de la complexité d’un même système (cuves) permet de projeter les performances de l’approche proposée en modélisation et détection.

Enfin, les algorithmes pour générer le modèle du filtre de commande et pour détecter des attaques sont testés sur des systèmes physiques réels.

2.1. Système à 3 cuves

Il s’agit d’un exemple classique de la littérature qui se retrouve dans de nombreuses publications comme (W. Li et al. 2016) et qui est représenté en Figure 48. Le système est composé de trois cuves, deux d’entre elles C_1 et C_2 sont considérées comme étant de capacités infinies et contiennent respectivement le produit A et le produit B. Chacune des cuves déverse son produit dans un réservoir de mélange C_3 afin de produire le produit C. L’étape de remplissage est déclenchée par l’ouverture des vannes de remplissage V_1 et V_2 . L’étape de vidange est réalisée par l’ouverture de la vanne V_3 . Trois capteurs de niveau sont placés dans le réservoir et permettent de connaître la quantité de produit dans la cuve de mélange :

- H_0 est le capteur de fin de cuve. Il indique un remplissage lorsqu’il prend la valeur 1 et une vidange totale de la cuve C_3 quand la valeur est nulle.
- H_1 est localisé à un niveau intermédiaire de la cuve de mélange.
- H_2 est le capteur de niveau haut de la cuve C_3 .

Les spécifications de la loi de commande imposent de démarrer un cycle de production en H_0 . Ensuite, un premier remplissage de la cuve C_3 est réalisé par le produit A en ouvrant la vanne V_1 jusqu’au capteur de niveau H_1 , puis par le produit B avec la vanne V_2 jusqu’au capteur H_2 . L’ouverture d’une vanne est conditionnée par la fermeture des deux autres dans la loi de commande. Lorsque le capteur de niveau haut H_2 est actif, la vanne de vidange V_3 est ouverte

jusqu'au capteur de fin de cuve H_0 (la vidange est considérée comme immédiate dans cet exemple). Un nouveau cycle de production peut alors démarrer.

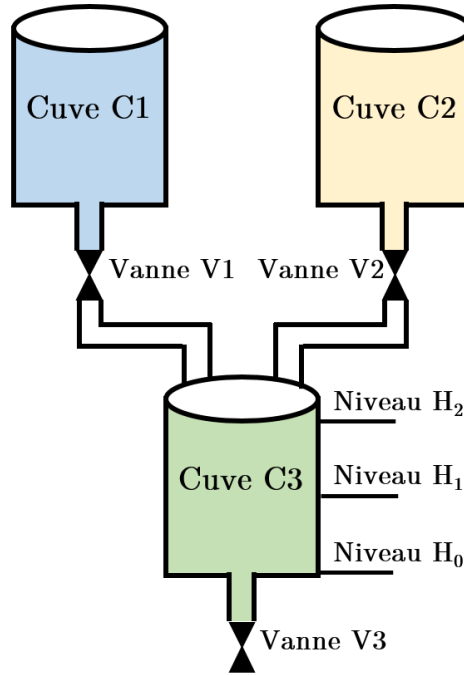


Figure 48 : Illustration du système à 3 cuves

Considérons que l'analyse de risques a identifié une seule zone critique pour la partie opérative (débordement de la cuve C_3) qui peut être atteinte lorsque le niveau de la cuve C_3 est supérieur au capteur de niveau haut H_2 . Afin de caractériser cet événement redouté dans notre modélisation, nous identifions les paramètres nécessaires et suffisants (Étape1) pour décrire notre système sous forme de :

- Vecteur des états $s \in S$ (21) qui représente le comportement normal du système puisque les défaillances ne sont pas modélisées. Il se compose des valeurs des capteurs de niveaux et les valeurs de commandes des vannes telles que dans l'équation (21) :

$$s = [\text{Valeurs Capteurs} ; \text{Valeurs Actionneurs}] = (H_i, V_1, V_2, V_3) \quad (21)$$

$$\text{Avec : } H_i \in \{0, 1, 2\}, V_1 \in \{0, 1\}, V_2 \in \{0, 1\}, V_3 \in \{0, 1\}$$

Dans l'équation (21), H_i indique quel capteur de niveau est actif (0 aucun capteur n'est activé, 1 le capteur H_1 est activé, 2 le capteur H_2 est activé). Pour ce système, l'ensemble des états possibles S est composé d'au plus 25 états et est noté S_{25}

- Vecteur actions $a \in A$ qui représente les actions pouvant être exécutées sur le système telles que $a = a_1, a_2, a_3$ avec $a_i \in \{0 \text{ ferme vanne } V_i, 1 \text{ ouvre vanne } V_i\}$ $i \in \{1..3\}$. Cet ensemble A est composé de 8 actions et est nommé par la suite A_8 .
- Ensemble des états atteignables $S_{Reach} \subset S$ qui contient 12 états dont 6 états optimaux (obtenus à l'aide du modèle de commande présenté en Figure 50) $S_{Opt} = \{[0 \ 0 \ 0 \ 0], [1 \ 1 \ 0 \ 0], [1 \ 0 \ 0 \ 0], [2 \ 0 \ 1 \ 0], [2 \ 0 \ 0 \ 0], [0 \ 0 \ 0 \ 1]\}$, 1 état interdit $S_{Pro} = s_{Débordement}$ et 6 états dangereux $S_{Dan} = S_{Reach}(S_{Opt} \cap S_{Pro}) = \{[1 \ 0 \ 1 \ 0], [2 \ 1 \ 1 \ 0], [0 \ 1 \ 0 \ 1], [0 \ 0 \ 1 \ 1], [0 \ 1 \ 1 \ 1], [2 \ 1 \ 0 \ 0]\}$. Cette modélisation ne

prend pas en compte les états intermédiaires et considère que l’exécution d’un ordre conduit directement à l’activation/désactivation du capteur de niveau associé. Lorsque la modélisation tient compte de cette hypothèse, nous atteignons 18 états sur les 25 états possibles avec 9 états optimaux, 8 dangereux et 1 interdit.

- Ensemble des états inatteignables $S_{Unreach} = S \setminus S_{Reach}$ qui contient donc 7 états.

Dans l’environnement de simulation Matlab, le dossier permettant de modéliser le modèle du filtre de commande, en Figure 49, se compose d’un script Matlab (*Automate_process_3cuves.m*) et de plusieurs fonctions Matlab détaillées en Annexe 3. Le script suit la méthodologie proposée dans la Partie 2. Dans un premier temps (Étape 1), l’ensemble des ordres et états possibles est calculé à l’aide de la fonction *all_comb.m* et stocké dans la structure *Param_System.mat*. Dans le même temps, les paramètres nécessaires pour caractériser les états du système sont renseignés, particulièrement l’état initial et l’état interdit.

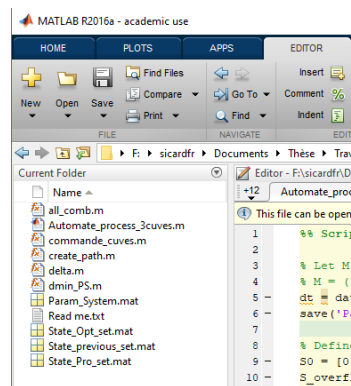


Figure 49 : Espace de travail pour générer le modèle du filtre de commande sous Matlab

Dans l’étape 2, une modélisation de la partie commande (Étape 2) est réalisée sur la base des spécifications de la loi de commande représentée en Figure 50. De la même manière, le modèle de la partie opérative (Étape 2) est construit en équation (22) à l’aide de l’automate M :

$$M = \{S_{25}, A_8, \delta, s_0, S_{final}\} \quad (22)$$

L’ensemble S_{25} représente l’ensemble des états possibles pour le système, A_8 est l’ensemble des ordres qui peuvent être appliqués sur le système (toutes les combinaisons possibles), δ est la fonction de transition qui permet de faire évoluer le système et dont les effets sont synthétisés dans le Tableau 8, s_0 est l’état initial du système défini par $s_0 = [0 \ 0 \ 0 \ 0]$, S_{final} représente l’ensemble des états finaux de notre automate et donc nos conditions d’arrêts (S_{pro} , s_0 et S_{prev}). Le Tableau 9 regroupe les conditions sur le couple État/Action qui conduisent à l’état interdit de débordement.

Tableau 8 : Récapitulatif des effets des actions sur le système à 3 cuves

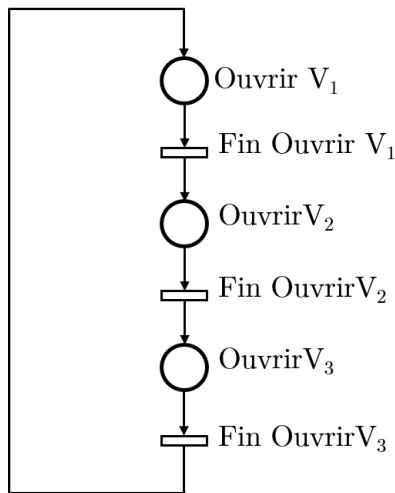
Ordre $a \in A_8$	État $s \in S_{25}$ du système avant l’action $a \in A_8$	Exception / Commentaire
Ouvrir la vanne V_1 ($a_1 = [1 \ 0 \ 0]$) ou V_2 ($a_2 = [0 \ 1 \ 0]$)	$s = [H_i \times \times 0]$	Lorsqu’une seule des vannes de remplissage est ouverte alors le niveau augmente d’une unité détectée par un capteur de niveau.

	$s' = \delta s, a_1$ ou $\delta s, a_2$ avec $s' = [H_i + 1 \times \times 0]$	
Ouvrir les vannes V_1 et V_2 $(a_4 = [1 \ 1 \ 0])$	$s = [H_i \times \times 0]$ $s' = \delta s, a_4$ avec $s' = [H_i + 2 \ 1 \ 1 \ 0]$	Lorsque les vannes V_1 et V_2 sont ouvertes alors la cuve C_3 se remplit plus vite et le niveau augmente de deux capteurs de niveau.
Ouvrir la vanne V_3 $(a_3 = [\times \times 1])$	$s = [H_i \times \times \times]$ $s' = \delta s, a_3$ avec $s' = [0 \times \times 1]$	Lorsque la vanne de vidange V_3 est ouverte alors la vidange de la cuve est déclenchée (peu importe l'activation des vannes de remplissage)
Aucun ordre $(a_8 = [0 \ 0 \ 0])$	$s = [H_i \times \times \times]$ $s' = \delta s, a_8 = [H_i \ 0 \ 0 \ 0]$	Lorsqu'aucun ordre d'ouverture de vannes n'est envoyé le système reste dans l'état dans lequel il était

Tableau 9 : Récapitulatif des conditions menant à l'état interdit de débordement (Système à 3 cuves)

État interdit S_{Pro}	Ordre $a \in A_8$	État $s \in S_{25}$ du système avant l'action $a \in A_8$
Débordement de la cuve C_3	Ouvrir la vanne $V_1 (a = [1 \ 0 \ 0])$ ou $V_2 (a = [0 \ 1 \ 0])$	Capteur H_2 activé : $s = [2 \times \times \times]$
	Ouvrir la vanne V_1 et $V_2 (a = [1 \ 1 \ 0])$	Capteurs H_1 ou H_2 activés : $s = [2 \times \times \times]$ ou $s = [1 \times \times \times]$

Les modèles de la partie commande et de la partie opérative sont alors générés dans l'environnement de simulation. Le code Matlab modélisant le modèle de commande est en Figure 50 tandis que celui de la partie opérative est présenté en Annexe 3. Sur la base de ces modèles, l'algorithme d'exploration des états atteignables est exécuté et représenté en Figure 51. Pour chaque nouvel état trouvé lors d'une itération, l'algorithme (algorithme 1) s'assure qu'il n'a pas été rencontré lors d'une précédente itération (appel à la structure *State_previous.mat*) puis, s'il s'agit d'un état interdit (appel à la structure *State_Pro_set.mat*). Si l'état n'appartient à aucune des deux précédentes structures, il est enregistré dans la structure *State_previous.mat* signifiant que l'état a été rencontré lors d'une précédente itération. Les états qui doivent être explorés pour une prochaine itération sont sauvegardés dans la matrice *State*. Lorsque cette dernière est vide alors tous les états atteignables ont été explorés (fin de la boucle *while*).



```
function [Ordre_new, Control_Step] = commande_cuves (Control_Step, CR)
% PLC code: SFC (Grafcet)
% Stage 1: Inputs Reading
CR;

% Stage 2: Logical Treatment (Sequential and combinational logic)
% ==> Transitions
if (CR(1,1)==0 && CR(1,2)==0 && CR(1,2)==0 && CR(1,3)==0 && Control_Step==0)
    Control_Step = 1; % Déclenchement V1
end

if (CR(1,1)==2 && CR(1,2)==1 && Control_Step==1)
    Control_Step = 2; % Arrêt V1
end

if (CR(1,1)==2 && CR(1,2)==0 && Control_Step==2)
    Control_Step = 3; % Déclenchement V2
end

if (CR(1,1)==3 && CR(1,3)==1 && Control_Step==3)
    Control_Step = 4; % Arrêt V2
end

if (CR(1,1)==3 && CR(1,3)==0 && Control_Step==4)
    Control_Step = 5; % Déclenchement V3
end

if (CR(1,1)==0 && CR(1,4)==1 && Control_Step==5)
    Control_Step = 6; % Arrêt V3
end

if (CR(1,1)==0 && CR(1,4)==0 && Control_Step==6)
    Control_Step = 1;
end

% ==> POST file
if (Control_Step==1)
    Ordre_new=[1 0 0];
end

if ((Control_Step==2) || (Control_Step==4) || (Control_Step==6))
    Ordre_new=[0 0 0];
end

if (Control_Step==3)
    Ordre_new=[0 1 0];
end

if (Control_Step==5)
    Ordre_new=[0 0 1];
end

if (Control_Step==0)
    Ordre_new=[0 0 0];
end

% State 3: Outputs Writing
Ordre_new;
end
```

Figure 50 : Illustration de la loi de commande du système à 3 cuves et fonction correspondante sur Matlab

```
Etats_suivant_inter = [];
Etats_next = [];
while (isempty(State)--0)
    for i=1:1:size(State,1)
        for j=1:1:size(Orders_Set,1)
            Action = Orders_Set(j,:);
            Etat_suivant = delta(State(i,:), Action);

            % Study of State
            % Already meet or Initial State?
            if (isequal(Etat_suivant, (-1*ones(1,4)))--0)
                load('State_previous_set.mat');
                for k=1:1:size(State_previous,1)
                    if (isequal(Etat_suivant, State_previous(k,:))--1)
                        Etat_suivant = [];
                    end
                end
            end
            if (isempty(Etat_suivant)--0)
                State_previous = [State_previous; Etat_suivant];
                save('State_previous_set', 'State_previous')
            end
            else
            % Prohibited state?
            if (isequal(Etat_suivant, (-1*ones(1,4)))--1)
                load('State_Pro_set');
                State_Pro = [State_Pro; State(i,:)];
                Action_pro = [Action_pro; Action];
                save('State_Pro_set', 'State_Pro', 'Action_pro')
                Etat_suivant = [];
            end
            end
            % Sauvegarde des états à étudier
            Etats_suivant_inter = [Etats_suivant_inter; Etat_suivant];
        end
    end
    Etats_next = [Etats_next ; Etats_suivant_inter];
    Etats_suivant_inter = [];

    end
    State = Etats_next;
    Etats_suivant_inter = [];
    Etats_next = [];
    Etat_suivant = [];
end
```

Figure 51 : Algorithme d'exploration des états (algorithme 1) sous Matlab

L'étape 2 de la méthodologie se termine par l'identification des états obtenus selon les types suivants : optimaux, dangereux et interdits. L'ensemble des états optimaux est obtenu en appliquant le modèle de la loi de commande au modèle de la partie opérative (Annexe 3). De plus, la distance de chaque état atteignable à l'état interdit le plus proche est calculée à l'aide de la fonction *dmin_PS* qui reprend l'algorithme 2 présenté dans Partie 2 Chapitre 5 - 2 et qui est présenté en Annexe 3.

Mis à part les systèmes à 3 et 5 cuves, la modélisation nécessite une programmation sous Eclipse en langage Java. En effet, les performances de Matlab ne permettent pas de faire converger l'algorithme dans un délai raisonnable. Le code Java est disponible en Annexe 3 et fait appel à une liste pour stocker les états à explorer, liste qui est plus facile à manipuler en termes de mémoire qu'une structure Matlab.

Pour finir, l'algorithme de génération du modèle du filtre de commande (Étape 2) est lancé sur la base des modèles des parties commande et opérative, ce qui conduit à l'obtention de 13 états atteignables et 69 contextes menant à des états interdits. Le temps de calcul pris par l'algorithme pour converger vers une exploration totale du système est de 26ms avec un ordinateur équipé d'un Intel Core i5-63000 (CPU 2,40 GHz 2,50 GHz – 8Go RAM). Le temps d'exécution est donné pour l'algorithme programmé sous Eclipse en langage Java (temps similaire sous Matlab) et représenté dans la Figure 52. L'étape 3 de la méthodologie est alors appliquée, elle correspond à l'exploitation du filtre et aux mécanismes de détection.

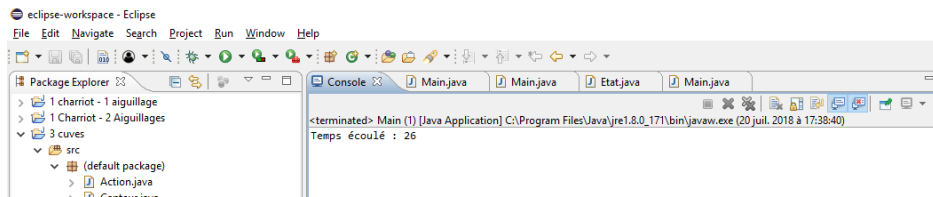


Figure 52 : Capture d'écran du temps de calcul pour obtenir le modèle du filtre de commande sous Eclipse

2.2. Système à 5 cuves

L'exemple qui suit est inspiré du premier système à trois cuves et est composé de 5 cuves comme illustré en Figure 53. Le système est donc composé de 5 cuves, trois d'entre elles C_1 , C_2 et C_3 sont considérées comme étant de capacités infinies et contiennent respectivement le produit A, le produit B et le produit D. Les deux premières cuves déversent leurs produits dans un réservoir de mélange C_3 afin de produire le produit C. L'étape de remplissage est déclenchée par l'ouverture des vannes de remplissage V_1 et V_2 . L'étape de vidange est réalisée par l'ouverture de la vanne V_3 . Les cuves C_3 et C_4 déversent leurs produits dans la cuve de mélange C_5 pour aboutir au produit E. L'étape de remplissage est déclenchée par l'ouverture de la vanne de remplissage V_4 et de la vanne de vidange V_3 . L'étape de vidange est réalisée par l'ouverture de la vanne V_5 . Des capteurs de niveau sont placés dans chacun des réservoirs et permettent de connaître la quantité de produit dans les cuves de mélange. L'appartenance du capteur à la cuve dont il fait la mesure est indiquée en exposant dans la notation adoptée. Sept capteurs de niveau sont utilisés dans ce système :

- $H_0^{C_4}$ et $H_0^{C_5}$ sont les capteurs de fin de cuve. Ils indiquent un remplissage lorsqu’ils prennent la valeur 1 et une vidange totale de la cuve quand la valeur est nulle.
- $H_1^{C_4}$, $H_1^{C_5}$ et $H_2^{C_5}$ sont localisés à un niveau intermédiaire des cuves de mélange.
- $H_2^{C_4}$ et $H_3^{C_5}$ sont les capteurs de niveau haut des cuves C_3 et C_5 .

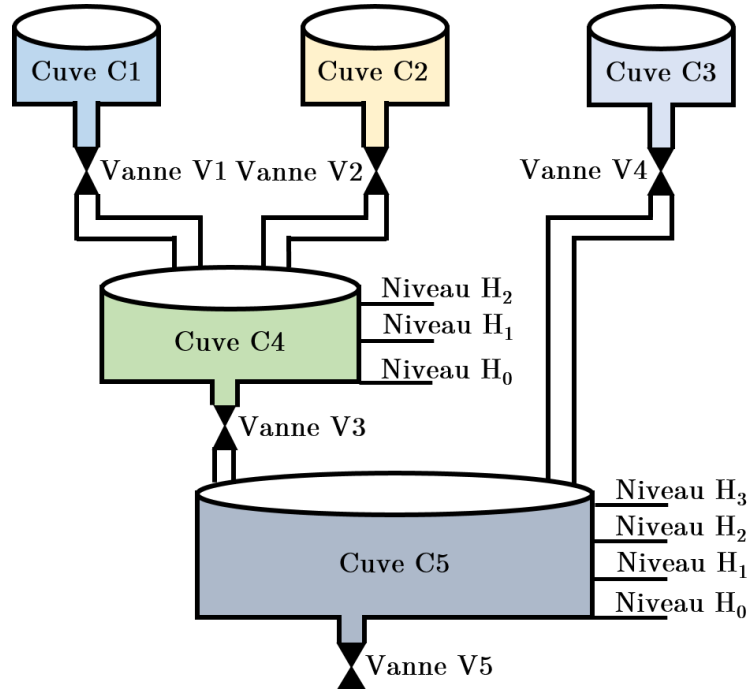


Figure 53 : Illustration du système à 5 cuves

Les spécifications de la loi de commande imposent de démarrer un cycle de production avec les cuves de mélange vides donc $H_0^{C_4}$ et $H_0^{C_5}$ nuls. Ensuite, un premier remplissage de la cuve C_3 est réalisé par le produit A en ouvrant la vanne V_1 jusqu’au capteur de niveau $H_1^{C_4}$, puis, par le produit B avec la vanne V_2 jusqu’au capteur $H_2^{C_4}$. Comme pour l’exemple précédent, l’ouverture d’une vanne est conditionnée par la fermeture des autres dans la loi de commande. Lorsque le capteur de niveau haut $H_2^{C_4}$ est actif, la vanne de vidange V_3 est ouverte jusqu’au capteur $H_1^{C_5}$ puis les vannes V_3 et V_4 sont ouvertes jusqu’à l’activation du capteur $H_3^{C_5}$. La vidange de la cuve C_5 est ensuite demandée par l’ouverture de la vanne V_5 jusqu’au capteur de fin de cuve $H_3^{C_5}$. Comme pour l’exemple précédent, les vidanges sont considérées comme instantanées. Un nouveau cycle de production peut alors démarrer.

Considérons que l’analyse de risques a permis d’identifier les zones critiques pour la partie opérative (débordement des cuves C_4 et C_5) qui, pour ce système, peuvent être atteintes lorsque les niveaux des cuves C_3 ou C_5 sont supérieurs aux capteurs de niveau haut $H_2^{C_4}$ ou $H_3^{C_5}$. Afin de caractériser ces événements redoutés dans notre modélisation, nous identifions les paramètres nécessaires et suffisants (Étape 1) pour décrire notre système. Cette étape s’effectue à l’aide d’un expert qui détermine, d’une part, les actions qui agissent sur le niveau de la cuve et, d’autre part les capteurs qui indiquent ce niveau afin de pouvoir caractériser l’état de débordement. Ainsi, à l’issue de l’étape 1, nous avons :

- Vecteur des états $s \in S$ qui représente le comportement normal du système puisque les défaillances ne sont pas modélisées. Il se compose des valeurs des capteurs de niveau et les valeurs de commandes des vannes telles que dans l'équation (23) :

$$s = [\text{Valeurs Capteurs} ; \text{Valeurs Actionneurs}] = (H_i^{C_4}, H_j^{C_5}, V_1, V_2, V_3) \quad (23)$$

$$\text{Avec : } H_i^{C_4} \in \{0..2\}, H_j^{C_5} \in \{0..3\}, V_1 \in \{0, 1\}, V_2 \in \{0, 1\}, V_3 \in \{0, 1\}$$

Dans l'équation (23), $H_i^{C_4}$ indique quel capteur de niveau de la cuve C_3 est actif (0 aucun capteur n'est activé, 1 le capteur H_1 est activé, 2 le capteur H_2 est activé), $H_j^{C_5}$ est l'équivalent pour la cuve C_5 . Pour ce système, l'ensemble des états possibles « S » est composé d'au plus 386 états et est noté S_{386}

- Vecteur actions $a \in A$ qui représente les actions pouvant être exécutées sur le système telles que $a = a_1, a_2, a_3, a_4, a_5$ avec $a_i \in \{0 \text{ ferme vanne } V_i, 1 \text{ ouvre vanne } V_i\} i \in \{1..5\}$. Cet ensemble A est composé de 32 actions et est nommé dans la suite A_{32} .
- Ensemble des états atteignables $S_{Reach} \subset S$ qui contient 209 états dont 11 états optimaux (obtenus à l'aide du modèle de commande présenté en Figure 54), 2 états interdits et 196 états dangereux $S_{Dan} = S_{Reach}(S_{Opt} \cap S_{Pro})$.
- Ensemble des états inatteignables $S_{Unreach} = S \setminus S_{Reach}$ qui contient donc 175 états.

Un modèle de la partie commande (Étape 2) est réalisé sur la base des spécifications de la loi de commande et est représenté en Figure 54. De la même manière, le modèle de la partie opérative (Étape 2) est construit par l'équation (24) à l'aide de notre automate M :

$$M = \{S_{209}, A_{32}, \delta, s_0, S_{final}\} \quad (24)$$

L'ensemble S_{209} représente l'ensemble des états atteignables pour le système, A_{32} est l'ensemble des ordres qui peuvent être appliqués sur le système (toutes les combinaisons possibles), δ est la fonction de transition qui permet de faire évoluer le système et dont les effets sont synthétisés sur le Tableau 10, s_0 est l'état initial du système défini par $s_0 = [0 \ 0 \ 0 \ 0 \ 0 \ 0]$, S_{final} représente l'ensemble des états finaux de notre automate et donc nos conditions d'arrêts (S_{pro} , s_0 et S_{prev}). Le Tableau 11 regroupe les conditions sur le couple État/Action qui conduisent à l'état interdit de débordement.

Tableau 10 : Récapitulatif des effets des actions sur le système à 5 cuves

Ordre $a \in A_{32}$	État $s \in S_{209}$ du système avant l'action $a \in A_{32}$	Exception / Commentaire
Ouvrir la vanne V_1 ($a_1 = [1 \ 0 \ 0 \times \times]$) ou V_2 ($a_2 = [0 \ 1 \ 0 \times \times]$)	$s = [H_i^{C_4} H_j^{C_5} \times \times 0 \times \times]$ $s' = \delta s, a_1 \text{ ou } \delta s, a_2$ avec $s' = [H_i^{C_4} + 1 \ H_j^{C_5} \times \times 0 \times \times]$	Lorsqu'une seule des vannes de remplissage est ouverte alors le niveau $H_i^{C_4}$ augmente d'une unité détectée par un capteur de niveau et spécifiée par son indice.

Ouvrir les vannes V_1 et V_2 ($a_6 = [1 \ 1 \ 0 \times \times]$)	$s = [H_i^{C_4} H_j^{C_5} \times \times 0 \times \times]$ $s' = \delta \ s, a_6$ avec $s' = [H_i^{C_4} + 2 \ H_j^{C_5} \ 1 \ 1 \ 0 \times \times]$	Lorsque les vannes V_1 et V_2 sont ouvertes alors la cuve C_3 se remplit plus vite et le niveau $H_i^{C_4}$ augmente de deux capteurs de niveau.
Ouvrir la vanne V_3 ($a_3 = [\times \times 1 \times \times]$)	$s = [H_i^{C_4} H_j^{C_5} \times \times \times \times]$ $s' = \delta \ s, a_3$ avec $s' =$ $\begin{cases} [H_i^{C_4} - 1 \ H_j^{C_5} + 1 \ \times \times 1 \times \times] \text{ si } H_i^{C_4} > 0 \\ [H_i^{C_4} \ H_j^{C_5} \ \times \times 1 \times \times] \text{ si } H_i^{C_4} = 0 \end{cases}$	Lorsque la vanne de vidange V_3 est ouverte alors la vidange de la cuve C_3 est déclenchée (peu importe l’activation des vannes de remplissage V_1 et V_2).
Ouvrir la vanne V_4 ($a_4 = [\times \times \times 1 \times]$)	$s = [H_i^{C_4} H_j^{C_5} \times \times \times \times 0]$ $s' = \delta \ s, a_4$ avec $s' = [H_i^{C_4} \ H_j^{C_5} + 1 \ \times \times \times 1 \times]$	Lorsque la vanne V_4 est ouverte alors le niveau $H_j^{C_5}$ augmente dans la cuve C_5 .
Ouvrir la vanne V_5 ($a_5 = [\times \times \times \times 1]$)	$s = [H_i^{C_4} H_j^{C_5} \times \times \times \times 0]$ $s' = \delta \ s, a_5$ avec $s' =$ $\begin{cases} [H_i^{C_4} \ H_j^{C_5} - 1 \ \times \times \times \times 1] \text{ si } H_j^{C_5} > 0 \\ [H_i^{C_4} \ H_j^{C_5} \ \times \times \times \times 1] \text{ si } H_j^{C_5} = 0 \end{cases}$	Lorsque la vanne V_5 est ouverte la vidange de la cuve C_5 est déclenchée.
Aucun ordre ($a_{32} = [0 \ 0 \ 0 \ 0 \ 0]$)	$s = [H_i^{C_4} H_j^{C_5} \times \times \times \times \times]$ $s' = \delta \ s, a_{32} = [H_i^{C_4} H_j^{C_5} \ 0 \ 0 \ 0 \ 0 \ 0]$	Lorsqu’aucun ordre d’ouverture de vannes n’est envoyé le système reste dans l’état dans lequel il était.

Nous remarquerons que l’ensemble de ces effets sont cumulables les uns avec les autres. Ainsi, si l’ordre $[1 \ 0 \ 1 \ 0 \ 0]$ est envoyé à la PO alors la cuve C_5 sera remplie d’une unité.

Pour conclure l’étape 2, l’algorithme de génération du modèle du filtre de commande est exécuté sur la base des modèles construits ce qui conduit à l’obtention de 209 états atteignables et 2580 contextes menant à des états interdits. Le temps de calcul pris par l’algorithme (exécuté en hors ligne) est de 87 ms avec un ordinateur équipé d’un processeur Intel Core i5-63000 (CPU 2,40 GHz 2,50 GHz – 8Go RAM). Le temps est donné pour l’algorithme programmé sous Eclipse en langage Java. Ici, une différence significative est observée par rapport à l’algorithme programmé sous Matlab puisque le temps d’exécution est alors de 38 secondes. L’étape 3 de la méthodologie est alors appliquée et correspond à l’exploitation du filtre et de la stratégie de détection.

Tableau 11 : Récapitulatif des conditions menant aux états interdits de débordement (Système à 5 cuves)

État interdit S_{Pro}	Ordre $a \in A_{32}$	État $s \in S_{209}$ du système avant l’action $a \in A_{32}$
	Ouvrir la vanne V_1 ($a = [1 \ 0 \ 0 \times \times]$) ou V_2 ($a = [0 \ 1 \ 0 \times \times]$)	Capteur $H_2^{C_4}$ activé : $s = [2 \ \times \times \times \times \times \times]$

Débordement de la cuve C_3	Ouvrir la vanne V_1 et V_2 ($[1 \ 1 \ 0 \times \times]$)	Capteurs $H_1^{C_4}$ ou $H_2^{C_4}$ activés : $s = [2 \times \times \times \times \times \times]$ ou $s = [1 \times \times \times \times \times \times]$
Débordement de la cuve C_5	Ouvrir la vanne V_3 ($a = [\times \times 1 \ 0 \times]$) avec $H_i^{C_4} > 1$, sinon ouvrir la vanne V_1 ($a = [1 \ 0 \ 0 \times \times]$), V_2 ($a = [0 \ 1 \ 0 \times \times]$) ou V_4 ($a = [\times \times 0 \ 1 \times \times]$)	Capteur $H_3^{C_5}$ activé : $s = [3 \times \times \times \times \times \times]$
	Ouvrir la vanne V_1 et V_2 ($a = [1 \ 1 \ 0 \times \times]$)	Capteurs $H_2^{C_5}$ ou $H_3^{C_5}$ activés : $s = [3 \times \times \times \times \times \times]$ ou $s = [2 \times \times \times \times \times \times]$

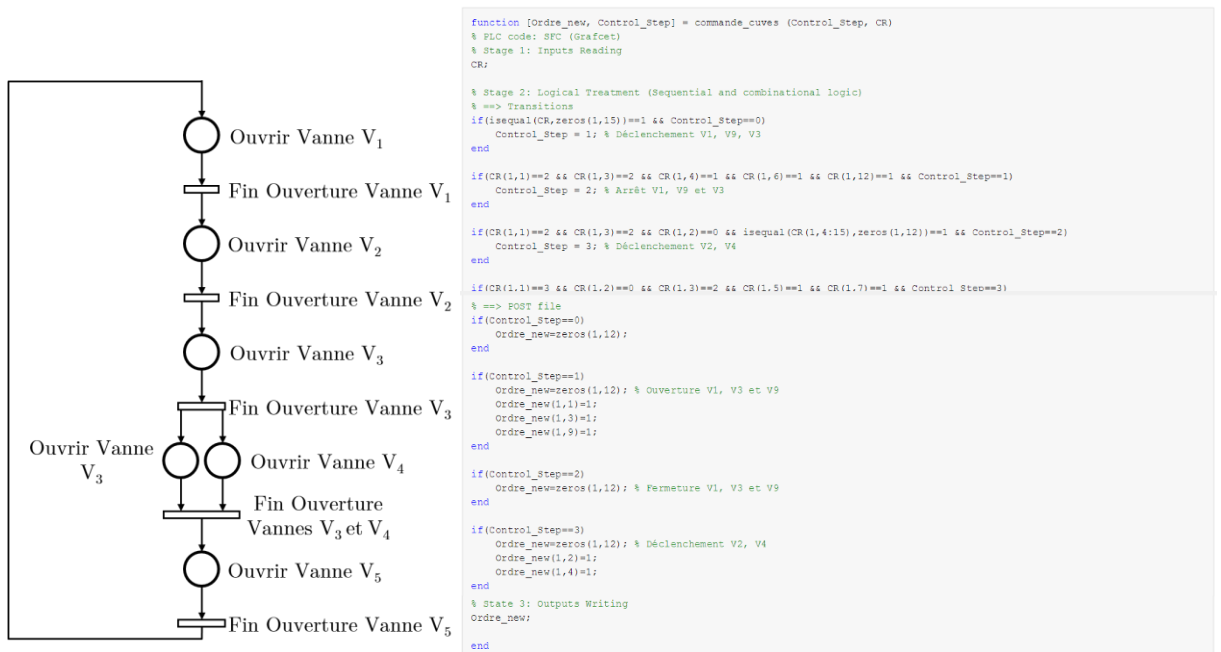


Figure 54 : Illustration de la loi de commande du système à 5 cuves et fonction correspondante sous Matlab

Afin d'alléger la lecture du manuscrit, nous avons mis à la disposition du lecteur en Annexe 2 la présentation des deux autres systèmes simulés, à savoir le système à 6 cuves et celui basé sur un chariot et un aiguillage.

3. Analyse des scénarii d'attaques

Les quatre scénarii d'attaques hérités des profils d'attaquants établis dans le paragraphe 1.2 ont été testés sur l'ensemble des systèmes présentés précédemment. La détection d'anomalies étant similaire d'un exemple à l'autre, nous avons choisi de représenter uniquement les résultats sur le système à 5 cuves qui permettent de comprendre les mécanismes de détection utilisés tout en présentant un niveau de complexité (en termes de nombre de paramètres) suffisamment élevé. Dans l'environnement de simulation, les filtres sont localisés comme sur la Figure 55.



Figure 55 : Capture d’écran du code Matlab permettant de lancer les simulations avec le filtre de commande

3.1. Comportement normal du système

Le comportement normal du système est caractérisé par les distances $D(a|A_{Opt})$ et $D(s|S_{Opt})$ nulles sur le filtre de commande comme illustré en Figure 57. Elles indiquent que les ordres analysés correspondent au modèle de commande et que la partie opérative évolue en suivant les états optimaux du système, c’est-à-dire ceux qui respectent les contraintes combinatoires. La distance aux états interdits $D(s|S_{Pro})$ évolue au fur et à mesure des évolutions du système. Lorsque la cuve C_4 se remplit, la trajectoire $T_{état}(s|S_{Pro})$ se rapproche de l’état interdit S_{Pro}^1 tandis que lorsqu’elle s’en éloigne au profit de S_{Pro}^2 le contenu de la cuve C_4 se déverse dans C_5 . Toutefois, comme les distances $D(a|A_{Opt})$ et $D(s|S_{Opt})$ sont nulles, alors, un rapprochement avec un état interdit est normal et compris dans les spécifications de la commande. Le mécanisme de calcul de la distance est situé en Figure 56. L’affichage de la Figure 57 est réalisé sur la base de ces calculs (par appel à la fonction *trajectoire* qui permet la représentation graphique de ces résultats). Le calcul est identique à celui implanté dans le filtre de CR représenté en Figure 58.

```

% 1) Comparaison entre l'ordre attendu et l'ordre reçu
if (isempty(Order_pre)==1)
else
    for i=1:size(Order_pre,1)
        if (isequal(Order, Order_pre(i,:))==1)
            alert = [alert 0];
        else
            alert = [alert 1];
        end
    end
end

% 2) Calcul des ordres possibles
Etat_suitant = delta(State_Sys, Order);
if (isequal(Etat_suitant, -1*ones(1,4))==0)
    Action = Order;
else
    Action = zeros(size(Order,1), size(Order,2));
end

% 3) Calcul de l'ordre optimal, state optimal et état interdit le plus
% proche

% Chargement des derniers éléments
load('Dist_Filt_Cmd')
if (isempty(d_opt)==1)
    d_opt_old_inter = 0;
    ds_opt_old_inter = 0;
else
    d_opt_old_inter = [];
    ds_opt_old_inter = [];
    d_opt_old_inter = d_opt(1,end);
    ds_opt_old_inter = ds_opt(1,end);
end

% Ordre optimal
load('State_Opt_set.mat')
Ordre_Opt_inter = [Ordre_Opt(1:end,:)];
if (isequal(Ordre_Opt_inter(Synch_PLC_order,:), Order)==1)

% Ordre optimal
load('State_Opt_set.mat')
Ordre_Opt_inter = [Ordre_Opt(1:end,:)];
if (isequal(Ordre_Opt_inter(Synch_PLC_order,:), Order)==1)
    d_opt = d_opt_old_inter;
else
    d_opt = d_opt_old_inter + 1;
end

% Etat optimal
State_Opt_inter = [State_Opt(1:end,:)];
if (isequal(State_Opt_inter(Synch_PLC_state,:), Etat_suitant)==1)
    ds_opt = 0;
else
    ds_opt = ds_opt_old_inter + 1;
end

% Etat interdit
top = 0;
load('State_previous_set.mat')
for i=1:size(State_Path_crit,1)
    if (isequal(State_Path_crit(i,2:5), Etat_suitant)==1)
        index = State_Path_crit(i,1);
        ds_int = dmin(index,:);
        top = 1;
    end
end

if (isequal(Etat_suitant, -1*ones(1,4))==1 && top == 0)
    ds_int = 0;
end

% Créer un bloc pour se rappeler des distances précédentes + calcul des
% autres.
d_opt_old = [d_opt_old d_opt];
ds_opt_old = [ds_opt_old ds_opt];
ds_int_old = [ds_int_old ds_int];

d_opt = d_opt_old;
ds_opt = ds_opt_old;
ds_int = ds_int_old;

if (Synch_PLC < 7)
    Synch_PLC = Synch_PLC + 1;
else
    Synch_PLC = 1;
end

save('Dist_Filt_Cmd.mat', 'd_opt', 'ds_opt', 'ds_int', 'd_opt_old', 'ds_
trajectoire(1)

```

Figure 56 : Capture d'écran de l'algorithme de détection dans le filtre de commande

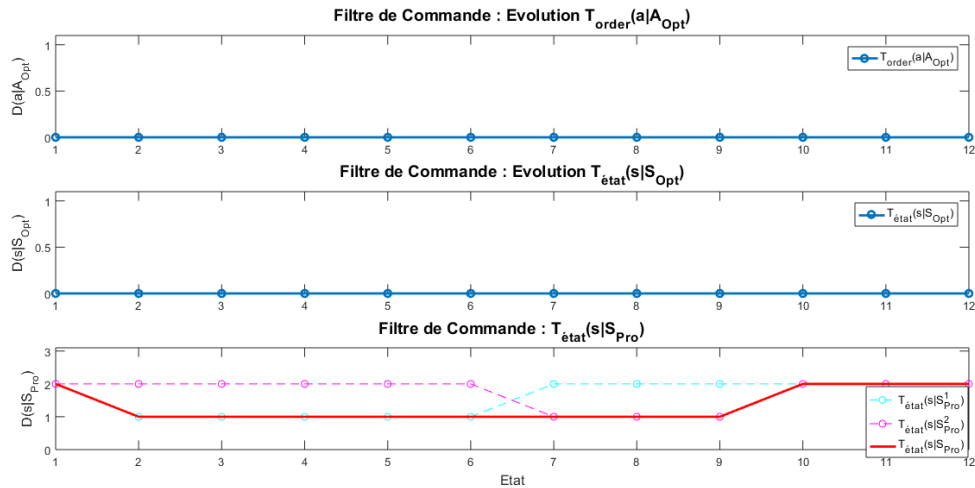


Figure 57 : Trajectoires du comportement normal du système à 5 cuves (filtre de commande)

Un constat similaire peut être fait par l'observation des trajectoires du filtre de comptes rendus en Figure 58. Ce dernier vérifie que les états mesurés par les capteurs correspondent à ceux attendus. La trajectoire $T_{\text{état}}(s|S_{Pro})$ évolue similairement à celle calculée par le filtre de commande.

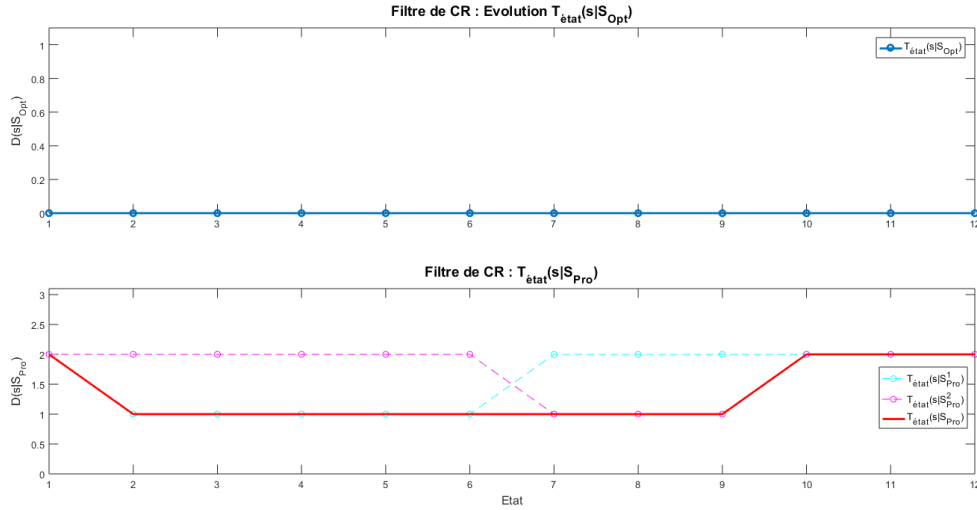


Figure 58 : Trajectoires du comportement normal du système à 5 cuves (filtre de comptes rendus)

3.2. Scénario 1 : Attaque sur l'ICS et son environnement

Dans ce scénario d'attaque, il est considéré que l'attaquant a pour objectif de dégrader le système et son environnement en ayant peu de connaissances sur l'ICS. Ainsi, la stratégie mise en place dans cette approche permet de détecter des anomalies et de prévenir la destruction du système. La Figure 59 illustre une partie de l'algorithme permettant de détecter l'apparition d'une anomalie ou d'un contexte. Les résultats de cette détection sont représentés en Figure 60.

```
load ('State_Pro_set.mat')
for i=1:1:size(Context_uniq,1)
    if (isequal(State_Sys,Context_uniq(i,1:size(State,1)))==1) && (isequal(Order,Context_uniq(i,size(State,1)+1:end))==1)
        Alert = 1;
        disp('Contexte menant vers Etat Interdit')
    else
        Alert = 0;
    end
end

if (isempty(Order_pre)==1)
else
    for i=1:1:size(Order_pre,1)
        if (isequal(Order, Order_pre(i,:))==1)
            alert = [alert 0];
        else
            alert = [alert 1];
            disp('Eloignement Trajectoire Optimale')
        end
    end
end
end
```

Figure 59 : Capture d'écran d'une partie de l'algorithme sous Matlab permettant de détecter l'attaque du scénario 1

Sur la Figure 60, deux états sont intéressants dans l'analyse de l'attaque : l'état n°3 et n°8. Le premier correspond à l'ouverture de la vanne V_2 au lieu de la fermeture de toutes les vannes à la suite du remplissage de C_4 par V_1 . Le filtre de commande détecte une anomalie puisque l'ordre analysé ne correspond pas aux modèles de commande et de la partie opérative ($D(a|A_{Opt})$ et $D(s|S_{Opt})$ égaux à 1). Ainsi, le manque de connaissances de la part de l'attaquant laisse des traces de l'intrusion que nos mécanismes détectent. Toutefois, comme l'anomalie ne conduit pas à un état interdit alors l'action n'est pas bloquée et une alerte est envoyée à l'opérateur. En revanche, le deuxième état où se produit l'attaque (état n°8) conduit le système dans un état interdit ($D(s|S_{Pro}) = 0$). Cette anomalie est causée par l'ouverture de la vanne de remplissage V_4 au lieu de la vanne de vidange V_5 ce qui conduit à l'état interdit S_{Pro}^2 , caractéristique du

débordement de la cuve C_5 . Comme pour la précédente anomalie, les distances ($D(a|A_{Opt})$ et $D(s|S_{Opt})$) montrent que les caractéristiques combinatoires ne sont pas respectées, c'est pourquoi le filtre de commande bloque l'ordre analysé. L'état interdit n'est donc pas atteint.

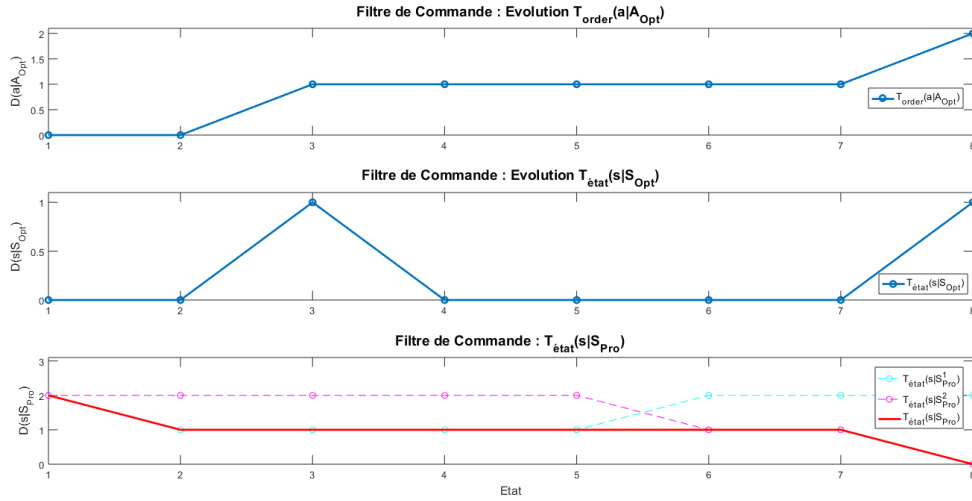


Figure 60 : Trajectoires caractéristiques d'une attaque du scénario 1 (filtre de commande)

3.3. Scénario 2 : Attaque sur la qualité

Ce scénario d'attaque met en jeu un attaquant ayant plus de connaissances sur l'ICS ciblé que le profil précédent et ayant pour objectif de dégrader la qualité du produit issu du système considéré. Pour ce faire, il peut jouer sur la violation des contraintes combinatoires (comme l'état n°3 dans le scénario 1) ou des contraintes temporelles. Nous considérerons ces dernières pour tester la stratégie de détection et l'exploitation des résultats représentée en Figure 62.

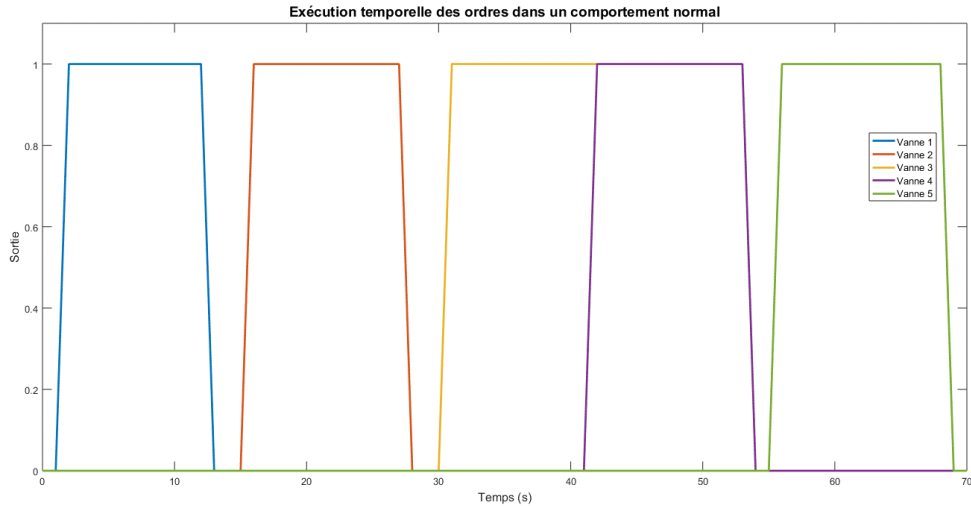


Figure 61 : Représentation de l'exécution temporelle des ordres du système à 5 cuves dans le cas d'un comportement normal

La Figure 61 illustre l'exécution au niveau temporel des ordres appliqués au système à 5 cuves. Les valeurs sont issues de l'exploitation des spécifications temporelles du système. Nous remarquons que lors d'un cycle de production normal, l'activation des ordres correspond bien aux contraintes combinatoires.

La Figure 62 représente l’évolution, au niveau temporel, des capteurs représentatifs de l’état de la partie opérative. Ainsi, à la suite de la l’ouverture de la vanne V_1 à $t = 2s$, le capteur de niveau $H_0^{C_4}$ est activé ($t = 3s$). Ainsi, pour chacun des capteurs, les fenêtres temporelles d’apparition et de disparition ont été représentées, l’événement devant être maintenu entre ces deux dernières. L’attaque réalisée par l’attaquant (courbe rouge en pointillés sur $H_1^{C_4}$) consiste à intercepter les données des capteurs pour ensuite modifier la valeur du capteur de niveau $H_1^{C_4}$ afin d’induire une action de l’API qui dégradera la qualité du produit final. Ainsi, sur la Figure 62, nous pouvons remarquer que dès l’activation du capteur $H_0^{C_4}$, le capteur du niveau supérieur $H_1^{C_4}$ est également activé pour une courte durée. Ainsi, les proportions des produits A et B dans la cuve C_4 ne sont pas respectées ce qui dégradera la qualité du produit C. Les fenêtres temporelles associées à ce capteur ne sont plus respectées ainsi que la durée moyenne d’exécution de l’événement qui est inférieure dans le cadre de cette attaque aux spécifications. Ainsi, ces indicateurs permettent d’envoyer des alertes aux opérateurs afin de les prévenir d’une dérive. Dans ce scénario, les indicateurs de temporalité ont permis de mettre en évidence une anomalie ne respectant pas les contraintes liées au temps dans le système protégé.

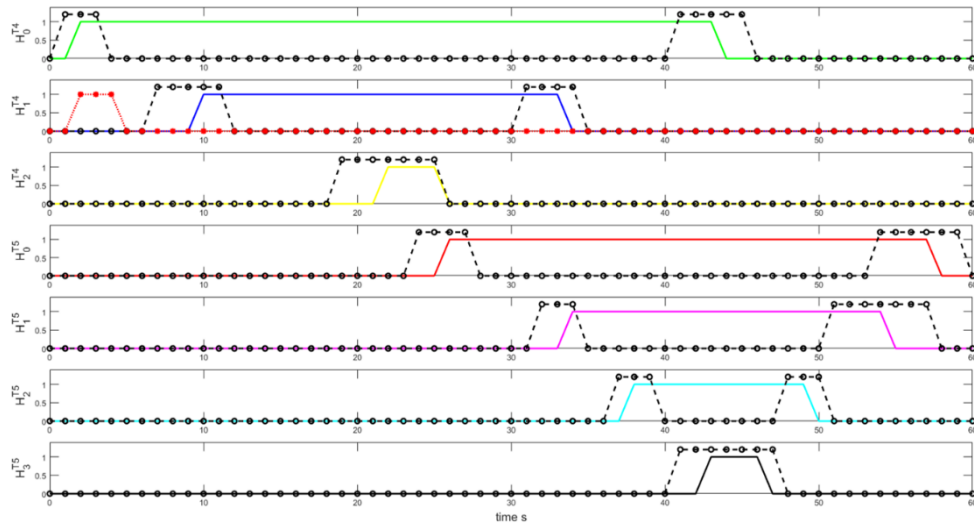


Figure 62 : Représentation de l’exécution temporelle des comptes rendus du système à 5 cuves lors d’un cycle de production normal (en couleur, fenêtres temporelles en noir) et d’une attaque (en rouge)

Une partie de l’algorithme de détection basée sur les contraintes temporelles est présentée en Figure 63. Pour chaque actionneur et capteur du système, la date d’apparition de l’événement est indiquée ce qui aboutit à la construction de gabarits devant être respectés. La durée moyenne d’activation est également calculée.


```

timer_comande(1,1:2)=0;
timer_comande(1,3:13)=1;
timer_comande(1,14:71)=0;

timer_comande(2,1:16)=0;
timer_comande(2,17:28)=1;
timer_comande(2,29:71)=0;

timer_comande(3,1:31)=0;
timer_comande(3,32:54)=1;
timer_comande(3,55:71)=0;

timer_comande(4,1:42)=0;
timer_comande(4,43:54)=1;
timer_comande(4,55:71)=0;

timer_comande(5,1:56)=0;
timer_comande(5,57:69)=1;
timer_comande(5,70:71)=0;

if(T_moy <= Timer_comande_min || T_moy >= Timer_comande_max)
    alert = [alert 0];
else
    alert = [alert 1];
end

T_moy_global = [T_moy_global T_moy];

```

Figure 63 : Capture d'écran de l'algorithme de détection temporelle

3.4. Scénario 3 : Attaque discrète

Le profil d'attaquant de ce scénario possède une connaissance parfaite de l'ICS ciblé. Son objectif de dégrader discrètement et de manière ponctuelle la qualité, ou alors, de mettre en danger le système vis-à-vis de la sécurité des biens et des personnes. Ainsi, la personne malveillante réalisera des actions dégradant l'ICS très localisées dans le temps et de manière aléatoire afin de faire passer la malveillance dans le bruit de faux positifs liés à la variabilité des ICSs. La Figure 64 représente le comportement de la stratégie de détection face à ce type d'attaque (les algorithmes de détection présentés précédemment sont mis à profit dans ce scénario).

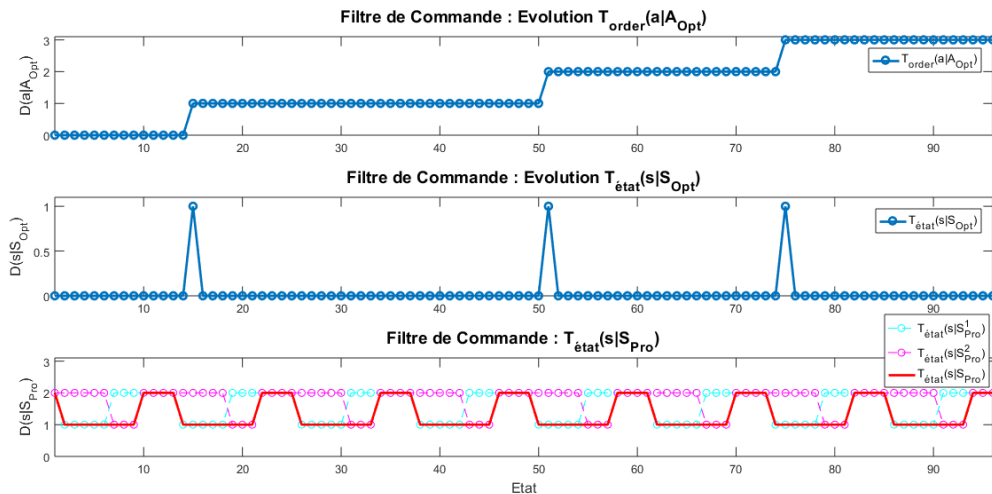


Figure 64 : Représentation des trajectoires du filtre de commande pour le système à 5 cuves dans le cas d'une attaque discrète (faible occurrence d'anomalies)

Des anomalies, au nombre de trois, apparaissent de manière ponctuelle dans les trajectoires $T_{order}(a|A_{Opt})$ et $T_{état}(s|S_{Opt})$ en l'état n°15, 51 et 75 (sur 8 cycles de production) en Figure 64.

Ces anomalies correspondent à un retard dans la fermeture de la vanne V_2 ce qui modifie les proportions du mélange pour produire le composant C. Nos mécanismes détectent ces anomalies et les signalent aux opérateurs. Cependant, elles restent limitées dans le temps, n’impactent pas gravement la production (pas de rapprochement à des états critiques) et sont complètement aléatoires. Dans le cadre de ce scénario, la partie détection ne pose aucun doute, au contraire de celle du « pré-diagnostic » pour orienter vers une attaque.

3.5. Scénario 4 : Attaque sur les équipements

Ce dernier scénario représente un attaquant ayant là encore, une connaissance parfaite du système attaqué ce qui lui a permis d’identifier un ensemble d’actionneurs à cibler afin de mettre en échec la production. Pour ce faire, il provoque des sur-sollicitations de ces équipements critiques afin de les user plus rapidement, d’obliger des maintenances plus fréquentes (arrêtant ponctuellement le système) voire de les casser.

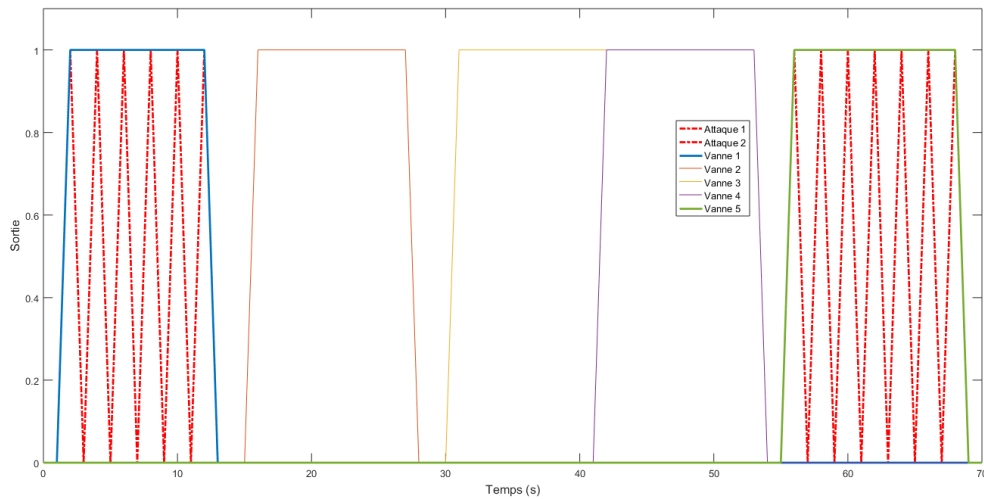


Figure 65 : Représentation de l’exécution temporelle des ordres du système à 5 cuves dans le cas d’une attaque sur les équipements

Dans le système à 5 cuves considéré, seules les vannes de remplissage V_1 et de vidange V_5 sont prises pour cibles. Lorsque l’ordre d’ouverture est envoyé à la partie opérative, ce dernier n’est pas maintenu durant toute l’action. Les conséquences sont l’ouverture et la fermeture de la vanne jusqu’à ce qu’une autre action soit décidée par l’API. Ce phénomène est détecté par l’indicateur de sur-sollicitation qui a été développé à cet effet, ainsi que par les contraintes temporelles puisque les fenêtres temporelles spécifiant le déclenchement des actions ne sont plus respectées. Une alerte est alors envoyée aux opérateurs pour les avertir des observations réalisées par les mécanismes implantés dans les filtres de commande et de comptes rendus.

4. Tableau récapitulatif

Tableau 12 : Récapitulatif des systèmes simulés lors des tests de l'approche S.A.F.E. (Intel Core i5-63000 - CPU 2,40 GHz 2,50 GHz – 8Go RAM)

Système	3 cuves	5 cuves	6 cuves	1 chariot – 1 aiguillage
Nombre d'actionneurs	3	5	12	4
Nombre de capteurs	3	7	10	7
Nombre d'actions possibles	8	32	4096	16
Nombre d'états possibles	25	386	147 459	192
États atteignables	18	209	46 376	113
États	9	11	13	18
• Optimaux,	8	196	46360	94
• Dangereux,	1	2	3	1
• Interdits,				
Temps de calcul de l'algorithme d'exploration des états, de calcul des distances et durée totale (Hors ligne)	Eclipse : ~10 ms ~16 ms ~26 ms Matlab : ~550 ms ~407 ms ~957 ms	Eclipse : ~10 ms ~77 ms ~87 ms Matlab : ~24600 ms ~13400 ms ~38000 ms	Eclipse : ~23150 ms ~820 ms ~23970 ms Matlab : Ø	Eclipse : ~60 ms ~70 ms ~130 ms Matlab : ~7000 ms ~ Ø ¹ ~ Ø
Références	(Sicard, Zamaï, et Flaus 2017a, 2017b, 2018a)	(Sicard, Zamaï, et Flaus 2018b)	-	-

¹ Algorithme de calcul de distance stoppé au bout de 20 min (Matlab)

5. Bilan du chapitre

Dans ce chapitre, l’approche a pu être testée sur des exemples inspirés de systèmes industriels réels dans un environnement de simulation. Dans un premier temps, plusieurs profils d’attaquants ont été élaborés sur la base de l’étude bibliographique réalisée dans la Partie 1 qui vont de l’attaquant souhaitant dégrader l’ICS et son environnement coûte que coûte jusqu’à la personne malveillante qui souhaite dégrader très ponctuellement le système afin de rester à l’abri d’une détection. Ensuite, les différents modèles utilisés ont été présentés et la méthodologie de conception des filtres a été alors appliquée. La variété des exemples envisagés pour tester l’approche a mis en avant la généricité du processus de synthèse des filtres de commande et comptes rendus qui a été détaillée étape par étape.

Enfin, les différents scénarii, correspondant aux profils d’attaquants, ont été appliqués sur les exemples de simulation. Pour une meilleure lecture, le système 5 cuves a été retenu pour présenter les résultats puisqu’il représente un bon rapport entre complexité de modélisation et compréhension du fonctionnement du système. A la suite des scénarii, des anomalies ont été détectées par les indicateurs soulignant la cohérence de la stratégie de détection.

Chapitre 8

Expérimentations sur plateforme industrielle

Ce chapitre est consacré à l'expérimentation de l'approche sur une plateforme industrielle. Le changement d'application d'un environnement de simulation à celui de production dans un contexte industriel permet d'évaluer l'approche en conditions « réelles ». La première partie de ce chapitre consistera à présenter le système industriel, support de cette phase de test, ainsi que le protocole expérimental. Par la suite, la méthodologie de conception des filtres est ensuite appliquée pour générer le modèle du filtre de commande. La dernière partie est consacrée à l'expérimentation en tant que telle de l'approche sur différents scénarii. Des données acquises sur la plateforme industrielle seront réinjectées sous Matlab pour vérifier le comportement des filtres dans ces situations.

Table des matières

1.	Présentation de la plateforme de test	158
1.1.	Description du système industriel	158
1.2.	Description du protocole de test	159
2.	Mise en œuvre de l'approche sur la plateforme	160
3.	Analyse des résultats	162
4.	Bilan du chapitre.....	166

1. Présentation de la plateforme de test

La plateforme réelle utilisée a été mise à notre disposition par le pôle S.mart Dauphiné-Savoie²³. Elle est notamment composée d’un système de pilotage d’un atelier à 3 boucles représentant le fonctionnement d’un système de production. Deux boucles extérieures sont équipées de postes de travail qui permettent d’effectuer des opérations sur les palettes circulant sur un convoyeur. Une boucle centrale permet de distribuer les palettes vers les différents postes d’assemblage. Chacune des boucles est contrôlée par un automate Schneider Electric de la gamme Modicon Premium TSX et d’une supervision PCVue. La communication entre le niveau de supervision et la partie commande se fait en Ethernet TCP/IP tandis que la communication entre la partie commande et la partie opérative s’effectue en bus ASI et en liaison point à point. La partie du système que nous chercherons à sécuriser, qui sera nommée plateforme dans la suite de ce chapitre, se situe au niveau d’un poste de travail d’une boucle extérieure. Ce système est décrit dans la suite de ce paragraphe.

1.1. Description du système industriel

La plateforme industrielle à protéger est représentée en Figure 66 et se compose d’un convoyeur permettant de transporter des palettes d’un poste de travail à un autre. Celui considéré dans cet exemple est le Poste de Présentation (PdP). Le reste de la partie opérative se compose de sept capteurs, inductifs et capacitifs, et de cinq actionneurs, pouvant être des vérins double (extracteurs) et simple (les autres actionneurs) effets.

L’*ancrage PdP* permet à la pièce d’entrer dans le poste de présentation lorsque le capteur *ancrage PdP* détecte une palette. Le convoyeur amène la pièce dans le poste de présentation, au contact de la butée où le capteur *dans PdP* s’active. L’extracteur amène alors la palette sur l’indexeur en pilotant un vérin double effet avec les commandes *vers Indexeur* et *vers Convoyeur*. La commande est maintenue durant toute l’opération. Le capteur *côté Indexeur* permet de s’assurer de la bonne exécution de l’ordre par l’extracteur. L’indexeur est alors activé via la commande *indexeur haut* afin de simuler la durée d’une opération sur une palette. Les capteurs *indexeur haut* et *indexeur bas* permettent de vérifier les actions de l’actionneur. L’extracteur est ensuite ramené du côté du convoyeur, en réinitialisant la commande *vers Indexeur*. Les capteurs *côté Convoyeur* et *dans PdP* permettent de détecter que la palette est de retour sur le convoyeur. La pièce est ensuite libérée du poste de présentation via l’ordre *ouvrir butée*. Le capteur *PdP libre* permet de refermer la butée. Un nouveau cycle de production peut alors commencer lorsqu’une palette sera détectée par le capteur *ancrage PdP*.

² Anciennement AIP-Priméca DS : <http://aip.grenoble-inp.fr/accueil/plateforme-cim-764234.kjsp>

³ Nous tenons à remercier le pôle S.mart pour cette mise à disposition ainsi que M. Duong Quoc Bao, responsable technique de cette plateforme, pour ses conseils et son support aux expérimentations qui ont été menées.

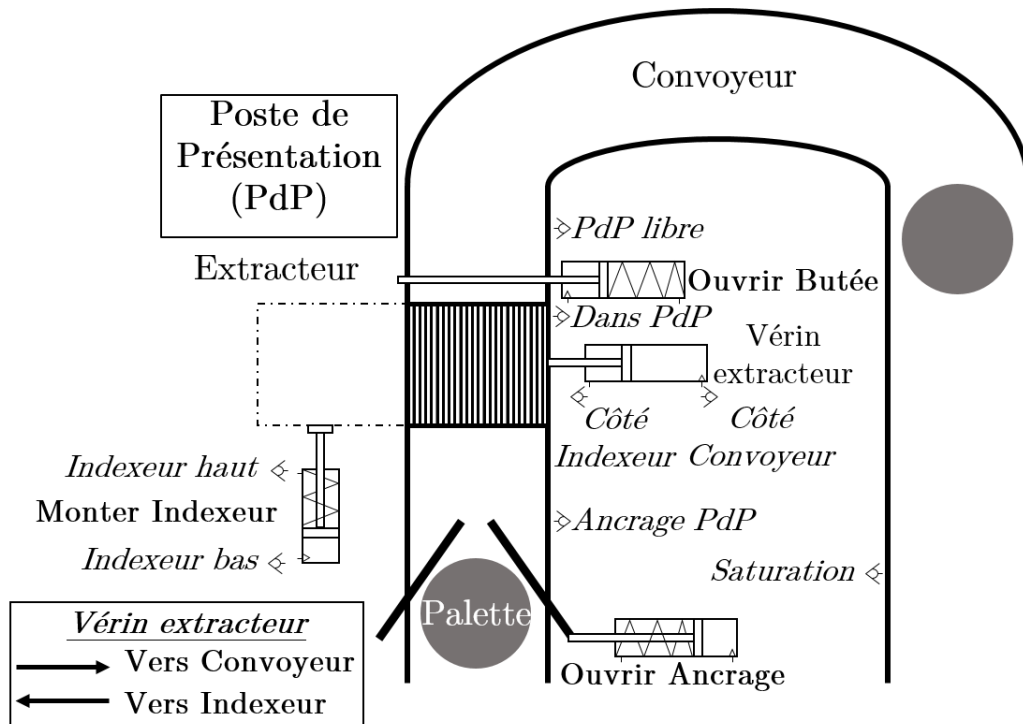


Figure 66 : Illustration de la plateforme industrielle utilisée pour évaluer l'approche S.A.F.E. en condition réelle

1.2. Description du protocole de test

Le logiciel de supervision PcVue est utilisé pour faire l'acquisition de points expérimentaux du fonctionnement de la plateforme afin de les réutiliser dans un environnement de simulation. La fonction « Exports de données » permet d'enregistrer les valeurs des variables sous forme d'un fichier csv. Ces variables, à l'aide de la table d'échange, reprennent celles contenues dans l'API et représentent les évolutions des ordres envoyés et de la partie opérative. Il est bien sûr considéré dans cet exemple qu'aucune attaque ou défaillance n'était présente au moment de l'acquisition et que l'environnement informatique était également « sûr » d'un point de vue informatique. La fréquence d'échantillonnage pour l'acquisition est fixée à 100 ms.

Plusieurs séquences ont fait l'objet d'une acquisition notamment lorsque le système est dans des modes ou des situations de fonctionnement particuliers comme :

- Le mode initialisation, où le système se met en condition pour produire (vider le PdP et initialiser le comportement de l'extracteur avant de commencer le cycle de production normale).
- Le mode production normale avec 1 palette, qui représente la situation basique décrite dans le Chapitre 8 - 1.1. Description du système industriel.
- Le mode de production normale avec plusieurs palettes, qui représente une situation particulière avec des aléas de production.

Les données acquises sont ensuite traitées pour les réinjecter dans un environnement de simulation où ces séquences sont rejouées afin de tester le comportement des filtres de commande et de comptes rendus. Des séquences supplémentaires sont également créées à partir de la séquence normale afin d'expérimenter la stratégie de détection des filtres vis-à-vis d'anomalies.

Les attaques temporelles sont exclues de l’expérimentation. L’analyse temporelle du système ne sera donc pas abordée. Dans le prochain paragraphe, la méthodologie de conception proposée dans ces travaux est appliquée au système du poste de présentation afin de construire des modèles de commande et de la partie opérative.

2. Mise en œuvre de l’approche sur la plateforme

La méthodologie de conception des filtres est appliquée au système afin de modéliser le comportement de la plateforme industrielle et d’alimenter les mécanismes de détection. Dans un premier temps, considérons que l’analyse de risques révèle le fait que trois zones critiques pour la partie opérative peuvent être atteintes lorsque :

- Les commandes *vers convoyeur* et *vers indexeur* sur le vérin double effet pilotant la position de l’extracteur sont réinitialisées, il y a un claquage du vérin.
- L’extracteur passe du côté indexeur au côté convoyeur alors que la commande *monter indexeur* est active et donc l’indexeur est en position haute.
- L’extracteur bouge vers le côté indexeur alors que ce dernier est activé.

Afin de caractériser ces événements redoutés dans notre modélisation, nous identifions les paramètres nécessaires et suffisants (Étape 1) pour décrire notre système sous forme de :

- Vecteur états $s \in S$ qui représente le comportement normal du système puisque les défaillances ne sont pas modélisées. Il se compose des valeurs des capteurs de niveaux et les valeurs de commandes des vannes telles que dans l’équation (25) :

$$\begin{aligned}
 s &= [\text{Valeurs Capteurs} ; \text{Valeurs Actionneurs}] \\
 &= [\text{Ancrage PdP}, \text{Côté convoyeur}, \text{Côté index}, \text{Index haut}, \text{Index bas}, \text{Dans PdP}, \text{PdP libre}] \\
 &\quad [\text{Ouvrir ancrage}, \text{Vers convoyeur}, \text{Vers indexeur}, \text{Monter indexeur}, \text{Ouvrir butée}]
 \end{aligned}
 \tag{25}$$

L’ensemble des éléments de l’équation (25) sont des variables tout-ou-rien. Ainsi, pour ce système, l’ensemble des états possibles « S » est composé d’au plus 1024 états et est noté S_{1024} .

- Vecteur actions $a \in A$ qui représente les actions pouvant être exécutées sur le système $a = \text{Ouvrir ancrage}, \text{Vers convoyeur}, \text{Vers index}, \text{Monter index}, \text{Ouvrir butée}$. Cet ensemble A est composé de 32 actions et est nommé dans la suite A_{32} .
- Ensemble des états atteignables $S_{Reach} \subset S$ qui contient 67 états dont 10 états optimaux (obtenus à l’aide du modèle de commande présenté en Figure 67), 3 états interdits et 54 états dangereux $S_{Dan} = S_{Reach}(S_{Opt} \cap S_{Pro})$.
- Ensemble des états inatteignables $S_{Unreach} = S \setminus S_{Reach}$ qui contient donc 957 états.

Un modèle de la loi de commande est réalisé à partir du cahier des charges (Étape 2) et est représenté en Figure 67. De la même manière, le modèle de la partie opérative (Étape 2) est construit en équation (26) à l’aide de notre automate M :

$$M = \{S_{1024}, A_{32}, \delta, s_0, S_{final}\} \tag{26}$$

L'ensemble S_{1024} représente l'ensemble des états atteignables pour le système, A_{32} est l'ensemble des ordres qui peuvent être appliqués sur le système (toutes les combinaisons possibles), δ est la fonction de transition qui permet de faire évoluer le système et dont les effets sont synthétisés sur le Tableau 13, s_0 est l'état initial du système défini par $s_0 = [1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0]$, S_{final} représente l'ensemble des états finaux de notre automate et donc nos conditions d'arrêts (S_{pro} , s_0 et S_{prev}). Le Tableau 14 regroupe les conditions sur le couple État/Action qui conduisent à l'état interdit de débordement.

Tableau 13 : Récapitulatif des effets des actions sur le système

Ordre $a \in A_{32}$	État $s \in S_{1024}$ du système avant l'action $a \in A_{32}$	Exception / Commentaire
Ouvrir Ancrage ($a_1 = [1 \times \times \times \times]$)	$\text{Si } \text{Ancrage } PdP = 1 \rightarrow \begin{cases} \text{Ancrage } PdP = 0 \\ \text{Dans } PdP = 1 \end{cases}$	-
Vers Indexeur $a_2 = [\times 1 \times \times \times]$	$\text{Si } \text{Dans } PdP = 1 \rightarrow \begin{cases} \text{Dans } PdP = 0 \\ \text{Côté indexeur} = 1 \\ \text{Côté convoyeur} = 0 \end{cases}$ $\text{Si } \text{Dans } PdP = 0 \rightarrow \begin{cases} \text{Côté indexeur} = 1 \\ \text{Côté convoyeur} = 0 \end{cases}$ $\text{Si } \text{Indexeur haut} = 1 \rightarrow \emptyset$	-
Vers Convoyeur $a_3 = [\times \times 1 \times \times]$	$\text{Si } \text{Dans } PdP = 1 \rightarrow \emptyset$ $\text{Si } \text{Dans } PdP = 0 \rightarrow \begin{cases} \text{Côté indexeur} = 0 \\ \text{Côté convoyeur} = 1 \end{cases}$	-
Monter Indexeur $a_4 = [\times \times \times 1 \times]$	$\text{Si } \text{Index haut} = 0 \rightarrow \begin{cases} \text{Index haut} = 1 \\ \text{Index bas} = 0 \end{cases}$	Lorsque la commande <i>Monter Indexeur</i> est arrêtée alors l'indexeur redescend (<i>Index bas</i> = 1)
Ouvrir butée ($a_5 = [\times \times \times \times 1]$)	$\text{Si } \text{Dans } PdP = 1 \rightarrow \begin{cases} \text{Dans } PdP = 0 \\ PdP \text{ Libre} = 1 \end{cases}$	La variable <i>PdP libre</i> est mise à zéro lorsque l'ordre <i>Ouvrir butée</i> est stoppé
Aucun ordre ($a_{32} = [0 \ 0 \ 0 \ 0 \ 0]$)	$s = [\times \times \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a_{32} = [\times \times \times \times \times \times \times 0 \ 0 \ 0 \ 0]$	-

Tableau 14 : Récapitulatif des conditions menant à l'état interdit de débordement

État interdit S_{Pro}	Ordre $a \in A_{32}$	État $s \in S_{1024}$ du système avant l'action $a \in A_{32}$
Claquage du vérin double effets de l'extracteur	$\begin{cases} \text{Vers Convoyeur} = 0 \\ \text{et} \\ \text{Vers Indexeur} = 0 \end{cases} \Leftrightarrow a = [\times 0 \ 0 \times \times]$	$\forall s \in S_{1024}$

Extracteur vers Convoyeur avec Indexeur haut	$a = [\times 1 0 \times \times]$	<i>Indexeur haut = 1</i>
Extracteur vers Indexeur avec Indexeur haut	$a = [\times 1 1 \times \times]$	<i>Indexeur haut = 1</i>

Nous pouvons désormais passer à l’application de l’algorithme de génération du modèle du filtre de commande (Étape 2) sur la base des modèles des parties commande et opérative. Cette étape conduit à l’obtention de 67 états atteignables et 1536 contextes menant à des états interdits. Le temps de calcul pris par l’algorithme pour converger vers une exploration totale du système est de 150 ms avec Intel Core i5-63000 - CPU 2,40 GHz 2,50 GHz – 8Go RAM programmé sous Eclipse en langage Java. L’étape 3 de la méthodologie est alors appliquée et correspond à l’exploitation du filtre et de la stratégie de détection dont les résultats sont détaillés dans le paragraphe suivant.

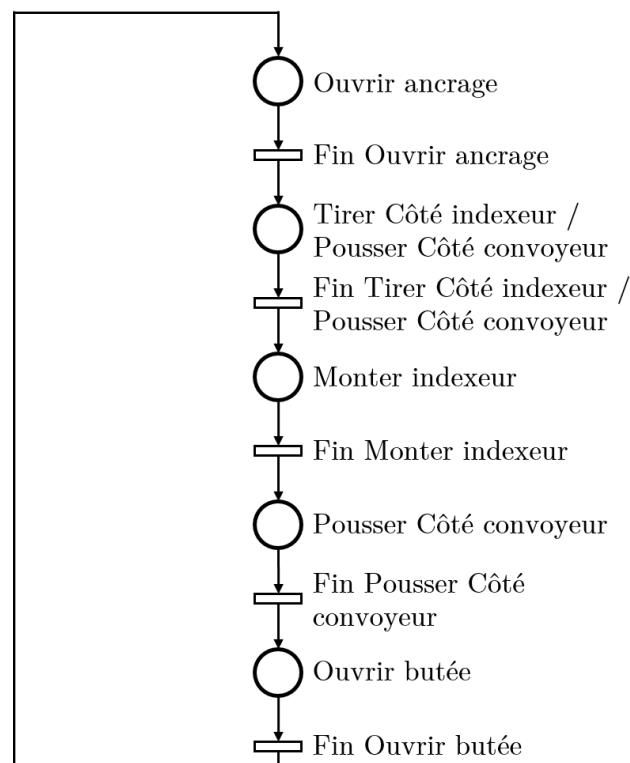


Figure 67 : Représentation de la loi de commande de la plateforme industrielle

L’analyse temporelle du système n’est pas faite puisque le périmètre d’expérimentation de l’approche sur plateforme industrielle ne tient pas compte des attaques temporelles.

3. Analyse des résultats

Les différents jeux de données acquis avec PC Vue sont ensuite réinjectés dans l’environnement de simulation Matlab. Ils permettent de vérifier le comportement des filtres de commande et de comptes rendus dans un environnement industriel dans plusieurs situations de

production. Comme les scénarii⁴ acquis ne comprennent pas de défaillances ou d'attaques, les données du comportement normal seront modifiées pour créer une attaque directe sur la plateforme.

Le premier jeu de données considéré représente le comportement normal de la plateforme en mode production avec une seule palette sur le convoyeur. Nous remarquons qu'aucune différence ne se fait d'un point de vue combinatoire avec les modèles élaborés durant la méthodologie de conception (modèle : courbe pleine, expérience : courbe avec cercle) comme en témoigne les Figure 68 et Figure 69. En effet, les trois trajectoires $T_{Order}(a|A_{Opt})$, $T_{état}(s|S_{Opt})$, $T_{état}(s|S_{Pro})$ sont superposées les unes avec les autres et les deux premières sont bien égales à zéro ce qui indique qu'il n'y a aucune déviation par rapport aux modèles construits tant au niveau du filtre de commande (Figure 68) que du filtre de comptes rendus (Figure 69). Ces résultats tendent à montrer que les modèles créés avec la méthodologie proposée sont cohérents avec le comportement normal attendu du système industriel.

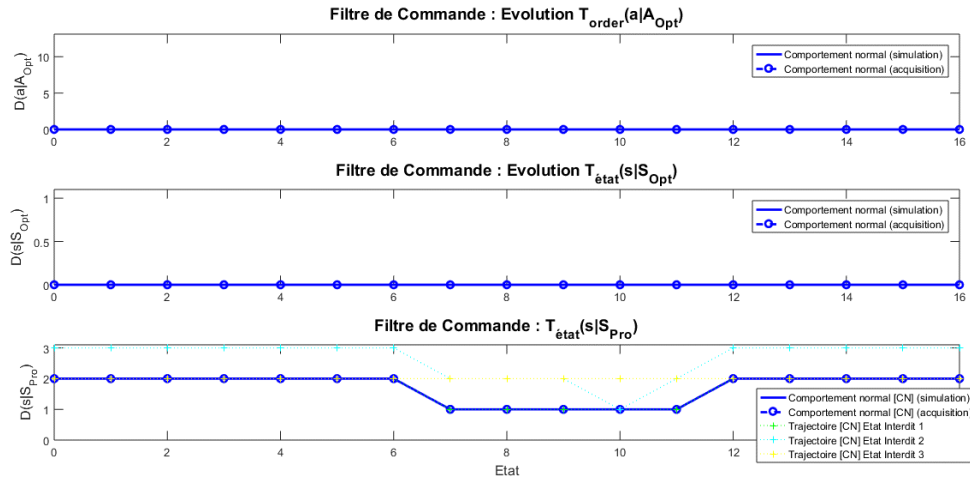


Figure 68 : Trajectoires calculées par le filtre de commande et comparaison avec les données acquises dans le mode de production et avec une seule palette sur le convoyeur

Le deuxième scénario proposé met en avant une situation où plusieurs palettes sont sur le convoyeur dans le mode production. Là encore, les modèles créés (courbe pleine) avec la méthodologie de conception sont cohérents avec les données acquises sur la plateforme (courbe rouge en pointillés avec cercle). Toutefois, au changement d'état n°14, une anomalie est détectée par les algorithmes ce qui se traduit sur la plateforme par la non-ouverture de la butée qui libère la palette du PdP. Cette déviation ne conduit pas à un état interdit comme le montre la trajectoire $T_{état}(s|S_{Pro})$ représentée en Figure 70 puisque la distance $D(s|S_{Pro})$ n'est jamais égale à zéro. De plus, cette anomalie est corrigée au changement d'état suivant, puisque les trajectoires $T_{Order}(a|A_{Opt})$ et $T_{état}(s|S_{Opt})$ sont nulles. La butée s'est alors ouverte libérant la palette sur le reste du convoyeur et permettant de retrouver le comportement normal de l'ICS. Cette anomalie, détectée par les mécanismes, est due au contexte multi-palettes de cette acquisition. Le capteur de saturation, présenté en Figure 66, est activé temporairement et, au moment de l'acquisition, par une autre palette circulant sur le convoyeur. Cela entraîne la non-ouverture de la butée pour

⁴ Les algorithmes étant similaires à ceux présentés dans le Chapitre 7, nous ne les remontrons pas.

éviter une saturation sur le poste de travail suivant. Ce comportement n'étant pas pris en compte dans notre modèle (puisque le capteur a été exclu du périmètre des paramètres à prendre en compte), cet événement est considéré comme anormal.

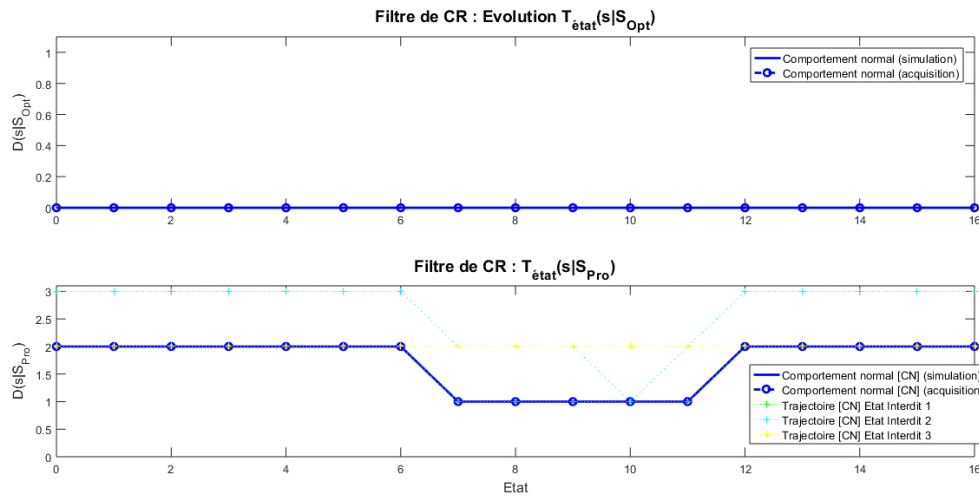


Figure 69 : Trajectoires calculées par le filtre de comptes rendus et comparaison avec les données acquises dans le mode de production et avec une seule palette sur le convoyeur

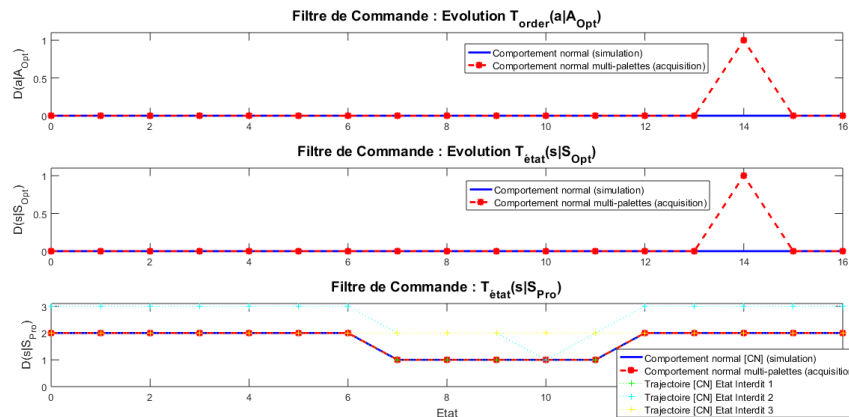


Figure 70 : Trajectoires calculées par le filtre de commande et comparaison avec les données acquises dans le mode de production et avec plusieurs palettes sur le convoyeur

Le troisième scénario illustre le comportement du système lorsque ce dernier est en mode initialisation. Dans ce mode particulier, préparant le système à la production, les modèles comportementaux (courbe pleine en Figure 71) ne coïncident pas avec les données acquises (courbe violette en pointillés avec cercles en Figure 71). En effet, une divergence forte entre les ordres émis par la partie commande et les ordres attendus par le modèle de commande est observée sur la courbe $T_{order}(a|A_{Opt})$ de la Figure 71. De la même manière, la trajectoire $T_{état}(s|S_{Opt})$ montre que les états courants du système ne correspondent pas à ceux définis comme optimaux dans le modèle de partie opérative. Cela est normal puisque les modèles de commande et de comptes rendus n'intègrent pas ce mode de fonctionnement. Toutefois, nous noterons que le filtre de commande ne bloque pas les ordres envoyés par la partie commande puisque la partie opérative n'est pas mise en danger comme l'indique $T_{état}(s|S_{Pro})$ sur la Figure 71.

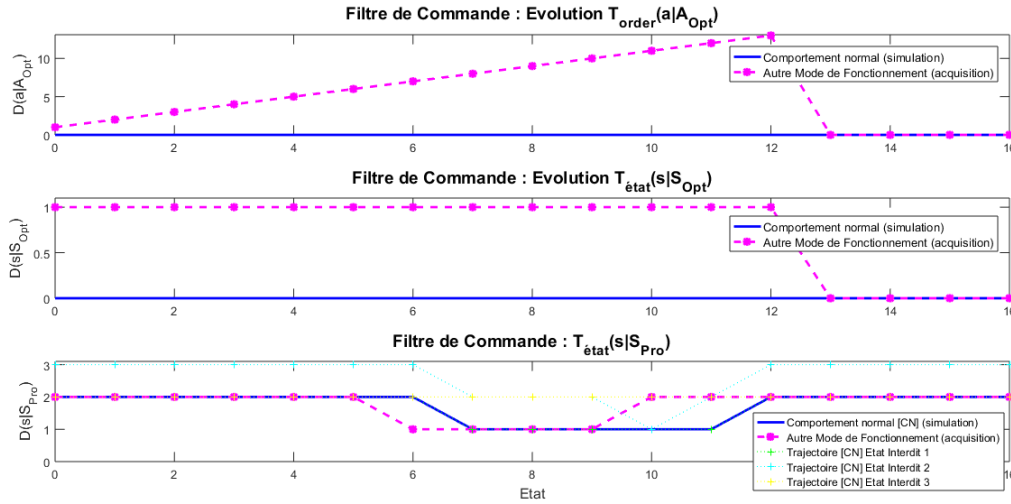


Figure 71 : Trajectoires calculées par le filtre de commande et comparaison avec les données acquises dans le mode initialisation

Le dernier scénario est élaboré à partir des données acquises sur la plateforme afin de simuler une attaque brutale sur le système. Les trajectoires illustrant le comportement normal du système (courbe pleine bleue) et le comportement sous l'attaque simulée (courbe violette en pointillés avec des cercles) sont représentées en Figure 72.

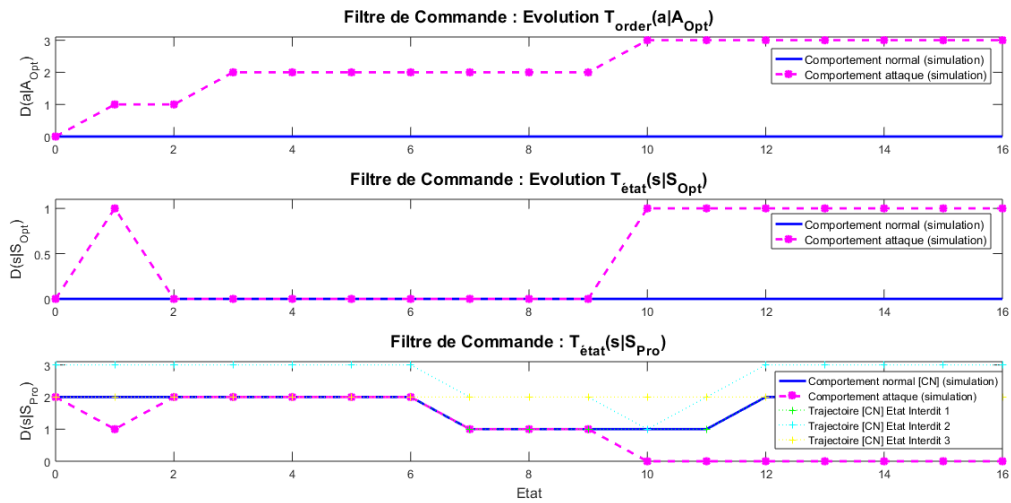


Figure 72 : Trajectoires calculées par le filtre de commande et comparaison avec le comportement face à une attaque simulée à partir des données acquises sur le comportement normal

Au changement d'état n°1, une première anomalie est détectée par les algorithmes, notamment $D(a|A_{Opt})$ et $D(s|S_{Opt})$. De plus, la trajectoire $T_{état}(s|S_{Pro})$ sur la Figure 72 indique que cette anomalie rapproche le système d'un état critique. En réalité, cela correspond au fait que l'indexeur est passé en position haute et qu'une action sur l'extracteur conduit le système à un état interdit. L'évolution en état n°2 corrige cette erreur en abaissant l'indexeur (tout en reprenant le cycle de production normale), toutefois, l'ordre ne correspond pas au modèle de la partie commande ce qui explique pourquoi $T_{order}(a|A_{Opt})$ augmente alors que les autres courbes deviennent similaires au comportement normal.

Au changement d’état n°9, l’action de pousser l’extracteur côté convoyeur alors que l’indexeur est toujours en position haute est envoyée par la partie commande. Cela se traduit par des trajectoires $T_{order}(a|A_{Opt})$ et $T_{état}(s|S_{Opt})$ détectant une anomalie et une trajectoire $T_{état} s|S_{Pro}$ nulle traduisant le fait que le système atteint un état critique. Le filtre de commande bloque alors l’ordre malveillant avant qu’il ne soit appliqué à la partie opérative. Ainsi, une dégradation du système est évitée.

4. Bilan du chapitre

Dans ce chapitre, la plateforme industrielle servant de base pour nos expérimentations a été présentée. Elle reprend le fonctionnement d’un atelier flexible de production où un convoyeur permet de distribuer des palettes sur divers postes de montage. Une acquisition de données a été faite sur cette plateforme dans le but de retranscrire son comportement dans un environnement de simulation avec les filtres de commande et de comptes rendus. L’approche S.A.F.E. a été appliquée à ce système, à commencer par la méthodologie de conception des filtres. Ainsi, sur la base d’une analyse de risques supposée faite, les zones critiques de la partie opérative sont considérées comme identifiées. La méthodologie a permis de définir le périmètre devant être sécurisé et les paramètres à prendre en compte. Les modèles de commande et de la partie opérative ont également été construits afin d’alimenter les mécanismes de détection. Enfin, le comportement des filtres a été analysé sur la base de l’acquisition de plusieurs points expérimentaux représentant différents modes de fonctionnement et situations de production. Ainsi, les filtres permettent de détecter des anomalies par rapport à un comportement de référence et respectent leurs spécifications de ne bloquer que les actions mettant directement en danger l’intégrité physique de l’installation. En revanche, comme les modèles n’intègrent pas tous les modes de fonctionnement ou tous les paramètres pris en considération dans la loi de commande, des faux positifs peuvent être détectés (comme le capteur de saturation ou le mode d’initialisation). Toutefois, la stratégie de détection présente des résultats positifs sur des données expérimentales.

Conclusion Générale

Les travaux que nous avons présentés dans ce document traitent de la surveillance d'attaques dans les systèmes de contrôle-commande à événements discrets. Ils ont été développés au sein de l'équipe Gestion et Conduite des Systèmes de Production (GCSP) du laboratoire des Sciences pour la Conception, l'Optimisation et la Production de Grenoble (G-SCOP).

Cette thèse a apporté sa contribution au travers de la proposition d'un mécanisme de détection à base de modèles localisés entre l'automate programmable industriel et les capteurs/actionneurs. En s'appuyant sur les approches filtres développées au CRAN et au LAGIS, nous avons développé un modèle de filtre de commande adapté à la détection d'attaques sur la base duquel ont été proposés quatre indicateurs de surveillance permettant de répondre aux quatre principaux types d'attaques reconnus dans le domaine. Au-delà de la proposition de ces indicateurs, nous avons développé les algorithmes de détection exploitant les modèles proposés, ainsi que les indicateurs. Afin de valider l'approche, nous avons développé un prototype informatique fonctionnel du mécanisme de détection proposé. Ce prototype nous a permis de mener une évaluation de l'approche sur la base de plusieurs cas d'étude, dont un réel, auxquels nous avons fait subir, en simulation, plusieurs scénarii d'attaques. Les résultats obtenus sont significatifs, confortant chacun des apports majeurs de l'approche proposée, véritablement novatrice dans le domaine :

- Le mécanisme de détection est positionné en tant que dernier rempart avant que l'ordre malveillant ne soit envoyé aux actionneurs.
- Ce positionnement permet de faire face au problème de complexité rencontré à plus haut niveau dans l'architecture de contrôle-commande.
- La détection s'appuie sur des modèles comportementaux permettant l'analyse sémantique de l'information au sens métier (automaticien).
- L'intégration d'indicateurs spécifiques permet d'appréhender les problématiques liées aux attaques en séquence ainsi qu'à celles temporelles.

Au terme de ces travaux, plusieurs axes de recherche se dégagent pour envisager, du point de vue des perspectives, de prolonger l'étude menée pendant ces trois ans.

A court terme, les perspectives se situent autour de l'applicatif et de l'approfondissement des tests qui ont commencé à être menés. Plusieurs axes d'investigations peuvent être envisagés :

- En premier lieu, l'approche telle qu'elle a été développée doit être déployée sur la plateforme CIM. Ce travail devra se faire sur la base des premières expérimentations réalisées dans le cadre de cette thèse. L'exécution des mécanismes de détection en

temps réels pourra se faire dans un premier temps depuis un ordinateur connecté au réseau industriel avant d'être implanté sur un dispositif embarqué temps réel. Cette perspective peut faire l'objet d'un stage PFE et servira à compléter l'approche développée.

- Dans la suite de cette perspective, à l'aide des données acquises sur la plateforme, des investigations sur l'utilisation des trajectoires calculées pourront être lancées. Comme nous avons pu le voir dans ces travaux, le comportement normal du système impose une forme caractéristique à la trajectoire représentant l'écart avec les états interdits. Cette dernière se répétant dans le temps, des techniques basées sur la reconnaissance de formes (Pattern Recognition) pourront être utilisées pour détecter des anomalies. Déjà utilisée pour la détection d'attaques en se basant sur l'étude des réseaux de communication (Valdes et Cheung 2009; Sayegh et al. 2014), cette perspective permettra un meilleur traitement des données remontées par les filtres.
- Dans la continuité d'un déploiement de l'approche sur la plateforme CIM, une extension de cette dernière pour filtrer les échanges entre la partie commande (niveau 1) et la supervision (niveau 2) est à envisager. Dans cette extension, les filtres seront à localiser au plus proche de la partie commande afin de garantir les informations envoyées à l'automate. Dans ce nouveau cadre d'application, les modèles de commande et de comptes rendus se doivent d'être adaptés à cette nouvelle localisation.
- Dans le cadre de stages de master recherche au moins trois sujets peuvent être lancés. Un premier devra lever l'hypothèse d'unicité de l'attaque énoncée au début de ce manuscrit. Ainsi, un nouveau profil d'attaquant doit être envisagé pour expérimenter l'approche. Le scénario associé développera des attaques affectant la sécurité des biens et des personnes, la qualité du produit et les sur-sollicitations des équipements en parallèle, en ne satisfaisant pas à la fois les contraintes combinatoires et temporelles du système protégé. Un deuxième sujet pourra être lancé sur l'apprentissage des contraintes temporelles par une intelligence artificielle pour enrichir l'analyse qui en est faite lors de l'étape 1 de la méthodologie. Son utilisation permettra à l'approche de s'adapter au cours du cycle de vie de l'ICS. Un troisième pourra s'articuler autour du modèle du filtre de comptes rendus. Sur la base de la méthodologie proposée pour le modèle du filtre de commande et des algorithmes permettant de le générer, une adaptation pourra être développée pour évaluer la remontée d'informations de la part des capteurs. Cet ajout, par rapport à l'approche proposée dans ce manuscrit, devra être également expérimenté sur les cas d'étude académiques et la plateforme industrielle présentés.

A moyen terme, au moins six orientations de recherche doivent être envisagées :

- La première doit lever une hypothèse forte énoncée au début de ce manuscrit sur la fiabilité des capteurs présents dans le système protégé. Comme nous l'avons remarqué lors de l'étude des systèmes industriels, ces derniers opèrent dans un environnement soumis à des défaillances. Le couplage avec les résultats des travaux de recherche de

(Duong 2012) portant sur le degré de confiance à accorder à un réseau de capteurs est une perspective à étudier.

- La deuxième concerne l'extension de l'approche à plusieurs cellules d'automates. Ainsi, le périmètre des paramètres à prendre en compte pour modéliser le système à protéger est plus complexe à gérer. Les modèles ne prendraient plus en compte un seul sous-ensemble du système physique mais plusieurs. Ceci améliorerait la pertinence de l'approche puisque la vision du procédé en serait agrandie. L'approche devra alors s'ouvrir aux systèmes à base de modèles distribués.
- La troisième concerne l'enrichissement du modèle de la partie opérative qui ne tient pas compte pour l'heure des opérations induites. L'utilisation des travaux de (Henry 2005) sera une base de départ afin de compléter les mécanismes de détection.
- La quatrième concerne la mise en repli du système à la suite d'une détection de contexte menant le système à un état interdit. L'approche actuelle bloque les ordres de la partie commande pour protéger le système contre un danger immédiat. Toutefois, le blocage des ordres peut être pénalisant puisqu'il induit des procédures de redémarrage et donc des temps de redémarrage et des pertes financières. Il fait également abstraction des évolutions induites relatives à une dynamique de la partie opérative. Une perspective qui a commencé à être explorée réside dans la création d'algorithmes de mise en repli pour conduire le système dans un état défini comme sûr. Les travaux sur la reconfiguration dynamique de (Henry 2005) pourront être utilisés afin de gérer cette procédure.
- Enfin, la cinquième doit permettre d'utiliser les modèles développés, et de les étendre à des fins de diagnostic, ne serait-ce que pour localiser le point d'entrée de l'attaque, de pronostiquer ses impacts pour, enfin, proposer des reconfigurations à l'exploitant.
- Enfin, la sixième concerne le point d'entrée de notre méthodologie : l'analyse de risques. Dans notre approche, cette dernière est considérée comme une hypothèse à notre travail. Elle doit être développée.

A plus long terme, il serait particulièrement intéressant de coupler cette approche à base de modèles déterministes avec d'autres afin d'améliorer l'analyse des impacts d'une attaque sur l'ensemble de l'architecture de contrôle-commande. Par exemple, un couplage avec des techniques à base de réseaux bayésiens multidimensionnels (permettant de modéliser la complexité d'une architecture de contrôle-commande) ouvre des perspectives intéressantes pour modéliser l'impact de l'exploitation d'une vulnérabilité d'un niveau à l'autre.

Références

- Abbasi, Ali, et Majid Hashemi. 2016. « Ghost in the PLC Designing an Undetectable Programmable Logic Controller Rootkit via Pin Control Attack ». *Black Hat Europe*, 1–35.
- Abdo, H., M. Kaouk, J. -M. Flaus, et F. Masse. 2018. « A safety/security risk analysis approach of Industrial Control Systems: A cyber bowtie – combining new version of attack tree with bowtie analysis ». *Computers & Security* 72 (Supplement C): 175-95. <https://doi.org/10.1016/j.cose.2017.09.004>.
- Abrams, Marshall, et Joe Weiss. 2008. « Malicious Control System Cyber Security Attack Case Study - Maroochy Water Services, Australia ». *Securing Water and Wastewater Systems*, juillet.
- Abu-Samah, A., M. K. Shahzad, et E. Zamai. 2017. « Bayesian based methodology for the extraction and validation of time bound failure signatures for online failure prediction ». *Reliability Engineering & System Safety*, Special Section: Applications of Probabilistic Graphical Models in Dependability, Diagnosis and Prognosis, 167 (novembre): 616-28. <https://doi.org/10.1016/j.ress.2017.04.016>.
- AFNOR. 1991. *Norme NF X50-151 Analyse de la valeur, analyse fonctionnelle. Expression fonctionnelle du besoin et cahier des charges fonctionnel*. AFNOR. <https://www.decitre.fr/livres/norme-nf-x50-151-analyse-de-la-valeur-analyse-fonctionnelle-5552120000675.html>.
- Aftabjehani, S. A., et A. Das. 2017. « Robust secure design by increasing the resilience of Attack Protection Blocks ». In *2017 IEEE 2nd International Verification and Security Workshop (IVSW)*, 13-18. <https://doi.org/10.1109/IVSW.2017.8031538>.
- Ahmadi, A., C. Cherifi, V. Cheutet, et Y. Ouzrout. 2017. « A review of CPS 5 components architecture for manufacturing based on standards ». In *2017 11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 1-6. <https://doi.org/10.1109/SKIMA.2017.8294091>.
- Ahmed, I., S. Obermeier, S. Sudhakaran, et V. Roussev. 2017. « Programmable Logic Controller Forensics ». *IEEE Security Privacy* 15 (6): 18-24. <https://doi.org/10.1109/MSP.2017.4251102>.
- Alphonsus, Ephrem Ryan, et Mohammad Omar Abdullah. 2016. « A review on the applications of programmable logic controllers (PLCs) ». *Renewable and Sustainable Energy Reviews* 60 (juillet): 1185-1205. <https://doi.org/10.1016/j.rser.2016.01.025>.
- Amoah, R., S. Camtepe, et E. Foo. 2016. « Securing DNP3 Broadcast Communications in SCADA Systems ». *IEEE Transactions on Industrial Informatics* 12 (4): 1474-85. <https://doi.org/10.1109/TII.2016.2587883>.
- Ani, Uchenna P. Daniel, Hongmei (Mary) He, et Ashutosh Tiwari. 2017. « Review of cybersecurity issues in industrial critical infrastructure: manufacturing in perspective ». *Journal of Cyber Security Technology* 1 (1): 32-74. <https://doi.org/10.1080/23742917.2016.1252211>.
- ANSSI. 2012. « Maîtriser la SSI pour les systèmes industriels ». V1.0. Paris: ANSSI (Agence nationale de la sécurité des systèmes d'information). www.ssi.gouv.fr.

- . 2014. « Cas pratique d'un tunnel routier ». V1.0. Paris: ANSSI (Agence nationale de la sécurité des systèmes d'information). www.ssi.gouv.fr.
- . 2018a. « Principales menaces ». ANSSI. 2018. <https://www.ssi.gouv.fr/entreprise/principales-menaces/>.
- . 2018b. « Protection des OIV en France ». ANSSI. 2018. <https://www.ssi.gouv.fr/entreprise/protection-des-oiv/protection-des-oiv-en-france/>.
- . s. d. « Rançongiciel ». ANSSI. Consulté le 3 mai 2018. <https://www.ssi.gouv.fr/entreprise/principales-menaces/cybercriminalite/rancongiel/>.
- Ashibani, Yosef, et Qusay H. Mahmoud. 2017. « Cyber physical systems security: Analysis, challenges and solutions ». *Computers & Security* 68 (juillet): 81-97. <https://doi.org/10.1016/j.cose.2017.04.005>.
- Babu, Bijoy, Thafasal Ijyas, P. Muneer, et Justin Varghese. 2017. « Security issues in SCADA based industrial control systems ». In *Anti-Cyber Crimes (ICACC), 2017 2nd International Conference on*, 47-51. IEEE. <http://ieeexplore.ieee.org/abstract/document/7905261/>.
- Barbosa, Rafael Ramos Regis, Ramin Sadre, et Aiko Pras. 2012. « Difficulties in Modeling SCADA Traffic: A Comparative Analysis ». In *Passive and Active Measurement*, 7192:126-35. Berlin, Germany: Springer Verlag. https://doi.org/10.1007/978-3-642-28537-0_13.
- . 2013. « Flow whitelisting in SCADA networks ». *International journal of critical infrastructure protection* 6 (3-4): 150-58. <https://doi.org/10.1016/j.ijcip.2013.08.003>.
- . 2016. « Exploiting Traffic Periodicity in Industrial Control Networks ». *International Journal of Critical Infrastructure Protection* 13 (juin): 52-62. <https://doi.org/10.1016/j.ijcip.2016.02.004>.
- Barker, R. T., et C. J. Cheese. 2012. « The application of data diodes for securely connecting nuclear power plant safety systems to the corporate IT network ». In *7th IET International Conference on System Safety, incorporating the Cyber Security Conference 2012*, 1-6. <https://doi.org/10.1049/cp.2012.1514>.
- Bartman, Tom, et Kevin Carson. 2016. « Securing communications for SCADA and critical industrial systems ». In *Protective Relay Engineers (CPRE), 2016 69th Annual Conference for*, 1-10. IEEE. <http://ieeexplore.ieee.org/abstract/document/7914914/>.
- Basnight, Zachry, Jonathan Butts, Juan Lopez, et Thomas Dube. 2013. « Firmware Modification Attacks on Programmable Logic Controllers ». *International Journal of Critical Infrastructure Protection* 6 (2): 76-84. <https://doi.org/10.1016/j.ijcip.2013.04.004>.
- Belkacem, L., L. Mhamdi, Z. Simeu-Abazi, H. Messaoud, et E. Gascard. 2016. « Diagnosis of Hybrid Dynamical Systems through Hybrid Automata ». *IFAC-PapersOnLine* 49 (12): 990-95. <https://doi.org/10.1016/j.ifacol.2016.07.571>.
- Bellettini, Carlo, et Julian L. Rrushi. 2008. « A Product Machine Model for Anomaly Detection of Interposition Attacks on Cyber-Physical Systems ». In *Proceedings of The Ifip Tc 11 23rd International Information Security Conference*, édité par Sushil Jajodia, Pierangela Samarati, et Stelvio Cimato, 285-300. Springer US. http://link.springer.com.gate6.inist.fr/chapter/10.1007/978-0-387-09699-5_19.
- Ben Said, Anis, Muhammad Kashif Shahzad, Éric Zamaï, S. Hubac, et Michel Tollenaere. 2015. « Experts' Knowledge Renewal and Maintenance Actions Effectiveness in High-Mix Low-

- Volume Industries, Using Bayesian Approach ». *Cognition, Technology and Work*, 20. <https://hal.archives-ouvertes.fr/hal-01201796>.
- Bencsáth. 2012. « Duqu, Flame, Gauss: Followers of Stuxnet ». présenté à RSA Conference Europe 2012. https://www.rsaconference.com/writable/presentations/file_upload/br-208_bencsath.pdf.
- Berruet, Pascal, Jean-François Petin, Fabien Rigaud, Armand Toguyeni, et Éric Zamaï. 2007. « Architectures de pilotage de procédés industriels ». *Techniques de l'Ingénieur, Biomédical - Pharma | Médicaments et produits pharmaceutiques*, AG3510 V1: 22. <https://www.techniques-ingenieur.fr/base-documentaire/biomedical-pharmath15/production-des-medicaments-42610210/architectures-de-pilotage-de-procedes-industriels-ag3510/>.
- Bock, Patrice, Jean-Pierre Hauet, Romain François, et Robert Foley. 2017. « Ukrainian Power Grids Cyberattack ». *InTech, March/April*, avril 2017. http://www.isa-france.org/telechargement/fichiers/Intech_ISA99_2017/Article%20Intech.pdf.
- Bolton, W. 2009. *Programmable Logic Controllers*. 5th ed. Newnes.
- Bonney, Gregor, Hans Höfken, Benedikt Paffen, et Marko Schuba. 2015. « ICS/SCADA security analysis of a Beckhoff CX5020 PLC ». In *Information Systems Security and Privacy (ICISSP), 2015 International Conference on*, 1–6. IEEE. <http://ieeexplore.ieee.org/abstract/document/7509940/>.
- Carcano, A., A. Coletta, M. Guglielmi, M. Masera, I. Nai Fovino, et A. Trombetta. 2011. « A Multidimensional Critical State Analysis for Detecting Intrusions in SCADA Systems ». *IEEE Transactions on Industrial Informatics* 7 (2): 179–86. <https://doi.org/10.1109/TII.2010.2099234>.
- Carcano, Andrea, Igor Nai Fovino, Marcelo Masera, et Alberto Trombetta. 2009. « State-based network intrusion detection systems for SCADA protocols: a proof of concept ». In *International Workshop on Critical Information Infrastructures Security*, 138–150. Springer. http://link.springer.com/chapter/10.1007/978-3-642-14379-3_12.
- Cárdenas, Alvaro A., Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, et Shankar Sastry. 2011. « Attacks against process control systems: risk assessment, detection, and response ». In *Proceedings of the 6th ACM symposium on information, computer and communications security*, 355–366. ACM. <http://dl.acm.org/citation.cfm?id=1966959>.
- Cárdenas, Alvaro A., Saurabh Amin, et Shankar Sastry. 2008. « Research Challenges for the Security of Control Systems. » *Proceedings of the 3rd conference on Hot topics in security*. https://www.usenix.org/legacy/events/hotsec08/tech/full_papers/cardenas/cardenas_html/hotsecHTML.html.
- Cardenas, Alvaro A., Tanya Roosta, et Shankar Sastry. 2009. « Rethinking security properties, threat models, and the design space in sensor networks: A case study in SCADA systems ». *Ad Hoc Networks* 7 (8): 1434–47. <https://doi.org/10.1016/j.adhoc.2009.04.012>.
- Cardin, O., F. Ounnar, A. Thomas, et D. Trentesaux. 2017. « Future Industrial Systems: Best Practices of the Intelligent Manufacturing and Services Systems (IMS2) French Research Group ». *IEEE Transactions on Industrial Informatics* 13 (2): 704–13. <https://doi.org/10.1109/TII.2016.2605624>.

- Cardin, Olivier. 2016. « Contribution to design, evaluation and implementation of cyber-physical production systems ». Habilitation à diriger des recherches, Université de Nantes. <https://tel.archives-ouvertes.fr/tel-01443318>.
- Caselli, Marco, Emmanuele Zambon, et Frank Kargl. 2015. « Sequence-aware Intrusion Detection in Industrial Control Systems ». In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, 13–24. New York, NY, USA: ACM. <https://doi.org/10.1145/2732198.2732200>.
- Cassandras, Christos G., et Stéphane Lafortune. 2008. *Introduction to Discrete Event Systems*. 2. ed. New York, NY: Springer.
- Chaillet, A., M. Combacau, et M. Courvoisier. 1993. « Specification of FMS real-time control based on Petri nets with objects and process failure monitoring ». In *International Conference on Industrial Electronics, Control, and Instrumentation, 1993. Proceedings of the IECON '93*, 144-49 vol.1. <https://doi.org/10.1109/IECON.1993.339091>.
- Checkoway, Stephen, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, et Tadayoshi Kohno. 2011. « Comprehensive Experimental Analyses of Automotive Attack Surfaces. » In *USENIX Security Symposium*. San Francisco. http://static.usenix.org/events/sec11/tech/full_papers/Checkoway.pdf.
- Cheminod, Manuel, Luca Durante, et Adriano Valenzano. 2013. « Review of Security Issues in Industrial Networks ». *IEEE Transactions on Industrial Informatics* 9 (1): 277-93. <https://doi.org/10.1109/TII.2012.2198666>.
- Cherdantseva, Yulia, Pete Burnap, Andrew Blyth, Peter Eden, Kevin Jones, Hugh Soulsby, et Kristan Stoddart. 2016. « A review of cyber security risk assessment methods for SCADA systems ». *Computers & Security* 56 (février): 1-27. <https://doi.org/10.1016/j.cose.2015.09.009>.
- Cherepanov, Anton. 2017. « A New Threat for Industrial Control Systems (WIN32/INDUSTROYER) ». *ESET*, 17.
- Cheung, Steven, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner, et Alfonso Valdes. 2007. « Using model-based intrusion detection for SCADA networks ». In *Proceedings of the SCADA security scientific symposium*, 46:1–12. Citeseer. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.141.2076&rep=rep1&type=pdf>.
- Clarhaut, Joffrey, Nicolas Dupoty, Franck Ebel, Jérôme Hennecart, et Frédéric Vicogne. 2015. *Cyberdéfense: La sécurité de l'informatique industrielle (domotique, industrie, transports)*. Epsilon. France: Editions ENI, 2015.
- CLUSIF. 2014. « Cybersécurité des systèmes industriels: par où commencer? Panorama des référentiels », juin, 26. <https://clusif.fr/publications/cybersecurite-des-systemes-industriels-par-ou-commencer-synthese-des-bonnes-pratiques-et-panorama-des-referentiels/>.
- Colla, M., T. Leidi, et M. Semo. 2009. « Design and implementation of industrial automation control systems: A survey ». In *2009 7th IEEE International Conference on Industrial Informatics*, 570-75. <https://doi.org/10.1109/INDIN.2009.5195866>.
- Combacau, M., et M. Courvoisier. 1990. « A hierarchical and modular structure for FMS control and monitoring ». In *Proceedings [1990]. AI, Simulation and Planning in High Autonomy Systems*, 80-88. <https://doi.org/10.1109/AIHAS.1990.93923>.

- Combacau, Michel. 1991. « Commande et surveillance des systèmes à événements discrets complexes : application aux ateliers flexibles ». Toulouse 3. <http://www.theses.fr/1991TOU30220>.
- Combacau, Michel, Philippe Esteban, et Alexandre Nketsa. 2005. « Commande à réseaux de Petri - Modélisation ». *Techniques de l'Ingénieur* 7572 (octobre): 1-18.
- Commission du Livre blanc. 2008. *Défense et Sécurité Nationale: le Livre blanc*. Édité par O. Jacob : Documentation française. Paris.
- . 2013. *Défense et Sécurité Nationale: le Livre blanc*. Paris: Documentation française.
- CPNI. 2011. « Securing the Move to IP-Based SCADA/PLC Networks ». CPNI (Centre for the Protection of National Infrastructure). https://www.ncsc.gov.uk/content/files/protected_files/document_files/The%20Move%20to%20IP%20Based%20SCADA%20Networks%20151111.pdf.
- Cruette, D., J. P. Bourey, et J. C. Gentina. 1991. « Hierarchical specification and validation of operating sequences in the context of FMSs ». *Computer integrated manufacturing systems* 4 (3): 140–156. <http://www.sciencedirect.com/science/article/pii/095152409190043X>.
- Denning, D. E. 1987. « An Intrusion-Detection Model ». *IEEE Transactions on Software Engineering* SE-13 (2): 222-32. <https://doi.org/10.1109/TSE.1987.232894>.
- Dequeant, Kean. 2017. « Workflow variability modeling in microelectronic manufacturing ». Theses, Université Grenoble Alpes. <https://hal.archives-ouvertes.fr/tel-01652884>.
- Duong, Quoc Bao. 2012. « Approche probabiliste pour l'estimation dynamique de la confiance accordée à un équipement de production : vers une contribution au diagnostic de services des SED ». Phdthesis, Université de Grenoble. <https://tel.archives-ouvertes.fr/tel-00910079/document>.
- Dzung, D., M. Naedele, T.P. Von Hoff, et M. Crevatin. 2005. « Security for Industrial Communication Systems ». *Proceedings of the IEEE* 93 (6): 1152-77. <https://doi.org/10.1109/JPROC.2005.849714>.
- Efthymiou, K., A. Pagoropoulos, N. Papakostas, D. Mourtzis, et G. Chryssolouris. 2012. « Manufacturing Systems Complexity Review: Challenges and Outlook ». *Procedia CIRP*, 45th CIRP Conference on Manufacturing Systems 2012, 3 (janvier): 644-49. <https://doi.org/10.1016/j.procir.2012.07.110>.
- Ellis, A. 2015. « Integrating Industrial Control System (ICS) safety and security #x2014; A potential approach ». In *10th IET System Safety and Cyber-Security Conference 2015*, 1-7. <https://doi.org/10.1049/cp.2015.0294>.
- ENISA. 2011. « Protecting Industrial Control Systems: Recommendations for Europe and Member States ».
- . 2017. « Communication Network Dependencies for ICS/SCADA Systems ».
- Erez, Noam, et Avishai Wool. 2015. « Control variable classification, modeling and anomaly detection in Modbus/TCP SCADA systems ». *International Journal of Critical Infrastructure Protection* 10 (septembre): 59-70. <https://doi.org/10.1016/j.ijcip.2015.05.001>.
- EtherCAT Technology Group. 2012. « EtherCAT: The Ethernet Fieldbus ». octobre.
- Evangelista, Thierry. 2004. *Les IDS - Les systèmes de détection d'intrusions informatiques*. 1ère édition. Dunod.

- Falliere, Nicolas, Liam O. Murchu, et Eric Chien. 2011. « W32. Stuxnet Dossier ». Version 1.4. Symantec Security Response. http://i69.fr/autoblog/autoblogs/korben.info/media/0ae4f7a1.w32_stuxnet_dossier.pdf.
- Fauri, D., B. de Wijs, J. den Hartog, E. Costante, E. Zambon, et S. Etalle. 2017. « Encryption in ICS networks: A blessing or a curse? » In *2017 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 289-94. <https://doi.org/10.1109/SmartGridComm.2017.8340732>.
- Flaus, J. M. 2013. *Risk Analysis: Socio-Technical and Industrial Systems*. Wiley-ISTE.
- Fourastier, Yannick, et Ludovic Pietre-Cambacedes. 2015. *Cybersécurité des installations industrielles : défendre ses systèmes numériques*. Cépaduès Editions.
- Fovino, I. N., A. Carcano, T. D. L. Murel, A. Trombetta, et M. Masera. 2010. « Modbus/DNP3 State-Based Intrusion Detection System ». In *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 729-36. <https://doi.org/10.1109/AINA.2010.86>.
- Fovino, I. Nai, A. Coletta, A. Carcano, et M. Masera. 2012. « Critical State-Based Filtering System for Securing SCADA Network Protocols ». *IEEE Transactions on Industrial Electronics* 59 (10): 3943-50. <https://doi.org/10.1109/TIE.2011.2181132>.
- Fovino, Igor Nai, Marcelo Masera, et Alessio De Cian. 2009. « Integrating cyber attacks within fault trees ». *Reliability Engineering & System Safety*, ESREL 2007, the 18th European Safety and Reliability Conference, 94 (9): 1394-1402. <https://doi.org/10.1016/j.ress.2009.02.020>.
- Friedberg, Ivo, Kieran McLaughlin, Paul Smith, David Laverty, et Sakir Sezer. 2017. « STPA-SafeSec: Safety and security analysis for cyber-physical systems ». *Journal of Information Security and Applications* 34 (juin): 183-96. <https://doi.org/10.1016/j.jisa.2016.05.008>.
- Gao, Wei, Thomas Morris, Bradley Reaves, et Drew Richey. 2010. « On SCADA control system command and response injection and intrusion detection ». In *eCrime Researchers Summit (eCrime)*, 2010, 1-9. IEEE. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5706699.
- Garcia, L., S. Zonouz, Dong Wei, et L. P. de Aguiar. 2016. « Detecting PLC control corruption via on-device runtime verification ». In *2016 Resilience Week (RWS)*, 67-72. <https://doi.org/10.1109/RWEEK.2016.7573309>.
- Gentil, Sylviane, et Éric Zamaï. 2003. « Principes des chaînes de régulation ». *Techniques de l'Ingénieur, Automatique - Robotique | Automatique et ingénierie système*, S7090 V1: 25. <https://www.techniques-ingenieur.fr/base-documentaire/automatique-robotique-th16/modelisation-et-analyse-de-systemes-asservis-42391210/principes-des-chaines-de-regulation-s7090/>.
- Goldenberg, Niv, et Avishai Wool. 2013. « Accurate Modeling of Modbus/TCP for Intrusion Detection in SCADA Systems ». *International Journal of Critical Infrastructure Protection* 6 (2): 63-75. <https://doi.org/10.1016/j.ijcip.2013.05.001>.
- González-Prida, Vicente, Jayakumar Shambhu, Antonio Jesus Guillen, Joel Adams, François Pérès, et Khairy Kobbacy. 2016. « An approach to risk quantification based on pseudo-random failure rates ». *IFAC-PapersOnLine* 49 (n° 28): 179-84. <https://doi.org/10.1016/j.ifacol.2016.11.031>.

- Graham, J., J. Hieb, et J. Naber. 2016. « Improving cybersecurity for Industrial Control Systems ». In *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, 618-23. <https://doi.org/10.1109/ISIE.2016.7744960>.
- Gupta, Rachana Ashok, et Mo-Yuen Chow. 2010. « Networked Control System: Overview and Research Trends ». *IEEE Transactions on Industrial Electronics* 57 (7): 2527-35. <https://doi.org/10.1109/TIE.2009.2035462>.
- Hadeli, H., R. Schierholz, M. Braendle, et C. Tudu. 2009. « Generating configuration for missing traffic detector and security measures in industrial control systems based on the system description files ». In *IEEE Conference on Technologies for Homeland Security, 2009. HST '09*, 503-10. <https://doi.org/10.1109/THS.2009.5168079>.
- Hadziosmanovic, Dina, Robin Sommer, Emmanuele Zambon, et Pieter Hartel. 2013. « Through the eye of the PLC: towards semantic security monitoring for industrial control systems ». *International Computer Science Institute, Berkeley*. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.681.1295&rep=rep1&type=pdf>.
- Haight, Joel M. 2013. *Handbook of Loss Prevention Engineering*. 1st éd. Vol. Volume 1&2. Wiley-VCH.
- Hailesellasiye, Muluken, et Syed Rafay Hasan. 2017. « Intrusion Detection in PLC-Based Industrial Control Systems Using Formal Verification Approach in Conjunction with Graphs ». *Journal of Hardware and Systems Security*, octobre. <https://doi.org/10.1007/s41635-017-0017-y>.
- Han, S., M. Xie, H. H. Chen, et Y. Ling. 2014. « Intrusion Detection in Cyber-Physical Systems: Techniques and Challenges ». *IEEE Systems Journal* 8 (4): 1052-62. <https://doi.org/10.1109/JSYST.2013.2257594>.
- Hanssen, Dag H. 2015. *Programmable Logic Controllers: A Practical Approach to IEC 61131-3 Using CoDeSys*. 1st ed. Wiley.
- He, H., et J. Yan. 2016. « Cyber-physical attacks and defences in the smart grid: a survey ». *IET Cyber-Physical Systems: Theory Applications* 1 (1): 13-27. <https://doi.org/10.1049/iet-cps.2016.0019>.
- Henry. 2005. « Synthèse de lois de commande pour la configuration et la reconfiguration des systèmes industriels complexes ». Automatique / Robotique, Grenoble: Institut National Polytechnique de Grenoble - INPG,. <https://hal.inria.fr/docs/00/16/84/12/PDF/These2005-Henry.pdf>.
- Henry, S., E. Zamaï, et M. Jacomino. 2004. « Real time reconfiguration of manufacturing systems ». In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, 5:4266-71 vol.5. <https://doi.org/10.1109/ICSMC.2004.1401201>.
- Henry, Sébastien, Eric Zamaï, et Mireille Jacomino. 2012. « Logic Control Law Design for Automated Manufacturing Systems ». *Engineering Applications of Artificial Intelligence* 25 (4): 824-36. <https://doi.org/10.1016/j.engappai.2012.01.005>.
- Holloway, L.E., et B.H. Krogh. 1991. « Monitoring behavioral evolution for on-line fault detection ». In *IFAC/IMACS International Conference « Fault Detection, Supervision and Safety for Technical Processes », SAFEPROCESS'91*, 313-19. Baden Baden, Germany: IFAC.

- Hu, R., W. Hu, et Z. Chen. 2015. « Research of smart grid cyber architecture and standards deployment with high adaptability for Security Monitoring ». In *2015 International Conference on Sustainable Mobility Applications, Renewables and Technology (SMART)*, 1-6. <https://doi.org/10.1109/SMART.2015.7399218>.
- ICS-CERT. 2013. « W32.Duqu-Malware (Update E) ». ICS-CERT. 8 mai 2013. <https://ics-cert.us-cert.gov/alerts/ICS-ALERT-11-291-01E>.
- . 2015. « Schneider Electric Modicon M340 Buffer Overflow Vulnerability | ICS-CERT ». ICS-CERT. 2015. <https://ics-cert.us-cert.gov/advisories/ICSA-15-351-01>.
- . 2016a. « ICS-CERT Annual Vulnerability Coordination Report 2016 ». *NCCIC*, 17.
- . 2016b. « Rockwell Automation MicroLogix 1100 PLC Overflow Vulnerability | ICS-CERT ». ICS-CERT. 2016. <https://ics-cert.us-cert.gov/advisories/ICSA-16-026-02>.
- . 2016c. « ICS-CERT / The Industrial Control Systems Cyber Emergency Response Team ». 15 septembre 2016. <https://ics-cert.us-cert.gov/>.
- . 2017. « Indicators Associated With WannaCry Ransomware (Update I) | ICS-CERT ». 2017. <https://ics-cert.us-cert.gov/alerts/ICS-ALERT-17-135-01I>.
- . 2018a. « Overview of Cyber Vulnerabilities | ICS-CERT ». 2018. <https://ics-cert.us-cert.gov/content/overview-cyber-vulnerabilities>.
- . 2018b. « Schneider Electric Triconex Tricon (Update A) | ICS-CERT ». ICS-CERT. 17 avril 2018. <https://ics-cert.us-cert.gov/advisories/ICSA-18-107-02>.
- IIoT World. 2017. « The impact of WannaCry on industrial control systems (ICS) ». IIoT World. 13 mai 2017. <http://iiot-world.com/cybersecurity/the-impact-of-wannacry-on-industrial-control-systems-ics/>.
- Iung, Eric Levrat, Adolfo Crespo Marquez, et Heinz Erbe. 2009. « Conceptual framework for e-Maintenance: Illustration by e-Maintenance technologies and platforms ». *Annual Reviews in Control* 33 (2): 220-29. <https://doi.org/10.1016/j.arcontrol.2009.05.005>.
- Iung, M. Monnin, A. Voisin, P. Cocheteux, et E. Levrat. 2008. « Degradation state model-based prognosis for proactively maintaining product performance ». *CIRP Annals* 57 (1): 49-52. <https://doi.org/10.1016/j.cirp.2008.03.026>.
- Jeon, B. S., et J. C. Na. 2016. « A study of cyber security policy in industrial control system using data diodes ». In *2016 18th International Conference on Advanced Communication Technology (ICACT)*, 314-17. <https://doi.org/10.1109/ICACT.2016.7423374>.
- Kabir-Querrec, Maëlle. 2017. « Cyber security of the smart grid control systems: intrusion detection in IEC 61850 communication networks ». Theses, Université Grenoble Alpes. <https://hal.archives-ouvertes.fr/tel-01609230>.
- Kabir-Querrec, Maëlle, Stéphane Mocanu, Jean-Marc Thiriet, et Eric Savary. 2016. « A Test bed dedicated to the Study of Vulnerabilities in IEC 61850 Power Utility Automation Networks ». In *21st IEEE Emerging Technologies and Factory Automation*. Berlin, Germany. <https://hal.archives-ouvertes.fr/hal-01366270>.
- Kamel, Khaled, et Eman Kamel. 2013. *Programmable Logic Controllers: Industrial Control*. 1st ed. Mc Graw Hill Professional.
- Karnouskos, S. 2011. « Stuxnet worm impact on industrial cyber-physical system security ». In *IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society*, 4490-94. <https://doi.org/10.1109/IECON.2011.6120048>.
- Kaspersky Lab. 2014. « Black Energy APT Malware | Cyber Attaque En Ukraine | Kaspersky Lab ». Kaspersky. 2014. <https://www.kaspersky.fr/resource-center/threats/blackenergy>.

- Khorrami, F., P. Krishnamurthy, et R. Karri. 2016. « Cybersecurity for Control Systems: A Process-Aware Perspective ». *IEEE Design Test* 33 (5): 75-83. <https://doi.org/10.1109/MDAT.2016.2594178>.
- Kleinmann, Amit, et Avishai Wool. 2014. « Accurate Modeling of the Siemens S7 SCADA Protocol for Intrusion Detection and Digital Forensics ». *Journal of Digital Forensics, Security and Law*. <https://doi.org/10.15394/jdfsl.2014.1169>.
- . 2017. « Automatic Construction of Statechart-Based Anomaly Detection Models for Multi-Threaded Industrial Control Systems ». *ACM Trans. Intell. Syst. Technol.* 8 (4): 55:1–55:21. <https://doi.org/10.1145/3011018>.
- Klötzer, C., J. Weißenborn, et A. Pflaum. 2017. « The Evolution of Cyber-Physical Systems as a Driving Force Behind Digital Transformation ». In *2017 IEEE 19th Conference on Business Informatics (CBI)*, 02:5-14. <https://doi.org/10.1109/CBI.2017.8>.
- Knapp, Eric D. 2014. *Industrial network security: securing critical infrastructure networks for smart grid, scada, and other industrial control systems*. 2nd edition. Waltham, MA: Elsevier.
- Koscher, K., A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, et al. 2010. « Experimental Security Analysis of a Modern Automobile ». In *2010 IEEE Symposium on Security and Privacy*, 447-62. <https://doi.org/10.1109/SP.2010.34>.
- Koucham, Oualid, Stéphane Mocanu, Guillaume Hiet, Jean-Marc Thiriet, et Frédéric Majorczyk. 2016. « Detecting Process-Aware Attacks in Sequential Control Systems ». In *21st Nordic Conference on Secure IT Systems (NordSec 2016)*. Oulu, Finland. <https://hal.inria.fr/hal-01361081>.
- Koutsandria, G., V. Muthukumar, M. Parvania, S. Peisert, C. McParland, et A. Scaglione. 2014. « A hybrid network IDS for protective digital relays in the power transmission grid ». In *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 908-13. <https://doi.org/10.1109/SmartGridComm.2014.7007764>.
- Kriaa, S., M. Bouissou, et Y. Laarouchi. 2015. « A model based approach for SCADA safety and security joint modelling: S-cube ». In *10th IET System Safety and Cyber-Security Conference 2015*, 1-6. <https://doi.org/10.1049/cp.2015.0293>.
- Kriaa, Siwar, Ludovic Pietre-Cambacedes, Marc Bouissou, et Yoran Halgand. 2015. « A Survey of Approaches Combining Safety and Security for Industrial Control Systems ». *Reliability Engineering & System Safety* 139 (juillet): 156-78. <https://doi.org/10.1016/j.ress.2015.02.008>.
- Kumar, Ratnesh, Hok M. Cheung, et Steven I. Marcus. 1998. « Extension Based Limited Lookahead Supervision of Discrete Event Systems ». *Automatica* 34 (11): 1327-44. [https://doi.org/10.1016/S0005-1098\(98\)00077-6](https://doi.org/10.1016/S0005-1098(98)00077-6).
- Kushner, D. 2013. « The real story of stuxnet ». *IEEE Spectrum* 50 (3): 48-53. <https://doi.org/10.1109/MSPEC.2013.6471059>.
- Langner, Ralph. 2011. « Stuxnet: Dissecting a cyberwarfare weapon ». *IEEE Security & Privacy* 9 (3): 49–51. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5772960.
- Lee, Dave. 2017. « Global Cyber-Attacks Slow but Fears Remain ». *BBC News*, 15 mai 2017, sect. Technology. <http://www.bbc.com/news/technology-39920141>.
- Lee, Edward A. 2006. « Cyber-Physical Systems - Are Computing Foundations Adequate? » In *Position Paper for NSF Workshop On Cyber-Physical Systems*, 9. Austin, Texas.

- Lee, Jay, Behrad Bagheri, et Hung-An Kao. 2015. « A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems ». *Manufacturing Letters* 3 (janvier): 18-23. <https://doi.org/10.1016/j.mfglet.2014.12.001>.
- Lee, Robert M., Michael J. Assante, et Tim Conway. 2014. « German Steel Mill Cyber Attack ». SANS ICS 2014. http://ics3.sans.org/media/ICS-CPPE-case-Study-2-German-Steelworks_Facility.pdf.
- Leitão, Paulo, Armando Walter Colombo, et Stamatis Karnouskos. 2016. « Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges ». *Computers in Industry*, Emerging ICT concepts for smart, safe and sustainable industrial systems, 81 (septembre): 11-25. <https://doi.org/10.1016/j.compind.2015.08.004>.
- Leyden, John. 2008. « Polish teen derails tram after hacking train network ». The Register. janvier 2008. https://www.theregister.co.uk/2008/01/11/tram_hack/.
- Lhoste, P. 1991. « Surveillance des M.S.A.P. : les Atouts de la Modélisation de Comportement ». In . Paris, France.
- Li, Chunxiao, A. Raghunathan, et N. K. Jha. 2011. « Hijacking an insulin pump: Security attacks and defenses for a diabetes therapy system ». In *2011 IEEE 13th International Conference on e-Health Networking, Applications and Services*, 150-56. <https://doi.org/10.1109/HEALTH.2011.6026732>.
- Li, Weize, Lun Xie, Zulan Deng, et Zhiliang Wang. 2016. « False Sequential Logic Attack on SCADA System and Its Physical Impact Analysis ». *Computers & Security* 58 (mai): 149-59. <https://doi.org/10.1016/j.cose.2016.01.001>.
- Li, Z., D. Jin, C. Hannon, M. Shahidepour, et J. Wang. 2016. « Assessing and mitigating cybersecurity risks of traffic light systems in smart cities ». *IET Cyber-Physical Systems: Theory Applications* 1 (1): 60-69. <https://doi.org/10.1049/iet-cps.2016.0017>.
- Lin, Hui, Adam Slagell, Catello Di Martino, Zbigniew Kalbarczyk, et Ravishankar K. Iyer. 2013. « Adapting Bro into SCADA: Building a Specification-based Intrusion Detection System for the DNP3 Protocol ». In *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, 5:1-5:4. CSIIRW '13. New York, NY, USA: ACM. <https://doi.org/10.1145/2459976.2459982>.
- Linda, O., T. Vollmer, et M. Manic. 2009. « Neural Network based Intrusion Detection System for critical infrastructures ». In *2009 International Joint Conference on Neural Networks*, 1827-34. <https://doi.org/10.1109/IJCNN.2009.5178592>.
- Lipovsky, Robert, et Anton Cherepanov. 2016. « BlackEnergy Trojan Strikes Again: Attacks Ukrainian Electric Power Industry ». Eset - WeLiveSecurity. 4 janvier 2016. <https://www.welivesecurity.com/2016/01/04/blackenergy-trojan-strikes-again-attacks-ukrainian-electric-power-industry/>.
- . 2017. « Industroyer: Biggest Threat to Industrial Control Systems since Stuxnet ». Eset - WeLiveSecurity. 12 juin 2017. <https://www.welivesecurity.com/2017/06/12/industroyer-biggest-threat-industrial-control-systems-since-stuxnet/>.
- Loures, Eduardo Rocha. 2006. « Surveillance et diagnostic des phases transitoires des systèmes hybrides basés sur l'abstraction des dynamiques continues par réseau de Petri temporel flou », 181.

- Lu, Tianbo, Jiayi Lin, Lingling Zhao, Yang Li, et Yong Peng. 2015. « A Security Architecture in Cyber-Physical Systems: Security Theories, Analysis, Simulation and Application Fields ». *International Journal of Security and Its Applications* 9 (7): 1-16. <https://doi.org/10.14257/ijisa.2015.9.7.01>.
- Luallen, Matthew, et D. Harp. 2014. « Breaches on the Rise in Control Systems: A SANS Survey ». SANS Analyst Survey. <https://ics.sans.org/media/sans-ics-security-survey-2014.pdf>.
- L'Usine Nouvelle. 2018. « Chez Saint Gobain, “il y un avant et un après la cyber-attaque” - Economie ». *L'Usine Nouvelle*, 9 février 2018. <https://www.usinenouvelle.com/editorial/chez-saint-gobain-il-y-un-avant-et-un-apres-la-cyber-attaque.N651134>.
- Maheshwari, N., et H. Dagale. 2018. « Secure communication and firewall architecture for IoT applications ». In *2018 10th International Conference on Communication Systems Networks (COMSNETS)*, 328-35. <https://doi.org/10.1109/COMSNETS.2018.8328215>.
- Marangé, Pascale. 2009. « Synthèse et filtrage robuste de la commande pour des systèmes manufacturiers sûrs de fonctionnement ». Université de Reims.
- Markoff, John. 2009. « Old Trick Threatens the Newest Weapons ». *The New York Times*, 26 octobre 2009, sect. Science. <https://www.nytimes.com/2009/10/27/science/27trojan.html>.
- McIntyre, A., U. Lindqvist, B. Peterson, et Z. Tudor. 2012. « Host protection strategies for industrial control systems ». In *2012 IEEE Conference on Technologies for Homeland Security (HST)*, 87-92. <https://doi.org/10.1109/THS.2012.6459830>.
- McLaughlin, Stephen. 2013. « CPS: Stateful Policy Enforcement for Control System Device Usage ». In *Proceedings of the 29th Annual Computer Security Applications Conference*, 109-118. ACSAC '13. New York, NY, USA: ACM. <https://doi.org/10.1145/2523649.2523673>.
- . 2015. « Blocking Unsafe Behaviors in Control Systems Through Static and Dynamic Policy Enforcement ». In *Proceedings of the 52Nd Annual Design Automation Conference*, 55:1-55:6. DAC '15. New York, NY, USA: ACM. <https://doi.org/10.1145/2744769.2747913>.
- McLaughlin, Stephen, Charalambos Konstantinou, Xueyang Wang, Lucas Davi, Ahmad-Reza Sadeghi, Michail Maniatakis, et Ramesh Karri. 2016. « The Cybersecurity Landscape in Industrial Control Systems ». *Proceedings of the IEEE* 104 (5): 1039-57. <https://doi.org/10.1109/JPROC.2015.2512235>.
- McLaughlin, Stephen, et Saman Zonouz. 2014. « Controller-aware false data injection against programmable logic controllers ». In *Smart Grid Communications (SmartGridComm), 2014 IEEE International Conference on*, 848-853. IEEE. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=7007754.
- McLaughlin, Stephen, Saman A. Zonouz, Devin J. Pohly, et Patrick Drew McDaniel. 2014. « A Trusted Safety Verifier for Process Controller Code ». In *NDSS*. <http://www.stephenmclaughlin.org/ndss-2014.pdf>.
- Milinković, S. A., et L. R. Lazić. 2012. « Industrial PLC security issues ». In *2012 20th Telecommunications Forum (TELFOR)*, 1536-39. <https://doi.org/10.1109/TELFOR.2012.6419513>.

- Mitchell, Robert, et Ing-Ray Chen. 2014. « A Survey of Intrusion Detection Techniques for Cyber-Physical Systems ». *ACM Computing Surveys* 46 (4): 1-29. <https://doi.org/10.1145/2542049>.
- Mitchell, Robert, et Ray Chen. 2013. « Behavior-rule based intrusion detection systems for safety critical smart grid applications ». *IEEE Transactions on Smart Grid* 4 (3): 1254–1263. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6510000.
- Mohan, Sabin, Stanley Bak, Emiliano Betti, Heechul Yun, Lui Sha, et Marco Caccamo. 2013. « S3A: Secure system simplex architecture for enhanced security and robustness of cyber-physical systems ». In *Proceedings of the 2nd ACM international conference on High confidence networked systems*, 65–74. ACM. <http://dl.acm.org/citation.cfm?id=2461456>.
- Monostori, László. 2014. « Cyber-physical Production Systems: Roots, Expectations and R&D Challenges ». *Procedia CIRP, Variety Management in Manufacturing*, 17 (janvier): 9-13. <https://doi.org/10.1016/j.procir.2014.03.115>.
- Morris, Thomas, Rayford Vaughn, et Yoginder Dandass. 2012. « A Retrofit Network Intrusion Detection System for MODBUS RTU and ASCII Industrial Control Systems ». In , 2338-45. IEEE. <https://doi.org/10.1109/HICSS.2012.78>.
- Newman, Lily Hay. 2018. « Menacing Malware Shows the Dangers of Industrial System Sabotage ». WIRED. 2018. <https://www.wired.com/story/triton-malware-dangers-industrial-system-sabotage/>.
- Nguyen, Dang-Trinh. 2015. « Diagnostic en ligne des systèmes à événements discrets complexes: approche mixte logique/probabiliste ». Université Grenoble Alpes. <https://tel.archives-ouvertes.fr/tel-01227260/>.
- Nurika, O., A. H. B. Muhamad Aminz, A. S. B. A. Rahman, et M. N. B. Zakaria. 2012. « Review of various firewall deployment models ». In *2012 International Conference on Computer Information Science (ICCIS)*, 2:825-29. <https://doi.org/10.1109/ICCISci.2012.6297140>.
- ODVA. 2016. « EtherNet/IP - CIP on Ethernet Technology ».
- Pan, Shengyi, Thomas H. Morris, Uttam Adhikari, et Vahid Madani. 2013. « Causal Event Graphs Cyber-physical System Intrusion Detection System ». In *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, 40:1–40:4. New York, NY, USA: ACM. <https://doi.org/10.1145/2459976.2460022>.
- Papa, S., W. Casper, et S. Nair. 2012. « A transfer function based intrusion detection system for SCADA systems ». In *Homeland Security (HST), 2012 IEEE Conference on Technologies for*, 93-98. <https://doi.org/10.1109/THS.2012.6459831>.
- Papp, Dorottya, Zhendong Ma, et Levente Buttyan. 2015. « Embedded systems security: Threats, vulnerabilities, and attack taxonomy ». In *Privacy, Security and Trust (PST), 2015 13th Annual Conference on*, 145–152. IEEE. <http://ieeexplore.ieee.org/abstract/document/7232966/>.
- Peck, Daniel, et Dale Peterson. 2009. « Leveraging ethernet card vulnerabilities in field devices ». In *SCADA security scientific symposium*, 1–19.
- Pérès, François, Leïla Bouzaïene, Jean-Claude Bocquet, François Billy, André Lannoy, et Philippe Haïk. 2007. « Anticipating aging failure using feedback data and expert judgment ». *Reliability Engineering and System Safety* 92 (n° 2): 200-210. <https://doi.org/10.1016/j.ress.2005.11.063>.

- Pérès, François, et Daniel Noyes. 2003. « Evaluation of a maintenance strategy by the analysis of the rate of repair ». *Quality and Reliability Engineering International* 19 (n° 2): 129-48. <https://doi.org/10.1002/qre.515>.
- Peterson, James L. 1977. « Petri nets ». *ACM Computing Surveys (CSUR)* 9 (3): 223-252. <http://dl.acm.org/citation.cfm?id=356702>.
- Pétin, Jean-François. 2007. « Méthodes et modèles pour un processus sûr d'automatisation ». Habilitation à diriger des recherches, Nancy, France: Université Henri Poincaré – Nancy I. <https://tel.archives-ouvertes.fr/tel-00202431/document>.
- Petri, Carl Adam. 1966. « Communication with Automata », 97.
- Piètre-Cambacédès, L., et M. Bouissou. 2013. « Cross-fertilization between safety and security engineering ». *Reliability Engineering & System Safety* 110 (février): 110-26. <https://doi.org/10.1016/j.ress.2012.09.011>.
- Pin, Jean-Eric. 2006. « Algorithmique et Programmation. Automates finis. Chap. I/9 ». In *J. Akoka et I. Comyn-Wattiau. Encyclopédie de l'informatique et des systèmes d'information, Vuibert, pp.966-976, 2006*, pp.966-976. Vuibert. <hal-00143940>.
- Polityuk, Vukmanovic, et Jewkes. 2017. « Ukraine's Power Outage Was a Cyber Attack: Ukrenergo ». *Reuters*, 18 janvier 2017. <https://www.reuters.com/article/us-ukraine-cyber-attack-energy/kyev-power-outage-in-december-was-cyber-attack-ukrenergo-idUSKBN1521BA>.
- Ponomarev, S., et T. Atkison. 2016. « Industrial Control System Network Intrusion Detection by Telemetry Analysis ». *IEEE Transactions on Dependable and Secure Computing* 13 (2): 252-60. <https://doi.org/10.1109/TDSC.2015.2443793>.
- Poulsen, Kevin. 2003. « Slammer worm crashed Ohio nuke plant net ». *The Register*. 20 août 2003. https://www.theregister.co.uk/2003/08/20/slammer_worm_crashed_ohio_nuke/.
- Purdue Research Foundation. 1991. « A Reference Model For Computer Integrated Manufacturing (CIM): A Description from the Viewpoint of Industrial Automation ». Purdue Research Foundation.
- Ramadge, Peter J., et W. Murray Wonham. 1987. « Supervisory control of a class of discrete event processes ». *SIAM journal on control and optimization* 25 (1): 206-230. <http://epubs.siam.org/doi/abs/10.1137/0325013>.
- Rao, Nandan, Shubhra Srivastava, et K. S. Sreekanth. 2017. « PKI Deployment Challenges and Recommendations for ICS Networks ». *International Journal of Information Security and Privacy* 11 (2): 38-48. <https://doi.org/10.4018/IJISP.2017040104>.
- rconnon12. 2014. « The Original Logic Bomb ». *Cyber Warfare and Terrorism* (blog). 26 octobre 2014. <https://rconnon12.wordpress.com/2014/10/26/third/>.
- Reeves, Jason, Ashwin Ramaswamy, Michael Locasto, Sergey Bratus, et Sean Smith. 2012. « Intrusion detection for resource-constrained embedded control systems in the power grid ». *International Journal of Critical Infrastructure Protection* 5 (2): 74-83. <https://doi.org/10.1016/j.ijcip.2012.02.002>.
- Ren, X., R. D. Blanton, et V. G. Tavares. 2016. « A Learning-Based Approach to Secure JTAG Against Unseen Scan-Based Attacks ». In *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 541-46. <https://doi.org/10.1109/ISVLSI.2016.107>.
- « RISI - The Repository of Industrial Security Incidents ». 2016. 9 septembre 2016. http://www.risidata.com/Database/event_date/asc.

- Roberts, Paul F. 2005a. « Zotob Worms Target Windows 2000 Hole ». eWEEK. 2005. <http://www.eweek.com/security/zotob-worms-target-windows-2000-hole>.
- . 2005b. « Zotob, PnP Worms Slam 13 DaimlerChrysler Plants ». eweek. août 2005. <http://www.eweek.com/c/a/Security/Zotob-PnP-Worms-Slam-13-DaimlerChrysler-Plants>.
- Rrushy, Julian, et Kyoung-Don Kang. 2009. « Detecting Anomalies in Process Control Networks ». In *Critical Infrastructure Protection III*, édité par Charles Palmer et Sujeet Sheno, 151-65. Springer Berlin Heidelberg. http://link.springer.com.gate6.inist.fr/chapter/10.1007/978-3-642-04798-5_11.
- Rubio-Hernan, Jose, Luca De Cicco, et Joaquin Garcia-Alfaro. 2016. « Event-Triggered Watermarking Control to Handle Cyber-Physical Integrity Attacks ». In *Secure IT Systems: 21st Nordic Conference, NordSec 2016, Oulu, Finland, November 2-4, 2016. Proceedings*, édité par Billy Bob Brumley et Juha Röning, 3-19. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-47560-8_1.
- Said, Anis Ben. 2016. « Gestion dynamique des connaissances de maintenance pour des environnements de production de haute technologie à fort mix produit ». Phdthesis, Université Grenoble Alpes. <https://tel.archives-ouvertes.fr/tel-01342332/document>.
- Sandberg, C., et B. Hunter. 2017. « Cyber security primer for legacy process plant operation ». In *2017 Petroleum and Chemical Industry Technical Conference (PCIC)*, 97-102. <https://doi.org/10.1109/PCICON.2017.8188728>.
- Sayed-Mouchaweh, Moamar, éd. 2018. *Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems*. Cham: Springer International Publishing. <https://doi.org/10.1007/978-3-319-74962-4>.
- Sayegh, N., I. H. Elhajj, A. Kayssi, et A. Chehab. 2014. « SCADA Intrusion Detection System based on temporal behavior of frequent patterns ». In *MELECON 2014 - 2014 17th IEEE Mediterranean Electrotechnical Conference*, 432-38. <https://doi.org/10.1109/MELCON.2014.6820573>.
- Schuett, Carl, Jonathan Butts, et Stephen Dunlap. 2014. « An Evaluation of Modification Attacks on Programmable Logic Controllers ». *International Journal of Critical Infrastructure Protection* 7 (1): 61-68. <https://doi.org/10.1016/j.ijcip.2014.01.004>.
- Schwab, Wolfgang, et Mathieu Poujol. 2018. « The State of Industrial Cybersecurity 2018 ». *Kaspersky Lab*, juin, 33.
- Secrétariat Général de la Défense Nationale. 2009. « Communiquer de presse, création de l'ANSSI ». https://www.ssi.gouv.fr/uploads/IMG/pdf/ANSSI_Communique_de_presse.pdf.
- Sicard, Franck, J. M. Flaus, et Éric Zamaï. 2017. « Approche filtre basée sur la notion de distance pour la détection des cyberattaques ». In 15ème Colloque national AIP-Primeca. La Plagne, France.
- Sicard, Franck, Éric Zamaï, et J. M. Flaus. 2017a. « Distance Concept Based Filter Approach for Detection of Cyberattacks on Industrial Control Systems ». In 20th World Congress of the International Federation of Automatic Control (IFAC 2017), GdR MACS Young PhD Researchers - Open Invited Track of Extended Abstract. Toulouse, France.
- Sicard, Franck, Éric Zamaï, et Jean-Marie Flaus. 2017b. « Cyberdéfense des systèmes de contrôle-commande industriels : une approche par filtres basée sur la distance aux états critiques pour la sécurisation face aux cyberattaques ». In *César 2017 - La protection des données*

- face à la menace cyber*, 16p. Rennes, France. <https://hal.archives-ouvertes.fr/hal-01654260>.
- . 2018a. « Critical States Distance Filter Based Approach for Detection and Blockage of Cyberattacks in Industrial Control Systems ». In *Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems*, by M. Sayed-Mouchaweh, 117 (28p). Springer International Publishing. <https://www.springerprofessional.de/en/critical-states-distance-filter-based-approach-for-detection-and/15520488>.
- . 2018b. « Filter Based Approach with Temporal and Combinational Constraints for Cybersecurity of Industrial Control Systems ». In *10th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes, SAFEPROCESS 2018*, 8p. Warsaw, Poland: IFAC.
- . 2018c. « Cyberprotection des systèmes de contrôle-commande industriels : une approche à base de modèles pour détecter et réagir aux cyberattaques ». *Newsletter Observatoire FIC* (blog). mars 2018. <https://www.observatoire-fic.com/cyberprotection-des-systemes-de-contrôle-commande-industriels-une-approche-a-base-de-modeles-pour-detecter-et-reagir-aux-cyberattaques/>.
- . 2018d. « Cyberprotection of Industrial Control Systems: A Model-Based Approach to Detect and Respond to Cyberattacks ». *Newsletter Observatoire FIC* (blog). mars 2018. <https://www.observatoire-fic.com/cyberprotection-of-industrial-control-systems-a-model-based-approach-to-detect-and-respond-to-cyberattacks/>.
- Symantec Security Response. 2014. « Dragonfly ». Version 1.0. Symantec.
- Slay, Jill, et Michael Miller. 2006. « A security architecture for SCADA networks ». *ACIS 2006 Proceedings*, 12. <http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1008&context=acis2006>.
- . 2007. « Lessons Learned from the Maroochy Water Breach ». In *Critical Infrastructure Protection*, 73-82. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-75462-8_6.
- Sridhar, Siddharth, et Manimaran Govindarasu. 2014. « Model-Based Attack Detection and Mitigation for Automatic Generation Control ». *IEEE Transactions on Smart Grid* 5 (2): 580-91. <https://doi.org/10.1109/TSG.2014.2298195>.
- Sridhar, Siddharth, Adam Hahn, et Manimaran Govindarasu. 2012. « Cyber Physical System Security for the Electric Power Grid ». *Proceedings of the IEEE* 100 (1): 210-24. <https://doi.org/10.1109/JPROC.2011.2165269>.
- Stoler, Nimrod. 2018. « Anatomy of the Triton Malware Attack ». Threat Research Blog. CyberArk. 8 février 2018. <https://www.cyberark.com/threat-research-blog/anatomy-triton-malware-attack/>.
- Stouffer, Keith, Victoria Pillitteri, Suzanne Lightman, Marshall Abrams, et Adam Hahn. 2015. « Guide to Industrial Control Systems (ICS) Security ». NIST SP 800-82r2. National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.SP.800-82r2>.
- Symantec. 2005. « W32.Zotob.E | Symantec ». Symantec. 2005. <https://www.symantec.com/fr/fr/security-center/writeup/2005-081615-4443-99>.
- Taylor, A., N. Japkowicz, et S. Leblanc. 2015. « Frequency-based anomaly detection for the automotive CAN bus ». In *2015 World Congress on Industrial Control Systems Security (WCICSS)*, 45-49. <https://doi.org/10.1109/WCICSS.2015.7420322>.
- Thomas Reed. 2005. *At the Abyss: An Insider's History of the Cold War*. Presidio Press.

- Toublanc, Thomas, Sébastien Guillet, Florent de Lamotte, Pascal Berruet, et Vianney Lapotre. 2017. « Using a Virtual Plant to Support the Development of Intelligent Gateway for Sensors/Actuators Security ». *IFAC-PapersOnLine*, 20th IFAC World Congress, 50 (1): 5837-42. <https://doi.org/10.1016/j.ifacol.2017.08.541>.
- Trentesaux, Damien, Theodor Borangiu, et André Thomas. 2016. « Emerging ICT concepts for smart, safe and sustainable industrial systems ». *Computers in Industry*, Emerging ICT concepts for smart, safe and sustainable industrial systems, 81 (septembre): 1-10. <https://doi.org/10.1016/j.compind.2016.05.001>.
- Troelsen. 2007. *Idaho National Laboratory - Operation Aurora*. Youtube. <https://www.youtube.com/watch?v=bAWU5aMyAAo>.
- Urbina, David I., Jairo A. Giraldo, Alvaro A. Cardenas, Nils Ole Tippenhauer, Junia Valente, Mustafa Faisal, Justin Ruths, Richard Candell, et Henrik Sandberg. 2016. « Limiting the Impact of Stealthy Attacks on Industrial Control Systems ». In , 1092-1105. ACM Press. <https://doi.org/10.1145/2976749.2978388>.
- Urbina, David I, Jairo Giraldo, Alvaro A Cardenas, Junia Valente, Mustafa Faisal, Nils Ole Tippenhauer, Justin Ruths, Richard Candell, et Henrik Sandberg. 2016. « Survey and New Directions for Physics-Based Attack Detection in Control Systems ». NIST GCR 16-010. Gaithersburg, MD: National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.GCR.16-010>.
- Valdes, A., et S. Cheung. 2009. « Intrusion Monitoring in Process Control Systems ». In *42nd Hawaii International Conference on System Sciences, 2009. HICSS '09*, 1-7. <https://doi.org/10.1109/HICSS.2009.273>.
- Valette, R., L.A. Künzle, F. Girault, B. Pradin-Chézalviel, et J. Cardoso. 1996. « Vérification de contraintes temporelles pour des systèmes de contrôle commande à l'aide des réseaux de Petri ». In *Applications des méthodes formelles aux développement de systèmes en télécommunications, systèmes VLSI et systèmes de contrôle-commande temps réel*, p.255-267. Montréal. http://valetterobert.free.fr/papers.pri/logic.pri/montreal_texte.pdf.
- Vamosi, Robert. 2003. « SQL Slammer: How It Works--Prevent It ». ZDNet. 27 janvier 2003. <https://www.zdnet.com/article/sql-slammer-how-it-works-prevent-it/>.
- Villemeur, Alain. 1992. *Reliability, Availability, Maintainability and Safety Assessment, Volume 2, Assessment, Hardware, Software and Human Factors*. Wiley.
- Wang, Yong, Zhaoyan Xu, Jialong Zhang, Lei Xu, Haopei Wang, et Guofei Gu. 2014. « SRID: State Relation Based Intrusion Detection for False Data Injection Attacks in SCADA ». In *Computer Security - ESORICS 2014*, édité par Mirosław Kutylowski et Jaideep Vaidya, 401-18. Springer International Publishing. http://link.springer.com.gate6.inist.fr/chapter/10.1007/978-3-319-11212-1_23.
- Wardak, H., S. Zhioua, et A. Almulhem. 2016. « PLC access control: a security analysis ». In *2016 World Congress on Industrial Control Systems Security (WCICSS)*, 1-6. <https://doi.org/10.1109/WCICSS.2016.7882935>.
- Weiss, Joe. 2016. « Industrial Control System Cyber Security and the Critical Infrastructures ». *INSIGHT* 19 (4): 33-36.
- Xiao, Kun, Nianen Chen, Shangping Ren, Limin Shen, Xianhe Sun, Kevin Kwiatt, et Michael Macalik. 2007. « A workflow-based non-intrusive approach for enhancing the survivability of critical infrastructures in cyber environment ». In *Proceedings of the*

- Third International Workshop on Software Engineering for Secure Systems*, 4. IEEE Computer Society. <http://dl.acm.org/citation.cfm?id=1269064>.
- Xu, L. D., W. He, et S. Li. 2014. « Internet of Things in Industries: A Survey ». *IEEE Transactions on Industrial Informatics* 10 (4): 2233-43. <https://doi.org/10.1109/TII.2014.2300753>.
- Yang, D., A. Usynin, et J. W. Hines. 2006. « Anomaly-Based Intrusion Detection for SCADA Systems ». *Proceedings of the 5. International Topical Meeting on Nuclear Plant Instrumentation Controls, and Human Machine Interface Technology* 43 (47): 12-16. http://inis.iaea.org/Search/search.aspx?orig_q=RN:43118718.
- Yang, W., et Q. Zhao. 2014. « Cyber security issues of critical components for industrial control system ». In *Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference*, 2698-2703. <https://doi.org/10.1109/CGNCC.2014.7007593>.
- Yang, Y., T. Littler, S. Sezer, K. McLaughlin, et H. F. Wang. 2011. « Impact of cyber-security issues on Smart Grid ». In *2011 2nd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies (ISGT Europe)*, 1-7. <https://doi.org/10.1109/ISGTEurope.2011.6162722>.
- Yuan, Y., H. Yuan, L. Guo, H. Yang, et S. Sun. 2016. « Resilient Control of Networked Control System Under DoS Attacks: A Unified Game Approach ». *IEEE Transactions on Industrial Informatics* 12 (5): 1786-94. <https://doi.org/10.1109/TII.2016.2542208>.
- Zamaï, Éric. 1997. « Architecture de surveillance-commande pour les systèmes à événements discrets complexes ». Phdthesis, Université Paul Sabatier - Toulouse III. <https://tel.archives-ouvertes.fr/tel-00010078/document>.
- Zeller, Mark. 2011. « Myth or reality—Does the Aurora vulnerability pose a risk to my generator? » In *Protective Relay Engineers, 2011 64th Annual Conference for*, 130–136. IEEE. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6035612.
- Zetter, Kim. 2016. « Inside the Cunning, Unprecedented Hack of Ukraine's Power Grid ». *Wired*. 3 mars 2016. <https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/>.
- Zhang, X., Y. Du, L. Qi, et H. Sun. 2018. « An Approach for Repairing Process Models Based on Logic Petri Nets ». *IEEE Access*, 1-1. <https://doi.org/10.1109/ACCESS.2018.2843137>.
- Zhong, Ray Y., Xun Xu, Eberhard Klotz, et Stephen T. Newman. 2017. « Intelligent Manufacturing in the Context of Industry 4.0: A Review ». *Engineering* 3 (5): 616-30. <https://doi.org/10.1016/J.ENG.2017.05.015>.
- Zhou, Gan, Gautam Biswas, et Wenquan Feng. 2015. « A Comprehensive Diagnosis of Hybrid Systems for Discrete and Parametric Faults Using Hybrid I/O Automata ». *9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 2015* 48 (21): 143-49. <https://doi.org/10.1016/j.ifacol.2015.09.518>.
- Zhu, B., et S. Sastry. 2010. « Scada-specific intrusion detection/prevention systems: A survey and taxonomy ». *Proceedings of the First Workshop on Secure Control Systems (SCS)*.
- Zimmer, Christopher, Balasubramanya Bhat, Frank Mueller, et Sibin Mohan. 2010. « Time-based Intrusion Detection in Cyber-physical Systems ». In *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems*, 109–118. New York, NY, USA: ACM. <https://doi.org/10.1145/1795194.1795210>.

Zurawski, R., et MengChu Zhou. 1994. « Petri nets and industrial applications: A tutorial ». *IEEE Transactions on Industrial Electronics* 41 (6): 567-83. <https://doi.org/10.1109/41.334574>.

Glossaire

- AFD : Automate Fini Déterministe
- ANSSI : Agence Nationale de la Sécurité des Systèmes d'Information
- API : Automate Programmable Industriel, (voir PLC)
- CIM : Computer Integrated Manufacturing
- CPPS : Cyber-Physical Production System, Systèmes de Production Cyber-Physique
- CPS : Cyber-Physical System, Système Cyber-Physique
- DDOS : Distributed Denial of Service, Déni de service distribué
- DMZ : Demilitarized Zone, zone démilitarisée
- DNS : Directives Nationales de Sécurité
- DOS : Denial of Service, Déni de service
- EBIOS : Expression des Besoins et Identification des Objectifs de Sécurité
- ENISA : European Union Agency for Network and Information Security (UE)
- HIDS : Host Detection Intrusion System, Système de détection d'intrusion hôte
- ICPS : Industrial Cyber-Physical System, Système Cyber-Physique Industriel
- ICS : Industrial Control System, Système de Contrôle-Commande Industriel
- IDS : Intrusion Detection System, Système de Détection d'Intrusion
- IHM : Interface Homme-Machine
- IoT : Internet of Things, Internet des objets
- IT : Information Technologie (Informatique)
- LPM : Loi de Programmation Militaire
- MTU : Master Terminal Unit
- NIDS : Network Intrusion Detection System, Système de détection d'intrusion réseau
- NIST : National Institute of Standards and Technology (USA)
- OIV : Opérateur d'Importance Vitale
- OT : Operational Technology
- PC : Partie Commande (niveau 1 dans l'architecture ICS)
- PIV : Points d'Importance Vitale
- PKI : Public Key Infrastructure, infrastructure à clés publiques (ICP)
- PLC : Programmable Logic Controller (voir API)
- PO : Partie Opérative (niveau 0 dans l'architecture ICS)
- RdP : Réseau de Petri
- RTU : Remote Terminal Unit
- SAIV : Secteur d'Activités d'Importance Vitale
- SCADA : Supervisory Control And Data Acquisition
- SED : Systèmes à Evénements Discrets
- SI : Système d'information
- SIIV : Système d'Information d'Importance Vitale
- SIS : Safety Instrumented Systems, Systèmes Instrumentés de Sécurité
- ZIV : Zones d'Importance Vitale

Table des figures et des tableaux

Figure 1 : Évolution du nombre d'incidents répertoriés dans les systèmes industriels entre 2010 et 2015.....	17
Figure 2 : Répartition des secteurs industriels impactés pour l'année 2016	18
Figure 3 : Répartition des conséquences à la suite d'un incident de sécurité en 2017 et 2018 (Schwab et Poujol 2018).....	19
Figure 4 : Historique des principales attaques commises contre des environnements industriels au 21 ^{ème} siècle.....	20
Figure 5 : Illustration de l'attaque Stuxnet sur un système industriel.....	24
Figure 6 : Synthèse du périmètre d'action de la LPM	30
Figure 7 : Représentation d'un système cyber-physique (Leitão, Colombo, et Karnouskos 2016; Ashibani et Mahmoud 2017)	31
Figure 8 : Représentation d'un ICS sous forme fonctionnelle avec la norme CIM.....	34
Figure 9 : Architecture détaillée d'un Automate Programmable Industriel (Bolton 2009)	35
Figure 10 : Illustration des principales menaces et surfaces d'attaque (en rouge) dans un ICS (Niveau 2 - Supervision, Niveau 1 - Commande et Niveau 0 - Procédé).....	40
Figure 11 : Superposition des principales menaces et sources de défaillances dans un ICS (éclairage rouge : défaillance, drapeau pirate : menace)	42
Figure 12 : Illustration du principe de fonctionnement des principales solutions de sécurité utilisées dans l'IT	47
Figure 13 : Principe de fonctionnement d'une sonde IDS	48
Figure 14 : Principe de fonctionnement de l'approche par modèle de référence (M. Combacau et Courvoisier 1990).....	58
Figure 15 : Approche par Émulation (Holloway et Krogh 1991)	59
Figure 16 : Principe de fonctionnement de l'approche Filtre au sens de (Cruette, Bourey, et Gentina 1991) et (Lhoste 1991).....	60
Figure 17 : Principe de fonctionnement du filtre de commande.....	61
Figure 18 : Principe de fonctionnement du filtre de comptes rendus.....	61
Figure 19 : Localisation des filtres de l'approche S.A.F.E. dans l'architecture ICS	72
Figure 20 : Méthodologie et mise en application de l'approche S.A.F.E.....	75
Figure 21 : Méthodologie de conception des filtres de commande.....	83
Figure 22 : Répartition des états possibles dans un système de contrôle-commande	85
Figure 23 : Illustration des spécifications temporelles	88
Figure 24 : Illustration des propriétés graphiques des RdP – a) Modélisation d'une activité, b) Séquentialité, c) Parallélisme, d) Synchronisation, e) Exclusion mutuelle.....	91
Figure 25 : Représentation d'un automate fini déterministe.....	92
Figure 26 : Illustration des conditions d'arrêts de l'algorithme 1 (exploration des états en cours)...	96
Figure 27 : Illustration de la suppression d'un état déjà rencontré de la liste des états à explorer lors de la prochaine itération (état s_3'')	96

Figure 28 : Exemple d'illustration complet (l'état s44 est similaire à l'état s23 et n'est donc pas représenté)	97
Figure 29 : Illustration du calcul de la distance entre l'état courant et les états critiques pour un système à événements discrets (algorithme 2)	103
Figure 30 : Illustration de l'indicateur de trajectoire sur le système représenté en Figure 28 (ici, représentation de la trajectoire optimale)	108
Figure 31 : Illustration d'une sur-sollicitation sur un actionneur	111
Figure 32 : Trajectoires d'un système respectant la loi de commande (Filtre de commande) ..	117
Figure 33 : Trajectoires d'un système respectant la loi de commande (filtre de CR)	118
Figure 34 : Trajectoires d'un système avec un ordre menant à un état interdit (filtre de commande)	118
Figure 35 : Trajectoires d'un système avec un ordre menant à un état interdit (filtre de CR) ..	119
Figure 36 : Trajectoires d'un système avec un ordre menant à un état dangereux (filtre de commande)	119
Figure 37 : Trajectoires d'un système avec un ordre menant à un état dangereux (filtre de CR) ..	120
Figure 38 : Localisation des filtres dans la première version de l'approche S.A.F.E.	121
Figure 39 : Trajectoires d'un système avec un CR menant à un état dangereux (filtre de commande)	122
Figure 40 : Trajectoires d'un système avec un CR menant à un état dangereux (filtre de CR)	122
Figure 41 : Trajectoires d'un système avec une séquence menant à un état interdit (filtre de commande)	123
Figure 42 : Trajectoires d'un système avec une séquence menant à un état interdit (filtre de CR)	124
Figure 43 : Évolution des ordres envoyés à la partie opérative au cours du temps avec les FT correspondantes	125
Figure 44 : Évolution de la valeur des capteurs de la partie opérative au cours du temps avec les FT	125
Figure 45 : Évolution de la durée d'activation des ordres	126
Figure 46 : Étude des sur-sollicitations sur un système à 5 actionneurs (actionneurs 1 et 5 affectés)	127
Figure 47 : Récapitulatif des différents profils d'attaquants sélectionnés pour tester l'approche proposée	137
Figure 48 : Illustration du système à 3 cuves	139
Figure 49 : Espace de travail pour générer le modèle du filtre de commande sous Matlab	140
Figure 50 : Illustration de la loi de commande du système à 3 cuves et fonction correspondante sur Matlab	142
Figure 51 : Algorithme d'exploration des états (algorithme 1) sous Matlab	142
Figure 52 : Capture d'écran du temps de calcul pour obtenir le modèle du filtre de commande sous Eclipse	143
Figure 53 : Illustration du système à 5 cuves	144
Figure 54 : Illustration de la loi de commande du système à 5 cuves et fonction correspondante sous Matlab	147

Figure 55 : Capture d'écran du code Matlab permettant de lancer les simulations avec le filtre de commande.....	148
Figure 56 : Capture d'écran de l'algorithme de détection dans le filtre de commande.....	149
Figure 57 : Trajectoires du comportement normal du système à 5 cuves (filtre de commande).....	149
Figure 58 : Trajectoires du comportement normal du système à 5 cuves (filtre de comptes rendus).....	150
Figure 59 : Capture d'écran d'une partie de l'algorithme sous Matlab permettant de détecter l'attaque du scénario 1.....	150
Figure 60 : Trajectoires caractéristiques d'une attaque du scénario 1 (filtre de commande) ...	151
Figure 61 : Représentation de l'exécution temporelle des ordres du système à 5 cuves dans le cas d'un comportement normal	151
Figure 62 : Représentation de l'exécution temporelle des comptes rendus du système à 5 cuves lors d'un cycle de production normal (en couleur, fenêtres temporelles en noir) et d'une attaque (en rouge)	152
Figure 63 : Capture d'écran de l'algorithme de détection temporelle.....	153
Figure 64 : Représentation des trajectoires du filtre de commande pour le système à 5 cuves dans le cas d'une attaque discrète (faible occurrence d'anomalies).....	153
Figure 65 : Représentation de l'exécution temporelle des ordres du système à 5 cuves dans le cas d'une attaque sur les équipements.....	154
Figure 66 : Illustration de la plateforme industrielle utilisée pour évaluer l'approche S.A.F.E. en condition réelle	159
Figure 67 : Représentation de la loi de commande de la plateforme industrielle	162
Figure 68 : Trajectoires calculées par le filtre de commande et comparaison avec les données acquises dans le mode de production et avec une seule palette sur le convoyeur	163
Figure 69 : Trajectoires calculées par le filtre de comptes rendus et comparaison avec les données acquises dans le mode de production et avec une seule palette sur le convoyeur	164
Figure 70 : Trajectoires calculées par le filtre de commande et comparaison avec les données acquises dans le mode de production et avec plusieurs palettes sur le convoyeur ...	164
Figure 71 : Trajectoires calculées par le filtre de commande et comparaison avec les données acquises dans le mode initialisation	165
Figure 72 : Trajectoires calculées par le filtre de commande et comparaison avec le comportement face à une attaque simulée à partir des données acquises sur le comportement normal	165
Figure 73 : Illustration des propriétés graphiques des RdP – a) Modélisation d'une activité, b) Séquentialité, c) Parallélisme, d) Synchronisation, e) Exclusion mutuelle.....	196
Figure 74 : Illustration du système à 6 cuves.....	198
Figure 75 : Illustration de la loi de commande du système à 6 cuves	203
Figure 76 : Illustration du système avec 1 chariot et 1 aiguillage.....	203
Figure 77 : Illustration de la loi de commande du système à 1 chariot – 1 aiguillage.....	206
Figure 78 : Espace de travail pour générer le modèle du filtre de commande sous Matlab.....	208
Figure 79 : Capture d'écran d'un exemple de modèle de partie opérative	208
Figure 80 : Capture d'écran du code permettant d'obtenir les états optimaux du système.....	209

Figure 81 : Capture d'écran de la fonction permettant le calcul de la plus courte distance aux états interdits	209
Figure 82 : Capture d'écran du code Java adapté de Matlab permettant l'exploration des états du système.....	210
Tableau 1 : Secteurs d'Activités d'Importance Vitale et Ministères associés	29
Tableau 2 : Principales menaces dans un système de contrôle-commande industriel avec leurs localisations dans l'architecture ICS et les références correspondantes.....	43
Tableau 3 : Récapitulatif des spécificités de l'OT par rapport à l'IT.....	68
Tableau 4 : Principales notations utilisées dans le manuscrit	86
Tableau 5 : Résultats de l'algorithme 2 sur l'exemple développé sur la Figure 29.....	105
Tableau 6 : Regroupement des contextes menant à des états interdits du système illustré en Figure 28	116
Tableau 7 : Récapitulatif des attaques considérées pour notre approche.....	129
Tableau 8 : Récapitulatif des effets des actions sur le système à 3 cuves.....	140
Tableau 9 : Récapitulatif des conditions menant à l'état interdit de débordement (Système à 3 cuves)	141
Tableau 10 : Récapitulatif des effets des actions sur le système à 5 cuves.....	145
Tableau 11 : Récapitulatif des conditions menant aux états interdits de débordement (Système à 5 cuves)	146
Tableau 12 : Récapitulatif des systèmes simulés lors des tests de l'approche S.A.F.E. (Intel Core i5-63000 - CPU 2,40 GHz 2,50 GHz – 8Go RAM)	155
Tableau 13 : Récapitulatif des effets des actions sur le système	161
Tableau 14 : Récapitulatif des conditions menant à l'état interdit de débordement.....	161
Tableau 15 : Récapitulatif des effets des actions sur le système à 6 cuves.....	200
Tableau 16 : Récapitulatif des conditions menant aux états interdits de débordement (Système à 5 cuves)	201
Tableau 17 : Récapitulatif des conditions menant à l'état interdit de déraillement.....	205
Tableau 18 : Récapitulatif des effets des actions sur le système 1 chariot – 1 aiguillage	205

Liste des publications

1. Contribution à ouvrage

Sicard, F., Zamaï, E., Flaus, J.-M., 2018. Critical States Distance Filter Based Approach for Detection and Blockage of Cyberattacks in Industrial Control Systems, in: *Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems* by M. Sayed-Mouchaweh, 117 (28p). Springer International Publishing.

2. Revue

Sicard, F., Zamaï, E., Flaus, J.-M., 2018. An approach based on behavioral models and critical states distance notion for improving cybersecurity of industrial control systems, in: *Reliability Engineering & System Safety (en soumission)*

3. Conférences

Sicard, F., Escudero, C., Zamaï, É., 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: *IEEE CSNET (2nd Cyber Security In Networking Conference) 2018*. Paris, France

C. Escudero, F. Sicard, and E. Zamaï, "Process-Aware Model based IDSs for Industrial Control Systems Cybersecurity: Approaches, Limits and Further Research," in *Proceeding 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation*, Torino, Italy, 2018, p. 8.

Sicard, F., Zamaï, É., Flaus, J.M., 2018. Filter based approach with temporal and combinational constraints for Cybersecurity of Industrial Control Systems, in: *10th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS'18)*, Aug 2018, Warsaw, Poland. pp. 96-103.

Sicard, F., Zamaï, É., Flaus, J.-M., 2017. Cyberdéfense des systèmes de contrôle-commande industriels : une approche par filtres basée sur la distance aux états critiques pour la sécurisation face aux cyberattaques, in: *C&esar 2017 - La Protection Des Données Face à La Menace Cyber*. Rennes, France.

Sicard, F., Zamaï, É., Flaus, J.M., 2017. Distance Concept Based Filter Approach for Detection of Cyberattacks on Industrial Control Systems. Presented at the IFAC World Congress, GdR MACS Young PhD Researchers - Open Invited Track of Extended Abstract, Toulouse, France.

Sicard, F., Flaus, J.M., Zamaï, É., 2017. Approche filtre basée sur la notion de distance pour la détection des cyberattaques. Presented at the 15ème Colloque National AIP-Primeca, La Plagne, France.

4. Séminaires

Sicard, F., Zamaï, É., Flaus, J.M., March 2018. Cyberprotection des systèmes de contrôle commande industriels : une approche à base de modèles pour détecter et réagir aux cyberattaques – Newsletter Observatoire FIC

- **Version française** : <https://www.observatoire-fic.com/cyberprotection-des-systemes-de-contrôle-commande-industriels-une-approche-a-base-de-modeles-pour-detecter-et-reagir-aux-cyberattaques/>
- **Version anglaise** : <https://www.observatoire-fic.com/cyberprotection-of-industrial-control-systems-a-model-based-approach-to-detect-and-respond-to-cyberattacks/>

Sicard, F., Zamaï, É., Flaus, J.M., March 2018. Cyberdéfense des systèmes de contrôle commande industriels : une approche par filtre basée sur la notion de trajectoire pour la détection et la protection face aux cyberattaques – Presented at Séminaire S3 – GDR MACS, <http://gt-s3.cran.univ-lorraine.fr/reunions.html>

Sicard, F., Zamaï, É., Flaus, J.M., January 2018. Approche par filtre basée sur la distance aux états critiques pour la sécurisation des systèmes cyber-physiques face aux cyberattaques – Presented at Séminaire ISA France, « Cybersécurité de l'industrie, des infrastructures, des bâtiments... Nouvelles vulnérabilités, nouvelles attaques, nouvelles solutions »

Annexes

1. Annexe 1 : Les réseaux de Petri

Les réseaux de Petri (RdP) ont été définis pour la première fois par (Petri 1966; Peterson 1977) en 1966. Leurs propriétés à la fois graphiques et mathématiques permettent de fournir un environnement uniforme pour modéliser, concevoir et analyser des systèmes industriels (Zurawski et Zhou 1994; Zhang et al. 2018; Chaillet, Combacau, et Courvoisier 1993). Les RdP permettent notamment de mettre en évidence la séquentialité, les parallélismes, les synchronisations et les exclusions mutuelles comme expliqué dans (Zurawski et Zhou 1994; Murata 1989; Michel Combacau, Esteban, et Nketsa 2005). La Figure 73 illustre les différentes propriétés citées ci-dessus en représentation graphique ainsi que la définition d'une activité avec les RdP, indispensables à la modélisation de la commande (Michel Combacau, Esteban, et Nketsa 2005). Les RdP sont des réseaux places/transitions où, dans le cas de la modélisation de la commande, les places représentent les actions en cours d'exécution et les transitions correspondent aux comptes rendus envoyés par les capteurs. Les places contiennent un nombre entier de jetons, appelé marquage, représentant l'état du système et, dans le cas présent, l'état de la loi de commande. Le marquage initial est appelé M_0 . Les arcs permettent de relier les places et les transitions. Un nombre entier peut leur être affecté, appelé poids de l'arc, qui permet aux transitions d'être franchies si les places en amont possèdent un nombre de jetons supérieur ou égal. On parle de transitions sensibilisées.

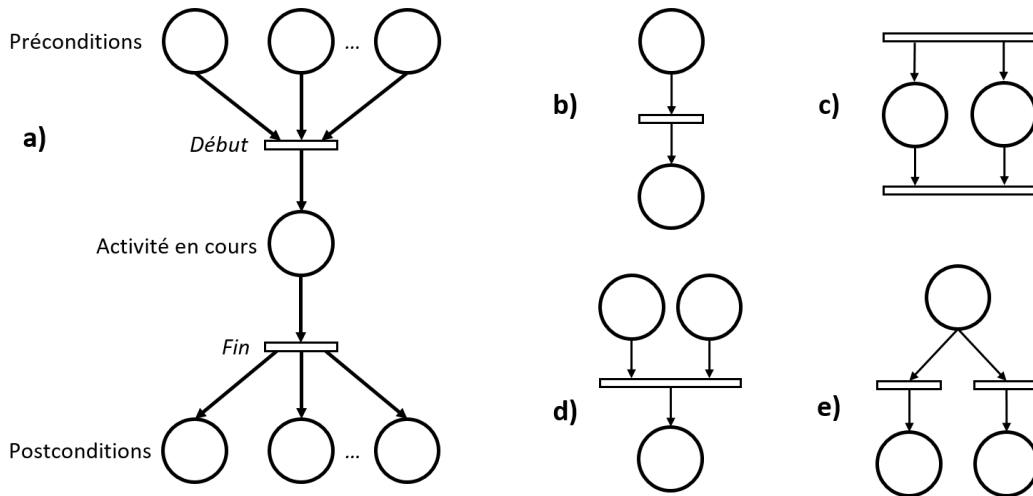


Figure 73 : Illustration des propriétés graphiques des RdP – a) Modélisation d'une activité, b) Séquentialité, c) Parallélisme, d) Synchronisation, e) Exclusion mutuelle

Au-delà d'une représentation graphique rendant aisée leur manipulation par l'homme, les réseaux de Petri peuvent être manipulés sous forme mathématique. Ils sont ainsi définis comme un quadruplet R (Ladet et Demongodin 2009) tel que montré dans l'équation (i) :

$$R = (P, T, Pré, Post) \quad (i)$$

P représente l'ensemble fini des places et T fait référence à l'ensemble fini de transitions, qui est disjoint de P . La matrice d'incidence avant notée $Pré$ est définie comme $Pré : P \times T \rightarrow \mathbb{N}$. De la même manière, la matrice d'incidence arrière notée $Post$ se définit comme $Post : P \times T \rightarrow \mathbb{N}$.

N. Ces matrices permettent de représenter le poids des arcs de manière mathématique. La matrice d'incidence W est calculée à partir des matrices $Pré$ et $Post$ tel que détaillé dans l'équation (ii) :

$$W = Post - Pré \quad (ii)$$

Ces deux représentations, graphique et mathématique, permettent de vérifier un certain nombre de propriétés afin de s'assurer qu'il n'y a aucune erreur sur le modèle de commande généré. Ainsi, comme décrit dans (Zurawski et Zhou 1994) et (Michel Combacau, Esteban, et Nketsa 2005), il est possible de s'assurer que le réseau de Petri est :

- Borné. Quel que soit le marquage M , accessible à partir du marquage initial M_0 , et quelle que soit la place P_i , le nombre de jetons est inférieur ou égal à une valeur k positive. Le réseau est alors k -borné et dans le cas où $k = 1$, le réseau est sauf. L'équation (iii) illustre cette condition. Cette vérification interdit les boucles générant une infinité de jetons lorsqu'elles sont parcourues,

$$\forall M \text{ et } \forall P_i \in P : M(P_i) \leq k \quad (iii)$$

- Vivant. Un RdP est vivant si et seulement si toutes ses transitions T_j le sont également, autrement dit, s'il existe pour chaque transition une séquence de franchissement Seq aboutissant à un marquage M permettant son franchissement. L'équation (iv) détaille cette définition. Cette analyse permet de s'assurer que toutes les places peuvent être atteintes et que, potentiellement, l'intégralité du code pourra être exécutée. Ainsi, nous garantissons l'atteignabilité, l'absence de blocage et de zone morte,

$$\forall M \text{ et } \forall P_i \in P \exists Seq : M \xrightarrow{Seq} M' \text{ tel que } T_j \subset Seq \quad (iv)$$

- Réinitialisable. Un RdP est réinitialisable, ou propre, si et seulement s'il existe toujours un chemin, aussi appelé séquence Seq , permettant d'atteindre le marquage initial M_0 à partir de n'importe quel marquage M du graphe. L'équation (v) illustre cette définition. Cet aspect est important dans la mesure où la commande a pour vocation de se répéter.

$$\forall M, \exists Seq : M \xrightarrow{Seq} M_0 \quad (v)$$

2. Annexe 2 : Exemples de simulation

Cette annexe complète la Partie 3 – Chapitre 7 – 2 en présentant deux exemples de simulation supplémentaires sur lesquels l'approche développée a été testée. Comme pour les systèmes à 3 et 5 cuves, les systèmes seront présentés, puis, la méthodologie développée dans nos travaux sera appliquée. Nous noterons que la programmation de ces exemples ainsi que les fonctions utilisées sont identiques à celles présentées dans le manuscrit.

A. Système à 6 cuves

Le système est illustré en Figure 74 et est composé de six cuves. Trois d'entre elles C_1 , C_2 et C_6 sont considérées comme étant de capacités infinies et contiennent des produits différents. La cuve C_1 peut déverser son produit dans les réservoirs de mélange C_3 , C_4 et C_5 via les vannes de remplissage V_3 , V_5 et V_9 . La cuve C_2 , quant à elle, est connectée avec les réservoirs C_3 et C_4 respectivement avec les vannes V_4 et V_6 . Enfin, La cuve C_6 est en capacité de remplir le réservoir C_4 par la vanne V_{11} . Le remplissage de ces réservoirs de mélange par les vannes V_3, V_4, V_5, V_6, V_9 et V_{11} est conditionné par l'ouverture au préalable des vannes V_1, V_2 et V_{10} . Les cuves C_1, C_3 et C_4 sont en mesure de déverser leurs contenus dans la cuve C_5 par l'intermédiaire respectivement des vannes V_9, V_7 et V_8 . Ces deux dernières vannes assurent par ailleurs la vidange respectivement des cuves C_3 et C_4 . La vanne V_{12} vidange la cuve C_5 .

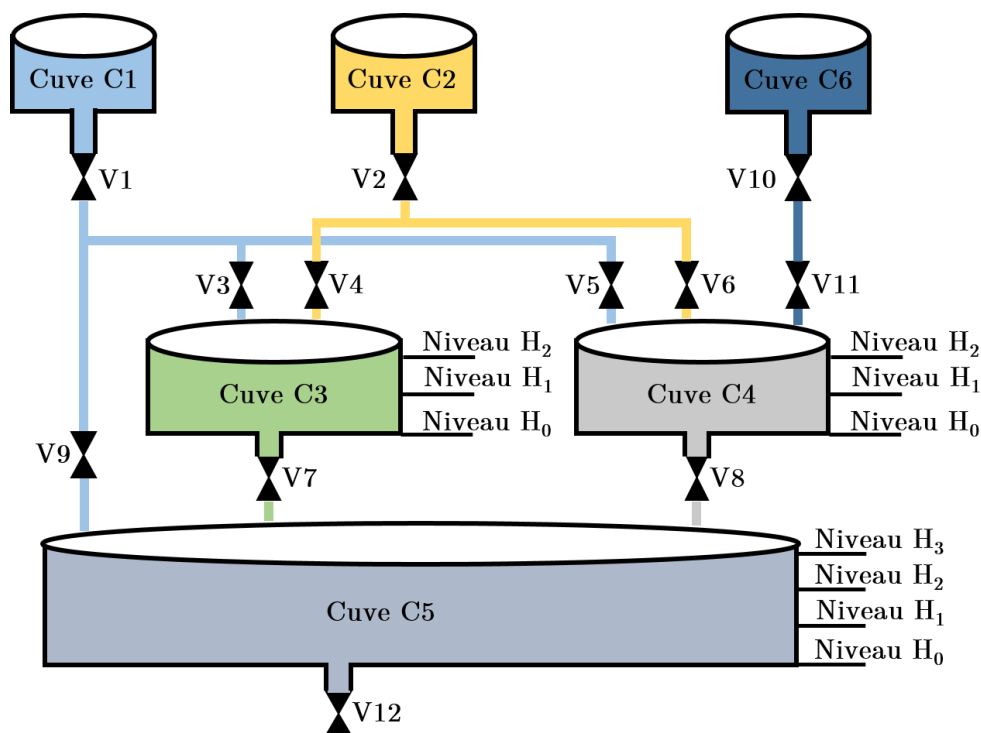


Figure 74 : Illustration du système à 6 cuves

Des capteurs de niveau sont placés dans chacun des réservoirs et permettent de connaître la quantité de produit dans les cuves de mélange. L'appartenance du capteur à la cuve dont il fait la mesure est indiquée en exposant dans la notation adoptée. Douze capteurs de niveau sont utilisés dans ce système :

- $H_0^{C_3}, H_0^{C_4}$ et $H_0^{C_5}$ sont les capteurs de fin de cuve. Ils indiquent un remplissage lorsqu'ils prennent la valeur 1 et une vidange totale de la cuve quand la valeur est nulle.
- $H_1^{C_3}, H_1^{C_4}, H_1^{C_5}$ et $H_2^{C_5}$ sont localisés à un niveau intermédiaire des cuves de mélange.
- $H_2^{C_3}, H_2^{C_4}$ et $H_3^{C_5}$ sont les capteurs de niveau haut des cuves C_3, C_4 et C_5 .

Les spécifications de la loi de commande imposent de démarrer un cycle de production avec toutes les cuves de mélange vides donc avec $H_0^{C_3}, H_0^{C_4}$ et $H_0^{C_5}$ nuls. Comme pour les exemples présentés dans le Chapitre 7 – 2, le remplissage d'une cuve se fait jusqu'à son capteur de niveau haut ($H_2^{C_3} H_2^{C_4} H_3^{C_5}$). Ainsi, la cuve C_3 est remplie par la cuve C_1 et C_2 avec l'ouverture des vannes V_1, V_3 et V_2, V_4 . La cuve C_4 est remplie par les cuves C_1, C_2 et C_6 avec l'ouverture des vannes V_1, V_5 et V_2, V_6 jusqu'au capteur de niveau $H_1^{C_4}$ puis des vannes V_{10}, V_{11} jusqu'au capteur de niveau $H_2^{C_4}$. Ensuite, la cuve C_5 est remplie par la cuve C_1 avec les vannes V_1, V_9 jusqu'au capteur $H_1^{C_5}$, puis C_3 avec la vanne V_7 jusqu'au capteur $H_2^{C_5}$ et enfin C_4 avec la vanne V_8 jusqu'au capteur $H_3^{C_5}$. La vidange de la cuve C_5 est alors réalisée par l'ouverture de la vanne V_{12} . Un nouveau cycle de production peut alors démarrer.

Considérons que l'analyse de risques a permis d'identifier les zones critiques pour la partie opérative (débordement de la cuve C_3, C_4 ou C_5) qui, pour ce système, peuvent être atteintes lorsque les niveaux des cuves C_3, C_4 ou C_5 sont supérieurs aux capteurs de niveau haut $H_2^{C_3}, H_2^{C_4}$ ou $H_3^{C_5}$. Afin de caractériser ces événements redoutés dans notre modélisation, nous identifions les paramètres nécessaires et suffisants pour décrire notre système (Étape 1). Cette étape s'effectue à l'aide d'un expert qui détermine d'une part les actions qui agissent sur le niveau de la cuve et, d'autre part, les capteurs qui indiquent ce niveau afin de pouvoir caractériser l'état de débordement. Ainsi, à l'issue de cette étape, nous avons :

- Vecteur des états $s \in S$ qui représente le comportement normal du système puisque les défaillances ne sont pas modélisées. Il se compose des valeurs des capteurs de niveau et des valeurs de commandes des vannes tel que défini dans l'équation (vi) :

$$s = [\text{Valeurs Capteurs} ; \text{Valeurs Actionneurs}] = (H_i^{C_3}, H_j^{C_4}, H_k^{C_5}, V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9, V_{10}, V_{11}) \quad (\text{vi})$$

$$\begin{aligned} \text{Avec : } H_i^{C_3} &\in \{0, .2\}, H_j^{C_4} \in \{0, .2\}, H_k^{C_5} \in \{0, .3\}, V_1 \in \{0, 1\}, V_2 \in \{0, 1\}, V_3 \in \{0, 1\}, V_4 \\ &\in \{0, 1\}, V_5 \in \{0, 1\}, V_6 \in \{0, 1\}, V_7 \in \{0, 1\}, V_8 \in \{0, 1\}, V_9 \in \{0, 1\}, V_{10} \\ &\in \{0, 1\}, V_{11} \in \{0, 1\}, V_{12} \in \{0, 1\} \end{aligned}$$

Dans l'équation (vi), $H_i^{C_3}$ indique quel capteur de niveau de la cuve C_3 est actif (0 aucun capteur n'est activé, 1 le capteur H_1 est activé, 2 le capteur H_2 est activé), $H_j^{C_4}$ et $H_k^{C_5}$ sont les équivalents pour les cuves C_4 et C_5 . Pour ce système, l'ensemble des états possibles « S » est composé d'au plus 147459 états et est noté S_{147459} .

- Vecteur actions $a \in A$ qui représente les actions pouvant être exécutées sur le système telles que $a = a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}$ avec $a_i \in \{0 \text{ ferme vanne } V_i, 1 \text{ ouvre vanne } V_i\} i \in \{1..12\}$. Cet ensemble A est composé de 4096 actions et est nommé dans la suite A_{4096} .

- Ensemble des états atteignables $S_{Reach} \subset S$ qui contient 46376 états dont 13 états optimaux (obtenus à l'aide du modèle de commande présenté en Figure 75), 3 états interdits et 46360 états dangereux $S_{Dan} = S_{Reach}(S_{Opt} \cap S_{Pro})$.
- Ensemble des états inatteignables $S_{Unreach} = S \setminus S_{Reach}$ qui contient donc 101083 états.

Un modèle de la partie commande (Étape 2) est développé sur la base des spécifications de la loi de commande et est représenté en Figure 75. De la même manière, le modèle de la partie opérative (Étape 2) est construit via l'équation (vii) à l'aide de l'automate M :

$$M = \{S_{46376}, A_{4096}, \delta, s_0, S_{final}\} \quad (vii)$$

L'ensemble S_{46376} représente l'ensemble des états atteignables pour le système, A_{4096} est l'ensemble des ordres qui peuvent être appliqués sur le système (toutes les combinaisons possibles), δ est la fonction de transition qui permet de faire évoluer le système et dont les effets sont synthétisés sur le Tableau 10, s_0 est l'état initial du système défini par $s_0 = zeros(1,12)$, S_{final} représente l'ensemble des états finaux de notre automate et donc nos conditions d'arrêts (S_{pro} , s_0 et S_{prev}). Le Tableau 11 regroupe les conditions sur le couple État/Action qui conduisent à l'état interdit de débordement.

Tableau 15 : Récapitulatif des effets des actions sur le système à 6 cuves

Ordre $a \in A_{4096}$	État $s \in S_{46376}$ du système avant l'action $a \in A_{4096}$	Exception / Commentaire
Ouvrir la vanne V_3, V_4, V_5, V_6, V_9 ou V_{11} sans ouvrir la vanne V_1, V_2 ou V_{10} ($a = [0 \ 0 \ 0 \ \times \dots \times]$)	$s = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \times \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a$ $= [H_i^{C_3} H_j^{C_4} H_k^{C_5} \ 0 \ 0 \ 0 \ \times \times \times \times \times \times \times \times \times]$	Lorsqu'un ordre d'ouverture de vannes V_3, V_4, V_5, V_6, V_9 ou V_{11} est envoyé sans l'ouverture des vannes V_1, V_2 ou V_3 correspondantes alors le système reste dans l'état.
Ouvrir une vanne V_3, V_4, V_5, V_6, V_9 ou V_{11} avec ouvrir la vanne V_1, V_2 ou V_3 correspondante $a_1 = [1 \ 0 \ 0 \ 1 \ 0 \dots 0]$.. $a_6 = [0 \ 0 \ 0 \ 0 \dots 0 \ 1 \ 1 \ 0]$	$s = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \times \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a_1$ avec $s' = [H_i^{C_3} + 1 \ H_j^{C_4} H_k^{C_5} \ 1 \ 0 \ 0 \ 1 \ 0 \dots 0]$	Lorsqu'une seule des vannes de remplissage est ouverte alors le niveau $H_i^{C_3}$, $H_j^{C_4}$ ou $H_k^{C_5}$ augmente d'une unité détectée par un capteur de niveau et spécifiée par son indice. Les cuves C_3, C_4 et C_5 peuvent être remplies en simultanée.
Ouvrir les vannes V_3 et V_4 avec ouvrir les vannes V_1 et V_2 $a_7 = [1 \ 1 \ 0 \ 1 \ 1 \ \times \dots \times]$	$s = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \times \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a_7$ $s' = [H_i^{C_3} + 2 \ H_j^{C_4} H_k^{C_5} \ 1 \ 1 \ 0 \ 1 \ 1 \ \times \dots \times]$	Lorsque les vannes V_3 et V_4 sont ouvertes alors la cuve C_3 se remplit plus vite et le niveau $H_i^{C_3}$ augmente de deux capteurs de niveau.

Ouvrir les vannes V_5 et V_6 avec ouvrir les vannes V_1 et V_2 $a_8 = [1 \ 1 \ 0 \ \times \times \ 1 \ 1 \ \times \dots \times]$	$s = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a_8$ $= [H_i^{C_3} H_j^{C_4} + 2 \ H_k^{C_5} \ 1 \ 1 \ 0 \ \times \times \ 1 \ 1 \ \times \dots \times]$	Lorsque les vannes V_5 et V_6 sont ouvertes alors la cuve C_4 se remplit plus vite et le niveau $H_i^{C_4}$ augmente de deux capteurs de niveau. <i>Le comportement est identique pour l'ouverture des vannes V_5, V_{11} et V_6, V_{11}.</i>
Ouvrir la vanne de vidange V_7 $(a_{11} = [\times \times \times \times \times \times \ 1 \ \times \dots \times])$	$s = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a_{11}$ avec $s' =$ $\begin{cases} [H_i^{C_3} - 1 \ H_j^{C_4} H_k^{C_5} + 1 \ \times \dots \times \ 1 \ \times \dots \times] \\ \quad si \ H_i^{C_3} > 0 \\ [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \dots \times \ 1 \ \times \dots \times] si \ H_i^{C_3} = 0 \end{cases}$	Lorsque la vanne de vidange V_7 est ouverte alors la vidange de la cuve C_3 est déclenchée (peu importe l'activation des vannes de remplissage V_3 et V_4). <i>Le comportement est identique pour l'ouverture de la vanne de vidange V_8 de la cuve C_4. Dans ce cas, l'effet portera sur le capteur $H_j^{C_4}$.</i>
Ouvrir la vanne V_{12} $(a_{12} = [\times \dots \times \ 1])$	$s = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a_{12}$ avec $s' =$ $\begin{cases} [H_i^{C_3} H_j^{C_4} H_k^{C_5} - 1 \ \times \dots \times \ 1] si \ H_k^{C_5} > 0 \\ [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \dots \times \ 1] si \ H_k^{C_5} = 0 \end{cases}$	Lorsque la vanne V_{12} est ouverte la vidange de la cuve C_5 est déclenchée.
Aucun ordre $(a_{4096} = [0 \dots 0])$	$s = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \times \times \times \times \times \times \times \times \times \times]$ $s' = \delta \ s, a_{4096} = [H_i^{C_3} H_j^{C_4} H_k^{C_5} \ 0 \ 0 \ 0 \ 0 \ 0]$	Lorsqu'aucun ordre d'ouverture de vannes n'est envoyé le système reste dans l'état dans lequel il était.

Comme pour les systèmes précédents, ces effets, dans leur ensemble, sont cumulables les uns avec les autres.

Tableau 16 : Récapitulatif des conditions menant aux états interdits de débordement (Système à 5 cuves)

État interdit S_{Pro}	Ordre $a \in A_{32}$	État $s \in S_{46376}$ du système avant l'action $a \in A_{4096}$
Débordement de la cuve C_3	Ouvrir la vanne V_3 (avec la vanne V_1 ouverte) ou V_4 (avec la vanne V_2 ouverte) $[1 \ 0 \ 1 \ 0 \ \times \dots \times]$ ou $[0 \ 1 \ 0 \ 1 \ \times \dots \times]$	Capteur $H_2^{C_3}$ activé : $s = [2 \ \times \dots \times]$

	Ouvrir les vannes V_3 et V_4 (avec les vannes V_1 et V_2 ouvertes) [1 1 1 1 $\times \dots \times$]	Capteurs $H_1^{C_3}$ ou $H_2^{C_3}$ activés : $s = [2 \times \dots \times]$ ou $s = [1 \times \dots \times]$
Débordement de la cuve C_4	Ouvrir la vanne V_5 (avec la vanne V_1 ouverte) ou V_6 (avec la vanne V_2 ouverte) ou V_{11} (avec la vanne V_{10} ouverte) [1 $\times \times \times$ 1 0 $\times \times \times \times$ 0 $\times$] ou [1 $\times \times$ 0 1 $\times \times \times \times$ 0 $\times$] ou [$\times \times \times \times$ 0 0 $\times \times \times$ 1 1 $\times$]	Capteur $H_2^{C_4}$ activé : $s = [\times 2 \times \dots \times]$
	Ouvrir les vannes V_5 et V_6 (avec les vannes V_1 et V_2 ouvertes) ou V_5 et V_{11} (avec les vannes V_1 et V_{10} ouvertes) ou V_6 et V_{11} (avec les vannes V_2 et V_{10} ouvertes) [1 1 $\times \times$ 1 1 $\times \times \times \times$ 0 $\times$] ou [1 $\times \times \times$ 1 0 $\times \times \times$ 1 1 $\times$] ou [$\times$ 1 $\times \times$ 0 1 $\times \times \times$ 1 1 $\times$]	Capteurs $H_1^{C_4}$ ou $H_2^{C_4}$ activés : $s = [\times 2 \times \dots \times]$ ou $s = [\times 1 \times \dots \times]$
	Ouvrir les vannes V_5, V_6 et V_{11} (avec les vannes V_1, V_2 et V_{10} ouvertes)	Quel que soit la valeur du Capteur $H_2^{C_4}$
Débordement de la cuve C_5	Ouvrir la vanne V_7 avec $H_i^{C_3} > 1$ ou ouvrir la vanne V_8 avec $H_j^{C_4} > 1$ ou ouvrir la vanne V_9 (avec la vanne V_1 ouverte)	Capteur $H_3^{C_5}$ activé : $s = [\times \times 3 \times \dots \times]$
	Ouvrir la vanne V_7 avec $H_i^{C_3} > 1$ et ouvrir la vanne V_8 avec $H_j^{C_4} > 1$ ou ouvrir la vanne V_7 avec $H_i^{C_3} > 1$ et ouvrir la vanne V_9 (avec la vanne V_1 ouverte) ou ouvrir la vanne V_8 avec $H_j^{C_4} > 1$ et ouvrir la vanne V_9 (avec la vanne V_1 ouverte) ou ouvrir la vanne V_7 avec $H_i^{C_3} > 1$ et ouvrir la vanne V_8 avec $H_j^{C_4} > 1$ et ouvrir la vanne V_9 (avec la vanne V_1 ouverte)	Capteurs $H_1^{C_5}, H_2^{C_5}$ ou $H_3^{C_5}$ activés : $s = [\times \times 3 \times \dots \times]$ ou $s = [\times \times 2 \times \dots \times]$ ou $s = [\times \times 1 \times \dots \times]$

Pour conclure l'étape 2, l'algorithme de génération du modèle du filtre de commande est exécuté sur la base des modèles construits ce qui conduit à l'obtention de 46376 états atteignables et 49364 contextes menant à des états interdits. Le temps de calcul pris par l'algorithme (exécuté en hors ligne) est d'environ 23970 ms (environ 24 s) avec un ordinateur équipé d'un processeur Intel Core i5-63000 (CPU 2,40 GHz 2,50 GHz – 8Go RAM). Le temps est donné pour l'algorithme programmé sous Eclipse en langage Java. Dans cet exemple, l'algorithme programmé sous Matlab ne peut nous fournir une modélisation pour le filtre de commande. L'étape 3 de la méthodologie est alors appliquée et correspond à l'exploitation du filtre et de la stratégie de détection dont les résultats sont présentés dans le Chapitre 7 – 3.

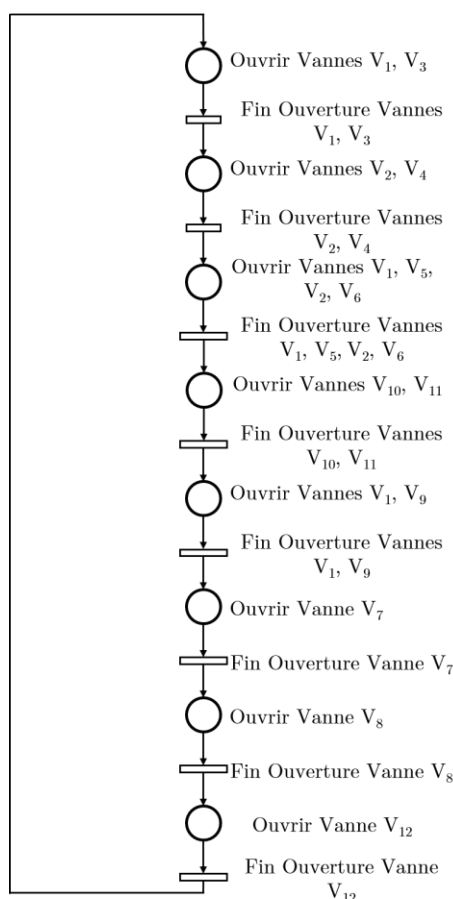


Figure 75 : Illustration de la loi de commande du système à 6 cuves

B. Système à 1 chariot – 1 aiguillage

Le système, illustré en Figure 76, est composé d'un chariot qui circule sur un réseau reliant les points A, B et C avec un aiguillage au point E. Ce dernier peut être contrôlé afin d'orienter le wagon vers le point B (*Vers le haut*) ou C (*Vers le Bas*) et est caractérisé par 3 états (haut, bas et intermédiaire). Le chariot peut être contrôlé afin de le faire avancer, reculer ou freiner. Quatre capteurs de position permettent de connaître la localisation du wagon sur le réseau.

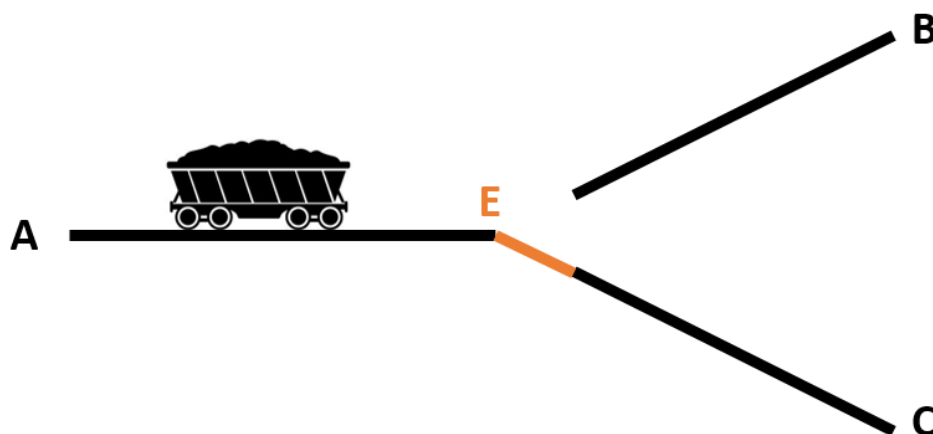


Figure 76 : Illustration du système avec 1 chariot et 1 aiguillage

Les spécifications de la loi de commande imposent de démarrer un cycle de production avec le wagon en A et l'aiguillage en position basse (orienté vers le point C). Ce dernier passe par la suite en position haute (orienté vers le point B) et le chariot avance jusqu'au point B. Après un temps d'arrêt, il revient ensuite au point A en reculant. L'aiguillage est alors mis en position basse (orienté vers le point C). Le chariot avance jusqu'au point C où il marque un temps d'arrêt avant de retourner au point A. Un nouveau cycle de production peut alors démarrer.

Considérons que l'analyse de risques a permis d'identifier une seule zone critique pour la partie opérative qui correspond au déraillement du wagon et qui peut être atteint lorsque le chariot recule alors qu'il est au point A, avance alors qu'il est localisé au point B ou au point C, ou encore, lorsque l'aiguillage est en position intermédiaire. Afin de caractériser ces événements redoutés dans notre modélisation, nous identifions les paramètres nécessaires et suffisants pour décrire notre système sous forme de :

- Vecteur des états $s \in S$ qui représente le comportement normal du système puisque les défaillances ne sont pas modélisées. Il se compose des valeurs des capteurs de position du chariot et de l'aiguillage ainsi que des valeurs de commandes de ces deux derniers éléments telles que dans l'équation (viii) :

$$\begin{aligned} s &= [\text{Valeurs Capteurs} ; \text{Valeurs Actionneurs}] \\ &= (Pos_{\text{Chariot}}, Pos_{\text{Aiguillage}}, Avancer, Reculer, Haut, Bas) \end{aligned} \quad (\text{viii})$$

$$\begin{aligned} \text{Avec : } Pos_{\text{Chariot}} &\in \{0..4\}, Pos_{\text{Aiguillage}} \in \{0..2\}, Avancer \in \{0,1\}, Reculer \in \{0,1\}, \\ Haut &\in \{0,1\}, Bas \in \{0,1\}. \end{aligned}$$

Dans l'équation (viii), Pos_{Chariot} indique le capteur de position actif (0 : chariot en A, 1 : chariot en B, 2 : chariot en C et 3 : chariot en E), $Pos_{\text{Aiguillage}}$ fournit la position de l'aiguillage (0 : en haut, 1 : intermédiaire et 2 : en bas). Pour ce système, l'ensemble des états possibles S est composé d'au plus 192 états et est noté S_{192}

- Vecteur actions $a \in A$ qui représente les actions pouvant être exécutées sur le système telles que $a = a_1, a_2, a_3, a_4$ avec $a_i \in \{Avancer, Reculer, Haut, Bas\}$ $i \in \{1..4\}$. L'arrêt du wagon est réalisé par l'activation simultanée des ordres a_1 (Avancer) et a_2 (Reculer). Cet ensemble A est composé de 16 actions et est nommé par la suite A_{16} .
- Ensemble des états atteignables $S_{\text{Reach}} \subset S$ qui contient 113 états dont 18 états optimaux (obtenus à l'aide du modèle de commande présenté en Figure 77), 1 état interdit et 94 états dangereux $S_{\text{Dan}} = S_{\text{Reach}}(S_{\text{Opt}} \cap S_{\text{Pro}})$.
- Ensemble des états inatteignables $S_{\text{Unreach}} = S \setminus S_{\text{Reach}}$ qui contient donc 79 états.

Un modèle de la partie commande est réalisé sur la base des spécifications de la loi de commande et est représenté en Figure 77. De la même manière, le modèle de la partie opérative est construit en équation (ix) à l'aide de notre automate M :

$$M = \{S_{192}, A_{16}, \delta, s_0, S_{\text{final}}\} \quad (\text{ix})$$

L'ensemble S_{192} représente l'ensemble des états atteignables pour le système, A_{16} est l'ensemble des ordres qui peuvent être appliqués sur le système (toutes les combinaisons possibles), δ est la fonction de transition qui permet de faire évoluer le système et dont les effets

sont synthétisés sur le Tableau 18, s_0 est l'état initial du système défini par $s_0 = [0\ 2\ 0\ 0\ 0\ 0]$, S_{final} représente l'ensemble des états finaux de notre automate et donc nos conditions d'arrêts (S_{pro} , s_0 et S_{prev}). Le Tableau 17 regroupe les conditions sur le couple État/Action qui conduisent à l'état interdit de débordement.

Tableau 17 : Récapitulatif des conditions menant à l'état interdit de débordement

État interdit S_{Pro}	Ordre $a \in A_{16}$	État $s \in S_{192}$ du système avant l'action $a \in A_{16}$
Déraillement du chariot	Reculer ($a = [0\ 1\ \times\ \times]$)	Chariot au point A : $s = [0\ \times\ \times\ \times\ \times\ \times]$
	Avancer ($a = [1\ 0\ \times\ \times]$)	Chariot en B ou C : $s = [1\ \times\ \times\ \times\ \times\ \times]$ ou $s = [2\ \times\ \times\ \times\ \times\ \times]$
	Avancer ou Reculer ($a = [1\ 0\ \times\ \times]$) ou ($a = [0\ 1\ \times\ \times]$)	Aiguillage en position intermédiaire : $s = [\times\ 1\ \times\ \times\ \times\ \times]$

Tableau 18 : Récapitulatif des effets des actions sur le système 1 chariot – 1 aiguillage

Ordre $a \in A_{16}$	État $s \in S_{192}$ du système avant l'action $a \in A_{16}$
Avancer ($a_1 = [1\ 0\ \times\ \times]$)	$s = [Pos_{Chariot}\ \times\ \times\ \times\ \times\ \times]$ $s' = \delta\ s, a_1$ $[3\ 1\ \times\ \times\ \times]$ si $Pos_{Chariot} = 0$ avec $s' = \begin{cases} [1\ \times\ 1\ \times\ \times\ \times] & \text{si } Pos_{Chariot} = 3 \text{ et } Pos_{aiguillage} = 0 \\ [2\ \times\ 1\ \times\ \times\ \times] & \text{si } Pos_{Chariot} = 3 \text{ et } Pos_{aiguillage} = 2 \\ \emptyset & \text{si } Pos_{Chariot} = \{1,2\} \text{ ou } Pos_{aiguillage} = 1 \end{cases}$
Reculer ($a_2 = [0\ 1\ \times\ \times]$)	$s = [Pos_{Chariot}\ \times\ \times\ \times\ \times\ \times]$ $s' = \delta\ s, a_2$ $[0\ \times\ 1\ \times\ \times]$ si $Pos_{Chariot} = 3$ avec $s' = \begin{cases} [3\ \times\ \times\ 1\ \times\ \times] & \text{si } Pos_{Chariot} = 1 \text{ et } Pos_{aiguillage} = 0 \\ [3\ \times\ \times\ 1\ \times\ \times] & \text{si } Pos_{Chariot} = 2 \text{ et } Pos_{aiguillage} = 2 \\ \emptyset & \text{si } Pos_{Chariot} = 0 \text{ ou } Pos_{aiguillage} = 1 \end{cases}$
Stopper ($a_3 = [1\ 1\ \times\ \times]$)	$s = [Pos_{Chariot}\ \times\ \times\ \times\ \times\ \times]$ $s' = \delta\ s, a_3$ avec $s' = [Pos_{Chariot}\ \times\ 1\ 1\ \times\ \times]$
Aiguillage en haut ($a_4 = [\times\ \times\ 1\ \times]$)	$s = [\times\ Pos_{aiguillage}\ \times\ \times\ \times\ \times]$ $s' = \delta\ s, a_4$ avec $s' = \begin{cases} [\times\ 1\ \times\ \times\ 1\ \times] & \text{si } Pos_{aiguillage} = 2 \\ [\times\ 0\ \times\ \times\ 1\ \times] & \text{si } Pos_{aiguillage} = 1 \\ [\times\ 0\ \times\ \times\ 1\ \times] & \text{si } Pos_{aiguillage} = 0 \end{cases}$

Aiguillage en bas $(a_5 = [\times \times \times 1])$	$s = [\times Pos_{aiguillage} \times \times \times \times]$ $s' = \delta s, a_5$ avec $s' = \begin{cases} [\times 2 \times \times \times 1] & si\ Pos_{aiguillage} = 2 \\ [\times 2 \times \times \times 1] & si\ Pos_{aiguillage} = 1 \\ [\times 1 \times \times \times 1] & si\ Pos_{aiguillage} = 0 \end{cases}$
Aucun ordre $(a_{16} = [0\ 0\ 0\ 0\ 0])$	$s = [\times \times \times \times \times \times]$ $s' = \delta s, a_{16} = [\times \times 0\ 0\ 0\ 0]$

Nous remarquerons que ces effets, dans leur ensemble, sont cumulables les uns avec les autres.

Pour conclure l'étape 2, l'algorithme de génération du modèle du filtre de commande est exécuté sur la base des modèles construits, ce qui conduit à l'obtention de 113 états atteignables et 466 contextes menant à des états interdits. Le temps de calcul pris par l'algorithme (exécuté en hors ligne) est d'environ 130 ms avec un ordinateur équipé d'un processeur Intel Core i5-63000 (CPU 2,40 GHz 2,50 GHz – 8Go RAM). Le temps est donné pour l'algorithme programmé sous Eclipse en langage Java. Dans cet exemple, l'algorithme programmé sous Matlab ne peut nous fournir un modèle complet du filtre de commande. L'étape 3 de la méthodologie est alors appliquée et correspond à l'exploitation du filtre et de la stratégie de détection dont les résultats sont présentés dans le Chapitre 7 – 3.

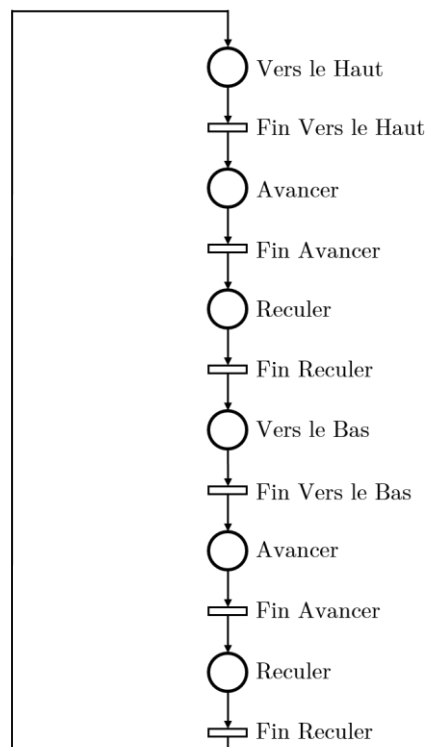


Figure 77 : Illustration de la loi de commande du système à 1 chariot – 1 aiguillage

3. Annexe 3 : Complément pour la construction du modèle du filtre de commande

A. Fonctions nécessaires à la modélisation

Le dossier permettant d'obtenir le modèle du filtre de commande, en Figure 78, se compose d'un script Matlab (*Automate_process_3cuves.m*) et de plusieurs fonctions Matlab :

- *all_comb* : cette fonction calcule toutes les combinaisons possibles d'une matrice $m \times n$ où m est un vecteur indiquant le point de départ, celui d'arrivée et le pas, et n le nombre de composantes à prendre en compte. Le résultat est donné sous forme d'une matrice dont chaque ligne indique une combinaison.
- *commande_cuves* : cette fonction représente le modèle de commande du système (Ladder ou Grafcet). Elle prend en entrée l'état du système et calcule l'ordre envoyé à la partie opérative.
- *delta* : cette fonction représente le modèle de la partie opérative du système. Cette fonction détermine l'état s_{i+1} atteint par le système en fonction de l'action a_i et de l'état s_i .
- *dmin_PS* : cette fonction calcule le plus court chemin pour arriver à un état interdit. La fonction renvoie également les séquences d'états et d'ordres menant à l'état interdit à partir de l'état de départ.
- *create_path* : cette fonction concatène les séquences d'états et d'ordres menant à des états interdits.
- *Param_System.mat* : structure créée par le script *Automate_process_3cuves.m* et qui contient les paramètres du système (Ensemble des états possibles, Ensemble des ordres possibles, Index qui regroupe au sein d'une matrice l'évolution optimale du système).
- *State_Opt_set.mat* : structure contenant les états et ordres optimaux.
- *State_previous.mat* : structure contenant les informations sur les états du système tel que :
 - $d_{\min}$ qui répertorie les distances aux états critiques pour chaque état atteignable du système.
 - *State_previous* qui liste les états atteignables.
 - *Order_Path_crit* et *State_Path_crit* qui listent les séquences pour atteindre un état interdit à partir de l'état initial.
- *State_Pro_set.mat* : structure contenant les contextes menant à des états interdits.

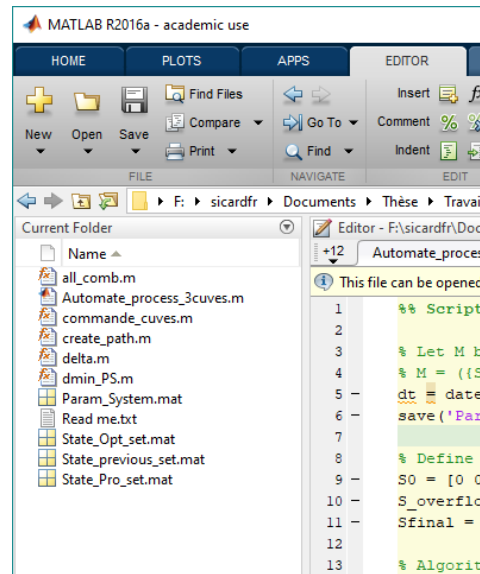


Figure 78 : Espace de travail pour générer le modèle du filtre de commande sous Matlab

B. Exemple de modèle de partie opérative

```
function [State_next] = delta(State, Action)

% Action is a vector 1*3 All or Nothing so that we have 8 combinations.

% On décrit les effets des actions sur le système
if ((Action(1,1)==1 && Action(1,2)==0) || (Action(1,1)==0 && Action(1,2)==1)) && Action(1,3)==0)
    % [1 0 0] and [0 1 0]
    % Débordement
    if(State(1,1)==3)
        State_next = [-1 -1 -1 -1]; % S_overflow
    % Augmentation du niveau de la cuve
    else
        State_next(1,1) = State(1,1)+1;
    end
elseif (Action(1,1)==1 && Action(1,2)==1 && Action(1,3)==0)
    % [1 1 0]
    % Débordement
    if(State(1,1)==3 || State(1,1)==2)
        State_next = [-1 -1 -1 -1]; % S_overflow
    % Augmentation du niveau de la cuve
    else
        State_next(1,1) = 3;
    end
elseif (Action(1,3)==1)
    % [0 0 1], [0 1 1], [1 0 1], [1 1 1]
    % Initialisation du système
    State_next(1,1)=0;
else
    % [0 0 0]
    State_next = State;
end

if(isequal(State_next, -1*ones(1,4))==0)
    if(Action(1,1)==1)
        State_next(1,2)=1;
    elseif(Action(1,1)==0)
        State_next(1,2)=0;
    end
    if(Action(1,2)==1)
        State_next(1,3)=1;
    elseif(Action(1,2)==0)
        State_next(1,3)=0;
    end
    if(Action(1,3)==1)
        State_next(1,4)=1;
    elseif(Action(1,3)==0)
        State_next(1,4)=0;
    end
else
    State_next = State_next;
end
```

Figure 79 : Capture d'écran d'un exemple de modèle de partie opérative

C. Calcul des états optimaux

```
% Calcul des Etats Optimaux
Control_Step = 0;
State = S0;

[Ordre, Control_Step] = commande_cuves(Control_Step,S0); % Calcul commande
Etat_suivant = delta(State,Ordre); % Calcul Etat_suivant
State_Otp = State;
Ordre_Otp = Ordre;
save('State_Otp_set', 'State_Otp', 'Ordre_Otp')
State = Etat_suivant;

while(isequal(State,S0)==0)
    [Ordre, Control_Step] = commande_cuves(Control_Step,State);
    Etat_suivant = delta(State,Ordre);
    load('State_Otp_set')
    State_Otp = [State_Otp; State];
    Ordre_Otp = [Ordre_Otp; Ordre];
    save('State_Otp_set', 'State_Otp', 'Ordre_Otp')
    State = Etat_suivant;
end
```

Figure 80 : Capture d'écran du code permettant d'obtenir les états optimaux du système

D. Fonction *dmin_PS* : Algorithme de calcul de la distance aux états interdits les plus proches

```
function [d_min, State_Path_crit, Order_Path_crit] = dmin_PS (State, Actions)
% dmin_PS calcule le plus court chemin pour arriver à un état
% interdit. La fonction renvoie également le chemin menant (état) et les
% ordres menant à l'EI à partir de l'état de départ.
%
% Input :
% State : Vecteur d'état actuel du système,
% Actions : Vecteur d'action possible,
%
% Output :
% d_min : distance minimale d'un état interdit
% State_Path : succession d'états menant vers un ou des état(s) interdit(s)
% Order_Path : succession d'ordres menant vers un ou des état(s) interdit(s)

S0 = State;
l_State = size(State,2);
l_Action = size(Actions,2);
Inter = [];
Etat_suivant_inter = [];
Order_suivant_inter = [];
Etat_next = [];
Order_next = [];
iteration = 0;
PS_detected = [0];
State_Path = [];
Order_Path = [];
Path = [State_Path Order_Path];

while(max(PS_detected)==0)
    for i=1:l_State(1)
        for j=1:l_Action(1)
            Order = Actions(j,:);
            Etat_suivant = delta(State(i,:), Order);

            Inter = [Inter; iteration State(i,:) Order Etat_suivant];
            Etat_suivant_inter = [Etat_suivant_inter;Etat_suivant];
            Order_suivant_inter = [Order_suivant_inter;Order];

            if(isequal(Etat_suivant, (-i*ones(1,4)))==1)
                PS_detected = [PS_detected 1];
            else
                PS_detected = [PS_detected 0];
            end

            end
            Etat_next = [Etat_next ; Etat_suivant_inter];
            Order_next = [Order_next ; Order_suivant_inter];

            Etat_suivant_inter = [];
            Order_suivant_inter = [];

        end
        [State_Path, Order_Path] = create_path(Path, State, Order_next, Etat_next);
        Path = [State_Path Order_Path];
        State = Etat_next;
        Etat_suivant_inter = [];
        Etat_next = [];
        Etat_suivant = [];
        Order_next = [];
        iteration = iteration+1;
    end

% Computation of distance from critical state
% For each state of State_previous_set, we compute distance from the
% nearest prohibited state

dmin = [];
load('State_previous_set.mat')
Order_Path_crit = [];
State_Path_crit = [];

for i=1:size(State_previous, 1)
    State_Path = [];
    Order_Path = [];
    load('State_previous_set')
    [d_min, State_Path, Order_Path] = dmin_PS(State_previous(i,:), Orders_Set);
    dmin = [dmin; d_min];

    State_Path = [i*ones(size(State_Path,1),1) State_Path];
    Order_Path = [i*ones(size(Order_Path,1),1) Order_Path];

    if(isempty(State_Path_crit)==1)
        State_Path_crit = [State_Path_crit State_Path];
        Order_Path_crit = [Order_Path_crit Order_Path];
    else
        taille_chemin_state = [];
        for i=1:size(State_Path_crit,1)
            taille_chemin_state = [taille_chemin_state size(State_Path_crit(i,:),2)];
        end
        taille_chemin_state = max(taille_chemin_state);

        taille_chemin_ordre = [];
        for i=1:size(Order_Path_crit,1)
            taille_chemin_ordre = [taille_chemin_ordre size(Order_Path_crit(i,:),2)];
        end
        taille_chemin_ordre = max(taille_chemin_ordre);

        if(size(State_Path,2)==taille_chemin_state)
            State_Path_crit = [State_Path_crit;State_Path];
            Order_Path_crit = [Order_Path_crit;Order_Path];
        elseif(size(State_Path,2)>taille_chemin_state)
            % Mettre des NaN sur State_Path
            for i=size(State_Path,2)+1:taille_chemin_state
                State_Path = [State_Path NaN(size(State_Path,1),1)];
            end
            for i=size(Order_Path,2)+1:taille_chemin_ordre
                Order_Path = [Order_Path NaN(size(Order_Path,1),1)];
            end
            State_Path_crit = [State_Path_crit;State_Path];
            Order_Path_crit = [Order_Path_crit;Order_Path];
        elseif(size(State_Path,2)>taille_chemin_state)
            % Reprendre chaque State_Path_crit(i,:) et compléter
            % éventuellement avec des NaN
            for i=size(State_Path_crit,2)+1:1:size(State_Path,2)
                State_Path_crit = [State_Path_crit NaN(size(State_Path_crit,1),1)];
            end
            for i=size(Order_Path_crit,2)+1:1:size(Order_Path,2)
                Order_Path_crit = [Order_Path_crit NaN(size(Order_Path_crit,1),1)];
            end
            State_Path_crit = [State_Path_crit;State_Path];
            Order_Path_crit = [Order_Path_crit;Order_Path];
        end
    end
    save('State_previous_set', 'State_previous', 'dmin', 'State_Path_crit', 'Order_Path_crit');
end
```

Figure 81 : Capture d'écran de la fonction permettant le calcul de la plus courte distance aux états interdits

E. Code pour l'exploration des états atteignables sous Java



```

1 import java.io.FileOutputStream;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         long time = System.currentTimeMillis();
7         int nombreActionneurs = 5;
8         Action[] actions = new Action[nombreActionneurs];
9         for (int i = 0; i < nombreActionneurs; i++) {
10             actions[i] = new Action(0, 1, 1);
11         }
12         Ordres ListesOrdres = new Ordres(actions);
13         Ordre[] ordres = ListesOrdres.getOrdres();
14
15         int nombreCapteurs = 2;
16         int[] valeursInitiales = new int[nombreActionneurs + nombreCapteurs];
17         for (int i = 0; i < valeursInitiales.length; i++) {
18             valeursInitiales[i] = 0;
19         }
20         TableauMeeting tab = new TableauMeeting();
21         Etat etatIni = new Etat(valeursInitiales);
22         ArrayList<CombinaisonInterdite> etatsInterdits = new ArrayList<CombinaisonInterdite>();
23         ArrayList<Etat> liste = new ArrayList<Etat>();
24         liste.add(etatIni);
25         tab.vu(etatIni);
26         int etatsrencontres = 0;
27         while (!liste.isEmpty()) {
28             Etat e = liste.remove(0);
29             etatsrencontres++;
30             for (Ordre ordre : ordres) {
31                 Etat etatTampon = new Etat(e.Transition(ordre));
32                 if (etatTampon.getCapteur(0) == -1) {
33                     etatsInterdits.add(new CombinaisonInterdite(e, ordre));
34                 } else if (!tab.dejavu(etatTampon)) {
35                     tab.vu(etatTampon);
36                     liste.add(etatTampon);
37                 }
38             }
39             if (etatsrencontres % 1000 == 0) {
40                 System.out.println(etatsrencontres);
41                 System.out.println(liste.size());
42             }
43         }
44         System.out.println("Temps d'execution : " + (System.currentTimeMillis() - time));
45         System.out.println("Travail Terminé");
46         //System.out.println("Nombre de Combinaison interdite" + etatsInterdits.size());
47
48         try {
49             PrintStream ExcelEtatsAtteignables = new PrintStream(new FileOutputStream("ExcelEtatsAtteignables.csv"));
50             ExcelEtatsAtteignables.println("Numero d'Etat; " + "hauteurc1; " + "hauteurc2; " + "actionneur1; " + "actionneur2; " + "actionneur3; " + "actionneur4; " + "actionneur5;");
51             for (int c1 = 0; c1 < 3; c1++) {
52                 for (int c2 = 0; c2 < 4; c2++) {
53                     for (int v1 = 0; v1 < 2; v1++) {
54                         for (int v2 = 0; v2 < 2; v2++) {
55                             for (int v3 = 0; v3 < 2; v3++) {
56                                 for (int v4 = 0; v4 < 2; v4++) {
57                                     for (int v5 = 0; v5 < 2; v5++) {
58                                         if (tab.dejavu(c1, c2, v1, v2, v3, v4, v5)) {
59                                             ExcelEtatsAtteignables.println(";" + c1 + ";" + c2 + ";" + v1 + ";" + v2 + ";" + v3 + ";" + v4 + ";" + v5);
60                                         }
61                                     }
62                                 }
63                             }
64                         }
65                     }
66                 }
67             }
68             ExcelEtatsAtteignables.flush();
69             ExcelEtatsAtteignables.close();
70             ExcelEtatsAtteignables = null;
71         } catch (Exception e) {
72             System.out.println(e.toString());
73         }
74
75         try {
76             PrintStream ExcelEtatsInterdits = new PrintStream(new FileOutputStream("ExcelEtatsInterdits.csv"));
77             ExcelEtatsInterdits.println("Titre");
78             for (int i = 0; i < etatsInterdits.size(); i++) {
79                 ExcelEtatsInterdits.println(i + 1 + ";");
80                 ExcelEtatsInterdits.println(etatsInterdits.get(i).getEtat().ecrire() + etatsInterdits.get(i).getOrdre().ecrire());
81             }
82             ExcelEtatsInterdits.flush();
83             ExcelEtatsInterdits.close();
84             ExcelEtatsInterdits = null;
85         } catch (Exception e) {
86             System.out.println(e.toString());
87         }
88         System.out.println("Temps total écoulé : " + (System.currentTimeMillis() - time));
89     }
90 }

```

Figure 82 : Capture d'écran du code Java adapté de Matlab permettant l'exploration des états du système

4. Références des Annexes

- Chaillet, A., M. Combacau, and M. Courvoisier. 1993. "Specification of FMS Real-Time Control Based on Petri Nets with Objects and Process Failure Monitoring." In *International Conference on Industrial Electronics, Control, and Instrumentation, 1993. Proceedings of the IECON '93*, 144–49 vol.1. <https://doi.org/10.1109/IECON.1993.339091>.
- Combacau, Michel, Philippe Esteban, and Alexandre Nketsa. 2005. "Commande à Réseaux de Petri - Modélisation." *Techniques de l'Ingénieur* 7572 (October): 1–18.
- Ladet, Pierre, and Isabel Demongodin. 2009. "Outils de Modélisation Des Automatismes Séquentiels - Réseau de Petri." *Techniques de l'Ingénieur* 7252 (October): 1–26.
- Murata, T. 1989. "Petri Nets: Properties, Analysis and Applications." *Proceedings of the IEEE* 77 (4): 541–80. <https://doi.org/10.1109/5.24143>.
- Peterson, James L. 1977. "Petri Nets." *ACM Computing Surveys (CSUR)* 9 (3): 223–252. <http://dl.acm.org/citation.cfm?id=356702>.
- Petri, Carl Adam. 1966. "Communication with Automata," 97.
- Zhang, X., Y. Du, L. Qi, and H. Sun. 2018. "An Approach for Repairing Process Models Based on Logic Petri Nets." *IEEE Access*, 1–1. <https://doi.org/10.1109/ACCESS.2018.2843137>.
- Zurawski, R., et MengChu Zhou. 1994. « Petri nets and industrial applications: A tutorial ». *IEEE Transactions on Industrial Electronics* 41 (6): 567-83. <https://doi.org/10.1109/41.334574>.
- Zurawski, R., and MengChu Zhou. 1994. "Petri Nets and Industrial Applications: A Tutorial." *IEEE Transactions on Industrial Electronics* 41 (6): 567–83. <https://doi.org/10.1109/41.334574>.