Approche filtre basée sur la notion de distance pour la détection des cyberattaques

Franck Sicard, Jean-Marie Flaus, Eric Zamaï

Approche filtre basée sur la notion de distance pour la détection des cyberattaques

Franck SICARD
Univ.Grenoble Alpes, CNRS, G-SCOP
38000 Grenoble, France
franck.sicard@grenoble-inp.fr

Jean-Marie Flaus
Univ.Grenoble Alpes, CNRS, G-SCOP
38000 Grenoble, France
jean-marie.flaus@grenoble-inp.fr

Eric Zamaï
Univ.Grenoble Alpes, CNRS, G-SCOP
38000 Grenoble, France
eric.zamai@grenoble-inp.fr

Résumé*— Les systèmes de contrôle-commande sont devenus la nouvelle cible des hackers depuis le début du siècle. Le ver informatique Stuxnet a démontré la vulnérabilité de ces systèmes face aux cyberattaques. Les systèmes de contrôle commande ont été construits pour assurer la productivité et la fiabilité, face à des aléas de fonctionnement, du système et de son environnement. Toutefois, différents travaux et attaques, ont souligné le manque de protection de ces composants. De plus, les solutions proposées dans le domaine de l'informatique n'apportent pas de solutions complètes. Ce papier présente une approche innovante pour détecter une attaque dans des systèmes de contrôle commande basée sur la notion d'états et de distance entre ces derniers. L'évaluation au cours du temps d'une distance entre l'état courant du système et des états interdits permet d'anticiper et de discriminer des dérives. Cet algorithme analyse en temps réel les ordres envoyés aux actionneurs et stoppe ceux présentant un danger pour l'installation. Cette étude théorique est appliquée à des exemples simulés issus de systèmes classiques.

***Mots-clés*— Cybersécurité, Détection, Diagnostic, Filtre, Distance, Système de contrôle-commande industriel**

I. INTRODUCTION

Les systèmes de contrôle-commande industriels ou Industrial Control System (ICS) contrôlent et surveillent des procédés physiques. Ces systèmes sont présents dans plusieurs domaines et de plus en plus intégrés aux infrastructures critiques comme la production et distribution d'énergie (électricité, eau, gaz, …), les systèmes de production manufacturier, le transport, les services de santé ou encore la défense [1].

A. Qu'est ce qu'un système de contrôle commande industriel (ICS)?

Ces systèmes sont pensés pour répondre à des problématiques de production afin d'assurer la productivité et la sûreté de fonctionnement en présence d'aléas de fonctionnement « naturels ». Les ICS, dont un exemple sous forme d'architecture matérielle est donné en Figure 1, sont composés de plusieurs couches hiérarchiques. Une couche supervision (niveau 2), aussi appelée *SCADA* (Supervisory Control and Data Acquisition) donne aux opérateurs une image du procédé à un moment donné. Pour cela, elle récupère les données des couches terrains, les affiche et transmet les consignes à l'ensemble de l'architecture. Une couche commande (niveau 1) contrôle les équipements de terrain agissant physiquement avec le procédé, acquiert les données acquises sur ce dernier et communique avec la couche supervision. L'élément principal est l'Automate Programmable Industriel (API). Sur la base des données acquises par les capteurs et de la loi de commande programmée, l'API décide des actions à mener sur le système. Une couche basse (niveau 0), composée d'actionneurs et de capteurs, permet de faire le lien entre la couche cyber et physique. Cette couche fait évoluer le procédé d'un état initial à un état final. Des réseaux de communication permettent de faire le lien avec les couches précédentes. Les architectures récentes sont organisées autour de modules communiquant massivement, notamment avec le protocole TCP/IP. En conséquence, les ICS ont été fragilisés car la notion de sécurité n'a pas été prise en compte.

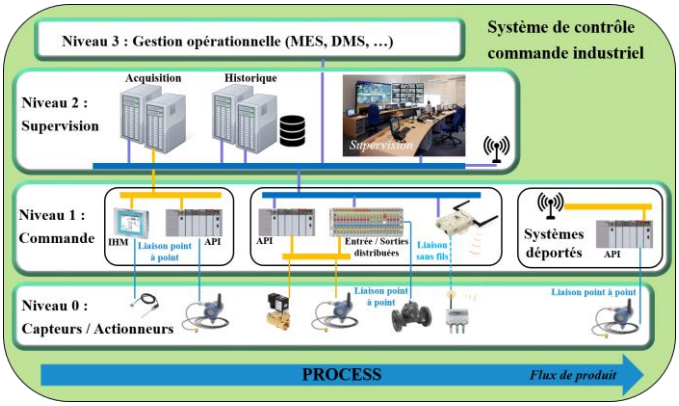

Figure 1. EXEMPLE D'UNE ARCHITECTURE MATERIELLE D'UN SYSTEME DE CONTROLE-COMMANDE INDUSTRIEL

B. Problématique face aux cyberattaques

Depuis le début du siècle, les systèmes de contrôle-commande industriels sont la cible de hackers qui exploitent leurs vulnérabilités au moyen d'attaques de type « cyber ». Un historique complet et détaillé des attaques contre des ICS se trouve dans [1], [2]. Malgré l'augmentation du nombre et de la complexité de ce type d'attaque, la référence reste le ver informatique Stuxnet qui a infecté les centrales iraniennes en 2010 [3]. Après l'infection d'un API via le réseau bureautique et d'une phase d'apprentissage, le programme automate était remplacé par celui du ver afin de dégrader le procédé jusqu'à la destruction d'actionneurs. L'intérêt des hackers pour les ICS provient de l'impact physique qu'aurait une cyberattaque si elle réussissait. D'importants dégâts peuvent ainsi être infligés au procédé (arrêt(s) de production, temps de réparation) et à son environnement (impacts en termes humains, sociaux, environnementaux, pertes financières, perte d'image ou encore sécurités).

Une liste exhaustive des vulnérabilités recensées sur les ICS se trouve sur le site de l'ICS-CERT [4]. Fovino et al [5] recense des attaques pouvant être exécutées contre des ICS. Nous retrouvons ainsi des attaques par exécution de commandes illicites, Scada-Dos, Man in the middle ou encore replay attack. Toutes ces attaques présentent la particularité d'être héritées du monde de l'informatique de gestion (Information Technology, IT) et infectent donc le niveau 2 de l'architecture ICS. Au contraire, d'autres attaques sont inhérentes aux niveaux 1 et 0 comme les *attaques aléatoires* qui envoient des informations sans tenir compte du procédé, les *attaques séquentielles* [6], [7] où le séquencement de la loi de commande est violé pour impacter le flux de production et les *attaques par injection de fausses données* [8] qui interceptent et modifient les données échangées entre les capteurs et l'API (mesures) ou l'API et les actionneurs (commandes).

Dans cette étude, nous nous intéresserons uniquement aux attaques sur la commande entre l'API et les actionneurs. Ce positionnement vient du fait que l'ensemble des composants de l'architecture ICS sont faillibles. De plus, aucune solution ne propose d'intervenir entre le niveau 1 et 0 alors qu'il s'agit de la dernière occasion pour stopper un ordre illicite avant qu'il soit appliqué sur le système. Des approches dans le domaine de la Surveillance en particulier dans celui de la Détection peuvent fournir des solutions pour sécuriser les ICS de cyberattaques. En effet, le but de ces dernières est d'entrainer une perte de service. La sécurité des ICS rejoint la problématique connue et classique de la Détection de perte de service. Ce dernier est un domaine de recherche bien connu où plusieurs approches efficaces ont été développées mais comment pouvons-nous distinguer une attaque (intentionnel) d'une défaillance (non-intentionnel) ? Comment améliorer les approches existantes pour déterminer l'origine des attaques ? Notre approche se base sur le concept de trajectoire, avec notamment la construction de formes caractéristiques d'attaques et la notion de distance par rapport à un état dangereux qui instant par instant donne des informations sur une éventuelle attaque. Cet article se propose de contribuer pour le moment à la définition de la distance dans le but de détecter une attaque.

II. VERS LE CONCEPT DE DISTANCE POUR DETECTER DES ATTAQUES CONTRE DES ICS

Les Intrusion Detection System (IDS) sont des mesures de protections largement déployés en IT pour détecter des attaques contre des réseaux informatiques. Dans ces travaux, Denning [9] a établi un framework qui est encore utilisé et efficace de nos jours. Il se base sur l'idée que peu importe les moyens utilisés pour assurer la sécurité d'un système, il existe toujours un risque résiduel d'intrusion. L'IDS est une mesure de sécurité à posteriori dont l'objectif est la détection d'une intrusion et le signalement à un opérateur après que l'intrusion ait eu lieu. Ainsi, un IDS va identifier comme une intrusion toutes activités en violation avec la politique de sécurité et mettant en péril la confidentialité, la disponibilité ou l'intégrité des données. Pour cela, les IDS se basent sur l'utilisation de sondes de détection où une réflexion est menée sur l'implantation dans l'architecture et le type de détection. Les travaux de Mitchell et Chen [10] distinguent la méthode de détection et la source des données. L'IDS peut détecter une déviation par rapport au modèle de fonctionnement, il s'agit d'une approche comportementale. Au contraire, l'IDS peut s'appuyer sur une approche par signature où il va chercher des ressemblances avec des comportements définis comme anormaux. De la même manière, l'IDS peut se baser sur l'analyse du réseau, il s'agit d'une sonde *NIDS* (Network Intrusion Detection System), ou sur l'analyse d'un nœud, il s'agit d'une sonde *HIDS* (Host Intrusion Detecting System).

Sur les différentes techniques envisagées pour sécuriser les ICS, les IDS sont un domaine de recherche bien investigué, [1], [2] en fourniront une liste assez exhaustive. Ainsi, les approches par signatures, comme la détection de comportements spécifiques comme énoncée dans [11], ne fournissent pas une solution satisfaisante. En effet, elle ne prend pas en compte les attaques *zero-day* qui, par définition, exploitent des vulnérabilités qui ne sont pas connues avant qu'elles ne se produisent. Cependant, les systèmes de contrôle-commande présentent la spécificité de contrôler des procédés physiques, ainsi les approches comportementales, ou à base de modèles, semblent être plus adaptées. Dans ces approches, les spécificités des systèmes sont prises en compte pour établir les règles garantissant la sécurité, la fiabilité et la performance du procédé. Dans [12], les auteurs extraient, des fichiers de configuration, les informations nécessaires pour établir les règles de bon fonctionnement. Cependant, si le développeur commet une erreur de programmation cela causera une erreur dans la configuration de l'IDS. Même si une majorité des publications se concentre sur l'analyse du réseau industriel, certaines se concentrent sur le fait que les ICS contrôlent des systèmes avec une vraie signification physique. Dans [13], les auteurs utilisent la connaissance du système pour écrire les fonctions de transfert du système et les transformées de Fourier et détecter des intrusions. Le problème majeur de cette approche est qu'une relation mathématique ne peut être trouvée pour tous les systèmes. Les travaux de Carcano et al [14] et Fovino [5] marquent un tournant dans l'utilisation de la connaissance du système. Après avoir modélisé le système en précisant les plages de fonctionnement de chaque variable, les auteurs utilisent le concept d'état et de distance d'un état critique pour détecter des attaques. La distance permet ainsi d'évaluer le rapprochement d'une zone définit comme critique. Les auteurs utilisent 2 types de distance :

- d_1 évaluant la différence entre 2 états composant par composant. Considérons $d_1$comme la distance entre 2 états tels que $d_1 : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^+$ et s et t être 2 vecteurs d'état à n composantes tels que $s \in \mathbb{R}^n$ et $t \in \mathbb{R}^n, d_1(s,t) = \sum_{i=1}^{n} |s_i - t_i|$
- d_v mesurant le nombre de composantes qui varie entre ces deux vecteurs. Considérons $d_v : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^+$ et s et t être 2 vecteurs d'état à n composantes tels que $s \in \mathbb{R}^n$ et $t \in \mathbb{R}^n$, $d_v(s,t) = \#\{i,\ s_i \neq t_i\}$

Dans le reste de notre étude, nous nous focaliserons sur des approches à forte connaissance du procédé en réutilisant cette notion de distance. Enfin, la structure d'une sonde, présenté dans [1], captant une donnée pour l'analyser et la philosophie de cette technique, où il existe toujours un risque résiduel malgré les moyens mis en œuvre, nous permet de faire une analogie avec l'approche filtre.

III. APPROCHE FILTRE

Cette approche a été pour la première fois énoncée par Cruette [15]. A l'origine, elle a été développée pour des applications en sûreté de fonctionnement mais elle présente des caractéristiques intéressantes permettant une adaptation pour la cyber sécurité des ICS. Le principe de base de cette approche est l'évaluation perpétuelle, par des filtres, des données échangées entre la partie commande ou PC (couche commande, niveau 1) et la partie opérative ou PO (couche basse, niveau 0). Ainsi, si les données ne respectent pas les contraintes implantées dans le filtre alors une alerte est envoyée aux opérateurs et l'information peut être bloquée. Un filtre de commande peut ainsi empêcher un ordre incorrect d'être envoyé à des actionneurs tandis qu'un filtre sur les CR permet d'alerter sur les informations de la PO vers la PC.

Dans la continuité de ces travaux, Marange [16] a proposé une adaptation de l'approche filtre avec une formalisation des contraintes basée sur la connaissance du procédé. Dans cette approche, seul le filtre de commande est présent puisque la partie opérative est considérée comme infaillible. Ainsi, les ordres de commande issus de l'API sont filtrés avant qu'ils ne soient envoyés aux actionneurs et analysés sur la base de contraintes interdisant l'accès à des états dangereux. Dans cette approche, les contraintes sont fixées par un expert qui définit des contraintes statiques (physiquement impossible) et dynamiques (interdiction d'un évènement sur la base d'autres évènements). Le fonctionnement de cette approche est illustré sur la Figure 2.

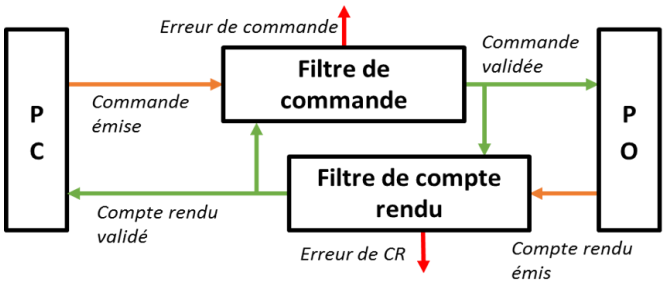

Figure 2. FONCTIONNEMENT DE L'APPROCHE FILTRE DANS UN ICS

L'approche filtre est une solution intéressante pour notre étude possédant de multiples avantages comme celui d'évaluer la véracité des informations avant qu'elles soient envoyées aux organes de commande par un filtre de commande ou à la PC via un filtre sur les comptes rendus (CR). Cette évaluation se fait sur la base de règles qui peuvent être issues de la connaissance du procédé et de la loi de commande. Enfin, cette approche est relativement non intrusive dans les architectures de contrôle-commande. Ce travail améliorera l'établissement des règles notamment au travers la notion d'état afin d'être certain que les experts n'en oublient pas. De plus, la structure du filtre nous permet d'implanter un algorithme permettant de détecter mais également de discriminer l'origine de la défaillance. Pour cela, nous utiliserons le concept de distance testé dans le domaine des IDS dont les sondes présentes une similitude avec l'approche filtre. L'étude s'appuie sur 2 filtres (commande et les CR) qui seront considérés comme invulnérables à une attaque.

IV. METHODE DE CONCEPTION DES FILTRES

La détection d'intrusion avec une approche filtre ne peut être efficace que si les règles à l'intérieur du filtre sont caractéristiques et représentatives du système surveillé. Pour cela, 3 étapes sont nécessaires pour construire les filtres. La première est l'analyse de risques qui permet d'identifier les états interdits du système, c'est-à-dire où le système sera endommagé, des autres états atteignables du système. La deuxième est la phase d'exploration du système, où sur la base de la description des actions, toutes les combinaisons (état et action) possibles sont évalués. Celles menant vers des états non-sûrs sont alors discriminées. Enfin, la phase de synthèse du filtre permet d'implémenter l'algorithme avec des règles de détection et la notion de distance. Ces étapes seront présentées dans les paragraphes suivants.

A. Etape 1 : Analyse de risques

Dans cette étape, nous utilisons le travail préliminaire réalisé en sûreté de fonctionnement où la capacité d'une entité à réaliser une ou plusieurs fonctions sous certaines conditions est vérifiée [1]. Ainsi, des informations sur les états dangereux pour le système et les actions permettant d'y accéder sont extraites. En effet, lorsqu'une attaque est menée contre un ICS, les hackers chercheront toujours à dégrader le système en l'amenant dans un état interdit avec des actions qui sont pas réalisées au bon moment. Ainsi, un système de contrôle-commande ne peut se trouver que dans l'un des ensembles de la Figure 3 détaillé ci-dessous.

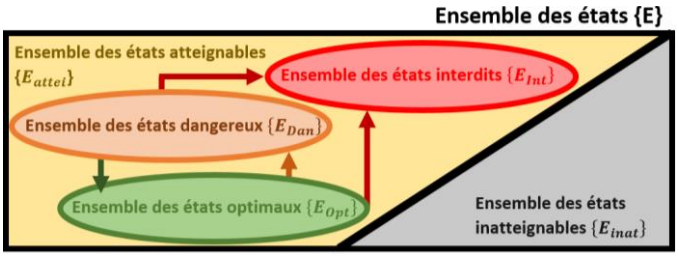

Figure 3. ILLUSTRATION DES EVOLUTIONS D'ETATS POSSIBLES D'UN ICS

Un état donné $E_i \in \{E\}$ d'un système peut-être atteignable ou non. Pour un ICS, le vecteur $E_i \in \{E_{attei}\}$, où E_i est un vecteur de n composantes où n représente le nombre d'entrées du système, est l'image du procédé à un moment donné qui est envoyée à l'API. De la même manière, un ordre O_j appartient à l'ensemble des ordres $\{O\}$ tel que $O_j \in \{O\}$ est une action envoyée à la PO où O_j est un vecteur de m composantes avec m le nombre de sorties du système. La notion d'état dans un procédé a été introduite avec Mitchell and Chen [10]. Enfin, un état atteignable peut appartenir au sous-ensemble des états :

- Optimaux E_{Opt}, $\{E_{Opt}\} \subset \{E_{attei}\}$. Cet ensemble regroupe les états respectant le bon fonctionnement du procédé et les contraintes de la loi de commande. L'API est programmé pour faire en sorte que le système reste dans cet ensemble d'états sûrs,
- Dangereux E_{Dan} , $\{E_{Dan}\} \subset \{E_{attei}\}$. Cet ensemble viole les contraintes de la loi de commande sans causer des dommages importants au système,

- Interdits E_{Int}, $\{E_{Int}\} \subset \{E_{attei}\}$. Ces états ne respectent pas le fonctionnement du procédé et dégradent fortement le système. Ce dernier doit être stoppé avant d'atteindre cet ensemble d'états.

Enfin, O_{Opt} représente les ordres optimaux correspondant à la loi de commande, E_{pre} et O_{pre} décrivent respectivement les états et ordres prédits par les filtres.

B. Etape 2 : Exploration des états du système

L'objectif de cette étape est d'explorer les différentes évolutions possibles du système et de les classer dans les différents sous-ensembles évoqués plus haut. Pour cela, nous nous appuierons sur un modèle de loi de commande et un modèle du procédé. Le premier permet de visualiser l'ensemble des actions à effectuer et l'ordre de réalisation pour atteindre l'état final de manière rapide et sûre. Le deuxième permet de détailler l'ensemble des états atteignables à partir de la description des effets des différentes actions sur le système.

Le modèle de commande est obtenu directement à partir de la loi de commande. La modélisation, utilisée dans l'étape 3 pour obtenir les états optimaux, est réalisée à partir de réseaux de Petri qui mettent en exergue les propriétés de séquencement (ordonnancement des tâches), parallélisme (exécution de tâche en simultané), exclusion mutuelle (l'exécution d'une tâche empêche celle d'une autre) et de synchronisation (attente de la fin d'une tâche). Concernant le modèle du procédé, un ICS comporte un nombre fini d'actions pouvant être exécutées et donc d'états dans lequel le système peut se trouver. Il est possible pour chacune des actions d'un système de décrire son effet sur le procédé. La modélisation par automate fini déterministe est donc particulièrement adaptée pour décrire ce processus. Un automate M, représentatif des capacités du procédé contenues dans la variable δ et obtenues en utilisant les travaux de [17], est défini par le quintuplé suivant :

$$M = \{\{\sum_{i=1}^{n} E_i\}, \{\sum_{j=1}^{m} O_j\}\ , \delta, E_0, E_{final}\}$$

- $\{\sum_{i=1}^{n} E_i\}$ est l'ensemble fini des états atteignables du système où n est le nombre d'états atteignables. E_i est un vecteur représentant l'état du procédé qui peut être dans $\{E_{Opt}\}, \{E_{Dan}\}$ ou $\{E_{Int}\}$,
- $\{\sum_{j=1}^{m} O_j\}$ est l'ensemble fini des actions qui peuvent être exécutées par le système,
- δ est la relation entre les états et les ordres. Cette dernière permet de faire évoluer l'automate M. Chaque action doit être décrite afin de modéliser son effet sur le système,
- $E_0 \in \{\sum_{i=1}^{n} E_i\}$ est l'état initial du système,
- $E_{final} = \{E_0, \{E_{Int}\}\} \subset \{\sum_{i=1}^{n} E_i\}$ est l'ensemble des états finis du système. Il peut s'agir de l'état initial E_0 ou d'un état dans lequel plus aucune évolution n'est possible (états interdits $\{E_{Int}\}$).

L'algorithme, présenté en Figure 4 et basé sur cet automate, va appliquer pour un état donné E_i une action $O_j \in \{O\}$ dans le but d'explorer tous les états atteignables $\{E_{attei}\}$ du système. Si la prédiction mène à un état interdit $E_{i+1} \in \{E_{Int}\}$ alors le contexte, composé de l'état E_i et de l'action O_j, est sauvegardé. De plus, si un état E_{i+1} a déjà été rencontré plus tôt alors l'algorithme ne le retient pas pour la prochaine itération. Il en est de même si la projection amène à l'état initial ou à un état interdit. Ainsi, l'exploration des états lors de la prochaine itération conduira à des états non rencontrés. La vitesse de traitement est ainsi améliorée, l'explosion combinatoire est évitée et une unique branche est explorée à chaque fois. Au final, un arbre représentant toutes les évolutions possibles du système est obtenu. Ces évolutions peuvent être différenciées en 3 types d'états possibles.

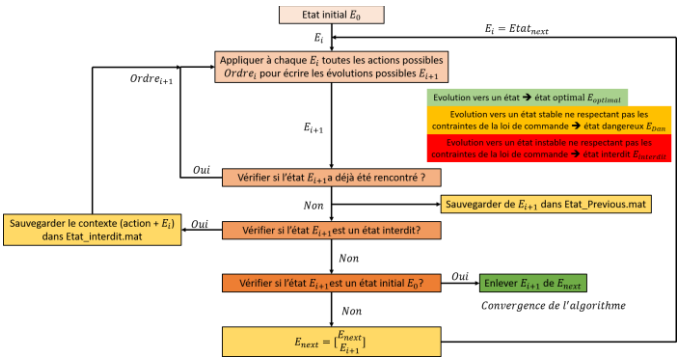

Figure 4. ILLUSTRATION DU FONCTIONNEMENT DE L'ALGORITHME D'EXPLORATION D'ETATS

C. Etape 3 : Synthèse des filtres

Sur la base du précédent paragraphe, nous pouvons écrire les règles qui décrivent les contraintes du procédé et qui seront implantées dans le filtre de commande. Pour écrire une règle $R = \{\sum_{k=1}^{l} R_k\}$ le modèle du procédé est utilisé et la liste de contexte menant vers un état interdit. Ainsi, nous avons :

$$E_i \in \{E | E_{Int}\}, O_j \in \{O\}\ t.q.\ \delta(E_i, O_j) \in E_{pro}$$

Les règles peuvent être décrites comme une matrice où :

$$\begin{cases} \forall (a,b) \in \mathbb{N} \\ 1 \le a \le n \\ 1 \le b \le m \end{cases}, R = (r_{a,b}) = vrai \ \leftrightarrow\ \delta(E_a, O_b) \notin \{E_{Int}\}$$

Les règles sont une mesure de sécurité statique. Le filtre de commande, dans lequel elles seront implantées, évaluera le contexte courant du système et le comparera aux règles implantées. Si le contexte respecte ces dernières alors l'ordre O_j est transmis à la PO. Toutefois, nous pourrions anticiper et quantifier le rapprochement d'un ICS d'états non-sûrs en calculant la distance entre l'état courant et des états interdits ou optimaux. En effet, dans le cas d'une attaque, un hacker cherchera toujours à amener l'ICS dans un état interdit afin de le rendre inopérant. Ainsi, dans un tel cas, la distance entre l'état courant et un état interdit ne cessera de se réduire au contraire d'une défaillance matérielle où elle se stabilisera. Ainsi, en reprenant la notion de distance développée dans [14] à l'aide du modèle du procédé, nous pouvons calculer dynamiquement la distance d'un état courant avec des états interdits. Cependant, en prenant en considération le modèle de commande, nous pouvons d'autant plus détecter des déviations. Ainsi, une attaque modifiant les ordres envoyés à la PO mais ayant les mêmes effets

serait anticipée. Pour cela, nous créons un vecteur distance d prenant en considération les 2 modèles :

$$d \in \mathbb{R}^+, d_1 \in \mathbb{R}^+, d_v \in \mathbb{R}^+, (i,j) \in \mathbb{N}\ and\ (s,t,o,p) \in \mathbb{R}^n$$

$$d = \begin{cases} d_1 = \sum_{i=1}^{n} |s_i - t_i| + \sum_{j=1}^{m} |o_j - p_j| \\ d_v = \#\{i,\ s_i \neq t_i\} + \#\{j,\ o_j \neq p_j\} \end{cases}$$

Ce vecteur nous permet de quantifier la distance entre l'état courant et les états optimaux et interdits du système à un instant t. En regardant l'évolution de cette distance, le concept de trajectoire est alors établi, il permettra, en plus de détecter, de discriminer une attaque d'une défaillance.

Afin d'assurer une redondance avec les contextes trouvés lors de la phase d'exploration des états (étape 2), nous implémentons dans le filtre de commande les fonctions permettant de prédire l'état futur du système en fonction de l'état courant et de l'ordre envoyé. A l'issue de cette phase, la distance par rapport à l'état optimal attendu et à l'état interdit le plus proche est calculée. Enfin, le contexte est testé pour savoir s'il respecte les règles du système. Pour le filtre de CR, seul le calcul de distance est implanté dans le filtre. Il renvoie uniquement une information sur la distance (entre l'état courant et l'état attendu) aux opérateurs.

V. EXEMPLE DE SIMULATION

Afin d'illustrer l'approche proposée dans la section précédente nous nous appuyons sur un exemple bien connu de la littérature [7]. Le système est composé de 3 cuves. Les cuves T_1 et T_2, de capacité infinie, contiennent respectivement un produit A et un produit B. Chacune des cuves déverse leur produit dans la cuve de mélange T_3 dans le but de produire le produit C. La phase de remplissage est assurée par l'ouverture des vannes V_1 et V_2. Des capteurs de niveaux permettent de suivre l'évolution du remplissage. Lorsque le capteur H_2 est actif, la vanne V_3 s'ouvre pour vidanger la cuve T_3. Les 3 capteurs de niveau $H_0(LL), H_1(H)\ and\ H_2(HH)$ sont utilisés pour gérer l'ouverture des vannes. Il existe un seul état interdit pour ce système qui est atteint lorsque le niveau dans la cuve T_3 dépasse H_2. Cet exemple est illustré en Figure 5.

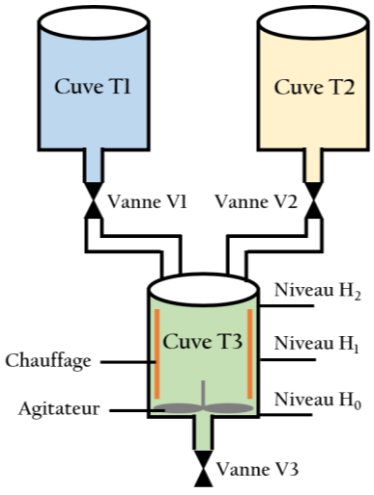

Figure 5. SCHEMA ILLUSTRANT L'EXEMPLE DES CUVES

Nous nous limiterons à ce fonctionnement pour illustrer notre méthode de conception et de détection. Ainsi, les vecteurs d'état : $[Capteur\ H_0, Capteur\ H_1, Capteur\ H_2]$, d'ordre : $[Vanne\ V_1, Vanne\ V_2, Vanne\ V_3]$ et de transition : $[E_{remplissage}, E_{vidange}]$ sont écrits.

Durant la phase d'exploration des états du système, nous avons 8 états possibles :

- Ensemble des états inatteignables : $\{E_{inat}\}$: {[0 0 0], [0 1 0], [0 0 1], [1 0 1], [0 1 1]},
- Ensemble des états atteignables : $\{E_{attei}\}$ avec $\{E_{Opt}\}$: {[1 0 0], [1 1 0], [1 1 1]}, $\{E_{Dan}\}$: {∅}, $\{E_{Int}\}$: $\{E_{débordement}\}$.

Les contextes menant à des états interdits $\{E_{Int}\}$ sont trouvés par l'algorithme d'exploration et regroupés en Tableau 1.

Etats interdits $\{E_{Int}\}$	**Ordre** O_j	**Etat** S_i **du procédé avant l'action** O_j
Cas 1 : un ordre uniquement		
Débordement	Ouverture V_1 ou ouverture V_2 : [1 0 0] ou [0 1 0]	Capteur H_2 activé : [1 1 1]
Cas 2 : ordres en simultanés		
Débordement	Ouverture V_1 ou ouverture V_2 : [1 0 0] ou [0 1 0]	Capteur H_2 activé : [1 1 1]
	Ouverture V_1 et V_2 : [1 1 0]	Capteur H_1 activé : [1 1 0]

Tableau 1. ANALYSE DE L'ANALYSE DE RISQUES

L'automate déterministe M représentant le fonctionnement du système est construit :

$$M = \{\{\sum_{i=1}^{n} E_i\}, \{\sum_{j=1}^{3} O_j\}\ , \delta, E_0, E_{final}\}$$

- $\{\sum E_i\}$: $[Capteur\ H_0, Capteur\ H_1, Capteur\ H_2]$
- $\{\sum_{j=1}^{3} O_j\}$: $[Valve\ V_1, Valve\ V_2, Valve\ V_3]$
- δ :

$$\forall i \in [0,3], \forall k \in [1,2]\ tel\ que \begin{cases} E_i(1,k) = 1 \\ E_i(1,k+1) = 0 \end{cases},$$
$$\begin{cases} \delta(E_i, O_1) = E_{i+1}\ avec\ E_{i+1}(1,k+1) = E_i(1,k+1) + 1 \\ \delta(E_i, O_2) = E_{i+1}\ avec\ E_{i+1}(1,k+1) = E_i(1,k+1) + 1 \end{cases}$$
$$\forall\ i \in [0,3],\ \ \delta(E_i, O_3) = E_{i+1} = E_0$$

- E_0 : [1 0 0],
- E_{final}: $\{E_0, E_{pro}\}$.

Afin de tester notre algorithme, qui diffère en fonction de la localisation du filtre dans l'architecture de contrôle commande, nous allons simuler une attaque « Man in the middle » sur la commande du système présenté. Lorsque l'attaque est lancée, tous les ordres sont interceptés et remplacés par des ordres prédéfinis. Dans notre exemple, les ordres $O_j \in O_{Opt}$ sont remplacés par l'ordre $O_{attack} \in \{O\}$ qui rejouera le premier ordre de la séquence $O_1 = [1\ 0\ 0]$ qui ouvre V_1. Au premier cycle automate, l'ordre O_{attack} est émis entrainant l'ouverture

de la vanne V_1 et le remplissage de la cuve T_3. Le filtre de commande ne détecte aucune déviation par rapport au modèle de commande et du procédé parce qu'il correspond à l'ordre O_1 (ordre optimal). Ainsi, le système va évoluer d'un état initial à un état optimal, la distance par rapport à l'état optimal est nul. Lorsque le capteur de niveau H_1 est actif, l'ordre d'ouvrir la vanne V_2 est émis et remplacé par O_{attack}. Le filtre de commande détecte une dérive par rapport au modèle de commande $O_{attack} \neq O_{Opt}$, toutefois, l'état prédit correspond à un état optimal $E_{Pre} = E_{Opt} = [1\ 1\ 0]$. Ainsi, la distance de l'état optimal s'éloigne à 1 et la distance minimale à des états interdits est aussi de 1. L'ordre est tout de même émis puisque le contexte ne correspond pas à ceux menant vers des états interdits. Une alerte est envoyée aux opérateurs pour les alerter que le système s'éloigne de la trajectoire optimale (vanne V_1 ouverte à la place de V_2). Suite à cela, le niveau de cuve active le capteur H_2, l'ordre de vidange est émis par l'API et modifié par l'attaquant. Le filtre de commande détecte une dérive par rapport à la loi de commande et dans ce cas, le contexte mène vers un état interdit (débordement de la cuve). Comme une règle est violée alors la distance par rapport à un état interdit est nulle alors l'ordre O_{attack} est stoppé par le filtre de commande. Nous avons ainsi protégé le système et son environnement d'une attaque. Le comportement du filtre de CR n'est pas évoqué puisqu'il ne détecte aucune déviation par rapport aux états attendus et optimaux dans notre exemple.

VI. CONCLUSION

Dans ce papier, nous avons présenté une nouvelle approche, située entre le niveau 1 et 0, permettant de détecter des cyberattaques contre des ICS. Le point principal de cette approche est la notion de distance entre les états d'un système. Cette distance nous permet de détecter une attaque dans la mesure où un attaquant cherchera toujours à amener le système dans un état dégradé. Cette notion de distance permet d'introduire le concept de trajectoire qui est basé sur l'évolution au cours du temps de cette distance. Elle permettra dans de prochains travaux de discriminer si le système fait face à des cyberattaques ou non.

L'approche développée permet également la détection d'attaque en se basant sur une connaissance du procédé physique sous forme de modèles. Ces modèles permettent d'aboutir à des règles qui empêchent l'exécution d'ordres dangereux. Les concepts de distance et de trajectoire seront améliorés afin de discriminer l'origine des détections. Enfin, les résultats de simulation valident l'approche proposée, héritée de l'approche filtre. D'autres expérimentations seront prochainement conduites sur des plateformes industrielles.

VII. REFERENCES

[1] Y. Fourastier et L. Pietre-Cambacedes, *Cybersécurité des installations industrielles : défendre ses systèmes numériques*. Cépaduès Editions, 2015.

[2] S. McLaughlin *et al.*, « The Cybersecurity Landscape in Industrial Control Systems », *Proc. IEEE*, vol. 104, nº 5, p. 1039-1057, mai 2016.

[3] N. Falliere, L. O. Murchu, et E. Chien, « W32. stuxnet dossier », *White Pap. Symantec Corp Secur. Response*, vol. 5, p. 6, 2011.

[4] ICS-CERT, « ICS-CERT / The Industrial Control Systems Cyber Emergency Response Team », 15-sept-2016. [En ligne]. Disponible sur: https://ics-cert.us-cert.gov/. [Consulté le: 15-sept-2016].

[5] I. N. Fovino, A. Coletta, A. Carcano, et M. Masera, « Critical State-Based Filtering System for Securing SCADA Network Protocols », *IEEE Trans. Ind. Electron.*, vol. 59, nº 10, p. 3943-3950, oct. 2012.

[6] M. Caselli, E. Zambon, et F. Kargl, « Sequence-aware Intrusion Detection in Industrial Control Systems », in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, New York, NY, USA, 2015, p. 13–24.

[7] W. Li, L. Xie, Z. Deng, et Z. Wang, « False sequential logic attack on SCADA system and its physical impact analysis », *Comput. Secur.*, vol. 58, p. 149-159, mai 2016.

[8] Y. Wang, Z. Xu, J. Zhang, L. Xu, H. Wang, et G. Gu, « SRID: State Relation Based Intrusion Detection for False Data Injection Attacks in SCADA », in *Computer Security - ESORICS 2014*, M. Kutyłowski et J. Vaidya, Éd. Springer International Publishing, 2014, p. 401-418.

[9] D. E. Denning, « An Intrusion-Detection Model », *IEEE Trans. Softw. Eng.*, vol. SE-13, nº 2, p. 222-232, février 1987.

[10] R. Mitchell et I.-R. Chen, « A survey of intrusion detection techniques for cyber-physical systems », *ACM Comput. Surv.*, vol. 46, nº 4, p. 1-29, mars 2014.

[11] S. Pan, T. H. Morris, U. Adhikari, et V. Madani, « Causal Event Graphs Cyber-physical System Intrusion Detection System », in *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, New York, NY, USA, 2013, p. 40:1–40:4.

[12] H. Hadeli, R. Schierholz, M. Braendle, et C. Tuduce, « 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*, 2009, p. 503-510.

[13] K. Xiao *et al.*, « 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*, 2007, p. 4.

[14] A. Carcano, A. Coletta, M. Guglielmi, M. Masera, I. Nai Fovino, et A. Trombetta, « A Multidimensional Critical State Analysis for Detecting Intrusions in SCADA Systems », *IEEE Trans. Ind. Inform.*, vol. 7, nº 2, p. 179-186, mai 2011.

[15] D. Cruette, J. P. Bourey, et J. C. Gentina, « Hierarchical specification and validation of operating sequences in the context of FMSs », *Comput. Integr. Manuf. Syst.*, vol. 4, nº 3, p. 140–156, 1991.

[16] P. Marangé, « Synthèse et filtrage robuste de la commande pour des systèmes manufacturiers sûrs de fonctionnement », Université de Reims, 2009.

[17] S. Henry, E. Zamaï, et M. Jacomino, « Logic control law design for automated manufacturing systems », *Eng. Appl. Artif. Intell.*, vol. 25, nº 4, p. 824-836, juin 2012.