Détection d'intrusion dans les systèmes industriels: Suricata et le cas de Modbus

David Diallo[1,2] et Mathieu Feuillet[1]

[1] Agence nationale de la sécurité des systèmes d'information
prenom.nom@ssi.gouv.fr
[2] École supérieure d'informatique, électronique, automatique

Résumé Les systèmes industriels offrent des caractéristiques très propices à la détection d'intrusion : une définition très précise du fonctionnement et une évolution lente du système d'information. Dans ce papier, nous proposons de détailler les règles de détection possibles dans un système industriel sur un exemple : le protocole Modbus. Ces règles ont été implémentées dans l'IDS Suricata et mises en œuvre sur un système réel.

Mots clés: IDS, SCADA, Modbus, Suricata, Systèmes industriels.

1 Introduction

Les systèmes industriels ou ICS [3] sont des systèmes d'information ayant pour finalité de piloter des procédés industriels au moyen de capteurs et d'actionneurs. Ils ont la caractéristique de pouvoir engendrer des actions physiques, contrairement aux systèmes d'information classiques. Les premières architectures étaient constituées de technologies spécifiques, souvent propriétaires, et étaient relativement isolées des autres systèmes d'information de l'entreprise. Aujourd'hui, notamment pour des raisons de coûts, les systèmes industriels utilisent de plus en plus des technologies développées pour des systèmes d'information classiques.

En contre-partie, les systèmes industriels deviennent vulnérables aux mêmes attaques que les systèmes d'information classiques [6]. Enfin, les systèmes industriels sont de plus en plus connectés avec les réseaux de gestion des entreprises, ce qui les exposent à des attaques à distance [7].

Les incidents se multipliant, une véritable prise de conscience sur la nécessité de sécuriser les systèmes industriels a émergé. De nombreuses initiatives apparaissent en ce sens. Ainsi, l'ANSSI a publié des guides de sensibilisation [1,2], de classification des installations [4] et de mesures à appliquer pour les sécuriser [3].

Les systèmes industriels ont des spécificités telles que les mesures usuelles de sécurité des systèmes d'information ne peuvent pas être appliquées systématiquement. Par exemple, il existe souvent des contraintes fortes de temps de réponse pour certains échanges, parfois de l'ordre de quelques dizaines de micro-secondes, qui rendent difficile l'usage de cryptographie pour assurer l'intégrité des données.

3. *Industrial Control Systems.*

De même, l'usage d'un simple équipement de filtrage peut introduire parfois une latence inacceptable.

Par ailleurs, la modification ou la mise à jour d'un système industriel ne peut pas se faire de manière aussi souple qu'un système d'information classique. Nombre de ceux-ci fonctionnent en continu et ne sont arrêtés que pour des opérations planifiées de maintenance. Ainsi, en cas d'apparition d'une vulnérabilité sur un équipement, il ne sera pas possible de mettre à jour celui-ci rapidement et le système industriel restera vulnérable.

Enfin, s'il est tout à fait envisageable d'intégrer l'ensemble des mesures de défense en profondeur préconisées dans le guide [3] sur les systèmes industriels à venir, ceci n'est pas forcément vrai pour les systèmes déjà existants. Or, la durée de vie typique d'un tel système est de plusieurs décennies et il faut trouver d'autres solutions pour atteindre un niveau de sécurité acceptable dans ces cas-là.

Pour toutes ces situations, le développement de systèmes de détection d'intrusion offre des caractéristiques très intéressantes car il permet d'augmenter les capacités de surveillance sans perturber le fonctionnement et permet de réagir rapidement en cas d'évolution de la menace.

Afin d'évaluer la pertinence d'un tel développement, une preuve de concept a été réalisée en implémentant un moteur de détection complet pour le protocole Modbus à l'intérieur de l'IDS [4] Suricata [15]. Ce préprocesseur est en cours d'intégration dans le projet. L'objectif n'est pas de détecter seulement les violations protocolaires ou les tentatives d'exploitation de vulnérabilités connues mais de proposer un jeu de règles permettant de décrire le comportement admissible du système industriel.

Les systèmes industriels disposent déjà d'une supervision, souvent appelée système SCADA [5]. Cette supervision récupère des informations au travers de requêtes aux différents éléments du système (automates, entrées/sorties) afin d'informer l'opérateur sur l'état du système industriel. Cette supervision peut être utilisée pour détecter certains comportements anormaux du procédé. Ceci est une approche complémentaire de l'IDS qui va observer le trafic sans interagir avec le système industriel. Une réflexion doit être menée sur l'articulation de ces deux systèmes de supervision.

Dans la section 2, un état de l'art succinct des différentes techniques de détection d'intrusion est proposé. Dans la section 3, des éléments de réflexion sont présentés sur les architectures de détection et sur l'articulation possible entre système SCADA et IDS. Dans la section 4, des détails sur le protocole Modbus, sont présentés ainsi que l'implémentation et des exemples d'utilisation pour détecter des évènements anormaux dans le système industriel. Dans la section 5, les résultats de tests de performance effectués sont donnés. Quelques perspectives sont présentées dans la section 6.

---

4. *Intrusion Detection System.*

5. *Supervisory Control And Data Acquisition.* Cet acronyme est souvent utilisé pour désigner les systèmes industriels dans leur ensemble.

2 Détection d'intrusion pour les systèmes industriels

La détection d'intrusion consiste à déceler toute tentative d'atteinte à l'intégrité, à la disponibilité ou éventuellement à la confidentialité du système industriel. Les systèmes de détection d'intrusion ou IDS se décomposent en deux grandes familles : les systèmes de détection sur les équipements ou HIDS [6] et les systèmes de détection sur le réseau ou NIDS [7]. Un état de l'art détaillé est présenté dans [9]. Nous nous concentrons ici sur la détection d'intrusion en analysant le trafic du réseau.

Comme expliqué dans [8], l'efficacité d'un IDS peut se mesurer à l'aide de trois métriques : le taux de faux positifs appelé *pertinence*, le taux de faux négatifs appelé *complétude* et le débit maximum que l'IDS peut analyser appelé *performance*. La logique des IDS peut de plus être classée en deux grandes catégories : l'approche comportementale qui cherche à représenter le fonctionnement normal du système d'information et à vérifier que le système s'y conforme et l'approche par scénarios qui consiste à inventorier une liste d'attaques possibles sur le système et à mettre en place des règles pour les détecter.

Aujourd'hui, la plupart des IDS commerciaux utilisent une approche par scénarios avec une base de signatures de vulnérabilités connues. L'IDS ne conserve pas d'état à proprement parler et chaque requête ou réponse est analysée de manière indépendante. Ainsi Emergency Threats [10] fournit une base de signatures d'attaques connues contre les systèmes industriels pour les IDS Snort [16] et Suricata [15]. Cette technique offre souvent des performances intéressantes et une bonne pertinence mais une mauvaise complétude, notamment du fait de son incapacité à détecter une vulnérabilité inconnue.

De par leur comportement parfaitement défini et leur évolution lente, les systèmes industriels se prêtent extrêmement bien à l'approche comportementale. Dans les articles [5] et [14], les auteurs définissent différents types de règles pour le cas du protocole Modbus. Ces règles permettent de vérifier les conformités protocolaire et comportementale du système.

Par ailleurs, des méthodes avancées proposent d'analyser l'état du système industriel et de vérifier que celui-ci ne s'approche pas d'un état dangereux (voir [9]). Cette approche peut paraitre séduisante mais relève de la surveillance du processus qui incombe au SCADA. Il convient de prendre garde à ce que l'IDS ne cherche pas à remplacer celui-ci.

En se basant sur les règles proposées dans les articles [5] et [14] complétées avec de nouvelles règles, nous avons implémenté la détection d'intrusion sur le protocole Modbus à l'intérieur de l'IDS Suricata [15]. L'objectif était d'analyser la complexité d'écriture d'un jeu de règles pour arriver à un niveau de pertinence et de complétude acceptable. Par ailleurs, la performance d'un IDS dépend fortement du nombre de règles. Des tests de performance ont été réalisés sur un cas réel. Cependant, en premier lieu, il est important de mener une réflexion

6. *Host Intrusion Detection System.*
7. *Network Intrusion Detection System.*

sur le positionnement des sondes de détection dans l'architecture du système industriel.

3 Architectures de détection

3.1 Éléments sur la menace

Afin de bien positionner les sondes, il convient d'évoquer rapidement les menaces considérées pour un système industriel tel que rencontré actuellement en pratique. Pour la suite de cette section, nous considérerons une architecture représentative telle que schématisée sur la figure 1. Sur cette dernière, on retrouve une architecture classique avec une séparation entre le réseau de supervision sur lequel se trouvent les clients SCADA, les stations d'ingénierie, les serveurs d'historique, etc. et le réseau de procédé sur lequel se trouve également le serveur SCADA ainsi que les automates et les IHM permettant de contrôler le procédé au plus près des installations.

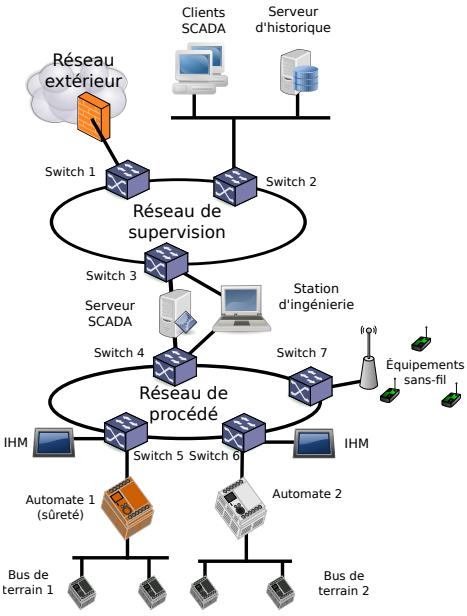


FIGURE 1: Architecture typique d'un système industriel.

Nous considérons ici que l'objectif de l'attaquant est de porter atteinte à l'intégrité ou à la disponibilité du procédé piloté et contrôlé par le système in-

dustriel. Nous supposons également que l'attaquant n'a pas accès physiquement aux bus de terrain avec les capteurs et les actionneurs [8].

Dans ces conditions, les cibles de l'attaquant seront les automates programmables industriels car, dans la plupart des systèmes industriels, ils pilotent le procédé industriel et l'on peut perturber le fonctionnement du procédé industriel

— en modifiant leur mode de fonctionnement (STOP/MARCHE) ;
— en modifiant leur programme automate ou leur *firmware* ;
— en le mettant hors service en exploitant une vulnérabilité.

Pour porter atteinte aux automates, deux pistes essentielles doivent être considérées :

— une corruption lors d'une phase de maintenance ;
— une attaque par le réseau (depuis l'extérieur, par corruption d'un équipement tiers, etc.).

Dans le cas d'une attaque par le réseau, plusieurs scénarios sont possibles. Le cas le plus probable est celui de l'attaquant arrivant par l'extérieur. Dans ce cas, il devra compromettre plusieurs équipements pour atteindre les automates : la passerelle d'interconnexion si elle existe, un équipement dans le réseau de supervision puis le serveur SCADA qui est en coupure entre le réseau de supervision et celui de procédé. Un cas un peu moins probable est celui de l'attaquant à portée du réseau sans-fil qui peut alors tenter de le compromettre. Il sera alors directement sur le réseau de procédé. Enfin, le cas le moins probable est celui de l'attaquant ayant accès physiquement au réseau de procédé, en mesure de connecter directement un équipement pirate sur celui-ci.

3.2 Les rôles de l'IDS et du système SCADA

Comme évoqué dans la section 2, les systèmes industriels disposent déjà d'un système de supervision du procédé : le système SCADA. Celui-ci collecte des informations récupérées auprès des automates, stocke un historique plus ou moins important et présente ces informations à l'opérateur pour qu'il puisse prendre des décisions. Dans certains cas, le système SCADA récupère directement les informations depuis les équipements de terrain. Il est donc en théorie tout à fait envisageable d'avoir un système SCADA dédié à la cybersécurité. L'utilisation des mêmes outils que ceux auxquels les opérateurs sont habitués permet de simplifier leur formation et accélère l'appropriation qu'ils peuvent en faire.

De par ses fonctions usuelles, un « système SCADA de cybersécurité » serait le meilleur candidat pour surveiller les déviations du procédé industriel, remonter les alarmes en cas de défaillance d'un équipement de terrain. Si l'on souhaite s'assurer du comportement normal de l'automate, il peut être envisageable pour le SCADA de vérifier la cohérence des informations collectées sur le terrain par rapport au comportement attendu de l'automate. Ainsi, une usure prématurée d'équipements due à des ordres dangereux de l'automate pourrait être repérée en analysant les informations reçues directement des capteurs au travers d'une

8. On peut noter que dans ce cas là, l'attaquant pourrait également directement porter atteinte au procédé industriel, limitant l'intérêt d'une attaque informatique

passerelle de conversion connectant le bus de terrain au réseau de procédé, par exemple. En particulier, le SCADA de cybersécurité est donc sans doute le plus adapté pour détecter une compromission d'équipement lors d'une phase de maintenance car elle ne peut être détectée que par la modification du comportement de l'équipement [9]

Cependant, le système SCADA de cybersécurité n'ayant pas accès au trafic reçu par les différents équipements présents sur le réseau mais uniquement aux informations qu'il récupère, il ne sera jamais à même de détecter un certain nombre d'attaques. Par exemple, une modification d'un programme automate ne sera pas forcément décelable si l'automate n'envoie pas de notification, une exploitation d'une vulnérabilité peut également ne pas être décelable. Enfin, ce système SCADA ne pourra jamais détecter des modifications des flux entre équipements. Par exemple, la disparition d'un flux entre deux automates ou des requêtes illégitimes depuis des machines vers des automates ne pourront pas être détectées. C'est sur ces aspects que l'IDS va pouvoir apporter des informations complémentaires.

Enfin, l'IDS et le système SCADA de cybersécurité ayant des points de vue différents sur le système industriel, cela peut permettre un recoupement des informations facilitant l'analyse d'un incident. À titre d'exemple, un système SCADA peut remonter une alarme en cas de détection d'une indisponibilité d'un automate. Celle-ci peut être inexpliquée et ne pas laisser d'informations exploitables dans les journaux d'évènements. Si en parallèle, l'IDS peut signaler du trafic illégitime, le diagnostic sera plus rapide.

3.3 Positionnement des sondes

Le placement des sondes est une question cruciale pour assurer la pertinence du système de détection dans sa globalité. Cependant, la multiplication des sondes augmente le coût, la quantité de travail nécessaire pour la configuration et la difficulté d'exploitation des alertes. Nous proposons ici quelques éléments de réflexion sur l'architecture et notamment sur les emplacements des sondes par ordre d'importance. L'architecture globale de détection est représentée sur la figure 2.

Il ressort des éléments de réflexion sur les menaces considérées qu'une sonde doit être placée au niveau de la passerelle d'interconnexion entre le système industriel et le réseau extérieur (Sonde S1 sur la figure 2). Il s'agit d'une sonde classique qui pourrait être gérée par les équipes des services informatiques généraux (par exemple les exploitants de l'informatique de gestion).

En second lieu, une sonde doit être placée de façon à analyser le trafic en provenance du système SCADA et de la station d'ingénierie et à destination des automates (Sonde S2 sur la figure 2). Dans le cas où un cloisonnement logique est effectué et où les automates n'ont pas besoin de communiquer entre eux, une telle sonde est en mesure d'analyser l'intégralité du trafic à destination

9. On pourrait également envisager un contrôle d'intégrité après chaque opération de maintenance mais peu d'équipements offrent cette possibilité aujourd'hui.

des automates. Cependant, une telle sonde ne sera pas en capacité de détecter une compromission du SCADA ou de la station d'ingénierie si l'attaquant utilise ensuite des actions légitimes pour perturber le fonctionnement du procédé industriel.

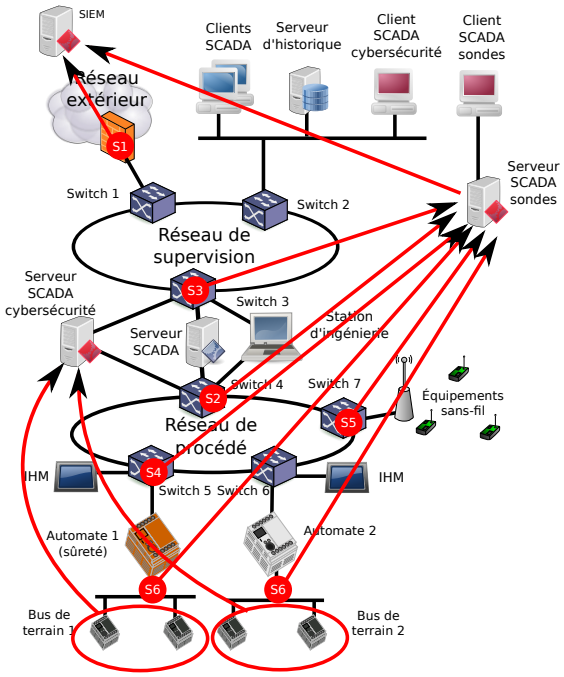


FIGURE 2: Architecture de détection.

En troisième lieu, la station d'ingénierie et le serveur SCADA étant des vecteurs naturels pour attaquer les automates, il est opportun de mettre une sonde au niveau du commutateur 3 pour détecter tout trafic anormal à destination de ces équipements (Sonde S3 sur la figure 2).

Enfin, lorsque nous sommes en présence de liens sans-fil ou d'automates particulièrement sensibles comme des automates de sûreté (systèmes instrumentés de sûreté), il est intéressant de placer une sonde au plus près des équipements pour surveiller le trafic qui y entre ou en sort (Sondes S4 et S5 sur la figure 2).

Enfin, dans le cas où certains actionneurs sont plus critiques et nécessitent une surveillance particulière lorsque l'on craint la compromission d'un équipement sur le bas de terrain, on peut mettre une sonde qui analyse le trafic sur ce bus (Sondes S6 sur la figure 2).

3.4 Traitement centralisé ou distribué

Une fois que les emplacements des sondes sont choisis, il faut réfléchir à la façon de traiter le trafic capté. Deux systèmes sont possibles. Dans le cas d'un traitement centralisé, les sondes ne sont pas des IDS à proprement parler mais effectuent un traitement minimal sur le trafic avant de tout faire remonter à un IDS centralisé qui va effectuer cette analyse. Cette architecture présente l'intérêt de permettre la corrélation entre les différents points de capture sur le réseau. Elle facilite la maintenance des sondes qui sont alors très simples. Cependant, le passage à l'échelle sur des installations de grande taille n'est pas forcément évident. En effet, les petites sondes vont générer un trafic équivalent au trafic qu'elles analysent et le réseau des sondes devra être dimensionné pour supporter l'intégralité d'un trafic généré. De manière similaire, l'IDS centralisé devra également être dimensionné pour traiter l'intégralité du trafic généré par le système industriel.

La deuxième approche possible consiste à ce que les sondes soient des IDS complets analysant le trafic localement et ne remontant que les alertes. Le trafic résultant de ce type d'analyse est beaucoup plus faible dès lors que les règles de détection sont choisies correctement. Le deuxième avantage d'un tel système est le passage à l'échelle automatique car le nombre de sondes croît en même temps que la taille de l'installation. Si un tel système ne permet pas la même souplesse que l'approche centralisée en matière de corrélation entre les différentes sources de trafic capté, elle permet cependant la corrélation entre des alertes remontées par différents IDS. En revanche, il faut s'assurer qu'une sonde avec des caractéristiques techniques similaires à celles de pare-feu ou VPN industriels d'entrée de gamme peut analyser le trafic généré par des équipements de terrain comme les automates.

Dans les deux cas, afin de ne pas perturber le système industriel surveillé, l'ensemble des sondes sont placées dans un réseau dédié. Que le système soit centralisé ou distribué, les alertes doivent être présentées à un opérateur. Les SCADA étant déjà munis d'une supervision du procédé, il parait pertinent d'utiliser le même genre d'interface en créant un superviseur Sondes pour permettre aux opérateurs de s'approprier ce système plus rapidement. Les opérateurs pourraient alors faire le rapprochement entre les évènements de sécurité remontés par le superviseur Sondes ou le système SCADA Cybersécurité (cf section 3.2) avec les informations fournies par le système SCADA du procédé.

Par exemple, l'opérateur peut observer un plantage inexpliqué d'un équipement ; le signalement de trafic non-conforme sur le système SCADA Sondes permettra un diagnostic plus rapide et accélèrera la réaction. La seule exception étant la sonde S1 dont les alertes ne correspondent pas forcément à des évènements métier et qui ne sont donc pas forcément pertinentes pour les opérateurs.

Dans la suite de ce papier, nous allons présenter la preuve de concept qui a été effectuée en se basant sur un choix d'architecture distribuée. Le protocole choisi pouvant servir à la communication entre le système SCADA et les automates ou entre les automates et leurs entrées/sorties, cette preuve de concept peut s'appliquer aux sondes S2, S4, S5 et S6 représentées sur la figure 2.

4 Intégration de Modbus dans Suricata

4.1 Le protocole Modbus

Modbus est initialement un protocole série publié par Modicon en 1979 et ses spécifications [13] sont ouvertes. Depuis 1999, ce protocole a été porté sur TCP/IP [11]. Dans la suite, nous utiliserons le terme Modbus uniquement pour parler de cette variante.

Ce protocole fonctionne sur le modèle du client (aussi appelé maître) et du serveur (aussi appelé esclave). Le client envoie des requêtes auxquelles le serveur répond. Aucune connexion ne peut être initiée par le serveur. À réception de la requête, le serveur la traite avant de renvoyer une réponse au client. La réponse donne le résultat de la requête et, le cas échéant, les données demandées.

Chaque paquet TCP contient un ADU [10] avec une structure comme représentée sur la figure 3. L'en-tête MBAP [11] contient un identifiant de transaction pour pouvoir associer une réponse à la requête correspondante, un champ `Protocole` correspondant à la version du protocole Modbus, la longueur du PDU [12] Modbus contenu dans l'ADU [13] ainsi qu'un champ pour les passerelles Modbus (appelé `Unité`).

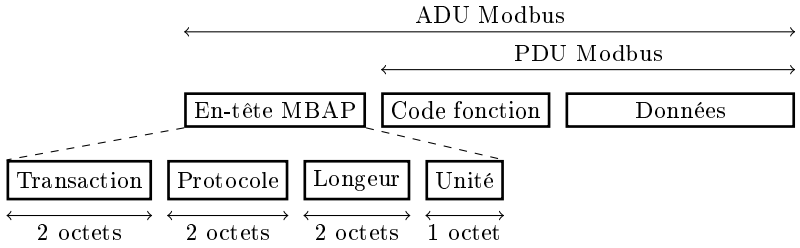


FIGURE 3: Structure d'un ADU Modbus et de l'en-tête MBAP.

Le PDU, quant à lui, contient un code identifiant une fonction et les données associées. Les fonctions sont réparties en trois catégories :

— les fonctions publiques, définies dans la spécification [13] ;
— les fonctions utilisateur, destinées à des besoins internes ;
— les fonctions réservées, utilisées par des fournisseurs pour leurs produits mais qui ne sont pas documentées.

L'essentiel des fonctions publiques ont pour objet de permettre au client de lire ou d'écrire dans des registres présents sur le serveur.

10. La norme n'interdit pas de mettre plusieurs *Application Data Unit* (ADU) dans un paquet mais le déconseille fortement.
11. *Modbus Application Protocol.*
12. *Protocol Data Unit.*
13. Pour être tout à fait exact, il s'agit de la longueur du PDU plus un octet.

À titre d'exemple, la fonction `Read Coils` permet de lire des bits (appelés *coils* en Modbus) consécutifs. Comme indiqué sur la figure 4, lors de la requête, les données sont l'adresse de départ suivie de la quantité de bits à lire. Lors de la réponse, les données associées sont le nombre d'octets renvoyés, suivi des bits demandés avec éventuellement du bourrage pour compléter le dernier octet.

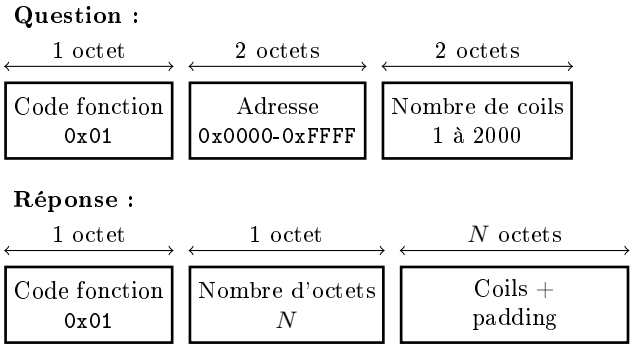


FIGURE 4: Requête et réponse de la fonction `Read Coils`.

La fonction symétrique est la fonction `Write Multiple Coils`. Comme indiqué sur la figure 5, dans la requête, le client spécifie l'adresse de départ, le nombre de bits et leurs valeurs. Dans la réponse, le serveur redonne l'adresse de départ et le nombre de bits.

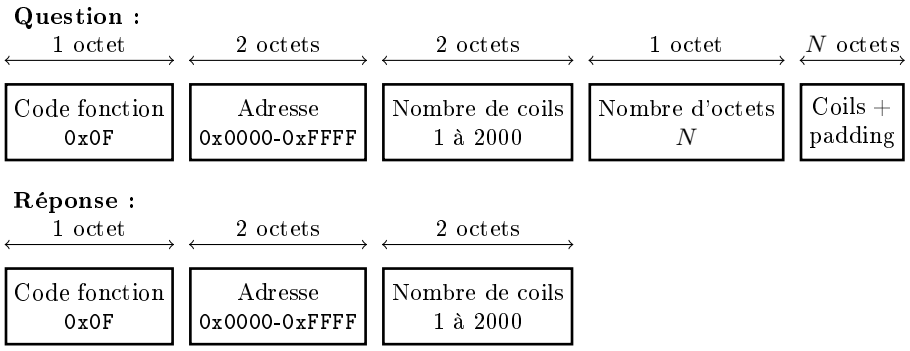


FIGURE 5: Requête et réponse de la fonction `Write Multiple Coils`.

Il existe également des fonctions de diagnostic, utiles pour la maintenance, mais qui ne permettent ni d'accéder ni de modifier le contenu des registres.

4.2 Conformité au protocole Modbus

À partir de la spécification de Modbus [13], il est possible d'établir des règles pour détecter des trames non conformes au protocole. Dans la suite, nous donnons quelques exemples pour illustrer les moyens disponibles pour vérifier la conformité des échanges à la norme.

Grâce à l'analyse de l'en-tête et à partir de la valeur des champs ou de la relation existante entre un ou plusieurs champs de la trame, il est possible de détecter un comportement anormal ou illicite.

En effet, le champ `Protocole` de la figure 3 représente la version du protocole, et conformément à la spécification [13], la version actuelle du protocole Modbus est 0. Toute valeur différente n'est donc pas possible et constitue une violation de la norme.

Le champ `Unité` est utilisé pour le routage inter-systèmes pour communiquer au travers d'une passerelle et sert à adresser des serveurs Modbus connectés en lien série. Dans le protocole Modbus sur lien série [12], les adresses comprises entre 248 et 254 ne sont pas possibles et constituent aussi une violation du protocole.

Au delà de l'en-tête, il est également possible de vérifier la cohérence entre des champs pour lesquels il existe des relations implicites ou explicites.

Par exemple, dans le cas d'une réponse à une requête `Read Coils` comme présentée sur la figure 4, le champ `Nombre d'octets` est forcément compris entre 1 et 250 car le nombre de coils est compris entre 1 et 2000 selon la norme [13]. Mais il est également possible de vérifier la relation implicite avec le champ `Longueur` de l'en-tête MBAP qui doit valoir $3 + N$ où N est la valeur du champ `Nombre d'octets` de la même trame.

Le protocole définit des échanges (appelés transactions) bidirectionnels et, à quelques exceptions près, une requête adressée à un serveur génère une réponse. Comme explicité au 4.1, aucune connexion ne peut être initiée par le serveur, donc il est possible de détecter une requête initiée par un serveur, une requête sans réponse ou une réponse orpheline à l'aide du champ `Transaction` qui doit être identique dans la requête et la réponse.

La cohérence entre la requête et la réponse peut être vérifiée plus en profondeur. En effet, le code fonction dans les deux trames doit être identique si la requête s'est exécutée correctement [14]. Dans le cas d'une transaction `Read Coils`, le champ `Nombre d'octets` de la réponse vaut $\lceil N/8 \rceil$ où N est la valeur du champ `Nombre de coils` de la requête associée.

L'ensemble de ces vérifications protocolaires ont été implémentées dans le préprocesseur Modbus pour Suricata et peuvent être utilisées de manière transparente à l'aide du jeu de règles fourni dans Suricata. Un exemple détectant les erreurs sur le champ longueur est donné dans la figure 6.

14. Dans le cas contraire, une exception doit être renvoyée d'après la spécification.

```
alert modbus any any -> any any
    (app-layer-event:modbus.invalid_length;
     msg:"Modbus invalid Length"; sid:5; rev:1;)
```

FIGURE 6: Règle de détection des longueurs invalides.

4.3 Vérification de conformité du processus industriel

Comme expliqué à la section 2, les systèmes industriels ont un comportement extrêmement bien défini qu'il est possible de décrire au niveau protocolaire afin que l'IDS puisse valider la conformité du trafic par rapport au comportement attendu. Cependant, la mise en place de règles fines pour un IDS nécessite d'avoir une cartographie qui n'a pas encore été établie pour de nombreux systèmes. Dans ces cas là, il est malgré tout possible de définir des comportements dangereux qui permettent de couvrir beaucoup de menaces. Par exemple, la règle donnée dans la figure 7 permet de détecter l'utilisation de fonctions réservées.

```
alert modbus any any -> any any
    (msg:"Diagnostic functions not allowed";
     modbus.function: reserved; sid:31; rev:1; )
```

FIGURE 7: Règle de détection des fonctions réservées.

Pour les systèmes industriels critiques, une des premières mesures préconisées par l'ANSSI est l'établissement d'une cartographie du système [4]. Elle peut être utilisée pour écrire des règles de détection de plus en plus précises. A l'aide de la matrice de flux, il est possible de distinguer les clients des serveurs et de spécifier la liste des serveurs que chaque client a le droit de contacter. À partir de là, tout échange entre deux machines non identifiées comme un couple client/serveur légitime sera anormal. Les règles associées sont données sur la figure 8.

```
alert modbus $MODBUS_CLIENT any -> !$MODBUS_SERVER
    any (msg:"Invalid Server"; sid:20; rev:1;)
alert modbus !$MODBUS_CLIENT any -> $MODBUS_SERVER
    any (msg:"Invalid Client"; sid:21; rev:1;)
```

FIGURE 8: Détection d'une communication illégitime.

Pour aller plus loin, pour chaque couple de client/serveur, il est possible de définir la liste des fonctions autorisées. Ainsi, en fonctionnement normal, l'usage de fonctions de diagnostic est considéré comme illégitime et doit générer des alertes. De même, certains clients peuvent ne pas avoir le droit d'écrire et l'usage de fonctions telles que `Write Multiple Coils` devra alors générer une alerte.

Enfin, pour chaque fonction autorisée, il est possible de spécifier les comportements autorisés. Ainsi, il est possible de définir des plages d'adresses accessibles en lecture ou en écriture. De plus, certaines valeurs peuvent avoir des bornes (par exemple, la vitesse maximum d'un moteur ou deux coils ne pouvant valoir 1 en même temps) et toute tentative d'écriture violant ces règles doit générer une alerte.

```
alert modbus any any -> $MODBUS_SERVER any
      (msg:"Holding regs write at address 8603 value >2000 not allowed";
       modbus.access: write holding, address 8603, value >2000;
       sid:52; rev:1; )
```

FIGURE 9: Détection de valeurs interdites à une adresse donnée.

5 Tests de performance

Comme expliqué à la section 2, trois critères sont utilisés pour évaluer la qualité d'un IDS pour les systèmes industriels. Parmi ceux-ci, se trouve la performance qui est la capacité de l'IDS à analyser le trafic réseau sans perte de paquet. L'objectif de cette section est de présenter des tests de performance qui ont été effectués sur Suricata.

5.1 Banc d'essai

Les tests de performances sont réalisés sur un système embarqué, Mirabox [15]. Le prix de ce boîtier et de ses accessoires est compris entre 100 et 200 euros. S'il n'est pas compatible avec un usage en milieu industriel, ses caractéristiques techniques et ses dimensions sont proches de celles des équipements industriels de sécurité tels que les pare-feu d'entrée de gamme.

Pour les tests, une Debian unstable de septembre 2014 avec un noyau Linux 3.15.8 a été installée. La version 2.0.3 de Suricata a été utilisée avec les développements du préprocesseur Modbus réalisés.

Le même trafic a été utilisé pour l'ensemble des tests et rejoué à différents débits en fonction des besoins. Il a été généré grâce à la machine virtuelle *Target Service* [16], de la société Digital Bond, qui simule un serveur Modbus et un client réalisé avec l'extension modLib [17] de Scapy. Il est composé de requêtes et réponses de la fonction `Write Single Register`.

Afin d'évaluer les performances de Suricata, différents scénarios ont été joués avec les règles suivantes :

15. `https://www.globalscaletechnologies.com`.
16. `http://www.digitalbond.com/tools/scada-honeynet/`.
17. `https://www.scadaforce.com/modLib.py`.

— une règle pour détecter l'usage de la fonction `Write Single Register`, générant ainsi une alerte pour chaque transaction, en effectuant une recherche de motif sans utiliser le préprocesseur Modbus ;
— une règle similaire à la précédente en utilisant le préprocesseur Modbus ;
— les règles permettant d'activer l'ensemble des vérifications de conformité au protocole Modbus ;
— les règles permettant d'activer l'ensemble des vérifications de conformité au protocole Modbus et cent règles identiques utilisant le préprocesseur Modbus mais ne générant pas d'alerte.

L'utilisation de cent règles identiques permet d'évaluer l'influence de leur nombre sur les performances de Suricata car il n'optimise pas les règles redondantes ; toute règle déclarée dans le fichier est évaluée.

5.2 Résultats

La première expérience vise à évaluer l'influence de l'utilisation du préprocesseur Modbus sur les performances. À cette fin, trois tests sont menés.

Le premier test consiste à utiliser la version officielle de Suricata sans le préprocesseur Modbus. La règle de détection de `Write Single register` par recherche de motif est utilisée. La courbe bleue de la figure 10a montre que l'équipement est capable d'analyser plus de 35 000 paquets par seconde sans perte.

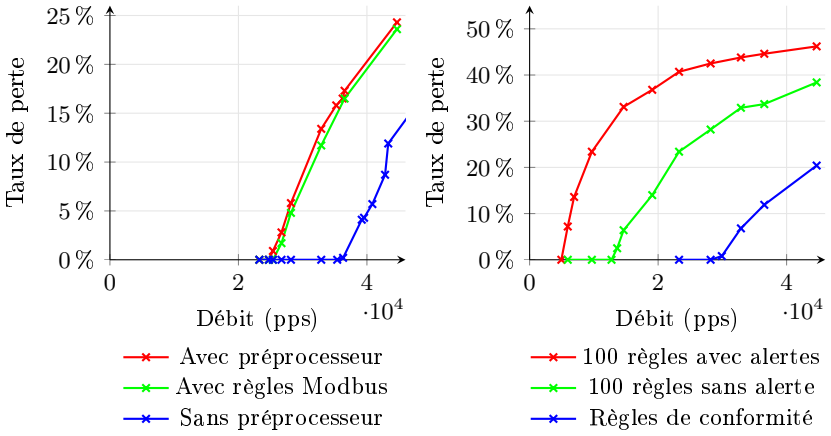
Pour le second test, le préprocesseur Modbus est présent mais la même règle de détection a été utilisée. La courbe rouge de la même figure montre que la capacité de traitement chute alors à 25 000 paquets par seconde soit environ 30 % de moins. Cette différence s'explique par le fait que, même si aucune règle Modbus n'a été spécifiée, le préprocesseur analyse les paquets transmis sur le port de Modbus.

Enfin, le dernier test vise à évaluer Suricata avec le préprocesseur et la règle équivalente utilisant le préprocesseur Modbus. La courbe verte montre que les résultats sont légèrement meilleurs que dans le test précédent. L'évaluation d'une règle avec les mots-clés Modbus est légèrement moins coûteuse que la règle équivalente avec les options génériques.

On peut noter que la génération d'une alerte pour chaque requête grève fortement les performances et que dans un cas d'usage plus réaliste avec peu d'alertes, la capacité de traitement est bien meilleure comme le montre l'expérience suivante qui cherche à caréctériser l'influence du nombre d'alertes générées et du nombre de règles.

Le premier test est effectué avec le préprocesseur Modbus et les règles de vérification de la conformité protocolaire. Le trafic étant conforme, aucune alerte n'est générée pendant ce test. La courbe bleue sur la figure 10b montre que la capacité de traitement dans ce cas est légèrement inférieure à 30 000 paquets par seconde.

Pour le deuxième test, cent règles sont ajoutées en plus des règles de conformité mais aucune alerte n'est générée. La capacité de traitement chute alors à environ 13 000 paquets par seconde.



(a) Performance du préprocesseur.

(b) Performance des règles.

FIGURE 10: Évaluation de performances du préprocesseur Modbus.

Enfin, pour le dernier test, les cent règles sont remplacées par cent autres qui génèrent toutes une alerte pour chaque transaction. La capacité de traitement chute alors à 5 000 paquets par seconde. Ce scénario est un cas très défavorable car la situation où chaque transaction génère une alerte pour chaque règle est très irréaliste.

Si l'on imagine un scénario avec de la sûreté de fonctionnement, les échanges typiques nécessitent une transaction toutes les 4 ms ; ce qui correspond à un débit de 500 paquets par seconde. Dans le cas du pire scénario, ce boîtier avec Suricata est dimensionné pour, au plus, une dizaine de couples d'équipements. Pour le cas d'un scénario plus réaliste avec une centaine de règles et peu d'alertes, ce boîtier est alors en capacité de surveiller plus d'une vingtaine de couples d'équipements. Ceci valide la faisabilité d'un tel système avec des équipements peu onéreux.

6 Conclusion

Dans cet article, nous avons pu explorer les possibilités offertes par la détection d'intrusion dans les systèmes industriels. Les caractéristiques de ces derniers font que cette mesure de sécurité peut donner d'excellents résultats alors que le caractère non-intrusif des IDS facilite leur déploiement et leur usage.

En guise de preuve de concept, un préprocesseur complet pour le protocole Modbus a été développé au sein de l'IDS Suricata. Son intégration prochaine dans le projet le rendra accessible au plus grand nombre pour une expérimentation voire un usage opérationnel. Les tests de performance menés montrent la faisabilité de sondes de détection avec du matériel bon marché.

Cependant, avant d'arriver à un produit pleinement opérationnel, de nombreux travaux restent à faire. Tout d'abord, Modbus ne représente qu'une frac-

tion des protocoles utilisés sur ce marché et il est indispensable d'effectuer le même travail pour d'autres tels que Profinet ou EtherNet/IP. Cela devra également être fait pour des protocoles propriétaires tels que S7 ou UMAS qui sont utilisés pour mettre à jour les *firmwares* et les programmes des automates.

Afin, d'avoir la supervision la plus précise possible, il est nécessaire d'avoir une cartographie très détaillée du système industriel supervisé. Pour faciliter cette tâche, il parait indispensable de développer des outils assistant les experts dès la phase de conception du système et tout au long de son existence. Si la cartographie est suffisamment précise, un tel outil peut également générer automatiquement les règles de détection des IDS.

Références

1. ANSSI : La cybersécurité des systèmes industriels : Cas d'étude (juin 2012)
2. ANSSI : La cybersécurité des systèmes industriels : Maitriser la SSI pour les systèmes industriels (juin 2012)
3. ANSSI : La cybersécurité des systèmes industriels : Mesures détaillées (janvier 2014)
4. ANSSI : La cybersécurité des systèmes industriels : Méthode de classification et mesures principales (janvier 2014)
5. Cheung, S. et al : Using model-based intrusion detection for SCADA network. In : Proc. SCADA Security Symposium. pp. 1–12 (2007)
6. Clusif : Enjeux de sécurité des infrastructures scada. `http://www.clusif.asso.fr/fr/production/ouvrages/pdf/CLUSIF-SCADA-Intro.pdf` (avril 2008)
7. Clusif : Cybercrime : évolution des cibles. `http://www.clusif.asso.fr/fr/production/ouvrages/pdf/CLUSIF-20120126-Cybercrime-Nouvelles-Cibles-UCL.pdf` (janvier 2012)
8. Debar, H., Marc, D., Andreas, W. : A revised taxonomy for intrusion detection systems. Annals of Telecommunications 55, 361–378 (2000)
9. Demongeot, T. : Détection d'intrusion pour les systèmes industriels. In : C&ESAR (2013)
10. Emergency Threats : Emergency threats SCADA ICS. `http://www.emergingthreats.net/2011/08/29/energysec-and-the-oisf-announce-new-scada-research/` (Consulté en mai 2014)
11. Modbus Foundation : MODBUS messaging on TCP/IP implementation guide v1.0b (octobre 2006)
12. Modbus Foundation : MODBUS over serial line specifiations and implementation guide v1.02 (décembre 2006)
13. Modbus Foundation : MODBUS application protocol specification v1.1b3 (avril 2012)
14. Morris, Thomas et al. : Deterministic intrusion detection rules for modbus protocols. In : 46th Hawaii International Conference on System Science. pp. 1773–1781 (2013)
15. OISF : Suricata, open source IDS. `www.suricata-ids.org` (Consulté en mai 2014)
16. Sourcefire : Snort, open source IDS. `www.snort.org` (Consulté en mai 2014)