

*Thèse professionnelle soumise afin d'obtenir le diplôme*

**Mastère spécialisé Cybersécurité des systèmes complexes pour  
l'Industrie et la Défense**

*Réalisée par*

**NOUAILHAC BAPTISTE**

---

Comment mettre en place un outil automatisé de génération  
de règles de détection d'intrusion ?

---

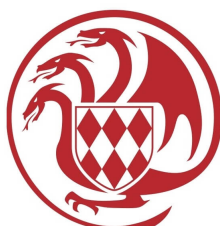
Soutenue le 18/10/2024 devant le Jury composé de :

CDT Alain MENELET  
CNE Sébastien REY  
Christian PEREZ  
Caroline FOSSATI  
Bruno VALENTIN

**Président du jury**  
**Relecteur**  
**Membre du jury**  
**Membre du jury**  
**Tuteur de stage**

*Projet de Fin d'Etudes fait à*

Agence Monégasque de Sécurité Numérique



# Remerciements

J'aimerais tout d'abord remercier l'Agence Monégasque de Sécurité Numérique et plus particulièrement Frédéric FAUTRIER, son directeur, pour avoir accepté de m'accueillir en tant que stagiaire au sein de l'agence.

Je suis très reconnaissant envers Bruno VALENTIN (mon tuteur de stage et responsable du CERT-MC) et Sébastien ABBONDANZA (responsable du SOC-MC) qui m'ont introduit et accompagné tout au long de mon stage. Ils m'ont fait confiance pour mener à bien les missions qu'ils m'ont attribuées, en me laissant une grande liberté d'action dans mon travail voire émettre des propositions.

Je remercie également toute l'équipe pédagogique du Mastère spécialisé CYBERSCID ainsi que les intervenants de la formation qui ont toujours été disponibles pour dispenser de l'aide et des conseils.

## **Résumé**

Ce travail s'inscrit dans le cadre du projet de fin d'études réalisé au sein de l'Agence Monégasque de Sécurité Numérique en vue de l'obtention du Mastère spécialisé Cybersécurité des systèmes complexes pour l'Industrie et la Défense (CYBERSCID).

Au cours de mon stage de six mois, j'ai pu travailler en étroite collaboration avec le SOC-MC et le CSIRT-MC, qui ont été une source indispensable d'information, un soutien dans mon travail quotidien et dans la réalisation de cette thèse.

J'ai pu acquérir une compréhension approfondie du fonctionnement d'une autorité nationale de sécurité numérique et les résultats de mon travail ont participé à l'efficacité opérationnelle de l'Agence.

# Table des matières

<b>Table des Figures</b>	<b>iii</b>
<b>Introduction</b>	<b>1</b>
<b>1 Contexte du projet</b>	<b>3</b>
1.1 Présentation de l'Agence Monégasque de Sécurité Numérique . . . . .	3
1.1.1 Le Pôle Expertise . . . . .	4
1.1.2 Le Centre de réponse et de traitement en matière d'attaques numériques (CERT-MC) . . . . .	5
1.2 Problématique . . . . .	6
1.3 Organisation du stage . . . . .	8
<b>2 Définition du sujet</b>	<b>9</b>
2.1 L'enjeu de la détection des menaces . . . . .	9
2.2 Les systèmes de détection d'intrusion . . . . .	10
2.2.1 Types d'IDS . . . . .	10
2.2.2 Méthode de détection . . . . .	14
2.3 Les indicateurs de compromission . . . . .	16
<b>3 Réalisation de la solution</b>	<b>18</b>
3.1 Génération de règles normées . . . . .	20
3.1.1 Récupération des règles . . . . .	24
3.1.2 Contrôle des règles en entrée . . . . .	25
3.1.3 Filtrage des règles . . . . .	28
3.1.4 Modification des règles . . . . .	31
3.1.5 Traitement des résultats . . . . .	33
3.2 Intégration de MISP . . . . .	37
3.2.1 Fonctionnement de MISP . . . . .	38
3.2.2 Exploitation de MISP . . . . .	41
3.3 Filtrage des règles générées par OIV . . . . .	51
<b>4 Résultat</b>	<b>54</b>
4.1 Analyse des résultats . . . . .	54
4.2 Bilan personnel . . . . .	56
<b>Conclusion et Perspectives</b>	<b>57</b>

<b>Glossaire</b>	<b>58</b>
<b>Annexe</b>	<b>60</b>
.1    configTestSuricata.yaml . . . . .	60
.2    Résultat Test Suricata . . . . .	62
.3    Fonctions MISP . . . . .	63
<b>Bibliographie</b>	<b>65</b>

# Table des Figures

1.1	Exemple de disposition de sonde IDS dans le réseau d'un OIV (source : <a href="http://eventus-networks.blogspot.com">eventus-networks.blogspot.com</a> ) . . . . .	6
2.1	Exemple de systèmes de détection d'intrusion réseau (NIDS) (source : <a href="http://techno-skills.com">techno-skills.com</a> )	11
2.2	Exemple de systèmes de détection d'intrusion au niveau de l'hôte (HIDS) (source : <a href="http://techno-skills.com">techno-skills.com</a> ) . . . . .	12
2.3	Procédure de détection d'attaques d'un IDS à base de signatures (source : <a href="http://techno-skills.com">techno-skills.com</a> ) . . . . .	14
2.4	Procédure de détection d'attaques d'un IDS à base d'anomalies (source : <a href="http://techno-skills.com">techno-skills.com</a> ) . . . . .	15
2.5	Exemple de liste d'IOC de hachages de fichiers malveillants fournie par <i>abuse.ch</i> (source : <a href="https://sslbl.abuse.ch/blacklist/sslblacklist.csv">sslbl.abuse.ch/blacklist/sslblacklist.csv</a> ) . . . . .	16
3.1	Processus d'alimentation des sondes . . . . .	18
3.2	Diagramme de flux du programme Python développé . . . . .	21
3.3	Exemple de la structure d'une règle Suricata (source : <a href="http://redmine.openinfosecfoundation.org">redmine.openinfosecfoundation.org</a> )	22
3.4	Exemple de fichier de règles (source : <a href="http://emerging-dns.rules">emerging-dns.rules</a> ) . . . . .	23
3.5	Diagramme de flux de la section responsable de la récupération des règles . . . . .	24
3.6	Récupération des règles . . . . .	24
3.7	Diagramme de flux de la section responsable du contrôle des règles . . . . .	25
3.8	Test des fichiers de règles . . . . .	26
3.9	Identification des règles ayant provoqué des erreurs . . . . .	26
3.10	Récupération des règles en excluant les erreurs . . . . .	26
3.11	Diagramme de flux de la section responsable du filtrage des règles . . . . .	28
3.12	Création de dataframe <i>pandas</i> . . . . .	29
3.13	Suppression des règles dupliquées . . . . .	29
3.14	Gestions des règles en collision de sid . . . . .	30
3.15	Création de dataframe pour chaque sonde . . . . .	30
3.16	Diagramme de flux de la section responsable de l'application des modifications . . . . .	31
3.17	Exemple de fichier CSV de configuration des règles à modifier . . . . .	31
3.18	Modifications des règles . . . . .	32
3.19	Diagramme de flux de la fin du programme . . . . .	33
3.20	Exemple de fichier de règles généré . . . . .	34
3.21	Exemple de fichier CSV d'erreurs dans les fichiers testés . . . . .	34
3.22	Exemple de fichier CSV de règles dupliquées non géré . . . . .	35
3.23	Exemple de fichier CSV d'erreur de modification de règle . . . . .	35
3.24	Exemple de fichier CSV de statistiques du programme . . . . .	36

3.25	Démonstration de l'échange d'informations entre entités via MISP (source : <a href="http://www.misp-project.org">www.misp-project.org</a> ) . . . . .	38
3.26	Exemple d'événement MISP (source : capture d'écran de l'interface graphique de l'instance de test locale MISP) . . . . .	40
3.27	Installation sécurisée de MISP dans le réseau de l'Agence (source : <a href="https://www.misp-project.org/img/blog/misp-guard-architecture.png">https://www.misp-project.org/img/blog/misp-guard-architecture.png</a> ) . . . . .	41
3.28	Récupération des IOC via l'API de notre instance MISP locale (source : <a href="https://so-sonajaa.medium.com/misp-malware-information-sharing-platform-ep1-eea91df7415b">https://so-sonajaa.medium.com/misp-malware-information-sharing-platform-ep1-eea91df7415b</a> ) . . . . .	41
3.29	Enrichissement de l'instance locale de MISP en évènement . . . . .	43
3.30	Création événement MISP . . . . .	44
3.31	Suppression des IOC inutiles . . . . .	45
3.32	Exemple de "Decaying Model" (source : capture d'écran de l'interface graphique de l'instance de test locale MISP) . . . . .	46
3.33	Génération de règle Suricata depuis l'instance local MISP . . . . .	47
3.34	Exemple de génération de règle Suricata par MISP (source : capture d'écran de l'interface graphique de l'instance de test locale MISP) . . . . .	48
3.35	Modification des règles générées par MISP . . . . .	49
3.36	Fichier CSV des statistique du programme après intégration des règles provenant de MISP . . . . .	51
3.37	Exemple de fichier CSV de contrôle des catégories de règles . . . . .	52
3.38	Contrôle des catégories sur les sondes . . . . .	52
3.39	Fichier CSV des statistiques du programme après sélection des catégories par sonde . . . . .	53

# Introduction

Par la dépendance croissante de nos sociétés à l'égard des technologies numériques, la sécurité des systèmes d'information est devenue une priorité incontournable. Ces dernières années ont vu les cyberattaques se multiplier autant en nombre qu'en sophistication, menaçant la confidentialité, l'intégrité et la disponibilité des données des individus, des entreprises comme des États.

Pour faire face à ces défis croissants, la stratégie de Défense qui s'est la plus démocratisée est celle de la prévention par la détection préalable des menaces. L'une des technologies les plus utilisées à cette fin est le système de détection d'intrusion (IDS), qui analyse les flux du réseau afin d'alerter les utilisateurs en cas d'activité malveillante présumée.

C'est pourquoi cette thèse s'intéresse au sujet des IDS, à leur fonctionnement et à l'optimisation de leur exploitation autour de la problématique : "Comment mettre en place un outil automatisé de génération de règles de détection d'intrusion ?".

L'objectif de cette thèse est de proposer une méthodologie pour concevoir et mettre en place un outil capable de générer automatiquement des règles de détection d'intrusion. Cette automatisation vise non seulement à accélérer la mise à jour des règles de détection, mais aussi à améliorer leur précision et leur efficacité pour faciliter le travail des analystes.

Ce travail s'est déroulé dans le cadre d'un stage de six mois au sein de l'Agence Monégasque de Sécurité Numérique (AMSN), l'organisme national de cybersécurité de la Principauté de Monaco (équivalent de l'Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI) française). L'AMSN travaille quotidiennement à la protection des infrastructures nationales et des Opérateurs d'Importance Vitale (OIV) grâce au SOC-MC qu'elle abrite, offrant ainsi un cadre idéal pour étudier le fonctionnement d'un IDS devant s'adapter à un grand nombre d'acteurs différents sur un territoire restreint comme celui de la Principauté.



La structure de cette thèse sera la suivante : dans un premier temps, je présenterai l'AMSN ainsi que la problématique de la thèse en détail puis je passerai en revue les concepts fondamentaux et les technologies existantes en matière de détection d'intrusion. Ensuite, je détaillerai la conception de l'outil automatisé, en décrivant les choix méthodologiques et les algorithmes utilisés. Enfin, j'exposerai les résultats des tests et des évaluations effectués, ainsi que les perspectives d'amélioration et les futures orientations de recherche.

# Chapitre 1

## Contexte du projet

### 1.1 Présentation de l'Agence Monégasque de Sécurité Numérique

L'Agence Monégasque de Sécurité Numérique (AMSN)<sup>1</sup>, créée par Ordonnance Souveraine le 23 décembre 2015<sup>2</sup>, est l'autorité nationale en charge de la sécurité des systèmes d'information. Fondée sur le modèle de l'Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI)<sup>3</sup> française, l'Agence est sous l'autorité directe du Ministre d'État (équivalent du Premier ministre français) et a pour rôle de constituer un centre d'expertise, de réponse et de traitement en matière de sécurité et d'attaques numériques pour l'Etat et les Opérateurs d'Importance Vitale (OIV) monégasques.

L'AMSN est aujourd'hui structurée autour de deux pôles, exposés ci-après.

---

1. Site officiel de l'Agence Monégasque de Sécurité Numérique : <https://amsn.gouv.mc/>

2. Ordonnance Souveraine n° 5.664 du 23 décembre 2015

3. Site officiel du CERT-FR : <https://www.cert.ssi.gouv.fr/>

### 1.1.1 Le Pôle Expertise

Le Pôle Expertise forme un groupe d'experts qui joue un rôle crucial dans la conception, la mise en œuvre et le suivi des stratégies de cybersécurité de l'État, ainsi que dans la sensibilisation et l'évaluation de la sécurité de l'infrastructure numérique nationale. Ses missions peuvent être résumées comme suit :

- Conseiller et coordonner les travaux interministériels sur la sécurité des systèmes d'information.
- Contrôler l'application des mesures de sécurité adoptées par le Gouvernement sur les systèmes d'information des administrations et des opérateurs publics ou privés.
- Sensibiliser les services publics et les opérateurs publics et privés aux exigences en matière de sécurité numérique.
- Évaluer et qualifier les acteurs et services de sécurité numérique de la Principauté.
- Mise en place et maintenance du service national de certification électronique pour les services de l'État, de la Commune, ainsi que les personnes physiques ou morales autorisées.

Dans le cadre de mon stage, il m'a parfois été donné l'occasion de côtoyer le Pôle Expertise. Ce qui m'a permis d'assister à certaines de leurs activités quotidiennes ainsi que de bénéficier de leur point de vue sur l'orientation de ma thèse.

### 1.1.2 Le Centre de réponse et de traitement en matière d'attaques numériques (CERT-MC)

Afin d'être à même de participer à la coopération internationale face aux menaces numériques, l'AMSN s'est dotée d'un CERT suivant les standards établis par le FIRST<sup>4</sup>, l'organisme qui coordonne l'action des différents CERTs et CSIRTs au niveau mondial.

Celui-ci réalise diverses missions réparties entre les trois divisions suivantes :

- **La division en charge de la supervision et de la détection des événements de sécurité numérique ou « Security Operations Center » (SOC-MC)**

Cette division est dédiée à la supervision et à la détection des événements de sécurité numérique. Sa mission principale est de protéger les systèmes d'information de la Principauté de Monaco contre les cybermenaces en temps réel, en assurant une vigilance constante et une réponse rapide aux événements de sécurité. Une fois identifiés et analysés, ces événements peuvent être déclarés comme incidents de sécurité lorsque leur dangerosité est avérée.

- **La division en charge de la réponse aux incidents de sécurité numérique ou « Computer Security Incident Response Team » (CSIRT-MC)**

Cette division a pour mission principale de répondre aux incidents de sécurité numérique (une fois ces derniers déclarés par le SOC-MC) en fournissant une assistance technique, des analyses approfondies et des solutions de remédiation aux parties prenantes (Étatiques et OIV publics ou privés) victimes.

- **La division en charge de l'analyse, du partage et de l'information ou « Information Sharing and Analysis Center » (ISAC-MC)**

Cette division réalise la collecte, l'analyse et le partage d'informations liées à la cybersécurité provenant de diverses sources. Elle agit comme un centre névralgique pour la coordination des efforts de sécurité numérique, favorisant la collaboration entre les différents acteurs nationaux et internationaux.

Dans le cadre de mon stage, j'ai travaillé de façon concomitante au CSIRT-MC et au SOC-MC, sous la supervision de Bruno VALENTIN (mon tuteur de stage et responsable du CERT-MC) et de Sébastien ABBONDANZA (responsable du SOC-MC), car l'objectif de ma mission était d'améliorer certains processus de travail communs aux deux divisions. La suite de ce document se concentrera sur le CERT-MC ; c'est auprès de ses équipes et avec son soutien que mon travail a été réalisé.

---

4. <https://www.first.org>

## 1.2 Problématique

Sur le territoire de la Principauté, l'AMSN doit épauler la sécurité numérique de plusieurs dizaines d'administrations publiques et d'entreprises classées OIV, représentant autant de réseaux à superviser avec des milliers d'utilisateurs et d'appareils connectés. L'Agence ne peut mener à bien cette mission qu'avec la coopération de ces acteurs, qui l'autorisent à placer des sondes en amont, et parfois à l'intérieur, de leurs réseaux. Toutes les actions des utilisateurs qui entrent et sortent des réseaux supervisés passent par les sondes qui disposent d'un IDS, en l'occurrence ici Suricata, lequel contrôle le contenu de ces flux réseau à l'aide de règles de détection qui, si elles sont déclenchées, génèrent des événements de sécurité gérés par le SOC-MC.

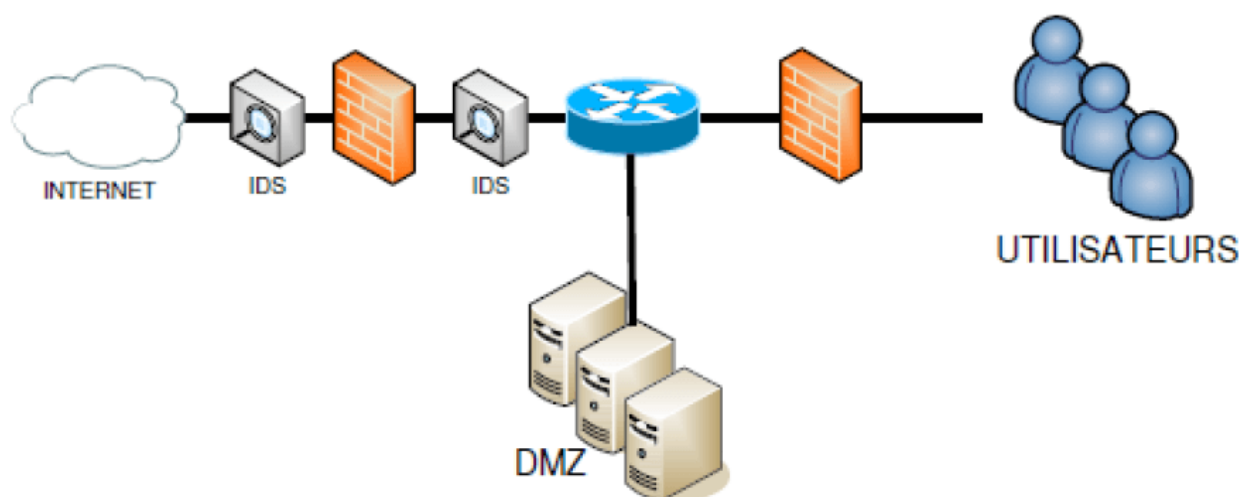


FIGURE 1.1 – Exemple de disposition de sonde IDS dans le réseau d'un OIV

Ces règles sont créées à partir d'indicateurs de compromission (IOC) établis en interne ou grâce à des renseignements externes provenant de CERT partenaires (comme le CERT-FR de l'ANSSI) ou d'agences privées de cyber-renseignement (également appelées CTI). Les règles générées par ces partenaires externes posent souvent des problèmes :

- Les règles générées sont fréquemment génériques et mal adaptées, engendrant trop d'événements inutiles ou redondants dans le réseau où elles sont mises en œuvre (ces réseaux sont tous basés sur des technologies différentes).
- Les renseignements extérieurs ne proviennent que d'une poignée d'acteurs, et l'agence ne peut être certaine de pouvoir détecter toutes les menaces existantes à l'état de l'art.

Ces difficultés de l'AMSN sont communes aux institutions publiques et aux entreprises qui doivent remplir les mêmes missions. Cela légitime la problématique posée : "Comment mettre en place un outil automatisé de génération de règles de détection d'intrusion ?"

Pour aborder cette problématique, il faut se poser les questions suivantes :

- Où recueillir le cyber-renseignement nécessaire à l'établissement des IOC ? Et comment évaluer la qualité de ces renseignements ?
- Comment générer des règles automatiquement à partir de ces IOC ?
- Comment s'assurer que ces règles soient aussi pertinentes que possible pour la partie prenante supervisée ?

Mes travaux, menés dans le cadre de l'AMSN, visent à répondre à cette problématique et à fournir une solution capable de résoudre les problèmes posés à l'Agence ou à tout autre acteur similaire oeuvrant pour la protection des systèmes d'information.

## 1.3 Organisation du stage

Le stage s'est déroulé du 1er avril au 30 septembre 2024. Pendant cette période, j'ai bénéficié d'un bureau au sein du SOC-MC avec un ordinateur personnel connecté à un réseau séparé sur lequel j'ai été libre de réaliser tous les développements nécessaires à la réalisation de mon projet. Cette configuration avait l'avantage de me permettre d'être proche des personnes impliquées dans l'outil sur lequel je travaillais, ainsi tout au long de mon stage j'ai pu disposer de l'expérience et des conseils, d'une part, des analystes SOC de l'AMSN et, d'autre part, de mon tuteur de stage Bruno VALENTIN, qui m'a partagé les connaissances et les avis du CSIRT.

Le stage s'est déroulé en trois parties distinctes :

1. Apprentissage des technologies et du contexte
2. Développement de la solution
3. Rédaction de la thèse

La première partie correspond à mon premier mois de stage, au cours duquel j'ai pris connaissance de mon environnement de travail et des missions de l'Agence, ainsi que des différents corps qui la composent. Au cours des semaines inaugurales de mon stage, j'ai dédié le plus clair de mon temps à suivre le travail quotidien de mes collègues pour appréhender les outils qu'ils utilisaient et comprendre comment ils fonctionnaient.

Après cette phase d'apprentissage, j'ai pu commencer à développer la solution relative à la problématique visée. Cette phase a duré quatre mois, à l'issue desquels j'ai soumis la solution et la documentation à mon tuteur. Mon tuteur s'est chargé quant à lui de la mise en production de mon travail pendant le mois qui a suivi.

Le dernier mois de stage et jusqu'à son terme, ayant finalisé la mission qui m'avait été confiée par mon tuteur, j'ai pu me concentrer avec son appui à parfaire, au sein de l'agence, la rédaction de la présente thèse, en l'étoffant par des échanges constructifs.

Tout au long de ces périodes, Bruno VALENTIN et Sébastien ABBONDANZA ont suivi mon travail en programmant des rapports bimensuels au cours desquels j'ai régulièrement fait état de mes progrès et reçu des conseils de leur part.

# Chapitre 2

## Définition du sujet

### 2.1 L'enjeu de la détection des menaces

Les cyberattaques représentent l'une des menaces les plus sérieuses pour les institutions publiques et les entreprises privées. Les attaquants, qu'il s'agisse de pirates informatiques ayant des objectifs malveillants ou d'agents étatiques, utilisent des méthodes de plus en plus sophistiquées pour s'infiltrer dans les systèmes, voler des données sensibles, perturber les opérations et causer des dommages financiers et/ou des atteintes à la réputation. [8]

Ces acteurs malveillants poursuivent l'amélioration constante de leurs capacités à des fins de gain financier, d'espionnage ou encore de déstabilisation. Cette amélioration s'illustre en particulier dans le ciblage des attaquants qui cherchent à obtenir des accès discrets et pérennes aux réseaux de leurs victimes. Les acteurs malveillants tentent de compromettre des équipements périphériques qui leur offrent un accès plus furtif et persistant. Ce ciblage périphérique se transpose également dans le type d'entités attaquées et confirme l'intérêt des attaquants pour les prestataires, les fournisseurs, les sous-traitants, les organismes de tutelle et l'écosystème plus large de leurs cibles finales.

Cette amélioration continue des stratégies et des compétences des attaquants met en évidence les limites de la sécurité des réseaux. La mise en place d'architectures sécurisées (pare-feu, antivirus, réseaux locaux virtuels, etc.) ne garantit pas qu'une intrusion informatique soit impossible. Face à des attaquants suffisamment motivés, une erreur humaine ou une vulnérabilité encore inconnue (dite Zero-Day) finira par permettre une intrusion sur le réseau défendu ou sur celui d'un partenaire dont le réseau dépend, quelle que soit la qualité de la sécurité mise en place.

Cette faiblesse intrinsèque de la sécurité informatique rend obligatoire la mise en place, au sein des infrastructures numériques sécurisées, d'outils capables de gérer la possibilité que l'infrastructure défendue soit déjà compromise. C'est le rôle que jouent les systèmes de détection d'intrusion en permettant la détection des tentatives d'intrusion au sein d'un réseau supervisé. [5]



## 2.2 Les systèmes de détection d'intrusion

La détection des intrusions consiste à surveiller les événements qui se produisent dans un système ou un réseau informatique et à les analyser pour y déceler des signes d'intrusion, définis comme des tentatives pour compromettre la confidentialité, l'intégrité, la disponibilité ou pour contourner les mécanismes de sécurité d'un ordinateur ou d'un réseau.

Les intrusions sont causées par des attaquants qui accèdent aux systèmes depuis l'internet, par des utilisateurs autorisés des systèmes qui tentent d'obtenir des privilèges supplémentaires pour lesquels ils ne sont pas autorisés, et par des utilisateurs autorisés qui abusent des privilèges qui leur sont accordés. Les systèmes de détection d'intrusion (IDS) sont des produits logiciels ou matériels qui automatisent ce processus de surveillance et d'analyse.

Un IDS peut être configuré pour surveiller différents types de trafic, tels que les paquets de données, les connexions réseau, les logs de systèmes, etc. Lorsqu'il détecte une activité suspecte, l'IDS génère une alerte qui est envoyée, permettant de prendre des mesures pour bloquer l'attaque ou corriger la vulnérabilité. [3]

### 2.2.1 Types d'IDS

La façon la plus courante de classer les IDS est de les regrouper par source d'information. Certains IDS analysent les paquets du réseau, capturés à partir des flux passants au sein du réseau pour trouver les attaquants. D'autres IDS analysent les sources d'information générées par les systèmes d'exploitation ou les logiciels d'application présents sur les postes utilisateurs pour y trouver des signes d'intrusion. [6 - 7]

#### Les systèmes de détection d'intrusion réseau

La majorité des systèmes commerciaux de détection d'intrusion sont basés sur les réseaux. Ces systèmes de détection d'intrusion réseau (ou NIDS : Network Intrusion Detection System) détectent les attaques en capturant et en analysant les paquets du réseau. À l'écoute d'un segment de réseau ou d'un commutateur, un système de détection d'intrusion en réseau peut surveiller le trafic réseau affectant plusieurs hôtes connectés au segment de réseau.

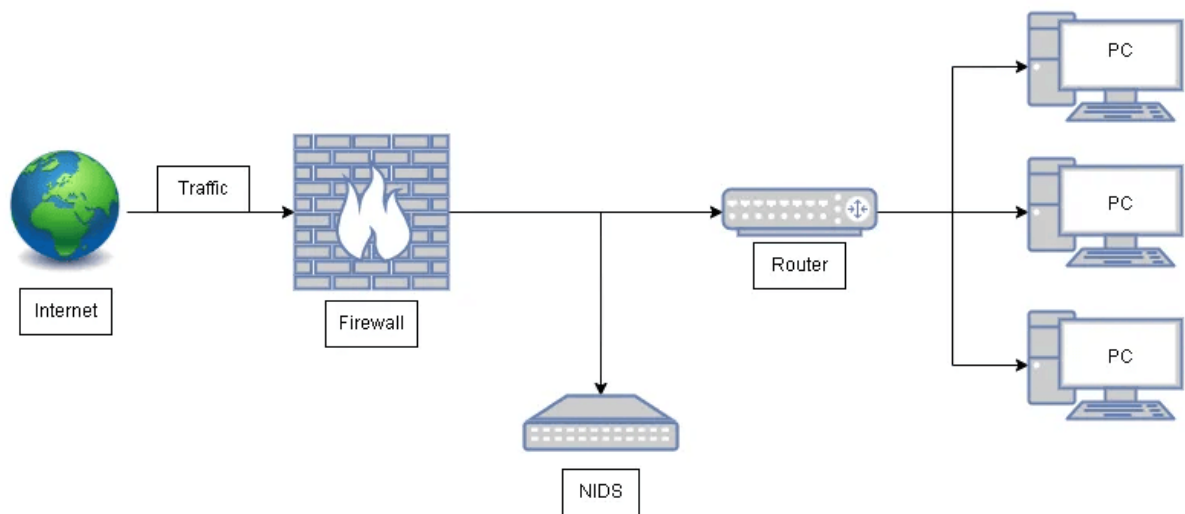


FIGURE 2.1 – Exemple de systèmes de détection d'intrusion réseau (NIDS)

#### *Avantages des IDS en réseau :*

- Quelques IDS bien placés peuvent surveiller un grand réseau ;
- Le déploiement d'IDS en réseau a peu d'impact sur un réseau existant. Les IDS basés sur le réseau sont généralement des dispositifs passifs qui écoutent sur un fil de réseaux sans interférer avec le fonctionnement normal d'un réseau ;
- Les IDS en réseau peuvent être configurés de manière à être hautement sécurisés contre les attaques et très difficiles à détecter pour les attaquants.

#### *Inconvénients des IDS en réseau :*

- Les IDS basés sur le réseau peuvent avoir des difficultés à traiter tous les paquets dans un réseau important ou très fréquenté et, par conséquent, ne pas reconnaître une attaque lancée pendant les périodes de fort trafic ;
- Les IDS basés sur le réseau ne peuvent pas analyser les informations chiffrées. Ce problème s'aggrave car de plus en plus d'organisations (et d'attaquants) utilisent des réseaux privés virtuels ;

- La plupart des IDS basés sur le réseau ne peuvent pas déterminer si une attaque a réussi ou non ; ils peuvent seulement discerner qu'une attaque a été initiée. Cela signifie qu'après la détection d'une attaque par un système IDS basé sur le réseau, un examen manuel (ou une corrélation automatique à l'aide d'autres sources de données comme des logs) devra être effectué afin de caractériser si l'attaque a été réussie.

### Systèmes de détection d'intrusion au niveau de l'hôte

Les systèmes de détection d'intrusion au niveau de l'hôte (ou HIDS : Host-based Intrusion Detection System), reposant sur l'hôte, fonctionnent sur la base des informations collectées à l'intérieur des appareils des utilisateurs. Les IDS basés sur l'hôte utilisent normalement des sources d'information de deux types : les pistes d'audit du système d'exploitation et les journaux du système. Ils permettent également de prendre un instantané des fichiers système existants et de les comparer à l'instantané précédent pour générer des alertes en cas de modifications.

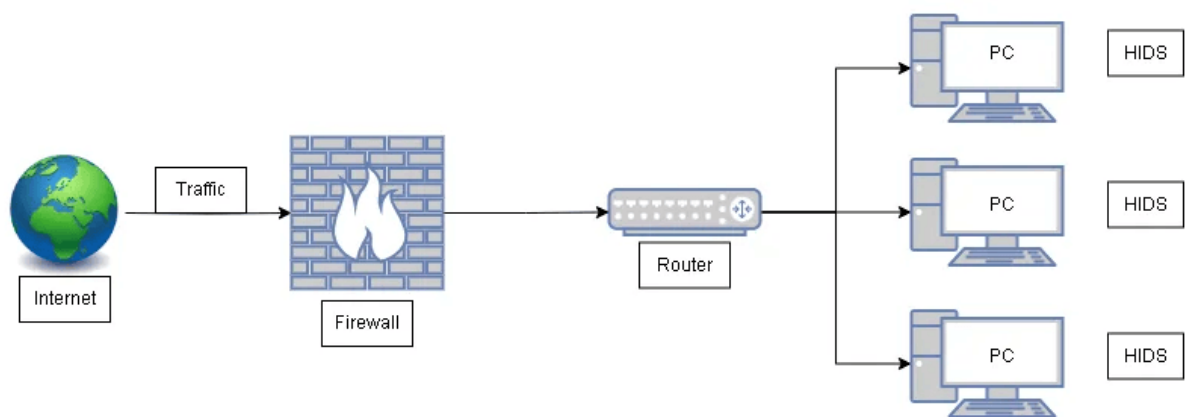


FIGURE 2.2 – Exemple de systèmes de détection d'intrusion au niveau de l'hôte (HIDS)

#### *Avantages des IDS basés sur les hôtes :*

- Les IDS basés sur l'hôte, grâce à leur capacité à surveiller les événements locaux d'un hôte, peuvent détecter des attaques qui ne peuvent l'être par un IDS basé sur le réseau ;

- Les IDS basés sur l'hôte peuvent fonctionner dans un environnement où le trafic réseau est chiffré, les sources d'information étant générées avant le chiffrement des données et/ou après le déchiffrement des données sur l'hôte de destination ;
- Les IDS basés sur l'hôte ne sont pas affectés par les réseaux commutés.

*Inconvénients des IDS basés sur les hôtes :*

- Les IDS basés sur l'hôte sont plus difficiles à gérer, car les informations doivent être configurées et gérées pour chaque hôte surveillé ;
- Étant donné que les sources d'information des IDS basés sur l'hôte résident sur l'hôte ciblé par les attaques, l'IDS peut être attaqué et désactivé dans le cadre de l'attaque ;
- Les IDS basés sur l'hôte ne sont pas bien adaptés à la détection des scans de réseau ou d'autres formes de surveillance ciblant l'ensemble d'un réseau, car l'IDS ne voit que les paquets de réseau reçus par son hôte.

## 2.2.2 Méthode de détection

Les technologies IDS utilisent de nombreuses méthodologies pour détecter les incidents. Ces méthodologies peuvent être regroupées en deux grandes catégories : celles basées sur les signatures et celles basées sur les anomalies. La plupart des technologies IDS utilisent plusieurs méthodologies de détection, séparément ou intégrées, afin de fournir une détection plus large et plus précise. [1]

### Les IDS à base de signatures

Les IDS à base de signatures ont une base de données comportant un ensemble de signatures d'attaques (base de signatures). Le principe de fonctionnement est le test de correspondance. Les données du réseau sont analysées et comparées aux signatures d'attaques connues stockées dans la base de signatures. En cas de correspondance, une alerte est émise. Un avantage de ce système est qu'il a une meilleure précision lorsque les règles de signature sont correctes. Cette base de signatures est en général pré-initialisée avec des données de l'éditeur de l'IDS et mise à jour régulièrement pour prendre en compte les nouvelles attaques.

La détection basée sur les signatures est très efficace pour détecter les menaces connues, mais largement inefficace pour détecter les menaces inconnues. Elle a l'avantage d'être facile à mettre en place, mais sa principale limite est que si l'attaquant est conscient des règles de détection en place, il peut modifier légèrement sa technique pour qu'elle ne soit plus détectable.

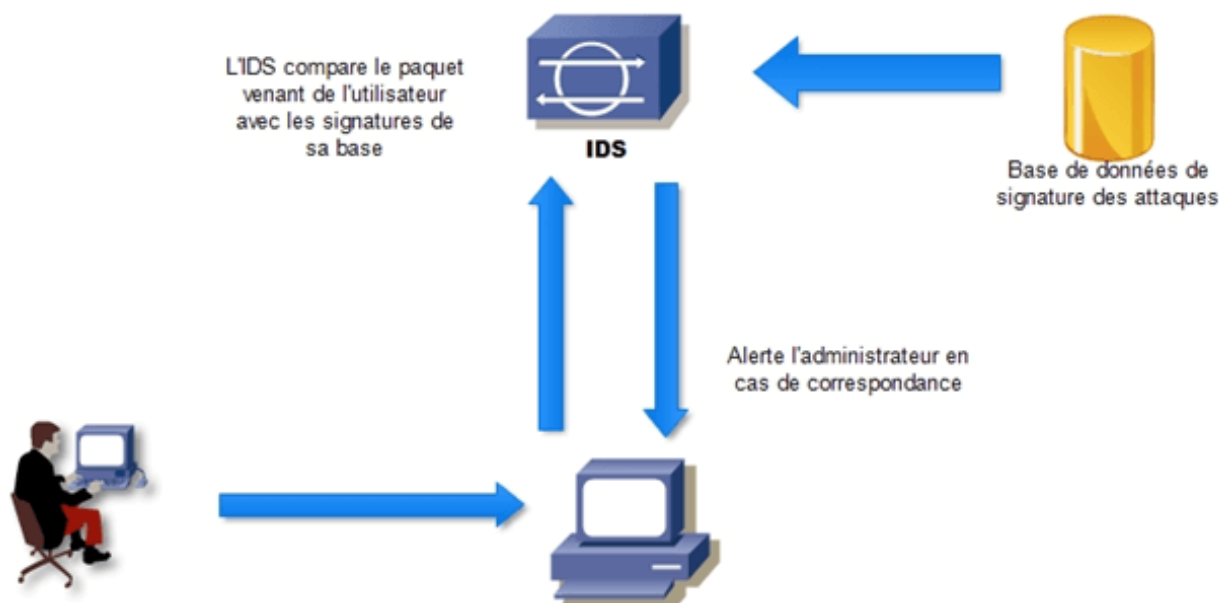


FIGURE 2.3 – Procédure de détection d'attaques d'un IDS à base de signature

## Les IDS à base d'anomalies

Cette catégorie d'IDS utilise un modèle statistique du fonctionnement de référence du réseau qui peut comprendre la bande passante utilisée, les protocoles définis pour le trafic, les ports et les périphériques qui font partie du réseau. Il surveille régulièrement le trafic réseau et le compare au modèle statistique. En cas d'anomalie ou de divergence, l'administrateur est alerté. Un avantage de ce système est qu'il peut détecter des attaques nouvelles dont le comportement s'éloigne suffisamment de la normale.

Malheureusement, les détecteurs d'anomalies et les systèmes de détection d'intrusion qui en découlent produisent souvent un grand nombre de fausses alertes, car les modèles normaux de comportement des utilisateurs et des systèmes peuvent varier considérablement sans être prévisibles. Ce type de détection est difficile à utiliser de manière efficace.

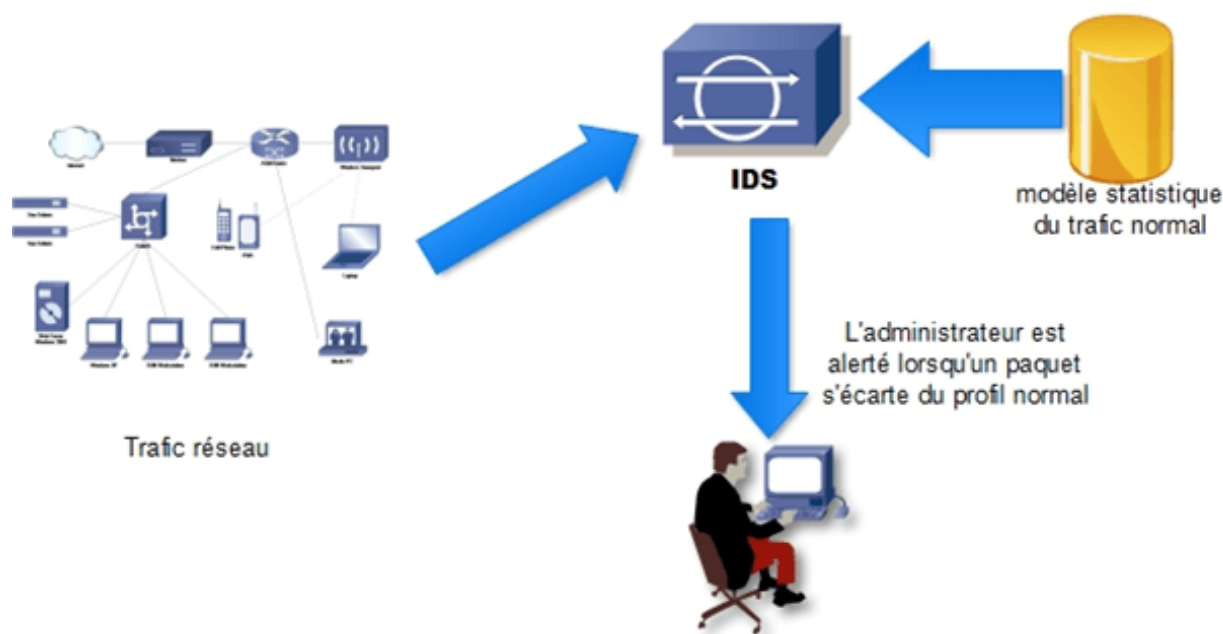


FIGURE 2.4 – Procédure de détection d'attaques d'un IDS à base d'anomalies

### *Note*

Dans le cadre de mes travaux, j'ai travaillé avec des systèmes de détection d'intrusion réseaux faisant de la détection par signatures. Ce choix technologique se justifie dans le contexte de l'AMSN par le besoin de posséder une capacité de détection pouvant fonctionner sur les réseaux de tous les OIV en même temps sans perturber leurs activités usuelles. L'utilisation de la détection à base d'anomalies et des Systèmes de détection d'intrusion au niveau de l'hôte serait beaucoup plus difficile par la diversité des OIV et demanderait un niveau de connaissance sur les réseaux supervisés que l'Agence n'a pas vocation à connaître.

## 2.3 Les indicateurs de compromission

Les IDS, notamment ceux faisant de la détection par signatures, ont besoin d'une base de référentiels pour savoir quoi détecter. C'est ce rôle que jouent les indicateurs de compromission (ou IOC : Indicator Of Compromise)

Un indicateur de compromission, en sécurité informatique, est une déviance ou un artefact observé sur un réseau ou dans un système d'exploitation qui indique, avec un niveau de certitude déterminé, une intrusion informatique. Des indicateurs de compromission peuvent être : des signatures virales, des adresses IP particulières, des hachages de fichiers malveillants, des URLs ou des noms de domaine de serveurs de commande et de contrôle de botnets, etc. [4]

#####		
# abuse.ch SSLBL SSL Certificate Blacklist (SHA1 Fingerprints) #		
# Last updated: 2024-07-31 07:28:20 UTC		
#		
# Terms Of Use: <a href="https://ssllbl.abuse.ch/blacklist/">https://ssllbl.abuse.ch/blacklist/</a>		
# For questions please contact ssllbl [at] abuse.ch		
#####		
#	SHA1	Listingreason
2024-07-31 07:28:20	733afe38870f2f38f5347cabcf6df68eb77b8937	DCRat C&C
2024-07-30 15:22:33	1da50d2a6e5d411f5a57b0c97839506fa2f3d763	AsyncRAT C&C
2024-07-30 15:22:26	c9ae758fef9766146db0c0dc54f11ddb223daacf	AsyncRAT C&C
2024-07-30 15:22:05	ae2c1d70ba8bece5de991266d55ae81b09015f49	DCRat C&C
2024-07-29 13:09:08	5dc1f7199d988a4d08ac662ee8222373b8874823	AsyncRAT C&C
2024-07-29 13:09:06	c2b5eee7fb4c99ba03371e83ee3f3a27998ebe23	Vidar C&C
2024-07-29 13:00:35	fcf5db845d993b620e12b5becf31305cb1fb6cd6	AsyncRAT C&C
2024-07-29 13:00:24	665c8b508ec328b12f8f1a2a20662bf0dba9f069	QuasarRAT C&C
2024-07-29 10:51:11	2178ba5544742fb6df5eca2bc85e52be4d90f065	PureLogStealer C&C
2024-07-28 13:22:27	192f20495a153c9448b7da3ca76ff008908468a2	CobaltStrike C&C
2024-07-26 05:55:04	606353d44d707da46679b462a2f5b168dc038fa1	Rhadamanthys C&C
2024-07-25 07:10:17	8f94b8f979a07b51199754584781c52cd0812905	Vidar C&C
2024-07-25 07:09:56	ca0a1f12325ff7ac080bcc1a739866245d723090	QuasarRAT C&C
2024-07-23 16:24:10	604345c0edd52fce1c546a4a79a4947a41361916	AsyncRAT C&C
2024-07-23 16:24:08	3992bdf1b82e697c8ea2b8542c03c64f678e936d	VenomRAT C&C
2024-07-19 07:47:19	854c68ab7ad70b6f95f76ec64b8a46c145997629	AsyncRAT C&C
2024-07-16 06:03:34	a9990103ef2745e7deac94dbfe252f14cb4d63e9	AsyncRAT C&C
2024-07-16 06:01:50	0ede2a9982ef00c0549718f1ec7e149171bb9035	AsyncRAT C&C
2024-07-14 08:10:05	e357ec145e21fb4a28a71afa35b05d4bd8056e71	PureLogStealer C&C
2024-07-14 07:44:10	84260b67bb540e01384d2254ad8a7dcfefc81013	DCRat C&C
2024-07-12 09:05:39	6f5596f3c304a5877e16764352142ca646c5927a	QuasarRAT C&C
2024-07-12 07:46:58	3ecd22a76155f63c9a83206a22a01279fe386ea5	Havoc C&C
2024-07-12 07:46:50	bf0e7fbc28eca379529d30950f2e65905c8167f6	AsyncRAT C&C
2024-07-11 07:15:27	1bd1fee41dac6fda021becc6ed67c26e7e7315ed	Sliver C&C
2024-07-11 07:08:09	0cb3da710c6dc833050a82f4d3864c05ddcb76c8	Vidar C&C
2024-07-11 06:57:29	772c8f46f11ba3c7612b0fddc01ef4348f21483d	AsyncRAT C&C
2024-07-10 05:49:35	7332640210cc4cf0157a7d6843e9793f7d5e75bb	Vidar C&C
2024-07-10 05:49:26	6ca86b5f28ce1182eacbc9e19231dc39fc4f920a	Vidar C&C
2024-07-09 09:23:38	051b8e4f4a3e279ba2179dbf148bf6d16982b8ce	AsyncRAT C&C
2024-07-09 09:23:28	a0cf3ddb0fa7f0bdc42c025c84d9dd0e2331f832	AsyncRAT C&C
2024-07-08 10:44:17	f48552e08a2e56b11738285f406eabacdee550f0	Latrodectus C&C
2024-07-07 09:06:14	9e0626623e416cfe23dd33d211b3525c02288492	AsyncRAT C&C
2024-07-07 09:05:36	549263bd72dd751a0e0dbf721dc74ba5195e3493	DCRat C&C
2024-07-05 05:12:39	5293bdb3680e323126b7d71396e15395af39d9a3	Vidar C&C
2024-07-05 05:12:11	0fc77c988b47ec2927fb9a9cd12e28b97fa13e51	AsyncRAT C&C
2024-07-05 05:12:05	4d9ddde9eef45886532adc566ea5e8b99b9c2546	AsyncRAT C&C
2024-07-04 06:19:47	3ad3f41c7332657b42e9995dd368b7c94ce24b10	Vidar C&C

FIGURE 2.5 – Exemple de liste d'IOC de hachages de fichiers malveillants fournie par *abuse.ch*

Ces IOC sont découverts à la suite d'investigations numériques (également appelées renseignements sur les menaces ou *threat intelligence*) menées sur les traces laissées par des acteurs malveillants en ligne ou à partir d'une analyse post-incident des techniques utilisées par les attaquants qui ont tenté ou réussi à pénétrer dans le système informatique. Ces IOC sont ensuite partagés par différents acteurs :

- **Organismes de cybersécurité**

La plupart des CERT étatiques (CERT-FR Français, US-CERT Américains, etc.) ou privés (CERT Orange, CERT Crédit Agricole, etc.) réalisent en partie eux-mêmes de la collecte d'IOC qu'ils partagent en privé entre agences lorsque des accords de collaboration existent, mais aussi publiquement pour une part <sup>1</sup>.

- **Entreprises spécialisées de CTI**

Le besoin de renseignement sur les menaces existantes ne cessant de croître, des sociétés spécialisées se sont créées pour répondre à cette demande (CrowdStrike, Sekoia.io, etc.) en plus des autres services de cybersécurité qu'elles peuvent fournir. En échange d'un contrat rémunéré, ces sociétés fournissent des listes d'IOC régulièrement mises à jour.

- **Plate-formes d'échange libre de renseignement cyber**

Pour faire face à l'ampleur des menaces, des plateformes de coopération CTI ont commencé à apparaître en ligne, impliquant des organisations étatiques et privées ainsi que des professionnels individuels du secteur. Grâce à ces plateformes (MISP, alienvault, threatfox, etc.), les utilisateurs peuvent s'alerter mutuellement des événements cyber en cours et partager les IOC qu'ils possèdent.

---

1. Exemple source publique du CERT-FR : <https://www.cert.ssi.gouv.fr/ioc/>



## Chapitre 3

### Réalisation de la solution

Comme énoncé dans la section **Problématique**, l'AMSN dispose déjà de sondes installées à des points clés des réseaux des OIV. Il s'agit de sondes qualifiées par l'ANSSI et délivrées par une société spécialisée qui fournit également un logiciel permettant d'envoyer directement des règles Suricata aux sondes. Les règles de détection des sondes sont quant à elles mises à jour quotidiennement afin de rester opérationnelles face à l'état de l'art connu de la menace.

Le processus de gestion des sondes préexistant au sein de l'Agence, lorsque j'ai commencé à travailler, était le suivant :

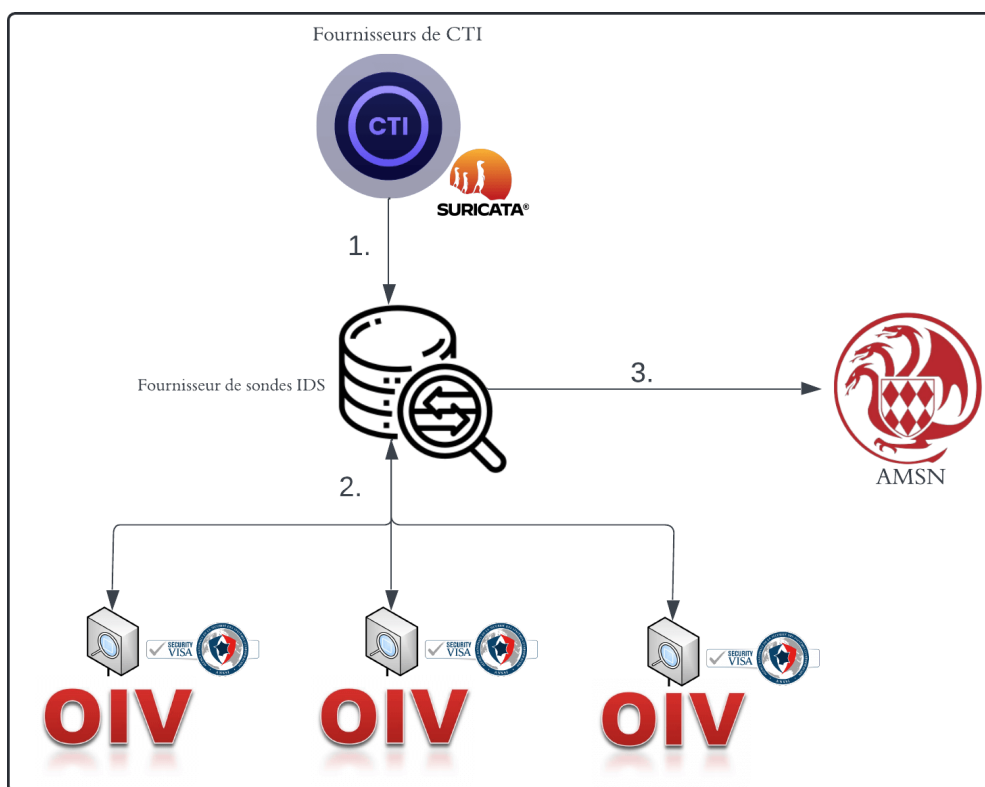


FIGURE 3.1 – Processus d'alimentation des sondes

1. L'agence recoit, de la part de partenaires, de la CTI sous la forme de règles Suricata exploitables ;
2. Les règles sont envoyées au système de gestion des sondes fourni par l'entreprise partenaire spécialisée, qui les distribue aux sondes des OIV ;
3. Les alertes générées par les sondes sont remontées jusqu'au SOC-MC de l'AMSN.

Mon projet professionnel n'a pas profondément modifié ce processus, mais l'a amélioré en s'interposant entre les fournisseurs de CTI et le système de gestion des sondes afin d'améliorer la qualité des règles de détection arrivant finalement dans les sondes IDS.

Ce travail d'amélioration s'est organisé en trois grandes étapes :

1. Création d'un programme automatisé pour contrôler la qualité des règles fournies par les partenaires afin de les filtrer et de les formater avant de les envoyer aux sondes des OIV ;
2. Mise en place et exploitation d'une instance locale de MISP pour enrichir le programme avec de nouvelles sources de renseignement ;
3. Exploitation des travaux précédents pour adapter automatiquement les règles envoyées en fonction de l'OIV cible.

J'ai réalisé ces différentes missions en suivant la même méthodologie. Tout d'abord, une phase de réflexion était menée conjointement avec mon tuteur de stage et le responsable du SOC-MC pour identifier les attendus du travail à produire. Puis, une phase de développement et de test s'ensuivait pour correspondre à ces attentes. Pour conclure par une phase de rédaction de documentation et de mise en place de tests unitaires afin de permettre la continuité de mon travail sur le temps et sa potentielle prise en main par d'autres personnes.

## 3.1 Génération de règles normées

Le premier mois de mon stage, passé en immersion au sein de l'Agence et aux côtés des membres du SOC-MC, m'a permis d'observer et de comprendre le fonctionnement et l'enjeu des données ainsi que des systèmes sur lesquels j'allais travailler. A la suite des recommandations des analystes du SOC-MC et autres entretiens avec Bruno VALENTIN et Sébastien ABBONDANZA, un premier cahier des charges des besoins a été établi. Cette documentation a servi de base au développement d'un outil qui contribuera à améliorer le processus actuel de mise à jour des règles en répondant aux besoins définis ci-après :

- L'outil doit être capable de récupérer et de centraliser toutes les règles, quelle que soit leur source (partenaires de CTI ou internes) ;
- L'outil doit vérifier la conformité des règles récupérées avec les sondes utilisées et supprimer toute règle dupliquée ;
- L'outil devra créer des fichiers de règles portant un nom unique pour chaque sonde ;
- L'outil permettra aux analystes du SOC-MC de spécifier des règles à modifier ou à supprimer avant de les envoyer aux sondes ;
- L'outil devra rendre compte des résultats du programme.

L'objectif de cet outil est de répondre à une partie des problèmes énoncés dans ma problématique et à ceux de l'Agence. Le choix technologique de développement de l'outil s'est porté sur le langage de programmation python. L'outil aurait pu être développé à l'aide d'autres technologies, mais Python a été choisi pour la facilité d'utilisation qu'il offre dans la production de scripts facilement exportables. De plus, ce langage me fut suggéré car il est également maîtrisé par la majorité de mes collègues au sein de l'AMSN, ce qui permet à mon travail d'être facilement relu et repris par d'autres personnes au sein de l'Agence.

Le diagramme de flux de l'outil développé, qui sera analysé en détail dans la suite de mon argumentaire, est le suivant :

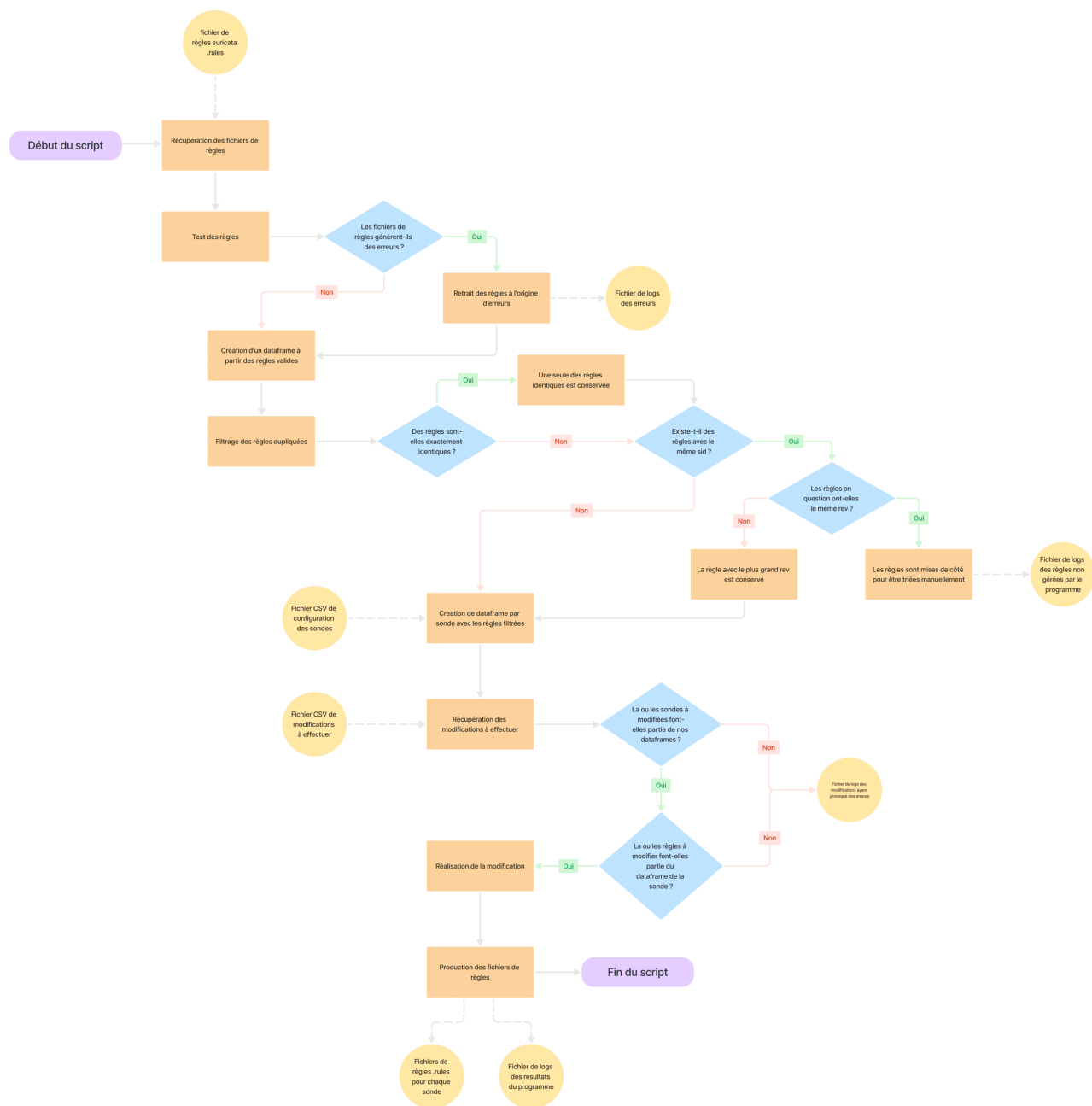


FIGURE 3.2 – Diagramme de flux du programme Python développé

### 3.1.0 Suricata

Le programme est conçu pour récupérer les règles de Suricata uniquement. Suricata<sup>1</sup> est un système de détection d'intrusion (IDS), de prévention d'intrusion (IPS) et de surveillance de la sécurité du réseau (NSM). Créé et maintenu par l'OISF<sup>2</sup>, il est utilisé au sein de l'AMSN pour ses fonctions IDS au service des objectifs de détection de l'Agence. Le choix de l'Agence s'est porté sur Suricata car il s'agit du seul outil disponible sur les sondes qualifiées par l'ANSSI.

Suricata fonctionne comme un **IDS à base de signatures** c'est à dire grâce à des signatures pré-configurées, des règles servant à détecter des actions malveillantes réalisées sur un réseau. Les règles Suricata se structurent ainsi en trois parties :

- Une partie *action* qui stipule ce que la règle fera en cas de détection (dans ce cas, où l'Agence ne fait que de la supervision, seule l'action "alerte" est utilisée pour créer des signalements pour le SOC-MC) ;
- Un entête dans lequel sont définis le protocole et le réseau sur lesquels la détection aura lieu ;
- Une section *option*, qui permet de définir des paramètres variables sur la règle.

```
drop tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"ET
TROJAN Likely Bot
Nick in IRC (USA +..)"; flow:established,to_server;
flowbits:isset,is_proto_irc; content:"NICK "; pcre:"/NICK
.*USA.*[0-9]{3,}/i"; classtype:trojan-activity;
reference:url,doc.emergingthreats.net/2008124;
reference:url,www.emergingthreats.net/cgi-
bin/cvsweb.cgi/sigs/VIRUS/TROJAN_IRC_Bots;
sid:2008124; rev:2;)
```



FIGURE 3.3 – Exemple de la structure d'une règle Suricata

1. Site officiel de Suricata : <https://suricata.io>

2. Open Information Security Foundation : <https://oisf.net/>

Ces règles sont stockées dans des fichiers de règles *.rules* (un format de fichiers texte utilisé par Suricata), lesquels sont directement fournis par des partenaires de renseignement cyber.

```
1 # This Ruleset is EmergingThreats Open optimized for suricata-7.0.3-enhanced.
2
3 alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"GPL DNS zone transfer UDP";
  content:"|00 00 FC|"; offset:14; reference:cve,1999-0532; reference:nessus
  ,10595; classtype:attempted-recon; sid:2101948; rev:8; metadata:created_at
  2010_09_23, cve CVE_1999_0532, updated_at 2019_07_26;)
4
5 alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"GPL DNS named version attempt";
  content:"|07|version"; offset:12; nocase; content:"|04|bind|00|"; offset:12;
  nocase; reference:nessus,10028; classtype:attempted-recon; sid:2101616; rev:9;
  metadata:created_at 2010_09_23, updated_at 2019_07_26;)
6
7 alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"GPL DNS named iquery attempt";
  content:"|09 80 00 00 00 01 00 00 00 00|"; depth:16; offset:2; reference:
  bugtraq,134; reference:cve,1999-0009; reference:url,www.rfc-editor.org/rfc/
  rfc1035.txt; classtype:attempted-recon; sid:2100252; rev:9; metadata:
  created_at 2010_09_23, cve CVE_1999_0009, updated_at 2019_07_26;)
8
9 alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"GPL DNS named authors attempt";
  content:"|07|authors"; offset:12; nocase; content:"|04|bind|00|"; offset:12;
  nocase; reference:nessus,10728; classtype:attempted-recon; sid:2100256; rev:8;
  metadata:created_at 2010_09_23, updated_at 2019_07_26;)
10
11 ...
```

FIGURE 3.4 – Exemple de fichier de règles

### 3.1.1 Récupération des règles

Ces fichiers de règles précédemment cités, sont mis à disposition du programme pour permettre leur centralisation. L'enjeu de rassembler les règles en un seul endroit est de s'assurer que l'outil soit capable de modifier les règles et de les envoyer aux sondes consécutivement.

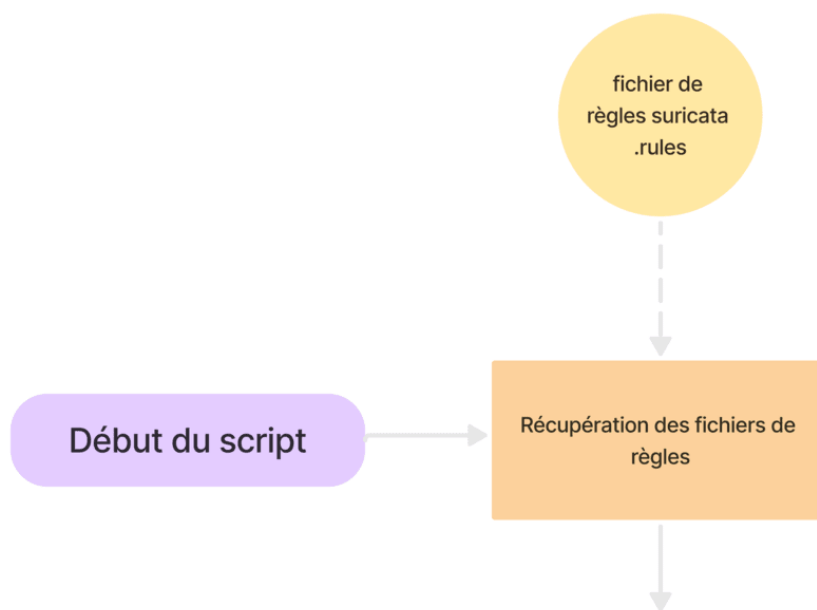


FIGURE 3.5 – Diagramme de flux de la section responsable de la récupération des règles

Le script a été conçu pour récupérer en entrée un dossier positionné à la racine du programme avec les fichiers de règles qui seront lus par le programme.

```
1 # Parcours chaque fichier dans le repertoire fourni
2 for filepath in os.listdir("rules"):
3     suricata_alerts = extract_suricata_alerts(filepath)
```

*Traite chaque fichier de règles individuellement depuis le dossier "rules"*

FIGURE 3.6 – Récupération des règles

Chaque fichier présent dans le dossier va être testé puis ensuite les règles seront récupérées et rassemblées.

### 3.1.2 Contrôle des règles en entrée

Avant la récupération des règles, le programme doit être capable de tester les règles récupérées car selon les retours des membres du CERT-MC, il arrivait que des erreurs (faute de frappe, mauvaise valeur inscrite...) soient contenues dans les règles de détection des fichiers récupérés ou que lesdites règles soient destinées à une version de Suricata qui n'est pas compatible avec celles des sondes. Ce qui pouvait provoquer le blocage d'une sonde si les règles en question arrivaient en bout de chaîne.

Pour cette raison il était attendu de mon programme qu'il soit capable de contrôler la validité des règles avec les sondes pour prévenir, en cas de panne, le risque de perte en capacité de détection.

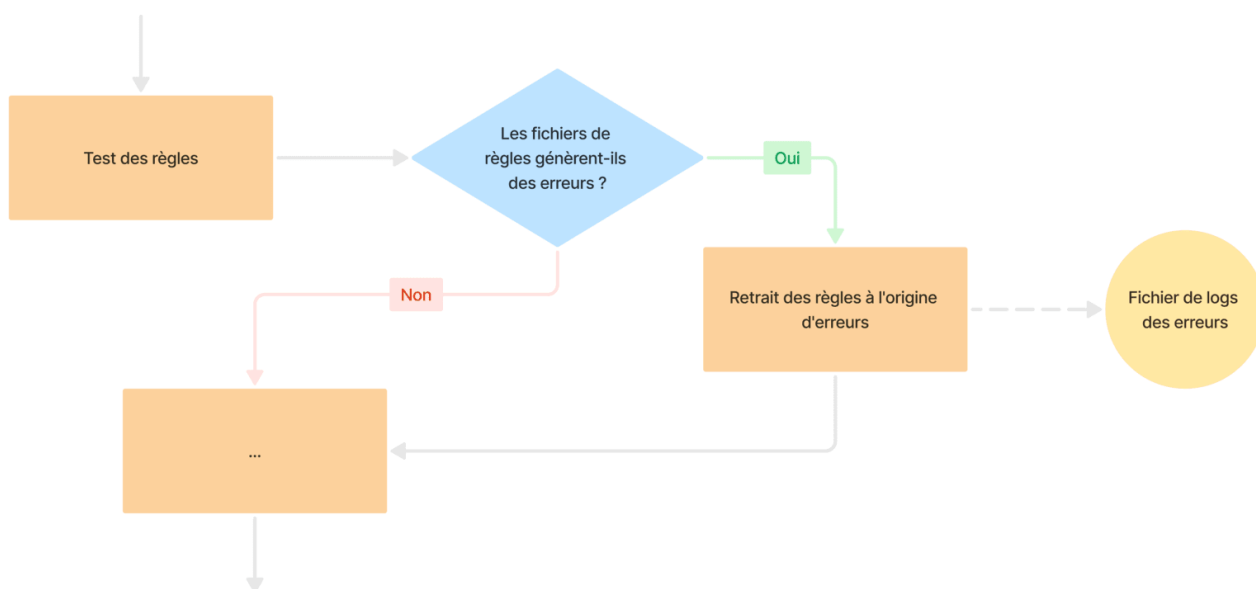


FIGURE 3.7 – Diagramme de flux de la section responsable du contrôle des règles

Le programme va ainsi tester individuellement chaque fichier récupéré. Pour cela Suricata dispose d'un mode test qui permet de compiler un fichier de règles. A cette fin, une version de Suricata comparable à celle présentée sur les sondes a été installée localement sur mon poste de travail.



```

1 testresult = subprocess.run(['suricata', '-T', '-c', 'configTestSuricata.yaml', '-l',
2                               '/tmp/', '-S', filepath], capture_output=True, text=True)

```

*Teste le fichier présent dans la variable "filepath" en utilisant la version locale de Suricata et le fichier de configuration 'configTestSuricata.yaml' [A.1]*

FIGURE 3.8 – Test des fichiers de règles

Pour chaque fichier testé, la version locale de Suricata génère des journaux d'erreurs qui désignent chaque règle ayant provoqué des erreurs avec une description de celle-ci [A.2]. A partir de ces journaux des résultats des tests, j'identifie les potentielles règles défectueuses.

```

1 # Si le fichier produit une erreur
2 if (testresult.returncode == 1):
3     # 'stderr' = toute les erreurs qui ont encourue
4     for line in testresult.stderr.split('\n'):
5         if line.startswith('E: detect: error parsing signature'):
6             # Extraction de la ligne des messages d'erreur
7             linenumber = line.split(' ')[-1]
8             line_to_remove.append(int(linenumber))
9             # Recuperation des erreurs pour creation csv
10            csvError["collumnRules"].append(line)
11        else:
12            csvError["collumnError"].append(line)

```

*Parcours de chaque ligne des journaux d'erreurs pour dénombrer les lignes non fonctionnelles. "Linenumber" est la variable qui stocke les numéros des lignes des règles qui provoquent des erreurs.*

FIGURE 3.9 – Identification des règles ayant provoqué des erreurs

Celles-ci seront mises de côté pour la prochaine mise à jour des sondes pour n'utiliser que les valides pour le reste du programme.

```

1 # Boucle dans les fichiers de regles
2 with open(filepath, 'r') as file:
3     for i, line in enumerate(file, start=1):
4         # On ne garde que les lignes qui ne font pas partie des erreurs
5         if i not in line_to_remove and line.startswith("alert "):
6             suricata_alerts.append({'rules': line, 'source': sourceName})
7
8 return suricata_alerts

```

*Parcours à nouveau du fichier de règles testé, en supprimant toutes les lignes qui ont été détectées comme étant à l'origine d'erreurs. "suricata\_\_alerts", les règles finales récupérées.*

FIGURE 3.10 – Récupération des règles en excluant les erreurs

Afin de pouvoir tenir compte des règles qui commettent des erreurs au fil du temps, celles-ci sont stockées dans des fichiers CSV (Figure 3.12) avec leur code d'erreur. Cela permet d'établir des statistiques sur le taux de règles valides envoyées par les partenaires de renseignement sur la menace.

***Note***

Le format CSV a été choisi pour stocker les informations relatives aux fichiers de configuration et aux résultats du programme. Ce format est utilisé parce qu'il est compatible avec Excel et Splunk, deux logiciels largement utilisés au sein de l'Agence.

### 3.1.3 Filtrage des règles

Une fois que les règles non valides ont été supprimées de l'ensemble traité, un autre problème se pose. Par la multitude de fournisseurs de règles, il existe le risque que la même règle (ayant probablement été établie à partir du même IOC) soit partagée par plusieurs partenaires différents. C'est pourquoi, pour éviter la surcharge inutile des sondes, il m'a été demandé de permettre à l'outil développé de filtrer les règles dupliquées.

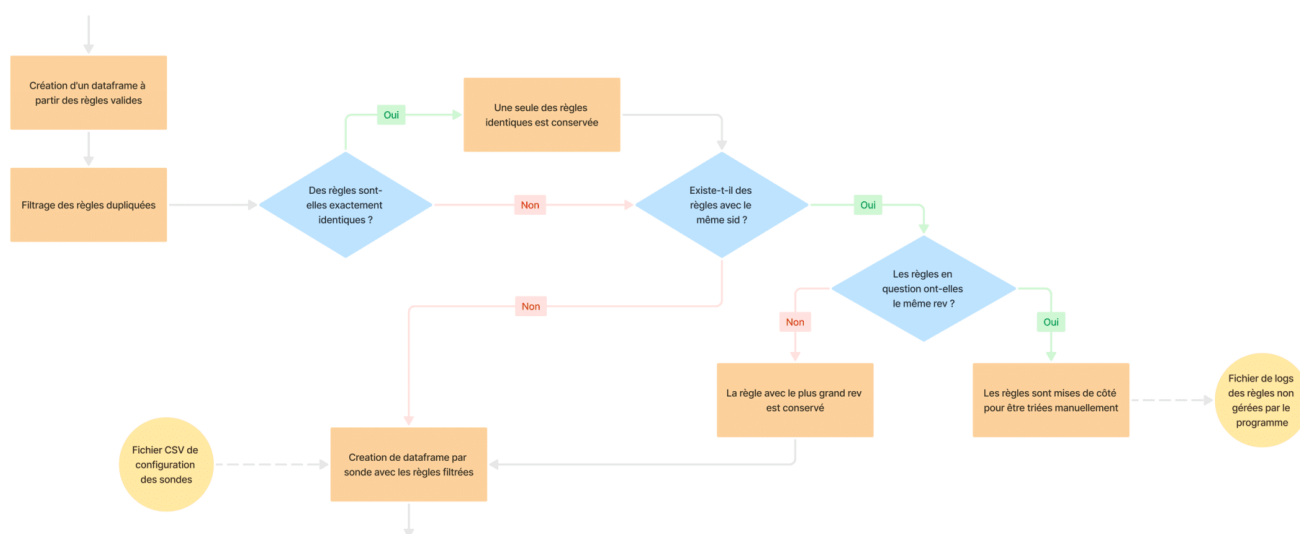


FIGURE 3.11 – Diagramme de flux de la section responsable du filtrage des règles

Pour mener à bien cette tâche (ainsi que d'autres manipulations que le programme devra appliquer à ces données par la suite), j'ai eu besoin d'utiliser une bibliothèque logicielle de data-science : *pandas*<sup>3</sup>. Elle permet de créer des ensembles de structures de données bidimensionnelles, *dataframes*, qui permettent au programme de rassembler toutes les règles dans une seule variable et d'appliquer des modifications à l'ensemble en quelques lignes de code seulement.

3. Pandas est une bibliothèque open-source flexible d'analyse et de manipulation de données pour Python : <https://pandas.pydata.org/>

```

1 #Import de la librairie
2 import pandas as pd
3
4 Dataframe_alerts = pd.DataFrame(suricata_alerts)

```

*Importation de la bibliothèque Pandas et création d'un dataframe à partir de "suricata\_alerts", une liste composée de toutes les règles récupérées jusqu'à présent.*

FIGURE 3.12 – Création de dataframe *pandas*

### Note

J'avais initialement essayé de réaliser ce programme en utilisant uniquement les outils fournis par défaut par Python, mais la taille colossale des données à traiter (plusieurs dizaines de milliers de règles) a rendu cette option inefficace et a provoqué un ralentissement excessif de l'exécution du programme. Cela m'a contraint à recourir à des outils de traitement de données. Le choix s'est porté sur Pandas car il s'agit d'une bibliothèque reconnue pour sa performance et qui est régulièrement maintenue.

Grâce à cette bibliothèque, le processus de filtrage s'est déroulé en deux étapes. Tout d'abord, les règles qui sont strictement identiques à d'autres sont supprimées.

```

1 # Trouve toutes les regles strictement en double pour n'en garder qu'une
2 duplicates = allrules[allrules.duplicated('rules', keep=False)]
3 if duplicates.empty == False:
4     grouped = duplicates.groupby('rules')['source'].agg(', '.join).reset_index()
5     allrules = pd.concat([allrules[~allrules['rules'].isin(duplicates['rules'])],
6                           grouped], ignore_index=True)

```

FIGURE 3.13 – Suppression des règles dupliquées

Puis elles sont triées en fonction du *sid* et de la *rev* des règles. Cette seconde étape est due au fonctionnement de Suricata qui exige que la section *options* des règles comporte une variable *sid* (pour signature ID) unique à chaque règle, qui est utilisée pour les identifier. Comme les partenaires de l'Agence respectent les conventions d'écriture des règles et en particulier les plages d'allocation de *sid*<sup>4</sup>, différentes règles ayant le même *sid* sont nécessairement basées sur le même IOC et le programme n'a plus qu'à décider lequel conserver.

---

4. <https://sidallocation.org/>

Pour départager deux règles ayant le même *sid*, j'utilise une autre variable présente dans les options des règles Suricata *rev* (pour revision) qui sert à spécifier un numéro de version d'une règle. La règle ayant la valeur numérique de sa variable *rev* la plus élevée est la règle qui à été mise à jour en dernier et sera donc privilégiée.

```

1 # On ceer une collone sid dans le dataframe a partir des sid des regles
2 allrules['sid'] = allrules['rules'].str.extract(r'sid:(.*?);')
3
4 # On trouve toutes les sid n'etant pas unique dans le dataframe
5 conflictrules = allrules[allrules.duplicated('sid', keep=False)]
6
7 # Creation d'une nouvelle collone rev pour priorisation par rev
8 conflictrules['rev'] = conflictrules['rules'].str.extract(r'rev:(.*?);').
    astype(int)
9 # Groupe par 'sid'
10 grouped = conflictrules.groupby('sid')
11 # Groupe par 'rev'
12 max_rev_per_group = grouped['rev'].max()
13
14 # Garde pour chaque sid la regle ayant la plus haute rev. si des regle on la meme
    sid et rev elles seront non resolu
15 conflictrules = conflictrules[conflictrules.apply(lambda x: x['rev'] ==
    max_rev_per_group[x['sid']], axis=1)]

```

FIGURE 3.14 – Gestions des règles en collisions de sid

Une fois le tri terminé, de nouveaux *dataframes* sont créés pour correspondre au nombre de sondes qui recevront des fichiers de règles en sortie du programme. Ces différents *dataframes* sont utilisés pour appliquer des modifications aux règles en fonction des sondes cibles.

```

1 # creer un dataframes de regles pour chaque sonde
2 rulesByProbes = {}
3 for name in SondesNames:
4     rulesByProbes[name[0]] = cleanedrules[:]

```

*Crée un dictionnaire d'images de données pour chaque sonde finale.*

FIGURE 3.15 – Création de dataframe pour chaque sonde

### 3.1.4 Modification des règles

Les membres du SOC-MC m’avaient fait part de leur besoin d’acquérir la capacité de modifier facilement les ensembles de règles. Les analystes du SOC-MC observent lors de leur travail quotidien l’efficacité des règles mises en place sur les sondes. Leurs observations et enquêtes ont mis en évidence des règles inutiles ou trop "bruyantes" que mon programme devait être capable de modifier.

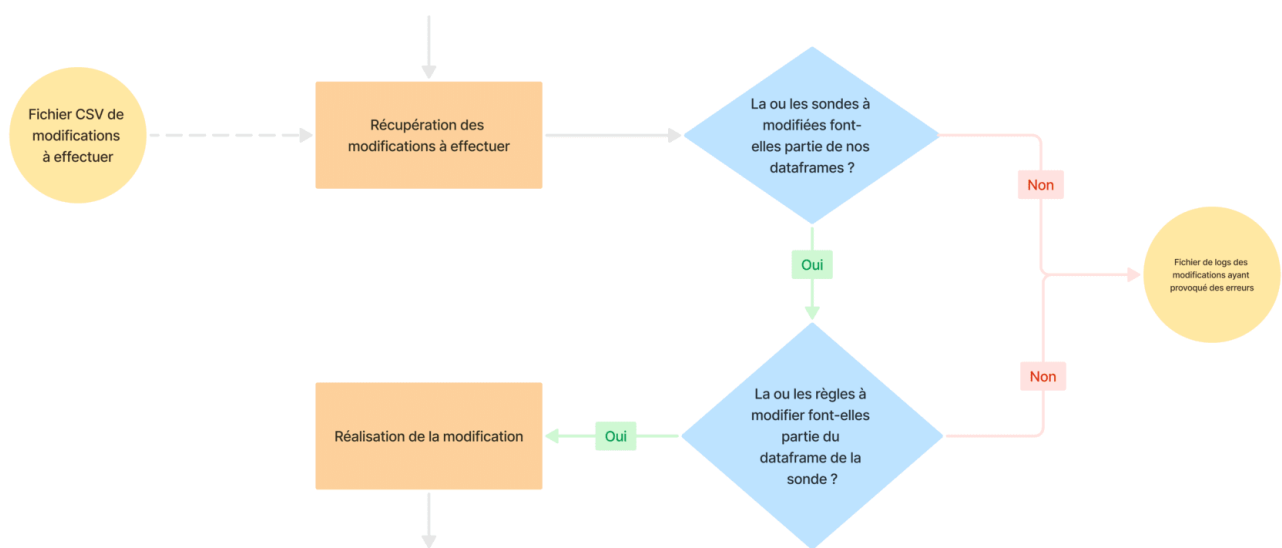


FIGURE 3.16 – Diagramme de flux de la section responsable de l’application des modifications

Les modifications à effectuer sont fournies au programme sous la forme d’un fichier CSV stipulant les règles à rectifier en fonction de leur *sid* ainsi que l’amendement à apporter et pour quelle sonde l’appliquer. Ce fichier est complété soit par les analystes SOC, soit par d’autres membres du CERT-MC sur la base de leurs constatations quant à l’efficacité des règles sur les sondes.

	A	B	C	D
1	SID	action parametre1 parametre2 parametreX	sonde1 sonde2 sondeX ALL	commentaire
2	3255564	disable	OIV1 OIV2 OIV3	Règle inutile sur les réseaux spécifiés
3	4666365	threshold by_dst 10 10	OIV2	La règle génère trop d'alertes sur le réseau OIV2
4	6526636	disable	ALL	Règle ne sonne jamais

FIGURE 3.17 – Exemple de fichier CSV de configuration des règles à modifier

Le programme récupère le contenu du fichier et désactive les règles ou applique des seuils pour réduire le nombre d'alertes qu'elles génèrent (des "threshold" dans la terminologie de Suricata).

```

1 # On applique la modification
2 if (changeToApply == "disable"):
3     # Pour 'disable' on enleve la regle de la liste
4     rulesByProbes[sonde[0]] = rulesByProbes[sonde[0]][rulesByProbes[sonde[0]]["
sid"] != targetSid]
5 elif len(changeToApply) > 3:
6     # Hors 'disable' il s'agit d'un changement de threshold
7     # On cree la nouvelle option
8     changeToApply = "threshold: type " + changeToApply[0] + ", track " +
changeToApply[1] + ", seconds " + changeToApply[3] + ", count " +
changeToApply[2] + ";";
9     # rulesToChange = la regle qui va etre modifiee
10    rulesToChange = rulesByProbes[sonde[0]].loc[rulesByProbes[sonde[0]]["sid"] ==
targetSid, 'rules']
11    # On modifie la section msg de la regle pour notifier qu'elle a ete modifiee
12    rulesToChange = rulesToChange.str.replace('msg:', 'msg:"AMSN_RULE ')
13    # Si l'option threshold existe deja dans rulesToChange on la remplace par la
nouvelle
14    if (re.search('threshold:', rulesToChange.tolist()[0])):
15        rulesToChange = rulesToChange.str.replace(r'threshold:(.*?);',
changeToApply, regex=True)
16    # S'il n'y a pas d'option threshold dans la regle modifiee on place la
nouvelle option a la fin
17    else:
18        rulesToChange = rulesToChange.values[0].rsplit(')', 1)[0] + changeToApply
+ ')\n'
19    # on reincopore la regle modifiee dans le dataframe de la sonde selectionnee
20    rulesByProbes[sonde[0]].loc[rulesByProbes[sonde[0]]["sid"] == targetSid, '
rules'] = rulesToChange

```

*Modifie les règles en écrivant à l'intérieur les nouvelles valeurs dans le cas de changement de 'threshold' et retire la règle du dataframe dans le cas d'un changement 'disable'.*

FIGURE 3.18 – Modifications des règles

Ces seuils sont une option activée par Suricata, permettant de limiter le nombre d'alertes par minute générées par une règle. Les modifications ne sont apportées qu'aux *dataframes* destinés aux sondes spécifiées dans le fichier CSV, permettant une différenciation par OIV.

### 3.1.5 Traitement des résultats

Afin de répondre aux exigences de l'Agence, le programme doit utiliser les opérations précédentes pour générer un fichier de règles, pour chaque sonde destinataire, qui puisse être exporté facilement. En outre, pour permettre à l'Agence de suivre les résultats du programme dans le temps, j'ai dû mettre en place un système de journalisation dans le code.

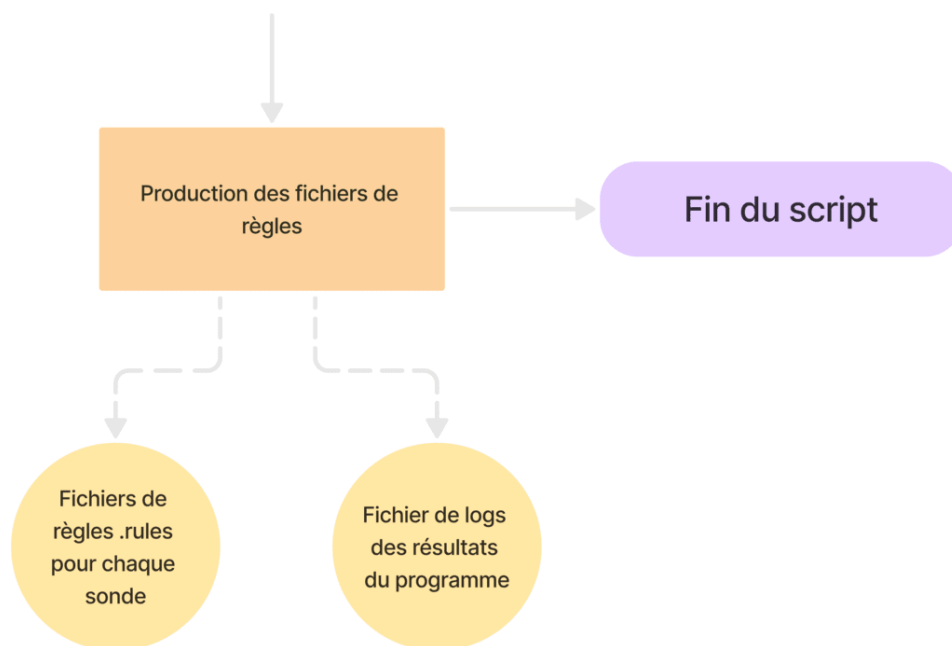


FIGURE 3.19 – Diagramme de flux de la fin du programme

À partir des *dataframes* mentionnés précédemment, un fichier de règles a été généré pour chaque sonde et est stocké dans un dossier distinct afin de pouvoir être exporté vers le système de gestion des sondes.



```

1  ### AMSN generated rules files ###
2
3  alert udp any any -> $HOME_NET 139 (msg:"ET NETBIOS Microsoft Windows NETAPI
    Stack Overflow Inbound - MS08-067 (1)"; content:"|0B|"; offset:2; depth:1;
    content:"|C8 4F 32 4B 70 16 D3 01 12 78 5A 47 BF 6E E1 88|"; reference:url,www
    .microsoft.com/technet/security/Bulletin/MS08-067.msp; reference:cve
    ,2008-4250; reference:url,www.kb.cert.org/vuls/id/827267; classtype:attempted-
    admin; sid:2008690; rev:5; metadata: amsn_source gcenter_netbios, created_at
    2010_07_30, cve CVE_2008_4250, updated_at 2017_11_22;)
4
5  alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"ET NETBIOS Microsoft SMB
    NetLogon UUID detected Big Endian SET"; flow:to_server,established; content:"|
    ff|SMB"; content:"|05 00 0b|"; distance:0; byte_test:1,!&,0x10,1,relative;
    content:"|12345678 1234 abcd ef00 01234567cffb|"; distance:29; flowbits:set,
    smb.netlogon.uuid.detected; flowbits:noalert; reference:cve,2010-2742;
    reference:url,www.microsoft.com/technet/security/bulletin/MS10-101.msp;
    classtype:attempted-user; sid:2800985; rev:2; metadata: amsn_source
    gcenter_netbios, created_at 2010_12_15, cve CVE_2010_2742, updated_at 2017
    _11_22;)
6
7  ...

```

FIGURE 3.20 – Exemple de fichier de règles généré

Pour pouvoir répondre au besoin d'information sur les résultats du programme, j'ai mis en place une journalisation via la génération de CSV tout au long de l'exécution du programme :

- Lorsque les règles récupérées sont testées, un fichier CSV est créé pour enregistrer les règles qui ont été détectées comme étant à l'origine d'erreurs et leurs sources.

	A	B
1	rules	error
2	E: detect: error parsing signature "alert http \$EXTERNAL_NET any -> \$HOME_NET any (msg:"E E: detect: previous sticky buffer has no matches	
3	E: detect: error parsing signature "alert http \$HOME_NET any -> \$EXTERNAL_NET any (msg:"E E: suricata: Loading signatures failed.	
4	E: detect: error parsing signature "alert http \$HOME_NET any -> \$EXTERNAL_NET any (msg:"E	
5	E: detect: error parsing signature "alert http \$HOME_NET any -> \$EXTERNAL_NET any (msg:"E E: detect: previous sticky buffer has no matches	
6	E: detect: error parsing signature "alert tcp \$HOME_NET any -> \$EXTERNAL_NET \$HTTP_PORT E: detect: previous sticky buffer has no matches	
7	E: detect: error parsing signature "alert tls \$HOME_NET any -> \$EXTERNAL_NET 443 (msg: "MI E: detect: previous sticky buffer has no matches	
8	E: detect: error parsing signature "alert http \$HOME_NET any -> \$EXTERNAL_NET any (msg: "I E: suricata: Loading signatures failed.	
9	E: detect: error parsing signature "alert http \$HOME_NET any -> \$EXTERNAL_NET any (msg: "I	
10	E: detect: error parsing signature "alert http \$HOME_NET any -> \$EXTERNAL_NET \$HTTP_POR E: detect-parse: invalid formatting to msg keyword: value must be double quoted 'msg'	
11	E: detect: error parsing signature "alert http \$HOME_NET any -> \$EXTERNAL_NET \$HTTP_POR E: detect-parse: bad input for keyword content: ""	
12	E: detect: error parsing signature "alert tcp \$EXTERNAL_NET \$HTTP_PORTS-> \$HOME_NET any E: detect-parse: unknown rule keyword '\ flow'.	
13	E: detect: error parsing signature "alert tcp \$EXTERNAL_NET \$HTTP_PORTS-> \$HOME_NET any E: detect-parse: unknown rule keyword '\ flow'.	

FIGURE 3.21 – Exemple de fichier CSV d'erreurs dans les fichiers testés

- Lorsque les règles en double sont filtrées, un fichier CSV est créé pour les règles potentielles qui ne peuvent pas être gérées automatiquement en vue d'un tri manuel.

	A	B	C	D
1	sid	source	rev	rules
2	2500000	gcenter_compromised, cti_community	6852	alert ip [101.47.10.60,101.96.119.195,103.195.238.130,104.131.166.202,104.131.167.19,104.131.168.56,10
3	13511174	gcenter_worm, misp_misp	1	alert ip 106.75.128.160 any -> \$HOME_NET any (msg: "MISP event:https://10.0.37.3/events/view/3836"
4	13511174	gcenter_worm, misp_misp	1	alert ip 106.75.128.158 any -> \$HOME_NET any (msg: "MISP event:https://10.0.37.3/events/view/3836"
5	2500000	gcenter_compromised, cti_community	6852	alert ip [101.47.14.60,101.96.119.195,103.195.238.130,104.131.166.202,104.131.167.19,104.131.168.56,10

FIGURE 3.22 – Exemple de fichier CSV de règles dupliquées non gérées

- Lorsque les modifications demandées par le personnel du CERT-MC sont appliquées, les modifications qui n'ont pas pu être effectuées (en raison d'une erreur dans les fichiers de configuration) sont sauvegardées dans un fichier CSV.

	A	B
1	Regle	Erreur
2	['2500002', 'disable', 'OIV1 OIV2 OIV3']	La sonde : 'OIV3' n'a pas été trouvé
3	['cti', 'community', 'gcap1']	La categorie : 'cti_community' n'a pas été trouvé
4	['gcenter', 'icmp', 'ALL']	La categorie : 'gcenter_icmp' pour la sonde : 'OIV1' n-a pas ete trouve
5	['gcenter', 'icmp', 'ALL']	La categorie : 'gcenter_icmp' pour la sonde : 'OIV2' n-a pas ete trouve
6	['gcenter', 'deleted', 'ALL']	La categorie : 'gcenter_deleted' pour la sonde : 'OIV1' n-a pas ete trouve
7	['gcenter', 'deleted', 'ALL']	La categorie : 'gcenter_deleted' pour la sonde : 'OIV2' n-a pas ete trouve
8	['gcenter', 'icmp_info', 'OIV4']	La categorie : 'gcenter_icmp_info' n'a pas été trouvé

FIGURE 3.23 – Exemple de fichier CSV d'erreur de modification de règle

- Lorsque le programme est terminé, un fichier de statistiques générales est généré.

	A	B
1	Nom	Valeur
2	Nombre d erreur Suricata pour info	1
3	Nombre d erreur Suricata pour trojan	3
4	Nombre d erreur Suricata pour cti_community	3796
5	Nombre de regles recupere (total sans erreur)	69605
6	Nombre de regles non recupere (erreur Suricata total)	3800
7	Categories de regles recupere par le programme	netbios, 'info', 'ciarmy', 'drop', 'botcc.portgrouped', 'inappropriate', 'dshield', 'mobile_malware', 'trojan', 'sql', 'web
8	Nombre de regle en conflit trouvee	2
9	Nombre de regle en conflit non resolu	2
10	Nombre de regles apres nettoyage (apres retrait des dupliquees ou commentees ou conflit)	69601
11	Nombre de sonde recuperee par le programme	2
12	Nombre de modification recuperee par le programme	0
13	Nombre de categories à desactiver recuperee par le programme	0
14	Nombre d'erreur lors de l'application des modifications	0
15	OIV1 nombre de regles	69601
16	OIV1 categories presente sur la sonde	['netbios', 'info', 'ciarmy', 'drop', 'inappropriate', 'dshield', 'mobile_malware', 'trojan', 'sql', 'web_specific_apps', 'ch
17	OIV2 nombre de regles	69601
18	OIV2 categories presente sur la sonde	['netbios', 'info', 'ciarmy', 'drop', 'inappropriate', 'dshield', 'mobile_malware', 'trojan', 'sql', 'web_specific_apps', 'ch

FIGURE 3.24 – Exemple de fichier CSV de statistiques du programme

Ces fichiers sont générés à chaque mise à jour des sondes et sont stockés séparément par date. Les fichiers ainsi créés permettent de comparer les résultats des différentes mises à jour des sondes et de suivre dans le temps quelles règles étaient actives sur chaque sonde à un moment donné.

L'importance de développer cette capacité est de permettre au programme de répondre à certaines exigences du référentiel *PDIS*<sup>5</sup> de l'ANSSI auxquelles l'ancien processus d'alimentation des sondes ne pouvait répondre.

### Note

Le développement de ce code a duré environ un mois et demi, comprenant la production de tests unitaires et de documentation, suivi d'une période de vérification du code par mon tuteur de stage et la planification de la mise en production du programme. Au moment de la publication de ce document, le programme est effectivement utilisé dans les processus courants de l'Agence.

5. "Le prestataire doit élaborer et tenir à jour pour chaque commanditaire la liste de l'ensemble des règles de détection mises en œuvre ou ayant été mises en œuvre dans le cadre de la prestation." [9]

## 3.2 Intégration de MISP

Après la finalisation du programme précédemment analysé, de nouveaux entretiens avec Bruno VALENTIN et Sébastien ABBONDANZA ont eu lieu afin de réfléchir aux améliorations à mettre en place durant le temps restant de mon stage.

Nos réflexions sur l'état actuel du système nous ont amenés à estimer que le nombre de sources de CTI de l'AMSN était trop faible pour garantir la capacité des sondes à détecter toutes les menaces pertinentes et existantes à l'état de l'art. Aussi, cette observation nous a conduits à envisager l'intégration de nouvelles sources de renseignements cyber pour mon programme et plus largement pour l'Agence.

Notre choix s'est porté sur l'intégration et l'exploitation d'une instance MISP<sup>6</sup>, qui est une plateforme collaborative de partage d'informations, conçue pour améliorer la détection et la prévention des incidents de sécurité. Ce projet open source a été initié par l'équipe de sécurité du CIRCL<sup>7</sup> afin d'encourager la collaboration internationale de lutte contre les cybermenaces.

La plateforme MISP est utilisée par des organisations du monde entier, notamment des agences gouvernementales, des entreprises privées, des équipes de réponse aux incidents de sécurité (CSIRT/CERT), ainsi que des chercheurs en cybersécurité. Les informations partagées via MISP sont partiellement publiques, en effet, une partie, à laquelle l'AMSN a le droit d'accéder, est confidentielle et partagée uniquement entre partenaires de confiance.

### *Note*

Il existe de nombreuses solutions de partage de renseignements (OpenCTI, alienvault, etc.) mais MISP nous a semblé être le choix le plus approprié car il s'agit d'une des plateformes les plus utilisées et qui permet de centraliser des informations provenant des autres outils. De plus, les partenaires de l'Agence (notamment l'ANSSI) s'appuient déjà sur cet outil pour partager des "feed"<sup>8</sup> d'information.

---

6. Malware Information Sharing Platform and Threat Sharing : <https://www.misp-project.org/>

7. Centre Gouvernemental de Cyberdéfense du Luxembourg : <https://www.circl.lu/>

8. <https://misp.cert.ssi.gouv.fr/feed-misp/>

### 3.2.1 Fonctionnement de MISP

L'échange d'informations entre entités utilisatrices de MISP se réalise via la création d'évènements qui sont partagés entre les instances locales de MISP dont bénéficient les différentes organisations. A titre d'exemple, ce processus peut prendre cette forme :

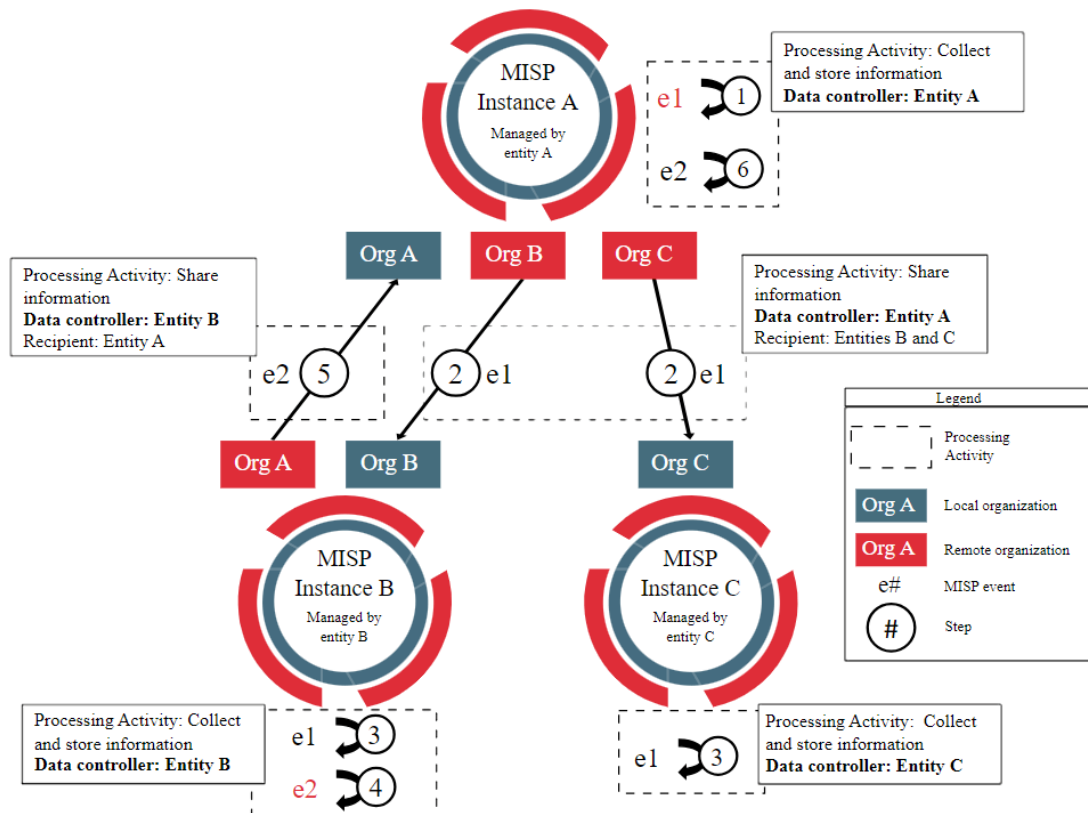


FIGURE 3.25 – Démonstration de l'échange d'informations entre entités via MISP

- **1** : Un événement MISP "e1" est créé par l'entité **A**.
- **2** : L'entité **A** décide de partager l'événement avec des organisations distantes partenaires (entités **B** et **C**).
- **3** : Les entités **B** et **C** stockent l'événement "e1" dans leur instance locale de MISP.
- **4** : L'entité **B** crée un second événement "e2".
- **5** : L'entité **B** décide de ne partager "e2" qu'avec l'entité **A**.
- **6** : L'entité **A** stocke l'événement "e2".

Un événement MISP est un ensemble structuré d'informations relatives à un incident, une menace ou une observation spécifique en matière de cybersécurité, qui est géré au sein de la plateforme MISP. Ces événements permettent d'organiser et de partager des informations détaillées sur divers types de cybermenaces, notamment les attaques de logiciels malveillants, les tentatives d'hameçonnage, les vulnérabilités et d'autres incidents de sécurité. Chaque événement MISP peut inclure plusieurs attributs et indicateurs de compromission (IOC) qui permettent de comprendre la menace et d'être capable de la détecter à nouveau. Un événement MISP se compose principalement des éléments suivants :

- **Attributes** : Il s'agit de données individuelles qui décrivent des caractéristiques spécifiques de l'événement. Les attributs peuvent inclure des adresses IP, des noms de domaine, des hachages de fichiers, des adresses électroniques, des URL, des échantillons de logiciels malveillants et d'autres détails techniques qui permettent d'identifier et d'analyser la menace.
- **Tags** : Les "Tags" sont utilisés pour classer et étiqueter les événements afin de faciliter la recherche et le filtrage. Les étiquettes peuvent inclure des classifications telles que "phishing", "APT" (Advanced Persistent Threat), "malware" ou "ransomware", ainsi que des informations sur la sensibilité ou les exigences de traitement des données.
- **Objects** : Ce sont des groupes d'attributs liés qui, ensemble, décrivent des entités ou des scénarios plus complexes, tels qu'une personne, un courrier électronique ou un périphérique de réseau. Cela permet de fournir un contexte et une structure aux données.
- **Relations** : Les événements peuvent être liés à d'autres événements ou objets pour indiquer les relations entre différents incidents, ce qui permet aux utilisateurs de voir les connexions et les corrélations qui ne sont pas toujours évidentes.
- **Comments** : Les utilisateurs peuvent ajouter des commentaires à un événement pour fournir un contexte, une analyse ou des recommandations supplémentaires, ce qui facilite la collaboration et le partage des connaissances entre les analystes.
- **Threat Level/Analysis** : Les événements comprennent souvent un indicateur de niveau de menace et de niveau d'avancement de l'analyse de l'événement (faible, moyen, élevé), qui peut aider à hiérarchiser la réponse à des menaces spécifiques.

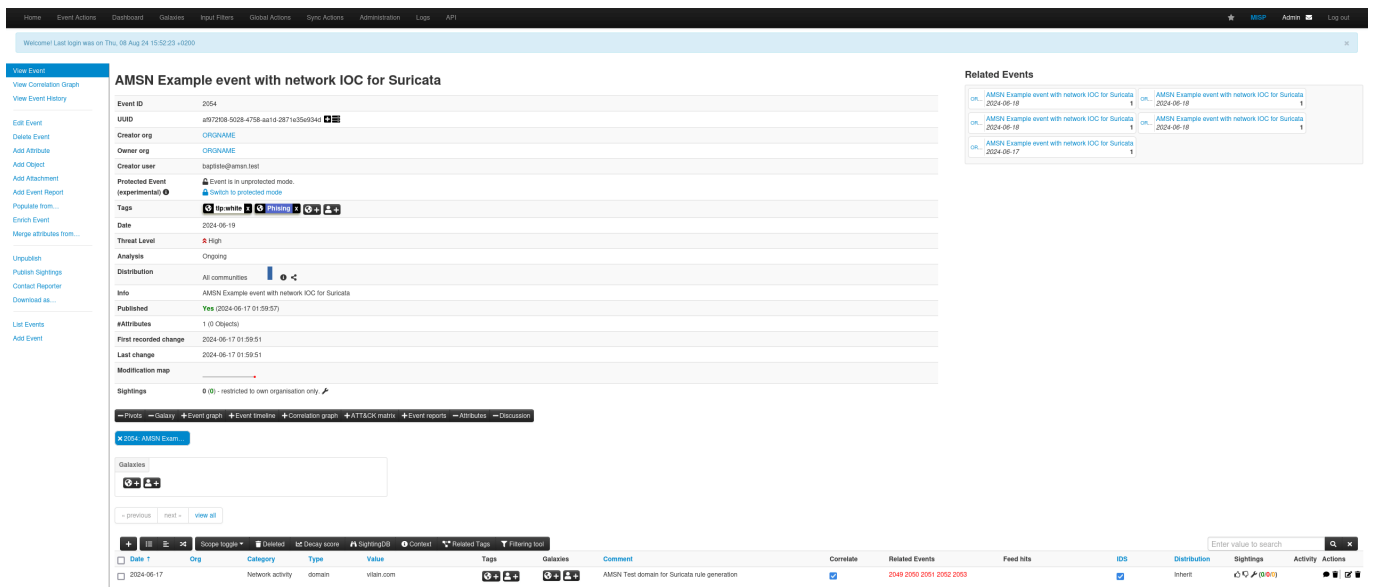


FIGURE 3.26 – Exemple d’événement MISP (source : capture d’écran de l’interface graphique de l’instance de test locale MISP)

Parmi ces éléments, les plus importants pour répondre à notre problématique sont les *Attributes* et les *Objects* (qui sont des ensembles d’*Attributes*), lesquels contiennent les éléments d’identification uniques des événements qui peuvent être utilisés pour créer des IOC, les autres servant principalement d’éléments de contexte et d’explication de l’événement. Dans ces *Attributes*, je vais extraire les IOC compatibles avec Suricata (adresses IP, noms de domaine, URL, hachages de fichiers et User-Agent), afin de les transformer automatiquement en règles de détection qui seront ajoutées à notre ensemble de règles envoyées aux sondes.

## Note

J’ai également eu l’occasion d’effectuer quelques tâches supplémentaires avec MISP afin de soutenir le SOC-MC au quotidien. Les analystes du SOC m’avaient fait part de leur difficulté à comprendre certaines règles envoyées sans contexte par les partenaires. MISP m’a permis de répondre en partie à cette difficulté en offrant la possibilité de contextualiser les règles qui génèrent des alertes à partir d’événements sur le MISP partageant les mêmes IOC. Cependant, ce travail sort du cadre de ma problématique, voire est négligeable par rapport à l’ensemble de mon travail, je n’y reviendrai donc pas dans la suite de ce document.

### 3.2.2 Exploitation de MISP

La première étape de l'intégration de MISP dans le processus de mise à jour des règles a consisté à installer une VM (machine virtuelle), dans le réseau interne de l'Agence, à l'aide des guides fournis par le CIRCL<sup>9</sup>. Pour la tenir à jour avec les derniers événements partagés sur le réseau MISP sans avoir à exposer notre instance locale, j'ai utilisé MISP Guard<sup>10</sup> qui permet de créer un proxy capable de récupérer les mises à jour de manière sécurisée sans menacer notre instance locale ou notre réseau interne.

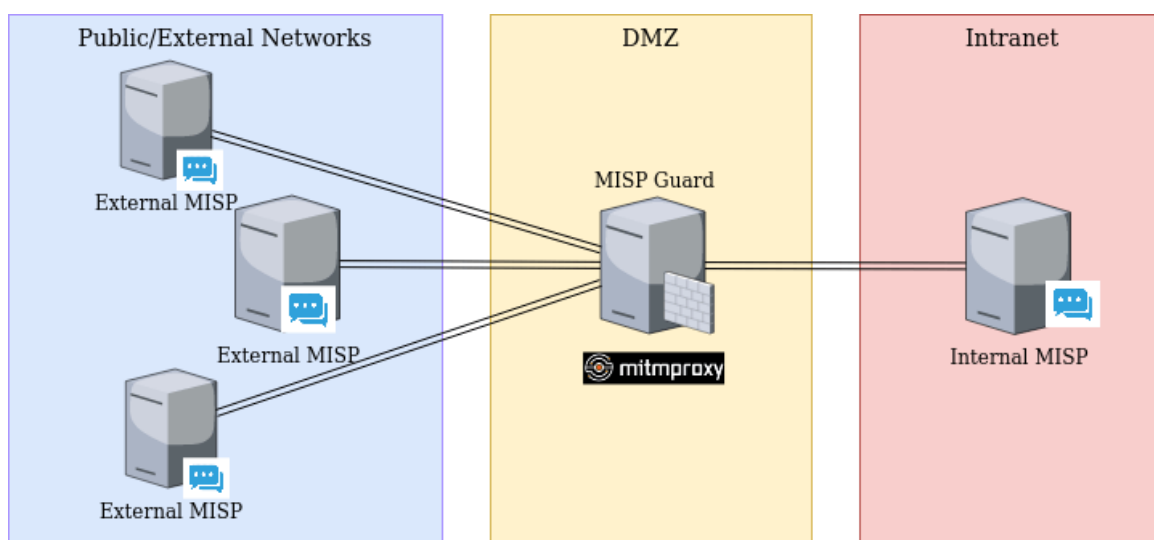


FIGURE 3.27 – Installation sécurisée de MISP dans le réseau de l'Agence

L'instance locale de MISP est utilisée systématiquement via l'API. Mon travail sur le MISP a alors principalement consisté à développer des scripts Python utilisant PyMISP<sup>11</sup>, une bibliothèque Python open-source officielle du CIRCL qui facilite l'utilisation d'une instance locale du MISP. Mon travail de développement a été divisé en trois parties :

- 1. Enrichissement de l'instance locale avec des IOC ;
- 2. Sélection des IOC ;
- 3. Génération de règles à partir des IOC.

---

9. <https://misp.github.io/MISP/>

10. un add-on mitmproxy développé par l'équipe qui est à l'origine de MISP : <https://www.misp-project.org/2022/09/13/misp-guard.html/>

11. <https://github.com/MISP/PyMISP>



## Enrichissement de l'instance locale avec des IOC

La première étape pour rendre MISP utile après l'installation est de centraliser les données de renseignements à l'aide de l'API. Pour ce faire, j'ai développé un script dont la fonction est d'alimenter notre instance locale avec des événements provenant de sources publiques, d'une part, et de nos propres sources internes, d'autre part.

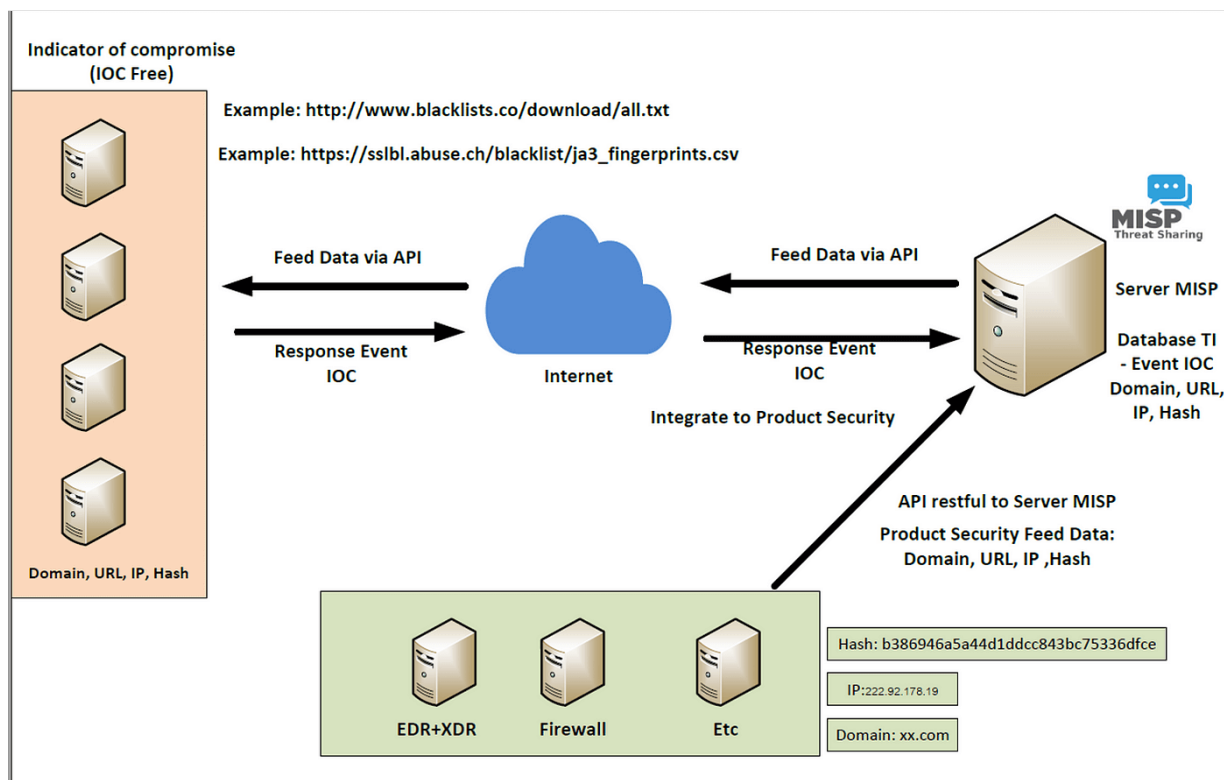


FIGURE 3.28 – Récupération des IOC via l'API de notre instance MISP locale

En ce qui concerne les sources publiques, si je n'ai utilisé que les deux "Feeds" fournis par défaut par le CIRCL<sup>12</sup><sup>13</sup> lors de l'installation de l'instance, le travail que j'ai effectué permettra cependant l'intégration de sources supplémentaires à l'avenir.

12. <https://www.circl.lu/doc/misp/feed-osint/>

13. <https://www.botvrij.eu/data/feed-osint/>

```

1 # Import PyMISP
2 from pymisp import PyMISP
3 # local misp credential
4 from conf.misp_conf import misp_url, misp_user_key, misp_verifycert
5
6 # source d'événements "feeds" :
7 feeds = [
8     "https://www.circl.lu/doc/misp/feed-osint/",
9     "http://www.botvrij.eu/data/feed-osint/"
10 ]
11 # Autres "feeds" potentiels :
12 # "https://bazaar.abuse.ch/downloads/misp/",
13 # "https://urlhaus.abuse.ch/downloads/misp/",
14 # "https://osint.digitalside.it/Threat-Intel/digitalside-misp-feed/",
15
16 # Connexion avec l'instance MISP
17 misp = PyMISP(misp_url, misp_user_key, misp_verifycert, 'json')
18
19 # Recupération des événements depuis les "feeds"
20 download_events(feeds)
21
22 # Intégration des événements dans l'instance locale MISP
23 import_events(misp)

```

*Télécharge les événements à partir des sources fournies et les importe dans notre instance locale. "download\_events(feeds)" la fonction qui télécharge les événements. "import\_events(misp)" celle qui les importe dans notre instance locale de MISP. [A.3]*

FIGURE 3.29 – Enrichissement de l'instance locale de MISP en évènement

La seule importation des événements, publiés par ces deux sources au cours des années 2023 et 2024, suffit à remplir notre instance de dizaine de milliers d'événements pouvant chacun partager des centaines d'IOC.

En plus de ces événements, je récupère les IOC détectés par nos propres équipements (Firewall, Sonde, etc.) ou par nos partenaires et un autre script Python permet la transformation de ceux-ci en évènement MISP.

```

1 # Import PyMISP et MISPEvent
2 from pymisp import PyMISP
3 # local misp credential
4 from conf.misp_conf import misp_url, misp_user_key, misp_verifycert
5
6 # Connexion avec l'instance MISP
7 misp = PyMISP(misp_url, misp_user_key, misp_verifycert, 'json')
8 event = MISPEvent()
9
10 # Créer un événement sur l'instance local MISP
11 response = create_event(3, 1, 1, "AMSN Example event with network IOC for
    Suricata", {"category": 'Network activity', "type": 'domain', "value": 'Vilain
    .com', "comment": 'AMSN Test domain for Suricata rule generation'}, ["tlp:
    white", "Phising"])

```

*Créer un événement exploitable à partir des informations associées. "create\_event(...)" la fonction qui crée un événement à partir d'information fournie. [A.3]*

FIGURE 3.30 – Création événement MISP

Les événements créés et les informations qui y sont enregistrées ont été conçus pour être compréhensibles par les analystes du SOC-MC et pour être exportables sous la forme de règles Suricata de la même manière que *les autres IOC*.

## Sélection des IOC

*Comme mentionné plus haut*, seule une minorité d'IOC peut être utilisée, en raison du choix des technologies de détection de l'Agence. Afin d'optimiser le poids de notre instance locale et sa vitesse de réaction aux requêtes API de mes scripts, j'ai créé un autre script qui nettoie automatiquement notre instance de tous les IOC que nous ne sommes pas en mesure d'utiliser.

```
1 # Import PyMISP et MISPEvent
2 from pymisp import PyMISP
3 # local misp credential
4 from conf.misp_conf import misp_url, misp_user_key, misp_verifycert
5
6 # Types d'IOC dont nous avons besoin
7 suricata_types = [
8     'ip-dst', 'ip-src', 'domain', 'domain|ip', 'url', 'uri', 'hostname',
9     'md5', 'sha1', 'sha256', 'sha512'
10 ]
11
12 # Recuperation de tous les events de l'instance local
13 events = misp.search(controller='events')
14
15 # Interrogation des events et suppression des attributs incompatibles avec les
    donnees de Suricata
16 for event in events:
17     event_id = event['Event']['id']
18     misp_event = misp.get_event(event_id)
19
20     for attribute in misp_event['Event']['Attribute']:
21         if not is_suricata_compatible(attribute):
22             # Suppression de l'attribut non compatible avec Suricata
23             misp.delete_attribute(attribute['id'])
```

*Supprime tous les IOC qui ne font pas partie de ceux désignés dans la liste 'suricata\_types' de l'instance locale MISP.*

FIGURE 3.31 – Suppression des IOC inutiles

En outre, j'utilise une fonctionnalité offerte par MISP, les "Decaying Models"<sup>14</sup> qui attribuent aux IOC une durée de vie en fonction de leur type et un score qui diminue au fil du temps.

---

14. <https://www.misp-project.org/2019/09/12/Decaying-Of-Indicators.html/>

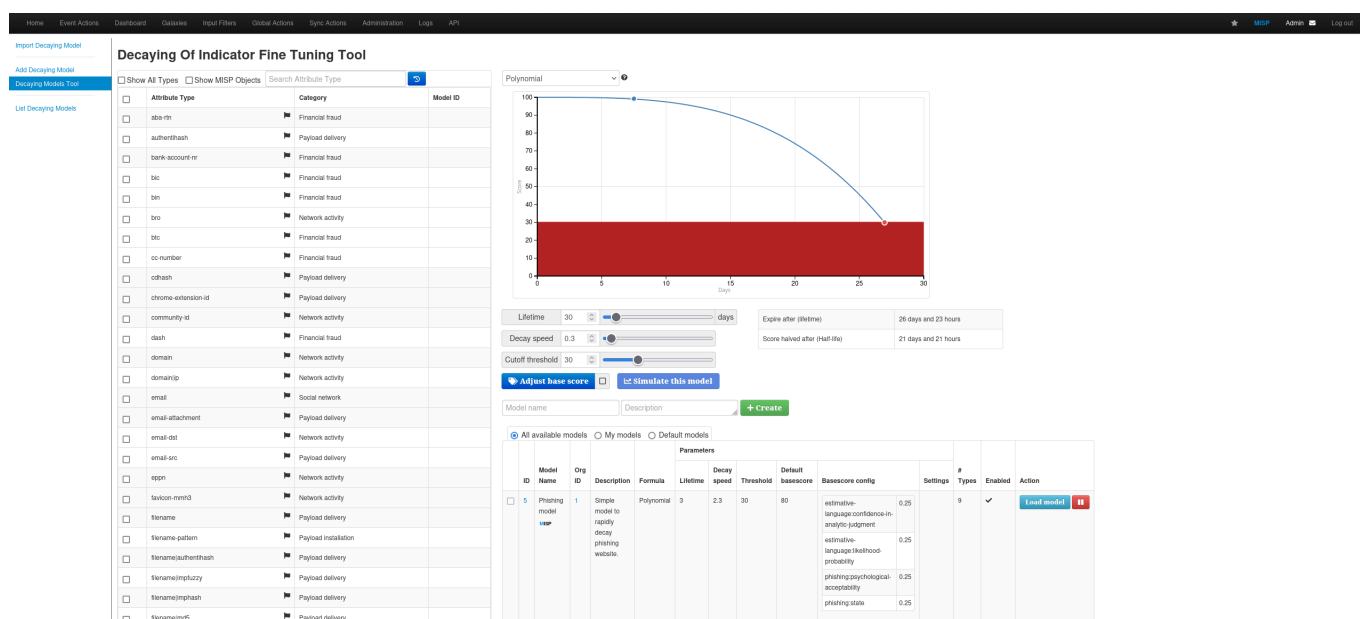


FIGURE 3.32 – Exemple de "Decaying Models" (source : capture d'écran de l'interface graphique de l'instance de test locale MISP)

Pour l'instant, le modèle par défaut proposé par le MISP "NIDS Simple Decaying Model" <sup>15</sup> est utilisé sur l'instance. Les adresses IP, les noms de domaine et les URL ont ainsi une durée de vie de deux mois, après quoi les IOC en question sont considérés comme obsolètes (car la probabilité que ces IOC ne soient plus utilisés, par les acteurs malveillants détectés à l'origine, est trop élevée). [2]

À terme, l'AMSN souhaiterait travailler à la création de ses propres modèles d'obsolescence des IOC sur la base du retour d'information des analystes du SOC-MC et de ses partenaires.

15. <https://github.com/MISP/misp-decaying-models/blob/main/models/nids-simple-model.json>

## Génération de règles à partir des IOC

Une fois que l'instance locale a été remplie d'événements et que les données ont été nettoyées, PyMISP offre la possibilité d'exporter directement les *Attributes* des événements en règles Suricata.

```
1 # Import PyMISP et MISPEvent
2 from pymisp import PyMISP
3 # local misp credential
4 from conf.misp_conf import misp_url, misp_user_key, misp_verifycert
5
6 # Prend tous les events et fabrique avec un fichier .rules exploitable
7 suricata_rules = misp.search(controller='events', return_format="suricata")
```

*Génère des règles de suricata à partir des IOC de tous les événements présents dans l'instance locale.*

FIGURE 3.33 – Génération de règle suricata depuis l'instance local MISP














Pour chaque attribut d'événement, une règle Suricata capable de détecter l'IOC de l'attribut est générée. Par exemple :

L'IOC :

Date ↑	Org	Category	Type	Value
2022-06-29		Network activity	ip-dst	192.95.20.8

Provenant de l'évènement :

## MAR-10382254.r1.v1: XMRIG Cryptominer

Event ID	128
UUID	21db8eec-8797-439d-94c1-4de8a6dac7e2  
Creator org	<a href="#">CUDESO</a>
Owner org	<a href="#">ORGNAME</a>
Creator user	admin@admin.test
Protected Event (experimental) 	 Event is in unprotected mode.
Tags	 <b>tlp:white</b>   <b>cryptominer</b>   
Date	2022-06-03
Threat Level	? Undefined
Analysis	Initial
Distribution	This community only   

Génère la règle :

```
1 alert ip $HOME_NET any -> 192.95.20.8 any (msg: "MISP e128 [] Outgoing To IP: 192.95.20.8"; classtype:trojan-activity; sid:2696602; rev:1; priority:4; reference:url,https://192.168.215.128/events/view/128;)
```

FIGURE 3.34 – Exemple de génération de règle Suricata par MISP (source : capture d'écran de l'interface graphique de l'instance de test locale MISP)

Les tests effectués avec le SOC-MC ont révélé que ces règles étaient fonctionnelles en l'état, mais qu'elles ne contenaient pas les informations contextuelles nécessaires pour comprendre et classer les alertes qu'elles généraient. Cela m'a conduit à créer un script supplémentaire pour reformater les règles générées par MISP. Pour ce faire, je n'ai modifié que deux variables dans la section *options* des règles qui n'affectent pas leur capacité de détection :

- Soit la variable 'msg' (qui est une variable facultative utilisée pour donner des informations contextuelles sur la règle) en indiquant le nom de l'événement lié à la règle avec un lien vers l'événement sur l'interface graphique de l'instance MISP, à laquelle les analystes SOC peuvent accéder depuis leur poste de travail à l'aide de leur navigateur ;
- Et la variable 'metadata' (qui est une variable optionnelle utilisée pour donner des informations supplémentaires à une règle) que j'ai complétée pour ajouter les 'Tags' de l'événement lié à cette règle.

```

1 # Prend tous les events
2 suricata_rules = misp.search(controller='events', return_format="suricata")
3
4 for rule in suricata_rules:
5     ruleurl = re.search(r'reference:url,(.*?);', rule)
6     if ruleurl:
7         # On remplace le message original des regles (peu lisible et trop long)
8         # par un message simple renvoyant vers la pages MISP de l'Event
9         rule = re.sub(r'msg: "(.*?)"', 'msg: "MISP event: ' + ruleurl.group(1) +
10         '";', rule)
11         # Ajout des tags de l'events
12         rule = re.sub(r'metadata: "(.*?)"', 'metadata: tags ' + eventTags + '";',
13         rule)
14     new_lines.append('alert ' + rule)

```

*Modifie les variables des règles générées par MISP.*

Règle après modifications :

```

1 alert ip $HOME_NET any -> 192.95.20.8 any (msg: "MISP Event : MAR-10382254.r1.v1:
  XMRIG Cryptominer https://192.168.215.128/events/view/128"; classtype:trojan-
  activity; sid:2696602; rev:1; priority:4; reference:url,https
  ://192.168.215.128/events/view/128; metadata: tags tlp:white|cryptominer;)

```

FIGURE 3.35 – Modification des règles générées par MISP

L'ajout du titre de l'événement présente l'avantage d'indiquer plus clairement aux analystes l'objectif de détection de la règle, et le lien vers l'interface graphique MISP leur permet de rechercher plus d'informations sur la règle directement sur MISP en cas d'investigation.



L'ajout de "Tags" à l'événement fournit également des informations supplémentaires sur la règle, mais permet surtout de classer les règles en catégories, qui peuvent ensuite être utilisées pour créer des statistiques basées sur celles-ci, ou évaluer leur utilité en fonction des sondes sur lesquelles elles seront appliquées.

Une fois reformatées, les règles générées peuvent être directement ajoutées aux règles ingérées par *mon premier programme* pour être intégrées dans les sondes.

### **Note**

Le développement de ce code a également pris environ un mois et demi, comprenant la rédaction de la documentation. L'instance de MISP mentionnée dans cette section a été installée et testée dans une section distincte du réseau interne avec des données de test. Les réactions de mon tuteur de stage et des analystes du SOC-MC sont positives quant au potentiel offert par cette instance locale, mais pour l'instant, elle n'est pas encore utilisée dans les processus actuels de l'Agence, dans l'attente de la planification du déplacement de la VM de l'instance sur le réseau de production.

### 3.3 Filtrage des règles générées par OIV

Un dilemme s'est posé lors de l'ajout des règles de MISP à celles des partenaires. Le nombre total de règles ingérées par les sondes se comptait auparavant en dizaine de milliers, mais une fois les règles MISP ajoutées, ce nombre s'élevait à des centaines de milliers.

	A	B
1	Nom	Valeur
2	Nombre d'erreur Suricata pour info	1
3	Nombre d'erreur Suricata pour trojan	3
4	Nombre d'erreur Suricata pour misp_misp	116
5	Nombre d'erreur Suricata pour cti_community	3796
6	Nombre de regles recupere (total sans erreur)	254837
7	Nombre de regles non recupere (erreur Suricata total)	3916
8	Categories de regles recupere par le programme	netbios', 'info', 'ciarmy', 'drop', 'botcc.portgrouped', 'inappropriate', 'dshield', 'mobile_malware', 'trojan', 'sql', 'web
9	Nombre de regle en conflit trouvee	4
10	Nombre de regle en conflit non resolu	4
11	Nombre de regles apres nettoyage (apres retrait des dupliquees ou commentees ou conflit)	254831
12	Nombre de sonde recuperee par le programme	2
13	Nombre de modification recuperee par le programme	0
14	Nombre de categories à desactiver recuperee par le programme	0
15	Nombre d'erreur lors de l'application des modifications	0
16	OIV1 nombre de regles	254831
17	OIV1 categories presente sur la sonde	['netbios', 'info', 'ciarmy', 'drop', 'inappropriate', 'dshield', 'mobile_malware', 'trojan', 'sql', 'web_specific_apps', 'ch
18	OIV2 nombre de regles	254831
19	OIV2 categories presente sur la sonde	['netbios', 'info', 'ciarmy', 'drop', 'inappropriate', 'dshield', 'mobile_malware', 'trojan', 'sql', 'web_specific_apps', 'ch

FIGURE 3.36 – Fichier CSV des statistique du programme après intégration des règles provenant de MISP

Ce nombre très élevé de règles risque de surcharger les sondes et de les ralentir, ce qui serait préjudiciable aux réseaux sur lesquels elles sont installées.

De plus, l'implémentation de centaines de milliers de règles sans filtrage des sondes augmente considérablement le risque d'avoir des règles inutiles ou de provoquer de nombreux faux positifs (détection erronée d'une menace ou d'une activité malveillante par un système de sécurité) qui peuvent surcharger les équipes de sécurité de l'Agence. Un véritable défi se pose alors pour filtrer les règles en fonction des OIV afin de maximiser leur efficacité et d'optimiser le nombre de règles appliquées à chaque sonde.

J'ai pu travailler sur ce point conjointement avec les membres du SOC-MC afin de hiérarchiser les règles en fonction de leur source et de leur type pour chaque OIV. D'une part, les règles fournies par les partenaires de renseignement sont classées en fonction de leur rôle et certaines d'entre elles sont destinées à des technologies spécifiques (par exemple 'SQL', 'ACTIVEX', 'SCADA', etc.). D'autre part dans l'environnement MISP, il est possible d'utiliser la fonctionnalité 'Tags' des événements pour trier les règles en catégories (par exemple les tags 'linux', 'windows', 'javascript', 'HealthCare', etc.).

En établissant ces catégories, nous pouvons ensuite, grâce à notre connaissance des technologies présentes sur les réseaux des OIV et leur collaboration avec l'Agence, estimer les règles pertinentes à retenir sur chaque sonde. Pour ce faire, j'ai complété *mon programme de gestion des règles* en ajoutant un segment permettant de désactiver des ensembles de règles en fonction de leur catégorie.

Ce nouveau segment fonctionne exactement comme *la section de modifications des règles*, mais utilise un autre fichier CSV fourni pour sélectionner les catégories de règles à supprimer pour les sondes.

fournisseur ▾	categorie ▾	sonde1 sonde2 sondeX ALL ▾	commentaire ▾
cti	community	OIV1	Revue du 23 janvier 2021
Fournisseur1	icmp	ALL	Trop bruyante
Fournisseur4	SQL	OIV1 OIV2	Technologies absentes
MISP	linux	OIV3 OIV4	Technologies absentes

FIGURE 3.37 – Exemple de fichier CSV de contrôle des catégories de règles

```

1 # Fonction pour parcourir les regles des sondes pour faire si possible la
  modification
2 def disable_categories(rulesByProbes, SelectedSondes, categorie, toDisable,
  modification_error):
3     # On parcourt chaque sondes cible
4     for sonde in SelectedSondes:
5         # On cherche si la categorie existe dans la sonde cible
6         if (categorie in rulesByProbes[sonde[0]]["source"].tolist()):
7             targetRow = rulesByProbes[sonde[0]][rulesByProbes[sonde[0]]["source"
8 ].str.contains(categorie)]
9             rulesByProbes[sonde[0]] = rulesByProbes[sonde[0]].drop(targetRow.
  index)

```

*Supprime les règles appartenant à certaines catégories sur des sondes ciblées.*

FIGURE 3.38 – Contrôle des catégories sur les sondes

Ce code ajouté permet de supprimer des milliers de règles ciblées sur chaque sonde, en ne conservant que celles qui sont les plus pertinentes, réduisant le nombre de règles par sonde à des niveaux plus supportables.

A		B	
1	Nom	Valeur	
2	Nombre d erreur Suricata pour info	1	
3	Nombre d erreur Suricata pour trojan	3	
4	Nombre d erreur Suricata pour misp_misp	116	
5	Nombre d erreur Suricata pour cti_community	3796	
6	Nombre de regles recupere (total sans erreur)	254837	
7	Nombre de regles non recupere (erreur Suricata total)	3916	
8	Categories de regles recupere par le programme	netbios', 'info', 'ciarmy', 'drop', 'botcc.portgrouped', 'inappropriate', 'dshield', 'mobile_malware', 'trojan', 'sql', 'we	
9	Nombre de regle en conflit trouvee	4	
10	Nombre de regle en conflit non resolu	4	
11	Nombre de regles apres netoyage (apres retrait des dupliquees ou commentees ou conflit)	254831	
12	Nombre de sonde recuperee par le programme	2	
13	Nombre de modification recuperee par le programme	153	
14	Nombre de categories à desactiver recuperee par le programme	33	
15	Nombre d'erreur lors de l'application des modifications	0	
16	OIV1 nombre de regles	60875	
17	OIV1 categories presente sur la sonde	['info', 'drop', 'inappropriate', 'dshield', 'web_client', 'imap', 'compromised', 'exploit', 'smtp', 'worm', 'tftp', 'malwa	
18	OIV2 nombre de regles	72045	
19	OIV2 categories presente sur la sonde	['netbios', 'info', 'ciarmy', 'drop', '3coresec', 'icmp_info', 'web_client', 'imap', 'compromised', 'exploit', 'smtp', 'wor	

FIGURE 3.39 – Fichier CSV des statistiques du programme après sélection des catégories par sonde

**Note**

Cette partie s’est déroulée sur quelques semaines, incluant la complétion de la documentation réalisée précédemment avec les nouveaux changements. Ensuite, le temps restant de mon stage a été consacré à la rédaction de ma thèse professionnelle, à sa relecture et à sa correction en collaboration avec mon tuteur de stage.

# Chapitre 4

## Résultat

### 4.1 Analyse des résultats

La gestion des sondes de détection peut être perçue comme une tâche complexe et contraignante pour les organisations impliquées dans la supervision des réseaux. Alors que les solutions proposées par des sociétés privées sont souvent très coûteuses, mon travail a permis de créer un outil interne facilitant le travail quotidien des membres du CERT-MC. Le *processus de gestion des sondes* de l'Agence a ainsi pu recevoir les améliorations suivantes :

- Les règles envoyées aux sondes sont désormais contrôlées avant d'être utilisées, minimisant ainsi le risque d'erreurs au moment du déploiement sur les sondes ;
- Il est maintenant possible de désactiver ou de modifier des règles spécifiques parmi celles envoyées aux sondes, offrant une meilleure flexibilité dans leur gestion ;
- Les règles provenant de MISP peuvent désormais être intégrées directement dans les sondes, enrichissant ainsi les sources d'informations pour une meilleure détection des menaces ;
- Les règles sont catégorisées, et il est possible de désactiver certaines catégories en fonction des besoins spécifiques de chaque sonde.

Ces améliorations ont considérablement optimisé le processus d'alimentation des sondes de l'Agence. En particulier, la possibilité de filtrer et d'ajuster dynamiquement les règles réduit les faux positifs, minimise la surcharge du système et facilite le travail des analystes du SOC-MC en leur fournissant un outil simple de gestion des règles.

A l'avenir, l'AMSN souhaite poursuivre le développement de l'outil, notamment son interfaçage avec MISP, afin d'étendre les capacités de détection de l'Agence par l'ajout de nouveaux flux d'information. Le dilemme de l'optimisation des règles, présentes sur les sondes, continuera à se poser et nécessitera un renforcement des relations entre l'Agence et ses OIV pour une meilleure sélection des catégories dont ils relèvent.

## 4.2 Bilan personnel

Mon expérience au sein de l'Agence Monégasque de Sécurité Numérique fut très enrichissante, tant sur le plan technique que professionnel. Ce stage m'a introduit au monde de la cybersécurité Étatique. J'ai pu côtoyer pour la première fois un CERT national et être témoin de leurs missions et organisations au quotidien. Ou encore travailler sur des problématiques concrètes, développer des solutions adaptées à des enjeux réels, et contribuer de manière significative à l'amélioration des processus de l'Agence.

Le projet de développement d'un outil automatisé de génération de règles de détection m'a permis de m'immerger dans des technologies clés des IDS de Suricata et de MISP, tout en approfondissant mes connaissances en programmation, notamment avec Python. La résolution de problèmes complexes, comme la gestion de milliers de règles et l'optimisation de leur efficacité, a développé ma capacité à analyser des situations et à proposer des solutions techniques robustes.

Sur le plan personnel, cette expérience m'a enseigné à mieux structurer mon travail, à collaborer avec des équipes multidisciplinaires, et à gérer un projet de bout en bout, depuis l'analyse des besoins jusqu'à la mise en production. J'ai également beaucoup appris en termes de communication, notamment en partageant régulièrement mes avancées avec mon tuteur de stage et les analystes du SOC-MC.

Enfin, cette période a renforcé mon intérêt pour le domaine de la cybersécurité et m'a confirmé dans mon choix de carrière. J'ai non seulement acquis des compétences techniques et méthodologiques, mais aussi développé une meilleure compréhension des enjeux stratégiques liés à la protection des systèmes d'information. Mon passage à l'AMSN m'a donné confiance en mes capacités à mener des projets ambitieux et à contribuer activement à la sécurisation des infrastructures numériques.

# Conclusion et Perspectives

Le développement d'un outil de gestion centralisée de règles de détection d'intrusion représente une étape clé dans l'amélioration de la réactivité et de la pertinence des systèmes de défense contre les cybermenaces. Ce document et le travail qu'il présente a permis d'apporter des solutions à une problématique complexe, tout en répondant aux exigences opérationnelles d'un organisme en charge de la protection des infrastructures critiques.

L'automatisation de la gestion des règles de détection permet non seulement de simplifier et d'accélérer les processus de mise à jour, mais aussi d'améliorer la précision et l'adaptabilité des systèmes de détection d'intrusion dans un contexte où les menaces évoluent rapidement. Ce projet a démontré l'importance de combiner des approches techniques pointues avec une vision stratégique, afin de mieux répondre aux enjeux de cybersécurité auxquels sont confrontés autant les États que les organisations.

Cependant, ce travail ne représente qu'une première étape dans l'évolution vers des systèmes de détection plus intelligents et plus proactifs. Les perspectives d'amélioration sont nombreuses. D'une part, une meilleure collaboration entre les différentes entités publiques et privées, à travers le partage accru de renseignements sur les menaces, est un progrès souhaitable pour renforcer l'efficacité des systèmes de détection d'intrusion. D'autre part, l'optimisation des règles en fonction des spécificités des infrastructures surveillées constitue un axe de développement essentiel pour minimiser les faux positifs et maximiser l'efficacité des sondes.

En conclusion, ce projet de fin d'études a non seulement contribué à répondre à des problématiques techniques concrètes, mais il ouvre également la voie à des évolutions futures dans le contexte étudié. Le cheminement entrepris dans ce cadre est le reflet d'un besoin constant d'innovation et d'adaptation face à un environnement en perpétuelle mutation, où la sécurité des systèmes d'information demeure un enjeu de premier plan.



# Glossaire

<b>AMSN (Agence Monégasque de Sécurité Numérique)</b>	Autorité nationale de l'Etat de Monaco en charge de la sécurité des systèmes d'information et des infrastructures critiques nationales
<b>ANSSI (Agence Nationale de la Sécurité des Systèmes d'Information)</b>	Organisme français chargé de la protection des systèmes d'information de l'État et des infrastructures critiques nationales
<b>API (Application Programming Interface)</b>	Ensemble de règles et de protocoles permettant à des applications ou services de communiquer entre eux
<b>CERT (Computer Emergency Response Team)</b>	Groupe de spécialistes formés et expérimentés dans le domaine de la réponse aux incidents de sécurité informatiques
<b>CIRCL (Computer Incident Response Center Luxembourg)</b>	Centre national luxembourgeois de réponse aux incidents de sécurité informatique
<b>CSV (Comma-Separated Values)</b>	Format de fichier texte où les données sont organisées en lignes et colonnes, chaque valeur étant séparée par une virgule ou un autre délimiteur
<b>CTI (Cyber Threat Intelligence)</b>	Collecte et analyse de données relatives aux cybermenaces pour anticiper, prévenir et répondre aux attaques
<b>IDS (Intrusion Detection System)</b>	Système de sécurité qui surveille le trafic réseau ou les activités système pour détecter des comportements suspects ou des intrusions
<b>IOC (Indicator of Compromise)</b>	Signe observable, tel qu'une adresse IP malveillante ou un fichier suspect, qui indique une potentielle compromission ou intrusion dans un système
<b>MISP (Malware Information Sharing Platform)</b>	Plateforme open-source dédiée au partage d'indicateurs de compromission et de renseignements sur les cybermenaces
<b>OIV (Opérateur d'Importance Vitale)</b>	Organisation identifiée par l'État comme ayant des activités indispensables à la survie de la nation ou dangereuses pour la population
<b>SOC (Security Operation Center)</b>	Équipe centralisée chargée de surveiller, détecter et répondre aux menaces de sécurité en temps réel au sein d'une organisation

<b>VM (Virtual Machine)</b>	Environnement informatique simulé qui émule un système physique complet, permettant d'exécuter un système d'exploitation et des applications de manière isolée
-----------------------------	--

# Annexe

## .1 configTestSuricata.yaml

Fichier de configuration Suricata utilisé lors de test de fichier. Les fonctionnalités non exploitées sont désactivées ("enabled : no") et les autres fonctionnalités (non montrées ici) sont laissées avec leurs valeurs par défaut.

```
1 %YAML 1.1
2 ---
3 # This configuration file generated by Suricata 7.0.3.
4 suricata-version: "7.0"
5 # The default logging directory.
6 default-log-dir: /var/log/suricata/
7 # Global stats configuration
8 stats:
9   enabled: yes
10   # The interval field (in seconds) controls the interval at
11   # which stats are updated in the log.
12   interval: 8
13 # Configure the type of alert (and other) logging you would like.
14 outputs:
15   # a line based alerts log similar to Snort's fast.log
16   - fast:
17     enabled: no
18     filename: fast.log
19   # Extensible Event Format (nicknamed EVE) event log in JSON format
20   - eve-log:
21     enabled: no
22     filename: eve.json
23     xff:
24       enabled: no
25       mode: extra-data
26       header: X-Forwarded-For
27   # output module to store certificates chain to disk
28   - tls-store:
29     enabled: no
30   # By default all packets are logged except:
31   # - TCP streams beyond stream.reassembly.depth
32   # - encrypted streams after the key exchange
33   - pcap-log:
34     enabled: no
```

```

1  # a full alert log containing much information for signature writers
2  # or for investigating suspected false positives.
3  - alert-debug:
4      enabled: no
5      filename: alert-debug.log
6      append: yes
7  # Stats.log contains data from various counters of the Suricata engine.
8  - stats:
9      enabled: no
10     filename: stats.log
11     append: yes
12 # a line based alerts log similar to fast.log into syslog
13 - syslog:
14     enabled: no
15     # reported identity to syslog. If omitted the program name (usually
16     # suricata) will be used.
17     facility: local5
18 # Logging configuration. This is not about logging IDS alerts/events, but
19 # output about what Suricata is doing, like startup messages, errors, etc.
20 logging:
21     # The default log level:
22     default-log-level: Notice
23     # Define your logging outputs.
24     outputs:
25     - console:
26         enabled: yes
27         level: Error
28     - file:
29         enabled: yes
30         level: Error
31         filename: suricata.log
32     - syslog:
33         enabled: yes
34         facility: local5
35         format: "[%i] <%d> -- "
36 rule-files:
37     - suricata.rules
38
39 ...

```

## .2 Résultat Test Suricata

Journal généré lors de l'exécution des fichiers de règles récupérées par le programme Python. Pour chaque règle non fonctionnelle sur notre version de Suricata, une première ligne décrit l'erreur qui s'est produite et une seconde ligne donne la règle qui a causé l'erreur ainsi que son fichier et sa ligne source.

```
1 [2449438 - Suricata-Main] 2024-09-26 09:14:21 Error: detect-parse: "$HOME_NET" is
   not a valid direction modifier, ">" and "<" are supported.
2 [2449438 - Suricata-Main] 2024-09-26 09:14:21 Error: detect: error parsing
   signature "alert tcp $EXTERNAL_NET $HTTP_PORTS-> $HOME_NET any (msg: "MISP
   event:https://10.0.37.3/events/view/430";flow:established,to_client; content:"
   // stop for sometime if needed"; classtype:trojan-activity;reference:url,https
   ://10.0.37.3/events/view/430; rev:1; sid:10924794;)" from file rules/misp/misp
   .rules at line 69127
3 [2449438 - Suricata-Main] 2024-09-26 09:14:21 Error: detect-parse: bad option
   value formatting (possible missing semicolon) for keyword content: '/4419
   d5733u64c6488819223d5belafc887d0.html_'
4 [2449438 - Suricata-Main] 2024-09-26 09:14:21 Error: detect: error parsing
   signature "alert http $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg: "MISP
   event:https://10.0.37.3/events/view/714"; flow:to_server,established; http.
   header; content:"notifyhubss.net"; fast_pattern; nocase; http.uri; content:"
   /4419d5733u64c6488819223d5belafc887d0.html_;!0epYZ6Q!6a0JnRhFyhQfncK-
   LhrwBwz4hNh8MS_qxKKgAFDPEKMOC0gQxgYx7CnrvR6bIX3SIR-
   AMI9IhS0nwyds8fRelwKk4eFibA$"; nocase; tag:session,600,seconds; classtype:
   trojan-activity; sid:11031604; rev:1; priority:2; reference:url,https
   ://10.0.37.3/events/view/714;)" from file rules/misp/misp.rules at line 76469
5 [2449438 - Suricata-Main] 2024-09-26 09:14:23 Error: detect-sid: invalid
   character as arg to sid keyword
6 [2449438 - Suricata-Main] 2024-09-26 09:14:23 Error: detect: error parsing
   signature "alert tcp any any -> any any (msg: "MISP event:https://10.0.37.3/
   events/view/1278"; sid:XX; rev:1; flow:established,to_server; content:"Accept
   |3a 20 2a 2f 2a|"; nocase; content:"HX1|3a|"; distance:0; within:6;
   fast_pattern; content:"HX2|3a|"; nocase; distance:0; content:"HX3|3a|"; nocase
   ; distance:0; content:"HX4|3a|"; nocase; distance:0; classtype:trojan-activity
   ; priority:X; sid:11468464;reference:url,https://10.0.37.3/events/view/1278;)"
   from file rules/misp/misp.rules at line 106142
7 [2449438 - Suricata-Main] 2024-09-26 09:14:28 Error: suricata: Loading signatures
   failed.
8 ...
```

## .3 Fonctions MISP

```
9 # Telechargement des events depuis le "feeds" mis en parametre dans un dossier
   local
10 def download_files(feed_url):
11     if not os.path.exists(directory):
12         os.makedirs(directory)
13     os.system(f"wget -r -np -nd -A json -P ./events/ {feed_url}")

14 # Reprend tous les evenements telecharges precedemment et les integre dans l'
   instance locale de MISP.
15 def import_events():
16     events_files = glob.glob("./events/*.json")
17     events = []
18     for file in events_files:
19         if (file != "manifest.json"):
20             with open(file, 'r') as f:
21                 file = json.load(f)
22             try:
23                 misp.add_event(file)
24                 print(f'Successfully imported to misp {file}')
25             except Exception as e:
26                 print(f'Error importing {file}: {e}')
27     return events
```

```

28 # Creer un evenement Pouvant etre exporter en regles suricata
29 def create_event(_distribution, _threat_level, _analysis, _info, _attribute,
30                 _tags):
31     event = MISPEvent()
32
33     # Votre choix de distribution:
34     # 0 - This Organization only
35     # 1 - This community only
36     # 2 - Connected communities
37     # 3 - All communities
38     event.distribution = _distribution
39
40     # niveau de menace:
41     # 1 - High
42     # 2 - Medium
43     # 3 - Low
44     # 4 - Undefined
45     event.threat_level_id = _threat_level
46
47     # Represente le niveau de maturite de l'analyse:
48     # 0 - Initial
49     # 1 - Ongoing
50     # 2 - Complete
51     event.analysis = _analysis
52     event.info = _info
53     event.date = datetime.now().date().isoformat() # Today's date
54
55     # Ajout d'un attribut (IOC) que Suricata peut utiliser, tel qu'un domaine
56     attr = MISPAtribute()
57     attr.category = _attribute["category"]
58     attr.type = _attribute["type"]
59     attr.value = _attribute["value"]
60     attr.to_ids = True # Mark as to_ids to be used for IDS export
61     attr.comment = _attribute["comment"]
62
63     event.add_attribute(**attr)
64     # Ajout de "tags"
65     for tag in _tags:
66         event.add_tag(tag)
67
68     # Telecharge l'evenement sur l'instance local de MISP
69     response = misp.add_event(event)

```

# Bibliographie

- [1] **A study of methodologies used in intrusion detection and prevention systems (IDPS)** (2012). *David Mudzingwa, Rajeev Agrawal*. source : <https://ieeexplore.ieee.org/document/6197080>.
- [2] **Decaying Indicators of Compromise** (2018). *Andras Iklody, Gérard Wagener, Alexandre Du-launoy, Sami Mokaddem*. source : [https://www.researchgate.net/publication/324104555\\_Decaying\\_Indicators\\_of\\_Compromise](https://www.researchgate.net/publication/324104555_Decaying_Indicators_of_Compromise).
- [3] **Guide to Intrusion Detection and Prevention Systems (IDPS)** (2007). *Karen Scarfone, Peter Mell*. source : <https://csrc.nist.gov/pubs/sp/800/94/final>.
- [4] **Indicators of Compromise as an Instrument for Threat Intelligence** (2015). *Denis Makrushin*. source : [https://www.researchgate.net/publication/349211330\\_Indicators\\_of\\_Compromise\\_as\\_an\\_Instrument\\_for\\_Threat\\_Intelligence](https://www.researchgate.net/publication/349211330_Indicators_of_Compromise_as_an_Instrument_for_Threat_Intelligence).
- [5] **Intrusion detection : a brief history and overview** (2002). *R.A. Kemmerer, G. Vigna*. source : <https://ieeexplore.ieee.org/document/1012428>.
- [6] **Intrusion Detection Systems** (2001). *Rebecca Bace, Peter Mell*. source : <https://www.govinfo.gov/content/pkg/GOVPUB-C13-PURL-LPS72073/pdf/GOVPUB-C13-PURL-LPS72073.pdf>.
- [7] **Intrusion Detection Systems : A Survey and Taxonomy** (2000). *Stefan Axelson*. source : [https://www.researchgate.net/publication/2597023\\_Intrusion\\_Detection\\_Systems\\_A\\_Survey\\_and\\_Taxonomy](https://www.researchgate.net/publication/2597023_Intrusion_Detection_Systems_A_Survey_and_Taxonomy).
- [8] **Panorama de la cybermenace 2023** (2023). *Agence nationale de la sécurité des systèmes d'information (ANSSI)*. source : <https://www.cert.ssi.gouv.fr/uploads/CERTFR-2024-CTI-001.pdf>.
- [9] **Prestataires de détection des incidents de sécurité - Référentiel d'exigences** (2017). *Agence nationale de la sécurité des systèmes d'information (ANSSI)*. source : [https://cyber.gouv.fr/sites/default/files/2022-10/pdis\\_referentiel\\_v2.0%5B1%5D.pdf](https://cyber.gouv.fr/sites/default/files/2022-10/pdis_referentiel_v2.0%5B1%5D.pdf).



