Titre: Title:	Ontologie de playbook pour la réponse à incident
Auteur: Author:	Kévin Abou Ahmed
Date:	2023
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Abou Ahmed, K. (2023). Ontologie de playbook pour la réponse à incident [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/53394/

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ontologie de playbook pour la réponse à incident

KÉVIN ABOU AHMED

Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie informatique

Mai 2023

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

Ontologie de playbook pour la réponse à incident

présenté par **Kévin ABOU AHMED**

en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

Amal ZOUAQ, présidente
Frédéric CUPPENS, membre et directeur de recherche
Nora BOULAHIA CUPPENS, membre et codirectrice de recherche
Alejandro QUINTERO, membre

DÉDICACE

A toutes les personnes qui m'ont soutenu

REMERCIEMENTS

Je souhaite tout d'abord remercier mon directeur de recherche Frédéric Cuppens ainsi que ma co-directrice Nora Boulahia-Cuppens pour m'avoir donné l'opportunité de réaliser cette recherche et pour m'avoir apporté leur aide tout au long de celle-ci.

Je suis aussi reconnaissant envers l'associé de recherche Jean-Yves Ouattara pour sa disponibilité et ses précieux conseils qui m'ont été d'une grande aide.

Je veux aussi remercier mes parents et mon petit frère qui ont été ma source de motivation et qui m'ont toujours soutenu dans la réalisation de cette recherche.

Enfin, je souhaite remercier tous mes collègues du laboratoire pour les bons moments que l'on a passés ensemble.

Cette recherche n'aurait probablement pas vu le jour sans le soutien de toutes ces personnes, et je souhaite leur exprimer ma profonde reconnaissance.

RÉSUMÉ

Durant ces dernières années, le monde de l'informatique n'a cessé de se développer jusqu'à devenir omniprésent dans notre société. Aujourd'hui, les entreprises sont dépendantes de leur environnement informatique. Leurs données et leurs activités passent par leurs systèmes d'information. Cela en fait une cible de choix pour les attaquants et ce n'est pas un hasard si le nombre d'attaques n'a cessé d'augmenter au cours des années.

Pour faire face à cela, les entreprises déploient divers mécanismes de défense. L'objectif est de protéger leurs actifs, de détecter de potentielles brèches de sécurité et d'y répondre le plus efficacement possible. Cependant, la démultiplication des attaques ainsi que leur complexification posent d'énormes problèmes aux méthodes de défense traditionnelles. Les capacités de réponse aux incidents de sécurité ne sont plus suffisantes. En effet, les analystes peinent de plus en plus à traiter manuellement le volume important d'alertes qu'ils reçoivent. Il est nécessaire de trouver de nouvelles alternatives afin de faire face à ces nouveaux défis que rencontre aujourd'hui le monde de la sécurité informatique.

C'est ainsi que de nouveaux concepts ont vu le jour. Celui de SOAR (Security Orchestration Automation & Response) notamment, dont l'arrivée récente dans le monde de la sécurité informatique est perçu comme opportune.

Ce mémoire de maîtrise présente des travaux basés sur l'utilisation des ontologies afin de permettre l'automatisation de la réponse à incident à l'aide de ce nouvel outil qu'est le SOAR.

Dans ce mémoire, nous proposons un format de spécification pour les playbooks de réponse à incident. Ce format, basé sur les ontologies, a pour but d'être un outil permettant de décrire formellement les processus de réponse à incident. L'objectif est de faciliter, améliorer et uniformiser la réponse à incident réalisée par les analystes.

Nous avons ensuite étudié les possibilités d'associer ce modèle ontologique aux SOAR. Nous proposons aussi dans ce mémoire une architecture permettant d'interconnecter notre base de connaissances ontologique à une telle plateforme.

Afin de tester notre solution, nous avons déployé un laboratoire de test réaliste et qui est inspiré des SOC (Security Operations Center). Cet environnement de test nous a permis de générer des alertes de sécurité en se basant sur deux scénarios. Nous avons ensuite vérifié le bon

fonctionnement de notre modèle ontologique et de son association avec le SOAR que nous avons déployé.

ABSTRACT

During the last few years, the world of information technology has kept growing until it became omnipresent in our society. Today, companies depend on their IT environment. Their data and their activities pass through their information systems. This makes them a prime target for attackers and it is not a coincidence that the number of attacks has been increasing over the years.

To deal with this, companies are deploying various defense mechanism. The objective is to protect their assets, to detect potential security breaches and to respond as efficiently as possible. However, the multiplication of attacks and their increasing complexity present news challenges to traditional defense methods. Security incident response capabilities are no longer sufficient. Indeed, analysts increasingly struggles to manually process the large volume of alerts they receive. It is necessary to find new alternatives in order to solve these new challenges that the IT security world is facing today.

This is why new concepts were born. SOAR (Security Orchestration Automation & Response) in particular, whose recent arrival in the world of IT security is seen as relevant.

This master's thesis presents work based on the use of ontologies to enable the automation of incident response using SOAR.

In this paper, we propose a specification format for incident response playbooks. This format, based on ontologies, aims to be a tool to formally describe incident response processes. The objective is to improve and standardize the way incident response is performed by analysts.

We then studied the possibilities to associate this ontological model with SOAR. We propose an architecture allowing to interconnect our ontological knowledge base to this kind of platform. In order to test our solution, we have deployed a realistic testbed inspired by Security Operations Center. This testbed allowed us to generate security alerts based on two scenarios. We then verified the proper functioning of our ontological model and its association with the SOAR we deployed.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

ABox	Assertional Knowledge
API	Application Programming Interface
ATOM	Abstractions Translation Ontology Method
CSIRT	Computer Security Incident Response Team
DDOS	Distributed Denial Of Service
IDS	Intrusion Detection System
ISBN	International Standard Book Number
MTTR	Mean Time To Respond
NIST	National Institute of Standards and Technology
NLP	Natural Language Processing
OVM	Ontology for Vulnerability Management
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SAMOD	Simplified Agile Methodology for Ontology Development
SI	Système d'information
SIEM	Security Information and Event Management
SIRP	Security Incident Response Platform
SOA	Security Orchestration and Automation
SOAR	Security Orchestration Automation Response
SOC	Security Operation Center
SPARQL	SPARQL Protocol And RDF Query Language
STIX	Structued Threat Information Expression

SWRL	Semantic Web Rule Language
TBox	Terminological Knowledge
TIP	Threat Intelligence Platform
UCO	Unified Cyber Ontology
URI	Uniform Resource Identifier
VDO	Vulnerability Database Ontology

CHAPITRE 1 INTRODUCTION

Durant ces dernières années, les systèmes informatiques ont continué de se développer et ont aujourd'hui pris une place importante dans notre société. Nous les utilisons quotidiennement que ce soit dans notre vie personnelle avec les réseaux sociaux par exemple ou dans notre vie professionnelle, en entreprise ou à l'université. Aujourd'hui, l'informatique est devenue incontournable et plus aucune entreprise ne peut s'en passer. Cette omniprésence de l'informatique s'est d'ailleurs accentuée avec l'avènement du télétravail à cause de la crise COVID. À cela s'ajoute aussi le fait que les systèmes informatiques sont de plus en plus complexes. Une entreprise aujourd'hui ne possède plus uniquement des ordinateurs, mais peut avoir dans son environnement informatique des outils déployés dans l'infonuagique, des terminaux mobiles et des objets connectés (IoT). Tout cela entraîne une surface d'attaque beaucoup plus importante et qu'il faut protéger.

Pour sécuriser son environnement informatique, une entreprise va mettre en place plusieurs outils de sécurité pour monitorer son réseau et son système d'information (SI). Parmi cet ensemble d'outils, il y a des systèmes de détection d'intrusion (IDS) qui ont pour but de détecter des attaques sur le système informatique de l'entreprise. À cela s'ajoute aussi les SIEM qui vont permettre d'agréger l'ensemble des logs de l'entreprise et de réaliser la détection grâce à des règles de corrélation. Les détections réalisées donnent lieu à des alertes de sécurité qu'il faut investiguer. Ce travail est réalisé par l'équipe de réponse aux incidents (CSIRT). Jusqu'à récemment, les entreprises utilisaient des moyens manuels pour répondre à ces alertes de sécurité. Cependant, avec la complexification des environnements informatiques, une entreprise possède aujourd'hui une trop grande surface d'attaque à surveiller. Cela donne lieu à un nombre d'alertes de sécurité toujours plus grand et qui dépasse souvent la capacité d'analyse et d'investigation qu'une entreprise possède. De plus, le manque de main-d'œuvre dans le domaine de la sécurité informatique n'aide pas les entreprises à résoudre ce problème. Pour décrire cette incapacité à répondre à un volume d'alertes toujours plus important, le concept *Alert fatigue* a été créé. Dans le contexte de la réponse à incident, ce terme signifie qu'à force de recevoir des alertes de sécurité, les analystes sont devenues insensibles aux alertes. Il en résulte un temps de réponse à incident plus long ou parfois même l'absence de réponse [1].

1.1 L'automatisation de la réponse

Les problèmes évoqués ci-dessus sont d'une grande importance et il n'est pas surprenant de voir que plusieurs travaux ont été réalisés afin d'épauler les analystes dans leurs activités de réponse aux incidents. Parmi les dernières avancées dans ce domaine, une partie porte sur l'automatisation des actions prises par les analystes dans leur activité de réponse à une alerte. Une des possibilités de faire de l'automatisation de la réponse est au travers d'un outil de sécurité : le SOAR (Security Orchestration, Automation and Response). Le SOAR est un outil qui va intervenir lors de la phase de réponse à incident. Il va servir de plateforme de gestion des incidents en ingérant les données des senseurs et du SIEM. Il va aussi permettre d'aider l'analyste grâce à sa capacité d'automatisation d'actions de réponse à incident. La naissance de cet acronyme est attribuée à la société de conseil et de recherche « Gartner » qui définit pour la première fois le concept de SOAR en 2017. Bien qu'il soit récent, cet outil de sécurité a rapidement gagné en popularité au cours des dernières années. Son utilisation dans le monde de l'entreprise est en augmentation constante. Dans leur étude sur l'état du marché des SOAR parue en 2020, Gartner définit le SOAR comme étant le futur volet de contrôle des SOC (Security Operations Center) [2]. Cette même étude révèle que la demande pour ces solutions de sécurité est en augmentation. Cette affirmation du Gartner est confirmée par un sondage réalisé en 2020 par l'entreprise Palo Alto qui estime que 24% des répondants prévoyaient d'implémenter un SOAR dans les 12 prochains mois et que 33% des répondants allaient augmenter le budget alloué au SOAR [3]. Cet outil s'impose petit à petit comme un incontournable dans le domaine de la réponse à incident et pour les entreprises possédant une certaine maturité dans leur posture de sécurité.

1.2 Élément de la problématique

Bien que le concept de SOAR soit récent, cet outil de sécurité jouit aujourd'hui d'une importante popularité dans le monde de l'entreprise. Rares sont les organisations avec une maturité suffisante n'ayant pas installé un SOAR. Cependant, bien que populaires, les résultats de l'utilisation de cet outil sont loin de ceux escomptés. C'est un outil certes puissant, mais aussi d'une extrême complexité. L'adoption d'un SOAR nécessite des délais d'intégration importants, allant de six à neuf mois selon les estimations de Gartner. À cela s'ajoute la nécessité d'avoir un personnel

qualifié et disponible pour réaliser un tel projet d'intégration. De plus, pour tirer profit d'un SOAR, celui-ci doit être en constante amélioration. Cela donne finalement un outil intéressant pour les grandes entreprises, mais qui reste bien loin du discours commercial si prometteur des éditeurs de SOAR. En fait, la réalité du terrain est toute autre et il est difficile d'obtenir un retour sur investissement dans les délais attendus. C'est par exemple ce que l'on peut lire du retour d'expérience sur l'utilisation des SOAR fait par la société Orange CyberDéfense [4].

De plus, très peu de recherche existe sur les SOAR. Cela en fait un domaine qui nécessite encore de nombreux travaux avant d'atteindre les résultats espérés. C'est une des conclusions des travaux de Kinuya et al. [5] dans leur article portant sur les futures directions de recherche en intelligence artificielle et en apprentissage machine dans le domaine des SOAR. D'après eux, il est nécessaire d'investiguer la possibilité de coupler les ontologies aux plateformes d'orchestration et d'automatisation de la réponse.

C'est donc dans ce cadre que s'inscrit notre recherche. La motivation de nos travaux est d'utiliser des ontologies afin de permettre à une entreprise de définir de façon plus formelle ses processus de sécurité et ensuite de faciliter l'automatisation de la réponse à incident.

1.3 Objectif de recherche

L'objectif de cette recherche est de vérifier la capacité d'une ontologie à contenir les informations servant à définir un playbook de réponse à incident et ensuite, d'analyser les possibilités d'intégration de l'ontologie dans un SOAR.

Nous allons découper cet objectif général en plusieurs sous-objectifs de recherche. Nous allons donc définir les objectifs de recherche suivants :

1. Analyser l'utilisation des SOAR dans les processus de réponse à incident. L'objectif est de réaliser un état de l'art des SOAR et de leur utilisation ainsi que des limites de cette solution.
2. Modéliser des playbooks de réponse à incident à l'aide d'une ontologie. L'ontologie doit être en mesure de représenter l'ordre des actions à exécuter ainsi que les outils qui permettent de réaliser ces étapes. De plus, l'ontologie doit permettre d'apporter une plus grande expressivité du playbook pour permettre à un analyste de comprendre son objectif.

L’ontologie va enrichir le playbook avec des informations utiles pour l’analyste qui doit réaliser la réponse à incident.

3. Évaluer l’architecture proposée dans le but d’intégrer un playbook sous format ontologique au SOAR
4. Analyser si l’architecture proposée permet d’abstraire le playbook et de le rendre interopérable. Le but est d’évaluer la transférabilité du playbook basé sur une ontologie d’un SOAR à un autre.

1.4 Plan du mémoire

Ce mémoire présente les travaux selon l’ordre suivant. Le chapitre 2 présente notre état de l’art sur la réponse à incident et sur les SOAR. Ce chapitre établit aussi les notions de base des ontologies. Il contient une revue de la littérature des ontologies et de leur application en sécurité informatique.

Nous présentons ensuite dans le chapitre 3 une revue des SOAR existants ainsi que les limites que nous avons pu observer dans l’utilisation de ces outils. Cette partie inclut aussi la présentation de l’ontologie qui a été développée afin de modéliser des playbook de réponse à incident.

Dans le quatrième chapitre, nous présentons l’architecture qui permet la communication entre le SOAR et l’ontologie. Ce chapitre contiendra aussi un exemple d’utilisation avec le SOAR Splunk Phantom.

Le chapitre 5 présente l’analyse de nos travaux. Nous évaluons dans cette partie l’ontologie qui a été développée en vérifiant sa capacité à modéliser des playbooks et sa capacité à répondre à des questions de compétence. De plus, nous analysons aussi l’architecture proposée afin de permettre la communication avec le SOAR.

Pour finir, le chapitre 6 expose les conclusions sur nos travaux. Ce chapitre traite des limites des travaux ainsi que des futurs développements possibles.

CHAPITRE 2 RÉPONSE À INCIDENT ET ONTOLOGIES

Ce chapitre présente la thématique de la réponse à incident en cybersécurité, ainsi que l'utilisation des ontologies dans ce domaine. La première partie est consacrée à la façon dont est mise en place la réponse à incident dans une entreprise. Cette partie aborde aussi le sujet de l'automatisation des étapes de la réponse à incident. Pour finir, la deuxième partie traite des ontologies et de leur application dans le domaine de la réponse à incident.

2.1 Réponse à incident et SOAR

Avant de pouvoir définir la réponse à incident, il faut d'abord s'accorder sur ce qu'est un incident en sécurité informatique. Dans son guide de gestion des incidents de sécurité informatique, le NIST (National Institute of Standards and Technology) définit un incident de sécurité comme étant « une violation ou une menace imminente de violation des politiques et des standards de sécurité informatique » [6]. Par exemple, un attaquant obtient l'accès à des données confidentielles et menace de les rendre publiques. De plus, le Gartner ajoute à cette définition qu'un incident de sécurité a des conséquences sur la confidentialité, l'intégrité ou la disponibilité des systèmes et réseaux [7]. Un incident de sécurité est donc un événement ayant lieu sur le système d'information d'une entreprise et qui entraîne des conséquences négatives sur celui-ci. Il faut le distinguer d'une alerte, qui est une détection d'un événement qui a lieu sur le SI et qui peut ne pas être malveillant [7]. Après investigation, une alerte peut devenir un incident de sécurité ou au contraire n'être qu'une fausse alerte. Afin de répondre aux alertes et aux incidents de sécurité, les organisations développent des processus de réponse à incident.

La réponse à incident peut se définir comme étant : « le processus de détecter les événements de sécurité qui affectent le SI et d'y répondre en réalisant les étapes appropriées pour contenir l'attaque et récupérer de celle-ci » [8]. Le but de la réponse à incident est donc d'investiguer les événements de sécurité, et dans le cas d'une attaque, de minimiser les conséquences négatives d'un incident de sécurité et les impacts qu'il peut avoir sur le SI de l'organisation.

La division chargée de réaliser les opérations de sécurité d'une organisation, dont celle de réponse à incident, est le SOC (Security Operations Center). Ce terme englobe tout ce qui se rapporte à la sécurité informatique, donc les personnes, les processus et les outils de sécurité. L'objectif du SOC est de monitorer, de détecter et de répondre aux incidents de sécurité. Un SOC est construit selon

les besoins de sécurité d'une entreprise, par conséquent sa capacité et les outils dont dispose cette division dépendent de la maturité de l'organisation. Pour réaliser les diverses fonctions et opérations de sécurité, un SOC va développer ces processus de réponse à incident et s'armer de différents outils, dont un SOAR (Security Orchestration Automation Response) si la maturité de l'entreprise le permet.

Les processus de réponse à incident ont un cycle de vie. Dans la littérature, on peut trouver plusieurs travaux modélisant les cycles de vie de la réponse à incident dans une entreprise. Le plus connu est sans aucun doute celui proposé par le NIST dans son guide de gestion des incidents de sécurité informatique [6].

Selon le NIST, le cycle de vie de la réponse à incident est constitué des étapes suivantes :

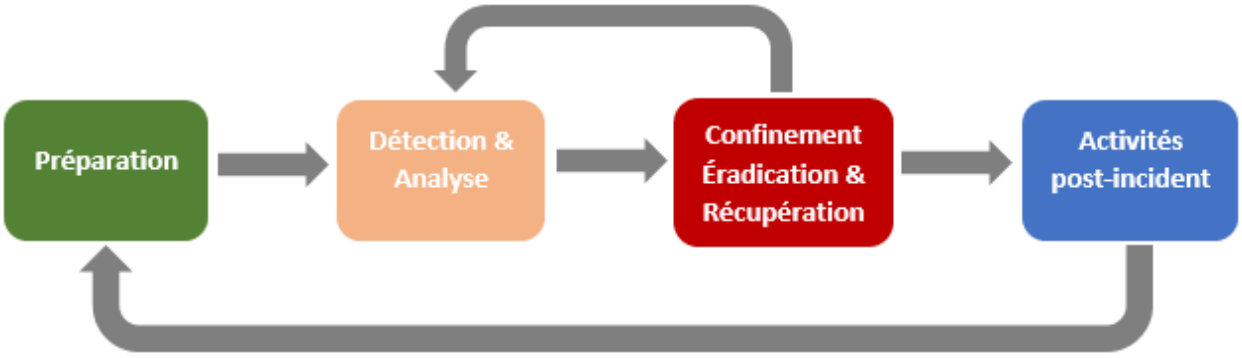

Figure 2.1 : Cycle de vie de la réponse à incident du NIST

- Préparation : Il s'agit de l'étape où l'entreprise va mettre en place tout ce dont elle a besoin pour pouvoir répondre à une attaque informatique. Le NIST découpe cette phase en quatre sous-parties.
    - Le déploiement des outils de sécurité nécessaires pour identifier et répondre aux menaces.
    - La création des procédures de réponse à incident ainsi que la préparation des outils de communication qui seront utilisés pendant la gestion de l'incident.

  - La préparation de ressources permettant l'analyse des incidents. Cela passe par la documentation des applications de l'entreprise, des protocoles de réseaux, systèmes d'exploitation, etc.
  - La création de copie propre de systèmes d'exploitation et d'applications. Cela permettra de les utiliser lors des étapes de remédiation et d'éradication de la menace.

- Détection et Analyse : Cette étape correspond à la phase de détection de l'incident. Cette détection peut se faire de plusieurs manières, de façon automatisée par un outil de sécurité, par des règles de corrélation dans le SIEM, voire manuellement. Chaque règle de détection va correspondre à un « *cas d'usage* ». Par exemple, une règle de corrélation qui va détecter toute modification d'une tâche programmée dans Windows est un *cas d'usage*. L'objectif de celui-ci va être de détecter des maliciels qui vont s'implanter dans le système et devenir persistants. Cette étape de détection est suivie de l'analyse de l'événement de sécurité qui a été détecté. L'analyse de l'incident est réalisée par l'équipe de réponse aux incidents : CSIRT (Computer Security Incident Response Team). Les analystes vont investiguer l'alerte selon les *cas d'usages* et selon les procédures établies lors de la phase de préparation. Si l'analyse de l'événement confirme qu'il ne s'agit pas d'une erreur de détection, alors la réponse à incident passe à l'étape suivante.
- Confinement, Éradication et Récupération : Dans le cas d'un événement de sécurité confirmé par l'analyse faite par l'équipe CSIRT, l'étape suivante est de mettre en place la stratégie de gestion de crise. Le but est d'éviter que l'attaque ne se propage vers d'autres systèmes et de limiter les dégâts qu'elle peut occasionner. Cette étape englobe aussi la collecte des informations afin d'entamer des procédures judiciaires.
  - Le confinement permet de limiter la propagation de l'attaque. Cela consiste comme son nom l'indique en la mise en quarantaine des éléments touchés par l'attaque.
  - L'éradication consiste en l'élimination de la menace. Cela peut être par exemple la suppression d'un maliciel ou la mise à jour d'un serveur afin d'éliminer une vulnérabilité.
  - Enfin, la récupération consiste en la restauration du SI dans son état de fonctionnement normal.

- Activités post-incident : Une fois la menace éliminée, il faut réaliser une analyse de la réponse à incident. D'après le NIST, c'est l'une des étapes les plus importantes, mais aussi celle qui est la plus oubliée. Le NIST définit cette étape en deux mots : « Apprendre et Améliorer ». L'objectif est d'analyser ce qui s'est passé et comment l'attaque a pu se produire. De plus, il faut évaluer si la procédure définie lors de l'étape de préparation était adaptée et ce qu'il faudrait changer. C'est aussi l'opportunité d'analyser si le SOC contient suffisamment de ressources (humaines comme matérielles) pour réaliser une réponse à incident efficace.

Le cycle de vie de la réponse à incident du NIST est certainement le plus connu. Il détaille précisément les étapes à suivre et comment s'y préparer. De plus, il contient différents scénarios afin de montrer comment il peut être utilisé. Cependant, un sujet qui n'est pas évoqué dans le guide de gestion des incidents du NIST est la possibilité d'automatiser des éléments de cette réponse à incident. C'est un élément important que nous souhaitons aborder.

L'automatisation peut intervenir à différentes étapes de la réponse à incident. Il est important pour une entreprise d'analyser sa capacité d'automatisation lors de l'étape de préparation, car elle peut s'avérer d'une grande aide pour les étapes suivantes. L'automatisation de la réponse peut jouer un rôle important dans la phase de détection des menaces grâce à l'automatisation d'action de « Threat Hunting » [9]. Cela peut permettre de détecter des menaces grâce à une recherche périodique d'indicateurs de compromissions présents dans le réseau de l'entreprise. À cela s'ajoute la possibilité d'automatiser les tâches de renseignement sur les menaces lors de l'analyse d'un incident de sécurité. L'apport ici est important, car les actions de renseignement sur les menaces sont des tâches qui sont à la base de l'analyse d'une alerte de sécurité. Leur automatisation permet d'accélérer de façon conséquente la réponse à incident [9]. Ensuite, l'automatisation de la réponse peut aussi intervenir lors de la phase de confinement, d'éradication et de récupération. Il est possible de réaliser des actions automatisées sur les différents outils de sécurité afin de bloquer plus rapidement une adresse IP ou afin de supprimer des fichiers avec un certain hash. C'est un des éléments qui a fait gagner en popularité le concept de SOAR. Enfin, concernant l'étape de post-incident, il existe pour le moment peu d'outils et de capacité d'automatisation. C'est une des conclusions obtenues par Palo Alto lors de leur analyse du marché sur l'automatisation de la réponse à incident et les SOAR [3].

2.1.1 Playbook

La réponse à incident suit un cycle de vie comme celui du NIST présenté ci-dessus, mais elle suit aussi un ensemble de procédures qui vont être définies par le SOC d'une entreprise. Parmi cet ensemble de procédures, il y a ce que l'on peut appeler un « *playbook* ». Un *playbook* est un document qui va expliciter les actions à prendre en réponse à un type d'alerte. Une entreprise peut avoir autant de *playbooks* différents que de cas d'usages. Le consortium Oasis définit un *playbook* de réponse à incident comme étant la documentation de l'ensemble des étapes de détection, d'investigation, de prévention et de remédiation [10]. C'est un élément essentiel dans la réponse à incident, car des processus bien documentés permettent l'uniformisation de la réponse, ainsi qu'une plus grande rapidité dans l'exécution de celle-ci. Il va être utilisé au quotidien par les analystes de l'équipe CSIRT. C'est le premier ensemble d'actions qui doit être exécuté après la détection d'un potentiel incident de sécurité. Bien qu'il existe plusieurs travaux sur le concept de playbook, il n'y a aujourd'hui, à notre connaissance, aucun standard. Chaque organisation a sa manière de décrire les playbooks et ils peuvent prendre la forme de document textuel, de schéma, ou même de fichier JSON ou YAML. De même, d'après l'étude de l'entreprise Palo Alto Networks, il n'y a pas non plus de moyens efficaces afin de partager des playbooks de réponse à incident entre organisations [3].

Notre recherche porte sur la réponse à incident et les SOAR et nous nous sommes intéressés aux différents travaux sur la formalisation des *playbooks* de réponse à incident.

Durant notre revue de littérature, nous avons pu trouver cinq travaux qui traitent de la formalisation des playbook de réponse à incident. À cela s'ajoute l'article de recherche de Schlette et al. [5]. Il étudie les formats existants de playbook de réponse à incident. Il définit des notions en lien avec la représentation des playbooks et de workflow. Notre travail s'est basé en partie sur cet article et sur les notions qui ont été définies. Nous présentons les travaux que nous venons d'évoqués ci-dessous.

Il y a d'abord les travaux réalisés par le groupe IACD sur l'automatisation de la réponse à incident [11]. Leur objectif est de faciliter l'automatisation en fournissant un format de playbook qui va permettre à une entreprise d'implémenter ses procédures dans un SOAR. Ils définissent trois niveaux d'abstraction qui sont : les playbooks, les workflows et les instances locales. Dans cette

partie, nous nous intéresserons au concept de playbook. Un playbook est défini comme étant « un ensemble d'étapes qui permettent à une organisation de répondre aux exigences spécifiées dans leurs politiques de sécurité » [12]. De plus, ils ajoutent que « les playbook comblent le fossé entre les politiques et procédures d'une organisation et l'automatisation et l'orchestration de la sécurité » [11]. Bien qu'intéressants, nous décelons certaines limites à ces travaux. Comme le fait remarquer Schlette et al. [5], la spécification reste relativement informelle. Un playbook IACD peut être composé d'une multitude d'actions par étape et n'indique pas formellement qu'elle est la prochaine étape à réaliser. Cela a pour conséquence que le choix de l'action à effectuer reste dépendant de l'analyste. On se retrouve donc avec une spécification de playbook qui ne permet pas d'uniformiser la réponse prise par les analystes. Cela pose un problème pour une entreprise qui souhaite mettre en place de l'automatisation, car le playbook IACD ne sera pas suffisamment précis pour pouvoir être implémenté dans le SOAR. Un travail devra être réalisé sur le playbook afin de définir de façon plus formelle les actions à exécuter. De plus, la spécification reste relativement peu précise et la seule dimension observée est l'ordre des actions. Par exemple, le playbook ne contient pas d'informations sur les outils qui peuvent permettre à un analyste de réaliser ces actions. De plus, les concepts de workflows et d'instances locales et leur exécution au niveau de la machine restent non spécifiés par le groupe IACD. C'est une des conclusions des travaux de Schlette et al. [5] sur ces travaux et nous partageons leur avis.

CACAO (Collaborative Automated Course Of Action) [10] est un autre format de playbook que nous avons étudié. Ce travail a été réalisé par le consortium OASIS. La spécification définit un moyen de formaliser des *playbooks* et la manière dont ils peuvent être créés, documentés et partagés. L'objectif de ce travail est de permettre l'automatisation au travers d'un langage pour la gestion de technologies de cyberdéfense et l'exécution de commande sur ces mêmes outils. Le nom de ce langage est OpenC2 et est aussi une réalisation du groupe OASIS. Un playbook CACAO est enregistré au format JSON. Il contient des informations sur l'ordre des actions à exécuter et sur les commandes qui le permettent. De plus, le format JSON contient plusieurs champs qui permettent de rajouter des informations. Le format est clairement défini et contient un exemple de playbook. Il n'y a pas, à notre connaissance, de preuve de concept sur l'exécution du playbook, néanmoins les travaux sont encore en cours. Cependant, étant donné la direction choisie à savoir l'automatisation au travers du langage Openc2, ce projet s'éloigne de nos objectifs de recherche, mais l'étude de celui-ci reste intéressante et nous a aidés dans la création de notre ontologie.

En plus des deux formats précédents, il existe d'autres travaux qui ont été réalisés sur la même thématique. Premièrement, le format Collaborative Open Playbook Standard (COPS) décrit un playbook de réponse à incident au format YAML. C'est devenu un format propriétaire réutilisé par le SOAR de l'entreprise Palo Alto Networks. Malheureusement, il n'existe finalement que peu d'indications sur ce format et sur sa réutilisation. De plus, à notre connaissance, il n'existe pas d'autre source d'informations qu'un répertoire GitHub qui ne contient qu'un seul exemple de la structure du fichier YAML. Bien que le travail soit intéressant, il ne nous est pas possible de le reprendre. Cependant, il reste utile et il nous aura servi dans nos travaux car c'est un exemple de représentation de playbook et de workflow. Nous avons donc pu nous inspirer des informations que nous avons trouvées sur ces travaux pour réaliser notre propre modèle.

Pour finir, le dernier travail lié à la formalisation des playbooks que nous avons recensé est la matrice RE&CT [13]. Ce travail s'inspire de la matrice du framework MITRE ATT&CK qui est une base de connaissance de techniques et tactiques d'attaques. Cependant, la matrice RE&CT oriente la thématique autour de la réponse à incident. Les colonnes de cette matrice présentent les différentes étapes de réponse à incident comme définies par le NIST (expliqué plus haut). Chacune des colonnes contient des actions liées à la réponse à incident. La matrice sert à titre informatif et permet à une entreprise de voir les actions qui existent pour une étape donnée. Par exemple, pour l'étape de « Préparation », il y a l'action « Obtenir le plan de l'architecture réseau » ou encore « S'assurer que l'on a la possibilité de bloquer un nom de domaine ». Dans le cas de la matrice RE&CT, un playbook est un ensemble d'actions pouvant être réalisées pour répondre à une alerte. Ce n'est donc pas réellement un playbook, mais plutôt des indications pouvant servir à élaborer des playbooks. Les actions de sécurité contiennent des informations sur les outils nécessaires à leur réalisation et nous nous sommes inspirés de cela dans nos travaux. Cependant, nous souhaitons définir de façon formelle un playbook et apporter plus de détails aux analystes concernant les actions à exécuter, ce qui n'est pas le cas pour la matrice RE&CT. Les travaux restent tout de même intéressants et nous avons notamment repris pour notre ontologie les actions présentes dans la matrice RE&CT.

Cet ensemble de spécifications de playbook nous a servi dans nos travaux même si l'objectif final reste différent.

Outre les travaux présentés ci-dessus, le concept de playbook est aussi présent dans un outil de sécurité : le SOAR.

2.1.2 SOAR

L'abréviation SOAR veut dire « Security Orchestration Automation and Response ». C'est le Gartner qui évoque ce concept pour la première fois en 2017. Voici leur définition : « Les SOAR sont les technologies qui prennent en entrée des données surveillées par les outils de sécurité et qui aident les organisations et les analystes dans le travail d'investigation des incidents de sécurité » [14]. Cette définition reste relativement vague, mais elle est une première description du concept de SOAR. Une définition plus complète est celle des travaux d'Islam et al. [15] : « Le SOAR permet de planifier, d'intégrer et de coordonner les activités des experts et des outils de sécurité pour produire et automatiser les actions requises en réponse à un incident de sécurité ». Cette définition complète celle du Gartner, car elle met en avant un élément essentiel du SOAR, à savoir la capacité d'automatiser des actions de réponse à un événement de sécurité informatique.

Dans leurs travaux, Islam et al. séparent le SOAR en deux blocs :

- Le bloc d'orchestration : il est défini comme étant l'élément qui permet au SOAR de se connecter à différents outils de sécurité. Ce bloc comprend aussi l'intégration des fonctionnalités des outils avec lesquelles il est interconnecté. Il doit donc permettre d'ingérer des informations, mais aussi de permettre la réutilisabilité de ces intégrations dans un but d'automatisation.
- Le bloc d'automatisation : cet élément comprend la codification sous un format compréhensible par la machine d'un playbook de réponse à incident. Cette codification porte le terme de « workflow ». Cette unité est connectée au bloc d'orchestration afin de réaliser automatiquement des actions prédéfinies.

Bien que ces deux unités soient les parties principales d'un SOAR, la définition fournie par Islam et al. reste incomplète. C'est pourquoi nous complétons cette représentation des SOAR en reprenant un élément du Gartner qui est la plateforme de gestion des incidents de sécurité [2]. En effet, cet élément du SOAR permet de documenter l'alerte et les actions prises pour y répondre.

C'est cet ensemble d'unités qui donne ses fonctionnalités aux SOAR. Islam et al. [15] définissent les fonctionnalités suivantes :

- La capacité d'agir comme un centre névralgique. Cela signifie que le SOAR va s'intégrer au sein d'un SOC. Il doit être en mesure de récupérer les données auprès des différents capteurs et du SIEM. Il doit pouvoir fournir les informations à l'analyste et lui apporter du contexte sur l'alerte. Cela doit permettre des investigations plus approfondies. De plus, il doit pouvoir réaliser des actions sur les outils avec lesquels il est interconnecté.
- Le SOAR va orchestrer les activités de sécurité dès la réception de l'alerte et jusqu'à sa résolution. Cela passe par l'enrichissement des alertes auprès des plateformes de renseignement de la menace, mais aussi par la capacité à réaliser du triage d'alerte. Il doit aussi être en mesure d'effectuer des actions de confinement et d'éradication. Cela passe par sa capacité à appliquer automatiquement des actions prédéfinies de réponse à incident. Il sert aussi de tableau de bord central et permet aux analystes de travailler collaborativement sur une même plateforme. L'objectif est d'aider l'analyste à répondre le plus rapidement et le plus efficacement possible à une alerte.

Le Gartner définit globalement les mêmes fonctionnalités, mais en les présentant autrement :

- Le SOAR permet de réaliser du triage d'incident.
- Le SOAR intervient dans l'automatisation de la réponse. On parle ici d'actions plus poussées de confinement et de remédiation. Le SOAR doit aussi permettre la gestion en un point des outils informatiques et de leur conformité.

C'est donc une plateforme qui combine les activités d'orchestration, d'automatisation, de suivi d'incident et de renseignement sur la menace, le tout en un point.

En fait, le SOAR est le résultat du regroupement de trois technologies de sécurité (voir figure 2.2).

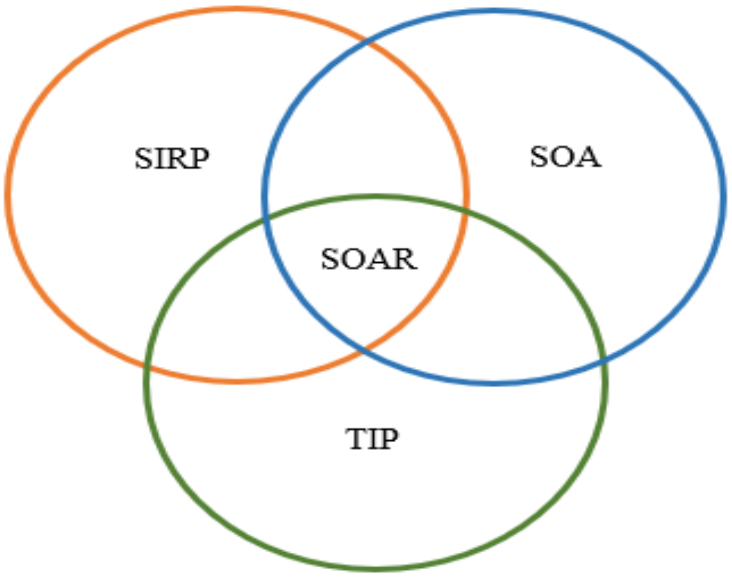

Figure 2.2 : La convergence de trois technologies

L'acronyme SIRP (Security Incident Response Platform) fait référence aux technologies qui permettent la gestion des incidents de sécurité. Ce sont des plateformes qui permettent la création de tickets contenant les informations de l'alerte. À l'instar des plateformes de suivi d'incident classique, les SIRP sont adaptés au contexte d'incident de sécurité informatique. La création des tickets se fait automatiquement en récupérant les informations depuis le SIEM. De plus, une telle plateforme est adaptée aux besoins de sécurité et possède des concepts comme celui d'artefact. Un artefact est un type de donnée. Dans le cas des plateformes SIRP on parle plutôt d'artefact de CTI (Cyber Threat Intelligence), donc d'artefact de renseignement des menaces informatiques. Cela peut être par exemple un hash, une adresse IP, un nom de domaine. Comme le fait remarquer Schlette et al. [5], c'est un élément important, car il permet une meilleure visualisation des alertes remontées par les outils de sécurité. De plus, ce concept permet l'implémentation de réponses automatisées en fonction du type de données que l'on souhaite enrichir.

L'acronyme SOA (Security Orchestration and Automation) est en fait une simplification des SOAR. C'est cette technologie qui contient le module d'orchestration des outils de sécurité et celui d'automatisation des actions de réponse à incident. Cette automatisation se fait à l'aide des « workflows ». D'après la définition du consortium IACD, un workflow est une transformation d'un playbook dans un format compréhensible par la machine [11]. Nous sommes d'accord avec cette définition, cependant elle reste relativement incomplète. Nous souhaitons la compléter avec

celle de Schlette et al. [5] qui vont plus loin dans la définition du concept de workflow en le définissant comme étant la combinaison de trois éléments :

- Un actionneur
- Une action
- Un artefact

Une étape d'un workflow est donc la réalisation d'une action par un actionneur sur un artefact. Plus précisément, dans le contexte de la réponse à incident, il s'agit d'un outil de sécurité (un pare-feu, une plateforme de renseignement de la menace, un système de détection d'intrusion, etc.) qui va réaliser sur un artefact, une action de réponse (bloquer une adresse IP, obtenir des informations sur un nom de domaine, analyser un fichier à l'aide de son hash, etc.).

C'est une définition du concept de workflow que nous trouvons pertinente et nous avons en partie construit notre ontologie basée sur ce triplet actionneur-action-artefact.

Pour finir, le dernier outil technologique qui a permis de donner naissance au SOAR est la « Plateforme de Renseignement sur les Menaces » qui porte l'acronyme « TIP » en anglais pour « Threat Intelligence Platform ». Le SOAR n'est pas à proprement parler une plateforme de renseignement sur la menace. Cependant, il doit être en mesure d'être interconnecté avec celle de l'entreprise. En effet, parmi les fonctionnalités d'un SOAR, l'enrichissement des alertes avec des données externes est l'une des plus importantes. Cet ensemble de fonctionnalités permettent aux SOAR d'apporter une plus-value lors de son déploiement dans un SOC.

2.1.3 Les avantages d'un SOAR

Dans leur article, Norem et al. [16] ont réalisé un sondage auprès d'analystes du SOC afin de leur demander quelles étaient les fonctionnalités les plus importantes d'un SOAR. Le sondage montre que « les deux plus importantes fonctionnalités que les analystes recherchent dans un SOAR sont la capacité à automatiser des tâches courantes et la fonctionnalité de playbook ». L'ordre des fonctionnalités classées du plus important au moins important est le suivant :

1. Playbook – Workflow
2. Automatisation de tâche courante

3. Gestion des alertes
4. Priorisation des alertes

Cet ensemble de technologies et de fonctionnalités fait du SOAR un outil qui prend de plus en plus d'importance au sein du SOC. Comme expliqué plus haut, les analystes sont inondées d'alertes de sécurité. Ils ne sont pas en mesure de tout traiter manuellement et il y a donc ce besoin d'un outil d'automatisation qui va venir épauler les analystes dans leurs tâches quotidiennes.

Le SOAR a pour but de répondre à ce besoin. Premièrement, du fait de la gestion unifiée en un seul point des différents outils de sécurité, le SOAR vient réduire la complexité du travail d'un analyste. Le travail se concentre davantage sur les technologies d'outils de sécurité à sa disposition plutôt que sur une instance d'un éditeur en particulier. L'objectif étant que le SOAR soit le seul outil qu'un analyste va utiliser sans non plus réduire le panel d'actions qu'il peut réaliser pour répondre à une alerte.

Deuxièmement, le SOAR, avec sa capacité d'automatisation de tâches, va accélérer le temps de réponse moyen à une alerte. L'entreprise Swimlane explique que les équipes qui utilisent de l'automatisation dans leur processus de réponse à des alertes avec un SOAR réduisent leur temps moyen pour répondre (MTTR) [17]. Plus précisément, Swimlane estime que leurs clients ont en moyenne réduit leur MTTR de 50%. Un SOAR permet donc une réponse plus rapide et cela a pour conséquence de réduire les dégâts et les risques qui seraient liés à une attaque.

Pour finir, l'automatisation de certaines actions permet de libérer du temps aux analystes pour des tâches plus importantes. Par exemple, comme le soulignent Islam et al. [15], le SOAR « Hexadite » a permis au total d'automatiser 800 000 heures de travail en deux ans. Ces tâches auraient dû initialement être réalisées par des analystes.

L'apport des SOAR permet donc aux analystes d'être plus productifs dans leur travail. Tout cela apporte une amélioration générale de la gestion du volume des alertes que peut recevoir un SOC. Cet outil a donc un réel impact et répond à des besoins précis. Bien qu'il soit de plus en plus utilisé, le SOAR reste néanmoins un concept jeune et qui est encore loin d'atteindre son état de maturité. Il y a encore beaucoup de problèmes et de risques liés à l'utilisation de cet outil.

2.1.4 Les limites des SOAR

Durant nos lectures et nos travaux, nous avons pu observer que l'utilisation des SOAR n'était finalement pas si avancée que cela. Certes, de multiples entreprises en possèdent un, mais leur utilisation reste associée à des cas et des actions très simples. C'est un constat mis en avant par Oliver Rochford, un des pionniers des SOAR, dans l'article qu'il a écrit en 2021 et intitulé « Pourquoi l'automatisation avec les SOAR échoue-t-elle ? » [18]. Oliver Rochford explique que la plupart des entreprises n'automatisaient que de petites tâches comme du renseignement sur la menace. C'est certes une plus-value, mais qui reste bien loin des attentes que les entreprises ont quand il s'agit d'investir dans les SOAR.

Nos travaux et lectures nous ont aussi permis de déceler de nombreuses limites qui peuvent expliquer pourquoi les SOAR reste à un stade relativement précoce dans leur utilisation.

Premièrement, l'intégration des outils à une plateforme SOAR peut s'avérer complexe. Même si les éditeurs proposent des intégrations par défaut, la liste est loin d'être exhaustive. Tous les outils de sécurité ne sont pas forcément intégrés. Le phénomène devient plus marquant lorsqu'une entreprise possède, pour un même concept, des instances d'outils provenant d'éditeurs différents. Cela crée une réelle problématique lors de l'adoption d'un SOAR, car il faut avant tout passer par une phase de développement. C'est un des problèmes qui est mentionné dans l'article d'Islam et al. [15] qui présente l'interopérabilité des outils avec le SOAR comme étant un défi important.

Deuxièmement, le SOAR est un outil peu flexible. Nous avons pu remarquer lors de nos discussions avec des experts dans le domaine qu'il était difficile de modifier cet outil pour l'adapter aux besoins de l'entreprise. Par exemple, les intégrations proposées par les éditeurs ne sont pas modifiables par l'entreprise cliente. Elle ne peut que la déployer, mais si cette intégration ne lui convient pas, alors elle doit développer une nouvelle elle-même.

Troisièmement, le SOAR est un outil rigide. Cela rejoint en partie le manque de flexibilité que nous venons juste d'évoquer, mais à cela s'ajoute que tous les travaux réalisés sur le SOAR ne sont pas facilement extensibles et pas du tout exportables entre les différents éditeurs de ces solutions. Un changement de solution nécessite de tout reprendre à zéro ou presque. Il n'existe pas, à notre connaissance, de travaux pour standardiser et permettre l'interopérabilité au niveau des SOAR.

Ensuite, le SOAR est un outil très complexe. C'est une des conclusions que tire Orange Cyberdéfense dans son retour d'expérience sur les SOAR [4]. C'est un outil qui demande une expertise technique élevée afin de le mettre en place. Le déploiement des workflows demande aussi des compétences en programmation. Ce travail est souvent réalisé par des développeurs. Cela rajoute donc une contrainte au niveau de ces outils, car il est utilisé dans un but de réponse à incident de sécurité informatique, mais il est configuré par des personnes n'ayant souvent pas de compétences dans ce domaine. C'est une contrainte qu'il faut gérer pour une entreprise et la communication entre les équipes du SOC et les personnes travaillant sur le SOAR est donc cruciale.

Cependant, cette contrainte se retrouve accentuée par le fait que les processus et procédures liés à la réponse à incident manquent de maturité. C'est une des conclusions que nous tirons de nos travaux et des discussions avec des équipes travaillant sur le SOAR. Le principal obstacle rencontré, outre la pénurie de main-d'œuvre qualifiée pour réaliser le travail, est le manque de maturité des processus de réponse à incident. Les playbooks de réponse à incident ne sont pas suffisamment formels ce qui résulte en une non-uniformisation de la réponse par les analystes. Cette problématique se répercute sur le SOAR, car il n'y a pas ou peu de procédés clairement définis qui pourraient être automatisés. Ce problème que nous avons pu voir est aussi confirmé par le Gartner dans leur toute dernière analyse du marché des SOAR : « Le principal obstacle à l'adoption d'une solution SOAR reste l'absence ou la faible maturité des processus et procédures au sein du SOC » [2].

Un autre inconvénient du SOAR est pointé par le consortium IACD. D'après eux, un workflow doit pouvoir interagir avec les humains si nécessaire et être partageable entre organisations [11]. Nous tirons les mêmes conclusions de nos travaux. Cependant, cette fonctionnalité est presque inexistante parmi tous les SOAR que nous avons pu analyser.

Certains des problèmes évoqués ci-dessus sont repris par l'article « Les inconvénients d'un SOAR » [19]. Cependant, l'article met aussi en avant un autre élément. L'auteur estime qu'il y a encore trop peu de connaissances sur le SOAR et sur la place qu'il doit occuper au sein d'un SOC. Cet outil n'améliore pas la posture de sécurité dans le sens où il ne permet pas de réaliser des actions que l'on ne pouvait pas déjà réaliser. Le SOAR se situe plus dans l'optimisation de cette posture mais celle-ci doit être déjà bien définie avant l'adoption du SOAR. L'article conclut sur le fait que ce manque de connaissance crée des attentes non réalisables sur le SOAR et cela crée un

risque d'échec dans les projets de mise en place de cet outil. De plus, nous ajoutons à cette problématique le fait qu'il n'y a pas encore de guide de bonnes pratiques pour aider une entreprise à piloter son projet de mise en place d'un SOAR. Cela a pour conséquence que certains projets peuvent être mal pensés et risquent donc d'échouer.

Un autre problème évoqué par l'article d'Oliver Rochford [18] est que les éditeurs de SOAR ne fournissent que peu de workflows prêts à l'utilisation. Cela a pour conséquence que la majeure partie des workflows sont construits par l'entreprise. Ces workflows deviennent un élément à développer, maintenir et mettre à jour régulièrement. C'est un élément à gérer qui s'ajoute à une liste déjà longue au sein d'un SOC (ensemble de signature pour l'IDS, règle de corrélation dans le SIEM, règle dans le pare-feu, etc.). Cela rend l'adoption d'un SOAR encore plus délicate, car si elle est mal pensée, alors son utilisation peut s'avérer coûteuse en termes de temps et de personnel, ce qui est ironique quand on sait que le SOAR a justement pour but de pallier ces problèmes. Nous souhaitons aussi ajouter que les workflows des SOAR, une fois codifiés, sont relativement complexes à mettre à jour. Il peut y avoir un réel manque de visibilité pour une entreprise qui a déployé une cinquantaine de workflows sur son SOAR. C'est un constat qui vient renforcer la problématique présentée dans l'article d'Oliver Rochford « Pourquoi l'automatisation avec les SOAR échoue-t-elle ? » [18]. De plus, les workflows sont stockés dans un format propre à chaque éditeur. Il n'existe pas, à notre connaissance, de travaux portant sur l'interopérabilité et la standardisation comme pourraient l'être ceux de Sigma sur les règles de SIEM [20]. Cela pose un problème évident dans le cas où une entreprise souhaite changer d'éditeur.

Bien que cette technologie soit prometteuse et permet déjà de répondre à certains besoins, l'ensemble des problèmes cités montre que le marché n'en est qu'à ses débuts. De plus, les limites des SOAR en font un outil qui n'est pas immédiatement prêt à l'utilisation et qui va demander du travail en amont. Le retour sur investissement est rarement atteint lors des premiers mois comme l'explique Orange CyberDéfense dans son retour d'expérience [4]. Il est donc nécessaire d'explorer de nouvelles solutions afin de développer cette technologie.

2.2 Les ontologies

La solution que nous proposons dans nos travaux se base sur le concept d'ontologie. C'est une discipline relativement récente dans le domaine de l'informatique mais qui existe depuis plusieurs siècles maintenant du côté de la philosophie. C'est d'ailleurs de cette branche de la philosophie

qu'est inspiré le concept d'ontologie en informatique. L'ontologie, dans son sens le plus général, signifie l'étude de l'être. Voici une définition de ce concept par le dictionnaire Le Robert : « L'ontologie est une partie de la philosophie qui traite de l'être indépendamment de ses déterminations particulières ».

Ce concept s'est ensuite étendu à d'autres disciplines jusqu'à apparaître en informatique pour la première fois en 1993. En effet, c'est cette année-là que Tom Gruber définit une ontologie informatique comme étant « une spécification explicite d'une conceptualisation » [21]. La définition va ensuite s'étendre avec les travaux de Willem Nico Borst pour devenir « une spécification formelle d'une conceptualisation partagée ». Elle va finalement devenir, par suite des travaux de Studer et al. en 1998, « une spécification formelle et explicite d'une conceptualisation partagée » [22]. En d'autres termes, une ontologie est un ensemble structuré de concepts. C'est un modèle de données qui va permettre de représenter les concepts d'un domaine sous un format compréhensible à la fois par l'humain et par la machine. Les concepts sont représentés par des classes et ils sont reliés par des propriétés. Une ontologie peut se représenter sous forme graphique, avec des nœuds qui correspondent à ses classes et des arêtes qui correspondent à ses propriétés. À cela s'ajoute la présence d'individus qui sont des instances des classes. On utilise le terme graphe de connaissance pour parler des différentes instances qui peuplent une ontologie.

Un modèle de représentation de l'ontologie est le « Resource Description Framework » (RDF). Il définit une ontologie en tant qu'ensemble de triplets, c'est-à-dire un ensemble d'entités reliées entre elles par des propriétés. C'est une structure générique qui sert de base à un certain nombre de schémas ou langages comme RDFS (RDF Schema) ou OWL (Web Ontology Language). Chaque entité de l'ontologie est représentée par son URI.

Le format RDF définit un triplet comme étant la composante d'un « Sujet » lié à un « Objet » par un « Prédicat ». Par exemple, si l'on souhaite représenter le fait qu'un livre est écrit par un auteur, alors nous définirons le triplet suivant :

« Livre » « PossèdeAuteur » « Auteur »

2.2.1 Les composantes d'une ontologie

Tel qu'expliqué précédemment, une ontologie est composée de plusieurs éléments.

Les classes

Une classe est la définition d'un concept qui peut être réel ou abstrait. Les classes possèdent des attributs, des instances et des propriétés. Si l'on souhaite spécifier le concept de livre, nous allons définir la classe Livre à l'aide du triplet suivant :

Livre rdf:type rdfs:Class

Ici, nous spécifions le concept de livre en le nommant directement.

Les classes peuvent être liées à d'autres classes par la relation de sous-classe. Par exemple, si l'on souhaite ajouter à notre ontologie sur les livres, le concept de livre policier alors nous pourrons le faire avec les triplets suivants :

Livre_Policier rdf:type rdfs:Class

Livre_Policier rdfs:subClassOf Livre

Nous définissons donc un livre policier comme étant une sous-classe du concept déjà existant de livre. L'intérêt de cette définition, en plus d'ajouter plus de spécification dans l'ontologie, permet à la classe Livre_Policier d'hériter des propriétés de la classe Livre.

Les individus

Un individu est une instance d'une classe ou plusieurs classes. Il représente un élément particulier de l'ontologie. Si l'on poursuit avec notre ontologie sur les livres, nous pouvons définir les individus Dragon Rouge et Thomas Harris comme étant respectivement un livre et un auteur. Cela peut se faire avec les deux triplets suivants :

Thomas_Harris rdf:type Auteur

Dragon_Rouge rdf:type Livre_Policier

C'est ce qui s'appelle peupler l'ontologie. Les individus ici définis font partie de l'ontologie et peuvent hériter des relations de leur classe et des classes parentes.

Ces individus peuvent aussi être reliés entre eux par des propriétés.

Propriétés

Les propriétés sont des relations que possèdent les classes et individus. Il en existe de deux types, les propriétés de données et d'objet. Une propriété d'objet relie une entité de l'ontologie à une autre. Par exemple, si l'on souhaite définir que Thomas Harris est l'auteur du livre Dragon Rouge, alors nous allons utiliser une propriété d'objet :

Dragon_Rouge possèdeAuteur Thomas_Harris

La relation « possèdeAuteur » est donc une propriété d'objet qui lie deux entités, ici deux individus, de notre ontologie. Si nous souhaitons spécifier la date de parution de Dragon Rouge, nous pouvons le faire à l'aide d'une propriété de données et avec le triplet suivant :

Dragon_Rouge dateDeParution « Octobre 1981 »

Ici, la propriété « dateDeParution » est une propriété de données, c'est-à-dire qu'elle lie une entité à un littéral. Un littéral peut prendre la forme de plusieurs types de données comme une chaîne de caractères, un booléen, un entier, etc.

En spécifiant des classes, des propriétés et des individus, nous avons créé une ontologie sur les livres. Celle-ci peut être représentée sous forme graphique (voir figure 2.3).

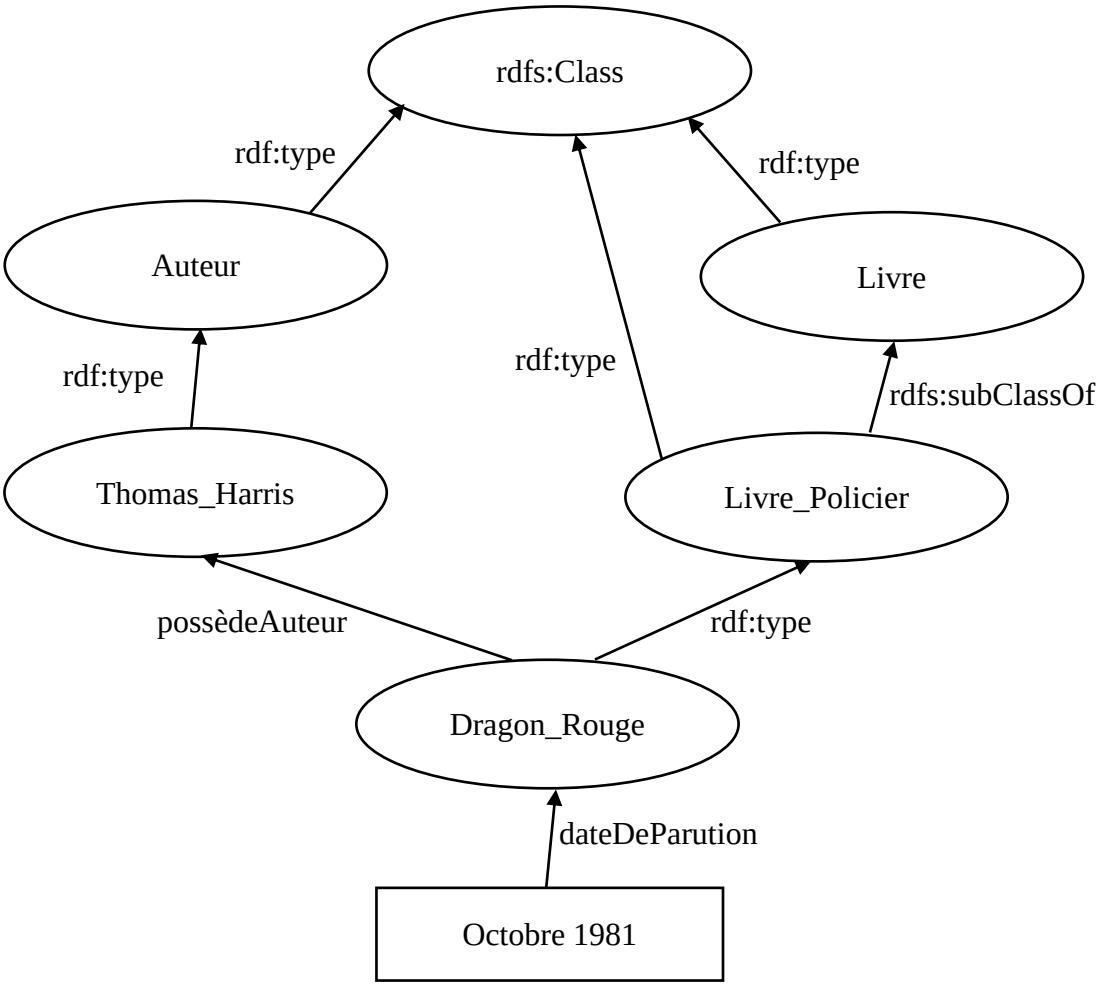

Figure 2.3 : Exemple d'ontologie

Nous avons construit cette base de connaissances en définissant manuellement à l'aide de triplets, chacun des nœuds et chacune des relations qui les lient. Cependant, nous pouvons aller plus loin dans notre représentation du concept de livre en nous servant de la capacité d'inférence des ontologies afin de générer grâce à la logique, de nouveaux triplets.

2.2.2 Le concept d'inférence

D'après le consortium W3, l'inférence dans le domaine des ontologies est « la capacité à découvrir de nouvelles relations » [23]. Ces nouvelles relations vont agrandir la base de connaissances en ajoutant de nouveaux triplets. Cette capacité d'inférence ou de raisonnement va réaliser sa déduction à partir des données déjà présentes dans l'ontologie. L'idée est d'imiter la capacité de raisonnement d'un être humain. Par exemple, si l'on définit que des « auteurs » sont des « êtres humains » et que l'on définit que « Thomas Harris » est un « auteur », alors un cerveau humain va

déduire que « Thomas Harris » est un « être humain ». Le raisonneur d'une ontologie, qui est l'élément qui gère sa capacité de déduction, va en faire de même.

Il existe différents types de logique qui vont permettre la déduction de nouvelles connaissances.

Premièrement, il y a la logique descriptive. C'est un ensemble de formalismes qui va représenter une base de connaissances. Cela permet de représenter formellement les concepts d'un domaine. Par exemple, si l'on définit que la classe Livre doit obligatoirement posséder un titre, un auteur et un éditeur alors tout individu de l'ontologie qui possédera ces trois relations sera considéré comme une instance de la classe Livre. La logique descriptive dispose de syntaxes permettant de créer des axiomes qui vont permettre d'enrichir l'ontologie et de réaliser de nouvelles déductions. Les formalismes de la logique descriptive peuvent se séparer en deux classes :

- La T-Box. Elle contient un ensemble d'axiomes définissant la structure d'une classe. Dans l'exemple précédent, nous définissons le concept de Livre comme étant tout individu possédant un titre, un éditeur et un auteur. Cet exemple de raisonnement fait partie de la T-Box, car nous inférons qu'un individu appartient à la classe Livre, s'il respecte les conditions définies. Par procédé inverse, toute instance de la Classe Livre doit aussi posséder les propriétés qui le lie à un titre, un éditeur et un auteur. Dans le cas contraire, l'ontologie est dite inconsistante, c'est-à-dire qu'il est impossible de représenter la base de connaissances qui a été décrite. Par conséquent, les règles de la T-Box permettent de déduire de nouvelles connaissances, mais aussi de vérifier la consistance de l'ontologie.
- La A-Box. C'est un ensemble d'axiomes qui va permettre des assertions sur les individus. Par exemple, si nous définissons que tous les livres écrits par Thomas Harris font partie de la catégorie « thriller », alors le raisonneur de l'ontologie va déduire de notre base de connaissances que Dragon Rouge est un thriller. Ici, l'assertion se fait sur l'individu et le raisonneur va déduire une nouvelle donnée liée à l'instance Dragon Rouge et aux autres livres écrits par l'individu Thomas Harris.

Il y a un autre moyen de faire de l'inférence grâce aux règles de logique écrites en SWRL (Semantic Web Rule Language). Comme l'explique le consortium W3, le langage SWRL est un langage par règles [24].

Une règle est construite à partir de deux éléments :

- Un antécédent qui est le corps de la règle, ou la condition à remplir.
- Une conséquence qui va générer un triplet.

Si un individu remplit les conditions de l'antécédent, alors la conséquence lui sera appliquée. Le raisonnement par règle de SWRL s'applique très bien à celui que l'on réalise naturellement dans le domaine de la famille. Voici un exemple :

possèdeParent(?x, ?y) ^ possèdeFrère(?y, ?z) → possèdeOncle(?x, ?z)

Ici, nous inférons que si un individu x possède un parent y et que ce parent possède un frère z, alors l'individu z est l'oncle de x. La partie à gauche de la flèche est l'antécédent, et la partie à droite est la conséquence.

2.2.3 Le requêtage

Le requêtage est un élément essentiel de l'ontologie. Il permet, au travers de requêtes, d'obtenir des informations sur les individus de l'ontologie. Le langage le plus connu est certainement le SPARQL. C'est un langage de requête pour RDF [25]. Une requête se construit avec un minimum de deux composantes :

- L'opération à effectuer sur la base de données. Il y a plusieurs opérations possibles, représentées par différents mots-clés (Select, Ask, Construct, Describe).
- Il y a ensuite les conditions que la requête doit respecter. C'est à ce niveau que la mise en forme de la requête se fait. Cela est permis grâce au mot-clé « Where ».

Reprenons notre ontologie sur les livres présentée ci-dessus. Si l'on souhaite obtenir de celle-ci tous les livres écrits par l'auteur Thomas Harris, nous allons réaliser la requête SPARQL suivante (tableau 2.1) :

```
SELECT ?livre
WHERE {
        ?livre possèdeAuteur ?auteur .
        ?auteur nomComplet « Thomas Harris » .
}
```

Tableau 2-1 : Exemple de requête SPARQL

La requête ci-dessus va chercher tous les livres qui possèdent un auteur qui a pour nom complet Thomas Harris. Dans le cas de l'ontologie que nous avons créée, il nous faudra ajouter la propriété nomComplet à Thomas Harris afin d'obtenir le résultat « Dragon Rouge ». C'est grâce à ce genre de requête que l'on peut interroger la base de données ontologique.

2.2.4 Méthodes de développement d'ontologies

Nous avons, dans les sections précédentes, créé une ontologie sur les livres, cependant celle-ci est très limitée. Une ontologie peut être bien plus complète et complexe que ce que nous avons réalisé et il faut donc organiser son travail selon une certaine méthodologie si l'on souhaite atteindre ses objectifs.

Les premiers travaux sur des méthodes de développement d'ontologies sont apparus en 1995 avec ceux de Uschold et King [26]. Aujourd'hui, il en existe un grand nombre, cependant pour nos travaux de recherche nous nous sommes principalement appuyés sur deux méthodes de développement. Ces deux méthodologies se basent sur ce que l'on appelle des questions de compétences. Ce sont des questions exprimées en langage naturel et l'ontologie doit être en mesure d'y répondre. Un exemple est la question suivante : « Qui est l'auteur du livre Dragon Rouge ? ».

La première que nous avons utilisée est SAMOD. L'acronyme signifie « Simplified Agile Methodology for Ontology Development » [27]. C'est une méthodologie itérative et il est donc souvent nécessaire de réaliser plusieurs itérations afin d'obtenir le résultat final.

La première étape n'est cependant pas incluse dans les itérations. Il s'agit tout d'abord de recueillir un maximum d'information sur le domaine pour lequel nous souhaitons construire notre ontologie. Cela est un point commun à toutes les méthodes mais c'est une étape nécessaire, car il faut bien

comprendre le domaine et ses concepts si l'on souhaite construire une ontologie. Les étapes suivantes sont ensuite itératives. Voici les différentes étapes de cette méthode :

- Énumérer une liste de questions en langage naturel. Ces questions doivent pouvoir être répondues par l'ontologie.
- Lister les concepts qui sont présents dans les questions.
- Réaliser un glossaire pour définir chacun des concepts.
- Traduire les questions en langage naturel en requête SPARQL.
- Insérer les concepts dans l'ontologie

Le résultat d'une itération est appelé un modelet. Chaque modelet généré par une itération doit être fusionné avec le reste de l'ontologie. L'itération se poursuit jusqu'à répondre à la totalité des questions de compétences. Les questions de la première itération doivent concerner les concepts les plus basiques du domaine. Par exemple, dans le cas d'une ontologie sur les Livres, les premières questions de compétences peuvent être :

- Quel est l'auteur d'un livre ?
- Quel est l'éditeur d'une livre ?
- Quel est le titre d'un livre ?

La seconde itération permet de rentrer plus dans le détail et peut concerner des concepts comme le thème d'un livre ou son code ISBN.

La seconde méthode de développement d'ontologie est ATOM [28]. Une partie de la méthodologie d'ATOM est similaire à SAMOD.

Voici les étapes de cette méthode :

- Définir en langage naturel les requêtes de l'ontologie
- Traduire les requêtes en SPARQL
- Spécifier les informations brutes. Cela signifie qu'il faut utiliser les compétences d'un expert du domaine afin de savoir quel concept permet de répondre à la question.

- Décomposer la question en requêtes intermédiaires. Logiquement, l'étape précédente génère une multitude de concepts qui peuvent être décomposés en plusieurs autres concepts.
- Écrire les règles de traduction. C'est à cette étape que la capacité de raisonnement doit être analysée et si besoin développée à l'aide de règles SWRL.

Nous avons utilisé ces méthodes de développement même si nous voyons une limitation importante. Il n'est pas explicitement indiqué dans aucune des étapes de réutiliser des ontologies existantes. C'est pourtant un élément essentiel des ontologies que de permettre la réutilisabilité et d'être facilement extensible. Pour pallier cela, nous avons donc réalisé un état de l'art des ontologies appliqué à la sécurité informatique. Nous avons par la suite réduit au fur et à mesure notre recherche de travaux aux playbook et à la réponse à incident.

2.2.5 Ontologies en sécurité informatique

L'utilisation des ontologies en informatique et plus particulièrement dans le domaine de la sécurité a beaucoup gagné en popularité aux cours des dernières années. Il y a eu premièrement divers travaux qui ont eu pour objectif de formaliser des concepts clés en sécurité informatique.

Les travaux de l'organisation Mitre avec leur ontologie « Digital Artifact Ontology » ont pour objectif de formaliser sous forme ontologique plusieurs concepts en sécurité informatique. Chacun des concepts (par exemple un réseau ou un trafic réseau entrant) est associé à des attaques existantes. Les travaux sont toujours aux stades d'expérimentation ce qui explique que le niveau d'abstraction des concepts soit relativement haut. Cependant, certains éléments sont tout de même intéressants pour nos travaux. L'ontologie créée par Mitre contient notamment une classification des attaques et cela est intéressant pour notre recherche. Nous avons donc réutilisé les concepts de tactiques et de techniques offensives ainsi que leurs propriétés.

D'autres exemples de travaux sont ceux sur les vulnérabilités et leur gestion. On recense parmi ces travaux ceux du NIST [29] avec VDO et ceux de Wang et al. avec OVM [30]. Les travaux du NIST ont pour objectif de permettre une meilleure description et caractérisation des vulnérabilités informatiques. Leur but est ensuite d'assister l'utilisateur dans la gestion de la vulnérabilité. L'ontologie OVM quant à elle permet de faire le lien entre les vulnérabilités et des produits informatiques qui les contiennent. De plus, l'ontologie permet de montrer des contremesures pour

pallier ces vulnérabilités. Bien que nous n'ayons pas réutilisé ces ontologies dans notre recherche, la lecture et la compréhension de ces ontologies nous ont aidés à orienter nos travaux.

Les articles cités traitent de l'utilisation des ontologies en sécurité informatique. Nous avons par la suite réduit notre recherche d'articles à l'utilisation des ontologies pour la réponse à incident et pour l'automatisation de celle-ci.

2.2.6 Les ontologies pour la réponse à incident et l'automatisation

Les travaux de Moreira et al. [31] portent sur l'utilisabilité des ontologies pour la gestion et le traitement des incidents de sécurité. Cette ontologie permet à une entreprise et à des analystes de documenter les actions prises en réponse à un incident de sécurité. Plusieurs concepts sont définis dans l'ontologie. Parmi eux, deux éléments sont intéressants pour notre recherche. Le concept d'action et d'événement. Une action est « quelque chose qui peut être réalisée ». Un événement est « quelque chose qui peut arriver dans un contexte numérique ». En pratique, des événements sont le résultat des actions réalisées sur un système d'information et sont représentés par la journalisation des événements qui ont eu lieu. On retrouve aussi ces concepts dans l'ontologie UCO [32]. Nous les avons réutilisés dans nos travaux.

Nous avons ensuite lu et analysé l'article écrit par Applebaum et al. [33] dans le cadre de leurs travaux au sein du Mitre. Ces travaux sont les prémices de leur réflexion sur l'utilisabilité d'une ontologie pour décrire des playbooks de réponse à incident. L'objectif est de proposer une ontologie qui permet de réaliser une spécification de playbook et ainsi fournir à l'analyste un ensemble d'actions précis à réaliser en réponse à un événement. L'objectif est de permettre une réponse à incident plus précise, plus rapide et plus efficace. Nos motivations pour réaliser ce travail de recherche se sont en partie basées sur cet article. Cependant, celui-ci reste à un stade d'idéation. Même si certains concepts sont énoncés, il y a un grand nombre d'éléments qui reste non spécifiés. De plus, il y a, à notre connaissance, aucune autre ressource de la part du Mitre lié à ce travail de recherche.

Un article basé sur les ontologies et intéressant pour nos travaux est celui de Hutschenreuter et al. [34] sur l'application des ontologies à la résilience d'une infrastructure portuaire. L'objectif des travaux est de permettre la résilience au travers d'actions automatisées et initiées par l'ontologie. Pour cela, l'article met en avant des éléments qui constituent la T-Box de l'ontologie. Les travaux

ont été découpés en 3 concepts importants, la T-Box sur le matériel informatique, celle sur les logiciels, et enfin une dernière sur la réponse à incident. Bien que l'objectif des travaux soit d'appliquer l'ontologie à la résilience d'une infrastructure portuaire, certains concepts comme celui de « Menace » ou de « Mesure de sécurité » sont réutilisables dans notre ontologie avec quelques modifications. Toutefois, une limite que nous voyons à ces travaux est le fait que l'ontologie ne soit pas plus détaillée et que la preuve de concept n'a pas été réalisée. De plus, le concept de mesure de sécurité, bien qu'important, n'est pas rattaché à un quelconque ordre d'action ce qui peut être une limite importante pour l'automatisation.

Les derniers travaux portant sur les ontologies que nous allons présenter dans ce mémoire sont appliqués au concept de SOAR.

Le premier a pour but d'automatiser l'intégration des outils de sécurité en générant automatiquement les commandes nécessaires à la communication avec l'API. Ce travail a été réalisé par Islam et al. [35]. Leur ontologie contient des informations permettant à un modèle d'intelligence artificielle de traitement automatique des langues de générer les commandes nécessaires à la réalisation des actions sur un outil de sécurité. Une preuve de concept a aussi été mise en place. Les travaux sont détaillés et l'ontologie est disponible. Cependant, les limitations importantes que nous voyons avec ces travaux se situent premièrement sur l'utilisation de l'intelligence artificielle sur un aspect crucial de l'automatisation. Bien que les résultats de leurs expérimentations soient élevés, il n'est pas mentionné comment agirait le système en cas d'erreur du modèle d'intelligence artificielle. Cela est pourtant un élément essentiel, car une erreur à ce stade de l'automatisation de la réponse peut entraîner d'importantes conséquences sur l'activité de l'organisation. Enfin, il n'est pas mentionné comment le système pourrait s'intégrer à une plateforme de SOAR existante. Dans le cadre de leurs travaux, il n'y a pas d'utilisation d'un SOAR. Toutefois, le travail réalisé est important et il est, à notre connaissance, la première réflexion sur l'utilisabilité des ontologies pour améliorer le concept de SOAR. L'ontologie est composée de trois classes principales, « Outil de sécurité », « Activité » qui est en fait une action à réaliser, et enfin « Capacité » qui correspond à la capacité de réponse d'un outil de sécurité. Ces trois classes ont été reprises dans notre ontologie avec quelques légères modifications.

Pour finir, le SOAR « Siemplify », maintenant devenu Chronicle SOAR, se sert des ontologies pour sa plateforme de suivi des alertes de sécurité. Plus précisément, d'après la documentation du

produit, l'ontologie fournit une spécification formelle qui permet la représentation des alertes et événements qu'ingère le SOAR. Cela lui permet de construire des relations entre les différentes entités impliquées dans l'alerte. Pour cela, l'ontologie contient différents types de scénarios, nommés des types de familles, qui permettent la représentation d'alertes. Par exemple, il existe la famille « DDoS » qui correspond à un type d'attaque de déni de service. Pour ce scénario, l'ontologie contient différents artefacts comme « SourceAddress » et « DestinationAddress ». Ces artefacts vont contenir les données de l'alerte. Lorsqu'un événement correspond à une alerte de type DDoS, l'ontologie va associer les données avec la représentation liée à cette famille d'attaques et va ainsi modéliser l'alerte. Ce travail, bien qu'intéressant, reste difficile à réutiliser, car nous manquons d'informations sur celui-ci. En effet, le détail de l'ontologie n'est, à notre connaissance, pas public. De plus, même en utilisant le SOAR, nous ne pouvons obtenir que de légères informations. Tout de même, il reste intéressant et permet d'adapter notre format de données.

2.3 Synthèse

Cette section présente la synthèse du chapitre 2.

Dans ce chapitre, nous avons présenté comment était organisée la réponse à incident au sein d'un SOC. Celle-ci suit un cycle de vie constitué de plusieurs étapes. La description la plus connue de ce cycle de vie est certainement celle du NIST. Elle est organisée selon les étapes suivantes : La phase de Préparation. Ensuite, il y a la phase de Détection & Analyse des incidents. S'en viennent les étapes de Confinement, Éradication et Récupération. Il y a enfin la phase des Activités Post-Incident.

Les différentes étapes de ce cycle de vie sont réalisées par les membres du SOC. Plus précisément, ce sont les analystes de réponse à incident qui réalisent la plupart de ces actions, notamment les phases d'analyse et de remédiation aux incidents de sécurité. Ils appliquent des playbooks de réponse à incident. Ce sont des documents organisationnels qui décrivent les actions à réaliser pour répondre à un incident précis. En fonction des entreprises et des travaux, les playbooks peuvent prendre la forme de documents textuels, de représentations graphiques etc. L'application des actions des playbooks de réponse à incident a pendant longtemps été exclusivement réalisée de façon manuelle par l'analyste.

Cependant, avec la démultiplication des attaques et leur complexification, les analystes se retrouvent surchargés par les alertes de sécurité. Cela a pour conséquence de réduire la qualité et la rapidité de la réponse à incident. Ce concept porte le nom d'« Alert Fatigue ». Il a donc été nécessaire de trouver des solutions à cette problématique.

C'est de ce besoin qu'est né le concept des SOAR. Il décrit les plateformes permettant de réaliser l'orchestration et l'automatisation de la réponse à incident. Ces plateformes apportent un certain nombre d'avantages.

Cependant le concept des SOAR est récent et il possède encore plusieurs inconvénients. C'est un outil très complexe et son adoption nécessite du personnel qualifié. Les délais peuvent être longs (de 6 à 12 mois) avant d'obtenir les premiers retours sur investissement. La flexibilité qu'il offre est aussi un problème souvent remonté lors des retours d'expérience. Un autre élément soulevé par le Gartner est le manque de maturité des processus de réponse à incident. Ce n'est pas une limite des SOAR mais c'est une contrainte qui se répercute directement sur l'outil. Cet ensemble de problématique se répercute directement sur les projets d'adoption d'un SOAR qui sont à risque d'échouer. Il est donc nécessaire d'explorer des pistes afin de solutionner ces problématiques.

Dans ce mémoire, nous étudions l'utilisation des ontologies comme outil pour formaliser des playbooks de réponse à incident. Nous étudions ensuite la capacité d'une ontologie à s'intégrer avec le SOAR.

CHAPITRE 3 PROPOSITION D'UNE ONTOLOGIE DE PLAYBOOK

Dans les chapitres précédents, nous avons vu que les SOC font fassent à un volume grandissant d'alertes. Le besoin d'une réponse plus efficace et rapide est devenu nécessaire. Nous avons aussi analysé l'apport des SOAR et déterminé certaines limites quant à l'utilisation de cet outil. Nous proposons dans ce chapitre, une ontologie de playbook afin de résoudre certains problèmes liés à la réponse à incident et aux SOAR.

3.1 Modèle ontologique

Dans ce chapitre, nous présenterons en détail l'ontologie, ainsi que la méthodologie utilisée pour la développer.

3.1.1 Les classes

Comme nous l'avons expliqué dans le chapitre précédent, une ontologie se compose de classes, de propriétés et d'individus. Ces éléments nous permettent de formaliser des playbooks de réponse à incident. Un playbook est constitué d'une succession d'étapes (Playbook_Step), qui correspondent à des actions de réponse à incident (Action). Ces actions doivent être réalisées à l'aide d'outils de sécurité (Tool et Security_Tool) et sur des données (Artefact). La réalisation de ces playbooks permet de répondre aux tactiques et techniques d'attaques utilisés par les attaquants (ATTACK Thing). La formalisation de cet ensemble d'informations est permise grâce aux classes, aux propriétés et aux individus qui composent notre ontologie.

Nous allons premièrement présenter les classes de l'ontologie. La figure 3.1 provient du logiciel d'édition d'ontologie « Protégé ».

Figure 3.1 : Classes du modèle ontologique

Certaines classes proviennent d'ontologies déjà existantes. Elles sont présentées ci-dessous. C'est le cas notamment de la classe « Action » qui a été directement tirée de l'ontologie UCO [32]. Elle est définie comme étant « quelque chose qui peut être réalisé ». La définition de ce concept convient à notre domaine d'application, car elle reste relativement large et est adaptée à la réponse à incident. En effet, il est possible de classer les actions de réponse à incident en trois groupes :

- Des actions organisationnelles, comme l'envoi d'un courriel pour avertir d'une campagne d'hameçonnage en cours
- Des actions d'enrichissement ou d'analyse, qui vont apporter plus de données et de contexte à l'alerte qui est en cours d'investigation
- Des actions opérationnelles qui vont entraîner une conséquence sur l'environnement informatique de l'organisation. Il s'agit par exemple de supprimer des courriels.

Les classes « Tool » et « Security Tool » proviennent aussi d'ontologies existantes, respectivement de l'ontologie UCO et de l'ontologie SecOrP [35]. Ces classes correspondent à des éléments physiques ou logiciels de l'environnement informatique d'une organisation. Ces entités ont des fonctions définies et réalisent des actions. La distinction entre les deux concepts a pour objectif

d'obtenir une ontologie plus précise. De plus, les outils de sécurité sont impliqués dans la réalisation des actions de réponse à une alerte. Les sous-classes de « Security Tool » proviennent aussi de l'ontologie SecOrP. Elles ne sont pas obligatoires dans l'utilisation de notre ontologie mais nous souhaitons conserver cette importation des classes, car une catégorisation des outils de sécurité apporte plus de clarté. De plus, cela permet à notre ontologie une meilleure extensibilité avec d'autres modèles existants.

Enfin, les derniers concepts importés dans notre ontologie sont ceux liés à la matrice MITRE ATT&CK. Ils sont représentés par les classes suivantes : « ATTACK Thing », « Offensive Tactic » et « Offensive Technique ». Ils proviennent de l'ontologie Digital Artifact de l'organisation Mitre. Ces classes représentent les concepts définis par Mitre pour leur matrice « Att&ck » et qui ont été expliqués précédemment.

Les classes que nous avons réutilisées sont nécessaires pour le bon fonctionnement de notre ontologie, cependant, elles n'introduisent à aucun moment le concept de playbook. Nous avons donc dû associer ces classes déjà existantes à de nouvelles classes que nous avons créées. Ces concepts ont déjà été explicités dans les chapitres précédents hormis pour les classes « Playbook Step » et « Command ». La première classe est abstraite, car elle ne correspond à proprement parler à aucun concept de la réponse à incident. Cependant, sa présence dans l'ontologie est nécessaire, car chaque individu de cette classe garde en mémoire sa position dans l'ordre des actions d'un playbook. Un playbook de notre ontologie sera donc constitué d'un ou plusieurs individus de la classe « Playbook Step » qui vont définir l'étape à laquelle l'action se situe.

Concernant la classe « Command », les individus de cette classe correspondent à des commandes à réaliser pour exécuter une action. Chaque commande est associée à un outil de sécurité et à une action de l'ontologie.

3.1.2 Les propriétés

Nous avons créé plusieurs propriétés pour notre ontologie. La capture d'écran du logiciel « Protégé » de la figure 3.2 montre les différentes propriétés que contient l'ontologie.

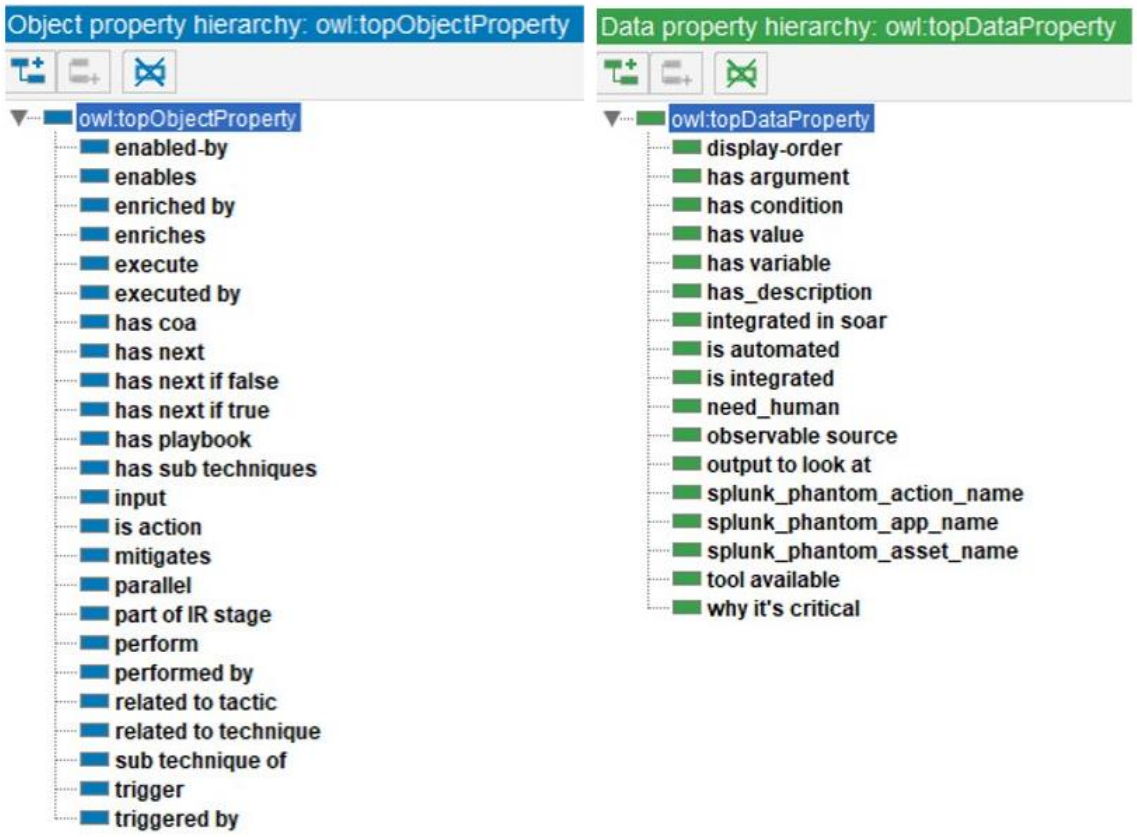

Figure 3.2 : Propriétés du modèle ontologique

Nous n'allons pas détailler chacune des propriétés, cependant nous allons expliquer un élément essentiel de notre ontologie qui est la façon dont nous représentons un ordre d'actions. Une ontologie ne contient par défaut rien qui permettrait de définir une séquence d'actions. Il faut donc créer des propriétés qui, associées à des individus, vont permettre de déterminer un ordre. Pour réaliser notre représentation des playbooks, nous avons donc créé les propriétés suivantes :

- « has coa ». Cette propriété détermine si un playbook contient un ordre d'action. Si c'est le cas, ce qui doit l'être pour chaque playbook, alors la propriété dirige vers la première étape. Cela donne un triplet : « Playbook – has_coa – Étape 1 »
- « has next ». Cette propriété détermine l'élément suivant dans l'ordre des actions. Cela résulte en la création d'un triplet : « Étape 1 – has next – Étape 2 »
- Pour finir, il existe les propriétés « has next if false » et « has next if true » qui vont permettre de représenter un ordre d'action dans le cas où il y aurait des conditions. Cela

permet donc de définir des ordres d'actions différents dépendamment du contexte d'une alerte.

Il n'existe, par défaut, pas de solution permettant de décrire une séquence d'action en OWL. Nous avons donc opté pour la solution ci-dessus car elle est adaptée à notre objectif et permet de répondre aux besoins de modélisation des playbooks. Nous discutons de cette limitation de notre solution dans le chapitre sur les contraintes liées à l'utilisation d'une ontologie pour la formalisation d'une séquence d'actions.

3.1.3 Les individus

Nous avons présenté les concepts et les propriétés qui constituent l'ontologie. Chacune des classes de l'ontologie va contenir des individus ce qui va permettre à une entreprise de formaliser ses playbooks de réponse à incident. Notre ontologie doit permettre de créer une base de données de playbooks. Afin de permettre une formalisation claire du concept de playbook, nous avons réutilisé des bases de connaissances existantes et reconnues dans le monde de la sécurité informatique.

Nous avons réutilisé la grammaire du framework MITRE ATT&CK concernant le langage lié aux attaques et le framework RE&CT concernant le langage de réponse à incident. C'est en nous servant de ces deux bases de connaissances reconnues que nous avons peuplé notre ontologie.

Pour le langage lié aux attaques, nous avons instancié les classes « Offensive Tactic » et « Offensive Technique » avec les éléments que constitue la matrice MITRE ATT&CK. Cela représente donc 14 individus pour les tactiques d'attaques et 567 pour les techniques d'attaques. Chacun des individus est créé dans l'ontologie selon l'identifiant de sa technique ou tactique. Une étiquette a été ajoutée afin de l'associer au nom réel de sa technique ou tactique. Enfin, l'ontologie contient, pour chacun des individus, une URL vers le site du framework MITRE ATT&CK qui contient une description plus détaillée des techniques et tactiques.

Cette instanciation a été automatisée en se servant du langage STIX qui contient les données de la matrice MITRE ATT&CK. STIX est un langage et un format libre de droit pour le partage des données de renseignements sur les menaces (Cyber Threat Intelligence) [36]. Il possède entre autres les données liées à la matrice MITRE ATT&CK. La création des individus dans l'ontologie reprend la structure et les termes utilisés par le consortium Mitre. Cela permet donc à l'ontologie d'être facilement mise à jour en cas de modification du framework.

Concernant le langage lié à la réponse à incident, nous avons réutilisé la matrice RE&CT. La matrice RE&CT s'inspire de la matrice MITRE ATT&CK. Cependant, ses colonnes correspondent aux différentes étapes du cycle de vie de réponse à incident du NIST. Chaque colonne est constituée de plusieurs actions de sécurité. C'est donc cette grammaire que nous avons réutilisée afin d'instancier les classes de « Incident Response Phase » et de « Action ». La première est constituée de 6 individus et la seconde de 216 individus. De la même façon que pour la matrice MITRE ATT&CK, chacun des individus est défini par son identifiant. Une étiquette a aussi été ajoutée pour nommer les individus. Les individus dans l'ontologie sont associés aux étapes de la réponse à incident de la même manière que dans la matrice RE&CT. L'ajout de ces individus a aussi été automatisé et suit la même structure que la matrice. Cela permet donc de mettre à jour l'ontologie en cas de modification du framework.

3.2 Méthodologie utilisée

Pour construire une ontologie, il existe différentes méthodes de développement. Parmi les méthodes existantes, nous avons préféré utiliser SAMOD, car elle s'applique bien à ce genre de cas d'usages. Cependant, nous avons aussi introduit une étape d'ATOM afin de créer des règles SWRL et ajouter de la logique descriptive dans la définition des classes.

Étape 1 : Énumérer une liste de questions en langage naturel.

Nous avons généré une liste de questions liées aux playbooks de réponse à incident. Nous avons déterminé les questions à l'aide des différentes lectures et travaux que nous avons réalisés. De plus, nous avons eu des contacts avec des analystes SOC afin d'affiner les questions. Les questions de compétences sont orientées vers l'interrogation d'une base de connaissances playbooks.

Nous avons regroupé ces questions en plusieurs blocs. Premièrement, l'ontologie stocke les cas d'usages de détection et de réponse à incident et leur playbook. Nous avons donc les questions suivantes :

- Quels sont les cas d'usages de détection et de réponse à incident de l'entreprise ?
- Quels sont les playbooks de réponse à incident lié à ces cas d'usages ?
- Quelles sont les descriptions des cas d'usages ? Des playbooks ?

- À quelles techniques et tactiques de la matrice MITRE ATT&CK correspondent ces cas d'usage ?

Ces questions correspondent à la première itération de SAMOD. Elles ont pour but d'introduire les concepts clés liés à la réponse à incident. Ici, elles sont formulées au sens général, c'est-à-dire que nous souhaitons obtenir une liste de cas d'usage. Cependant, il est possible de se concentrer sur uniquement un cas d'usage ou un playbook, car chaque individu de l'ontologie est défini par la propriété rdfs:label, ce qui le rend requêtable directement.

Étape 2 : Lister les concepts qui sont présents dans les questions.

Voici les entités introduites par la première itération :

- Use_case
- Playbook
- OffensiveTactic
- OffensiveTechnique
- has_playbook
- has_description
- related_to_tactic
- related_to_technique

Nous introduisons donc quatre classes et quatre propriétés liées à ces classes.

Étape 3 : Réaliser un glossaire pour définir chacun des concepts.

Il faut maintenant définir les concepts qui ont été introduits lors de l'étape précédente. Ces définitions vont être ensuite insérées dans l'ontologie au niveau de l'annotation rdfs:comment.

Voici le glossaire pour les classes et propriétés :

UseCase : Cela correspond à un cas d'utilisation spécifique surveillé par des outils de détection d'intrusion. Par exemple, une attaque par force brute sur un compte AWS peut être un exemple d'instance de la classe UseCase.

Playbook : Un playbook est une succession d'actions qu'il faut réaliser pour répondre à une alerte de sécurité.

OffensiveTactic : Cela correspond à une tactique d'attaque figurant dans la matrice MITRE ATT&CK.

OffensiveTechnique : Cela correspond à une technique d'attaque figurant dans la matrice MITRE ATT&CK.

has_playbook : Cette propriété relie un cas d'usage à un playbook de l'ontologie.

has_description : Cette propriété contient une description textuelle d'un playbook ou un cas d'usage

related_to_tactic : Elle associe un cas d'usage à la technique d'attaque correspondante.

related_to_technique : Cette propriété associe un cas d'usage à une technique d'attaque.

Étape 4 : Traduire les questions en langage naturel en requête SPARQL.

Une fois les questions en langage naturel formulées et les concepts définis, il faut ensuite rédiger les requêtes SPARQL qui permettront de répondre à ces questions (voir tableau 3.1).

Questions de compétences	**Requête SPARQL**

Q1	PREFIX :<http://kevin.abouahmed/IncidentResponseOntologyPlaybook#>
	SELECT ?usecase
	WHERE {
	?u a :UseCase .
	?u rdfs:label ?usecase
	}

Q2	PREFIX :<http://kevin.abouahmed/IncidentResponseOntologyPlaybook#> SELECT ?usecase ?playbook

	WHERE {
	?usecase a :UseCase .
	?playbook a :Playbook .
	?usecase :has_playbook ?playbook .
	}

Q3	PREFIX :<http://kevin.abouahmed/IncidentResponseOntologyPlaybook#>
	SELECT ?usecase ?uc_description ?playbook ?pb_description
	WHERE {
	?usecase a :UseCase .
	?playbook a :Playbook .
	?usecase :has_playbook ?playbook ;
	:has_description ?uc_description .
	?playbook :has_description ?pb_description .
	}

Q4	PREFIX :<http://kevin.abouahmed/IncidentResponseOntologyPlaybook#>
	SELECT ?usecase ?tactic ?technique
	WHERE {
	?usecase a :UseCase .
	?usecase :related_to_tactic ?tactic .
	?usecase :related_to_technique ?technique .
	}

Tableau 3-1 : Requêtes SPARQL du premier modelet

Étape 5 : Insérer les concepts dans l'ontologie

À ce stade, nous avons défini tout ce dont nous avions besoin pour créer le premier modelet de l'ontologie. Nous pouvons donc commencer la création de l'ontologie. Il s'agit ici de créer les classes et les propriétés définies lors des étapes précédentes. De plus, nous ajoutons à ces éléments les annotations rdfs:label et rdfs:comment qui correspondent respectivement au nom de la classe et à la définition de son concept telle qu'écrite dans le glossaire.

Voici une représentation de la première itération :

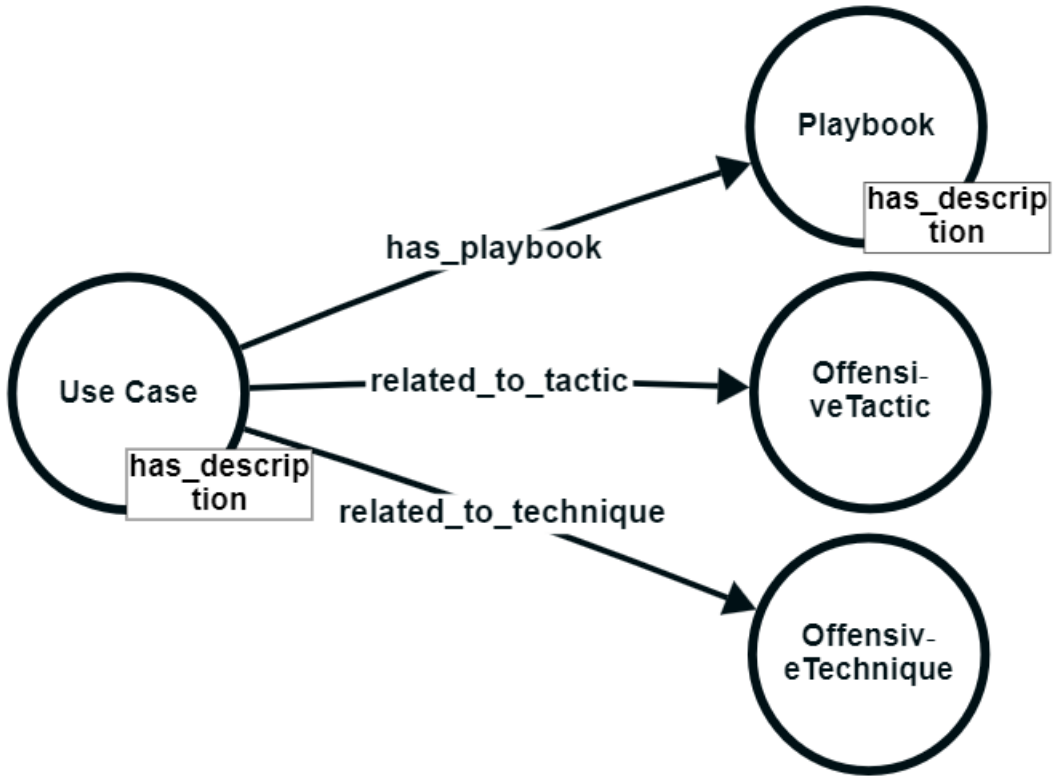

Figure 3.3 : Représentation graphique du premier modelet

Ici prend fin la première itération de SAMOD. Cependant, comme nous la trouvons en partie incomplète nous ajoutons une étape tirée de la méthodologie ATOM.

Étape 6 : Expliciter les règles de transformation

Cette étape consiste en l'ajout de règles SWRL et de logique descriptive. Dans le cas de notre premier modelet, il n'y a pas de règle SWRL à ajouter. Cependant, nous pouvons ajouter de la logique descriptive. Prenons l'exemple de la classe « UseCase ». Nous l'avons définie comme étant un élément surveillé par des outils de détection. Le but ici est de détecter un possible début d'attaque informatique. Cette attaque informatique est réalisée au travers d'une technique

offensive. Nous pouvons donc, dans l'ontologie, définir que la classe « UseCase » correspond à tout élément ayant exactement une seule propriété « related_to_technique » qui la relie à une technique d'attaque de la matrice MITRE ATT&CK.

Les règles SWRL, quant à elle, nous permettent la création de nouveaux axiomes afin d'obtenir une ontologie plus complète. Nous les utilisons pour permettre une meilleure représentation de la matrice MITRE ATT&CK.

Bien que cette étape ne soit pas nécessaire, nous la réalisons dans le but de définir de façon plus approfondie l'ontologie. La logique descriptive ainsi que les règles SWRL sont un moyen de réaliser de l'inférence et donc d'obtenir une ontologie qui va contenir plus d'axiomes et donc plus de connaissances sur les playbooks.

Une fois les étapes d'une itération terminées, nous pouvons en recommencer une autre si nous souhaitons encore développer l'ontologie.

3.3 Apports de l'ontologie

L'objectif de nos travaux s'est articulé autour du concept de playbook qui est un élément essentiel dans la réponse à incident et pour le SOAR. Notre objectif principal était de permettre la création d'une base de connaissances de playbook. Cela facilite la gestion et la mise à jour régulière d'un grand nombre de playbooks, problème que possède un grand nombre d'organisations. De plus, comme nous l'avons présenté dans le chapitre précédent, un des problèmes majeurs dans la réponse à incident et qui se répercutent sur les SOAR est le manque de maturité des processus de réponse à incident.

L'objectif de notre ontologie est de permettre à une entreprise de définir clairement les processus de sécurité à réaliser pour répondre à une alerte. Une définition plus formelle des playbooks améliore leur compréhension pour toutes les équipes impliquées dans les travaux de réponse à incident et dans le développement du SOAR. Cela entraîne une réponse plus efficace, élément qui est nécessaire aujourd'hui pour pouvoir répondre à un volume d'alertes grandissant.

Le fait de définir des playbooks sous un format ontologique apporte aussi la possibilité de réaliser des requêtes sur cette base de données. Nous défendons que cela est une plus-value et permet d'obtenir une meilleure visibilité sur la capacité de réponse à incident d'une organisation. De plus,

l'avantage d'un format ontologique et de l'utilisation d'une grammaire reconnue dans le monde de la sécurité informatique est de permettre le partage entre organisations.

Enfin, un autre objectif de la création d'un moyen de formalisation sous format ontologique est de permettre une meilleure gestion de l'ensemble des playbooks. Dans un contexte où une entreprise doit déjà gérer régulièrement ses règles de corrélation dans le SIEM, ses règles de détection dans un IDS ainsi que les règles d'un pare-feu, il est nécessaire d'avoir un outil qui facilite la gestion de l'ensemble des playbooks.

3.4 Contrainte de l'utilisation d'une ontologie

Dans cette section, nous discutons des contraintes liées à l'utilisation d'une ontologie à des fins de représentation d'un ordre d'actions. Nous avons décelé deux contraintes, qui ne sont cependant pas des limitations à nos travaux mais nous souhaitons tout de même les présenter.

Premièrement, nous souhaitons émettre une réserve quant à la modélisation de certaines conditions au sein d'un playbook. L'ontologie est en mesure de représenter les conditions grâce aux propriétés d'objets « has next », « has next if true » et « has next if false ». Cela permet donc de spécifier que si la condition est respectée, alors l'action suivante est X, sinon elle est Y et dans tous les cas, l'action Z doit être réalisée. Cependant, un playbook peut contenir des conditions plus complexes. Par exemple, si un courriel possède une URL alors nous souhaitons réaliser l'action X, et s'il possède une pièce jointe, alors l'action Y doit être réalisée. Il n'est actuellement pas possible de modéliser ce séquencement d'actions en un seul nœud. Afin de représenter cet enchaînement à l'aide de notre ontologie, il faudra définir deux nœuds de condition, un premier pour vérifier si le courriel possède une URL, et un autre pour vérifier s'il possède une pièce jointe. Le dernier nœud se situera à la suite de la propriété « has next ». Cet ajout est une contrainte du concept des ontologies mais qui n'a finalement qu'un impact mineur sur le résultat final.

Deuxièmement, nous émettons une réserve quant à l'utilisation des ontologies à des fins de représentation d'une séquence d'actions. Pour permettre la modélisation d'un playbook, nous avons créé une classe (Playbook Step) dont l'objectif est d'avoir des individus qui vont contenir l'ordre des actions. Un individu de cette classe représente une étape du playbook et non pas une action. Cette première étape est ensuite reliée à une seconde étape et ainsi de suite. Chaque étape est ensuite associée à une action à l'aide d'une propriété de données. C'est une classe abstraite pour

l’humain mais nécessaire pour notre ontologie. En effet, il n’est pas possible de relier directement les actions entre elles, car si une action est incluse dans plusieurs playbooks, alors l’ordre des actions suivantes sera perdu pour ces playbooks.

C’est une contrainte propre aux ontologies et qui se retrouvera quel que soit le domaine d’application. Il n’y a, à notre connaissance, pas de solution plus efficace que celle que nous avons proposée pour pallier ce problème. Cependant, cela peut sembler contre-intuitif pour l’humain et pour une personne qui ne serait pas familière avec les ontologies.

CHAPITRE 4 INTÉGRATION AU SOAR

La suite de nos travaux porte sur l'utilisation de notre modèle ontologique comme outil de communication avec le SOAR. L'objectif de cette partie est de voir s'il est possible d'abstraire un playbook du SOAR tout en permettant son exécution. Nous reprenons donc l'ontologie présentée au chapitre précédent qui va contenir le playbook ainsi que les informations permettant la communication avec le SOAR.

4.1 Méthodologie

Cette partie présente la méthodologie que nous avons mise en place pour nos travaux d'intégration d'une ontologie à un SOAR. Pour cela, nous avons d'abord réalisé une revue du fonctionnement des outils existant ainsi que des possibilités de communication.

4.1.1 Analyse du fonctionnement des SOAR

La première étape de nos travaux de communication avec un SOAR a été de réaliser une revue des outils existants. Il existe un grand nombre d'outils disponible sur le marché, cependant très peu remplissent les conditions nécessaires pour qu'un outil soit classé comme SOAR. La très grande majorité des outils ne remplissent qu'une petite partie des conditions. Ces outils ressemblent plus à des SOA qu'à de réel SOAR. De plus, les SOAR disponibles en version communautaire ou gratuite ne permettent l'accès qu'à un panel réduit de fonctionnalités. Du côté du logiciel libre, il n'y a réellement qu'un seul outil (Shuffle) qui sort du lot mais il se rapproche plus d'une plateforme d'automatisation que d'un SOAR.

Ces diverses raisons sont très certainement la cause de l'absence de publication par le Gartner de son « Magic Quadrant ». Nous avons tout de même réalisé une revue des SOAR existants afin d'analyser leur fonctionnement et les moyens dont on dispose afin de communiquer avec cet outil.

Comme nous l'avons expliqué dans notre état de l'art, un workflow, dans le cadre de l'automatisation de la réponse, est constitué des éléments suivants :

- Une action
- Un outil qui réalise l'action
- Un artefact sur lequel on réalise cette action

Les SOAR ne dérogent pas à cette règle. Ils sont constitués d'applications (le terme peut varier selon les éditeurs mais le principe reste le même). Ces applications correspondent à des intégrations préalablement développées par les éditeurs, la communauté ou les utilisateurs de l'outil. Ces applications peuvent réaliser diverses actions. Par exemple, une intégration avec VirusTotal aura pour but de réaliser diverses actions d'enrichissement.

La mise en place de cette intégration se fait en déployant une instance de celle-ci. Cela veut dire que pour un même SOAR, nous pouvons avoir plusieurs instances de la même application. Ce concept prend sens notamment pour une entreprise qui aurait physiquement plusieurs pare-feux d'une même solution. Dans le SOAR, cela se traduira par le déploiement de plusieurs instances de l'intégration préalablement déployée pour cette solution de pare-feu.

Les instances d'intégration peuvent exécuter des actions. Ces actions sont réalisées sur des artefacts (ici aussi, le terme peut varier selon les éditeurs). Un artefact est un type de données d'un point de vue de la sécurité informatique. Cela est par exemple une adresse IP, ou un hash.

Enfin, le concept de ticket va correspondre au suivi d'une alerte. C'est donc dans ce contexte qu'une action va être exécutée.

La compréhension de ces éléments est importante, car notre modèle ontologique doit prendre en compte les concepts ci-dessus afin de communiquer avec le SOAR. En effet, notre ontologie en communiquant avec le SOAR fournit les informations suivantes :

- Le nom de l'action à réaliser
- L'instance ayant la capacité de la réaliser
- L'artefact sur lequel nous souhaitons exécuter cette action
- L'identifiant du ticket pour permettre le suivi de cette réponse

Notre analyse des SOAR nous a permis de comprendre ces fondements. Globalement, les SOAR les plus développés du marché ont mis en place ces concepts. Cette étude préliminaire nous a donc permis de réaliser une première analyse et d'avancer dans la construction de notre solution. La deuxième partie de notre revue consiste en l'analyse des possibilités de communication entre l'ontologie et le SOAR.

4.1.2 API des SOAR

Il existe globalement deux moyens de communication avec un SOAR. Le premier est de passer par l'API de l'outil. Généralement, les éditeurs mettent à disposition une API REST. Cette interface est un moyen de communiquer avec le SOAR et de récupérer des informations de celui-ci.

Une autre possibilité est de passer par le serveur d'intégration de l'outil. C'est cette interface qui est utilisée par les applications pour permettre leur intégration. C'est une possibilité que nous avons explorée cependant, elle possède des limites importantes.

Premièrement, il y a la complexité de la mise en place d'une telle solution. Cela se répercutera sur l'ontologie car elle devra contenir plus de classes et de propriétés propre à un SOAR. Il faudra par la suite étendre cette spécification à chacun des éditeurs et cela rendrait l'ontologie inutilement complexe.

Deuxièmement, le code qu'il est possible d'exécuter sur ce serveur d'intégration est limité aux librairies développées par l'éditeur du SOAR. C'est un frein important notamment, car c'est au travers de notre code que nous accédons à l'ontologie.

Enfin, l'utilisation des librairies de l'éditeur rajoute une surcouche de complexité lors de la compilation du code. Cela peut entraîner des problèmes de performances, élément que nous avons pu apercevoir lors de nos échanges avec des experts du domaine.

C'est donc pour cet ensemble d'éléments que nous nous sommes tournés vers une solution différente qui est plus adaptée à notre objectif. Notre solution ontologique communique au travers de l'API REST du SOAR afin d'orchestrer la réponse à une alerte.

Ce choix impose que notre modèle de communication puisse créer des requêtes HTTP et de les envoyer à l'API du SOAR. De plus, il doit aussi être en mesure de recevoir des informations de la part de l'outil et c'est donc pour cela que nous avons aussi déployé une API pour notre application ontologique.

4.2 Cas d'utilisation

Dans cette section, nous présentons comment notre modèle ontologique s'insère dans un SOC et s'interface avec le SOAR pour permettre l'orchestration de la réponse à incident.

La figure 4.1 montre la place qu'occupe notre application ontologique au sein d'un SOC et quels sont les liens avec le SOAR.

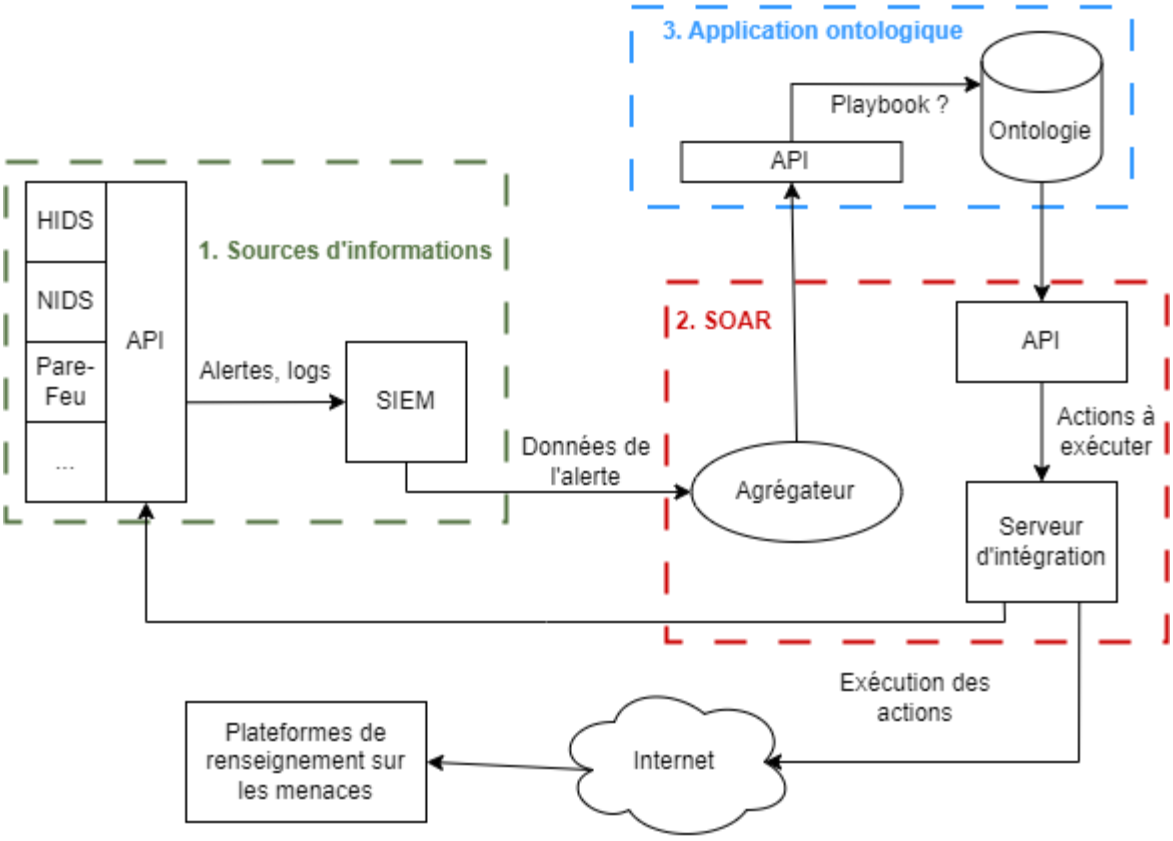

Figure 4.1 : Architecture SOAR – Ontologie

Au sein d'un SOC d'entreprise, les éléments déclencheurs des alertes sont les capteurs et le SIEM. Ce sont ces outils de sécurité qui monitorent le réseau, les machines et les serveurs d'une organisation. Ils vont détecter des événements potentiellement malveillants.

Ces événements seront remontés jusqu'au SOAR qui va agréger les données de l'alerte et créer un ticket pour permettre le suivi de celle-ci. Dans le même temps, le SOAR va déclencher notre application ontologique à l'aide de l'API que nous avons développée. Il lui fournit les données de l'événement informatique responsable de l'alerte ainsi que des informations liées au ticket. Ces dernières permettent à notre application ontologique d'exécuter les actions sur le ticket en cours d'investigation.

Avec les données reçues, l'application ontologique va interroger sa base de connaissances pour trouver un playbook de réponse à incident pour l'alerte. Le cas échéant, l'application ontologique

continue ses requêtes pour obtenir les actions à exécuter et les éléments lui permettant la communication avec le SOAR. L'ontologie contient des propriétés qui associent aux actions et aux outils leurs équivalents dans le SOAR. Ainsi, l'application ontologie peut compléter son format de requête avec les informations récupérées dans l'ontologie. Elle peut ensuite envoyer une ou plusieurs requêtes à l'API du SOAR pour exécuter la réponse.

Le SOAR va ensuite utiliser les intégrations qu'il possède afin d'exécuter les actions directement sur les outils de sécurité.

Dans ce schéma, notre ontologie se pose comme une base de connaissances comportant des playbooks de réponse à incident. Ces playbook sont constitués d'une ou plusieurs actions. Pour des fins d'expressivité du playbook et de réutilisabilité du modèle ontologique, nous avons utilisé une grammaire agnostique aux outils de sécurité. Elle est destinée à être compréhensible par des analystes, donc des humains.

Cependant, pour permettre la réutilisabilité du modèle comme orchestrateur de la réponse à incident, nous avons, dans l'ontologie, créé des propriétés qui permettent la traduction des concepts agnostiques en leur équivalent dans le SOAR. Ainsi, pour une action de réponse, l'application obtient de l'ontologie les informations nécessaires pour compléter la requête à envoyer à l'API. Ces informations correspondent au nom et à l'intégration de l'outil dans le SOAR, ainsi qu'aux informations relatives à l'action à exécuter et sur quelle donnée. Les propriétés en question contiennent l'information nécessaire pour que l'application ontologique puisse contacter le SOAR et lui demander d'utiliser une intégration existante pour réaliser la réponse. Elles font le lien entre les individus de la matrice RE&CT et les outils de sécurité de l'ontologie (langage agnostique à tout SOAR) et la représentation de ces informations dans le SOAR. Ces informations ont été ajoutées à l'ontologie.

C'est grâce à cela que notre application ontologique peut communiquer avec le SOAR.

CHAPITRE 5 VALIDATION DU MODÈLE

Dans ce chapitre, nous présentons les travaux de validation de notre modèle ontologique. Nous commençons par l'ontologie et analysons sa capacité à contenir et représenter des playbooks ainsi que des informations liées à ce concept. Nous présentons ensuite comment l'ontologie s'intègre à un SOAR et permet l'exécution d'actions sur celui-ci.

5.1 Validation de l'ontologie

Notre ontologie se veut être une base de connaissances qui va contenir les playbooks d'une entreprise. L'évaluation de notre modèle se basera sur des critères qui vont mettre en avant la capacité de l'ontologie à exprimer de façon formelle un processus de réponse à incident et à être modifiable et extensible. Pour cette étape de validation, nous avons instancié l'ontologie avec un playbook fictif de réponse à incident. Celui-ci se rapproche des playbooks que l'on peut rencontrer au sein d'entreprises et même plus généralement sur Internet.

5.1.1 Réponse aux questions de compétences

Premièrement, nous avons défini des questions de compétences en langage naturel. Afin que les requêtes soient précises et concises, nous avons défini des questions de compétences aussi élémentaires que possible. C'est au travers de ces questions de compétences converties en requêtes SPARQL que nous sommes en mesure d'interroger notre base de connaissances de playbook.

Pour des soucis de lisibilité, nous avons regroupé nos questions de compétences en différents blocs :

1. Cas d'usage de détection et de réponse à incident. Ce bloc recense les questions liées aux cas d'usage ainsi que les playbooks associés. De plus, la grammaire de MITRE ATT&CK vient se rajouter afin de communiquer des informations sur l'objectif général que couvre le cas d'usage et les playbooks qui lui sont reliés.
    a) Quels sont les cas d'usages de réponse à incident de l'entreprise ?
    b) Quels sont les playbooks de l'entreprise ?
    c) Pour un cas d'usage, quel est le playbook de réponse ?

   d) Quelle est la description du playbook de réponse à incident ?
   e) Quelle est la tactique d'attaque liée à un cas d'usage ?
   f) Quelle est la technique d'attaque liée à un cas d'usage ?

2. Actions et outils de sécurité. Ce bloc regroupe des informations sur les outils qui permettent de réaliser les actions des playbooks. De même, un exemple de la commande peut être relié à une action et expliciter la façon dont on peut exécuter ladite action.
   a) Quels outils de sécurité permettent de réaliser une action donnée ?
   b) Quelles commandes permettent de réaliser ces actions ?
   c) Quels sont les outils qui permettent d'analyser un artefact ?
   d) Quelles sont les informations que l'on souhaite obtenir d'un outil ?
3. Automatisation. Ce bloc contient des questions associées au SOAR. On peut notamment retrouver des informations sur la surface couverte par le SOAR d'un point de vue de l'automatisation. Cela permet à un analyste de savoir quelles actions ou quels playbooks sont déjà automatisés dans le SOAR.
   a) Quelles sont les actions automatisées dans le SOAR ?
   b) Quels outils sont intégrés dans le SOAR ?
   c) Quels playbooks sont automatisés dans le SOAR ?
4. Posture de sécurité. Les questions suivantes servent à améliorer la visibilité sur la posture de sécurité de l'entreprise. Un lien est fait entre les cas d'usages et la matrice MITRE ATT&CK pour savoir quelle est la surface de couverture du panel d'attaques que contient ce framework. De même, il est possible de voir quelles sont les actions les plus utilisées ainsi que les outils de sécurité associés.
   a) Quelle partie de la matrice MITRE ATT&CK est couverte pas les cas d'usages ?
   b) Quel est le top 10 des actions qui reviennent le plus parmi les playbooks ?
   c) Quel est le top 10 des outils de sécurité qui reviennent le plus parmi les playbooks ?

Le tableau 5.1 présente les réponses à ces questions de compétences.

Questions de compétences	Résultats
1.a	Une liste des cas d'usage : [UseCase : Suricata Test My Ids]
1.b	Une liste des playbooks : ['Playbook_Suricata_TestMyIds']
1.c	Le playbook pour le cas d'usage « Suricata Test My Ids » est : [Playbook_Suricata_TestMyIds]
1.d	La description pour le playbook Suricata_TestMyIds est : « This playbook is intended to answer the alert we get from suricata when running a curl command to an external domain name. »
1.e	La tactique d'attaque lié au cas d'usage Suricata_TestMyIds est : [Command_and_Control]
1.f	La technique d'attaque liée au cas d'usage Suricata_TestMyIds est : [('T1219', 'Remote Access Software')]
2.a	Les outils qui permettent d'exécuter l'action « Analyze IP » sont : ['AbuseIPdb', 'VirusTotal']
2.b	Les commandes qui permettent de réaliser l'action « block external ip address » sont :

	[‘Command_Suricata_Block_IP’]
2.c	Les outils suivants permettent d’analyser une adresse IP : ['VirusTotal']
2.d	Les informations que l’on souhaite obtenir de VirusTotal sont : ['Number of positives']
3.a	Les actions automatisées dans le SOAR sont : [‘analyze domain name’, ‘analyze ip’, ‘block external ip address’]
3.b	Les outils intégrés dans le SOAR sont : [‘VirusTotal’, ‘Suricata’]
3.c	Les playbooks automatisés dans le SOAR sont : [‘Enrichissement’, ‘Dos_Attack’]
4.a	Pour cette question le résultat est la génération d’un fichier json pour le navigateur d’attaque de Mitre. Voir figure 5.1.
4.b	Le top 5 des actions qui reviennent le plus parmi les playbooks est : [('RA2104', '1'), ('RA2105', '1'), ('RA3101', '1'), ('RA3105', '1'), ('RA5101', '1')]
4.c	Le top 5 des outils les plus utilisés parmi les playbooks est :

	[('UrlScanIo', '1'), ('AbuseIPdb', '1'), ('VirusTotal', '1'), ('Suricata', '1'), (‘Whois’, ‘1’]

Tableau 5-1 : Exemple de réponse aux questions de compétences

La figure 5.1 est une capture d’écran d’une partie du navigateur d’attaque de Mitre. Il contient la matrice MITRE ATT&CK dans sa globalité. Les cases surlignées en rouge représentent la surface de couverture des cas d’usage de l’organisation. Cette information est récupérée de l’ontologie car les cas d’usages et les playbooks sont associés aux techniques et tactiques de la matrice MITRE ATT&CK (questions 1.e et 1.f). Pour obtenir la figure ci-dessous, nous parcourons l’ontologie et pour chaque liaison entre un cas d’usage et une technique d’attaque, nous complétons la représentation de la matrice.

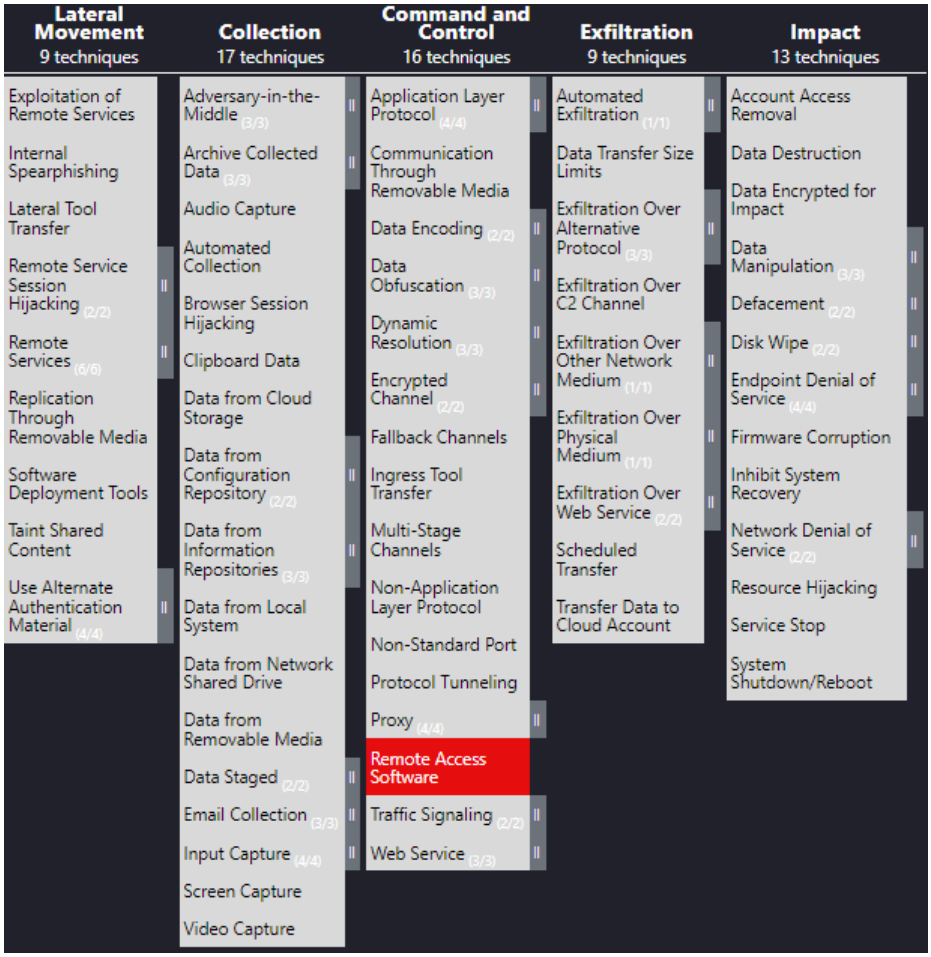

Figure 5.1 : Capture d'écran du navigateur d'attaque de Mitre

Notre ontologie contient diverses informations complémentaires aux playbooks de réponse à incident et les exemples de requêtes ci-dessus sont un moyen d'y accéder. Cependant, il est aussi possible de parcourir les instances de l'ontologie à l'aide d'un gestionnaire de base de connaissances ce qui apporte une meilleure expérience d'utilisation.

5.1.2 Modélisation de playbooks

La validation de notre modèle ontologique porte aussi sur sa capacité à contenir et représenter l'information liée à des playbooks de réponse à incident. Voici le début de représentation d'un playbook constitué de 4 étapes (figure 5.2).

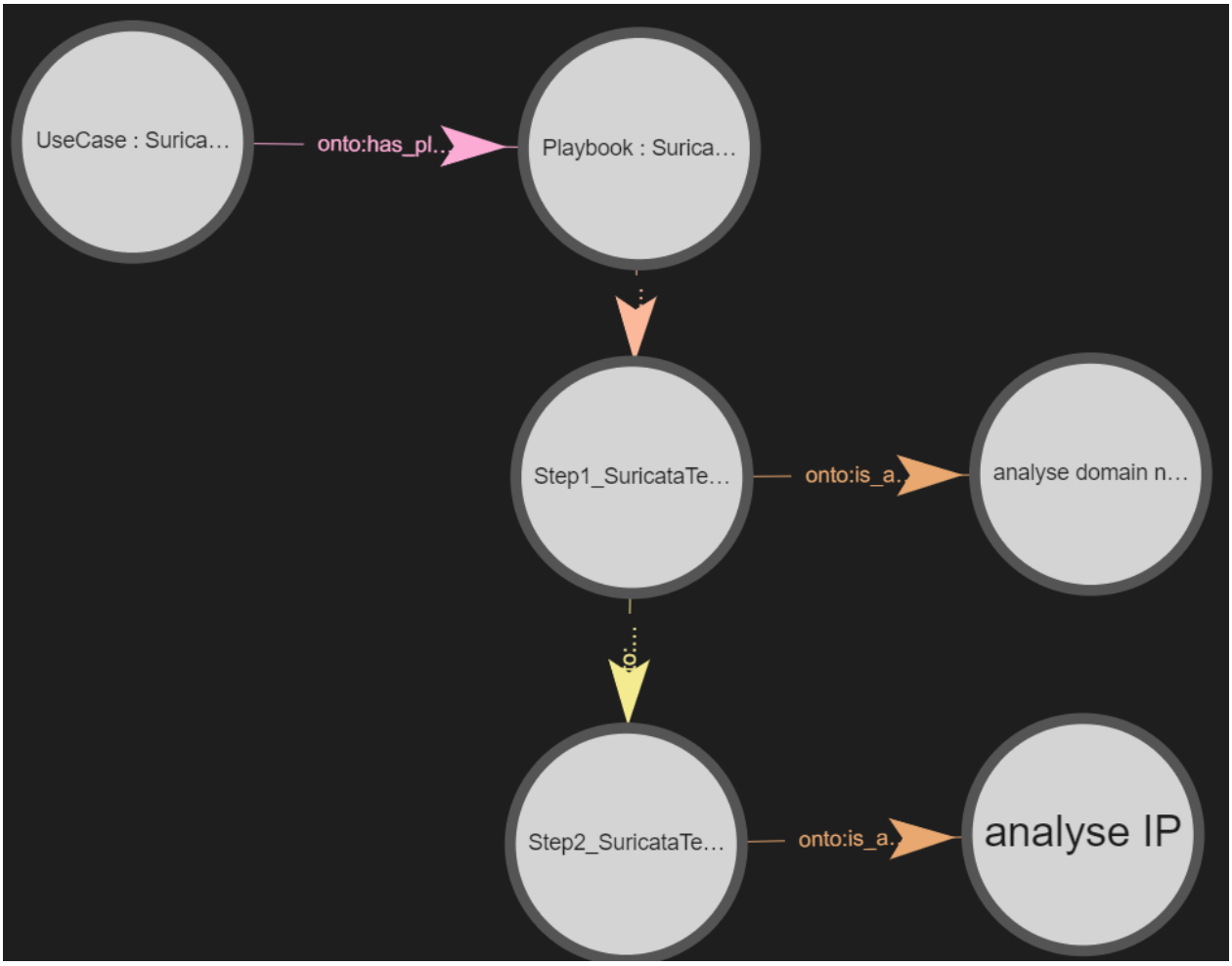

Figure 5.2 : Modélisation d'un playbook

Ici, nous pouvons voir comment est instancié un playbook dans notre modèle ontologique. Le nom de ce dernier est « Playbook : Suricata TestMyIds ». Il est composé de deux étapes :

- Étape 1 : Analyser le nom de domaine
- Étape 2 : Analyser l'adresse IP

Ce sont les deux actions qui doivent être réalisées dans le cadre de ce playbook. Le nom des actions est tiré du framework RE&CT. Pour chacune des actions, il y a dans l'ontologie une annotation qui explique l'objectif de cette dernière.

Le playbook se poursuit ensuite avec deux autres actions (figure 5.3).

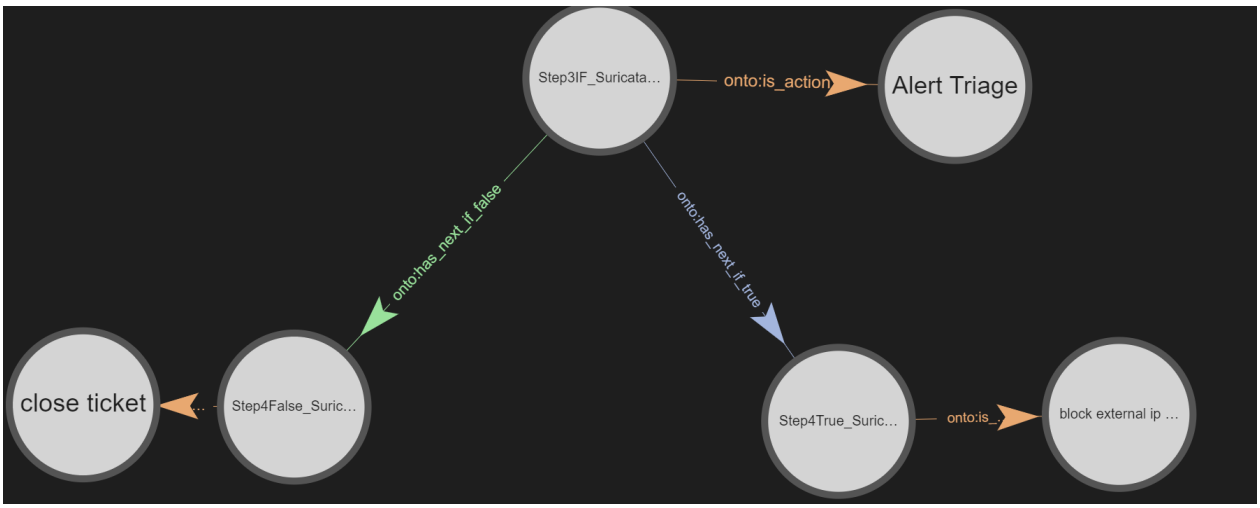

Figure 5.3 : Suite du playbook

Une fois l'analyse de l'alerte réalisée s'en vient le triage de celle-ci. Dans le cas d'un vrai positif, alors la prochaine action du playbook est de bloquer l'adresse IP externe. Dans le cas d'un faux positif, l'alerte peut être fermée.

Comme le montrent les figures 5.2 et 5.3, un playbook est défini par une suite d'étapes bien précises. La grammaire pour les actions de réponses provient de la matrice RE&CT. L'objectif derrière l'utilisation de ce framework et des ontologies pour la représentation de playbooks est de standardiser le processus de réponse ce qui peut avoir pour conséquence une uniformisation de la réponse. De plus, le fait d'avoir un format commun et ontologique permet le partage de playbooks entre organisations. C'est aussi un moyen pour une organisation de permettre la gestion de son grand panel de playbooks et d'obtenir des informations plus détaillées.

5.2 Exécution de playbooks

La suite de nos travaux concerne l'utilisation de l'ontologie pour permettre l'exécution des actions sur un SOAR. Pour valider cette partie des travaux, nous avons mis en place un environnement de test.

Voici les éléments de notre environnement :

- Une machine cliente qui simule un poste Windows 10 classique tel qu'on le rencontre dans une entreprise.
- Un outil de détection et de prévention d'intrusion du nom de Suricata. Ce dernier contient des règles de détection qui vont permettre la génération d'alertes.
- Un SIEM avec la présence de Splunk ES. L'outil permet d'agréger les données en provenance des capteurs, dans notre cas de l'IDPS Suricata.
- Splunk Phantom, un SOAR qui va être interconnecté au SIEM et à notre ontologie.
- Une plateforme de renseignement sur les menaces. Il s'agit de VirusTotal. Cette plateforme ne se situe pas dans notre environnement mais est accessible sur Internet.
- Une machine qui contient l'ontologie ainsi que son API.

Le SOAR possède un accès à Internet et une intégration préalable a été déployée sur celui-ci afin de permettre la communication avec VirusTotal.

L'environnement de test est directement inspiré des SOC d'entreprise. Nous avons souhaité le rendre le plus proche possible de ce qui se fait dans le monde de la sécurité pour avoir un scénario plus réaliste.

5.2.1 Scénario 1 : Enrichissement d'une alerte

Avant de présenter le scénario, nous souhaitons présenter les éléments sur lesquels nous nous basons.

Premièrement, nous partons du principe que pour l'alerte considérée, il y a un playbook présent dans l'ontologie pour permettre l'enrichissement. C'est en cela que nous avons au préalable instancié l'ontologie avec un playbook qui contient deux actions d'enrichissement à l'aide d'une plateforme de renseignement sur les menaces.

Deuxièmement, l'IDS possède une règle de détection pour l'événement informatique que nous allons réaliser. Pour cela, nous avons utilisé la règle du tableau 5.2 pour permettre la détection.

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ET POLICY curl User-Agent Outbound"; flow:established,to_server; http.user_agent; content:"curl/"; nocase; startswith; reference:url,www.useragentstring.com/pages/useragentstring.php; classtype:attempted-recon; sid:2013028; rev:7)

Tableau 5-2 : Règle de détection de Suricata

Enfin, pour la réalisation de ce scénario, il faut que le SOAR puisse communiquer avec VirusTotal. C'est ce que nous avons fait au préalable en déployant une intégration avec cet outil de sécurité. De même, le SOAR doit être en mesure de réaliser une requête HTTP vers l'API de notre ontologie. Pour cela, nous avons réalisé une intégration sur Splunk Phantom pour la réalisation de requêtes HTTP.

Ces éléments sont essentiels au bon déroulement du scénario. Cependant, ils ne font pas directement partie de la portée de notre recherche, car celle-ci se situe sur la réponse à incident et non les actions en amont de ce concept. Néanmoins, nous les avons mis en place pour obtenir un scénario plus réaliste.

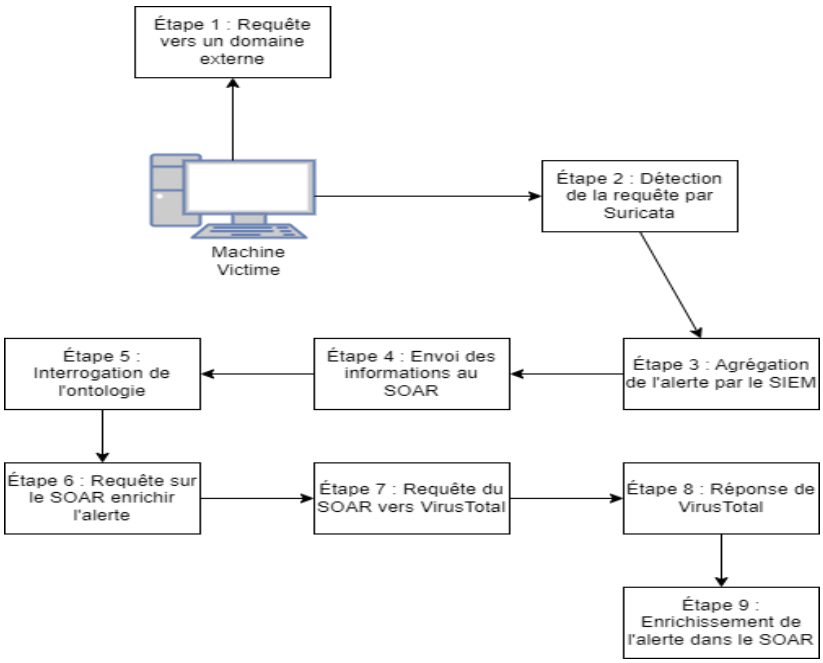

Figure 5.4 : Diagramme de déroulement du scénario d'enrichissement

Le scénario ci-dessus est constitué de plusieurs étapes (figure 5.4) :

1. Il commence par l'exécution d'une requête « curl » depuis la machine cliente vers un nom de domaine externe au réseau. Nous avons choisi de réaliser ce scénario en réalisant une requête vers le nom de domaine « example.com »
2. Cette requête va être détectée comme potentiellement malveillante par l'IDS. Cela peut en effet correspondre au comportement d'un logiciel malveillant qui souhaite accéder à son serveur de commandement et de contrôle.
3. Le SIEM va ingérer les données de l'événement ayant déclenché la règle de détection de l'IDS.
4. Le SIEM va ensuite créer une alerte et l'envoyer au SOAR.
5. Le SOAR va créer un ticket avec les données de l'alerte. Il va ensuite enclencher la communication avec l'API de l'ontologie. Les données de l'alerte sont envoyées. À l'aide des données reçues, un script en python va réaliser des requêtes SPARQL sur l'ontologie. La première requête qui est exécutée permet d'obtenir un playbook en capacité de répondre à cette alerte. Ensuite d'autres requêtes sont exécutées pour obtenir le plan d'action du playbook. Pour chaque action, l'application ontologique interroge la base de connaissance pour obtenir les outils permettant de réaliser ces actions. De même pour les informations nécessaires à la réalisation d'une requête sur l'API du SOAR. Le script va obtenir en réponse une succession de deux actions d'enrichissement, une première portant sur une adresse IP et l'autre portant sur un nom de domaine. Les actions se réalisent sur les artefacts « requestURL » et « destinationAddress ». Un outil de sécurité permettant de les réaliser est la plateforme de renseignement VirusTotal. Il va aussi obtenir les informations afin de créer une requête à envoyer au SOAR.
6. L'ontologie va ensuite envoyer une requête à l'API du SOAR afin qu'il exécute les deux actions du playbook sur VirusTotal.
7. Le SOAR va s'exécuter et envoyer des requêtes vers VirusTotal à l'aide d'une intégration préalablement établie. Les requêtes contiennent les artefacts de l'alerte que nous souhaitons enrichir.

8. VirusTotal va répondre au SOAR avec les informations qu'il possède sur les artefacts qui lui ont été envoyés.
9. Les informations sont ajoutées au SOAR et deviennent visibles dans le ticket de suivi de l'alerte. Les images suivantes montrent le résultat obtenu (Figure 5.5 et Figure 5.6)

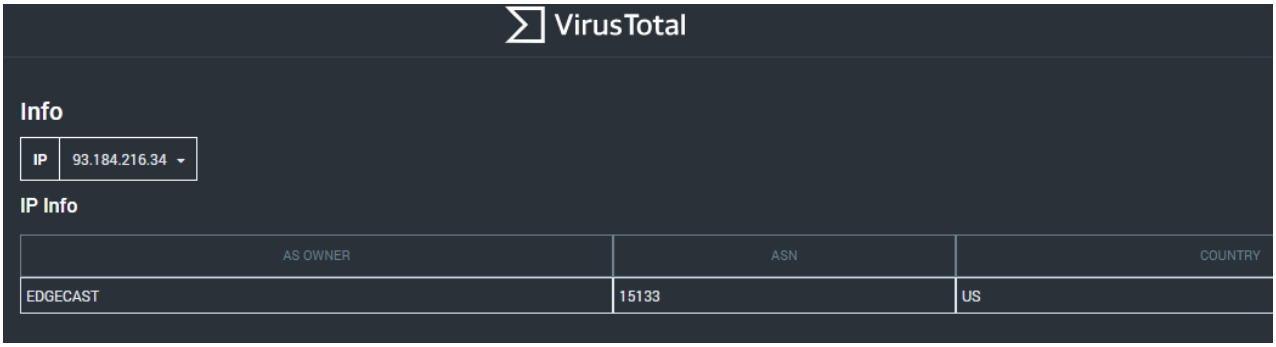

Figure 5.5 : Enrichissement de l'IP avec VirusTotal

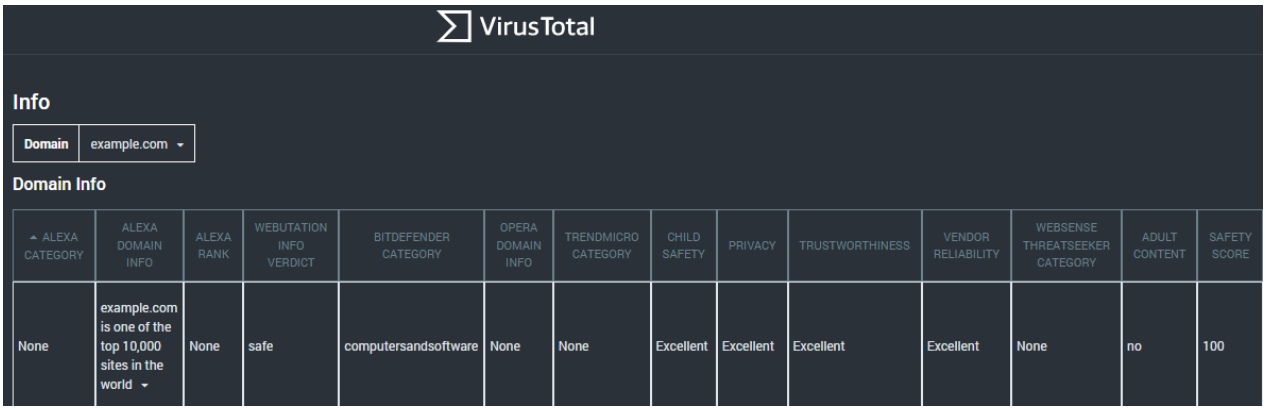

Figure 5.6 : Enrichissement de l'URL avec VirusTotal

La réalisation du scénario ci-dessus nous permet de vérifier que notre modèle ontologique est en mesure d'exécuter des actions sur un SOAR. Comme on peut le voir avec le déroulement du scénario, notre ontologie est capable de recevoir les informations de l'alerte sur son API et ensuite d'orchestrer la réponse que doit réaliser le SOAR pour cette alerte.

5.2.2 Scénario 2 : Déni de Service

Tout comme pour le scénario précédent, nous allons présenter les éléments mis en place pour son déroulement.

Premièrement, nous avons instancié l'ontologie avec un playbook de réponse aux attaques de type déni de service. Ce playbook contient une action qui est de bloquer l'adresse IP responsable de l'attaque.

Deuxièmement, nous avons ajouté une règle dans le SIEM qui va permettre la détection de l'attaque. La règle est la suivante (tableau 5.3) :

```
alert tcp any any -> $HOME_NET 8499 (msg:"LOCAL DOS SYN packet flood inbound, Potential DoS"; flow:to_server; flags: S,12; threshold: type both, track by_dst, count 10, seconds 5; classtype: misc-activity; sid:979;)
```

Tableau 5-3 : Règle de détection - Déni de Service

Cette règle va monitorer le trafic TCP ayant pour destination les machines présentes dans le réseau interne de notre environnement de test. Pour éviter la génération de fausse alerte par Suricata, nous monitorons uniquement le trafic entrant sur le port 8499. C'est sur ce port que nous avons déployé un faux serveur HTTP qui va jouer le rôle de victime de l'attaque.

Enfin, nous avons au préalable déployé une intégration avec Suricata sur le SOAR. Cette intégration permet au SOAR de communiquer avec Suricata et d'ajouter une règle.

Dans ce scénario, nous considérons Suricata comme un IDPS (Intrusion Detection and Prevention System). Il peut à la fois réaliser de la détection mais aussi de la prévention. C'est sur ce deuxième point que nous l'utiliserons pour faire de la réponse à incident en ajoutant une règle qui va interrompre l'attaque.

Le scénario d'attaque « Déni de Service » est le suivant (figure 5.5) :

Figure 5.7 : Diagramme de déroulement du scénario Déni de Service

Le scénario est constitué des étapes suivantes :

1. Un attaquant réalise une attaque en déni de service sur un serveur HTTP déployé sur une machine du laboratoire de test. Dans ce scénario, l'adresse IP de l'attaquant est : 192.168.56.1/24.
2. L'outil de détection et de prévention d'intrusion détecte l'attaque à l'aide de la règle que nous avons ajoutée.
3. Le SIEM va agréger les données de l'alerte.
4. Il va ensuite envoyer les informations au SOAR qui crée un ticket contenant les données de l'alerte.
5. Le SOAR va par la suite déclencher la communication avec l'API de l'ontologie. Il envoie les données de l'alerte ainsi que le nom et l'identifiant du ticket.
6. L'API va par la suite exécuter le script python qui a pour but de requêter l'ontologie. Pour une alerte de « Déni de Service », le script en interrogeant l'ontologie va récupérer un playbook de réponse. Celui-ci contient une action de remédiation : « Bloquer l'adresse IP ». Le script va poursuivre ses requêtes sur l'ontologie afin d'obtenir les informations lui permettant de communiquer au SOAR l'action à réaliser.

7. À l'aide des informations obtenues, le script fabrique et envoie une requête à envoyer à l'API du SOAR.
8. Le SOAR va exécuter l'action tel que demandé par le script Python.
9. Une règle permettant de bloquer l'adresse IP de l'attaquant (192.168.56.1) est ajoutée à Suricata qui fonctionne en mode IPS (tableau 5.4). Par conséquent, celui-ci est en mesure de bloquer les requêtes de l'attaquant.

drop ip 192.168.56.1 any -> $HOME_NET any (msg:Ip blocked from splunk phantom suricata integration;)

Tableau 5-4 : Règle de blocage de l'adresse IP attaquante

La réalisation des deux scénarios montre que le modèle ontologique, à l'aide de l'architecture que nous proposons, est en mesure d'orchestrer la réponse à incident sur un SOAR. Nous discutons plus en détails des résultats obtenus et des limitations de nos travaux dans le chapitre 6.

La base de connaissance obtenue intègre les scénarios ci-dessus. Elle contient aussi tous les éléments que nous avons évoqués lors de la présentation de notre modèle ontologique (Chapitre 3). Le tableau ci-dessous permet d'obtenir une idée de la taille de la base de connaissance développée.

Nombre d'axiomes	Nombre de classes	Nombre de propriété d'objet	Nombre de propriété de données	Nombre d'individus	Nombre de règle SWRL
5992	20	25	19	845	5

Tableau 5-5: Taille de la base de connaissance développée

CHAPITRE 6 DISCUSSION ET CONCLUSION

Cette partie du mémoire a pour objectif d'analyser les travaux réalisés en fonction des objectifs et hypothèses de recherche énoncés dans le premier chapitre. Nous discuterons ensuite des limitations des travaux ainsi que des futures voies de recherche qu'il serait intéressant d'explorer.

6.1 Discussions

Nous avons défini quatre objectifs de recherche lors de nos travaux. Nous analyserons et discuterons de ceux-ci dans cette partie.

6.1.1 Objectif n°1 : Analyse de l'utilisation des SOAR et de ses limites

Une partie importante de nos travaux a porté sur l'analyse des SOAR et des capacités de cet outil ainsi que sur son utilisabilité en situation réelle. Comme nous l'avons expliqué dans ce mémoire, le SOAR est un concept qui a pour but d'améliorer l'efficacité de la réponse à incident afin de répondre à un volume d'alertes toujours grandissant. Cela passe par sa capacité à permettre une gestion unifiée de l'ensemble des opérations de réponse que réalise un analyste. C'est un outil qui doit permettre de réaliser le suivi et la gestion d'une alerte, d'orchestrer et d'automatiser la réponse à incident pour cette alerte. C'est donc un outil complet qui a pour objectif d'être la plateforme principale qu'utilise un analyste dans ses activités quotidiennes. Cependant, dans les faits, la majorité des outils disponibles sur le marché ne comporte qu'une partie des fonctionnalités que doit posséder un SOAR. C'est ce que nous avons pu constater suite à notre analyse des outils disponibles sur le marché. À cela s'ajoutent les différentes limites que nous avons recensées à l'aide de notre revue de littérature, de nos discussions avec des experts du domaine et avec nos travaux. Les limites sont les suivantes :

- L'intégration des outils est complexe et il y a un manque d'intégration déjà existante et vérifiée par les éditeurs
- La flexibilité de l'outil est limitée, car les intégrations existantes sont difficilement adaptables en fonction des besoins. De plus, la programmation sur cet outil se fait souvent dans un contexte restreint aux librairies de l'outil

- Il y a un manque de maturité des processus de réponse à incident. Ceci n'est pas une contrainte du SOAR mais c'est une problématique qui se répercute directement sur cet outil, car il n'est pas possible d'automatiser des processus qui ne sont pas clairement définis.
- L'automatisation de la réponse par les workflows ne comporte que peu ou pas d'interaction avec l'humain ce qui limite l'utilisabilité de l'outil. Cela est d'autant plus vrai dans un contexte d'actions opérationnelles qui peuvent avoir d'importantes conséquences sur l'environnement informatique de l'entreprise.
- Il y a un manque de workflow prêt à l'utilisation ainsi qu'une absence de moyen de partage entre organisations.
- Le SOAR rajoute la contrainte de devoir gérer son propre parc de workflows et il n'y a pas d'outils ou de travaux existants, à notre connaissance, pour aider dans cette gestion.

Bien que prometteur, le concept de SOAR et la déclinaison de ce concept nécessitent encore de gagner en maturité.

6.1.2 Objectif n°2 : Modélisation des playbooks à l'aide d'une ontologie

Nous avons analysé la capacité d'une ontologie à représenter et contenir des informations liées aux playbooks de réponse à incident selon les sous-objectifs que nous avons définis.

Sous-objectif n°1 : Évaluer la capacité de l'ontologie à améliorer l'expressivité du playbook

Un premier critère d'évaluation de nos travaux est la capacité de l'ontologie à améliorer l'expressivité d'un playbook. En effet, le modèle ontologique que nous proposons a pour but de définir de façon formelle les playbooks de réponse à incident. Cela passe par une meilleure expressivité du playbook afin d'obtenir des processus de réponse plus matures.

Pour cela, nous avons construit notre ontologie en nous basant sur des grammaires connues dans le monde de la sécurité informatique :

- Le framework MITRE ATT&CK pour le langage lié aux techniques et tactiques d'attaques
- Le framework RE&CT en ce qui concerne le langage lié à la réponse à incident.

Ces travaux sont déjà utilisés dans le monde industriel tout comme dans le monde de la recherche. Les termes et concepts utilisés sont tout à fait compréhensibles par les personnes ciblées par nos

travaux, à savoir des analystes de réponse à incident. De plus, notre playbook est représenté à chaque étape par une action bien définie. Cette action est associée à des outils de sécurité qui ont la capacité de la réaliser et ces outils sont classés selon leur concept. De plus, la représentation d'une succession d'actions se fait selon un principe simple. Chaque étape du playbook correspond à une action de la grammaire RE&CT et possède une propriété qui spécifie l'action suivante.

De plus, nous avons présenté à plusieurs reprises l'évolution de nos travaux à des analystes de réponse à incident. Les discussions nous ont aidé à orienter la construction de nos travaux. Les critiques, négatives comme positives nous ont permis d'affiner notre modèle ontologie.

Cet ensemble d'éléments nous conforte dans l'idée que notre ontologie améliore l'expressivité du playbook, car l'ensemble de la grammaire utilisé et présenté à l'analyste se base sur des travaux connus ainsi que sur des concepts clairement définis en sécurité informatique. De plus, chaque étape est clairement définie tout comme la succession des actions au sein d'un même playbook. Nous soutenons qu'un playbook au format ontologique est plus clair que des playbooks au format textuel, tels que l'on peut en voir au sein des entreprises ou sur Internet.

Sous-objectif n°2 : Évaluer l'extensibilité et la réutilisabilité du modèle

Un critère important dans nos travaux est la capacité d'étendre le modèle en insérant de nouvelles classes ou propriétés, ou en instanciant les classes déjà existantes avec de nouveaux individus. Le modèle a été conçu de sorte qu'il soit réutilisable par d'autres travaux ontologiques. Nous avons repris autant que possible des classes déjà existantes dans d'autres ontologies. Chacune des classes est décrite et possède les annotations « label » et « comment ». Cela a pour but d'obtenir une meilleure lisibilité de l'ontologie tout en permettant sa réutilisabilité.

Concernant le concept de playbook, le peuplement de l'ontologie à l'aide de nouvelle instance passe par l'utilisation d'une grammaire connue et compréhensible aisément par des analystes en sécurité informatique. Cette grammaire est agnostique à tout outil de sécurité et est destinée à être compréhensible par l'humain. L'association des termes agnostiques à des termes spécifiques à un outil passe par des propriétés de données.

De plus, les frameworks que l'on réutilise dans l'ontologie ont été insérés automatiquement à l'aide d'un script. Cela permet à l'ontologie d'être facilement tenue à jour en cas de publication d'une nouvelle version pour les frameworks MITRE ATT&CK et RE&CT.

Cet ensemble d'éléments fait que notre modèle ontologique est réutilisable par des organisations qui souhaitent améliorer la formalisation de leurs procédures de réponses à incident. De plus, il est plus facile de mettre à jour cette base de connaissances plutôt qu'un ensemble de playbook aux formats variés et qui manquent de précision.

Enfin, le modèle est facilement extensible. Nous avons pris soin de représenter les concepts à l'aide de termes précis et nous avons autant que possible réutilisé d'autres ontologies ce qui facilite l'extensibilité du modèle en ajoutant d'autres concepts ou propriétés.

6.1.3 Objectif n°3 : Évaluer la capacité du modèle à orchestrer l'exécution du playbook ontologique sur le SOAR

La seconde partie de nos travaux porte sur la capacité de l'ontologie à orchestrer l'exécution d'un playbook de réponse à incident sur un SOAR. L'objectif était d'analyser et d'évaluer si un playbook au format ontologique pouvait contenir les informations nécessaires afin de permettre son exécution automatique sur un SOAR.

Pour la réalisation de nos deux scénarios de validation, l'ontologie a été instanciée avec deux playbooks. Le premier comporte deux actions qui ont pour objectif d'enrichir l'alerte. Le second contient une action qui vise à contenir une attaque par déni de service.

Lors du premier scénario, l'ontologie a été déclenchée par le SOAR, car une alerte a été levée pour une communication entre un ordinateur interne et un nom de domaine externe au réseau. L'API de l'ontologie a reçu les données de l'alerte. Elle a pu interroger l'ontologie pour savoir quel est le playbook à exécuter pour répondre à l'alerte. L'API a été en mesure, à l'aide des informations contenues dans l'ontologie, d'orchestrer la réponse en créant les requêtes HTTP appropriées pour ordonner au SOAR d'exécuter les deux actions d'enrichissement de l'alerte.

De même, lors du second scénario, l'API de l'ontologie a pu recevoir les données liées à l'alerte pour une attaque de déni de service. Toujours en interrogeant l'ontologie, l'API a pu obtenir les informations nécessaires afin de construire la requête HTTP permettant de demander au SOAR de bloquer l'adresse IP.

Comme nous pouvons le voir, pour chacun des deux scénarios, l'ontologie a été en mesure, en fonction des données de l'alerte, d'orchestrer le processus de réponse à incident. La communication avec le SOAR a été fonctionnelle à tous les niveaux de la phase de réponse à l'alerte.

Les résultats obtenus suite à la réalisation de nos scénarios de validation montrent qu'une ontologie est en mesure de contenir les informations nécessaires pour orchestrer la réponse à une alerte sur un SOAR.

6.1.4 Objectif n°4 : Analyser l'utilisabilité du modèle pour permettre l'interopérabilité

Notre validation concernant l'exécution des playbooks a été réalisée sur le SOAR Splunk Phantom. Cependant, nous avons réalisé une analyse du fonctionnement des SOAR en général ainsi que des moyens de communication dont on dispose avec cet outil. Bien que nous n'ayons pas réalisé de preuve de concept avec d'autres éditeurs, notre analyse nous pousse à soutenir que nos travaux permettent l'interopérabilité du playbook.

En effet, l'ontologie contient l'ordre des actions sous une forme agnostique, et les propriétés de données permettent d'associer ces actions aux spécificités que peut avoir chaque éditeur. De plus, la communication SOAR – Ontologie dans nos travaux est réalisée au travers de l'API REST. Cela rajoute une couche de standardisation, car les différences au niveau des requêtes HTTP se feront uniquement sur les paramètres que contiennent les données envoyées au format json. Des propriétés de données peuvent être ajoutées pour contenir ces informations nécessaires à la construction de la requête HTTP.

6.2 Limitations des travaux

Nous avons recensé plusieurs limitations à nos travaux.

Premièrement, si pour une action donnée, il y a plusieurs outils en capacité de la réaliser, le choix dans l'utilisation de l'outil pour l'exécution est tout simplement le premier élément de la requête SPARQL. Il n'y a actuellement pas, dans nos travaux, de moyens d'estimer quel outil serait le plus adapté au vu du contexte de l'alerte. Une piste à explorer serait d'assigner aux outils de sécurité un score pour chacune des actions qu'il peut réaliser. Si plusieurs outils peuvent réaliser une même action, alors l'ontologie choisit de l'exécuter sur l'outil qui possède le meilleur score.

Deuxièmement, la validation de nos travaux s'est faite sur des playbooks constitués de peu d'actions. Bien que multiplier le nombre d'actions à exécuter ne semble pas avoir d'impact sur le temps d'exécution, nous n'en avons pas la certitude et donc nous émettons une critique à ce sujet.

Une analyse de la capacité de l'ontologie à orchestrer la réponse dans des scénarios complexes est nécessaire pour valider son comportement dans cette situation.

Enfin, la difficulté et le temps nécessaire pour instancier, lors de la mise en place de l'ontologie, l'ensemble des playbooks d'une entreprise est très variable. Pour une entreprise possédant un grand nombre de playbook, cela peut être un travail lent et fastidieux. Nous présentons cependant, dans la section des travaux futurs, des idées pouvant solutionner ce problème.

6.3 Travaux Futurs

Le concept des SOAR est encore récent et l'outil nécessite encore de gagner en maturité. Bien que nos travaux soient une première étape vers une amélioration de l'automatisation de la réponse à incident, il y a plusieurs pistes d'amélioration.

Premièrement, une piste à explorer serait d'analyser la possibilité d'utiliser des techniques de « Traitement Automatique du Langage Naturel » sur des documents textuels servant à décrire des playbooks de réponse à incident. L'objectif serait d'observer s'il est possible, à l'aide de ces techniques, d'instancier automatiquement notre ontologie à partir des playbooks au format textuel. Cela permettrait la réutilisabilité de travaux déjà existants, ainsi qu'une transition plus simple pour les entreprises possédant déjà un grand nombre de playbooks.

Deuxièmement, les SOAR présents sur le marché et possédant des fonctionnalités de plateforme de gestion des alertes ont chacun leur propre format de données pour décrire les événements informatiques. Cela a pour conséquence que pour un même événement, la description de celui-ci sera différente entre éditeurs. Une standardisation de ces formats pour permettre le passage de l'un à l'autre serait une suite intéressante à nos travaux d'exécution de playbooks à l'aide de l'ontologie. De plus, cela serait une avancée certaine vers l'obtention d'une interopérabilité des playbooks et des travaux portant sur les SOAR.

Troisièmement, des travaux intéressants porteraient sur l'utilisation de notre ontologie comme standard pour permettre le transfert de playbooks d'un SOAR à un autre. De plus, il serait pertinent d'analyser les librairies des différents éditeurs des SOAR afin de déterminer s'il serait possible de réaliser un transfert de code d'un SOAR à un autre. Cela serait une avancée de plus vers l'interopérabilité des travaux sur les SOAR.

Quatrièmement, durant nos travaux, nous n'utilisons que très peu la capacité d'inférence des ontologies. Nous nous sommes principalement limités à l'utilisation de la logique descriptive afin de décrire les classes qui constituent notre ontologie. Nous avons aussi utilisé l'inférence autour des propriétés afin d'obtenir une ontologie plus expressive. Cependant, nous n'avons pas exploré l'utilisation de la capacité d'inférence des ontologies ou d'une autre méthode d'intelligence pour permettre d'adapter notre playbook ontologique au contexte de l'alerte. Cela pourrait permettre d'aller plus loin en adaptant la réponse au contexte plutôt que de l'appliquer de façon générique et serait un progrès et une amélioration de l'automatisation de la réponse à incident.

Cinquièmement, notre ontologie répond à une alerte de sécurité en exécutant un playbook de réponse à incident. Bien que cela soit déjà utile, il serait intéressant d'approfondir nos travaux afin de coordonner l'exécution de plusieurs playbook pour faire face à une attaque coordonnée. De tel travaux seraient une avancée importante aux notre et permettraient encore plus d'améliorer la capacité d'automatisation des SOAR.

Enfin, nous n'avons pas pu faire tester notre ontologie à des analystes en situation réelle et cela faute de temps. Nous émettons cette idée en tant que travail futur car elle permettrait d'améliorer notre modèle ontologie en fonction du retour d'expérience des analystes.

RÉFÉRENCES

[1] Techslang, 2021, “What is Alert Fatigue?”, [En ligne]. Disponible : https://www.techslang.com/definition/what-is-alert-fatigue/

[2] Gartner, “Market Guide for Security Orchestration, Automation and Response Solutions”, Gartner Research, 2020.

[3] Palo Alto Networks, 2020, “The state of SOAR report”. [En ligne]. Disponible sur: https://www.paloaltonetworks.com/cortex/xsoar-state-of-soar-report-2020

[4] Orange Cyberdefense, “SOAR : Quelles conclusions pour 2020 ?” , *France*. https://orangecyberdefense.com/fr/insights/blog/threat_management/soar-quelles-conclusions-en-2020/

[5] D. Schlette, M. Caselli, et G. Pernul, “A Comparative Study on Cyber Threat Intelligence: The Security Incident Response Perspective ", *IEEE Commun. Surv. Tutor.*, vol. 23, n° 4, p. 2525-2556, 2021, doi: 10.1109/COMST.2021.3117338.

[6] P. Cichonski, T. Millar, T. Grance, et K. Scarfone, “Computer Security Incident Handling Guide : Recommendations of the National Institute of Standards and Technology”, National Institute of Standards and Technology, NIST SP 800-61r2, août 2012. doi: 10.6028/NIST.SP.800-61r2.

[7] Gartner, “How to Create an Incident Response Plan”, *Gartner*. 2021.

[8] Blue Teams Academy, 2021, “Module 1 - Incident Response and Security Operations Fundamentals - Blue Teams Academy”. [En ligne], Disponible : https://www.blueteamsacademy.com/incident-response-and-security-operations-fundamentals/

[9] Crowdstrike, 2022, "Étapes De La Réponse À Incident ”, [En ligne], Disponible : https://www.crowdstrike.fr/cybersecurity-101/incident-response/incident-response-steps/ (consulté le 5 octobre 2022).

[10] Oasis Open, 2021, “CACAO Security Playbooks Version 1.0”, [En ligne], Disponible : https://docs.oasis-open.org/cacao/security-playbooks/v1.0/cs02/security-playbooks-v1.0-cs02.html

[11] IACD, “Introduction to IACD Playbook”. [En ligne]. Disponible sur: https://static1.squarespace.com/static/5a94b67ff93fd440f0516297/t/5b3fb74170a6ad89b74de3b0/1530902339037/Introduction+to+IACD+Playbook+.pdf

[12] IACD, "Intro to Playbooks and Workflows”, [En ligne], Disponible : https://www.iacdautomate.org/intro-to-playbooks-and-workflows

[13] RE&CT, “RE&CT Framework”, [En ligne], Disponible : https://atc-project.github.io/atc-react/

[14] Gartner, "Definition of Security Orchestration, Automation and Response (SOAR)", *Gartner Information Technology Glossary*. [En Ligne], Disponible : https://www.gartner.com/en/information-technology/glossary/security-orchestration-automation-response-soar

[15] C. Islam, M. A. Babar, et S. Nepal, “A Multi-Vocal Review of Security Orchestration”, *ACM Comput. Surv.*, vol. 52, n° 2, p. 1-45, mai 2019, doi: 10.1145/3305268.

[16] S. Norem, A. E. Rice, S. Erwin, R. Bridges, S. Oesch, et B. Weber, “A Mathematical Framework for Evaluation of SOAR Tools with Limited Survey Data” , *Computer Security. ESORICS 2021 International Workshops,* vol. 13106*, p. 557-575,* 2021, doi: 10.1007/978-3-030-95484-0_32.

[17] Swimlane, 2022, “What is SOAR?”, [En ligne], Disponible : https://www.swimlane.com/blog/what-is-soar

[18] Oliver Rochford, “Why Is Automation Failing to SOAR?”, *Securonix*. [En ligne], Disponible : https://www.securonix.com/blog/why-is-automation-failing-to-soar/

[19] C. Tozzi, 2022, “The Drawbacks of a SOAR”, *The New Stack*, [En ligne], Disponible : https://thenewstack.io/the-drawbacks-of-a-soar/

[20] A. Swan, « SIGMA Rules: The Beginner’s Guide », *SOC Prime*, (2022). [En ligne], Disponible : https://socprime.com/blog/sigma-rules-the-beginners-guide/.

[21] T. R. Gruber, “Toward principles for the design of ontologies used for knowledge sharing?”, *Int. J. Hum.-Comput. Stud.*, vol. 43, n° 5, p. 907-928, nov. 1995, doi: 10.1006/ijhc.1995.1081.

[22] R. Studer, V. R. Benjamins, et D. Fensel, “Knowledge engineering: Principles and methods”, *Data Knowl. Eng.*, vol. 25, nº 1, p. 161-197, mars 1998, doi: 10.1016/S0169-023X(97)00056-6.

[23] W3C, “Inference”, [En ligne], Disponible : https://www.w3.org/standards/semanticweb/inference

[24] W3C, “SWRL: A Semantic Web Rule Language Combining OWL and RuleML”, [En ligne], Disponible : https://www.w3.org/Submission/SWRL/

[25] W3C, “SPARQL 1.1 Overview”, [En ligne], Disponible : https://www.w3.org/TR/2013/REC-sparql11-overview-20130321/

[26] M. Uschold et M. King, “Towards a Methodology for Building Ontologies”, Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence, 1995. [En ligne], Disponible : https://www.aiai.ed.ac.uk/publications/documents/1995/95-ont-ijcai95-ont-method.pdf

[27] S. Peroni, “SAMOD : an agile methodology for the development of ontologies”, Digital And Semantic Publishing Laboratory, Department of Computer Science and Engineering, University of Bologna, Bologna, Italy”. [En ligne], Disponible : https://essepuntato.it/samod/

[28] S. Malenfant-Corriveau, “Proposition d’une méthode de développement d’ontologie pour un système expert en sécurité", Master's thesis, École Polytechnique de Montréal, Montréal, QC, 2017, [En ligne], Disponible : https://publications.polymtl.ca/2923/

[29] H. Booth, “Draft NISTIR 8138, Vulnerability Description Ontology (VDO)", NIST, 2016, [En ligne], Disponible : https://csrc.nist.gov/csrc/media/publications/nistir/8138/draft/documents/nistir_8138_draft.pdf

[30] J. A. Wang et M. Guo, “OVM: an ontology for vulnerability management”, in *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research Cyber Security and Information Intelligence Challenges and Strategies - CSIIRW ’09*, Oak Ridge, Tennessee, 2009, p. 1. doi: 10.1145/1558607.1558646.

[31] G. B. Moreira, V. M. Calegario, J. C. Duarte, et A. F. P. D. Santos, « CSIHO: An Ontology for Computer Security Incident Handling », Instituto Militar de Engenharia, 2018. [En ligne]. Disponible sur: https://sol.sbc.org.br/index.php/sbseg/article/download/4239/4170/

[32] Z. Syed, A. Padia, T. Finin, L. Mathews, et A. Joshi, « UCO: A Unified Cybersecurity Ontology », AAAI Workshop on Artifical Intelligence for Cyber Security, févr. 2016. [En ligne], Disponible : https://www.researchgate.net/publication/287195565_UCO_A_Unified_Cybersecurity_Ontology

[33] A. Applebaum, S. Johnson, M. Limiero, et M. Smith, "Playbook Oriented Cyber Response", in *2018 National Cyber Summit (NCS)*, juin 2018, p. 8-15. doi: 10.1109/NCS.2018.00007.

[34] H. Hutschenreuter, S. Çakmakçı, C. Maeder, et T. Kemmerich, "Ontology-based Cybersecurity and Resilience Framework", in *Proceedings of the 7th International Conference on Information Systems Security and Privacy*, 2021, p. 458-466. doi: 10.5220/0010233604580466.

[35] Islam, C., Babar, M.A., Nepal, S. (2019). Automated Interpretation and Integration of Security Tools Using Semantic Knowledge. In: Giorgini, P., Weber, B. (eds) Advanced Information Systems Engineering. CAiSE 2019. Lecture Notes in Computer Science(), vol 11483. Springer, Cham. https://doi.org/10.1007/978-3-030-21290-2_32

[36] Oasis Open, « Introduction to STIX ». [En ligne], Disponible : https://oasis-open.github.io/cti-documentation/stix/intro.html.