Comprendre les menaces sophistiquées

Aimad Berady

DOCTORAT BRETAGNE LOIRE / MATHSTIC

CentraleSupélec

THÈSE DE DOCTORAT DE

CENTRALESUPÉLEC

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

Par

Aïmad BERADY

Comprendre les menaces sophistiquées

Intentions, moyens, manières et connaissances convergentes des adversaires

Thèse présentée et soutenue à Rennes, le 10 novembre 2022
Unité de recherche : Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA)
Thèse N° : 2022CSUP0004

Rapporteurs avant soutenance :

Marie-Laure POTET	Professeur des universités, Ensimag / Grenoble INP
Aurélien FRANCILLON	Professeur, EURECOM

Composition du Jury :

Président :	Vincent NICOMETTE	Professeur des universités, INSA Toulouse
Examinateurs :	Jérôme FRANÇOIS	Chercheur, Inria Nancy Grand-Est
	Éric FREYSSINET	Général de brigade, Gendarmerie nationale
Directeur de thèse :	Valérie VIET TRIEM TONG	Professeur, CentraleSupélec
Encadrants :	Gilles GUETTE	Maître de conférences, Université Rennes 1
	Mathieu JAUME	Maître de conférences, Sorbonne Université

Invité(s) :

Christophe BIDAN	Professeur, CentraleSupélec

United we conquer.

Devise des Commandos marine.

Liste des publications

Articles dans des revues avec comité de lecture

— **Aimad Berady**, Mathieu Jaume, Valérie Viet Triem Tong et Gilles Guette : ***From TTP to IoC: Advanced Persistent Graphs for Threat Hunting***. *IEEE Transactions on Network and Service Management (TNSM) - Special Issue on Latest Developments for Security Management of Networks and Services*, juin 2021. DOI : 10.1109/TNSM.2021.3056999, HAL Id : hal-03131262

— **Aimad Berady**, Mathieu Jaume, Valérie Viet Triem Tong et Gilles Guette : ***PWNJUTSU: A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns***. *IEEE Transactions on Network and Service Management (TNSM) - Special Issue on Recent Advances in Network Security Management*, juin 2022. DOI : 10.1109/TNSM.2022.3183476 HAL Id : hal-03694719

Communications avec actes

— **Aimad Berady**, Valérie Viet Triem Tong, Gilles Guette, Christophe Bidan et Guillaume Carat : ***Modeling the Operational Phases of APT Campaigns***. *IEEE Conference on Computational Science & Computational Intelligence (CSCI)*, décembre 2019, USA. DOI : 10.1109/CSCI49370.2019.00023 HAL Id : hal-02379869

Communications sans actes

— **Aimad Berady**, Valérie Viet Triem Tong, Gilles Guette et Mathieu Jaume : **Caractérisation tactique d'un attaquant évoluant dans un réseau compromis**. *Journée thématique du GT "Sécurité des Systèmes, Logiciels et Réseaux" (SSLR) 2021 sur la sécurité des réseaux*, mai 2021, en ligne, France.
URL : `https://gtsslr21-reseau.sciencesconf.org/353075`

Remerciements

La finalisation et le succès de cette thèse n'auraient pas été possibles sans les soutiens que vous m'avez fournis, sans la force et le courage que vous m'avez donnés, sans le temps que vous m'avez consacré et sans la confiance sans faille que vous m'avez accordée. Au travers de ces quelques lignes, je souhaite sincèrement vous remercier d'avoir pris part à ce projet depuis les premiers balbutiements.

Je me dois de commencer par remercier **Erwan A.** (docteur XSS). Il a été celui qui m'a motivé à me lancer dans cette aventure et qui m'a permis de rencontrer ceux qui sont ensuite devenus mes encadrants : ma directrice de thèse **Valérie Viet Triem Tong** (cheffe du "Département Commando", toujours prête à s'aventurer en dehors des sentiers battus) ; mon encadrant **Gilles Guette** (habile comme un cavalier ♘, il anticipe les coups pour que le roi ♔ ne soit jamais en prise) ; et mon co-directeur **Christophe Bidan** (effectivement une thèse, c'est un plus un marathon qu'un 10). Quelques mois plus tard, j'ai été enchanté de faire la rencontre de **Mathieu Jaume** (`$\machine\exec\rrule\cmp{}i$`), qui a ensuite rejoint mon aventure en tant que co-encadrant. Merci à tous les quatre d'avoir compris mon logiciel interne et ainsi d'avoir été attentifs, de m'avoir testé, mis au défi, agacé puis consolé et rassuré (la câlinothérapie, ça marche) ; mais surtout d'avoir toujours répondu présent et de m'avoir laissé une totale liberté de manœuvre. J'ai énormément appris à votre contact. À ce titre, j'ai toujours en tête une remarque faite une des premières semaines, alors que je travaillais sur mon état de l'art et que je cherchais des pistes à suivre : "*Si tu as une idée, dis-toi que quelqu'un d'autre l'a aussi eue et qu'il l'a sûrement déjà publiée*". Sur le coup j'ai été vexé, puis... je l'ai vérifié chaque jour et sur chaque idée "inédite" que j'ai cru avoir. Je vous ai donc suivi et j'ai appris à faire mieux, à penser plus loin et surtout à tourner sept fois les idées dans ma tête avant de me lancer. Pour cela, l'équipe projet **CIDRE** (IRISA) m'a véritablement permis de m'élever intellectuellement (d'ailleurs, je dois avouer que je ne comprends toujours pas la moitié de ce qui est écrit sur le tableau de la salle de pause).

Dans le cadre du projet PWNJUTSU, qui m'a occupé pendant la majeure partie de ma thèse, j'ai pu profiter de l'appui de l'**IRSN**, grâce à la confiance d'**Olivier F.** et à l'investissement de **Vincent C.** J'ai aussi pu compter sur le précieux renfort de **Benoît F.**,

sans qui l'implémentation de cette plateforme n'aurait pas été possible. Enfin, **Romain L.** et ses équipes ont rapidement compris mes besoins et ils m'ont permis de travailler avec des experts de la communauté **YesWeHack** dans des conditions exceptionnelles.

Je remercie aussi chaleureusement mes relecteurs personnels **Baptiste B.** et **Chloé H.** pour leur disponibilité, leur ouverture d'esprit et leur expertise. Votre regard extérieur sur mes travaux et vos validations m'ont toujours rassuré et j'ai pu poursuivre sereinement.

Tout au long de cette période, j'ai également pu compter sur de très nombreux soutiens parmi mes collègues, qui m'ont amené à "transformer les contraintes en opportunités" et qui m'ont ensuite permis de me lancer dans ce projet délirant en complément de mon activité professionnelle. Je pense également à tous ceux avec qui j'ai pu échanger à propos de ma thèse et qui m'ont aidé, parfois en me fournissant un pointeur, en contredisant une de mes assertions ou simplement en me questionnant et en me permettant d'identifier des non-sens dans mon discours. Je ne vous oublie pas et je vous suis extrêmement reconnaissant, TOUS ! Cette page n'est juste pas le meilleur endroit pour vous remercier...

Pour conclure cette thèse avec autant de fierté qu'elle a commencée, j'ai eu l'immense privilège de la faire évaluer par un jury de haut vol : **Marie-Laure POTET**, **Aurélien FRANCILLON** comme rapporteurs ; ainsi que **Vincent NICOMETTE**, **Jérôme FRANÇOIS** et **Éric FREYSSINET** comme examinateurs. Je vous remercie de m'avoir consacré une partie de votre temps, que je sais précieux.

Pour finir, j'ai une pensée affectueuse pour mes amis les plus proches, notamment **Arnaud**, **Edine** et **Karim**, qui m'ont donné la force de me battre et de mener à bien ce projet ; mais aussi pour ma famille, qui a toujours suivi mes choix. Mes derniers mots seront pour **ma tribu, F., S. et Y.** (les SYNS) : merci infiniment pour votre patience. C'est terminé.

Sommaire

Chapitre 1

Contexte général

1.1 Introduction

Au cours de l'année 2021, l'Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI) a eu connaissance de 1082 intrusions avérées dans des entités françaises [1]. Ce chiffre en constante hausse fait référence aux campagnes d'attaques détectées par les défenseurs. Pourtant, les attaquants, motivés par des gains financiers, des vols d'informations sensibles (espionnage), ou encore des manœuvres de déstabilisation (destruction ou désinformation), continuent à faire chaque jour de nouvelles victimes.

Si la chasse des attaquants dans les Systèmes d'Information (SI) semble être un éternel recommencement, la posture et le niveau de compréhension des défenseurs ont considérablement évolué ces dernières années. En effet, grâce à l'émergence de la discipline de *Cyber Threat Intelligence* (CTI) [2], la communauté ne se contente pas de prendre connaissance des nouvelles techniques mises en œuvre par les attaquants et de se préparer à y faire face, mais elle s'attache également à prendre de la hauteur sur la nature de la menace, notamment grâce aux travaux de caractérisation des intentions et des modes opératoires des attaquants. Aussi, il est aujourd'hui admis qu'un grand nombre de concepts empruntés au monde militaire sont transposables pour les opérations conduites dans le cyberespace [3]. Il est donc fréquent que les experts de la discipline s'inspirent de la doctrine militaire afin de mieux conceptualiser le champ de bataille, l'assaillant et ses offensives.

L'objectif de cette thèse est de comprendre les menaces sophistiquées en décrivant les perceptions croisées des adversaires (attaquant et défenseur). Pour cela, nous plaçons au cœur de notre réflexion les intentions qui les animent (tactiques), les moyens qu'ils mettent en œuvre (techniques), leurs manières d'agir (procédures) et leurs connaissances convergentes alimentées tout au long de leurs opérations respectives.

1.2 Menaces sophistiquées (APT)

Dans le domaine de la sécurité informatique, la communauté scientifique et les professionnels ont pris conscience depuis quelques années de l'existence de menaces dites "persistantes et avancées", communément appelées *Advanced Persistent Threat* (APT). Elles se concrétisent par des attaques qui ciblent ou impliquent régulièrement des États-nations [4] ou des entreprises de toutes nationalités. Burita *et al* [5] ont recensé les groupes APT dont la communauté CTI reconnaît leurs nationalités voire leurs affiliations à des États [6].

Le mode opératoire des APT a depuis été repris par des acteurs avec des intentions et des profils différents. Il s'agit notamment de groupes cybercriminels évolués qui pratiquent ces attaques afin de déployer des rançongiciels chez leurs victimes, comme l'ANSSI l'a récemment précisé [1]. Dès 2011, le *National Institute of Standards and Technology* (NIST) a défini les APT dans une publication spéciale [7].

The Advanced Persistent Threat

(i) pursues its objectives repeatedly over an extended period of time ;
(ii) adapts to defenders' efforts to resist it ; and
(iii) is determined to maintain the level of interaction needed to execute its objectives.

Cette définition souligne trois aspects fondamentaux pour ce type de menace :
— la durée de l'attaque et l'atteinte de l'objectif de façon répétée ;
— la capacité à s'adapter et à contourner les systèmes de défense ; et
— la détermination et la motivation à atteindre les objectifs.

Elle doit être complétée par une autre définition présente dans la littérature [8] [9] [10] :
— ***Advanced*** : fait référence à la furtivité et au fait que les victimes soient spécifiquement ciblées par ces attaques (par opposition aux attaques totalement opportunistes). De plus, ces attaquants travaillent en groupes organisés et sont animés par des commanditaires ou des intérêts communs (cas particulier de l'*hacktivisme*).
— ***Persistent*** : indique que ces attaques s'inscrivent sur le temps long. Les attaquants qui ont pris pied dans un SI peuvent y rester plusieurs mois voire plusieurs années sans que leurs actions ne soient détectées.
— ***Threat*** : rappelle que la présence de ces intrus dans le SI n'est pas anodine et qu'ils sont particulièrement motivés par l'atteinte d'un objectif. Cet objectif opérationnel pour les attaquants est logiquement considéré comme un événement redouté par les victimes (vol de données sensibles, destruction d'équipements, rançonnage de données informatiques).

Toutefois, comme l'évoque l'ANSSI [1], on observe une professionnalisation et une réorganisation des groupes cybercriminels, qui ont finalement tendance à se comporter sur les SI de leurs victimes comme des groupes APT. Par conséquent, nous utiliserons le terme générique **"sophistiqué"**, adjectif usuellement employé par Kaspersky [11] pour désigner ces intrusions conduites par des individus motivés par l'atteinte d'un objectif précis.

Afin de représenter la progression de ces attaques sophistiquées, la communauté scientifique et l'industrie ont concouru à définir des représentations du cycle de vie suivi par les attaquants pendant leurs opérations. Ainsi, les mesures d'atténuation des risques pour lutter contre ces menaces font souvent référence aux modèles appelés *Kill Chains*.

1.3 Modélisations du cycle de vie d'une campagne d'attaque (*Kill Chains*)

Alors que plusieurs visions s'opposent ou se complètent, comme le montrent Tatam *et al* dans leur enquête scientifique [12], la majorité des modèles font référence au plus populaire d'entre eux, celui introduit par la compagnie américaine Lockheed Martin : *Cyber Kill Chain*. Ce concept emprunté à la tactique militaire, qui l'utilisait pour décrire le déroulement d'une offensive vers la destruction d'un objectif [13], a popularisé la notion de phasage tactique d'une opération cyber offensive.

Pour répondre au besoin naissant d'analyse, de compréhension et de réponses aux attaques sophistiquées auxquelles les organisations commençaient à faire face, la société Lockheed Martin a décrit pour la première fois en 2011 [14] sa désormais célèbre *Cyber Kill Chain*® et ses différentes phases (Figure 1.1). Dans cette contribution, les auteurs ont également proposé des contremesures visant à stopper une attaque pour chacune de ces phases. Cependant, ce modèle, à cause de sa linéarité et du fait qu'il se focalise sur l'accès initial au SI (*Initial Foothold*), n'est pas pertinent pour illustrer les actions d'un attaquant qui progresse à l'intérieur d'un SI compromis en quête de ses objectifs finaux. S'il est à ce jour particulièrement adapté pour l'étude de *malware*, il retranscrit difficilement le cycle de vie que suit un attaquant au cours d'une campagne, notamment les éventuelles régressions qu'il peut subir ou la répétition des actions opérées. Dans le Chapitre 2, nous proposons un modèle qui vient pallier ce manque.

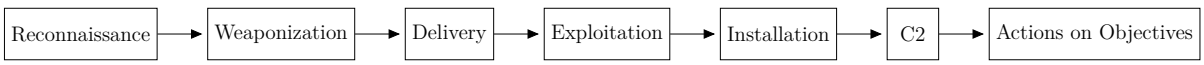

FIGURE 1.1 – Phases de la *Cyber Kill Chain* de Lockheed Martin

Toujours en 2011, il est intéressant d'observer que, dans une publication proposant une définition globale pour décrire les attaques APT, Command Five Pty a introduit [15] pour la première fois la notion de *maintenance* dans son modèle (Figure 1.2). Derrière cette notion se cache le besoin de l'attaquant de persister sur le temps long et d'être résiliant face aux événements imprévus qui pourraient venir déranger ses actions. Sa forme circulaire suggère également la possibilité de revenir dans un état antérieur dans le cas où l'accès à l'objectif aurait été perdu.

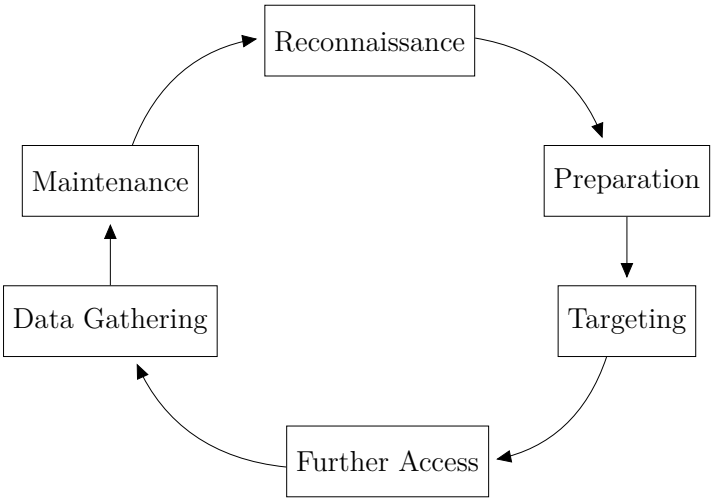

FIGURE 1.2 – Phases du modèle de Command Five Pty

C'est tout naturellement que la combinaison des modèles linéaires et circulaires ont donné naissance aux modèles à boucles. Dès 2013, Mandiant a utilisé des boucles (Figure 1.3) pour représenter dans leurs rapports les cycles de vie des attaquants [16]. Cela a permis de restaurer la notion de répétitivité, mise de côté par Lockheed Martin dans son article fondateur. Par la suite, en 2017, Pols (Fox-IT) a présenté dans sa thèse [17] l'idée d'une *Unified Kill Chain*, qu'il a construite en agrégeant les modèles précédemment publiés et deux cas d'études (*Red Team* de Fox IT et APT28). Sa *Kill Chain* consiste en dix-sept étapes optionnelles réparties sur trois boucles (Figure 1.4).

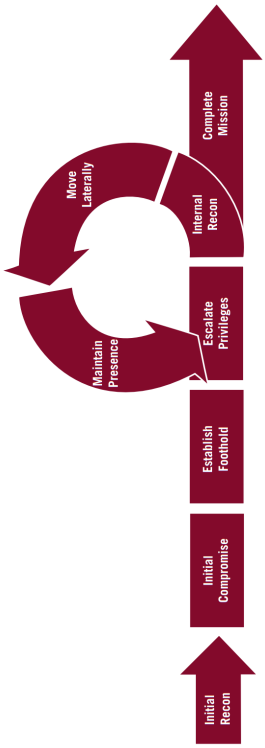

FIGURE 1.3 – *Mandiant's Attack Lifecycle Model*

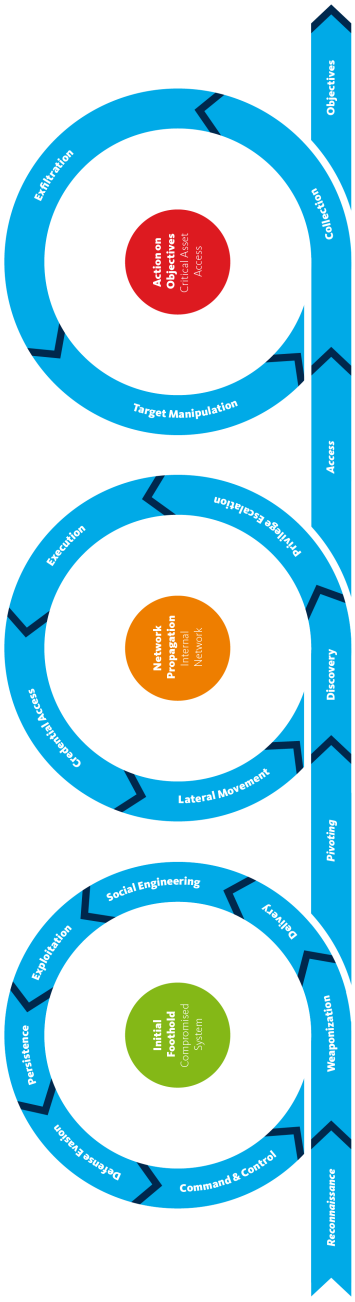

FIGURE 1.4 – *Unified Kill Chain* de Pols

Nous sommes convaincus que ces modèles à boucles sont plus adaptés pour décrire la notion essentielle des campagnes d'attaques sophistiquées, car ils considèrent la durée et donc la notion de persistance. Pourtant, nous considérons que ces modèles souffrent de deux principaux défauts. Premièrement, ils ne considèrent pas la possible régression d'un attaquant au cours de sa progression. Cette régression peut faire suite à des actions involontaires de la victime ou encore à des contremesures appliquées par le défenseur afin de stopper l'attaque. Deuxièmement, ces modèles ne laissent pas transparaître les périodes d'attente, alors qu'elles sont souvent précisées dans les rapports d'analyse de la menace. Ces périodes et leurs contextes sont pourtant primordiaux afin obtenir une bonne compréhension des processus et des capacités opérationnelles des attaquants. La Figure 1.5 issue du cas d'espionnage rapporté [18] par le *GovCERT* suisse en est un exemple concret. Ce diagramme représente, jour par jour, le volume de données exfiltrées du SI de l'entreprise suisse RUAG. La société, active dans les domaines de l'aérospatial et de la défense, a été la victime d'une campagne APT, découverte en 2016. La figure laisse clairement apparaître des périodes où l'activité de l'attaquant est inexistante, mais également des pics qui témoignent d'exfiltrations massives de données.

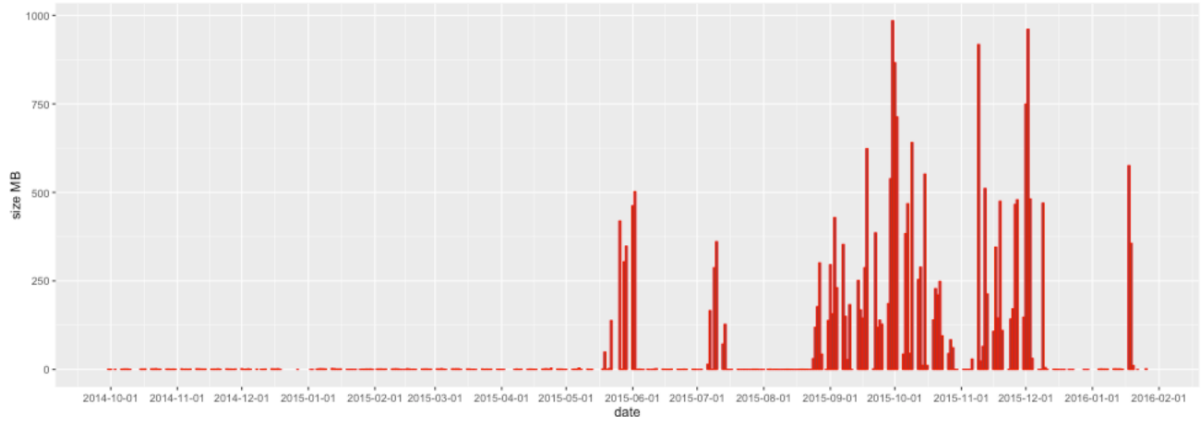

FIGURE 1.5 – Diagramme présentant la volumétrie des données exfiltrées du SI du RUAG.

C'est ce constat qui nous a poussé à modéliser un cycle de vie sous la forme d'un automate fini. L'approche que nous avons proposée permet de visualiser les différentes phases opérationnelles par lesquelles passe l'attaquant au cours de sa campagne. Cette première contribution est présentée dans le Chapitre 2.

1.4 Tactiques, Techniques et Procédures (TTP)

Le concept de Tactiques, Techniques et Procédures (TTP) a également été emprunté à la doctrine militaire. En 2017, Maymi *et al.* [19] ont initié des travaux de définition des TTP pour le cyberespace. Mettant notamment en avant la notion séquence quelque soit le niveau d'abstraction (Tactiques, Techniques ou Procédures). La Figure 1.6, issue de leur article, présente une séquence de procédures pour exécuter une technique dite "*User self-exploits*" mise en œuvre par APT28.

Dans cet article, ils présentent également un extrait de spécification pour une procédure. Cette publication a été complétée par nos travaux présentés en Section 4.4 où nous détaillons une sémantique permettant de décrire formellement des techniques et qui est une contribution de cette thèse.

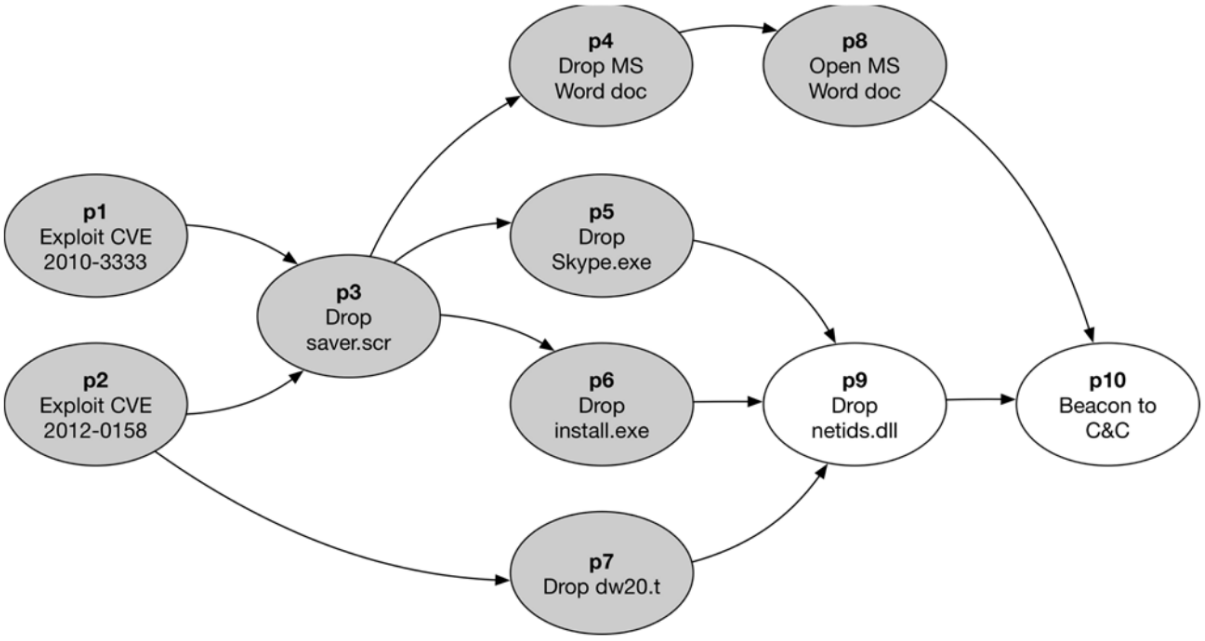

FIGURE 1.6 – Séquence de procédures permettant l'exécution d'une technique par APT28, présentée dans l'article de Maymi *et al.*

Par la suite, la société américaine *MITRE Corporation* a réussi à populariser le concept des TTP. Les acteurs de la cybersécurité ont immédiatement adopté ce concept puisqu'il était accompagné d'une base de ressources documentaires [20] qui fait référence : les matrices ATT&CK™ [21]. La Figure 1.7 présente un extrait de la matrice *Enterprise*.

Ces matrices se lisent de la façon suivante :

— les colonnes sont les Tactiques

— dans chaque colonne, les lignes sont les Techniques associées à la Tactique.

— certaines Techniques, marquées par un onglet gris, ont été regroupées et sont désignées par la notion de Sous-Techniques.

— chaque tactique et technique est précédée d'un numéro de nomenclature qui permet de l'identifier de façon unique.

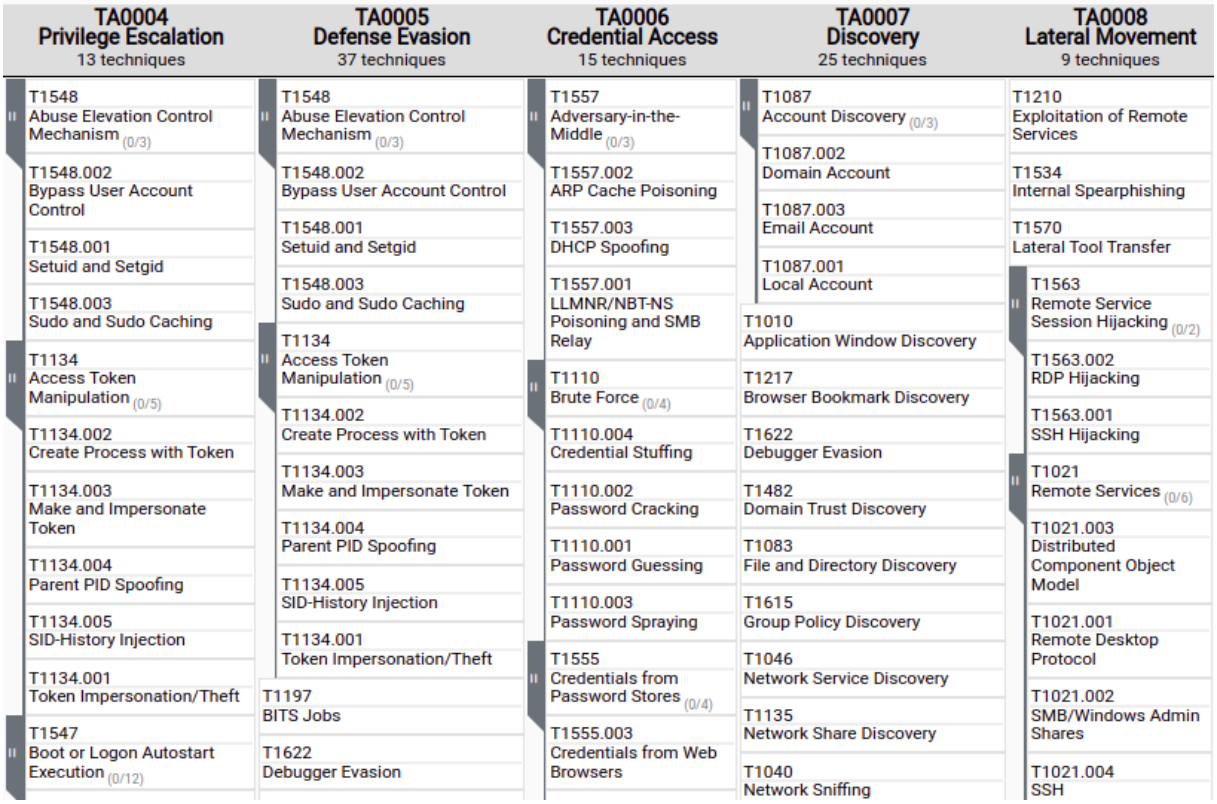

FIGURE 1.7 – Extrait de la matrice ATT&CK en date du 7 juillet 2022. La matrice complète est disponible sur https://attack.mitre.org/matrices/enterprise/

1.4.1 Tactiques (intentions)

MITRE définit les *Tactiques* comme étant le "pourquoi" d'une *Technique*. Il précise qu'il s'agit de l'objectif tactique de l'attaquant. Il s'agit de la raison pour laquelle il effectue une action.

La tactique fait référence à l'intention de l'attaquant quand il mène son action. Par exemple, les principales tactiques décrites par le MITRE sont l'acquisition d'un accès initial, la persistance, l'élévation de privilèges.

1.4.2 Techniques (moyens)

MITRE définit les *Techniques* comme étant le "comment" pour l'attaquant d'atteindre un objectif tactique en effectuant une action. A noter que dans la matrice ATT&CK, la notion de "sous-technique" a été introduite pour décrire plus spécifiquement le comportement de l'attaquant lorsqu'il met en œuvre certaines techniques.

La technique fait référence au moyen mis en œuvre par l'attaquant pour satisfaire une intention (*i.e.*, une tactique). Par exemple, les principales techniques décrites par le MITRE pour la tactique de Persistance sont la création d'un compte, la création d'un processus système, les tâches planifiées.

Dans le Chapitre 4, nous formalisons treize techniques grâce à une sémantique qui permet de décrire l'évolution de la connaissance de l'attaquant.

1.4.3 Procédures (manières)

MITRE définit les *Procédures* comme étant les implémentations spécifiques des *Techniques* utilisées par les attaquants.

La procédure fait référence à la manière dont est exécutée une technique. Le MITRE présente au sein de sa base de connaissance quelques exemples qu'ils consignent sous le nom de "procédures" cependant il ne s'agit que de phrases permettant de lier des groupes d'attaquants à des techniques. Par exemple, sur la fiche de la technique de création ou de modification d'un processus système[1], on retrouve la pseudo-procédure simpliste suivante pour le groupe d'attaquant APT3 : *APT3 has a tool that creates a new service for persistence.*

1. https://attack.mitre.org/techniques/T1543/003/

La notion de procédure est bien plus profonde, car elle est écrite par la main de l'attaquant. C'est sa manière d'agir, sa signature. Contrairement à ce qui est régulièrement pratiqué [22] en cherchant à caractériser les attaquants selon les techniques qu'ils mettent en œuvre, c'est l'analyse de la procédure qui permettra aux experts en CTI d'identifier des motifs similaires entre les différents artefacts découverts sur différentes victimes.

Les rapports décortiquant les *malware* issus de campagnes d'attaques sophistiquées retranscrivent d'ailleurs une partie des procédures d'attaquants détectées, retrouvées ou observées sur les SI des victimes. Les *malware* sont produits par les attaquants afin d'automatiser l'exécution des procédures redondantes ou qui nécessitent une exécution sans interaction avec la machine où elles sont déployées. D'ailleurs, nous retrouvons parfois dans la littérature [23] une variante du concept de TTP : les TTTP (*Tools, Tactiques, Techniques, and Procedures*). Cette version est également intéressante puisqu'elle intègre l'idée que l'outil (*i.e.*, le *malware*) est une partie intégrante du profil tactique de l'attaquant. A ce titre, le MITRE intègre les outils dans sa base de connaissance et les relie aux tactiques, techniques et groupes d'attaquants ayant été observés.

Pour l'attaquant, l'outil est une forme automatisée d'une partie de ses procédures qu'il va déployer sur le SI. Pour le défenseur, le *malware* est une forme fossilisée de la procédure de l'attaquant qui est découverte sur le SI.

1.5 Connaissance situationnelle

Le concept de connaissance situationnelle (*Situational Awareness*) est un domaine des sciences cognitives qui étend la réflexion philosophique sur la théorie de la connaissance et sur la manière dont cette connaissance est acquise par le sujet. Ce concept est accompagné dans sa formalisation, proposée par Endsley en 1995 [24], d'un processus d'acquisition et de prise de conscience de l'environnement qui entoure le sujet. Il s'agit du *Situational Assessment*. La Figure 1.8 présente les facteurs impliqués dans ce processus.

La littérature [25] fait également référence à de nombreuses publications dans le domaine des sciences informatiques, sous le nom de *Cyber Situational Awareness*. Dans un contexte d'étude des menaces informatiques sophistiquées, cela équivaut à la mise en œuvre de techniques de *Discovery*. Nous spécifions formellement plusieurs de ces techniques dans le Chapitre 4. Nous présentons la posture de l'attaquant fouillant un SI dans le Chapitre 2, au travers du cycle de vie opérationnel suivi par l'attaquant.

Par ailleurs, dans cette thèse nous faisons régulièrement référence aux notions de vision, de perception, de perspective et de point de vue, qui font elles-mêmes référence à la notion plus globale de **connaissance** et aux processus associés. En effet, nous considérons que l'attaquant comme le défenseur manipulent les mêmes concepts et développent chacun leurs représentations de la campagne d'attaque. Ces représentations, conditionnées par leurs connaissances et leurs expériences respectives, suivent donc une tendance convergente vers la réalité du terrain (campagne d'attaque et architecture du SI).

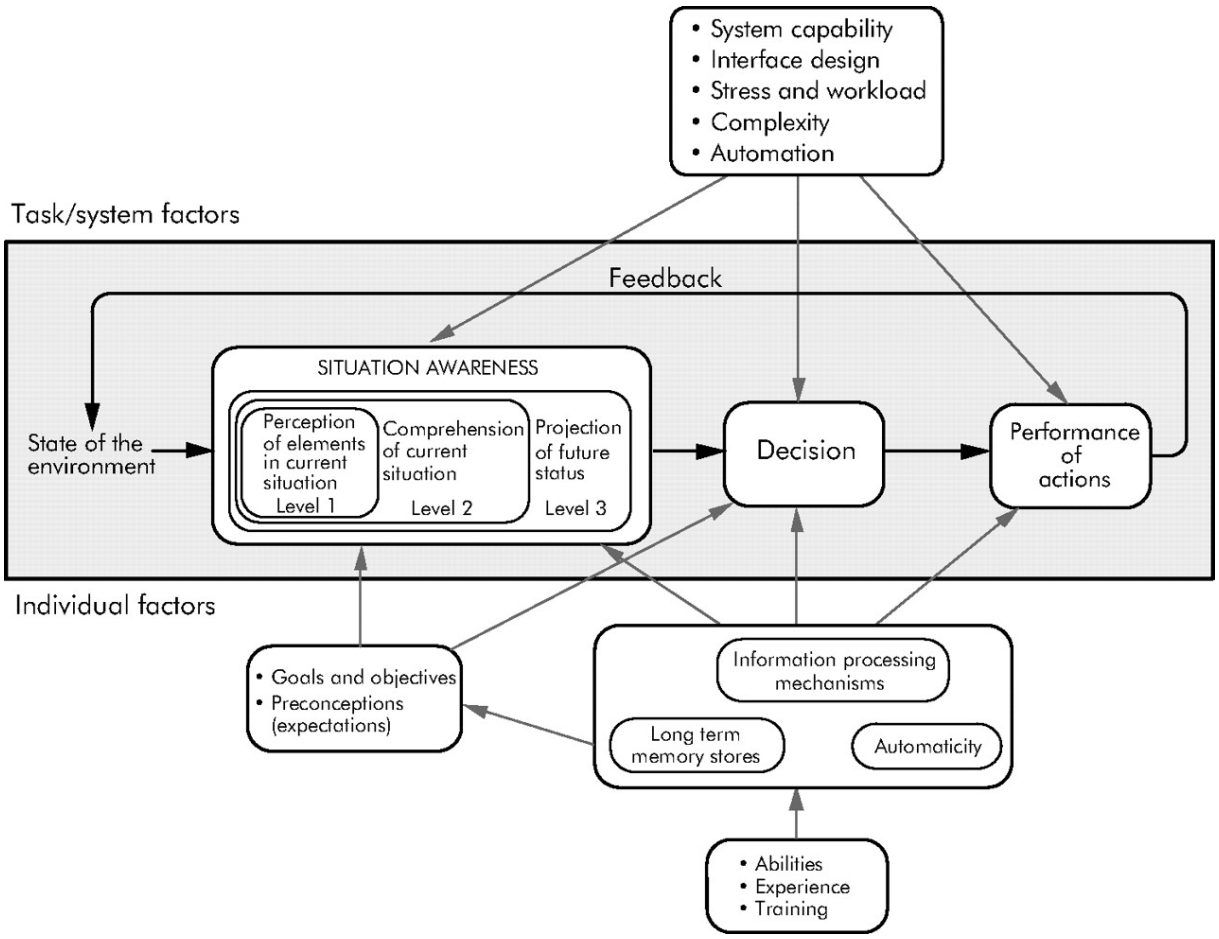

FIGURE 1.8 – Modèle de connaissance situationnelle (*Situational Awareness*) dans un contexte de prise de décision dynamique, proposé par Endsley en 1995.

1.6 Processus de réponse à incident (PICERL)

Les équipes de réponse à incident qui interviennent sur un SI compromis, suivent généralement le processus PICERL [26]. Ce processus est composé de six phases que nous détaillons dans cette section.

1.6.1 *Preparation*

Il s'agit simplement de la **phase de préparation** des ressources qui seront impliquées dans l'opération de réponse à incident. C'est également au cours de cette phase que les défenseurs vont collecter de la documentation sur le SI.

1.6.2 *Identification*

La **phase d'identification** est particulièrement critique puisqu'elle conditionne la pertinence des phases ultérieures. En effet, si le défenseur n'a pas détecté de façon exhaustive l'espace dans lequel l'attaquant s'est propagé, son confinement et son éradication ne seraient probablement pas efficaces, puisque cette action n'impacterait qu'une partie du déploiement opérationnel de l'attaquant dans le SI. Nous proposons dans le Chapitre 3 un cadre permettant, pour le défenseur, l'amélioration de la qualité des traces collectées dans cette phase.

1.6.3 *Containment*

Une fois que le défenseur possède une vision, qu'il considère comme suffisante, sur l'espace de propagation de l'attaquant, il va chercher à contenir la menace dans un périmètre qu'il contrôle. À ce moment, il est possible que le défenseur tolère la présence de l'attaquant dans le SI afin de mieux préparer la phase suivante et ainsi de pouvoir l'exécuter dans une fenêtre temporelle très courte. Pendant cette **phase de confinement**, il est primordial pour le défenseur qu'il ne dévoile pas son intention à l'attaquant au risque de le faire réagir, se réorganiser et élargir son espace de propagation, pis encore conduire des opérations destructrices sur le SI.

1.6.4 *Eradication*

La **phase d'éradication** de la menace est une opération "coup de poing" pour le défenseur où en quelques heures, il doit anéantir tous les accès de l'attaquant du SI compromis afin qu'il ne puisse plus reprendre pied sur le réseau. Certaines actions de la phase de rétablissement de l'activité doivent être faites dès la phase d'éradication. Par exemple, l'application de correctifs de sécurité sur les *assets* compromis doit être fait dans un même temps. Cela a pour but d'empêcher l'attaquant de pénétrer une nouvelle fois le SI avec le même mode opératoire.

1.6.5 *Recovery*

La **phase de rétablissement de l'activité** consiste à rétablir les éventuels *assets* qui auraient été compromis ou détruits par l'attaquant et ceux qui auraient été volontairement désactivés par le défenseur dans le cadre de l'opération de réponse à incident.

1.6.6 *Lessons Learned*

La **phase de retour d'expérience** vise à identifier à chaud, puis à froid, les erreurs commises dans le SI qui auraient rendu possible cette attaque. Il s'agit également de capitaliser sur les succès obtenus et erreurs commises durant l'opération du défenseur afin d'améliorer son déroulement sur un prochain incident.

1.7 *Red versus Blue*

La prise de conscience collective de l'existence de menaces sophistiquées a contraint les entreprises à reconsidérer leur approche de cybersécurité. Ainsi, ces entités défient régulièrement leurs niveaux de résilience face à ces menaces, grâce à des exercices réalistes parfois appelés *Red versus Blue*, qui simulent des campagnes d'attaques réelles et impliquent les équipes de défense pour les contrer [27]. Dans ces exercices, la *Blue Team* chasse la *Red Team* dans le cadre d'une opération de *Threat Hunting* [28], pendant que la *Red Team* essaye de conduire furtivement son attaque. Ces exercices permettent de tester le niveau de sécurité global de l'entreprise et de vérifier la pertinence des configurations des sondes, du *Security Information Event Management* (SIEM) ou encore des procédures de réponse à incident. Il est à préciser que les exercices d'attaque et de défense d'une infrastructure

critique sont également inspirés du monde militaire [29], où des attaques cinétiques fictives conduites par des unités offensives impliquent des unités défensives pour les entraîner et tester leurs degrés de préparation pour le combat. On trouve également dans la communauté scientifique dès le début des années 90, des expérimentations qui semblent être les prémices des exercices de *Red Team* à des fins de cybersécurité [30].

En cybersécurité, ces exercices aident les organisations à maintenir un niveau de sécurité suffisant pour protéger les *assets*. La *Red Team* est composée d'experts de haut niveau pouvant endosser le rôle de potentiels attaquants motivés par l'atteinte d'un objectif stratégique. Par exemple, dérober des données sensibles, utiliser les capacités technologiques de l'entreprise à des fins malveillantes ou encore mettre en échec des services de l'entreprise. La *Blue Team* défend l'entreprise et doit s'assurer que les *assets* ne sont pas compromis, dans le cas où la *Red Team* aurait découvert une vulnérabilité et l'aurait exploitée sur l'un d'entre eux. Ainsi, la *Blue Team* doit rapidement contraindre la progression de la *Red Team* afin de contenir la menace.

1.8 Jeux de données existants

Afin d'étudier les menaces sophistiquées, il est intéressant de pouvoir disposer des journaux d'événements de compagnies fraîchement compromises par des acteurs malveillants. Cependant, cela paraît inconcevable à cause des risques encourus par les victimes à exposer publiquement ces informations et ces données techniques. De plus, cela nuirait au travail des experts en CTI qui gagnent progressivement des connaissances sur les attaquants ayant été détectés dans les SI des victimes. C'est pourquoi la communauté scientifique a dû explorer d'autres alternatives pour pallier le manque de jeux de données publics.

1.8.1 Dépôts de rapports publics

Régulièrement, les experts du domaine publient des rapports relatifs à des campagnes d'attaques sophistiquées ou à leurs outillages sur leurs blogs, sur les sites web des compagnies de cybersécurité, ou encore sur les sites institutionnels des Computer Emergency Response Team (CERT). Par exemple, le dépôt APTnotes [31] présente une collection de 634 rapports publics décrivant des modes opératoires observés ou des *malware* décortiqués. Généralement, ces rapports sont attribués spécifiquement à des groupes d'attaquants. On y retrouve des détails sur les différents aspects des campagnes, tels que les objectifs visés,

les vecteurs d'accès initiaux puis de propagation, les principales TTP, les effets observés sur les systèmes ou encore des *Indicators of Compromise* (IoC) et des règles de détection pouvant être immédiatement ajoutés aux systèmes d'alerte des SIEM.

À titre de d'exemple, certains rapports tels que *UNC3524 : Eye Spy on Your Email* [32] de Mandiant, documentent de façon sommaire et informelle les campagnes d'attaques. Dans ce type de rapport, nous déplorons l'absence de formalisme et la mise en évidence d'une infime partie de la campagne d'attaque.

D'autres équipes, comme celle des analystes du groupe *The DFIR Report* produisent des rapports extrêmement détaillés et précis sur les campagnes d'attaques observées et essayent d'utiliser les *Kill Chains* les plus adaptées à l'analyse de la campagne. Par exemple, dans *Stolen Images Campaign Ends in Conti Ransomware* [33], ils présentent une frise chronologique qui retrace toutes les commandes exécutées par l'attaquant. Nous regrettons la focalisation sur les détails de procédures au détriment d'une analyse au niveau tactique, qui permettrait une lecture opérationnelle de la campagne. Néanmoins, au regard des éléments présents dans le rapport, le modèle présenté au Chapitre 2 pourrait aisément être instancié par cette campagne d'attaque.

Les experts CTI ont besoin de ces rapports pour disposer d'une vue d'ensemble sur la menace. Cependant, il est difficile d'en extraire des informations techniques qui produiraient un renseignement sur la menace directement actionnable [34] par des équipes de détection ou de chasse. Dans la Section 4 nous présentons un formalisme qui permet de décrire précisément une campagne d'attaque observée sur un SI.

1.8.2 Collecte de journaux d'événements sur des systèmes réels

Les journaux d'événements système et réseau observés sur des SI réels compromis sont indiscutablement d'excellentes sources d'information pour étudier les menaces sophistiquées. En 2017, Turcote *et al.* [35] [36] ont publié leur troisième jeu de données qui était composé de trafic réseau et de journaux d'événements collectés sur les machines de l'entreprise *Los Alamos National Laboratory* sur une période aléatoire de 90 jours consécutifs. Afin de protéger la société, ils ont pris soin de désidentifier certaines valeurs sensibles avant de publier le jeu de données. En 2021, Ho *et al.* [37] ont présenté un système pour détecter les mouvements latéraux dans les journaux d'événements d'un SI. Pour l'illustrer, ils ont utilisé un jeu de données, resté privé, de 780 millions de lignes collectées sur les serveurs internes de l'entreprise Dropbox. Pendant qu'ils étaient en train de construire leur jeu de données, une *Red Team* a été engagée afin de simuler un scénario de type APT.

La société a également mis en place plusieurs produits de sécurité commerciaux ainsi que des outils internes afin de tester les limites de leurs systèmes.

Si ce genre de jeu de données est particulièrement intéressant pour étudier les procédures d'attaques exposées par les adversaires, il apparaît que les données demeurent néanmoins incomplètes voire altérées pour des raisons de confidentialité. De plus, il est plus difficile de comprendre le contexte exact de capture de ces données si l'architecture du SI n'est pas décrite précisément. Ainsi, la reconstruction détaillée de la campagne d'attaque par les experts est compliquée.

1.8.3 Simulation contradictoire ou imitation de TTP

Gianvecchio *et al.* [38] ont proposé BRAWL : un environnement de simulation d'attaque totalement automatisé. Cette expérimentation impliquait un défenseur, appelé *Blue Bot*, et un attaquant, appelé *Red Bot*. L'attaquant conduisait des actions offensives générées automatiquement grâce à l'outil CALDERA [39] et il ciblait directement les machines présentes dans l'environnement de jeu.

En parallèle et de façon continue, le défenseur réalisait une extraction des journaux d'événements présents sur les machines attaquées. Puis, le défenseur appliquait sur ces traces des règles de détection. Un jeu de données a pu être créé grâce à cet ensemble de traces. Il n'a cependant pas été rendu public.

Une autre approche suivie a été de générer un jeu de données à partir d'actions offensives atomiques, telles que précédemment observées par des experts CTI. C'est l'objet du projet "Security Datasets" [40] mené par Rodriguez. Il s'est fixé pour objectif d'enregistrer les traces caractéristiques de l'exécution de techniques populaires, selon des procédures spécifiques, ou de la mise en œuvre d'outils offensifs populaires, tels que ceux utilisés par certains attaquants. Ces traces peuvent être exploitées individuellement afin de créer des règles de détection dans des SIEM. Pourtant, puisqu'elles ont été produites suite à l'exécution de fragments de procédures, elles ne sont pas suffisantes pour constituer une source de données adaptée à l'étude de ces acteurs malveillants.

Rodriguez est également à l'origine de la publication du jeu de données Mordor [41], qui imite les TTP du groupe APT28. Nous l'avons exploité afin de valider notre modèle théorique proposé dans le Chapitre 3. Le jeu de données est aussi présenté dans ce chapitre.

1.8.4 Observation de *Red Teams* sur des plateformes dédiées

Le projet "Operationally Transparent Cyber (OpTC)" [42], porté par la Defense Advanced Research Projects Agency (DARPA), a permis de partager avec la communauté scientifique un important jeu de données. Il est composé de 17,4 milliards d'événements [43], collectés sur des sondes système et réseau déployées sur une infrastructure dédiée de 500 à 1000 machines. Pendant trois jours, des machines choisies aléatoirement ont été attaquées par une *Red Team*.

Myneni *et al.* [44] ont proposé le jeu de données DAPT pour étudier les APT dans l'optique de développer une solution basée sur du *machine learning* qui permettrait de mieux détecter ce type d'attaques. Pendant la période de capture des événements une *Red Team* a également été engagée. Elle a conduit des actions offensives contre le SI. Il s'agissait plus particulièrement de satisfaire des intentions techniques de reconnaissance, de primo-accès, de latéralisation et d'exfiltration de données. Malheureusement, l'infrastructure utilisée lors de cette expérimentation n'a pas été clairement décrite par les auteurs. De plus, nous regrettons qu'une seule *Red Team* interne ait été engagée. Cela peut introduire un biais dans le jeu de données à cause du manque de diversité des TTP mises en œuvre et ainsi ne pas exprimer réellement toute la richesse des possibilités d'attaques offertes.

Les jeux de données qui permettent l'étude des menaces sophistiquées n'ont de valeur pour la communauté scientifique que si leur description est suffisamment précise pour que l'expérimentation puisse être reproduite. Sur cet aspect, les jeux de données actuellement disponibles demeurent insuffisamment documentés, notamment en terme de description d'infrastructure ou de procédures d'attaques exécutées, mais également en terme de diversité (*i.e.*, un seul attaquant par expérimentation).

C'est pourquoi, nous proposons dans la Section 4 une expérimentation qui implique 22 attaquants, chacun sur une infrastructure dédiée, ainsi qu'un formalisme qui permet de décrire précisément l'infrastructure du SI et les techniques employées par les attaquants.

1.9 Contributions

Les travaux de cette thèse, motivés par la compréhension des menaces dites "sophistiquées" au travers d'un prisme tactique, nous ont permis de proposer quatre contributions majeures qui se complètent mutuellement.

Dans un premier temps, à un niveau macroscopique, nous avons proposé une modélisation du cycle de vie suivi par un attaquant au cours d'une campagne. Ces travaux

améliorent les différentes *Kill Chains* présentées dans la Section 1.3 en présentant une vision plus opérationnelle de l'attaquant, qui progresse dans le SI en quête de ses objectifs finaux. Ces travaux ont fait l'objet d'une publication, en 2019, dans le cadre de la conférence scientifique *Conference on Computational Science & Computational Intelligence* avec l'article "*Modeling the Operational Phases of APT Campaigns*". Cette contribution est développée dans le Chapitre 2.

Alors que se dessinait le cycle de vie de l'attaquant et à mesure que nous l'instancions sur des attaques ayant été documentées publiquement, il nous est apparu que l'une des phases de l'attaque (Propagation réseau) était favorable au défenseur pour détecter plus efficacement l'attaquant. Nous avons donc poursuivi notre réflexion en nous focalisant sur cette phase et en nous concentrant sur l'élément que nous considérons comme le point de jonction entre l'attaquant (au cours de sa campagne) et le défenseur (au cours de sa chasse) : la trace. Nous avons donc pu formaliser la notion de trace et son cheminement depuis l'instanciation d'une procédure des TTP de l'attaquant jusqu'à l'IoC convoité par le défenseur pour chasser l'intrus dans le SI. Cette formalisation nous a également permis, grâce à une expérimentation sur un des rares jeux de données disponibles pour étudier les menaces sophistiquées, d'identifier les leviers sur lesquels pouvait agir le défenseur afin d'améliorer ses capacités de détection, notamment en réduisant le nombre de faux positifs.

Ces travaux ont fait l'objet d'une publication, en 2021, dans la revue scientifique *IEEE Transactions on Network and Service Management* avec l'article titré "*From TTP to IoC: Advanced Persistent Graphs for Threat Hunting*". Cette contribution est développée dans le Chapitre 3.

Lors de ces travaux, nous avons été confrontés à deux difficultés. La première est le manque de formalisme dans la capitalisation des TTP, portée notamment par le MITRE dans leur matrice ATT&CK. Ce manque se fait aussi ressentir dans les rapports de CTI où des structures informelles sont utilisées pour retracer les campagnes d'attaques observées.

La seconde difficulté rencontrée est le manque de diversité dans les jeux de données publiés et qui pourraient être utilisés afin d'étudier les menaces sophistiquées, plusieurs pistes ont cependant été suivies comme nous l'avons précisé dans la Section 1.8.

Nous avons donc proposé un modèle théorique, qui permet de décrire l'infrastructure de la victime, l'évolution de la connaissance de l'attaquant et son espace de propagation au cours de la campagne. Ce modèle est accompagné d'une sémantique, qui permet de spécifier formellement les techniques d'attaques mises en œuvre et les effets de leurs exécutions sur la connaissance de l'attaquant à propos du SI. De plus, grâce à une expérimentation

de grande envergure à laquelle ont participé 22 professionnels de la discipline, nous avons pu constituer et publier un jeu de données inédit. Nous avons également pu utiliser notre modèle théorique pour retracer les campagnes d'attaques des participants à partir des traces collectées par nos systèmes de supervision. Ces travaux ont fait l'objet d'une publication, en 2022, dans la revue scientifique *IEEE Transactions on Network and Service Management* avec l'article "*PWNJUTSU: A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns*". Ces contributions sont développées dans le Chapitre 4.

Chapitre 2

Modélisation des phases opérationnelles d'une campagne d'attaque

2.1 Introduction

Quand il s'agit de décrire le déroulement d'une campagne d'attaque, les experts s'appuient généralement sur les *Kill Chains*, telles qu'elles sont présentées dans la Section 1.3. Cependant, grâce à l'émergence et au développement des prestations d'audits en mode *Red Team*, nous sommes régulièrement, en tant qu'experts "métier", perplexes quant à la pertinence de ces modèles. En effet, ils sont considérés comme inadaptés pour relater des actions conduites dans ces opérations offensives qui s'étendent sur plusieurs mois.

Ce chapitre introduit un modèle, appelé Nuke, qui propose une lecture plus opérationnelle du cycle de vie des attaquants évoluant dans un SI compromis. Ce travail a été présenté dans l'article "*Modeling the Operational Phases of APT Campaigns*" [45].

Le modèle Nuke se démarque de ceux existant à ce jour, car il permet de considérer la notion de régression de l'attaquant vers un état antérieur, mais surtout la notion de répétitivité de l'atteinte des objectifs finaux par l'attaquant, qui est pourtant explicitement précisée dans la définition du NIST en Section 1.2 : "*[...]pursues its objectives repeatedly over an extended period of time*". Ce modèle introduit également, à un niveau macroscopique, deux concepts intrinsèques de l'attaquant. Il s'agit de l'espace de propagation et de la connaissance acquise à propos de l'environnement de la victime.

Ces dernières années, quelques incidents majeurs ont été documentés et diffusés publiquement, cela nous a fourni l'opportunité de confronter notre modèle avec deux exemples de campagnes d'attaque. Ainsi, nous avons pu mettre en évidence l'importance de reconstruire la chronologie des attaques dans les rapports d'analyse de la menace (CTI). De plus, nous avons souligné l'importance de porter l'effort sur la compréhension et la capitalisation des TTP mises en œuvre par les attaquants au cours de leurs campagnes.

Dans ce chapitre, nous présentons en Section 2.2 notre modèle qui représente le cycle de vie de l'attaquant dans un SI compromis et qui considère une éventuelle régression. Ce modèle introduit également le concept d'état d'attente, qui est incontournable dans les actions s'étalant sur une longue période. Puis, nous présentons en Section 2.3 une confrontation de ce modèle avec deux campagnes d'attaques récentes pour lesquelles les progressions des attaquants ont été décrites publiquement : la fuite de données subie par Equifax (2017) et le sabotage de TV5Monde (2015).

2.2 Le modèle Nuke

Comme cela a été présenté dans la Section 1.3, au cours de la dernière décennie, nous avons observé que plusieurs modèles avaient été proposés dans le cadre de différentes publications scientifiques. Cependant, nous regrettons qu'aucun d'entre eux ne considère les notions de répétitivité des actions sur l'objectif final. De plus, les précédents modèles ne tiennent pas compte du fait que l'attaquant puisse rencontrer des déconvenues au cours de sa campagne d'attaque ou qu'il puisse simplement être perturbé par des mesures préventives, voire des contremesures réactives, émanant du défenseur et qui auraient pour conséquence de le faire régresser opérationnellement. Il devrait donc régulièrement réexa-

miner son intérêt à poursuivre l'attaque et en assumer les coûts, dont Derbyshire *et al* ont établi [46] que les facteurs sont le temps, les finances et le risque.

Dans cette section, nous proposons un nouveau modèle du cycle de vie d'un attaquant évoluant dans un SI compromis. **Nous considérons ici que l'attaquant a déjà pris pied dans le SI, quelle que soit la technique de primo accès employée.** À titre d'exemple, ces techniques peuvent être l'exploitation d'une vulnérabilité sur un service exposé publiquement, une campagne d'hameçonnage ciblé (*Spearphishing*), une attaque par point d'eau (*Watering Hole*), une intrusion dans le réseau local sans-fil ou encore la compromission d'un employé de l'entreprise (*Insider Threat*) [47].

Notre modèle, *Nuke*, est formalisé par un automate fini, composé de six états. Ces six états peuvent être regroupés en trois super-états, qui correspondent aux trois phases opérationnelles d'une campagne d'attaque.

— la phase d'*Exploration* ;
— la phase d'*Exploitation* ;
— la phase de *Prise de décision*.

Dès lors qu'il a compromis une première machine dans le SI, la phase d'*Exploration* commence. Au cours de cette phase, l'attaquant découvre le SI au travers du réseau jusqu'à ce qu'il réussisse à atteindre la machine qui héberge son objectif final. Une fois que l'attaquant a réussi à dompter cette nouvelle machine, commence la phase d'*Exploitation*. Dans cette phase, il va régulièrement vérifier qu'il est toujours en mesure de conduire ses *techniques ultimes* et qu'elles produisent des effets concrets sur son objectif final. Ces deux premiers super-états correspondent à des phases tactiques, qui peuvent se retrouver dans d'autres modèles, présentés en Section 1.3. À ceux-là, nous avons ajouté un troisième super-état : la phase de *Prise de décision*, au cours de laquelle l'attaquant va évaluer la possibilité d'arrêter l'attaque. Dans le modèle Nuke, l'attaquant est caractérisé au travers de ses capacités opérationnelles, qui sont des techniques, au sens "TTP" du terme, correspondant en partie à celles listées dans la matrice MITRE ATT&CK, présentée dans la Section 1.4. Dans la suite de cette section, nous présentons les éléments fondamentaux impliqués dans notre modèle. Puis, nous présentons en détail le modèle Nuke.

2.2.1 Éléments de Nuke

Objectif final de l'attaquant et *assets* du SI

L'objectif final de l'attaquant peut-être, par exemple, d'exfiltrer des données ou encore d'agir sur le comportement d'un *asset*[1] du SI. Ces *assets* sont les éléments qui composent le SI. Ils peuvent être par exemple des stations de travail, des serveurs ou encore des équipements réseau. Dans un souci de simplicité, nous les appellerons des "**machines**". Dans la suite de cette section, nous allons utiliser le symbole $\mathbf{m}$ pour désigner une machine du SI. Ainsi, nous notons $\mathbf{m}_{init}$ le point d'implantation initial de l'attaquant, qui correspond à la première machine compromise dans le SI. Enfin, nous notons $\mathbf{m}_{final}$ la machine qui héberge l'objectif final de l'attaquant. Il n'est bien sûr pas exclu que l'objectif final se trouve sur le point initial d'implantation, auquel cas $\mathbf{m}_{init} = \mathbf{m}_{final}$.

Espace de propagation de l'attaquant et connaissance du SI

Au cours de sa campagne, l'attaquant gagne du terrain et progresse vers son objectif. À chaque obtention d'un accès à une nouvelle machine $\mathbf{m}$, il s'approprie le compte d'un utilisateur $\mathbf{u}$. Le couple $(\mathbf{m}, \mathbf{u})$ est donc un point de présence de l'attaquant dans le SI. On écrit $\mu = \{(\mathbf{m}_1, \mathbf{u}_1), (\mathbf{m}_2, \mathbf{u}_2), \dots\}$ l'ensemble de ses points de présence dans le SI. μ **est l'*espace de propagation* de l'attaquant**. La connaissance acquise par l'attaquant à propos du SI influence ses choix et donc son comportement tactique au cours de l'attaque. Ces concepts introduits d'espace de propagation et de connaissance du SI sont détaillés dans le Chapitre 4.

Capacités techniques de l'attaquant et leurs effets sur le SI

Les capacités techniques de l'attaquant désignent les *Techniques* qu'il maîtrise et qu'il est en mesure d'utiliser sur une machine. La communauté scientifique reconnaît désormais la matrice MITRE ATT&CK comme base de connaissances de référence pour la capitalisation des *Tactiques* et *Techniques*. Nous utiliserons donc les identifiants ATT&CK pour désigner les *Techniques*, notées $\mathbf{t}_{xxxx.yyy}$, où $xxxx$ correspond à l'identifiant de la technique et yyy correspond à l'identifiant de la sous-technique. Par exemple, $\mathbf{t}_{1550.002}$ désigne la *Technique Pass the Hash (PtH)*. Ainsi, nous écrivons $\mathbf{TTP}$ l'ensemble des techniques $\mathbf{t}_j$ existantes et $\mathbf{TTP}_A$ l'ensemble des techniques maîtrisées par l'attaquant,

1. Ce terme est parfois traduit en français par "biens essentiels".

tel que $\mathbf{TTP}_A \subseteq \mathbf{TTP}$. $\mathbf{TTP}_A$ correspond donc aux capacités techniques de l'attaquant.

La mise en œuvre d'une *Technique* est appelée ici une *action*. Cette *action* induit :

— un ensemble de traces involontairement disséminées par l'attaquant sur le SI ;

— un ensemble de machines nouvellement découvertes $\mathbf{m}_i$ et d'utilisateurs de ces machines qui viennent enrichir la connaissance de l'environnement de la victime.

2.2.2 États de Nuke

Dans cette section, nous décrivons précisément les différents états de l'automate de Nuke et les conditions de franchissement des transitions vers un autre état.

La Figure 2.1 est une représentation graphique du modèle Nuke et sa table de transition est présentée sur la Figure 2.2.

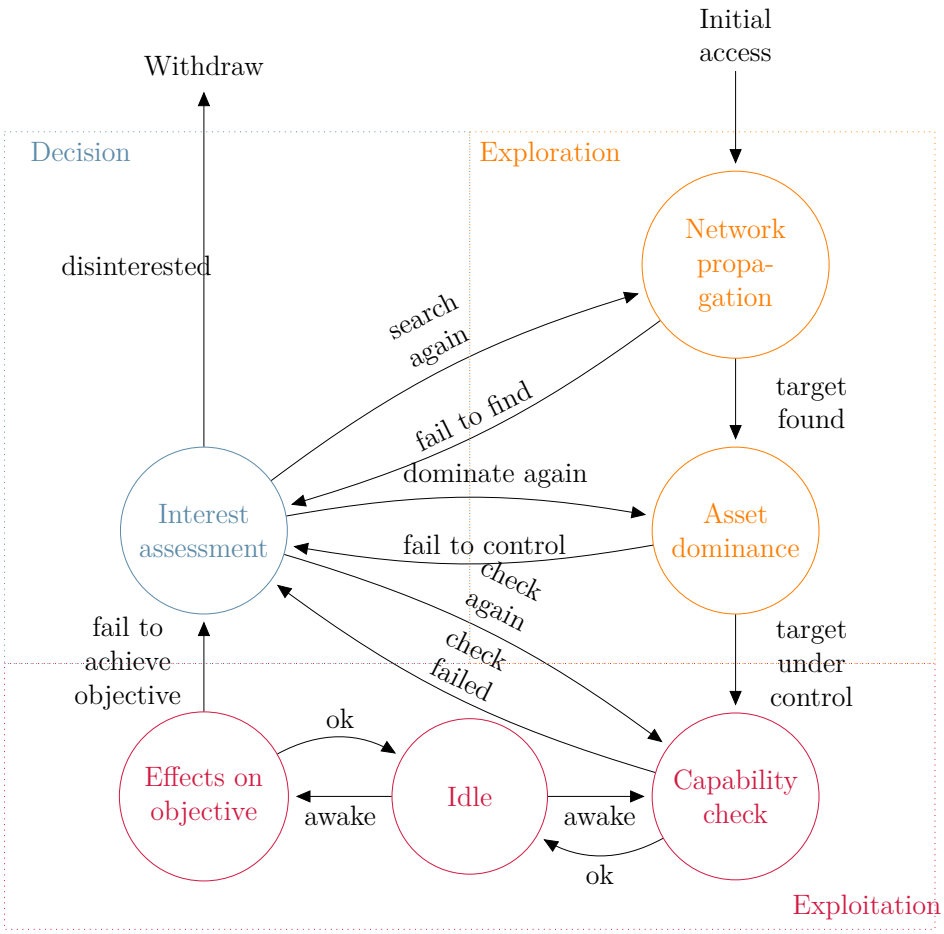

FIGURE 2.1 – Représentation du modèle Nuke

Event / State	$\mathbf{m}_{final}$ found	$\mathbf{m}_{final}$ owned	Failure	Awakening	Positive Review
Network Propagation	Asset dominance		Interest assessment		
Asset dominance		Capability check	Interest assessment		
Capability check			Interest assessment		Idle
Idle				Capability check Effect on objective	
Effect on objective			Interest assessment		Idle
Interest assessment					Network Propagation Asset dominance Capability check

FIGURE 2.2 – Table de transitions de Nuke

Phase d'Exploration

Propagation réseau. Depuis son point initial d'implantation $\mathbf{m}_{init}$, l'attaquant commence à découvrir ce qui l'entoure. De plus, grâce à des *Techniques* de latéralisation[2], il pivote de machine en machine et prend conscience de l'environnement dans lequel il progresse. Pour faire écho au concept de connaissance situationnelle abordé dans le Chapitre 1, il s'agit là de l'exécution du processus de *Situational Assessment*. Dans cet état, l'attaquant est très bruyant. Les actions techniques qu'il effectue sont **dissonantes et hasardeuses**, par rapport à l'activité légitime et habituelle observable sur le SI ; elles dénotent des actions légitimes qui se déroulent habituellement dans le SI. À chaque exécution, il dissémine des traces dans le réseau et sur les machines du SI. L'attaquant continue à évoluer dans cet état, et potentiellement à se propager dans tout le SI, tant qu'il n'a pas obtenu un accès sur la machine qui héberge son objectif final, c'est-à-dire tant que $(\mathbf{m}_{final}, \mathbf{u}) \notin \mu$. Il progresse donc en exécutant les techniques $\mathbf{t}_j \in \mathbf{TTP}_A$.

2. *Lateral Movement*

Dominance de la machine. Quand l'attaquant a découvert la machine qui héberge son objectif final et qu'il a obtenu un accès dessus, $(\mathbf{m}_{final}, \mathbf{u}) \in \mu$, il essaye de dompter la machine qui héberge cet objectif final pour être en mesure d'appliquer efficacement ses *techniques ultimes*. À ce titre, il est possible que l'attaquant prenne le temps d'apprendre de nouvelles techniques pour construire des capacités adaptées aux effets recherchés sur cet objectif final. Dans cette phase de dominance, l'attaquant se concentre sur la machine $\mathbf{m}_{final}$. Avant de quitter cet état, il identifie le meilleur chemin pour se rendre de son point d'implantation initial $\mathbf{m}_{init}$ jusqu'à la machine qui héberge son objectif final $\mathbf{m}_{final}$.

Phase d'Exploitation

Dès que l'attaquant a réussi à dompter la machine, qu'il a identifiée comme celle hébergeant son objectif final, la phase d'*Exploration* prend fin pour laisser la place à la phase d'*Exploitation*. Au cours de cette phase, l'attaquant va régulièrement vérifier qu'il est toujours en mesure de produire des effets sur son objectif final et il va également produire ces effets. Dans notre modèle, nous divisons cette phase en trois états. Dans un premier temps, l'attaquant se trouve dans un état où il vérifie ses capacités opérationnelles sur le SI afin de s'assurer qu'il est en mesure d'atteindre son objectif final. Dans un second temps, afin de préserver ses capacités et de rester discret, il se place dans un état d'*Attente*. Enfin, l'attaquant sera dans un état où il produira les effets sur son objectif final. Dans les états de la phase d'*Exploitation*, les actions techniques sont optimisées. Cela signifie qu'elles semblent légitimes et que l'empreinte sur le SI est faible, contrairement aux actions conduites dans la phase d'*Exploration*.

Vérification des capacités opérationnelles. Quand la machine qui héberge l'objectif final $\mathbf{m}_{final}$ est sous le contrôle de l'attaquant, il va construire un itinéraire entre son point d'implantation initial $\mathbf{m}_{init}$ et cette machine. Puis, il va vérifier qu'il est en mesure de mettre en œuvre ses *techniques ultimes* sur cette machine. Si l'une de ces techniques échoue, il retourne dans un état de *reconsidération de l'intérêt*, décrit dans la suite de cette section. Si elles réussissent, l'attaquant passe dans un état d'*attente*.

Attente. Dans cet état, l'attaquant patiente. Régulièrement, il se réveille et passe dans l'état de *vérification des capacités opérationnelles* ou dans l'état d'*effets sur objectif*.

Effets sur objectif. En suivant l'itinéraire qu'il a construit et validé dans la phase de *vérification des capacités opérationnelles*, l'attaquant met en œuvre ses *techniques ultimes* qui lui permettent d'atteindre son objectif final. Comme pour l'état de *vérification des capacités opérationnelles*, si ces techniques échouent, l'attaquant sera dans un état de *reconsidération de l'intérêt* et si elles réussissent, il retournera dans un état d'*attente*.

Phase de Prise de décision

La phase de *prise de décision* possède un seul état : *Reconsidération de l'intérêt*. Dans cette phase, l'attaquant n'exécute pas d'actions techniques sur le SI. Cependant, il va confronter deux paramètres :

— le niveau de ressources disponibles (*i.e.*, temps, argent, main-d'œuvre, outillage), ce qui correspond au coût effectif de l'attaque ;
— son intérêt à poursuivre l'attaque (*motivation*).

Reconsidération de l'intérêt. À la suite d'un événement imprévu rencontré dans le SI ou d'un désintérêt pour sa cible exprimé par les commanditaires, l'attaquant peut se retrouver dans cet état non technique où il va mettre en perspective le gain estimé de son attaque et son coût réel. Cet événement imprévu peut être la conséquence d'une contre-mesure implémentée par le défenseur. Cela peut également être à cause d'une difficulté technologique, dont le coût d'apprentissage et de maîtrise serait trop élevé. Si l'attaquant considère qu'il a toujours un intérêt à reconquérir les territoires perdus [3], il reviendra dans un état antérieur de la phase d'*Exploration*, sinon il renoncera à l'attaque. Dans ce modèle, nous considérons que tant que l'attaquant est capable de produire un effet sur l'objectif final, il maintient sa présence dans le SI.

2.3 Confrontation de Nuke avec deux campagnes

Dans cette section, nous confrontons notre modèle à deux campagnes d'attaque, qui ont été décrites dans des publications. Nous avons choisi de travailler sur ces attaques, car leurs rapports respectifs font précisément état de leurs déroulements étape après étape ;

3. Par extension de l'analogie de WikiLeaks dans leur analyse des capacités de Vault7 : *If there is a military analogy to be made, the infestation of a target is perhaps akin to the execution of a whole series of military maneuvers against the target's territory including observation, infiltration, occupation and exploitation.* [48]

contrairement à l'immense majorité des rapports disponibles qui se focalisent plutôt sur les moyens techniques utilisés par les attaquants au cours de leurs campagnes.

Les deux campagnes d'attaques choisies sont :

— la fuite de données subie par Equifax, une société américaine d'analyse financière ;

— le sabotage de TV5Monde, une chaîne de télévision française.

Bien que ces rapports décrivent un grand nombre d'événements techniques de l'attaque, ils demeurent incomplets parce qu'ils ne considèrent pas les problèmes qu'ont pu rencontrer les attaquants et auxquels ils ont dû faire face au cours de leurs progressions. Ainsi, les rapports ne relatent que les actions malveillantes subies par les victimes et qui ont été détectées par les défenseurs. Nous avons tout de même pu interpréter la plupart des informations, que nous avons ensuite projetées sur notre modèle, notamment pour les phases d'*Exploration* et d'*Exploitation*. Enfin, nous avons présumé de l'existence de transitions vers la phase de *prise de décision* qui pouvait intervenir à différentes phases de la campagne. L'instanciation de notre modèle fournit une vision opérationnelle sur le déroulement de l'attaque plutôt que chronologique ou purement technique.

2.3.1 La fuite de données subie par Equifax (2017)

Contexte et primo accès

Equifax est une société américaine d'analyse financière, cotée en bourse, qui emploie onze mille personnes partout dans le monde. Elle a été fondée en 1899 et affichait en 2017 un chiffre d'affaires d'environ trois milliards de dollars US. Cette même année, la sécurité informatique de la société a été compromise [49] et un grand nombre de données personnelles de ses clients a été volé.

La Figure 2.3 est la représentation de cette fuite de données selon notre modèle Nuke. Dans cette figure, les transitions entre états, qui ont été explicitement précisées dans le rapport public, sont tracées avec des lignes pleines ; et les transitions dont nous présumons l'existence sont tracées avec des lignes pointillées.

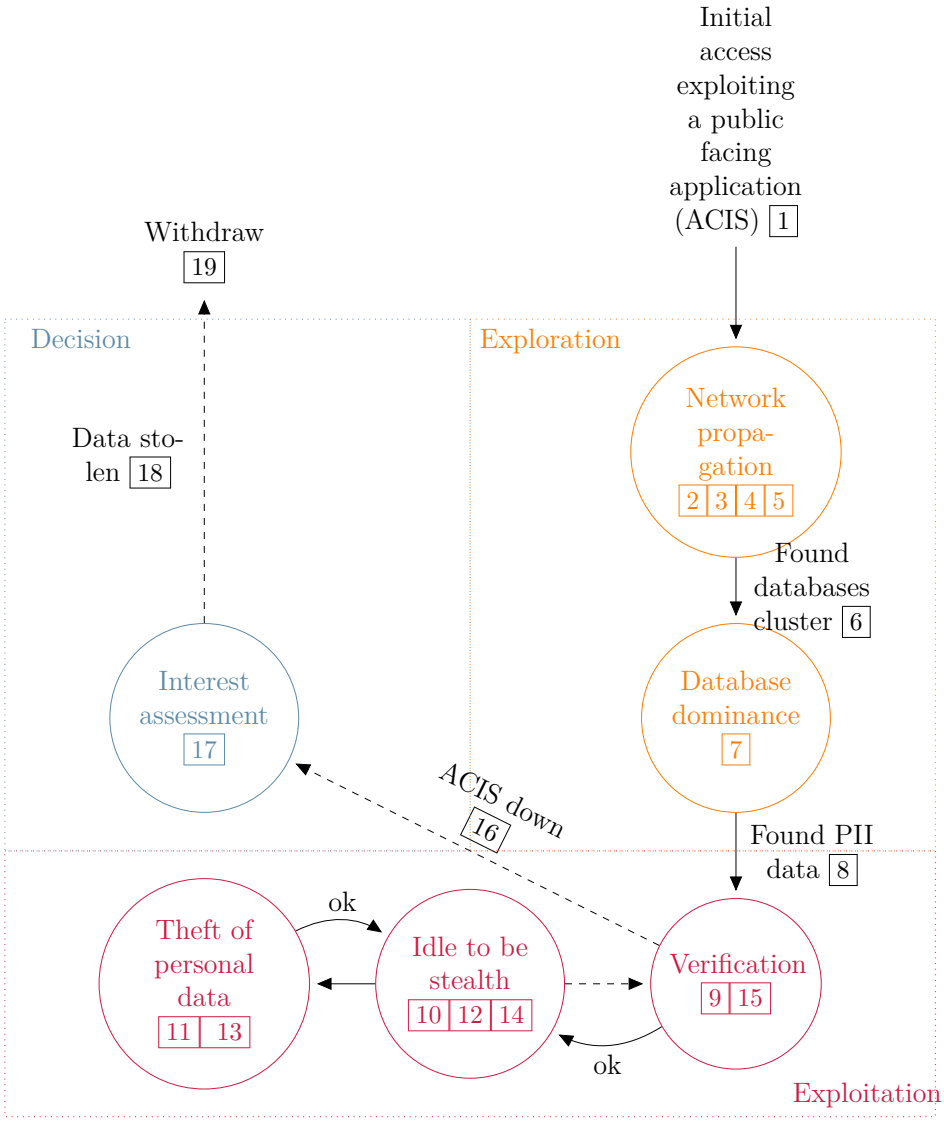

FIGURE 2.3 – Instanciation du modèle Nuke pour la fuite de données subie par Equifax

Le 13 mai 2017, l'attaquant exploita une vulnérabilité connue sur la solution Apache Struts, qui était installée sur une application publiquement exposée 1 ($\mathbf{t}_{1190}$) d'un système d'Equifax appelé *Automated Consumer Interview System* (ACIS). Ce système permettait aux clients de corriger d'éventuelles informations incorrectes inscrites dans leurs dossiers.

L'attaquant put ainsi établir son primo accès dans l'environnement ACIS. Il installa un *webshell* ($\mathbf{t}_{1505.003}$) sur ce premier serveur compromis. La seule machine qu'il connaissait à ce stade était ce serveur.

$$\mu_0 = \{(\mathbf{m}_{init}, user_{init})\} \text{ avec } \mathbf{m}_{init} = ACIS\_WS1\,^{4}$$

L'objectif final de l'attaquant était de collecter des données à caractère personnel (*Personally Identifiable Information* (PII)). À cet instant, l'attaquant ne connaissait pas la forme de cet objectif dans le SI (fichiers, bases de données, flux réseau...).

Phase d'Exploration

Propagation réseau. L'attaquant commença par découvrir le voisinage de la machine $\mathbf{m}_{init}$. Ainsi, il découvrit l'environnement ACIS qui était en fait composé de deux serveurs web et de deux serveurs d'application 2. Il installa des *webshells* 3 ($\mathbf{t}_{1505.003}$) sur chacune de ces nouvelles machines découvertes. Puis, il accéda à un répertoire partagé sur le réseau ($\mathbf{t}_{1039}$) et découvrit des identifiants stockés en clair dans un fichier de configuration de base de données ($\mathbf{t}_{1552.001}$). Il poursuivit sa découverte du SI ($\mathbf{t}_{1018}$, $\mathbf{t}_{1046}$) et trouva dans une zone réseau proche de l'environnement ACIS quarante-huit bases de données 4 et qui n'étaient pas liées à ACIS. L'attaquant réutilisa les identifiants découverts sur la précédente application pour obtenir un accès à ces bases de données. Il put ainsi confirmer sa capacité à fouiller ces bases 5.

$$\mu_1 = \mu_0 \cup \{(DB1, user1), \ldots, (DB48, user48)\}$$

Il avait désormais identifié que son objectif final était hébergé sur ce cluster de bases de données, mais il ne savait pas encore comment les exploiter pour atteindre son objectif 6. Afin de comprendre le fonctionnement de ces bases de données, il exécuta près de 9000 requêtes SQL qui lui ont permis de caractériser les différentes tables 7, notamment en récupérant leurs métadonnées.

4. Le nom donné à cette machine n'est pas son nom réel.

Dominance des bases de données. L'attaquant put finalement trouver une table contenant les PII [8]. Il put également déterminer la séquence de techniques adaptée à l'atteinte de son objectif final. Cette séquence de *techniques ultimes* était de :

— collecter les données en base et les stocker dans des fichiers ($\mathbf{t}_{1005}$) ;
— compresser ces fichiers ($\mathbf{t}_{1560}$) ;
— les rendre accessibles dans l'arborescence d'un serveur web de la victime et les télécharger depuis l'extérieur du réseau ($\mathbf{t}_{1048}$).

Phase d'Exploitation

Vérification des capacités opérationnelles. Il vérifia ses procédures et utilisa un *webshell* pour valider l'exfiltration de quelques données [9].

Période d'attente. L'équipe de réponse à incident découvrit par la suite que l'attaquant était resté dans le SI 76 jours sans se faire détecter [10].

Effets sur l'objectif. Afin de conduire ses actions sous le radar du défenseur, l'attaquant procéda à l'extraction des données personnelles de 145,5 millions de clients (américains, britanniques et canadiens) et les exfiltra progressivement [11] [12] [13] [14]. Cela s'est traduit par l'exécution de 275 requêtes sur la base de données. Il opéra sur une longue période afin de ne pas lever d'alertes liées à la volumétrie des données exfiltrées du SI.

Phase de Prise de décision

Reconsidération de l'intérêt. Après une intervention visant à rétablir la surveillance des flux *Secure Sockets Layer* (SSL) de l'environnement ACIS, l'équipe *Equifax Countermeasures* (le défenseur) découvrit la brèche de sécurité et l'attaque. Le défenseur identifia rapidement le point d'accès initial de l'attaquant, où il restait dormant [15]. Quelques jours plus tard, le 30 juillet 2017, le défenseur décida d'éteindre le portail web ACIS [16]. Puisque l'attaquant avait déjà collecté toutes les données qu'il visait, il n'avait plus d'intérêt à poursuivre sa campagne [17]. Il s'est donc retiré [19] avec les données dérobées [18].

2.3.2 Le sabotage de TV5Monde (2015)

Contexte et primo accès

TV5Monde est une chaîne télévision publique française, fondée en 1984. Elle diffuse ses programmes sur une douzaine de chaînes thématiques dans près de 200 pays et touchant 350 millions de téléspectateurs potentiels. En 2015, la société a été frappée par une cyberattaque [50] (transcrite par Suiche [51]). Cette attaque s'est concrétisée par l'affichage d'écrans noirs pendant plusieurs heures sur la majorité des canaux de la chaîne. La Figure 2.4 est la représentation de cette attaque selon notre modèle Nuke. Dans cette instanciation, les transitions entre états qui ont été explicitement décrites dans le rapport public sont marquées par des lignes pleines. Les transitions dont nous présumons l'existence sont marquées par des lignes pointillées.

Après une première tentative vaine de compromission du réseau au travers d'un serveur isolé, en février 2015, l'attaquant pénétra pour la première fois dans le réseau de TV5Monde en utilisant le *Virtual Private Network* (VPN) ($\mathbf{t}_{1133}$) avec un compte légitime d'un prestataire de service ($\mathbf{t}_{1078}$) [1]. Cette action lui permit d'utiliser un canal légitime en tant qu'accès initial.

$$\mu_0 = \{(\mathbf{m}_{init}, user_{init})\} \text{ avec } \mathbf{m}_{init} = VPN$$

Le principal objectif technique de l'attaquant était de saboter la diffusion des chaînes de télévision de TV5Monde. À ce moment de l'attaque, il ne savait pas encore quelle machine était la plus adaptée pour conduire cette action.

Phase d'Exploration

Propagation réseau (compromission du domaine). Depuis son point d'implantation, l'attaquant commença par scanner les machines clientes du VPN, mais également les autres machines des réseaux internes de TV5Monde [2] ($\mathbf{t}_{1046}$). Le 6 février, il fit la découverte de deux machines Windows appelées `ROB1` et `ROB2`. Il y installa son *Remote Access Trojan* (RAT). Puis, le 11 février [4], il utilisa un autre compte d'un prestataire externe, qui jouissait des privilèges d'administrateur du domaine Windows.

Il créa ensuite un nouveau compte avec ces mêmes privilèges d'administration de domaine [5] ($\mathbf{t}_{1136}$) afin de conserver durablement son niveau de privilèges et d'assurer l'efficacité de sa progression dans le SI. La connaissance qu'avait l'attaquant du SI augmenta quand il prit le contrôle de quatre machines supplémentaires.

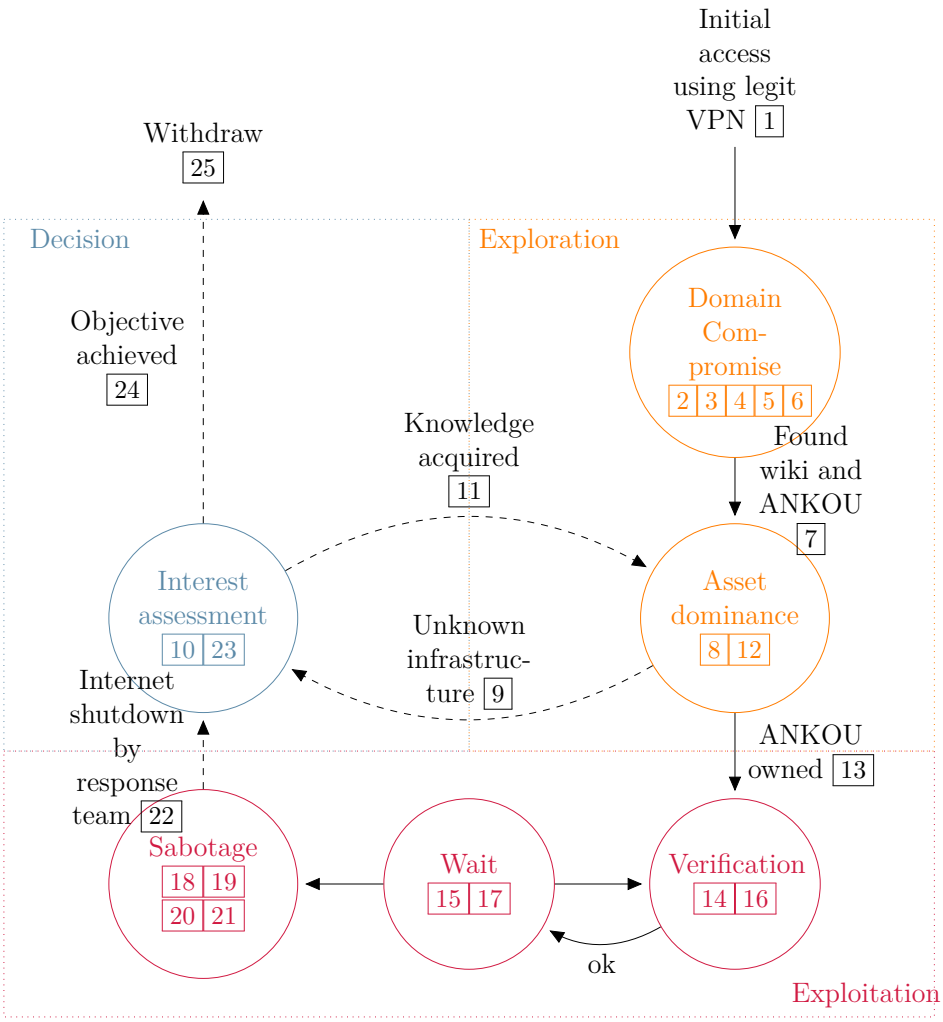

FIGURE 2.4 – Instanciation du modèle Nuke pour le sabotage de TV5Monde

Puis, il cibla le Wiki de TV5Monde [6] afin de collecter des informations ($\mathbf{t}_{1083}$) et des identifiants ($\mathbf{t}_{1552}$) sur la partie dite "métier" de la société de la victime. Dans ces documents, il découvrit une machine, appelée `ANKOU` [7], qu'il considéra comme la machine qui lui permettrait d'atteindre son objectif technique. Ce choix semble avoir été motivé par le positionnement de la machine dans le SI et par les privilèges que l'attaquant avait acquis dessus.

Dominance de la machine `ANKOU`. Après avoir découvert `ANKOU`, l'attaquant disposait de suffisamment d'information et de documentation technique sur les équipements présents dans l'infrastructure de la victime [8]. Il allait donc s'appuyer sur ces sources d'information pour élaborer un scénario, qui lui permettrait par la suite d'atteindre son objectif final : le sabotage des chaînes de TV5Monde. À partir de la mi-mars, l'attaquant semblait avoir effectué des recherches depuis l'extérieur du réseau, probablement parce qu'il ne connaissait pas certains équipements de l'infrastructure et qu'il avait besoin de les maquetter [9]. Le coût de cette étude était donc moindre par rapport à l'intérêt que l'attaquant avait pour atteindre son objectif final [10]. Il poursuivit donc en ce sens et revint dans le réseau de la victime quelques jours plus tard [11]. Cette fois, il décida de s'implanter sur `ANKOU` en y déposant également son RAT [12]. À ce moment, il prit complètement le contrôle de cette machine qui lui permit d'atteindre son objectif final [13].

Phase d'Exploitation

Vérification des capacités opérationnelles. Quelques jours plus tard, l'attaquant se reconnecta au réseau afin de vérifier la pertinence de son scénario final et notamment que les comptes qu'il possédait étaient toujours valides. Il pu ainsi confirmer qu'il était toujours capable d'interagir avec l'actionneur, qui produirait, au moment opportun, l'effet souhaité par le commanditaire sur l'objectif final [14].

Période d'attente. Pendant quelques semaines, aucune activité associée à l'attaquant n'a été observée [15]. Le 8 avril 2015 à 15h40, l'attaquant effectua une dernière vérification et s'assura qu'il était toujours capable de se connecter aux multiplexeurs, aux encodeurs, aux commutateurs réseau et aux routeurs [16]. Puis, il patienta quelques heures [17].

Effet sur objectif : sabotage. Finalement, à 19h57, il détruisit la configuration IP ($\mathbf{t}_{1498}$, $\mathbf{t}_{1499}$) des encodeurs et des multiplexeurs [18]. À 20h58, il défaça le site web de la victime ($\mathbf{t}_{1491}$) [19]. À 21h48, pour finir son sabotage, il effaça les micrologiciels ($\mathbf{t}_{1495}$) des commutateurs réseau et des routeurs [20]. À 22h40, il supprima plusieurs machines virtuelles sur l'hyperviseur ESX ($\mathbf{t}_{1485}$), notamment le serveur de messagerie [21]. À 23h50, l'équipe de réponse à incident prit la décision de couper le lien entre le réseau de la victime et Internet. Cela a rendu impossible l'accès au réseau pour l'attaquant [22].

Phase de Prise de décision

Reconsidération de l'intérêt. Il est probable que l'attaquant ait pu comprendre que le défenseur était en train de le chasser, parce que son action avait été visible (c'était son objectif). Le ratio coût-intérêt pu également lui paraître trop élevé [23], et parce qu'il avait déjà atteint son objectif [24], il se retira et ne donna pas suite à son attaque [25].

2.4 Conclusion du deuxième chapitre

Ces deux confrontations nous permettent de confirmer que, malgré le peu de données relatives aux cycles opérationnels disponibles publiquement dans les rapports de CTI, il est possible de reconstruire les phases des campagnes grâce notre modèle Nuke. Nous avons également mis en avant l'existence d'une phase dite de "prise de décision" et de son état de "reconsidération de l'intérêt", qui peut être sollicitée par l'attaquant depuis n'importe quelle autre phase de l'attaque, en fonction des situations imprévues qu'il rencontrerait.

Aussi, notre modèle souligne que certaines phases utilisent des moyens techniques et que d'autres impliquent des facteurs tiers qui permettent de mettre en perspective le coût de la campagne avec l'intérêt réel de poursuivre. De plus, il confirme l'aspect cyclique de ces attaques et ainsi permet de mieux percevoir la durabilité, caractéristique prépondérante des menaces sophistiquées. Nous notons cependant que le peu de rapports retraçant publiquement les chronologies des attaques laisse transparaître l'existence des états et des transitions décrits dans ce modèle. De plus, il convient de préciser que ces rapports sont biaisés puisqu'ils s'inscrivent dans une démarche d'éradication de la menace, généralement *a posteriori*. Dans un contexte de cyberdéfense active, notre vision permet à un défenseur de considérer l'opportunité de forcer un attaquant à régresser et à retourner dans un état antérieur. Cela permet également de l'inciter à révéler ses capacités opérationnelles ; ou encore de le dissuader à poursuivre son attaque.

Ces travaux ont été publiés dans

Aimad Berady, Valérie Viet Triem Tong, Gilles Guette, Christophe Bidan et Guillaume Carat : ***Modeling the Operational Phases of APT Campaigns***.
IEEE Conference on Computational Science & Computational Intelligence (CSCI),
décembre 2019, USA.
DOI : 10.1109/CSCI49370.2019.00023 HAL Id : hal-02379869

Chapitre 3

Mise en perspective des visions des adversaires

Ce chapitre présente la vision que nous avons développée au cours de nos travaux sur les menaces persistantes avancées et plus particulièrement sur la dualité qui existe entre l'attaquant et le défenseur. Nos observations ont été publiées dans "*From TTP to IoC : Advanced Persistent Graphs for Threat Hunting*" [52].

3.1 Introduction

Dans le cadre d'une réponse à incident, les défenseurs qui luttent contre une menace sophistiquées ont besoin de découvrir rapidement l'espace de propagation de leur adversaire. Cette phase opérationnelle, appelée le *Threat Hunting*, conduit les défenseurs à traquer les attaquants au sein du SI compromis [53]. Dans ce chapitre, nous proposons un modèle formel qui décompose et abstrait les différents éléments d'une attaque ; du point de vue de l'attaquant et de celui du défenseur. La création de ce modèle théorique nous a permis de construire deux graphes persistants à partir d'un ensemble de données partagées par les attaquants et les défenseurs ; des objets et des composants. Cela permet :

- pour un acteur omniscient, de comparer la différence de connaissance et de perception entre les deux parties (attaquant et défenseur) ;
- pour un attaquant, de prendre conscience des traces laissées sur le SI de sa victime ;
- pour un défenseur, d'améliorer l'efficacité de sa chasse (*Threat Hunting*) en identifiant des faux positifs et en adaptant la politique de journalisation du SI.

Nous avons ensuite évalué notre modèle en utilisant les traces d'une attaque imitant les TTP du groupe APT29, une menace réelle tristement célèbre, prélevée dans un réseau de simulation suivant un scénario imaginé par le MITRE Corporation. Nous avons mesuré expérimentalement la qualité de l'architecture défensive. Enfin, nous avons déterminé la stratégie la plus efficace pour exploiter les données collectées par le défenseur afin de produire de la CTI actionnable qui permettrait de dévoiler l'attaquant.

Dans le cadre d'un audit en mode *Red Team*, il n'existe pas de mesure précise pour évaluer l'efficacité des prestations des différents acteurs. Une approche naïve pourrait être de mesurer le temps nécessaire à la *Red Team* pour prendre le contrôle total du SI, d'une part ; et, d'autre part, le temps nécessaire à la *Blue Team* pour détecter et répondre à l'attaque. Nous considérons que le temps passé est étroitement lié à la pertinence des données collectées et à la manière dont elles sont exploitées, tant par l'attaquant que par le défenseur. C'est pourquoi, dans ce chapitre, nous formalisons les processus défensifs et offensifs mis en œuvre dans le cadre d'une campagne d'attaque. Nous formalisons

également les connaissances qu'ont l'attaquant et le défenseur sur la campagne d'attaque, ainsi que les évolutions de ces connaissances. Les points de vues convergents que nous décrivons rendent possible la confrontation des perceptions respectives des adversaires s'opposant au cours d'une campagne d'attaque.

La connaissance de l'attaquant est construite à partir :

— de l'exécution de ses procédures d'attaques choisies parmi ses TTP ;

— des ressources qu'il expose lorsqu'il les exécute ;

— des composants du SI qu'il a compromis.

La connaissance du défenseur est, elle, construite à partir :

— des traces qu'il a collectées sur le SI compromis ;

— de l'exploitation de ces traces grâce à ses procédures défensives.

Notre modèle permet de progresser sur deux fronts. Premièrement, il propose une représentation de haut niveau d'une campagne d'attaque, qui permet d'évaluer rapidement les progressions opérationnelles de l'attaquant, en quête de son objectif final, et du défenseur, avec l'ambition de découvrir l'espace de propagation de l'attaquant. Cette représentation peut également être utilisée par un acteur omniscient afin de mesurer le succès d'un exercice *Red versus Blue*. Deuxièmement, le modèle permet d'identifier les leviers qui pourraient être actionnés dans le cadre de procédures défensives ainsi que leurs impacts. Ces leviers sont paramétrables en ajustant les données d'entrée (*i.e.*, configurations des sondes et règles de détection). Nous avons ensuite pu conduire une expérimentation qui repose sur notre modèle, en nous appuyant sur la campagne d'attaque issue du projet public *Mordor* [41]. Cette campagne imite le groupe APT29 dans un scénario conçu par le MITRE Corporation, dans le cadre des *ATT&CK Evaluations* [54].

Dans ce chapitre, nous présentons dans la Section 3.2 une vue d'ensemble du modèle et des concepts manipulés. Les Sections 3.3 et 3.4 présentent respectivement les visions de l'attaquant et du défenseur. La Section 3.5 détaille l'architecture de l'expérimentation et précise les prérequis pour une intégration dans un SI. Enfin, la Section 3.6 discute les résultats et propose des perspectives d'évaluation de l'efficacité de la chaîne de détection.

3.2 Vue d'ensemble du modèle

Ce chapitre formalise le processus de *Threat Hunting* suivi par une équipe de réponse à incident, afin de pouvoir proposer une solution d'évaluation de l'efficacité de la chaîne de détection. Nous commençons notre chaîne par le point de vue de l'attaquant, car il a l'initiative et c'est lui qui donne le tempo. Ce concept d'initiative est par ailleurs détaillé par Monte dans "*Network attacks and exploitation*" [55].

3.2.1 L'infrastructure

L'infrastructure ciblée est le SI qui héberge l'objectif final de l'attaquant. Dans ce SI, nous distinguons les *composants* des *objets*. Les composants sont des *assets* (*i.e.*, des machines) avec des capacités de journalisation d'événements. Nous considérons ici que le défenseur dispose d'un contrôle total sur ces composants. Les objets sont des attributs relatifs à des événements et qui peuvent être exploités dans le cadre d'une opération de détection d'intrusion, de réponse à incident ou d'investigation numérique. Le standard STIX définit ces objets comme étant des *Observables* [56]. Chacun de ces objets joue un rôle précis dans le contexte de l'événement qui les laisse apparaître. Ce rôle spécifie la fonction qu'occupe l'objet dans l'événement. L'ensemble des rôles existants est noté $\mathbb{R}$. Chaque rôle $\mathbb{r}$ est à associé à un type unique qui spécifie la nature de l'objet qui possède ce rôle. Nous écrivons $\mathbb{T}$ l'ensemble des types possibles et le type $\mathbb{t}$ associé à un rôle $\mathbb{r}$ est défini grâce à la fonction $\tau : \mathbb{R} \rightarrow \mathbb{T}$.

Dans notre implémentation, nous avons utilisé le modèle de données *MITRE Cyber Analytics Repository* (CAR) [57] afin de nommer les différents rôles des objets. Il est à noter qu'un même objet peut avoir plusieurs rôles associés à différents types. Par exemple, l'objet `maliciousfile.com` peut endosser le rôle *destination hostname* avec le type *domain* ; le rôle *file name* ou encore le rôle *executable* avec le type *file*. Les trois premières colonnes du Tableau 3.1 détaillent quelques exemples d'objets ainsi que leurs types et rôles. Les types qui sont le plus fréquemment utilisés sont les *IP addresses*, *domain names* et *files*.

Dans ce modèle nous ne considérons que les composants de la victime, puisque ceux de l'attaquant ne peuvent pas être atteints par le défenseur adoptant une posture strictement défensive. Néanmoins, le défenseur peut observer certains objets liés aux composants de l'attaquant (*e.g.*, un nom de domaine, une valeur de *hash*, une adresse IP), parce que l'attaquant peut les avoir exposés au cours de son attaque.

$\mathbf{o} \in \mathbf{O}_D$ Object	$\mathbb{r} \in \mathbb{R}$ Role	$\tau(\mathbb{r}) = \mathbb{t} \in \mathbb{T}$ Type	$\mathbf{D}_A(\mathbb{t}, \mathbf{o})$ Attacker directory (extract)	$\mathbf{D}_D(\mathbb{t}, \mathbf{o})$ Defender directory (extract)
`10.0.1.4`	src_ip dest_ip	ip address	(ip address, scranton.dmevals.com)	(ip address, scranton.dmevals.com)
`m.exe`	file_name exe	file	None	None
`dmevals.com`	dest_hostname file_name	domain file	(domain, `newyork.dmevals.com`) None	None None
`warehouse`	dest_hostname	domain	None	(domain, `warehouse.dmevals.com`)

TABLE 3.1 – Exemples d'objets, leurs types relatifs, les rôles possibles, ainsi que leurs existences dans chacun des annuaires.

Cependant, l'attaquant sera en mesure de découvrir et de gagner des accès sur une partie des composants de la victime. C'est pour ces raisons que dans ce chapitre nous nous concentrons principalement sur l'ensemble des composants du SI.

Ainsi, nous notons :

— $\mathbf{C}$ l'ensemble de ces composants ;

— $\mathbf{O}_D$ l'ensemble des objets relatifs à la victime ;

— $\mathbf{O}_A$ l'ensemble des objets relatifs à l'attaquant ; et

— $\mathbf{O} = \mathbf{O}_D \cup \mathbf{O}_A$

Le modèle permet également d'exprimer la très discutée [58] notion de riposte de la part du défenseur (*Hack Back*), en considérant le défenseur comme un attaquant et l'attaquant comme sa victime.

3.2.2 Le périmètre de l'attaquant

L'attaquant peut être un individu, un groupe ou encore une *Red Team*, mais dans un souci de clarté, nous l'appelons simplement "l'attaquant". De la même manière, la présence de plusieurs attaquants dans le SI n'est pas un obstacle puisque l'ambition de ce modèle est de fournir une vue exhaustive sur les composants compromis du SI.

L'attaquant est à l'initiative de l'attaque et possède ses propres composants. Ils font partie de son infrastructure. Il dispose également d'une collection de procédures, notée $\mathbf{TTP}_A$. La matrice ATT&CK du MITRE capitalise ces différentes collections. Les procédures sont paramétrées par des objets, provenant de $\mathbf{O}_D$ ou de $\mathbf{O}_A$. Enfin, les procédures sont exécutées sur des composants du SI uniquement si l'attaquant a déjà découvert ces composants (la notion de découverte est abordée dans le Chapitre 4 au travers de la formalisation de plusieurs de ces techniques). Le périmètre de l'attaquant est entièrement formalisé dans la Section 3.3.

3.2.3 Le périmètre du défenseur

Le défenseur peut être simplement la victime, une équipe de réponse à incident de l'entité attaquée, une équipe externe ou encore une *Blue Team*. Comme pour l'attaquant, nous l'appelons simplement "le défenseur". Le défenseur dispose de procédures défensives, notées $\mathbf{TTP}_D$, qui lui permettent de faire face à des campagnes d'attaques menées contre le SI qu'il protège. Le défenseur est en mesure d'observer les événements qui se produisent sur les composants de $\mathbf{C}$, qu'il a choisi de surveiller. Les événements pertinents à surveiller sont spécifiés dans les configurations des sondes. Nous notons σ une configuration. Nous avons généralisé et formalisé dans la Section 3.4 deux procédures défensives :

— p_{logs} qui permet au défenseur de générer des traces lorsque se produisent des événements sur les composants du SI ;

— $p_{hunting}$ qui permet d'exploiter ces traces afin d'identifier dans $\mathbf{C}$ les composants compromis par l'attaquant.

3.2.4 La confrontation des visions

Nous proposons ici de représenter la perception d'un attaquant au cours d'une campagne au travers d'un graphe $\mathbf{G}_A$ reliant les objets et les composants. Ce graphe est calculé à partir d'une séquence de procédures exécutées par l'attaquant et laisse apparaître les objets impliqués. Une partie de ces objets représente des informations caractéristiques que l'attaquant ne peut pas dissimuler et qu'il est conscient d'exposer. Similairement, nous représentons la perception du défenseur grâce à un second graphe $\mathbf{G}_D$, dont le calcul résulte de l'exécution des procédures défensives. La Figure 3.1 propose une vue globale de la transformation du paramètre de procédure (attaquant) jusqu'à l'IoC (défenseur).

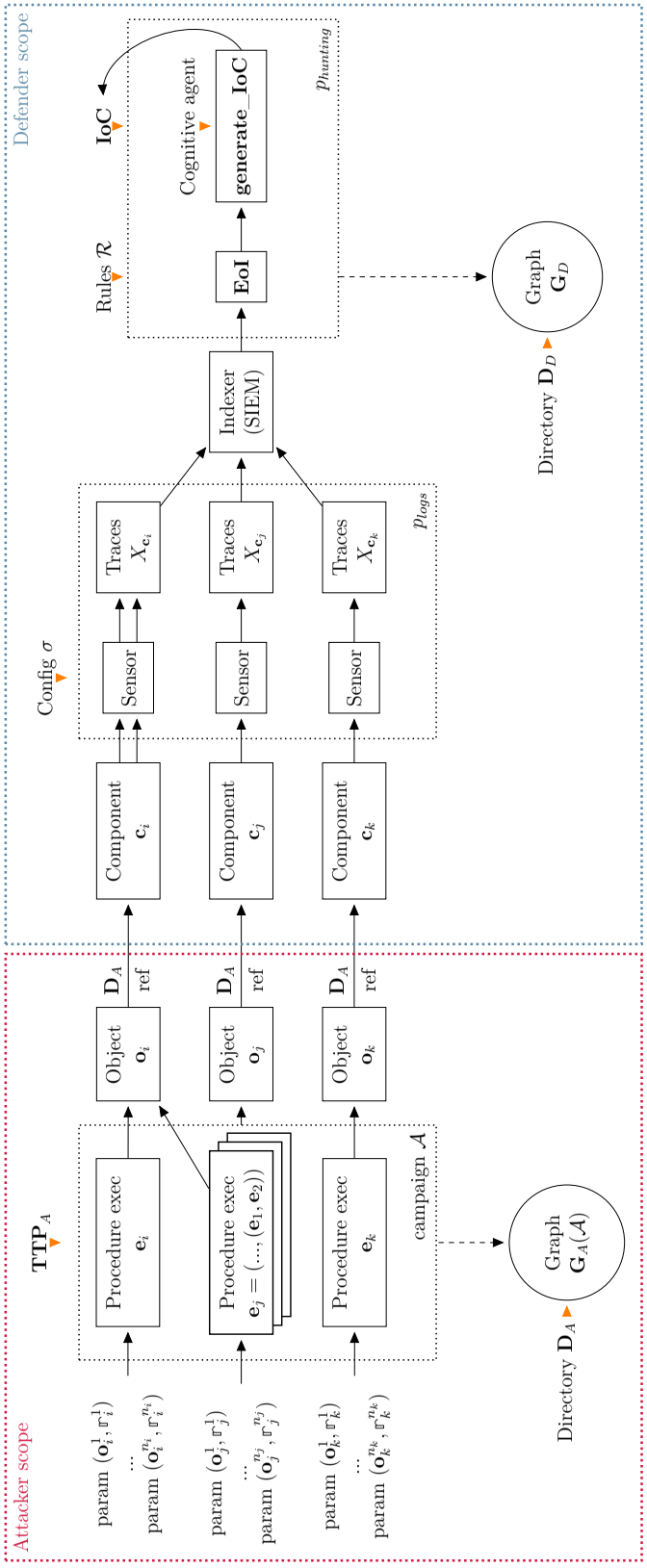

FIGURE 3.1 – Vue globale du modèle.

Les connaissances respectives de l'attaquant et du défenseur à propos du SI sont enrichies grâce à leurs annuaires. En effet, pour chacun d'entre eux, les annuaires permettent de relier des objets aux composants selon leurs propres perspectives (par exemple, la connaissance d'une adresse IP pour désigner une machine). Ainsi, nous notons $\mathbf{D}_A$ l'annuaire de l'attaquant à propos des composants SI et $\mathbf{D}_D$ l'annuaire du défenseur. $\mathbf{D}_A$ ou $\mathbf{D}_D$ peuvent être incomplets, voire imparfaits. Nous les représentons ici par deux fonctions de $\mathbb{T} \times \mathbf{O}_D$ dans $\mathbf{C} \cup \{\mathsf{None}\}$. Cela signifie qu'il s'agit de fonctions dont les arguments sont des paires $(\mathbb{t}, \mathbf{o})$ appartenant à $\mathbb{T} \times \mathbf{O}_D$ et dont le résultat est soit un élément de $\mathbf{C}$, soit None. Quand l'attaquant (respectivement le défenseur) n'a aucune information à propos d'un objet $\mathbf{o}$ avec un type $\mathbb{t}$, ou si l'objet $\mathbf{o}$ d'un type $\mathbb{t}$ n'est relatif à aucun composant, alors $\mathbf{D}_A(\mathbb{t}, \mathbf{o}) = \mathsf{None}$ (respectivement $\mathbf{D}_D(\mathbb{t}, \mathbf{o}) = \mathsf{None}$).

La représentation d'un composant $\mathbf{c}$ du point de vue de l'attaquant peut être différente de la représentation de ce même composant $\mathbf{c}$ du point de vue du défenseur. Par exemple, le défenseur peut connaître une machine, désignée par une adresse IP ou son nom de machine, comme étant un serveur Web, qui possède des fichiers ; alors que l'attaquant, lui, ne connaît que son adresse IP, voire celle d'une autre interface réseau. Dans l'exemple du Tableau 3.1, les deux dernières colonnes présentent, le cas échéant, un objet avec son type spécifique faisant référence à un composant de l'annuaire de l'attaquant $\mathbf{D}_A$ et du défenseur $\mathbf{D}_D$.

Dans la Section 3.5, nous présentons comment la comparaison de $\mathbf{G}_A$ et $\mathbf{G}_D$ permet de confronter deux perceptions de la campagne d'attaque ; selon le point de vue de l'attaquant et celui du défenseur. Puis, la Section 3.6 propose des manières d'améliorer la qualité des données collectées dans le cadre d'une opération de *Threat Hunting*, par exemple. L'objectif est de disposer d'IoC plus pertinents et de limiter les faux positifs parmi les *Event of Interest* (EoI).

3.2.5 Expérimentation sur la campagne simulée d'APT29

Dans le cadre de sa démarche d'évaluation des produits de cybersécurité, le MITRE crée régulièrement des scénarios opérationnels inspirés d'attaques réelles, imitant ainsi les groupes d'attaquants connus. Parmi eux, deux scénarios concernent le groupe APT29, qui est considéré par la communauté comme un groupe d'attaquants sponsorisés par un État. Le groupe APT29 est actif depuis 2008. Au cours de ces campagnes, les attaquants mettent en œuvre différentes procédures afin de collecter et d'exfiltrer des données sensibles du SI. Les deux scénarios détaillent les étapes d'une campagne d'attaque imitant les TTP

d'APT29, telles qu'elles ont été observées par la communauté des experts de la sécurité informatique. Roberto Rodriguez est à l'initiative du projet Mordor [41], qui a permis de publier un jeu de données d'événements enregistrés suite à l'exécution des procédures issues des scénarios du MITRE. Dans la suite de cette partie, nous avons fait le choix de regrouper les deux scénarios puisqu'ils présentent de nombreuses similitudes et qu'ils font référence au même groupe d'attaquants et à la même infrastructure de la victime. Parmi les traces du jeu de données, nous nous sommes concentrés sur celles produites par la sonde Sysmon, qui est celle recommandée à ce jour par la communauté [59]. L'infrastructure du SI est présentée dans la Section 3.5.2 et les résultats de l'expérimentation sont discutés dans la Section 3.6.

3.3 Vision de l'attaquant

L'attaquant est modélisé par un graphe d'objets et de composants.

— Les **composants** font partie de l'infrastructure de la victime et l'attaquant interagit avec eux à mesure qu'il exécute des procédures.

— Les **objets**, manipulés par l'attaquant sous la forme de paramètres de procédures, se retrouvent dans les traces collectées par le défenseur. L'attaquant est conscient de disséminer ces objets dans le SI au cours de sa campagne.

Cette section détaille la construction de ce graphe en considérant à la fois les actions de l'attaquant, dont la séquence correspond à la campagne, et l'évolution de la connaissance de l'attaquant à propos du SI.

3.3.1 Actions de l'attaquant

Une campagne d'attaque, notée $\mathcal{A}$ est composée d'une séquence $\mathbf{e}_1, \ldots, \mathbf{e}_N$ d'exécutions de N procédures $p_1, \ldots, p_N$ sur les composants $\mathbf{C}$ du SI.

Nous considérons ici que l'attaquant connaît au moins un composant dit "initial" : celui qui lui sert de point d'entrée dans le SI. Ce composant initial peut avoir été découvert par l'attaquant par le biais d'une reconnaissance externe sur les services publiquement exposés du SI ou encore par une action d'ingénierie sociale visant à piéger des utilisateurs du SI. Quand l'attaquant a compromis au moins un composant, il est en mesure d'exécuter des procédures à l'intérieur du SI. C'est à ce moment que commence la phase opérationnelle d'*Exploration* et plus spécifiquement de *Propagation réseau*, telle qu'elle a été présentée

en Section 2.3.1.

L'exécution $\mathbf{e}$ d'une procédure $p \in \mathbf{TTP}_A$ nécessite, de la part de l'attaquant, la connaissance de :

— une **machine**, notée $\mathbf{m}(\mathbf{e})$. Il s'agit d'un composant $\mathbf{c} \in \mathbf{C}$. Dans l'annuaire $\mathbf{D}_A$, la machine est désignée par un objet $\mathbf{o}$ et un type $\mathbb{t}$ relatif au composant. Ainsi, on a $\mathbf{c} = \mathbf{D}_A(\mathbb{t}, \mathbf{o})$. Cette machine est l'hôte sur laquelle la procédure sera exécutée.

— des **paramètres**, qui permettent de configurer la procédure grâce à un ensemble $\{(\mathbf{o}_1, \mathbb{r}_1), \ldots, (\mathbf{o}_n, \mathbb{r}_n)\}$ de couples d'objets $\mathbf{o}$ et rôles $\mathbb{r}$.

— des éventuelles exécutions de **sous-procédures**, notées $(\mathbf{e}_1, \ldots, \mathbf{e}_m)$, qui correspondent à l'invocation de procédures dans le contexte d'une exécution de p. Pour chaque $\mathbf{e}_i$, la machine $\mathbf{m}(\mathbf{e}_i)$ est accessible via un objet relatif $(\mathbf{o}_j, \mathbb{r}_j)$, qui apparaît dans les paramètres de p et tel que $\mathbf{m}(\mathbf{e}_i) = \mathbf{D}_A(\tau(\mathbb{r}_j), \mathbf{o}_j)$.

Une exécution $\mathbf{e}$, désignée par un identifiant unique $id_{\mathbf{e}}$, d'une procédure $p \in \mathbf{TTP}_A$ sur un composant $\mathbf{c} = \mathbf{D}_A(\mathbb{t}, \mathbf{o}) \neq \mathsf{None}$ peut ainsi être formalisée de la façon suivante :

$$\mathbf{e} = (id_{\mathbf{e}}, p, (\mathbf{o}, \mathbb{t}), ((\mathbf{o}_1, \mathbb{r}_1), \ldots, (\mathbf{o}_n, \mathbb{r}_n)), (\mathbf{e}_1, \ldots, \mathbf{e}_m))$$

Le Listing 1 présente l'exemple d'une procédure régulièrement utilisée par les attaquants. Cette procédure permet d'effectuer un mouvement latéral (*Lateral Movement*) en utilisant l'outil `psexec`. Cela permet à l'attaquant d'exécuter des commandes sur un système distant et d'en récupérer la sortie sur son système local. Dans cet exemple, le composant `SCRANTON.dmevals.local` (ligne 3) est celui sur lequel est exécutée la procédure principale et `NASHUA.dmevals.local` (ligne 19) est le composant sur lequel est invoquée la sous-procédure. Ici, l'attaquant lance à distance le fichier insidieusement nommé `python.exe` (ligne 27). De plus, il profite sur les deux composants des privilèges de l'utilisateur `pbeesly` (lignes 8 et 26).

```
{
  "name": "8.C",
  "cmp": "SCRANTON.dmevals.local",
  "description": "Execute payload on secondary computer",
  "obj":
    [
      {"role": "dest_ip", "value": "192.168.0.5"},
      {"role": "user", "value": "pbeesly"},
      {"role": "command_line",
      "value": ".\\PsExec64.exe -accepteula \\\\NASHUA -u
↪ \"dmevals\\pbeesly\" -p \"Fl0nk3rt0n!T0by\" -i {WILDCARD}
↪ \"C:\\Windows\\Temp\\python.exe\""},
      {"role": "exe", "value": "PsExec64.exe"},
      {"role": "dest_hostname", "value": "NASHUA"},
      {"role": "image_path", "value": "C:\\Program
↪ Files\\SysinternalsSuite\\PsExec64.exe"}
    ],
  "invol":
    [
      {
        "name": "8.C.1",
        "cmp": "NASHUA.dmevals.local",
        "description": "Execute payload",
        "obj":
          [
            {"role": "dest_ip", "value": "192.168.0.4"},
            {"role": "dest_ip", "value": "10.0.1.6"},
            {"role": "dest_fqdn", "value": "SCRANTON.dmevals.local"},
            {"role": "user", "value": "pbeesly"},
            {"role": "exe", "value": "python.exe"},
            {"role": "image_path", "value":
↪ "C:\\Windows\\Temp\\python.exe"}
          ]
      }
    ]
}
```

Listing 1: Procédure d'attaque : exemple d'exécution selon la vision de l'attaquant d'un mouvement latéral utilisant `psexec`.

3.3.2 Propagation de l'attaquant dans le SI

Nous représentons ici la progression de l'attaquant dans le SI par un graphe. Chaque exécution d'une procédure d'attaque nécessite pour l'attaquant de faire appel à des objets ou des composants connus, soit parmi ses propres ressources, soit parce qu'ils ont été découverts auparavant dans le SI. Dans un souci de simplicité, nous considérons que dans ce modèle la connaissance de l'attaquant à propos du SI est gelée et qu'elle est décrite dans l'annuaire de l'attaquant $\mathbf{D}_A$. Néanmoins, la formalisation des procédures des techniques de `Discovery` traitées dans la Section 4.4.4 permet de rendre dynamique cet annuaire.

Petit pas

Pour chaque exécution de $\mathbf{e}$ d'une procédure d'attaque p, nous construisons un graphe orienté $\mathbf{G}(\mathbf{e})$, dans lequel les nœuds sont les objets impliqués dans la campagne d'attaque et les composants auxquels ils sont attachés. Ce graphe représente une étape de la campagne d'attaque. Formellement, étant donnée une exécution

$$\mathbf{e} = (id_{\mathbf{e}}, p, (\mathbf{o}, \mathbb{t}), ((\mathbf{o}_1, \mathbb{r}_1), \ldots, (\mathbf{o}_n, \mathbb{r}_n)), (\mathbf{e}_1, \ldots, \mathbf{e}_m))$$

$\mathbf{G}(\mathbf{e})$ est défini par

$$\mathbf{G}(\mathbf{e}) = (V_{\mathbf{e}}, \rightarrow_{\mathbf{e}}) \oplus \bigoplus_{i=1}^{m} \mathbf{G}(\mathbf{e}_i)$$

et est calculé à partir :

— des nœuds $V_{\mathbf{e}}$, qui désignent les composants, tous les objets impliqués dans l'exécution de la procédure et les potentiels composants relatifs, connus par l'attaquant :

$$\{\mathbf{m}(\mathbf{e}), \mathbf{o}_1, \cdots, \mathbf{o}_n\} \cup \{\mathbf{c}' = \mathbf{D}_A(\tau(\mathbb{r}_i), \mathbf{o}_i) \neq \mathsf{None}\}$$

— des liens $\rightarrow_{\mathbf{e}}$ qui relient les composants avec les objets et par transitivité les composants entre eux le cas échéant. Des liens relient également les objets faisant référence aux composants selon l'annuaire de l'attaquant :

$$\bigcup_{i=1}^{n} \left\{ \mathbf{m}(\mathbf{e}) \xrightarrow{id_{\mathbf{e}}, \mathbb{r}_i} \mathbf{o}_i \right\} \cup \left\{ \begin{array}{l} \mathbf{o}_i \xrightarrow{ref} \mathbf{c}' \mid \\ \mathbf{c}' = \mathbf{D}_A(\tau(\mathbb{r}_i), \mathbf{o}_i) \neq \mathsf{None} \end{array} \right\}$$

— ainsi que l'union des graphes $\bigoplus_{i=1}^{m} \mathbf{G}(\mathbf{e}_i)$ produits suite à l'exécution des sous-procédures de $\mathbf{e}$. L'opérateur $\oplus$ désigne ici l'union entre les deux graphes (*i.e.*, union des liens et union des nœuds).

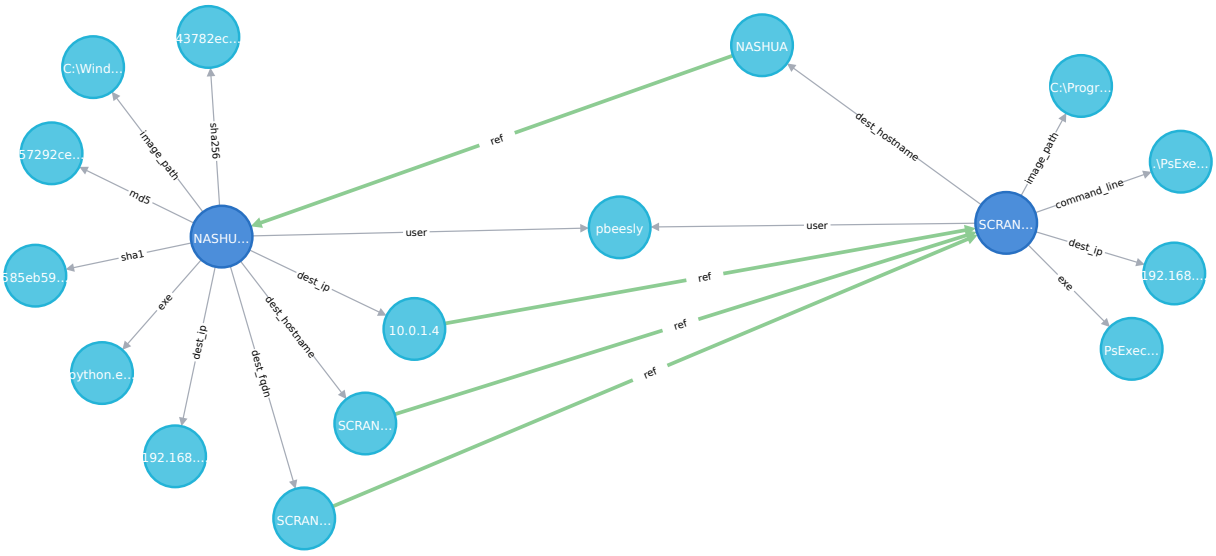

FIGURE 3.2 – $\mathbf{G}$(`psexec`)

La Figure 3.2 présente le graphe $\mathbf{G}$(`psexec`) calculé à partir de l'exécution spécifique de la procédure `psexec` telle que décrite précédemment dans le Listing 1. Dans ce graphe, les nœuds bleus foncés sont les composants et les nœuds bleus clairs sont les objets. Les étiquettes sur les liens noirs fins correspondent au rôle de l'objet dans la procédure. Les liens verts épais correspondent aux objets qui sont relatifs à un autre composant, selon l'annuaire de l'attaquant. Il est à préciser que, par essence, une procédure de `Lateral Movement` implique deux composants, car elle permet à un attaquant à se déplacer d'un composant vers un autre. Les deux composants sont donc représentés sur le graphe et leurs connexions indirectes utilisent des liens notés "ref".

Grand pas

Une campagne d'attaque $\mathcal{A}$ est composée d'une séquence d'exécution de procédures $(\mathbf{e}_1, \ldots, \mathbf{e}_n)$. Nous représentons cette campagne par le graphe $\mathbf{G}_A$, qui est le résultat de l'union de tous les graphes associés à chaque exécution de procédures :

$$\mathbf{G}_A = \bigoplus_{i=1}^{N} \mathbf{G}(\mathbf{e}_i)$$

La génération de ce graphe permet à l'attaquant de se représenter tous les objets qu'il a exposés au cours de sa campagne. Ce graphe peut être considéré comme l'empreinte visible de l'attaquant exposée à un défenseur. De plus, dans le contexte d'un exercice *Red versus Blue*, ce graphe permet à un acteur omniscient (*i.e.*, la *White Team*) de mesurer la distance entre ce que la *Red Team* a exposé et ce que la *Blue Team* a effectivement détecté.

La Figure 3.3 est la représentation du graphe $\mathbf{G}_A$ calculé par l'attaquant au cours de la campagne simulée APT29. Il contient 49 étapes qui correspondent chacune à l'exécution d'une procédure. Quatre composants compromis sont présents :

— `SCRANTON`
— `NASHUA`
— `NEWYORK`
— `UTICA`

Ils correspondent aux quatre machines sur lesquelles l'attaquant a effectivement exécuté des procédures au cours de sa campagne. Ces quatre composants sont également présents dans son annuaire. Ce scénario décrit parfaitement la notion de furtivité de l'attaquant puisqu'il visite un faible nombre de machines et limite ainsi son empreinte.

3.4 Vision du défenseur

Nous considérons cette fois la même campagne d'attaque, mais selon le point de vue du défenseur. Le but du défenseur est d'être en mesure de représenter l'espace de propagation de l'attaquant dans le SI. Plus cette représentation sera complète, plus les chances d'éradication de la menace seront élevées. En d'autres termes, l'objectif du défenseur est de calculer un graphe d'objets et de composants qui doit être constitué des mêmes éléments que ceux présents sur le graphe de l'attaquant.

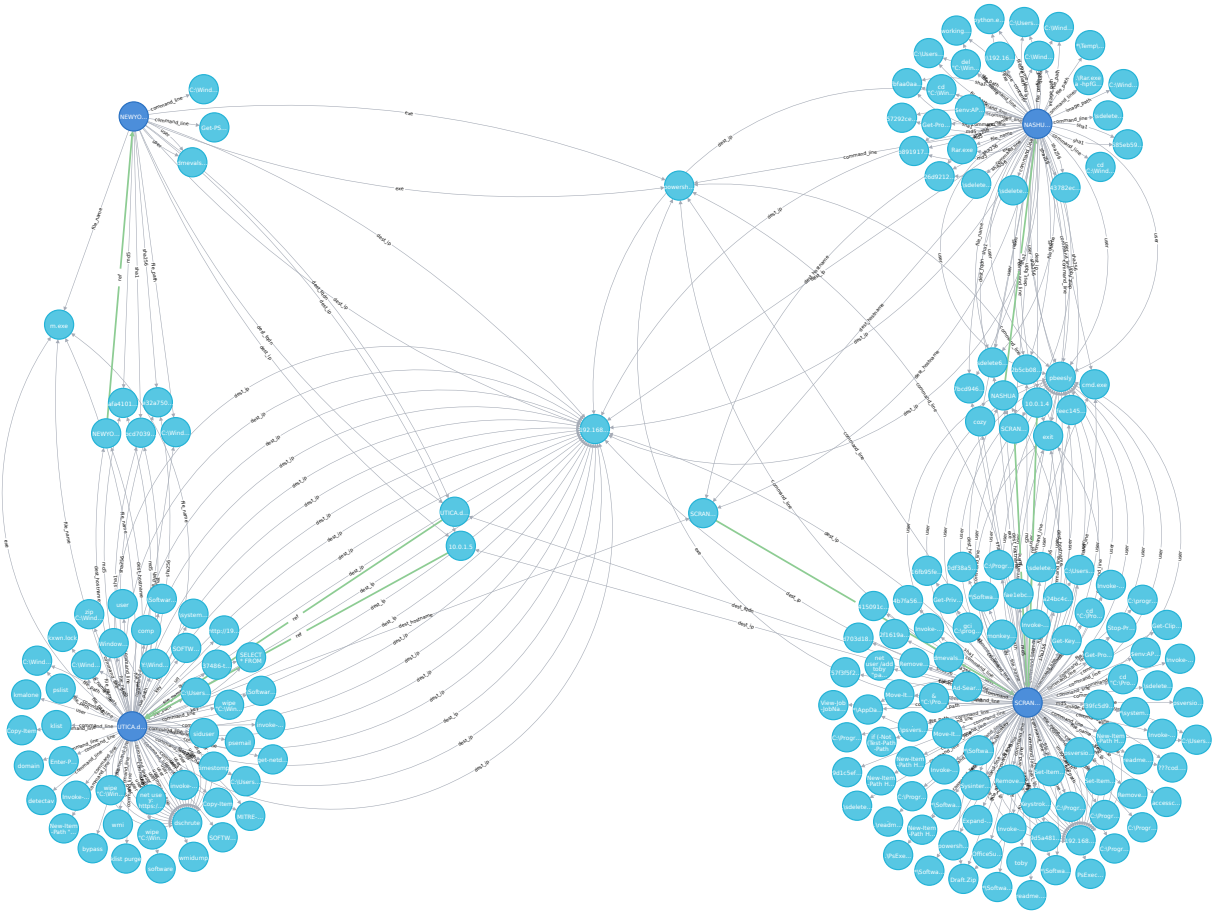

FIGURE 3.3 – $\mathbf{G}_A$ calculé par l'attaquant au cours de la campagne simulée APT29. Source : https://gitlab.inria.fr/cidre-public/from-ttp-to-ioc-dataset/-/blob/master/graph_attacker_full.svg

Pour arriver à ses fins, le défenseur dispose de procédures défensives, ses $\mathbf{TTP}_D$. Le cadre sémantique que nous proposons dans ce chapitre fournit une spécification formelle de ces procédures défensives. La formalisation de procédures d'attaques, telles que celles des techniques de `Discovery` ou de `Lateral Movement` est présentée dans la Section 4.4.

Dans la suite de cette section, nous décrivons formellement deux procédures défensives : p_{logs} et $p_{hunting}$. Leurs implémentations respectives sont des procédures que le défenseur pourra mettre en œuvre au cours d'une opération de réponse à incident. Les procédures exécutées par l'attaquant génèrent, elles, des événements sur les composants.

Nous représentons un événement ev par le tuple $(\epsilon, \mathbf{c}, \mathcal{O})$, où :

— $\epsilon \in \mathfrak{E}$ correspond au type d'événement ;
— $\mathbf{c} \in \mathbf{C}$ correspond au composant sur lequel l'événement a été observé ; et
— $\mathcal{O} = \{(\mathbf{o}_1, \mathbb{r}_1), \cdots, (\mathbf{o}_m, \mathbb{r}_m)\} \subseteq \mathbf{O} \times \mathbb{R}$ est l'ensemble qui contient tous les objets impliqués dans cet événement, ainsi que leurs rôles associés.

En fonction de la configuration des sondes, les événements peuvent générer des traces. Une trace $\mathbf{x}$ est un tuple $(id_{\mathbf{x}}, t, \epsilon, \mathbf{c}, \mathcal{O})$, où :

— $id_{\mathbf{x}}$ est l'identifiant unique d'une trace ;
— t est le *timestamp* d'observation de l'événement ;
— $\epsilon \in \mathfrak{E}$ est le type d'événement qui a provoqué la génération de la trace ;
— $\mathbf{c} \in \mathbf{C}$ est le composant où l'événement qui a provoqué la trace a été observé ; et
— $\mathcal{O}$ est l'ensemble de tous les objets impliqués dans la trace avec leurs rôles associés.

Un même objet peut prendre plusieurs rôles différents dans une même trace et par conséquent apparaître plusieurs fois dans l'ensemble $\mathcal{O}$. Dans une même trace, chaque rôle est unique.

Le défenseur peut influencer la qualité de ses résultats en actionnant trois leviers :

— la configuration des sondes ;
— les règles de détection ; et
— la base de référence **IoC**.

Dans un premier temps, le défenseur décide, grâce à l'exécution de sa procédure défensive p_{logs}, quels composants du SI sont supervisés et quels événements doivent produire des traces. Ce sont ces traces qui seront transmises au *Security Information Event Management* (SIEM). Cette procédure, détaillée dans la Section 3.4.1, va générer un grand nombre de traces qui permettent au défenseur d'identifier parmi elles les traces relatives à l'activité de l'attaquant. La seconde procédure défensive, appelée $p_{hunting}$, détaillée en Section 3.4.2, exploite ces traces afin de découvrir l'espace de propagation de l'attaquant.

3.4.1 Supervision du système d'information

Le défenseur est théoriquement en mesure de mettre en supervision l'immense majorité des événements qui se produisent sur un composant du SI. Pourtant, la plupart de ces événements ne sont pas pertinents dans une démarche de détection ou de réponse à incident. Aussi, ces événements peuvent être amenés à générer un nombre trop important de faux positifs dans le système d'alerte du SIEM.

Les traces remontées par les sondes sont le seul moyen à disposition du défenseur pour être en mesure de percevoir une partie de l'activité de l'attaquant. C'est pourquoi le défenseur doit prêter une attention toute particulière à la configuration σ des sondes. Plus une trace sera significative de l'activité de l'attaquant, plus elle aura de la valeur aux yeux du défenseur.

Nous considérons ici que tous les composants peuvent être observés et que les sondes déployées, pour en assurer la supervision, sont configurées selon σ. Dans cette configuration, le défenseur définit les types d'événements considérés comme étant pertinents pour chaque composant $\mathbf{c} \in \mathbf{C}$. Ce sont donc ces événements qui induisent la création de traces. La configuration $\sigma(\mathbf{c})$ rend possible la supervision du composant $\mathbf{c}$. Cette configuration est définie par un ensemble de tuples $(\epsilon, \phi, \mathsf{R}_\epsilon)$ où :

- $\epsilon \in \mathfrak{E}$ est le type d'événement qui sera spécifié dans la trace créée par la sonde, quand la condition ϕ est vraie.
- ϕ exprime les propriétés sur les objets invoqués dans l'événement observé $(\epsilon, \mathbf{c}, \mathcal{O})$ et leurs rôles correspondants. Nous écrivons $\mathcal{O} \models \phi$ quand un objet dans $\mathcal{O}$ satisfait une condition ϕ. La syntaxe de ϕ et la relation de satisfaction $\models$ sont définies ci-dessous.
- $\mathsf{R}_\epsilon \subseteq \mathbb{R}$ est un ensemble qui contient les rôles considérés comme étant pertinents pour le type d'événement ϵ. La notion pertinence relève ici de l'intérêt qu'aurait un défenseur à s'appuyer sur ce type d'événement pour conduire son opération de réponse à incident. Par exemple, la chaîne de caractère `10.0.1.4` (l'objet), ayant pour rôle `dest_ip` et un type d'*observable* `IP address`, peut aisément être recherchée dans le SIEM et désigne le même objet sur tous les composants. Ce type est par conséquent considéré comme étant pertinent ; il a sa place dans R_ϵ.

Les expressions ϕ des configurations de sondes sont construites à partir des constantes logiques true, false et les opérateurs and, or, not appliqués sur des couples $(\mathbb{r}, y)$ où $\mathbb{r}$ est un rôle et y est une propriété sur un objet qui joue ce rôle ; nous écrivons $y(\mathbf{o})$ afin d'exprimer le fait qu'un objet $\mathbf{o}$ satisfait la propriété y :

$$\phi ::= \mathsf{true} \mid \mathsf{false} \mid (\mathbb{r}, y) \mid \mathsf{not}\ \phi \mid \phi\ \mathsf{and}\ \phi \mid \phi\ \mathsf{or}\ \phi$$

Puis, étant donné un ensemble $\mathcal{O} = \{(\mathbf{o}_1, \mathbb{r}_1), \cdots, (\mathbf{o}_n, \mathbb{r}_n)\} \subseteq \mathbf{O} \times \mathbb{R}$, la relation de satisfaction $\models$ d'une condition est définie par :

$$\begin{array}{lll}
\mathcal{O} \models \mathsf{true} & & \\
\mathcal{O} \models (\mathbb{r}_i, y) & \text{ssi} & y(\mathbf{o}_i) \\
\mathcal{O} \models \mathsf{not}\ \phi & \text{ssi} & \mathcal{O} \not\models \phi \\
\mathcal{O} \models \phi_1\ \mathsf{and}\ \phi_2 & \text{ssi} & \mathcal{O} \models \phi_1 \text{ et } \mathcal{O} \models \phi_2 \\
\mathcal{O} \models \phi_1\ \mathsf{or}\ \phi_2 & \text{ssi} & \mathcal{O} \models \phi_1 \text{ ou } \mathcal{O} \models \phi_2
\end{array}$$

Nous considérons ici que pour chaque rôle dans ϕ, il existe un objet dans $\mathcal{O}$ qui joue ce rôle et qu'un rôle peut se produire au plus une fois dans $\mathcal{O}$.

```
<Sysmon schemaversion="4.22">
    <EventFiltering>
        <ProcessCreate onmatch="exclude">
            <CommandLine condition="is">
                \SystemRoot\System32\smss.exe
            </CommandLine>
        </ProcessCreate>
        <NetworkConnect onmatch="include">
            <Image condition="begin with">
                C:\Windows\Temp
            </Image>
        </NetworkConnect>
    </EventFiltering>
</Sysmon>
```

Listing 2: Exemple de configuration d'une sonde Sysmon, partie intégration d'une procédure p_{logs}.

Le Listing 2 présente une partie d'une configuration de la sonde Sysmon. Sysmon est un service maintenu par Microsoft pour Windows [1] et Linux [2]. Il permet de superviser et journaliser l'activité du système. Sysmon est la solution gratuite reconnue par la communauté comme étant la plus efficace à ce jour. Dans cet exemple, le composant supervisé est une machine Windows. La condition ϕ exprime que Sysmon va générer une trace :

— pour les événements de type `ProcessCreate` (création de processus) une trace sera générée, sauf pour les événements dont l'objet $\mathbf{o}$ a le rôle $\mathbb{r}$ `CommandLine` et possède une valeur qui correspond à `\SystemRoot\System32\smss.exe`. Cette règle permet donc d'éliminer un certain nombre de faux positifs dus à la fréquence élevée d'apparition de cet événement sur un système sain.

— pour les événements de type `NetworkConnect` (connexion réseau), une trace sera générée, pour tous les événements dont l'objet $\mathbf{o}$ a le rôle `Image` (*i.e.*, fichier binaire PE) et est présent sur le système de fichiers dans le répertoire `C:\Windows\Temp`.

Le Tableau 3.2 donne des exemples de rôles pertinents R_ϵ pour ces deux types d'événements ϵ. Nous utilisons ici le modèle de données de CAR.

Type d'événement $\epsilon \in \mathfrak{E}$	Rôles pertinents $\mathsf{R}_\epsilon \subseteq \mathbb{R}$
ProcessCreate (CAR : `process create`)	Image (`image_path`) CommandLine (`command_line`) ParentImage (`parent_image_path`) ParentCommandLine (`parent_command_line`) User (`user`) Hashes (`md5`, `sha1`, `sha256`)
NetworkConnect (CAR : `flow start`)	Image (`image_path`) User (`user`) SourceIp (`src_ip`) SourceHostname (`src_hostname`) DestinationIp (`dest_ip`) DestinationHostname (`dest_hostname`)

TABLE 3.2 – Rôles pertinents dans une trace Sysmon selon le type d'événement et leurs noms CAR respectifs.

1. https://docs.microsoft.com/fr-fr/sysinternals/downloads/sysmon
2. https://github.com/Sysinternals/SysmonForLinux

Concevoir une configuration de sonde parfaitement adaptée est une démarche complexe. Ainsi, il est nécessaire pour le défenseur de trouver un juste milieu entre une journalisation extrême de toute l'activité du système et une politique de journalisation trop restrictive (incluant des objets spécifiques ou excluant des objets génériques). Dans la première option, la sonde générerait un trop grand nombre de traces et risquerait d'avoir un impact négatif sur les performances du SI (congestion sur le réseau ou encore de saturation des espaces de stockage). Dans la seconde option, la condition ne serait que très rarement satisfaite et trop peu de traces seraient produites, ce qui conduirait le défenseur à mal évaluer la menace à laquelle il fait face, car certains événements ne seraient pas traduits dans le SIEM (*i.e.*, faux négatif).

Formellement, c'est la procédure défensive p_{logs} du défenseur qui détermine les événements à superviser et les traces à produire. Cette procédure est paramétrée par :

— σ, qui spécifie les configurations des sondes associées aux composants ; et

— E, qui correspond à l'ensemble des événements produits sur les composants.

La procédure défensive p_{logs} génère un ensemble de traces $\mathbf{X_c}$ pour chaque composant $\mathbf{c}$.

```
procedure p_logs(σ, E) :
    for all ev = (ϵ, c, O) ∈ E:
        if (ϵ, φ, R_ϵ) ∈ σ(c) and O ⊨ φ:
            X_c = X_c ∪ {(id_x, t, ϵ, c, {(o, r) ∈ O | r ∈ R_ϵ})}
    return {X_c | c ∈ C}
```

Le Listing 3 est un exemple de trace produite par une sonde Sysmon. Conformément à la configuration présentée dans le Listing 2, la survenue d'un événement de type `Network Connection`, impliquant un objet avec le rôle `Image` et une exécution depuis le répertoire `Temp`, a provoqué la génération d'une trace. Cette trace, transmise au SIEM, indique que cet événement s'est produit sur le composant `NASHUA.dmevals.local`. Elle contient des objets, tels que `DestinationIp`, `User` ou encore le chemin complet de `Image`, qui précisent le contexte dans lequel l'événement s'est produit.

3.4.2 Propagation de l'attaquant du point de vue du défenseur

Nous considérons ici que toutes les traces produites par les sondes sont transmises automatiquement au SIEM. La seconde procédure défensive que nous avons formalisée est la procédure $p_{hunting}$. Cette procédure permet au défenseur de construire un graphe $\mathbf{G}_D$ à partir des traces observées. Le graphe $\mathbf{G}_D$ est créé selon le même modèle que

```
{
    "Hostname": "NASHUA.dmevals.local",
    "UtcTime": "2020-05-02 03:16:19.454",
    "RecordNumber": 353256,
    "SourceName": "Microsoft-Windows-Sysmon",
    "EventID":3,
    "SourceIp": "10.0.1.6",
    "SourcePort": "60215",
    "DestinationIp": "192.168.0.4",
    "DestinationPort": "8443",
    "Image": "C:\\Windows\\Temp\\python.exe",
    "ProcessId": "2172",
    "User": "DMEVALS\\pbeesly",
    ...
}
```

Listing 3: Extrait d'une trace Sysmon générée suite à un événement de type Network connection.

celui utilisé par l'attaquant pour créer le graphe $\mathbf{G}_A$: les nœuds sont des objets ou des composants, les liens entre deux nœuds indiquent que ces objets ou composants sont relatifs à la campagne d'attaque qui est observée par le défenseur. Le graphe $\mathbf{G}_D$ n'est pas construit directement à partir des données brutes de l'ensemble des traces remontées au SIEM, car ces traces impliquent un nombre trop important de composants et d'objets qui ne sont pas pertinents dans une démarche de *Threat Hunting*. C'est pour cette raison que le défenseur a besoin de filtrer les traces sur le SIEM, afin qu'il puisse se focaliser uniquement sur les traces considérées comme des EoI.

Mise en évidence des *Events of Interest*

Le défenseur fait appel à une base de données d'IoC, et à un ensemble de règles de détection $\mathcal{R}$ afin d'identifier les traces qui doivent être considérées comme des EoI. Un IoC est un objet $\mathbf{o}$ avec un type $\mathbb{t}$ qui indique, avec un haut niveau de confiance, la présence d'une activité malveillante dans le SI. Le Tableau 3.3 propose un exemple de base de données d'IoC pouvant être utilisée par le défenseur au cours d'une potentielle opération de *Threat Hunting* pendant la campagne simulée d'APT29. Dans cet exemple, les objets tels que `toby` ou `m.exe` avec les types d'*observables* respectivement *user* et *file*, font sens à être recherchés dans un périmètre local. Ce sont des IoC dits "locaux". Le défenseur décide

de cette classification, car l'utilisateur a été créé et n'existe que dans le SI. Ainsi cela ne présente que peu de sens de le rechercher globalement sur Internet, dans les SI d'autres victimes. Cela risquerait d'apporter un nombre trop grand de faux positifs lors de ses détections. Cependant, `cod.3aka3.scr` et `9d1c5ef38e6073661c74660b3a71a76e`, avec les types d'*observables* respectivement *file* et *hash*, peuvent être recherchés globalement, dans un périmètre plus large que le seul SI. Ce sont des IoC dits "globaux".

Objet $\mathbf{o} \in \mathbf{O}_D$	Type $\mathbb{t} \in \mathbb{T}$	Périmètre
`toby`	user	local
`m.exe`	file	local
`cod.3aka3.scr`	file	global
`9d1c5ef38e6073661c74660b3a71a76e`	hash	global

TABLE 3.3 – Extrait d'une base de données d'IOC pouvant être utilisée pour une opération de *Threat Hunting* pendant la campagne simulée d'APT29.

La base $\mathbf{IoC} = \{(\mathbf{o}_1, \mathbb{t}_1), \cdots, (\mathbf{o}_n, \mathbb{t}_n)\}$ est maintenue par le défenseur qui peut la mettre à jour grâce au partage des connaissances acquises lors de précédentes investigations conduites par d'autres équipes de sécurité de la communauté (IoC globaux) ; ou grâce à ses propres investigations menées dans le SI (IoC locaux).

Une règle de détection $r \in \mathcal{R}$ exprime une condition spécifiquement définie par le défenseur afin d'identifier parmi les traces des EoI. Nous écrivons $\mathbf{x} \models r$ quand la condition spécifiée par r est satisfaite par la trace $\mathbf{x}$. La syntaxe des règles de détection et de la relation de satisfaction $\models$ est définie ci-dessous.

Les règles de détection r sont construites à partir des constantes logiques true, false et des opérateurs and, or, et not appliqués sur des couples (α, y) où α est :

— un *timestamp* t ; ou
— un type d'événement $\epsilon \in \mathfrak{E}$; ou
— un composant $\mathbf{c} \in \mathbf{C}$; ou
— un rôle $\mathbb{r} \in \mathbb{R}$.

et y une propriété appliquée sur α. $y(\alpha)$ exprime que α satisfait la propriété y :

$$r ::= \mathsf{true} \mid \mathsf{false} \mid (\alpha, y) \mid \mathsf{not}\ r \mid r\ \mathsf{and}\ r \mid r\ \mathsf{or}\ r$$

Donc, étant donné une trace $\mathbf{x} = (id_{\mathbf{x}}, t, \epsilon, \mathbf{c}, \mathcal{O})$, la relation de satisfaction $\models$ d'une règle de détection r est définie par :

$$
\begin{array}{lll}
\mathbf{x} \models \mathsf{true} & & \\
\mathbf{x} \models (t', y) & \text{ssi} & t' = t \text{ et } y(t) \\
\mathbf{x} \models (\epsilon', y) & \text{ssi} & \epsilon' = \epsilon \text{ et } y(\epsilon) \\
\mathbf{x} \models (\mathbf{c}', y) & \text{ssi} & \mathbf{c}' = \mathbf{c} \text{ et } y(\mathbf{c}) \\
\mathbf{x} \models (\mathbb{r}_i, y) & \text{ssi} & (\mathbf{o}, \mathbb{r}_i) \in \mathcal{O} \text{ et } y(\mathbf{o}_i) \\
\mathbf{x} \models \mathsf{not}\ r & \text{ssi} & \mathbf{x} \not\models r \\
\mathbf{x} \models r_1\ \mathsf{and}\ r_2 & \text{ssi} & \mathbf{x} \models r_1 \text{ et } \mathbf{x} \models r_2 \\
\mathbf{x} \models r_1\ \mathsf{or}\ r_2 & \text{ssi} & \mathbf{x} \models r_1 \text{ ou } \mathbf{x} \models r_2
\end{array}
$$

L'ensemble des règles de détection $\mathcal{R}$ et la base de données d'**IoC** permettent au défenseur d'implémenter la fonction $\mathbf{EoI}_{\mathcal{R},\mathbf{IoC}}$, qui filtre les traces, dans le SIEM, afin de mettre en évidence celles qui sont des EoI. $\mathbf{EoI}_{\mathcal{R},\mathbf{IoC}}(\mathbf{x})$ fournit donc deux ensembles :

— le sous-ensemble $R_{\mathbf{x}}$ de $\mathcal{R}$, qui contient les règles satisfaites par la trace $\mathbf{x}$; et

— le sous-ensemble $O_{\mathbf{x}}$ de **IoC**, qui contient les objets impliqués dans $\mathbf{x}$.

Par conséquent, une trace $\mathbf{x}$ est un événement d'intérêt si au moins un de ces deux sous-ensembles n'est pas vide. Formellement, étant donnée une trace $\mathbf{x} = (id_{\mathbf{x}}, t, \epsilon, \mathbf{c}, \mathcal{O})$, cette fonction est définie par $\mathbf{EoI}_{\mathcal{R},\mathbf{IoC}}(\mathbf{x}) = (R_{\mathbf{x}}, O_{\mathbf{x}})$, où :

$$
\begin{array}{rl}
 & R_{\mathbf{x}} = \{r \in \mathcal{R} \mid \mathbf{x} \models r\} \\
\text{et} & O_{\mathbf{x}} = \left\{ \begin{array}{l} (\mathbf{o}, \mathbb{t}) \mid (\mathbf{o}, \mathbb{t}) \in \mathbf{IoC} \text{ et} \\ (\mathbf{o}, \mathbb{r}) \in \mathcal{O} \text{ et } \tau(\mathbb{r}) = \mathbb{t} \end{array} \right\}
\end{array}
$$

Le Listing 4 présente une règle de détection fréquemment utilisée par les défenseurs afin de détecter l'exécution de `psexec` en analysant les événements de type `process_creation` observés par les sondes telles que Sysmon. À la suite de cette détection, la fonction **EoI** retourne une trace, à la manière de celle présentée sur le Listing 3.

3. https://github.com/SigmaHQ/sigma/blob/becf3baeb4f6313bf267f7e8d6e9808fc0fc059c/rules/windows/process_creation/proc_creation_win_tool_psexec.yml

```
        title: PsExec Tool Execution
        id: fa91cc36-24c9-41ce-b3c8-3bbc3f2f67ba
        related:
            - id: 42c575ea-e41e-41f1-b248-8093c3e82a28
              type: derived
        status: experimental
        description: Detects PsExec service installation and execution events
    ↪  (service and Sysmon)
        author: Thomas Patzke
        date: 2017/06/12
        modified: 2021/09/21
        references:
            - https://www.jpcert.or.jp/english/pub/sr/ir_research.html
            - https://jpcertcc.github.io/ToolAnalysisResultSheet
        tags:
            - attack.execution
            - attack.t1569.002
            - attack.s0029
        fields:
            - EventID
            - CommandLine
            - ParentCommandLine
            - ServiceName
            - ServiceFileName
            - TargetFilename
            - PipeName
        logsource:
            category: process_creation
            product: windows
        detection:
            sysmon_processcreation:
                Image|endswith: '\PSEXESVC.exe'
                User|startswith: 'NT AUTHORITY\SYSTEM'
            condition: sysmon_processcreation
        falsepositives:
            - Unknown
        level: low
```

Listing 4: Extrait d'une règle de détection Sigma [3] pour `psexec`.

Petit pas

Chaque observation de trace $\mathbf{x} = (id_{\mathbf{x}}, t, \epsilon, \mathbf{c}, \mathcal{O})$ considérée comme un EoI, entraîne la création d'un graphe $\mathbf{G}(\mathbf{x}, R_{\mathbf{x}}, O_{\mathbf{x}})$, où $\mathbf{EoI}_{\mathcal{R},\mathbf{IoC}}(\mathbf{x}) = (R_{\mathbf{x}}, O_{\mathbf{x}})$. Ce graphe représente la trace. Les nœuds de ce graphe sont :

— le composant $\mathbf{c}$ où la trace a été collectée ;
— les objets $\mathbf{o}$ impliqués dans $\mathcal{O}$;
— les composants ayant pour référence un objet de la trace, selon l'annuaire du défenseur $\{\mathbf{c}' = \mathbf{D}_D(\tau(\mathbb{r}), \mathbf{o}) \neq \mathsf{None} \mid (\mathbf{o}, \mathbb{r}) \in \mathcal{O}\}$.

Les liens dans $\mathbf{G}(\mathbf{x}, R_{\mathbf{x}}, O_{\mathbf{x}})$ permettent de connecter :

— tous les objets $\mathbf{o} \in \mathcal{O}$ au composant $\mathbf{c}$:

$$\bigcup_{(\mathbf{o},\mathbb{r})\in\mathcal{O}} \left\{ \mathbf{c} \xrightarrow{\mathbb{r},R_{(\mathbf{o},\mathbb{r})},\text{is\_ioc}} \mathbf{o} \right\}$$

Chaque lien est étiqueté par le rôle de l'objet $\mathbf{o}$, le sous-ensemble $R_{(\mathbf{o},\mathbb{r})}$ de $R_{\mathbf{x}}$ défini par :

$$R_{(\mathbf{o},\mathbb{r})} = \{r \in R_{\mathbf{x}} \mid (id_{\mathbf{x}}, t, \epsilon, \mathbf{c}, \mathcal{O} \setminus \{(\mathbf{o}, \mathbb{r})\}) \not\models r\}$$

et par un booléen is_ioc, dont la valeur est true ssi il existe $(\mathbf{o}, \mathbb{t}) \in \mathbf{IoC}$ tel que $\tau(\mathbb{r}) = \mathbb{t}$. Par conséquent, pour supprimer le lien $\mathbf{c} \xrightarrow{\mathbb{r},R_{(\mathbf{o},\mathbb{r})},\text{is\_ioc}} \mathbf{o}$ de $\mathbf{G}(\mathbf{x}, R_{\mathbf{x}}, O_{\mathbf{x}})$ il suffit de désactiver des règles de détection dans $R_{(\mathbf{o},\mathbb{r})}$ et de supprimer $(\mathbf{o}, \mathbb{t})$ dans la base de données $\mathbf{IoC}$ quand is_ioc est vrai. Concrètement, cette suppression peut être nécessaire dans le cas où le graphe du défenseur contiendrait trop de liens, rendant son exploitation difficile ; et
— tous les composants $\mathbf{c}' = \mathbf{D}_D(\tau(\mathbb{r}), \mathbf{o}) \neq \mathsf{None}$ référencés dans l'annuaire du défenseur. Ces liens sont simplement étiquetés ref.

La Figure 3.4 présente un exemple d'un graphe généré par le défenseur à partir de la trace présentée dans le Listing 3. Ce graphe est donc la perception qu'a le défenseur de l'exécution de la procédure d'attaque `psexec`. Le composant sur lequel a été observée la trace est au centre de ce graphe ; autour de ce composant, les objets pertinents impliqués dans la trace sont représentés. Ce graphe peut également être composé d'objets similaires au graphe de l'attaquant (Figure 3.2) à la suite de l'exécution de cette même procédure.

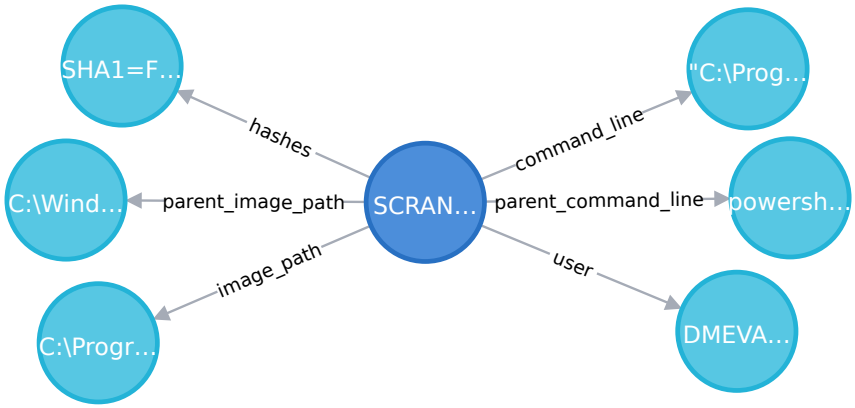

FIGURE 3.4 – Graphe généré à partir d'une trace, considérée comme un EoI.

Grand pas

Une campagne d'attaque $\mathcal{A}$ observée par un défenseur au travers d'une collection de traces $\mathbf{X}$ peut être représentée par le graphe $\mathbf{G}_D = \bigoplus_{\mathbf{x}\in\mathbf{X}} \mathbf{G}(\mathbf{x}, R_\mathbf{x}, O_\mathbf{x})$, résultat de l'union de tous les graphes associés à chaque EoI de l'ensemble de traces $\mathbf{X}$. La Figure 3.5 est le graphe généré par le défenseur pendant la campagne simulée d'APT29, avec les règles du projet public Sigma. Les conditions de génération de ce graphe sont précisées en Section 3.5. Le graphe de la perception du défenseur (Figure 3.5) doit maintenant être comparé avec le graphe de la perception de l'attaquant (Figure 3.3) pour essayer d'en évaluer sa qualité.

Qualité de vision du défenseur

La qualité du graphe $\mathbf{G}_D$ construit par le défenseur est influencée par trois facteurs :

— la configuration σ des sondes ;

— l'ensemble des règles de détection $\mathcal{R}$ implémentée dans le SIEM ; et

— la base de données d'IoC.

Dans un processus de *Threat Hunting*, la démarche de mise à jour des configurations des sondes ou des règles de détection n'est généralement pas compatible avec le tempo de l'opération et aurait un impact trop important sur le SI pour être réalisée en cours de la chasse de l'intrus. Cependant, la base de données des IoC peut être régulièrement mise à jour à partir d'objets pertinents identifiés dans les traces considérées comme des EoI grâce à la fonction **generate_IoC**$((R_\mathbf{x}, O_\mathbf{x}), \mathbf{x}, \mathbf{IoC})$. Tous les objets qui apparaissent dans une trace considérées comme des EoI sont des candidats pour devenir des IoC.

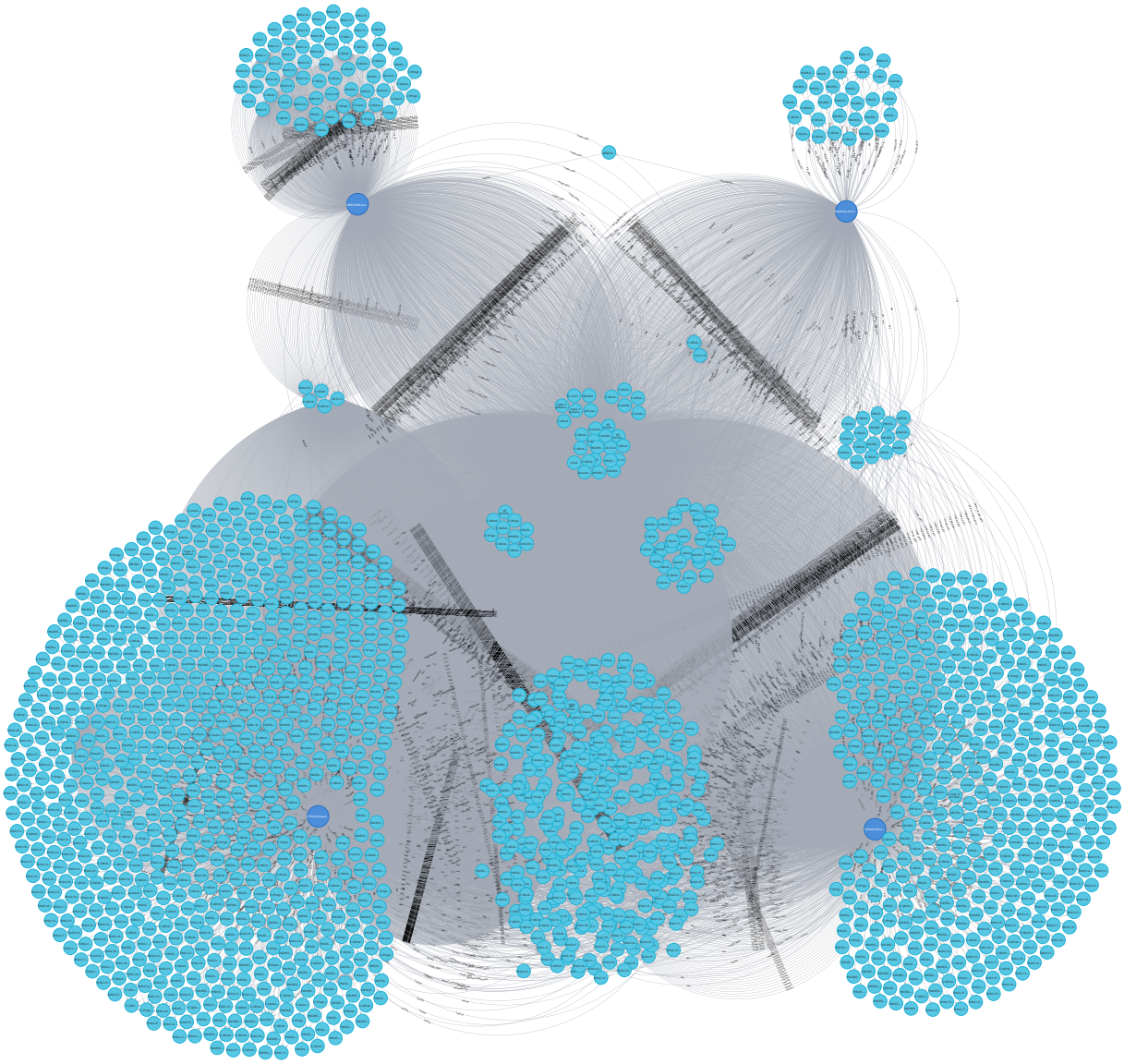

FIGURE 3.5 – $\mathbf{G}_D$ généré par le défenseur pendant la campagne simulée d'APT29. Source : https://gitlab.inria.fr/cidre-public/from-ttp-to-ioc-dataset/-/blob/master/graph_defender_big_full.svg

En particulier, les objets dont les rôles correspondent à des types d'observables tels que : `IP address`, `hashes` ou `domain`. Alors que Kurogome *et al* [60] ont proposé d'automatiser la fonction **generate_IoC**, l'intervention d'un expert doit toujours être considérée.

Le défenseur a donc à sa disposition deux procédures défensives principales :

— la procédure p_{logs}, qu'il utilise afin de désigner les composants à superviser et les traces à transmettre au SIEM ; et
— la procédure $p_{hunting}$, qui complète la précédente en générant le graphe du défenseur et en mettant à jour la base de données des IoC pour chaque trace considérée comme un EoI.

La procédure $p_{hunting}$ est formalisée de la façon suivante.

Soit, $\mathbf{G}_D = (\emptyset, \emptyset)$.

```
procedure p_hunting(X, G_D, IoC) :
    for all x ∈ X:
        (R_x, O_x) = EoI_{R,IoC}(x)
        if R_x ≠ ∅ or O_x ≠ ∅:
            G_D = G_D ⊕ G(x)
            IoC = IoC ∪ generate_IoC((R_x, O_x), x, IoC)
    return (G_D, IoC)
```

Au cours d'une opération de *Threat Hunting*, le défenseur construit à partir de $\mathbf{G}_D$ un graphe restreint, qui met en avant les objets partagés entre plusieurs composants. Cela signifie que ces objets, qui ont différents rôles et qui sont considérés comme des EoI, ont été observés dans les traces d'au moins deux composants. Ce graphe permet au défenseur d'orienter ses investigations pendant la chasse de l'attaquant. En effet, le défenseur s'alimente de ces objets pour conduire des investigations grâce au SIEM. Il recherche si des occurrences de ces objets sont présentes dans les traces observées sur d'autres composants. Cela signifierait que l'attaquant applique les mêmes procédures sur plusieurs composants. L'analyse des traces de l'expérimentation PWNJUTSU, présentée dans le Chapitre 4, laisse transparaître une certaine routine lors de l'exécution des procédures par l'attaquant. Cette faiblesse opérationnelle de la part de l'attaquant est un avantage non négligeable pour le défenseur qui le prend en chasse.

3.5 Expérimentation du modèle

L'intégration de notre approche dans un environnement de production permet à l'attaquant et au défenseur, chacun de leur côté, de prendre conscience des traces laissées sur le SI. L'attaquant peut ainsi utiliser ces informations pour améliorer la furtivité de ses procédures. De son côté, le défenseur peut les utiliser pour améliorer son processus de *Threat Hunting*. En pratique, le déploiement d'une implémentation de ce modèle dans une infrastructure existante nécessite la présence :

- du côté de l'attaquant :
  - d'un système de journalisation des procédures exécutées sur le SI. Le Listing 1 est un exemple d'une entrée de ce journal ;
  - d'un annuaire qui correspond à la connaissance actuelle de l'attaquant à propos des composants du SI.
- du côté du défenseur :
  - de sondes installées sur les composants à superviser. Le Listing 2 présente un exemple de configuration de ces sondes ;
  - d'un annuaire qui correspond à la connaissance actuelle du défenseur à propos des composants du SI ;
  - d'un indexeur (*i.e.*, SIEM) enrichi par un ensemble de règles de détection, telles que présentées dans le Listing 4 ;
  - d'une base de données d'IoC, comme présentée dans la Table 3.3 ;
  - d'un analyste pour les tâches qui ne sont pas automatisables, telles que la génération d'IoC.

Dans ce chapitre, nous prenons le rôle d'un acteur omniscient (*i.e.*, une *White Team*), ce qui nous permet de répondre aux questions suivantes.

Comment mesurer la qualité de l'architecture défensive ?

La comparaison des graphes $\mathbf{G}_A$ et $\mathbf{G}_D$ nous permet de calculer le taux de couverture des objets issus de la perspective de l'attaquant dans le graphe du défenseur. Ainsi, il est possible d'apprécier la pertinence de la chaîne de détection.

Comment réduire le graphe du défenseur ?

Si trop d'objets sont représentés dans le graphe $\mathbf{G}_D$, cela peut le rendre inutilisable. Nous cherchons par conséquent des moyens de réduire le nombre d'objets tout en maintenant un taux de couverture suffisant pour permettre à une équipe de *Threat Hunting* d'y récolter des éventuels IoC.

Pour cela, nous avons exploité les traces d'un scénario d'attaque réaliste sur une infrastructure représentative de celles que nous pouvons trouver dans les entreprises modernes.

3.5.1 Scénario d'attaque

Nous avons choisi d'expérimenter notre modèle sur un scénario d'attaque indépendant et représentatif. Nous nous sommes référés au projet de cybersécurité Mordor [41], initié et maintenu par Roberto Rodriguez.

Le projet Mordor fournit un jeu de données d'événements de sécurité générés suite à l'exécution de techniques adverses. En 2020, il a été mis à jour avec un nouveau jeu de données appelé APT29. Le jeu de données contient, lui, des journaux d'événements générés suite à l'exécution des procédures d'attaques décrites dans un scénario conçu par le MITRE dans le cadre de leur programme ATT&CK Evaluations [54]. Le scénario émule deux parties d'une campagne d'attaque opérée par un groupe d'acteurs dénommé APT29. L'attaque a pour objectif de collecter et d'exfiltrer des données sensibles du SI. La première partie est décrite par les auteurs comme une collecte et exfiltration de données digne d'un cambriolage [4], suite à une primo-infection grâce à une campagne de hameçonnage. Puis l'attaquant a déposé un outil qui lui a permis d'explorer plus en profondeur le réseau

4. *smash and grab*

compromis. La seconde partie est une intrusion méthodique et parfaitement ciblée, où la compromission du réseau de la victime se fait lentement et en restant sous le seuil de détection des dispositifs de sécurité déployés. Une partie de ces procédures d'attaque est décrite à haut niveau dans la Table 3.4. La liste complète est disponible sur le site de MITRE Engenuity [61]. Le Listing 5 présente les grandes lignes de notre retranscription de ces procédures, étape par étape.

Deux vidéos ont également été enregistrées par l'auteur du projet Mordor. Cela permet de mieux percevoir les actions de l'attaquant pour cette expérimentation, mais également de constater les objets effectivement manipulés lors de l'exécution de ses procédures d'attaque. Le Listing 5 est un extrait de la retranscription de toutes les procédures d'attaque exécutées au cours de la campagne. Cela nous permet de construire le graphe $\mathbf{G}_A$. Ainsi, conformément au modèle présenté en Section 3.3, pour chaque procédure un nœud central correspondant au composant est créé. Il est ensuite connecté par des liens à chacun des objets que l'attaquant manipule et qu'il est conscient d'exposer pendant l'exécution de la procédure d'attaque. Chacun de ces liens est annoté par le rôle de l'objet dans la procédure et par un identifiant. Lorsque l'exécution d'une procédure implique une autre procédure, un objet désignant le composant tiers est créé. L'objet est ainsi relié au nœud du composant par un lien annoté "Ref".

La Figure 3.3 représente ce graphe $\mathbf{G}_A$, caractérisé par :

— 4 composants ;
— 180 objets uniques ; et
— 359 nœuds.

5. ©2018 - 2022 The MITRE Corporation. This work is reproduced and distributed with the permission of The MITRE Corporation.

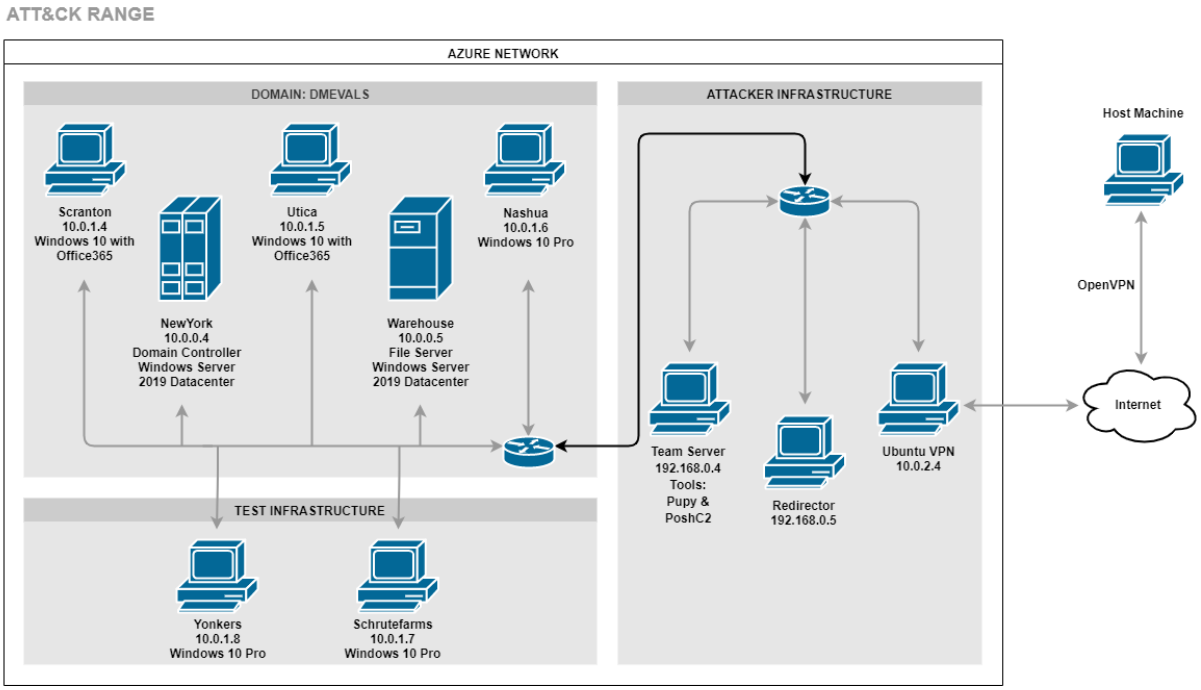

FIGURE 3.6 – Environnement d'évaluation APT29 (source : MITRE [5]).

1. The scenario begins with an initial breach, where a legitimate user clicks (T1204) an executable payload (screensaver executable) masquerading as a benign word document (T1036). Once executed, the payload creates a C2 connection over port 1234 (T1065) using the RC4 cryptographic cipher . The attacker then uses the active C2 connection to spawn interactive cmd.exe (T1059) and powershell.exe (T1086) shells.
2. The attacker runs a one-liner command to search for filesystem for document and media files (T1083, T1119), collecting (T1005) and compressing (T1002) content into a single file (T1074). The file is then exfiltrated over the existing C2 connection (T1041).
...
6. The attacker accesses credentials stored in a local web browser (T1081, T1003) using a tool renamed to masquerade as a legitimate utility (T1036). The attacker then harvests private keys (T1145) and password hashes (T1003).
...
8. The attacker uses Lightweight Directory Access Protocol (LDAP) queries to enumerate other hosts in the domain (T1018) before creating a remote PowerShell session to a secondary victim (T1028). Through this connection, the attacker enumerates running processes (T1057). Next, the attacker uploads a new UPX-packed payload (T1045) to the secondary victim. This new payload is executed on the secondary victim via the PSExec utility (T1077, T1035) using the previously stolen credentials (T1078).

TABLE 3.4 – Extrait de la description haut-niveau des procédures de la campagne d'attaque simulée d'APT29.

```
[
    {"name": "1.A", "description": "initial breach, click on pupy payload"},
    {"name": "1.B", "description": "get a (power)shell"},
    {"name": "2.A", "description": "collect document and media files and zip"},
    {"name": "2.B", "description": "download archive"},
    {"name": "3.A", "description": "upload meterpreter payload"},
    {"name": "3.B", "description": "privileges escalation via uac bypass"},
    {"name": "3.C", "description": "clean artifacts of priv esc"},
    {"name": "4.A", "description": "upload tools"},
    {"name": "4.B", "description": "kill and clean initial stager"},
    {"name": "4.C", "description": "Internal reconnaissance"},
    {"name": "5.A", "description": "Persist with a service"},
    {"name": "5.B", "description": "Persist with a startup file"},
    {"name": "6.A", "description": "Get Chrome creds"},
    {"name": "6.B", "description": "Get harvests private keys"},
    {"name": "6.C", "description": "Get harvests password hashes"},
    {"name": "7.A", "description": "Collect screenshots, clipboard and
↪ keystrokes"},
    {"name": "7.B", "description": "Exfiltrate files"},
    {"name": "8.A", "description": "Domain discovery via LDAP and remote host
↪ discovery"},
    {"name": "8.B", "description": "Get payload from secondary computer"},
    {"name": "8.C", "description": "Execute payload on secondary computer"},
    {"name": "9.A", "description": "upload tools"},
    {"name": "9.B", "description": "upload tools, collect and exfiltrate data"},
    {"name": "9.C", "description": "clean"},
    {"name": "11.A", "description": "initial breach #2"},
    {"name": "12.A", "description": "edit timestomp"},
    {"name": "12.B", "description": "detect av"},
    {"name": "12.C", "description": "list software"},
    {"name": "13.x", "description": "get local computer name"},
    {"name": "14.A", "description": "bypass uac"},
    {"name": "14.B", "description": "collect creds in memory (mimikatz)"},
    {"name": "15.A", "description": "create persistence"},
    {"name": "16.A", "description": "enum domain controller"},
    {"name": "16.B", "description": "enum sid"},
    {"name": "16.C", "description": "lateralize with winrm"},
    {"name": "16.D", "description": "lateralize with winrm"},
    {"name": "17.A", "description": "get email"},
    {"name": "17.B", "description": "collect file of interest"},
    {"name": "17.C", "description": "compress file of interest"},
    {"name": "18.A", "description": "exfiltrate to onedrive"},
    {"name": "19.A", "description": "clean files"},
    {"name": "19.B", "description": "clean files"},
    {"name": "19.C", "description": "clean files"},
    {"name": "20.B", "description": "generate kerberos ticket and establish winrm
↪ session"}
]
```

Listing 5: Extrait de la retranscription, étape par étape, des procédures de la campagne d'attaque simulée d'APT29.

3.5.2 Infrastructure ciblée et architecture défensive

L'infrastructure de la victime, qui a permis l'exécution de la campagne d'attaque et la création du jeu de données Mordor APT29, a été scrupuleusement reproduite par les auteurs du projet à partir de l'environnement décrit par le MITRE dans le cadre de leurs évaluations. Cet environnement est présenté sur la Figure 3.6.

Le SI est composé de :

— trois stations de travail ;

— un serveur de fichier ;

— un contrôleur de domaine.

Tous ces composants utilisent le système d'exploitation Microsoft Windows. Les systèmes sont supervisés par des sondes Sysmon, qui fournissent des informations détaillées sur les processus, les connexions réseau et les fichiers manipulés. Sysmon produit les traces qui seront utilisées ensuite par le défenseur pour chasser l'attaquant. Le jeu de données complet représente un volume total de 783367 traces, correspondant à deux jours d'observation (durée de l'attaque simulée).

Nous avons injecté les traces du projet Mordor APT29 dans un moteur Splunk [6] pour indexation. Dans ce jeu de données de traces brutes, nous avons donc besoin de découvrir des EoI afin de pouvoir chasser l'attaquant efficacement. Dans cette expérimentation, le jeu de règles de détection était composé de 565 règles communément utilisées et issues du projet public de détection Sigma [7]. Sigma est un projet communautaire ouvert qui a pour ambition de capitaliser les règles de détection dans un même formalisme, qui peuvent ensuite permettre de générer des requêtes pour plusieurs dizaines de SIEM ; ou encore être directement intégrées dans des plateformes d'analyse de *malware*, telles que VirusTotal [8].

Au début de notre expérimentation, notre base de données **IoC** était vide. La première passe a permis de découvrir 6100 EoI, grâce au déclenchement de 22 règles de détection, parmi les 565 activées. Nous avons donc pu construire le graphe du défenseur $\mathbf{G}_D$ avec ces **EoI**. Le graphe généré comporte :

— 4 composants et

— 1758 objets uniques.

La Table 3.5 présente les différentes caractéristiques du graphe généré. Les jeux de données utilisés, le code et les graphes complets ont été mis en ligne [9].

6. https://www.splunk.com
7. https://github.com/SigmaHQ/sigma
8. https://developers.virustotal.com/reference/sigma-rule-object
9. https://gitlab.inria.fr/cidre-public/from-ttp-to-ioc-dataset

items	valeurs
Composants	4
Objets	1758
Relations (liens entre les objets et les composants)	15788
EoI injectés	6100
Objets connectés à 4 composants	18
Objets connectés à 3 composants	48
Objets connectés à 2 composants	398
Objets connectés à 1 composant	1294
Taux de couverture des objets du point de vue de l'attaquant	27,78 %
Règles uniques ayant produit des EoI	22

TABLE 3.5 – Point de vue du défenseur : spécification du graphe $\mathbf{G}_D$.

Au travers de cette expérimentation, nous avons évalué la pertinence de deux stratégies de désactivation de règles afin de déterminer l'approche qui serait la plus efficace pour réduire le nombre de faux positifs tout en préservant un graphe $\mathbf{G}_D$ exploitable. Les résultats de cette évaluation sont discutés dans la Section 3.6.

3.6 Résultats

Comme nous l'avons précédemment évoqué, certains objets, dont l'attaquant ignore l'existence, ou qu'il n'est simplement pas conscient d'exposer, n'apparaissent pas dans le graphe $\mathbf{G}_A$. Suite à notre expérimentation, nous observons que les composants présents dans le graphe de l'attaquant sont également présents dans le graphe du défenseur. Cela n'est pas surprenant, dans la mesure où chaque composant était supervisé et que toutes les règles de détection étaient activées. Cependant, le nombre d'objets présents dans le graphe de l'attaquant (180) est considérablement inférieur au nombre d'objets présents dans le graphe du défenseur (1758). Cela suggère que le graphe du défenseur contient beaucoup de faux positifs. Dans ce contexte, les faux positifs sont généralement des objets légitimes ou natifs du système et dont le comportement déclenche des règles de détection trop laxistes ou simplement inappropriées. Cela peut également être dû à des configurations de sondes trop verbeuses. En d'autres termes, un faux positif est un objet qui ne doit jamais devenir un IoC, car cela n'aurait pas de sens de le rechercher dans un périmètre plus large ou parce qu'il n'est pas relatif à une activité malveillante sur un des composants du SI.

Afin de réduire le nombre de faux positifs, l'intervention d'un agent cognitif est souvent nécessaire, car il permet une qualification des traces et, par extension, des règles associées. Cependant, il pourrait être possible de marquer automatiquement une partie de ces traces qui correspondent à des comportements légitimes. Une attention toute particulière doit être portée sur les techniques faisant appel au paradigme *Living-off-the-Land* [62], qui consiste à faire appel à des outils légitimes des systèmes afin de satisfaire des intentions malveillantes.

Dans la suite de cette section, nous allons proposer une méthode de réduction du nombre de faux positifs.

3.6.1 Mesurer la qualité de l'architecture défensive

Considérant un acteur omniscient, capable d'observer tous les acteurs de la campagne d'attaque, le modèle permet de comparer les données présentes des deux côtés. Le nombre d'objets existant dans $\mathbf{G}_A$ et qui sont également présents dans $\mathbf{G}_D$ permet d'estimer l'efficacité de la chaîne de détection du défenseur. À la suite de notre expérimentation, nous avons calculé que $27,78\%$ des objets de l'attaquant étaient effectivement considérés par le défenseur comme des EoI. Alors que ce pourcentage est suffisamment élevé pour que le défenseur puisse envisager une opération de *Threat Hunting* pour chasser l'attaquant, il est important de tenir compte du fait qu'un nombre trop important d'objets présents dans le graphe risquerait de perturber le défenseur dans sa compréhension de la campagne d'attaque et certains IoC pourrait ne pas être considérés. Ces IoC sont des objets spécifiques présentant des caractéristiques discriminantes, tels que présentés dans la Table 3.3. Ainsi, il est particulièrement intéressant de les rechercher dans un périmètre plus large que celui de leur découverte. Enfin, un agent cognitif devra les qualifier avant de les intégrer dans la base de données **IoC**.

3.6.2 Réduire le graphe du défenseur pour dévoiler l'attaquant

Le *Threat Hunting* est une discipline cyclique où le défenseur :

— identifie et qualifie de nouveaux IoC ;

— modifie les règles de détection pour trouver ces IoC dans un périmètre plus large ;

— analyse les traces collectées ; et

— extrait de nouveaux IoC.

Il convient de préciser que dans ce processus, tous les objets ne deviennent pas systématiquement des IoC. Par exemple, si un attaquant utilise une procédure d'attaque `psexec` pour effectuer des mouvements latéraux entre plusieurs machines, et si l'équipe d'administration du SI utilise ce même outil pour effectuer des tâches légitimes, alors il ne faudra pas considérer le fichier binaire `psexec.exe` (ou l'une de ces empreintes cryptographiques) comme un IoC, au risque de provoquer l'apparition de faux positifs dans le graphe du défenseur. Une parade possible pourrait être d'écrire une règle de détection, qui permettrait d'exclure l'activité légitime en spécifiant le contexte d'exécution de ces outils ou de ces tâches administratives. Par exemple, en précisant les adresses IP et les comptes utilisateurs légitimement impliqués. La désactivation des règles trop verbeuses peut être faite en post-traitement afin de "nettoyer" $\mathbf{G}_D$ et de le rendre plus exploitable. Nous avons évalué deux stratégies de désactivation de règles, nommées *Top-objects* et *Top-events*.

La stratégie *Top-objects*, consiste à exécuter une séquence de tours de désactivation des règles de détection qui ont créé le plus grand nombre de nouveaux objets uniques dans le graphe du défenseur $\mathbf{G}_D$. De la même manière, la stratégie *Top-events*, consiste en la désactivation des règles de détection ayant produit le plus d'EoI.

Étant donné un graphe du défenseur $\mathbf{G}_D = (V_D, \rightarrow_D)$, la désactivation d'un jeu de règles R_D produit un nouveau graphe du défenseur, noté Update($\mathbf{G}_D, R_D$) et défini par la suppression dans $\mathbf{G}_D$ de tous les liens :

$$\mathbf{c} \xrightarrow{\mathbb{r}, R_{(\mathbf{o},\mathbb{r})}, \text{is\_ioc}} \mathbf{o}$$

tels que $R_{(\mathbf{o},\mathbb{r})} \setminus R_D = \emptyset$ et is_ioc = false et la suppression, s'il existe, du lien ref à partir de l'objet **o** et jusqu'au composant qui termine ce lien. Avec, comme valeurs initiales :

— le graphe de l'attaquant $\mathbf{G}_A = (V_A, \rightarrow_A)$ contenant les objets dans $O_A = V_A \cap \mathbf{O}$;

— le graphe du défenseur $\mathbf{G}_D = (V_D, \rightarrow_D)$ contenant les objets dans $O_D = V_D \cap \mathbf{O}$;

— un ensemble vide R_D de règles désactivées ; et

— le taux de couverture coverage = $v > 0$.

Le calcul du taux de couverture selon la stratégie *Top-events* s'implémente grâce à l'algorithme suivant :

```
while coverage > 0:
    r = most_used(→_D)
    R_D = R_D ∪ {r}
    Update(G_D, R_D)
    coverage = (Card(O_A ∩ O_D) / Card(O_A)) × 100
return R_D
```

Où,

— most_used($\rightarrow_D$) est une fonction qui retourne la règle de détection qui apparaît le plus de fois dans $\mathbf{G}_D$;

— Update($\mathbf{G}_D, R_D$) est une fonction qui met à jour la visualisation du graphe $\mathbf{G}_D$ en excluant les événements résultant des règles R_D.

L'implémentation de R_D en tant que liste ordonnée permet de conserver l'ordre de désactivation des règles de détection et donc d'être en mesure de rétablir le graphe dans un état antérieur.

La Figure 3.7 présente deux courbes 3D, qui correspondent respectivement aux stratégies de désactivation *Top-objects* et *Top-events*. Nous pouvons observer que la stratégie *Top-objects* semble être la plus efficace puisqu'elle permet de maintenir un haut taux de couverture tout en réduisant considérablement le nombre d'objets. Ainsi, alors qu'il ne reste plus que 63 objets sur le graphe et que seulement 4 règles ont été désactivées, le taux de couverture est de $24,4\%$.

3.6.3 Discussion sur la stratégie de désactivation de règles

En appliquant une stratégie de désactivation de règles de détection, le défenseur peut considérablement réduire le nombre d'objets présents dans le graphe et donc ceux qui pourraient nécessiter des investigations ; tout en maintenant un taux de couverture suffisamment élevé. Cependant, il est important de souligner que contrairement à des systèmes de sécurité qui ont pour objectifs de contrer une menace (antivirus, *Endpoint Detection and Response* (EDR)), les infrastructures défensives, telles que celles décrites dans ce chapitre, ont pour objectif de collecter un maximum de données pour profiter à la CTI. Ces données servent ensuite à chasser plus efficacement l'attaquant dans le SI compromis.

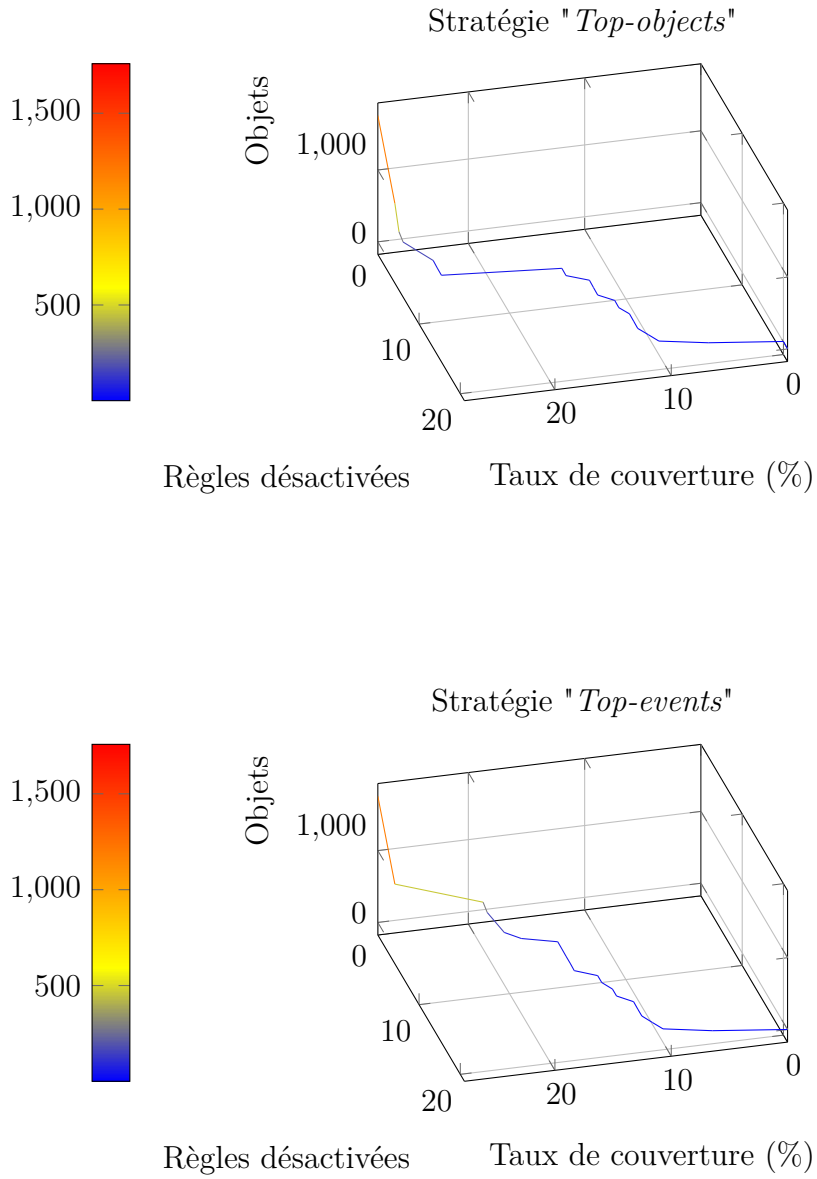

FIGURE 3.7 – Evolutions des taux de couvertures et du nombre d'objets présents dans le graphe $\mathbf{G}_D$ selon le nombre de règles désactivées. L'échelle de couleurs à gauche indique le nombre d'objets présents dans le graphe. L'objectif est conserver un taux de couverture élevé (vers la gauche) en désactivant un minimum de règles (vers le fond).

Ainsi, une détection exhaustive n'est pas nécessaire puisque l'objectif n'est pas de bloquer les attaques, mais de collecter suffisamment d'IoC pour traquer l'attaquant dans le SI. Enfin, l'approche de mesure du taux de couverture décrite dans la section précédente n'est évidemment possible que dans un contexte d'exercice *Red versus Blue*. Malgré cela, la désactivation des règles de détection trop verbeuses peut être faite en post-traitement sans observer directement l'évolution du taux de couverture, mais basiquement afin de nettoyer le graphe généré par le défenseur et le rendre plus exploitable.

3.7 Conclusion du troisième chapitre

Le *Threat Hunting* est une phase fondamentale dans une opération de réponse à incident, qui permet d'identifier les composants d'un SI qui ont été compromis par un attaquant. Dans ce processus, le défenseur a pour ambition de mettre en lumière l'espace de propagation de l'attaquant au sein du SI en vue de mieux préparer la phase d'éradication de la menace.

Dans ce chapitre, nous avons proposé un modèle pour analyser la propagation de l'attaquant dans le SI et la connaissance du défenseur à propos de cette propagation. Les différentes étapes du processus de *Threat Hunting* ont été formalisées ici. Notre approche permet également d'améliorer la base d'IoC du défenseur avec de nouveaux marqueurs, qui pourront par la suite être utilisés pour adopter une posture proactive de détection de l'adversaire dès l'exécution de ces premières procédures dans le SI. De plus, notre modèle et l'expérimentation conduite mettent en évidence l'existence de faux positifs, en particulier à cause de règles de détection trop laxistes. Cette fonctionnalité est précieuse pour le défenseur, car elle lui permet de gagner en efficacité et d'améliorer ses outils de détection. En particulier, parce que l'analyse du graphe généré par le défenseur favorise l'identification de motifs et la découverte de corrélation entre les objets au sein d'un même SI. Au cours de cette étude, nous avons mieux compris l'origine des objets qui deviennent ensuite des IoC : ils sont principalement issus des paramètres d'instanciation des procédures de l'attaquant. Nous avons également souligné le fait que certains objets, alors qu'ils n'ont un sens que dans le contexte du SI où ils ont été découverts (par exemple, une adresse IP privée ou un nom d'utilisateur du domaine), peuvent être utilisés comme des IoC dits "locaux" pour découvrir d'autres points d'implantation de l'attaquant. Enfin, notre modèle laisse apparaître la dualité entre l'attaquant et le défenseur (et, par extension, de la victime), où chacun infère la connaissance de l'autre.

Pour chacun d'entre eux, la connaissance la plus précieuse étant la découverte des procédures adverses et par extension de ses TTP, car cela permettrait pour le défenseur d'être capable de détecter chaque action de l'attaquant ; et pour l'attaquant de passer sous le radar des dispositifs de supervision déployés par le défenseur.

Ces travaux ont été publiés dans

Aimad Berady, Mathieu Jaume, Valérie Viet Triem Tong et Gilles Guette :
From TTP to IoC: Advanced Persistent Graphs for Threat Hunting.
IEEE Transactions on Network and Service Management (TNSM),
Special Issue on Latest Developments for Security Management of Networks and Services,
juin 2021.
DOI : 10.1109/TNSM.2021.3056999, HAL Id : hal-03131262

Chapitre 4

PWNJUTSU : RETRACER LES CAMPAGNES D'ATTAQUES

Ce chapitre présente une approche complémentaire permettant de collecter des données pertinentes à des fins d'études des menaces sophistiquées, ainsi qu'une formalisation de l'état de connaissance de l'attaquant au cours de sa progression dans le SI de la victime. Ces travaux ont été publiés dans "*PWNJUTSU : A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns*" [63].

4.1 Introduction

L'identification de motifs dans les modes opératoires des attaquants est un prérequis essentiel à l'étude des menaces sophistiquées. De précédentes études ont souffert du manque de jeux de données, qui soient à la fois précis, pertinents et représentatifs des menaces actuelles. Les journaux d'événements du système et le trafic réseau capturés pendant des attaques sur des entreprises réelles seraient la meilleure source d'information pour construire des jeux de données de cette qualité. Malheureusement, pour des raisons évidentes de confidentialité, de réputation et de sécurité, ce genre de données est très rarement rendu public.

Dans un premier temps, nous présentons un modèle formel décrivant l'état des connaissances de l'attaquant, puis les évolutions possibles de cet état. Cela nous permet d'exprimer formellement la progression tactique d'un attaquant pendant sa phase de propagation dans le SI de sa victime. Cette progression est exprimée selon l'état de l'attaquant, appelé **muSE**. Il spécifie son espace de propagation, les secrets qu'il a collectés et la connaissance de l'environnement qu'il a acquise. Ce modèle repose sur une sémantique opérationnelle des techniques d'attaques. Cette sémantique définit formellement une relation de transition entre les différents états de l'attaquant.

Par conséquent, le modèle peut être utilisé pour décrire un scénario d'attaque dans son entièreté. C'est donc cette formalisation qui a rendu possible la description précise de l'expérimentation PWNJUTSU, présentée dans ce chapitre.

Dans cette expérimentation, 22 *Red Teamers* ont attaqué une infrastructure, volontairement laissée vulnérable, afin de compromettre les machines qui la composent et d'y dérober des secrets. Chaque *Red Teamer* opérait sur une instance dédiée de l'infrastructure. Des sondes ont capturé les activités des systèmes et le trafic des réseaux des victimes sur chacune des instances. Cela a donné lieu à la publication du jeu de données PWNJUTSU, qui a pour ambition de combler le fossé souligné par Kovačević *et al* qui concluent leur enquête scientifique [64] en soulignant la pertinence de la notion de connaissance dans

l'étude des attaquants tout en regrettant l'absence de jeux de données.

Ce chapitre présente nos travaux visant à compenser le manque de données brutes qui peuvent être utilisées pour étudier les TTP des groupes responsables de campagnes d'attaques sophistiquées, ainsi que les deux principales contributions de ces travaux.

1. Premièrement, nous présentons un cadre formel qui nous permet de décrire précisément une campagne d'attaque. Cela consiste en (1) la création d'un modèle décrivant le SI au travers de la description de ce qu'il héberge (*i.e.*, services, utilisateurs et *assets*) et sa topologie ; (2) un modèle décrivant l'état, la perception et la connaissance de l'attaquant ; et (3) la sémantique opérationnelle des techniques d'attaques qu'un attaquant peut exploiter pour progresser vers ses objectifs. La Figure 4.1 présente les principaux éléments impliqués dans une attaque et la manière dont ils sont liés les uns aux autres dans notre cadre formel.
2. Cette représentation formelle a permis de décrire les attaques subies sur notre infrastructure au cours de l'expérimentation. Les traces produites au cours de cette expérimentation représentent 16 millions d'événements et 172Go de trafic réseau. Toutes ces données ont été agrégées dans le **jeu de données PWNJUTSU** que nous avons rendu public [1].

1. https://pwnjutsu.irisa.fr

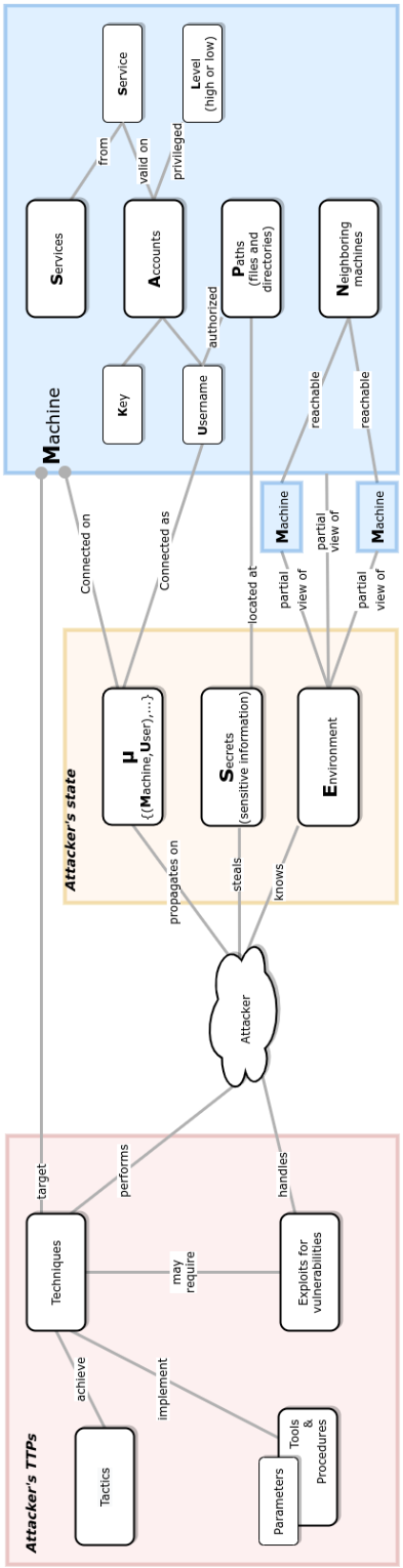

FIGURE 4.1 – Vue d'ensemble du modèle.

Dans ce chapitre, nous présentons en Section 4.2 le modèle générique permettant de décrire une infrastructure et notamment celle de l'expérimentation PWNJUTSU. La notion de progression de l'attaquant et l'évolution de sa perception du SI, ainsi que le panel impliqué dans l'expérimentation sont présentés en Section 4.3. La description précise de cette progression de l'attaquant grâce à notre sémantique opérationnelle est présentée dans la Section 4.4. À titre d'exemple, une instanciation du modèle avec un participant de l'expérimentation, extrait du jeu de données, est détaillée en Section 4.5. Enfin, les enregistrements issus du jeu de données PWNJUTSU et les résultats de l'évaluation subjective conduite sur les participants sont présentés dans la Section 4.6.

4.2 Architecture de l'expérimentation

Dans cette section, nous présentons l'infrastructure déployée sur chaque instance de l'expérimentation de PWNJUTSU. Nous commençons par fournir une description formelle générique de l'infrastructure. Sa généricité et sa flexibilité nous permettent également de décrire d'autres infrastructures dans lesquelles des attaquants peuvent opérer des actions en quête de l'atteinte de leurs objectifs. Le but de cette formalisation est de mettre en avant les principaux éléments nécessaires à la description d'une architecture, qui est susceptible d'être la cible d'une campagne d'attaque. Dans le reste de cette section, nous décrivons le déploiement de cette architecture, dans le contexte de l'expérimentation PWNJUTSU.

4.2.1 Infrastructure

Nous avons choisi de représenter l'infrastructure ciblée (*i.e.*, le SI) au travers de ses machines, de ses connectivités réseau entre les machines, de ses comptes utilisateurs, de ses fichiers ou répertoires et de ses services. Chacun de ces points peut être exploité par un attaquant pour progresser vers ses objectifs. Le réseau est représenté par un graphe de machines, que nous décrivons de la façon suivante.

Une machine est un tuple $\mathbf{m} = (\mathbb{S}_{\mathbf{m}}, \mathbb{P}_{\mathbf{m}}, \mathbb{A}_{\mathbf{m}}, \mathbb{N}_{\mathbf{m}})$ où :

— $\mathbb{S}_{\mathbf{m}}$ est l'ensemble des services fournis par la machine $\mathbf{m}$. Nous écrivons :

$$\texttt{svc\_name}(\texttt{logical\_identifier} : \texttt{value})$$

le service $\mathtt{s} \in \mathbb{S}_{\mathbf{m}}$ fourni par la machine $\mathbf{m}$.

L'identifiant logique d'un service peut être :
- un `port`, d'un point de vue réseau ;
- un `processus`, désigné par son nom ou son identifiant, d'un point de vue système ; ou encore
- un `fichier`, désigné par son chemin absolu, son nom de fichier ou encore son *hash*, d'un point de vue système de fichiers.

Par exemple, `apache`(`port` : `443`) correspond au service réseau `apache` écoutant sur le port TCP 443.

— $\mathbb{P}_{\mathbf{m}}$ est l'ensemble de toutes les paires $(\mathsf{p}, \mathcal{U})$ tel que p est un chemin absolu pour un fichier ou un répertoire sur le système de fichiers de la machine $\mathbf{m}$. Le contenu de ce fichier est noté $*\mathsf{p}$, à la façon des pointeurs du langage de programmation C. L'ensemble $\mathcal{U}$ contient tous les utilisateurs $\mathbf{u}$ d'une machine autorisés à accéder à ce chemin sur le système de fichiers. Parmi ces fichiers, certains contiennent des données sensibles, qui sont convoitées par l'attaquant. Dans notre modèle, nous désignons ces chemins vers des fichiers sensibles par $(^{\odot}\mathsf{p})$. On note $\mathsf{p}.x$ le chemin p suivi du chemin x (*i.e.*, x est dans l'arborescence de p).

— $\mathbb{A}_{\mathbf{m}}$ est un ensemble de comptes utilisateurs, représenté par un ensemble de tuples $(\mathbf{u}, \mathsf{s}, \mathsf{k}, \ell)$. Un compte pour l'utilisateur $\mathbf{u}$ est protégé par une clé k (*i.e.*, mot de passe, paire de clés, jeton...) et jouissant d'un niveau de privilège ℓ (`low`, `low*` [2] ou `high`) pour un service spécifique s de la machine $\mathbf{m}$. Si le service s ne requiert par d'authentification, alors $\mathsf{k} = \texttt{None}$. Les utilisateurs, les services et les niveaux de privilèges associés mis en place pour l'expérimentation PWNJUTSU sont décrits dans la Table 4.1. L'identifiant logique d'un compte est un nom d'utilisateur.

— $\mathbb{N}_{\mathbf{m}}$ est l'ensemble des machines du voisinage réseau, qui peuvent être jointes depuis la machine $\mathbf{m}$. Ainsi, les nœuds du graphe du réseau sont des machines et il y a un lien tracé d'une machine $\mathbf{m}_1$ vers une machine $\mathbf{m}_2$ ssi $\mathbf{m}_2 \in \mathbb{N}_{\mathbf{m}_1}$. L'identifiant logique d'une machine du voisinage peut être un nom de machine, une adresse IP ou une adresse MAC.

2. `low*` correspond aux utilisateurs étant autorisés à élever légitimement leurs privilèges (par exemple, les *sudoers*)

$\mathbb{A}_{n12-gateway}$			
u	s	k	ℓ
anonymous	network	None	low

$\mathbb{A}_{n12-vm1}$			
u	s	k	ℓ
root	continuum(port:8080)	[redacted]	high
vagrant	ssh(port:22)	[redacted]	low*
han_solo	ssh(port:22)	[redacted]	low*
lando_calrissian	ssh(port:22)	[redacted]	low
boba_fett	ssh(port:22)	[redacted]	low
www-data	apache(port:80)	None	low
mysql	mysql(port:3306)	None	low
boba_fett	unrealirc(port:6697)	[redacted]	low
chewbacca	rails(port:3500)	[redacted]	low

$\mathbb{A}_{n12-vm2}$			
u	s	k	ℓ
tomcat	tomcat(port:8080)	[redacted]	high
NT AUTHORITY\LOCAL SERVICE	webdav(port:80)	None	low*
NT AUTHORITY\SYSTEM	elastic(port:9200)	None	high
anonymous	smb(port:445)	None	low
boba_fett	smb(port:445)	[redacted]	low
lando_calrissian	smb(port:445)	[redacted]	low
Administrator	smb(port:445)	[redacted]	high
boba_fett	wmi/dcom(port:135)	[redacted]	low
lando_calrissian	wmi/dcom(port:135)	[redacted]	low
Administrator	wmi/dcom(port:135)	[redacted]	high
boba_fett	netbios(port:139)	[redacted]	low
lando_calrissian	netbios(port:139)	[redacted]	low
Administrator	netbios(port:139)	[redacted]	high
boba_fett	ssh(port:22)	[redacted]	low
lando_calrissian	ssh(port:22)	[redacted]	low
Administrator	ssh(port:22)	[redacted]	high
Guest	rdp(port:3389)	None	low
boba_fett	rdp(port:3389)	[redacted]	low
lando_calrissian	rdp(port:3389)	[redacted]	low
Administrator	rdp(port:3389)	[redacted]	high

$\mathbb{A}_{n12-vm3}$			
u	s	k	ℓ
ftp	vsftpd(port:21)	[redacted]	low*
root	ssh(port:22)	[redacted]	high
www-data	apache/drupal(port:80)	None	low
captain	vnc(port:5900)	[redacted]	low

TABLE 4.1 – Services, comptes utilisateur, clés et privilèges ($\mathbb{A}_{\mathbf{m}}$) sur les machines de l'expérimentation PWNJUTSU

P12 VPN address - 172.16.128.112			
n12-gateway - 172.16.1.12, 10.12.1.254			
$\mathbb{S}_{n12-gateway}$ (Services)	$\mathbb{P}_{n12-gateway}$ (Paths)	$\mathbb{A}_{n12-gateway}$ (Accounts)	$\mathbb{N}_{n12-gateway}$ (Neighboring)
$\emptyset$	$\emptyset$	Refer to Table 4.1	n12-vm1
n12-vm1 - 10.12.1.1 - Linux Ubuntu 14.04			
$\mathbb{S}_{n12-vm1}$ (Services)	$\mathbb{P}_{n12-vm1}$ (Paths)	$\mathbb{A}_{n12-vm1}$ (Accounts)	$\mathbb{N}_{n12-vm1}$ (Neighboring)
continuum(port:8080) ssh(port:22) apache(port:80) unrealircd(port:6697) rails(port:3500) mysql(port:3306)	(/home/boba_fett/flag.txt$^{\odot}$, {root, boba_fett}) (/root/flag.txt$^{\odot}$, {root}) (/home/han_solo/flag.txt$^{\odot}$, {root, han_solo}) (/home/lando_calrissian/flag.txt$^{\odot}$, {root, lando_calrissian}) (/home/vagrant/flag.txt$^{\odot}$, {root, vagrant}) (/opt/unrealircd/Unreal3.2/flag.txt$^{\odot}$, {root, boba_fett}) (/opt/apache_continuum/apache − continuum − 1.4.2/flag.txt$^{\odot}$, {root}) (/opt/readme_app/flag.txt$^{\odot}$, {root, chewbacca}) (/etc/shadow, {root})	Refer to Table 4.1	n12-vm2
n12-vm2 - 10.12.1.2 - Windows 2008			
$\mathbb{S}_{n12-vm2}$ (Services)	$\mathbb{P}_{n12-vm2}$ (Paths)	$\mathbb{A}_{n12-vm2}$ (Accounts)	$\mathbb{N}_{n12-vm2}$ (Neighboring)
tomcat(port:8080) webdav(port:80) elasticsearch(port:9200) ssh(port:22) wmi/dcom(port:135) netbios(port:139) smb(port:445) winrm(port:5985) rdp(port:3389)	(C : /Users/Administrator/Documents/files/password_flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C : /Windows/System32/flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C : /Program Files/OpenSSH/home/boba_fett/flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator, boba_fett}) (C : /Program Files/OpenSSH/home/lando_calrissian/flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator, lando_calrissian}) (C : /Program Files/Apache Software Foundation/tomcat/apache − tomcat − 8.0.33/flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C : /wamp/www/uploads/.flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator, NT AUTHORITY/LOCAL SERVICE}) (C : /Program Files/elasticsearch − 1.1.1/flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C:/Windows/System32/config/SAM,{NT AUTHORITY/SYSTEM, Administrator}) (C:/Program Files/OpenSSH/home/boba_fett/id_rsa,{NT AUTHORITY/SYSTEM, Administrator, boba_fett})	Refer to Table 4.1	n12-vm1 n12-vm3
n12-vm3 - 10.12.1.3 - Linux Ubuntu 20.04			
$\mathbb{S}_{n12-vm3}$ (Services)	$\mathbb{P}_{n12-vm3}$ (Paths)	$\mathbb{A}_{n12-vm3}$ (Accounts)	$\mathbb{N}_{n12-vm3}$ (Neighboring)
vsftpd(port:21) ssh(port:22) apache(port:80) vnc(port:5900)	(/var/www/html/drupal/flag.txt$^{\odot}$, {root, www − data}) (/usr/share/vsftpd/flag.txt$^{\odot}$, {root, ftp}) (/home/captain/Desktop/flag.txt$^{\odot}$, {root, captain}) (/home/boba_fett/flag.txt$^{\odot}$, {root, boba_fett}) (/root/final_flag.txt$^{\odot}$, {root}) (/etc/shadow,{root}) (/var/www/html/backup.tar.gz,{root,www-data}) (/home/captain/.bash_history,{root,captain})	Refer to Table 4.1	n12-vm1 n12-vm2

TABLE 4.2 – Caractéristiques des machines de l'architecture du SI à compromettre dans l'expérimentation PWNJUTSU.

Le jeu de données PWNJUTSU a été obtenu grâce aux dispositifs de supervision installés sur l'infrastructure dédiée. Les 22 attaquants ayant participé à l'expérimentation avaient à leur disposition un réseau isolé qu'ils devaient compromettre. Nous avons décrit ce réseau selon notre modèle. Concrètement, il s'agit de quatre machines dont les spécifications sont précisées dans la Table 4.2. Dans cette table, le symbole "$\mathsf{p}^{\odot}$" est utilisé pour désigner les chemins p vers des fichiers contenant des secrets.

4.2.2 Déploiement

L'expérimentation PWNJUTSU a été déployée sur un système virtualisé, installé sur un serveur dédié et hébergé dans un centre de données français de l'hébergeur OVH. Le serveur présentait les spécifications suivantes :

— CPU AMD EPYC 7371 avec 16 cœurs
— 256 Go de mémoire vive en DDR4
— 2 disques durs de 4 To
— VMWare ESXi 7

Afin d'automatiser la préparation et le déploiement des machines virtuelles, nous avons utilisé les outils de *DevOps* Packer [3], Vagrant [4] et Ansible [5]. Les scripts de déploiement et de *provisioning* sont présentés en Annexes A.1 et A.2.

L'infrastructure de l'expérimentation est composée de trois zones distinctes, telles que présentées sur la Figure 4.2 :

— **Zone d'administration** avec une machine nommée "Deployer", qui a pour rôle de préparer les instances des participants et de gérer leurs déploiements dans la zone de jeu.
— **Zone des routeurs** dans laquelle chacune des passerelles des zones de jeu a une interface. Ces dernières communiquent directement avec le SIEM, lui-même situé dans la zone des routeurs.
— **Zones de jeu** dans lesquelles les participants à l'expérimentation progressent individuellement au travers de trois machines vulnérables. Chaque participant travaille dans une zone de jeu qui lui est dédiée.

3. https://www.packer.io/
4. https://www.vagrantup.com/
5. https://www.ansible.com/

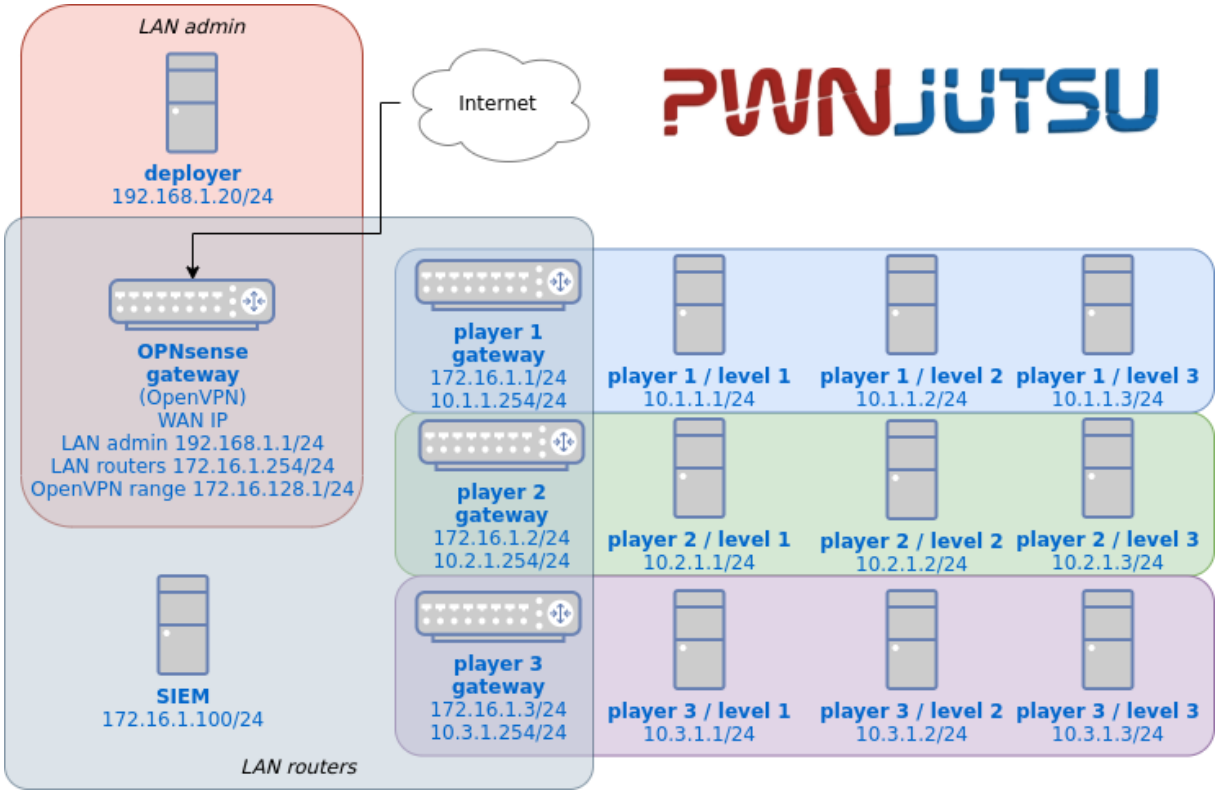

FIGURE 4.2 – Infrastructure de l'expérimentation PWNJUTSU

Systèmes de supervision

Afin d'observer correctement les actions des participants, nous avons installé plusieurs systèmes de supervision, qui se focalisent sur les systèmes et les réseaux.

Sur les machines Linux, nous avons :

— installé l'outil `Snoopy`[6], qui journalise dans `auth.log` toutes les exécutions de programmes et les lignes de commandes associées ;

— configuré le composant Linux `auditd`, en suivant le guide "Best Practice Auditd Configuration"[7] ;

— ajouté des règles spécifiques de supervision des accès en lecture à certains fichiers, notamment les *flags* cachés dans les systèmes de fichiers des machines vulnérables ;

— activé la journalisation des requêtes web reçues par le serveur `apache2` (`access.log` et `error.log`), ainsi que toutes les requêtes SQL reçues par le serveur `MySQL`.

6. https://github.com/a2o/snoopy
7. https://github.com/Neo23x0/auditd

— modifié le comportement du serveur `SSH`, afin que soient journalisés dans `auth.log`, en plus des noms d'utilisateurs, les mots de passe associés, pour chacune des tentatives de connexion.

Sur les machines Windows, nous avons :

— installé le service Sysmon [8] et avons appliqué une configuration populaire [9], généralement utilisée comme base de déploiement d'un système de supervision [65] ;

— configuré `Windows Audit` afin de superviser les accès en lecture à certains fichiers, tels que les *flags* cachés ;

— collecté les journaux d'événements natifs de l'*Operating System* (OS) : `System`, `Application` et `Security`.

Tous ces événements étaient transmis au SIEM (Splunk) pour y être indexé.

Les différentes sondes étaient donc volontairement configurées avec un haut niveau de verbosité afin d'optimiser les chances d'observation des participants. De plus, une capture indiscriminée du trafic réseau a été réalisée sur chacune des interfaces réseau (entrante et sortante) des machines virtuelles. L'outil natif de VMWare `pktcap-uw` a ainsi permis de produire un fichier PCAP par machine virtuelle.

Machines vulnérables

Sur les machines installées dans les zones de jeu, nous avons volontairement implémenté des vulnérabilités considérées comme évidentes et faciles à exploiter. Ainsi, les compétences techniques spécifiques des participants n'étaient pas une contrainte à prendre en compte dans le scénario ou dans la constitution du panel. Ce choix délibéré a permis aux participants de l'expérimentation PWNJUTSU de ne pas perdre de temps sur des considérations techniques pour gagner des accès sur les différentes machines. Ils pouvaient ainsi se concentrer sur le cœur de l'expérimentation : la progression tactique vers leurs objectifs respectifs, notamment grâce à des procédures d'attaques de *Lateral Movement*, de *Credential Access*, de *Privilege Escalation*, de *Discovery* ou encore de *Persistence*. Cet aspect de conception de l'expérimentation est essentiel, puisque si les participants avaient été confrontés à des défis techniques longs à résoudre afin de compromettre les machines, nous n'aurions pas observé leurs habitudes de propagation dans un SI, mais plutôt leurs compétences techniques spécifiques. Les traces que nous aurions collectées auraient par conséquent introduit un biais dans le jeu de données.

8. https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon
9. https://github.com/SwiftOnSecurity/sysmon-config

C'est pour ces raisons que les machines[10] `nXX-vm1` et `nXX-vm2` sont respectivement des machines Linux et Windows empruntées au projet public `Metasploitable 3`[11], que nous avons adaptées pour ne conserver que les services et comptes qui étaient pertinents pour l'expérimentation PWNJUTSU.

La machine `nXX-vm3` repose sur un système Linux Ubuntu personnalisé, dans lequel nous avons implémenté les vulnérabilités suivantes :

— `vsftpd` (CVE-2011-2523)
— `drupal` (CVE-2018-7600)
— `ssh` (mot de passe `root` faible)
— `vnc` (mot de passe faible)

Enfin, sur chaque machine des fichiers *flags* ont été dissimulés à l'intérieur des répertoires de travail des utilisateurs (*home directories*) et des services vulnérables (*working directories*). Il a été demandé aux participants de collecter les secrets contenus dans ces *flags*, systématiquement identifiables par des noms de fichier terminant par `flag.txt`. Nous considérons ainsi l'accès en lecture à ces fichiers comme des témoins de passage, qui atteste de la progression tactique du participant. Dans la mesure où, ils étaient rémunérés en fonction de ces découvertes, ces *flags* secrets étaient des objectifs intermédiaires pour les participants et la collecte du *flag* "final" leur permettait d'accéder au gain maximum.

Il est à noter que cette approche existe également dans le domaine de la défense active, où les défenseurs mettent en place des *honeytokens* (ou *canary*) [66] dans leurs SI et les surveillent activement pour détecter la présence d'un attaquant. Cela peut prendre la forme de fichiers, d'entrées dans une base de données, de messages électroniques, de comptes d'utilisateurs...

4.3 Progression de l'attaquant et panel de l'expérimentation

Dans cette section, nous proposons un modèle de progression tactique d'un acteur assimilé à une menace sophistiquée dans un SI compromis. Nous présentons également brièvement le profil des participants impliqués dans l'expérimentation PWNJUTSU. Le modèle repose sur une sémantique de techniques d'attaques, qui permet de représenter

10. où `XX` représente l'identifiant unique de l'instance et donc du participant
11. https://github.com/rapid7/metasploitable3

formellement la campagne d'attaque d'un participant de l'expérimentation. Cette représentation est détaillée dans la Section 4.5.

4.3.1 Description de l'attaquant

La connaissance d'un attaquant à propos du SI est initialement incomplète. À mesure que l'attaque progresse, sa connaissance à propos de ce SI évolue, grâce à l'usage de techniques de *Discovery*. Elles lui permettent d'en apprendre plus sur l'organisation, les données, les services ou encore les comptes des utilisateurs. Une catégorisation, proposée par Roy *et al* dans le cadre de leur enquête scientifique [67] est présentée sur la Figure 4.3.

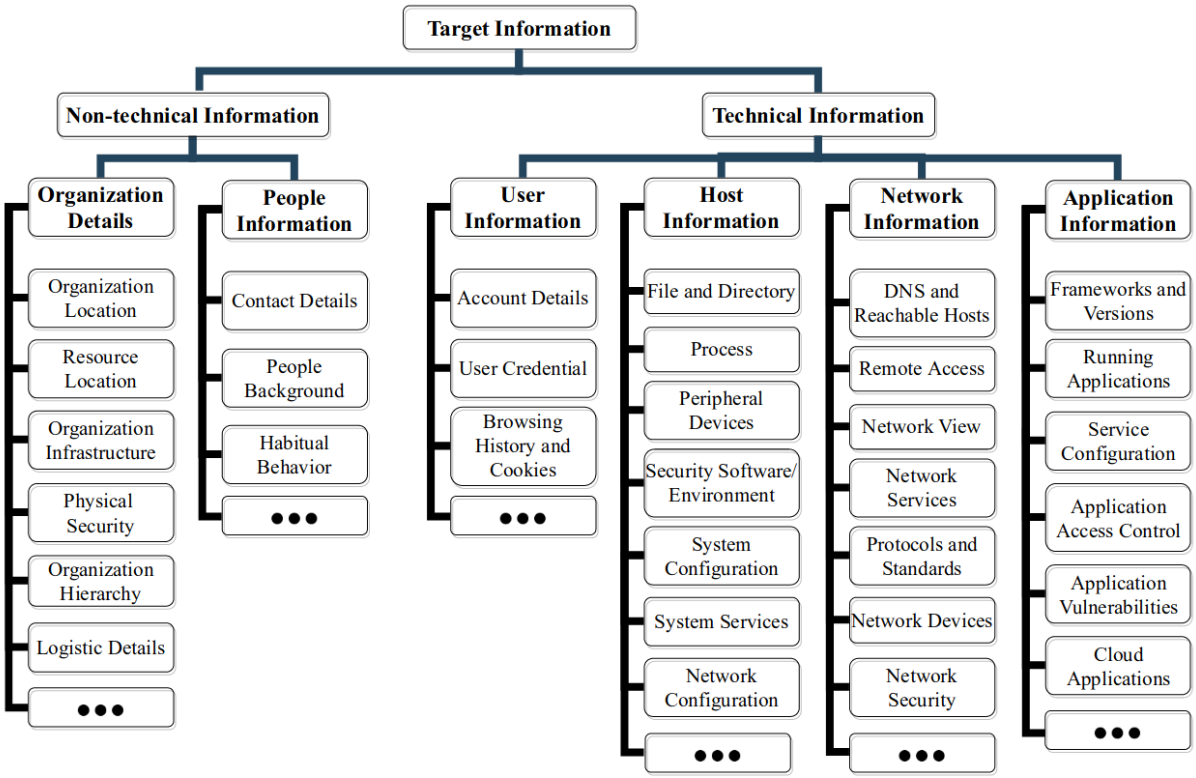

FIGURE 4.3 – Catégorisation des informations recherchées par l'attaquant lors de l'exécution de techniques de *Discovery*, proposée par Roy *et al*

La notion de connaissance partielle à propos d'une machine $\mathbf{m}$ est désignée par le tuple

$$\lfloor \mathbf{m} \rfloor = (\lfloor \mathbb{S}_{\mathbf{m}} \rfloor, \lfloor \mathbb{P}_{\mathbf{m}} \rfloor, \lfloor \mathbb{A}_{\mathbf{m}} \rfloor, \lfloor \mathbb{N}_{\mathbf{m}} \rfloor)$$

Il spécifie que l'attaquant connaît à propos des machines du SI :

— une partie des services,

— une partie des chemins sur les systèmes de fichiers,

— une partie des comptes utilisateurs et

— une partie des machines du voisinage réseau.

Ces ensembles peuvent également contenir soit des éléments de l'infrastructure de la victime, soit des éléments ajoutés par l'attaquant dans le SI et dont lui seul a la connaissance (par exemple, des *backdoors* [68] ou encore des comptes utilisateurs illégitimes). Ces éléments peuvent notamment être créés suite à l'exécution d'une technique de *Persistence*.

De plus, $\lfloor \mathbb{A}_{\mathbf{m}} \rfloor$ peut contenir des tuples $(\mathbf{u}, \mathsf{s}, \mathsf{k}, \ell)$ dans lesquels les valeurs de s, k, ou ℓ peuvent être $\perp$ si cette information est inconnue.

Quand un attaquant met à jour sa connaissance partielle à propos d'une machine $\mathbf{m}$ en considérant un ensemble P de nouveaux chemins sur le système de fichiers, nous écrivons $\lfloor \mathbb{P}_{\mathbf{m}} \rfloor \oplus P$ l'ensemble de chemins obtenus par l'ajout de chaque $(\mathsf{p}, \mathcal{U}) \in P$, nouveaux utilisateurs de $\mathcal{U}$ si p apparaît déjà dans $\lfloor \mathbb{P}_{\mathbf{m}} \rfloor$; sinon en ajoutant $(\mathsf{p}, \mathcal{U})$ dans $\lfloor \mathbb{P}_{\mathbf{m}} \rfloor$.

Par exemple,

$$\{(\mathsf{p}_1, \{\mathbf{u}_1, \mathbf{u}_2\})\} \ \oplus \ \{(\mathsf{p}_1, \{\mathbf{u}_2, \mathbf{u}_3\}), (\mathsf{p}_2, \{\mathbf{u}_1\})\} \quad = \quad \{(\mathsf{p}_1, \{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}), (\mathsf{p}_2, \{\mathbf{u}_1\})\}$$

De la même manière, étant donné un ensemble A de nouveaux comptes utilisateurs, nous écrivons $\lfloor \mathbb{A}_{\mathbf{m}} \rfloor \oplus A$, l'ensemble des comptes obtenus :

— en remplaçant les valeurs $\perp$, apparaissant dans les comptes des utilisateurs $\mathbf{u}$ déjà présents dans $\lfloor \mathbb{A}_{\mathbf{m}} \rfloor$ par les nouvelles valeurs fournies par A pour $\mathbf{u}$; et

— en ajoutant dans $\lfloor \mathbb{A}_{\mathbf{m}} \rfloor$ les comptes de A qui n'y apparaissent pas encore.

Par exemple,

$$\{(\mathbf{u}, \mathsf{s}_1, \perp, \ell)\} \quad \oplus \quad \{(\mathbf{u}, \mathsf{s}_1, \mathsf{k}_1, \ell), (\mathbf{u}, \mathsf{s}_2, \mathsf{k}_2, \ell)\} \qquad = \qquad \{(\mathbf{u}, \mathsf{s}_1, \mathsf{k}_1, \ell), (\mathbf{u}, \mathsf{s}_2, \mathsf{k}_2, \ell)\}$$

Nous avons choisi de représenter une campagne d'attaque comme la progression d'un attaquant dans sa conquête du SI. À chaque étape de cette conquête, l'attaquant fait évoluer son état et sa connaissance. Formellement, cet état est défini par le tuple *muSE* :

$$(\mu, \mathcal{S}, \mathcal{E})$$

— μ est un ensemble contenant toutes les paires $(\mathbf{m}, \mathbf{u})$ où l'attaquant possède un accès sur la machine $\mathbf{m}$ en empruntant l'identité de l'utilisateur $\mathbf{u}$: il s'agit de l'**espace de propagation** de l'attaquant.

— $\mathcal{S}$ est l'ensemble des contenus des fichiers connus par l'attaquant, qui contiennent des secrets et qui sont une partie de ses objectifs : tous les $*\mathsf{p}$ tels que $\mathsf{p}^{\odot}$.

— $\mathcal{E}$ est la connaissance acquise par l'attaquant à propos de l'environnement du SI de la victime. L'exhaustivité de cette connaissance est un idéal recherché par l'attaquant, mais qui n'a généralement pas besoin d'être atteint, puisque l'objectif final est souvent atteint avant. Nous représentons donc cette connaissance incomplète grâce à la notion de vue partielle : $\lfloor \mathbf{m} \rfloor_{\mathcal{E}}$ pour chaque machine $\mathbf{m}$.

Au cours d'une campagne, l'attaquant peut utiliser ses capacités techniques qui sont, d'une part, les techniques d'attaques qu'il maîtrise ; d'autre part, les vulnérabilités pour lesquelles il possède un *exploit* fonctionnel, privé ou public. Dans la suite de cette partie, nous appellerons `Exploits` la base de données d'*exploits* maîtrisés par l'attaquant. Par exemple, cela peut être la base de données publique "Exploit-DB" [12]. Les techniques sont détaillées dans la Section 4.4 et elles sont ensuite illustrées dans la Section 4.5.

4.3.2 Panel de l'expérimentation

Afin de conduire notre expérimentation, nous avions besoin d'acteurs qui étaient techniquement capables de se comporter comme des attaquants. C'est pourquoi nous avons recherché des profils de *Red Teamers*. Nous nous sommes donc tournés vers la plateforme YesWeHack et nous avons sélectionné 22 participants parmi les membres du TOP 100 [13] des utilisateurs actifs. Ces 22 participants représentaient 9 nationalités différentes. Pour les motiver à participer assidûment à l'expérimentation, nous leur avons proposé une rétribution financière intéressante [14], en phase avec la tendance du moment sur le marché de la chasse de vulnérabilités (*Bug Bounty*). Ils étaient donc motivés par l'atteinte d'un but concret : un gain financier.

12. https://www.exploit-db.com
13. https://yeswehack.com/ranking?period=All
14. Le financement de cette expérimentation a été assuré par le Bureau d'Etudes en Cyber sécurité et systèmes de Protection (BCyP) de l'Institut de Radioprotection et de Sûreté Nucléaire (IRSN).

Cette rétribution s'est faite suivant une grille progressive précise en fonction du niveau atteint par le participant :

— 50€ pour la découverte d'un *flag* du niveau 1 ;
— 200€ pour la découverte d'un *flag* du niveau 2 ;
— 300€ pour la découverte du *flag* final du niveau 3 ;
— 400€ pour la réponse à un questionnaire *a posteriori*.

Afin de tester l'architecture et le scénario de l'expérimentation PWNJUTSU, une dizaine de participants ont été gracieusement impliqués. Cette première phase a permis d'améliorer la qualité du scénario et du système de journalisation, mais également de corriger quelques erreurs d'implémentation. Au cours de la seconde phase, celle de conduite de l'expérimentation, 22 participants ont attaqué la version stable de la plateforme dans le but de découvrir les *flags* qui y étaient dissimulés, jusqu'à atteindre le *flag* final. Le jeu de données PWNJUTSU est en fait la concaténation des traces de ces 22 participants.

Le profil type d'un participant du panel est :

— 25-35 ans (63%) ;
— Diplôme de niveau 6-7 [15] (91%) ;
— Possède une certification en "*Hacking* éthique" (64%) ;
— Déclare être un autodidacte en sécurité offensive (100%).

Dans la Section 4.5, nous utilisons notre modèle formel pour détailler, étape par étape, l'attaque conduite par l'un des participants à l'expérimentation et nous fournissons un aperçu des traces relatives à ses actions techniques.

4.4 Sémantique des techniques d'attaques

Pour progresser vers ses objectifs, l'attaquant met en œuvre des techniques d'attaques qui satisfont des intentions tactiques. Par exemple, obtenir un nouvel accès, dévoiler de nouveaux identifiants valides, jouir de nouveaux privilèges ou encore acquérir de la connaissance sur l'environnement de la victime. Chaque exécution d'une technique conduit à faire évoluer la perception qu'a l'attaquant du SI ; ou à renforcer le contrôle qu'il a sur les équipements compromis. La matrice MITRE ATT&CK™ [21] fournit une liste détaillée des techniques connues à ce jour et observées par les experts CTI. Elle n'est certes pas exhaustive, mais c'est la base de connaissances la plus complète disponible publiquement. Néanmoins, la spécification des techniques est informelle (donnée en langage naturel), ce

15. Selon la classification ISCED, il s'agit d'un niveau Licence-Master ou équivalent.

qui restreint son utilisation pour retracer précisément les campagnes d'attaques.

Les outils développés à des fins de sécurité offensive et les *malware* sont les implémentations concrètes de ces techniques d'attaques, aussi appelées "procédures". Par conséquent, il existe plusieurs procédures pour une même technique d'attaque. Avant leurs instanciations, les techniques d'attaques sont paramétrées [19] en fonction de l'environnement de la victime qui subira l'exécution de la technique. Par exemple, ces paramètres peuvent être relatifs à des services, des machines, des utilisateurs...

La mise en œuvre de ces différentes techniques d'attaques permet à l'attaquant de :

— prendre le contrôle de certains comptes utilisateurs ou gagner de nouveaux privilèges : extension de l'espace de propagation μ selon notre modèle ;
— découvrir de nouveaux secrets : évolution de $\mathcal{S}$;
— affiner sa connaissance de l'environnement du SI : évolution de $\mathcal{E}$.

Nous définissons dans cette section une sémantique opérationnelle des techniques d'attaques qui peuvent être considérées comme des spécifications abstraites de procédures.

Ainsi, une progression de l'attaquant au cours d'une campagne est une séquence de transitions de la forme :

$$(\mu_i, \mathcal{S}_i, \mathcal{E}_i) \xrightarrow{\mathbf{t}(params)} (\mu_{i+1}, \mathcal{S}_{i+1}, \mathcal{E}_{i+1})$$

à partir de l'état initial $(\mu_0, \emptyset, \mathcal{E}_0)$ avec $\mu_0 = \{(\mathbf{m}_0, \mathbf{u}_0)\}$. Cela signifie que l'attaquant a obtenu un accès initial sur la machine $\mathbf{m}_0$ en empruntant l'identité de l'utilisateur $\mathbf{u}_0$ et qu'il a déjà acquis une connaissance $\mathcal{E}_0$ à propos de l'environnement de la victime.

L'exécution d'une technique $\mathbf{t}$ avec les paramètres *params* conduit l'attaquant d'un état $(\mu_i, \mathcal{S}_i, \mathcal{E}_i)$ vers un état $(\mu_{i+1}, \mathcal{S}_{i+1}, \mathcal{E}_{i+1})$ où les ensembles $\mu_i, \mathcal{S}_i$ ou $\mathcal{E}_i$ sont mis à jour selon l'effet attendu par l'attaquant suite à l'exécution de la technique si les préconditions sur $(\mu_i, \mathcal{S}_i, \mathcal{E}_i)$ sont satisfaites. Nous décrivons formellement la mise à jour de $\mathcal{E}$ en $\mathcal{E}'$, étant données de nouvelles connaissances sur la machine $\mathbf{m}'$ par :

$$\mathcal{E}' = \mathcal{E}[\mathbf{m}' \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}'} \rfloor, \lfloor \mathbb{P}_{\mathbf{m}'} \rfloor, \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor, \lfloor \mathbb{N}_{\mathbf{m}'} \rfloor)]$$

avec pour toute machine $\mathbf{m}$:

$$\lfloor \mathbf{m} \rfloor_{\mathcal{E}'} = \begin{cases} (\lfloor \mathbb{S}_{\mathbf{m}'} \rfloor, \lfloor \mathbb{P}_{\mathbf{m}'} \rfloor, \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor, \lfloor \mathbb{N}_{\mathbf{m}'} \rfloor) & \text{si } \mathbf{m} = \mathbf{m}' \\ \lfloor \mathbf{m} \rfloor_{\mathcal{E}} & \text{sinon} \end{cases}$$

Nous fournissons ici la sémantique pour 13 techniques, qui satisfont 5 tactiques pertinentes dans le déroulement du scénario de l'expérimentation PWNJUTSU. Parmi toutes les techniques d'attaques décrites dans la matrice ATT&CK du MITRE, nous nous sommes focalisés sur celles qui sont incontournables pour un attaquant qui chercherait à progresser en phase opérationnelle de *Propagation réseau*. Cependant, ce modèle est suffisamment flexible pour permettre de spécifier des techniques qui satisfont d'autres tactiques et qui pourraient être mises en œuvre par des attaquants en progression dans d'autres phases opérationnelles de leurs campagnes.

4.4.1 *Lateral Movement*

La tactique de *Lateral Movement* peut être définie comme l'exécution de techniques qu'utilisent les attaquants pour prendre pied sur un système distant du SI d'une victime et le contrôler à des fins malveillantes. En d'autres termes, l'usage de cette tactique permet à l'attaquant d'obtenir un accès sur une machine $\mathbf{m}'$ en empruntant l'identité d'un utilisateur $\mathbf{u}'$. Selon notre sémantique, nous exprimons l'exécution d'une de ces tactiques par l'évolution de μ_i. Nous décrivons ici deux techniques populaires qui satisfont des intentions de *Lateral Movement* :

— *Remote Services* ($\mathbf{t}_{1021}$)
— *Exploitation of Remote Services* ($\mathbf{t}_{1210}$)

En mettant en œuvre la technique $\mathbf{t}_{1021}$, l'attaquant peut obtenir un accès sur la machine $\mathbf{m}'$ en utilisant la clé k' d'un utilisateur $\mathbf{u}'$ sur le service s ; si l'attaquant a connaissance de l'existence de l'utilisateur $\mathbf{u}'$. Ces quatre éléments sont une partie des paramètres nécessaires à l'exécution de cette technique. De plus, trois préconditions doivent être satisfaites :

— $(\mathbf{m}, \mathbf{u})$ est l'emplacement d'où l'attaquant exécutera la technique. Ce couple machine $\mathbf{m}$ et utilisateur $\mathbf{u}$ doit être dans l'espace de propagation μ de l'attaquant. Cela signifie que l'attaquant doit avoir un accès à cette machine avec les privilèges de cet utilisateur ;
— la machine ciblée $\mathbf{m}'$ doit être joignable, par le réseau, depuis la machine $\mathbf{m}$ et l'attaquant doit avoir cette connaissance ;
— l'attaquant doit connaître l'existence du compte $(\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell)$ sur la machine $\mathbf{m}'$.

Après l'exécution de cette technique, l'attaquant aura étendu son espace de propagation avec le couple $(\mathbf{m}', \mathbf{u}')$. Cette technique est décrite formellement dans la Table 4.3.

La technique $\mathbf{t}_{1210}$, formellement décrite dans la Table 4.4, présente l'exploitation à

Technique	$\mathbf{t}_{1021}$: *Remote Services*
Tactique	*Lateral movement*
Description	Obtenir un accès sur la machine $\mathbf{m}'$ en utilisant la clé k' pour l'utilisateur $\mathbf{u}'$ sur le service s.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, $\mathbf{m}'$, $\mathbf{u}'$, k', s
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $\mathbf{m}' \in \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}}$ et $(\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu', \mathcal{S}, \mathcal{E})$ où $\mu' = \mu \cup \{(\mathbf{m}', \mathbf{u}')\}$
Variantes	$\mathbf{t}_{1021.001}$ avec $\mathsf{s} = \texttt{RDP}$; $\mathbf{t}_{1021.002}$ avec $\mathsf{s} = \texttt{SMB}$; $\mathbf{t}_{1021.003}$ avec $\mathsf{s} = \texttt{DCOM}$; $\mathbf{t}_{1021.004}$ avec $\mathsf{s} = \texttt{SSH}$; $\mathbf{t}_{1021.005}$ avec $\mathsf{s} = \texttt{VNC}$; $\mathbf{t}_{1021.006}$ avec $\mathsf{s} = \texttt{WinRM}$

TABLE 4.3 – *Lateral movement* par *Remote Services*

distance d'un service vulnérable. Elle est fortement similaire à la technique précédente, à l'exception des identifiants $\mathbf{u}''$ et k'' qui ne sont obligatoires que pour les *exploits* dits "post-authentication". L'attaquant a besoin de spécifier l'*exploit* x en tant que paramètre d'exécution de la technique. L'attaquant se réfère à $\mathit{Exploits}(\mathsf{s})$, qui est l'ensemble des *exploits* qu'il maîtrise pour les vulnérabilités applicables sur le service s. Cet ensemble ne peut pas être vide. Après l'exécution de cette technique, l'attaquant emprunte l'identité de l'utilisateur $\mathbf{u}'$ sur la machine $\mathbf{m}'$, alors qu'il ne connaît pas nécessairement la clé k' associée à ce compte.

4.4.2 *Credential Access*

La tactique de *Credential Access* peut être définie comme l'exécution de techniques qu'utilisent les attaquants pour dérober des identifiants valides, tels que des noms d'utilisateurs et des mots de passe. Dans notre modèle, l'attaquant met en œuvre ces techniques lorsqu'il collecte les identifiants liés à un service s. Nous décrivons ici deux techniques qui permettent de satisfaire cette intention :

— *Unsecured Credentials* ($\mathbf{t}_{1552}$)

— *Brute Force* ($\mathbf{t}_{1110}$)

Quand un attaquant met en œuvre la technique $\mathbf{t}_{1552}$, formellement décrite dans la Table 4.5, il collecte tous les identifiants liés au service s sur la machine $\mathbf{m}$ d'où il fait appel à la technique, en empruntant le compte de l'utilisateur $\mathbf{u}$. Plusieurs préconditions doivent également être satisfaites :

Technique	$\mathbf{t}_{1210}$: *Exploitation of Remote Services*
Tactique	*Lateral movement*
Description	Obtenir un accès sur la machine $\mathbf{m}'$ en exploitant à distance une vulnérabilité grâce à l'exploit x sur un service réseau exposé s.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, $\mathbf{m}'$, s, x
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $\mathbf{m}' \in \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}}$, $\mathsf{s} \in \lfloor \mathbb{S}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$ et $x \in \mathit{Exploits}(\mathsf{s})$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu', \mathcal{S}, \mathcal{E})$ où $\mu' = \mu \cup \{(\mathbf{m}', \mathbf{u}')\}$ avec $(\mathbf{u}', \mathsf{s}, \mathsf{k}, \ell) \in \mathbb{A}_{\mathbf{m}'}$
Variantes	Les vulnérabilités nécessitant d'être authentifiées font appel aux paramètres additionnels $\mathbf{u}''$ et k'' tels que $(\mathbf{u}'', \mathsf{s}, \mathsf{k}'', \ell'') \in \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$ (Note : $\mathbf{u}''$ peut être identique à $\mathbf{u}'$)

TABLE 4.4 – *Lateral movement* par *Exploitation of Remote Services*

— la machine $\mathbf{m}$ et l'utilisateur $\mathbf{u}$ doivent faire partie de l'espace de propagation de l'attaquant, cela signifie que l'attaquant possède un accès sur cette machine pour cet utilisateur ;

— le compte lié à l'identité de l'utilisateur $\mathbf{u}$ doit être un compte valide pour le service, mais cela fonctionne également si l'attaquant ne connaît pas la clé k pour ce compte (*i.e.*, $\mathsf{k} = \bot$).

Après l'exécution de cette technique, l'attaquant aura enrichi sa connaissance à propos de l'environnement $\mathcal{E}$ du SI, avec les nouveaux comptes utilisateurs collectés.

Pour la technique $\mathbf{t}_{1110}$, l'attaquant connaît l'existence de l'utilisateur $\mathbf{u}'$ pour le service s' sur la machine $\mathbf{m}'$ et il peut joindre cette machine par le réseau depuis la machine d'où il exécute la technique, qui est présente dans son espace de propagation.

Après l'exécution de cette technique, l'attaquant aura découvert la clé convoitée, il pourra ainsi l'intégrer à sa connaissance de l'environnement de la machine ciblée. Cette technique est formellement décrite dans la Table 4.6.

Technique	$\mathbf{t}_{1552}$: *Unsecured Credentials*
Tactique	*Credential access*
Description	Collecte de tous les identifiants valides liés au service s sur la machine $\mathbf{m}$
Paramètres	$\mathbf{m}$, $\mathbf{u}$, s
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $(\mathbf{u}, \mathsf{s}, \mathsf{k}, \ell) \in \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' = \mathcal{E}[\mathbf{m} \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}} \oplus \{(\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell') \in \mathbb{A}_{\mathbf{m}}\}, \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}})]$
Variantes	$\mathbf{t}_{1552.001}$ (avec $\mathsf{s} = \texttt{filesystem}$) ; $\mathbf{t}_{1552.003}$ (avec $\mathsf{s} = \texttt{bash}$) ; $\mathbf{t}_{1552.004}$ (avec $\mathsf{s} = \texttt{filesystem}$ sur un fichier de clé privée) ; $\mathbf{t}_{1003.008}$ (avec $\mathsf{s} = \mathtt{OS}_{\texttt{Linux}}$ et $\ell = \texttt{high}$) ; $\mathbf{t}_{1003.002}$ (avec $\mathsf{s} = \mathtt{OS}_{\texttt{Windows}}$ et $\ell = \texttt{high}$)

TABLE 4.5 – *Credential access* par *Unsecured Credentials*

Technique	$\mathbf{t}_{1110}$: *Brute Force*
Tactique	*Credential Access*
Description	Récupère la clé k' de l'utilisateur $\mathbf{u}'$ pour le service s' de la machine $\mathbf{m}'$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, $\mathbf{m}'$, $\mathbf{u}'$, s'
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $\mathbf{m}' \in \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}}$ et $(\mathbf{u}', \mathsf{s}', \bot, \ell') \in \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' =$ $\mathcal{E}[\mathbf{m}' \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}} \oplus \{(\mathbf{u}', \mathsf{s}', \mathsf{k}', \ell') \in \mathbb{A}_{\mathbf{m}'}\}, \lfloor \mathbb{N}_{\mathbf{m}'} \rfloor_{\mathcal{E}})]$ où $\mathsf{k}' \neq \bot$
Variantes	$\mathbf{t}_{1110.001}$ avec k obtenue par *guessing* ; $\mathbf{t}_{1110.002}$ avec k obtenue par *cracking*

TABLE 4.6 – *Credential access* par *Brute Force*

4.4.3 *Privilege Escalation*

La tactique de *Privilege Escalation* permet aux attaquants d'élever leur niveau de privilèges sur une machine. Selon notre modèle, cela correspond à la transition d'un utilisateur $\mathbf{u}$ à bas privilèges vers un utilisateur $\mathbf{u}'$ à hauts privilèges. Nous décrivons ici une technique qui permet de satisfaire cette intention tactique. Il s'agit de la technique *Abuse Elevation Control Mechanism* ($\mathbf{t}_{1548}$). Cette technique peut être exécutée si un utilisateur $\mathbf{u}$, à qui l'attaquant emprunte l'identité sur une machine $\mathbf{m}$, possède un niveau de privilège `low*`. Cela signifie que l'utilisateur est légitimement autorisé à élever ses privilèges (*i.e.*, grâce à la commande `sudo`).

Après l'exécution de cette technique, l'attaquant est toujours connecté sur la machine $\mathbf{m}$, mais il emprunte désormais l'identité d'un utilisateur $\mathbf{u}'$, qui possède de hauts privilèges sur la machine $\mathbf{m}$. Cette technique est formellement décrite dans la Table 4.7.

Technique	$\mathbf{t}_{1548}$: *Abuse Elevation Control Mechanism*
Tactique	*Privilege escalation*
Description	Obtient les hauts privilèges de l'utilisateur $\mathbf{u}'$ depuis l'utilisateur à bas privilèges $\mathbf{u}$ sur la machine $\mathbf{m}$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$ $(\mathbf{u}, \mathtt{OS}, \mathtt{k}, \mathtt{low*}) \in \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu', \mathcal{S}, \mathcal{E})$ où $\mu' = \mu \cup \{(\mathbf{m}, \mathbf{u}')\}$ avec $(\mathbf{u}', \mathtt{OS}, \mathtt{k}', \mathtt{high}) \in \mathbb{A}_{\mathbf{m}}$
Variantes	$\mathbf{t}_{1548.003}$ avec $\mathtt{s} = \mathtt{OS}_{\mathtt{Linux}}$ (`sudo` command)

TABLE 4.7 – *Privilege escalation* par *Abuse Elevation Control Mechanism*

4.4.4 *Discovery*

La tactique de *Discovery* permet à un attaquant d'enrichir sa connaissance à propos de l'environnement du SI. Selon notre modèle, l'enrichissement de cette connaissance consiste à ajouter des informations dans l'ensemble $\mathcal{E}$. La connaissance de l'attaquant est relative :

— aux services $\lfloor \mathbb{S}_{\mathbf{m}} \rfloor$;
— aux chemins des fichiers et répertoires $\lfloor \mathbb{P}_{\mathbf{m}} \rfloor$;
— aux comptes utilisateurs $\lfloor \mathbb{A}_{\mathbf{m}} \rfloor$;
— aux machines du voisinage réseau $\lfloor \mathbb{N}_{\mathbf{m}} \rfloor$.

Nous décrivons ici cinq techniques qui permettent de satisfaire cette intention tactique :

— *File and Directory Discovery* (local et distant) ($\mathbf{t}_{1083}$)
— *Network Service Scanning* ($\mathbf{t}_{1046}$)
— *Remote System Discovery* ($\mathbf{t}_{1018}$)
— *Account Discovery* (local et distant) ($\mathbf{t}_{1087}$)
— *System Service Directory* ($\mathbf{t}_{1007}$)

Les Tables 4.8 et 4.9 présentent la technique $\mathbf{t}_{1083}$, mais notre sémantique permet de différencier la version "locale" de la version "distante".

Dans le premier cas ($\mathbf{t}_{1083}$), l'attaquant découvre les fichiers et répertoires accessibles au chemin p sur le système de fichiers de la machine $\mathbf{m}$, d'où il exécute la technique. La machine $\mathbf{m}$ et l'utilisateur $\mathbf{u}$ doivent être dans l'espace de propagation μ de l'attaquant. De plus, l'utilisateur $\mathbf{u}$ doit être dans l'ensemble d'utilisateurs $\mathcal{U}$ autorisés à accéder au fichier situé au chemin p. Après l'exécution de cette technique, l'attaquant enrichit sa connaissance de l'environnent $\mathcal{E}$ avec les fichiers et répertoires découverts. Si cela est applicable (*i.e.*, $\mathsf{p}.x^{\odot}$: si le contenu du fichier contient un secret), l'attaquant collecte de nouveaux secrets et les capitalise dans $\mathcal{S}$.

Dans le second cas ($\mathbf{t}_{1083}{}'$), l'unique différence est que la machine ciblée par l'attaquant est un système distant $\mathbf{m}'$ et que l'attaquant doit fournir des identifiants valides, avec un utilisateur $\mathbf{u}'$ et une clé k', pour se connecter au service s.

La technique $\mathbf{t}_{1046}$, présentée dans la Table 4.10, permet à un attaquant de découvrir tous les services réseau exposés sur une machine distante en scannant ses ports réseau, compris entre 0 et 65535. Afin de mettre en œuvre cette technique, la machine ciblée $\mathbf{m}'$ doit être joignable depuis la machine $\mathbf{m}$, d'où l'attaquant exécute la technique. De plus, la machine $\mathbf{m}$ doit être dans l'espace de propagation μ de l'attaquant. Après l'exécution de cette technique, l'attaquant a enrichi sa connaissance à propos de l'environnement $\mathcal{E}$ du SI et plus particulièrement à propos des services $\mathbb{S}_{\mathbf{m}'}$ installés sur la machine ciblée.

Technique	$\mathbf{t}_{1083}$: *File and Directory Discovery* (local)
Tactique	*Discovery*
Description	Collecte tous les fichiers et répertoires situés au chemin p sur le système de fichiers de la machine $\mathbf{m}$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, p
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $(\mathsf{p}, \mathcal{U}) \in \lfloor \mathbb{P}_{\mathbf{m}} \rfloor_{\mathcal{E}}$ et $\mathbf{u} \in \mathcal{U}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}', \mathcal{E}')$ avec $\mathcal{S}' = \mathcal{S} \cup \{*\mathsf{p}.x \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathbf{m}}, \mathbf{u} \in \mathcal{U}' \text{ et } \mathsf{p}.x^{\odot}\}$ et $\mathcal{E}' = \mathcal{E}[\mathbf{m} \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{p}.x, \mathcal{U}') \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathbf{m}} \text{ et } \mathbf{u} \in \mathcal{U}'\}, \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}})]$
Variantes	$\mathbf{t}_{1083}{}'$, décrite dans la Table 4.9, est la même technique mais pour un système distant.

TABLE 4.8 – *Discovery* par *File and Directory Discovery* (local)

Technique	$\mathbf{t}_{1083}{}'$: *File and Directory Discovery* (distant)
Tactique	*Discovery*
Description	Collecte tous les fichiers et répertoires situés au chemin p sur le système de fichiers de la machine distante $\mathbf{m}'$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, $\mathbf{m}'$, $\mathbf{u}'$, k', p, s
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $(\mathsf{p}, \mathcal{U}) \in \lfloor \mathbb{P}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$ et $\mathsf{s} \in \lfloor \mathbb{S}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$, s est un service réseau (par exemple, Apache2, SMB...), $(\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$ avec $\mathsf{k}' \neq \bot$, $\mathbf{m}' \in \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}}$ et $\mathbf{u}' \in \mathcal{U}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}', \mathcal{E}')$ avec $\mathcal{S}' = \mathcal{S} \cup \{*\mathsf{p}.x \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathbf{m}'}, \mathbf{u}' \in \mathcal{U}' \text{ et } \mathsf{p}.x^{\odot}\}$ et $\mathcal{E}' = \mathcal{E}[\mathbf{m}' \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}'} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{p}.x, \mathcal{U}') \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathbf{m}'} \text{ et } \mathbf{u}' \in \mathcal{U}'\}, \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathbf{m}'} \rfloor_{\mathcal{E}})]$
Variantes	$\mathbf{t}_{1135}{}'$, où le chemin p est un partage réseau.

TABLE 4.9 – *Discovery* par *File and Directory Discovery* (distant)

Technique	$\mathbf{t}_{1046}$: *Network Service Scanning*
Tactique	*Discovery*
Description	Découverte de tous les services réseau d'une machine distante $\mathbf{m}'$ grâce au parcours de l'espace de nom des ports réseau $\Delta \subseteq \{0, \cdots, 65535\}$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, $\mathbf{m}'$, Δ
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $\mathbf{m}' \in \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' =$ $\mathcal{E}[\mathbf{m}' \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}'} \rfloor_{\mathcal{E}} \cup \{\mathsf{s}(\text{port} : i) \mid i \in \Delta\}, \lfloor \mathbb{P}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathbf{m}'} \rfloor_{\mathcal{E}})]$
Variantes	-

TABLE 4.10 – *Discovery* par *Network Service Scanning*

L'exécution de la technique $\mathbf{t}_{1018}$, présentée dans la Table 4.11, permet à un attaquant de découvrir les machines du voisinage réseau de la machine $\mathbf{m}$, d'où il exécute la technique. La machine $\mathbf{m}$ devant être dans son espace de propagation μ. Après l'exécution de cette technique, l'attaquant a enrichi ses connaissances à propos de l'environnement $\mathcal{E}$ du SI en ajoutant à $\lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}}$ l'ensemble de toutes les machines joignables depuis $\mathbf{m}$.

Technique	$\mathbf{t}_{1018}$: *Remote System Discovery*
Tactique	*Discovery*
Description	Découverte de toutes les machines voisines de $\mathbf{m}$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' = \mathcal{E}[\mathbf{m} \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}} \cup \mathbb{N}_{\mathbf{m}})]$
Variantes	$\mathbf{t}_{1049}$ (*System Network Connections Discovery*) permet de découvrir les machines voisines en observant les connexions établies sur la machine $\mathbf{m}$.

TABLE 4.11 – *Discovery* par *Remote System Discovery*

Les Tables 4.12 et 4.13 présentent deux variantes de la technique $\mathbf{t}_{1087}$. Il s'agit de découvrir les comptes utilisateurs d'un service s respectivement d'une machine locale $\mathbf{m}$ ou distante $\mathbf{m}'$. Afin d'exécuter cette technique, des préconditions doivent être satisfaites :

— la machine $\mathbf{m}$ doit faire partie de l'espace de propagation μ de l'attaquant ;
— l'attaquant doit connaître l'existence du service s sur la machine ciblée ($\mathbf{m}$ ou $\mathbf{m}'$) ;
— le compte $\mathbf{u}$ utilisé par l'attaquant doit être autorisé à accéder au service s.

Technique	$\mathbf{t}_{1087}$: *Account Discovery* (local)
Tactique	*Discovery*
Description	Récupération de tous les comptes utilisateur d'un service s sur la machine $\mathbf{m}$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, s
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $\mathsf{s} \in \lfloor \mathbb{S}_{\mathbf{m}} \rfloor_{\mathcal{E}}$ et $(\mathbf{u}, \mathsf{s}, \mathsf{k}, \ell) \in \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' = \mathcal{E}[\mathbf{m} \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}} \oplus \{(\mathbf{u}', \mathsf{s}, \bot, \ell) \mid (\mathbf{u}', \mathsf{s}, \mathsf{k}, \ell) \in \mathbb{A}_{\mathbf{m}}\}, \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}})]$
Variantes	$\mathbf{t}_{1087}{}'$, décrite ci-dessous, est la même technique mais pour un service distant.

TABLE 4.12 – *Discovery* par *Account Discovery* (local)

La différence avec la technique alternative $\mathbf{t}_{1087}{}'$, pour une machine distante $\mathbf{m}'$, est que l'attaquant a besoin de connaître le service s. Ce dernier doit être un service réseau. L'attaquant doit également pouvoir profiter d'un accès à ce service grâce au compte de l'utilisateur $\mathbf{u}'$ et de la clé k'. Enfin, la machine $\mathbf{m}'$ doit être joignable de la machine $\mathbf{m}$.

Après l'exécution de ces techniques, l'attaquant améliore sa connaissance $\mathcal{E}$ de l'environnement avec tous les comptes utilisateurs du service s. Il est important de préciser que cette technique permet la découverte des comptes et non la découverte des clés associées. Pour découvrir les clés, il est nécessaire de mettre en œuvre une technique de *Credential access*, présentée dans la Section 4.4.2.

Enfin, la technique formalisée dans la Table 4.14 permet à l'attaquant de découvrir tous les services installés sur une machine $\mathbf{m}$. La mise en œuvre de cette technique nécessite que la machine $\mathbf{m}$ et l'utilisateur $\mathbf{u}$ soient dans l'espace de propagation μ de l'attaquant. Après l'exécution de cette technique, l'attaquant aura enrichi sa connaissance $\mathcal{E}$ à propos de la machine $\mathbf{m}$ avec l'ensemble des services de la machine $\mathbb{S}_{\mathbf{m}}$.

Technique	$\mathbf{t}_{1087}{}'$: *Account Discovery* (distant)
Tactique	*Discovery*
Description	Récupération de tous les comptes utilisateur d'un service s sur la machine $\mathbf{m}'$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, $\mathbf{m}'$, $\mathbf{u}'$, k', s
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $\mathsf{s} \in \lfloor \mathbb{S}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$, s est un service réseau (par exemple, Apache2, SMB...) et $(\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}}$ with $\mathsf{k}' \neq \bot$ et $\mathbf{m}' \in \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' =$ $\mathcal{E}[\mathbf{m}' \leftarrow \left(\lfloor \mathbb{S}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}'} \rfloor_{\mathcal{E}} \oplus \left\{ \begin{array}{l} (\mathbf{u}', \mathsf{s}, \bot, \ell) \mid \\ (\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \mathbb{A}_{\mathbf{m}'} \end{array} \right\}, \lfloor \mathbb{N}_{\mathbf{m}'} \rfloor_{\mathcal{E}} \right)]$
Variantes	-

TABLE 4.13 – *Discovery* par *Account Discovery* (distant)

Technique	$\mathbf{t}_{1007}$: *System Service Discovery*
Tactique	*Discovery*
Description	Récupération de tous les services de la machine $\mathbf{m}$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' = \mathcal{E}[\mathbf{m} \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}} \rfloor_{\mathcal{E}} \cup \mathbb{S}_{\mathbf{m}}, \lfloor \mathbb{P}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}})]$
Variantes	$\mathbf{t}_{1057}$ (*Process discovery*) consiste à découvrir les services locaux grâce à leurs noms de processus.

TABLE 4.14 – *Discovery* par *System Service Discovery*

D'autres techniques de *Discovery* sont souvent mises en œuvre par les attaquants :

— *System Information Discovery* $\mathbf{t}_{1082}$;
— *System Network Configuration Discovery* $\mathbf{t}_{1016}$;
— *System Owner/User Discovery* $\mathbf{t}_{1033}$;
— *Internet Connection Discovery* $\mathbf{t}_{1016.001}$;

Ces techniques sont intéressantes pour l'attaquant afin d'acquérir de la connaissance sur la machine d'où elles sont exécutées. Cependant, elles ne contribuent pas directement à la progression tactique de l'attaquant vers ses objectifs. C'est pourquoi nous ne les avons pas décrites selon notre modèle.

4.4.5 *Persistence*

Les techniques de *Persistence* permettent aux attaquants de maintenir un accès sur un système du SI. La technique *Create Account* ($\mathbf{t}_{1136}$) est décrite dans la Table 4.15.

Afin de mettre en œuvre cette technique, l'attaquant doit fournir le 4-uplet $(\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell')$ qui permettra de créer le compte de l'utilisateur $\mathbf{u}'$, grâce aux privilèges de l'utilisateur $\mathbf{u}$, sur la machine locale $\mathbf{m}$. Les préconditions d'exécution sont l'existence du couple machine $\mathbf{m}$ et utilisateur $\mathbf{u}$ dans l'espace de propagation μ de l'attaquant. De plus, l'attaquant doit avoir accès au compte d'un utilisateur à hauts privilèges sur la machine. Après l'exécution de cette technique, le compte utilisateur fourni comme paramètre sera créé sur le SI et la connaissance $\mathcal{E}$ de l'attaquant sera enrichie avec ce nouveau compte. Comme nous l'avons indiqué dans la Section 4.3.1, cette technique illustre une situation où un élément de l'infrastructure de la victime n'est connu que de l'attaquant.

Technique	$\mathbf{t}_{1136}$: *Create account*
Tactique	*Persistence*
Description	Création d'un compte pour le service local s sur la machine $\mathbf{m}$.
Paramètres	$\mathbf{m}$, $\mathbf{u}$, $\mathbf{u}'$, k', ℓ', s
Préconditions	$(\mathbf{m}, \mathbf{u}) \in \mu$, $(\mathbf{u}, \mathsf{s}, \mathsf{k}, \texttt{high}) \in \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ avec $\mathcal{E}' = \mathcal{E}[\mathbf{m} \leftarrow (\lfloor \mathbb{S}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathbf{m}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathbf{m}} \rfloor_{\mathcal{E}} \oplus \{(\mathbf{u}', \mathsf{s}, \mathsf{k}', \ell')\}, \lfloor \mathbb{N}_{\mathbf{m}} \rfloor_{\mathcal{E}})]$
Variantes	

TABLE 4.15 – *Persistence* par *Create account*

4.5 Exemple d'une campagne d'attaque

Le modèle de l'attaquant présenté en Section 4.3, ainsi que la sémantique des techniques d'attaques présentée en Section 4.4, permettent de décrire formellement la progression des attaquants au cours de leurs campagnes. Nous avons pu reconstruire de manière semi-automatique les étapes des différentes campagnes des participants ayant contribué à l'expérimentation PWNJUTSU. La reconstruction de ce genre d'attaque est possible grâce à l'exploitation des journaux produits par les sondes système. Afin d'identifier les techniques mises en œuvre par l'attaquant, nous avons utilisé un outil[16] qui a permis d'extraire les EoI du jeu de données. Dans la suite de cette section, nous détaillons les actions entreprises par le participant `12`, noté `P12`.

L'attaque conduite par P12 a pu être reconstruite en 18 étapes, qui correspondent à son implémentation (procédure) de 8 techniques. Ces étapes sont considérées comme la partie dévoilée de ses TTP. La Figure 4.4 présente la séquence de techniques mises en œuvre par P12 : il s'agit de *modus operandi*[17]. Une routine semble également prendre forme sur la figure. Cela signifie que l'attaquant semble avoir tendance à exécuter régulièrement les mêmes techniques, dans des environnements similaires. Il est également donc possible d'identifier un motif dans l'attaque[18]. Le motif de P12 commence par un *scan* des ports de la première machine qui lui est présentée. Puis, il essaye d'exploiter une vulnérabilité afin de gagner un accès sur cette machine. Ensuite, il met en place un moyen de persistance en déposant une clé privée dans la configuration SSH. Enfin, il récupère les *flags* convoités et exécute une nouvelle fois cette séquence.

Les Tables 4.16 et 4.17 présentent chaque étape de l'attaque de P12 avec les paramètres de procédures de l'attaquant. Elles présentent également une corrélation avec les événements observés dans le jeu de données PWNJUTSU pour chacune de ces étapes. Par exemple, à l'étape `Step 2`, P12 met en œuvre depuis la machine `n12-gateway` la technique *Exploitation of Remote Services* sur le service `Apache Continuum`, qui écoute sur le port `8080` de la machine `n12-vm1`. Pour cela, il utilise l'*exploit* approprié trouvé dans la base de données publique ExploitDB. Cela produit une trace que l'on retrouve dans les captures réseau. Cette trace révèle une requête `HTTP POST` vers le chemin vulnérable de l'application web `/continuum/saveInstallation.action`.

16. https://github.com/wagga40/Zircolite
17. *operational flow*
18. *attack pattern*

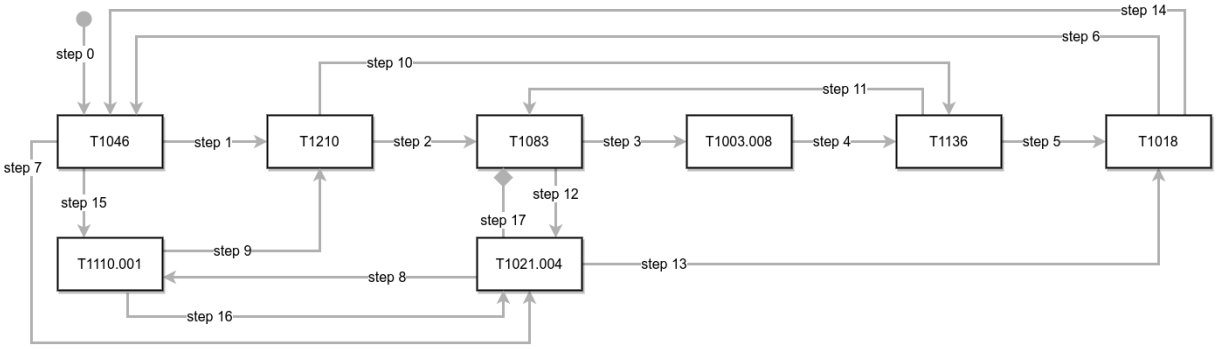

FIGURE 4.4 – *Modus operandi* de P12 au cours de l'expérimentation PWNJUTSU.

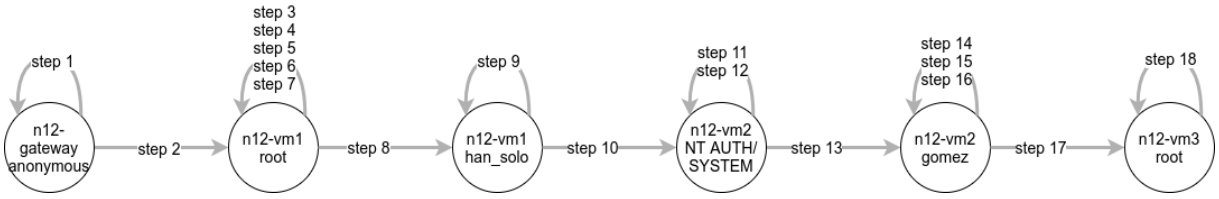

FIGURE 4.5 – Espace de propagation de P12 lors de sa campagne au cours de l'expérimentation PWNJUTSU.

À titre de comparaison, le rapport informel rédigé par P12 est disponible en Annexe A.2. Cet effort avait été demandé à chacun des participants afin de pouvoir le cas échéant comparer les actions annoncées par l'attaquant à celles perçues par le défenseur, comme nous l'avions fait dans le Chapitre 3.

De plus, notre modèle permet de représenter l'espace de propagation de cet attaquant au cours de sa campagne contre le SI de PWNJUTSU qui lui est dédié. La Figure 4.5 représente cet espace de propagation au travers des couples $(\mathbf{m}, \mathbf{u})$ contrôlés par l'attaquant. En d'autres termes, l'espace de propagation représente toutes les machines (et les utilisateurs) d'où l'attaquant peut exécuter des techniques. La découverte de cet espace est le Graal pour une équipe de réponse à incident qui se prépare à entrer dans une phase d'éradication de la menace.

Step 0	P12 got an initial access to `n12-gateway`.
Step 1	P12 performed network service scanning (T1046) from `n12-gateway` to `n12-vm1`.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{gateway}$, $\mathbf{u} = \texttt{anonymous}$ $\mathbf{m}' = \texttt{n12} - \texttt{vm1}$, $\Delta = \{\texttt{top1000portsnmap}\}$
Trace (net)	`21887 2021-05-09 20:07:49,695678 172.16.128.112 10.12.1.1 TCP 60 42548 → 445 [SYN] Seq=0 Win=1024 Len=0 MSS=1357`
Step 2	P12 exploited remote service (T1210) Apache Continuum (port 8080) from `n12-gateway` to `n12-vm1`.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{gateway}$, $\mathbf{u} = \texttt{anonymous}$ $\mathbf{m}' = \texttt{n12} - \texttt{vm1}$ $\mathtt{s} = \texttt{continuum}(\texttt{port} : 8080)$ $x = \texttt{EDB-ID} : 39945$ [19]
Trace (net)	`170511 2021-05-09 20:28:15,698503 172.16.128.112 10.12.1.1 HTTP 1411 POST /continuum/saveInstallation.action HTTP/1.1`
Step 3	P12 got a secret flag file (T1083) on $\texttt{n12} - \texttt{vm1}$.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{vm1}$, $\mathbf{u} = \texttt{root}$ $\mathtt{p} = \texttt{/opt/apache\_continuum/.../flag.txt}$
Trace (sys)	`May 9 20:28:54 n12-vm1 snoopy[1566]: [uid:0 sid:1309 tty:(none) cwd:/opt/apache_continuum/... filename:/bin/cat]: cat flag.txt`
Step 4	P12 got all credentials (T1003.008) of $\texttt{n12} - \texttt{vm1}$ OS.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{vm1}$, $\mathbf{u} = \texttt{root}$, $\mathtt{s} = \texttt{OS}_{\texttt{Linux}}$
Trace (sys)	`May 9 20:29:51 n12-vm1 snoopy[1569]: [uid:0 sid:1309 tty:(none) cwd:/opt/apache_continuum/... filename:/bin/cat]: cat /etc/shadow`
Step 5	P12 added a private key (from the root user's folder) for the user han_solo on the service SSH (T1136) on machine $\texttt{n12} - \texttt{vm1}$.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{vm1}$, $\mathbf{u} = \texttt{root}$, $\mathbf{u}' = \texttt{han\_solo}$, $\mathtt{k}' = \texttt{SSHprivatekey}$, $\ell' = \texttt{low}*$, $\mathtt{s} = \texttt{ssh}(\texttt{port} : 22)$
Trace (sys)	`May 9 20:36:16 n12-vm1 snoopy[1593]: [uid:0 sid:1309 tty:(none) cwd:/root filename:/bin/mv]: mv .ssh /home/han_solo/`
Step 6	P12 discovered remote system (T1018) `n12-vm2` from `n12-vm1` using ARP table.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{vm1}$, $\mathbf{u} = \texttt{root}$, $\mathbf{m}' = \texttt{n12} - \texttt{vm1}$
Trace (sys)	`May 9 20:39:45 n12-vm1 snoopy[1622]: [uid:0 sid:1309 tty:(none) cwd:/home/han_solo/.ssh filename:/usr/sbin/arp]: arp -an`
Step 7	P12 performed network service scanning (T1046) from `n12-vm1` to `n12-vm2` as user `root`.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{vm1}$, $\mathbf{u} = \texttt{root}$ $\mathbf{m}' = \texttt{n12} - \texttt{vm2}$, $\Delta = \{\texttt{top1000portsnmap}\}$
Trace (sys)	`May 9 20:44:15 n12-vm1 snoopy[1634]: [uid:0 sid:1309 tty:(none) cwd:/home/han_solo/.ssh filename:/usr/bin/nmap]: nmap -sS -vv -Pn -n 10.12.1.2-3`

TABLE 4.16 – Détails de la campagne d'attaque de P12 au cours de l'expérimentation PWNJUTSU 1/2.

Step 8	P12 got an access using SSH (T1021.004) service on `n12-vm1` as user `han_solo` with the previously added private key.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{gateway}$, $\mathbf{u} = \texttt{anonymous}$ $\mathbf{m}' = \texttt{n12} - \texttt{vm1}$, $\mathbf{u}' = \texttt{han\_solo}$ $\texttt{k}' = \texttt{SSHprivatekey}$, $\texttt{s} = \texttt{ssh}(\texttt{port} : 22)$
Trace (sys)	`May 9 20:44:37 n12-vm1 sshd[1636]: Accepted publickey for han_solo from 172.16.128.112 port 41134 ssh2`
Step 9	P12 bruteforce by guessing (T1110.001) the service `Tomcat/axis2` (port 8080) with the default username `admin` on $\texttt{n12} - \texttt{vm2}$.
Parameters	$\mathbf{m} = \texttt{n12} - \texttt{vm1}$, $\mathbf{u} = \texttt{han\_solo}$, $\mathbf{m}' = \texttt{n12} - \texttt{vm2}$, $\mathbf{u}' = \texttt{admin}$, $\texttt{s} = \texttt{tomcat}(\texttt{port} : 8080)$
Trace (net)	`182610 2021-05-09 21:02:25,553027 10.12.1.1 10.12.1.2` `HTTP 307 POST /axis2/axis2-admin/login HTTP/1.1`
Step 10	P12 exploited post authenticated remote service (T1210) `Tomcat` (port 8080) from $\texttt{n12} - \texttt{vm1}$ to $\texttt{n12} - \texttt{vm2}$.
Step 11	P12 added an account for the user `gomez` on the service SSH (T1136) on machine $\texttt{n12} - \texttt{vm2}$.
Step 12	P12 got a secret flag file (T1083) on $\texttt{n12} - \texttt{vm2}$.
Step 13	P12 got an access using SSH (T1021.004) service on `n12-vm2` as user `gomez`.
Step 14	P12 discovered remote system (T1018) `n12-vm3` from `n12-vm1` using ARP table.
Step 15	P12 performed network service scanning (T1046) from `n12-vm2` to `n12-vm3`.
Step 16	P12 bruteforce by guessing (T1110.001) the service `SSH` (port 22) with the username `root` on $\texttt{n12} - \texttt{vm3}$.
Step 17	P12 got an access using SSH (T1021.004) service on `n12-vm3` as user `root`.
Step 18	P12 got a secret flag file (T1083) on $\texttt{n12} - \texttt{vm3}$.

TABLE 4.17 – Détails de la campagne d'attaque de P12 au cours de l'expérimentation PWNJUTSU 2/2.

4.6 Bénéfices de l'expérimentation

4.6.1 Enregistrements du jeu de données

Les sondes déployées sur l'infrastructure PWNJUTSU (cf. Section 4.2.2) ont permis d'enregistrer des événements pendant les différentes campagnes d'attaques menées par les participants. Nous avons extrait ces événements du SIEM et les avons compilés. Ces données ont ensuite été préparées pour être téléchargeables depuis un site dédié [20]. Ce même site permet la consultation de ces données grâce à un moteur de recherche conçu pour les besoins du projet [21].

Dans ce jeu de données, pour chacun des 22 participants, les sondes ont produit les fichiers suivants :

— un fichier *JSON Lines* contenant les journaux d'événements système des trois machines vulnérables. Au total, cela représente plus de 16 millions d'événements (`n*-vm1` : 9.2M événements, `n*-vm2` : 50k événements, `n*-vm3` : 7.2M événements).

— des fichiers de capture PCAP contenant le trafic réseau brut des interfaces des machines de chaque instance. Ces captures ont ensuite été valorisées grâce à l'outil Zeek [22]. Au total, cela représente 172 Go de données brutes et 17 Go de résultats d'analyse Zeek, correspondant à 45 millions de lignes.

De plus, nous avons mis à disposition de la communauté un enregistrement dit "de référence" (`n99`), qui correspond à une instance de l'infrastructure supervisée, sans qu'aucune activité malveillante ne soit opérée.

4.6.2 Sondage auprès du panel

Immédiatement après leur participation à l'expérimentation, les membres du panel ont été interrogés au moyen d'un court questionnaire à propos de leurs habitudes opérationnelles et de leurs préférences tactiques. Les 22 participants ont répondu à ce questionnaire.

20. https://pwnjutsu.irisa.fr
21. Le moteur de recherche est le fruit du travail d'un groupe d'étudiants du Mastère Spécialisé en Cyber Sécurité de CentraleSupélec.
22. https://zeek.org

Pivoting

Dans un premier temps, nous avons cherché à comprendre quelles étaient les préférences des attaquants pour pivoter dans un réseau et ainsi progresser en profondeur dans le SI. Nous avons donc demandé aux membres du panel à quelles fonctionnalités ou protocoles ils avaient fait appel pour atteindre les machines `vm2` et `vm3` de l'expérimentation. Plusieurs réponses étaient autorisées. Il est important de préciser que chacun de ces protocoles ou fonctionnalités était présent dans le scénario de l'expérimentation. La Figure 4.6 présente leurs réponses.

Nous avons appris que les membres du panel préféraient majoritairement l'usage de deux fonctionnalités : **Proxy SOCKS** et **Port Forwarding**. Ces fonctionnalités étant toutes deux implémentées dans l'outil `SSH`. Cette information est intéressante à considérer lorsque, sur un SI opérationnel, il est nécessaire de faire des choix lors de l'implémentation de contremesures, telles que des machines agissant comme des bastions de sécurité, ou encore pour le développement de règles spécifiques pour ces fonctionnalités.

Influence du système d'exploitation sur le comportement

Nous avons également cherché à identifier l'influence du système d'exploitation de la machine ciblée sur le comportement de l'attaquant. Nous avons donc posé cette question au panel. **95% d'entre eux ont affirmé que leur comportement opérationnel différait en fonction du système d'exploitation rencontré**. Cela traduit un manque d'outils offensifs ou de procédures qui soient agnostiques du système d'exploitation de la machine ciblée. Dans un contexte de défense active (*i.e.*, *Cyber Deception*), cela permet de déployer des machines avec différents systèmes d'exploitation afin de leurrer l'attaquant et de l'inciter à dévoiler ses TTP.

Techniques préférées

Enfin, nous avons demandé aux membres du panel de classer, par préférence, les techniques qu'ils mettaient en œuvre au cours de leurs attaques. Cette question concernait les tactiques suivantes :

— *Lateral movement*
— *Credential access*
— *Privilege escalation*
— *Discovery*

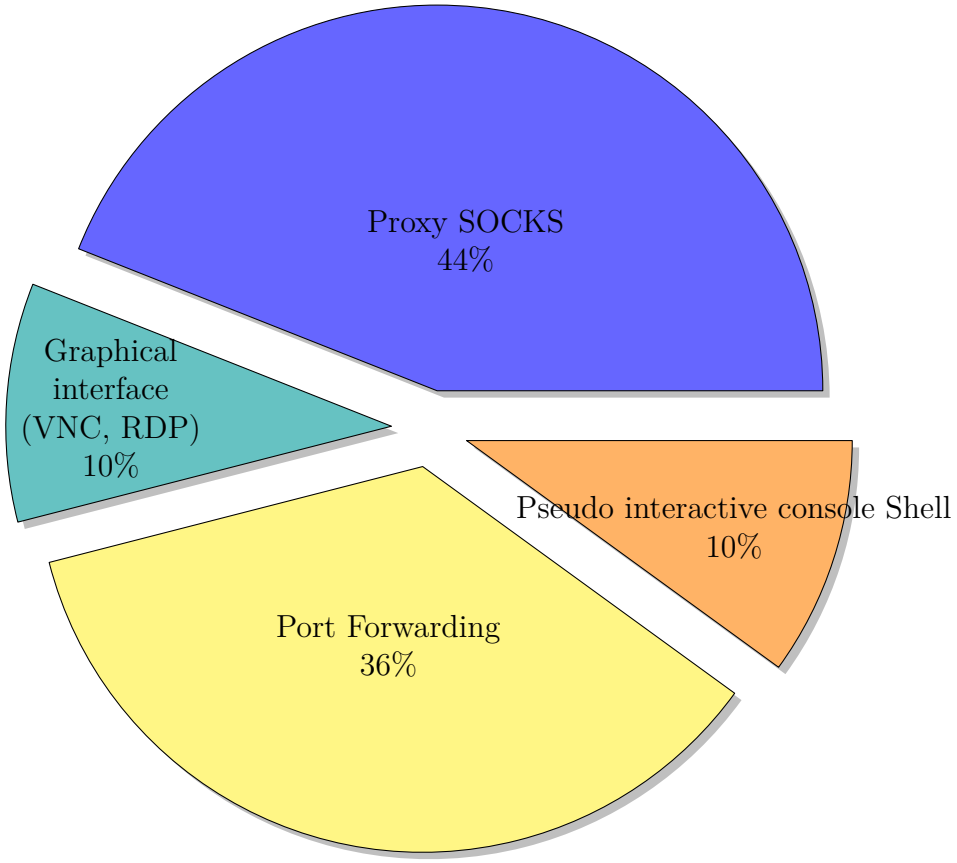

FIGURE 4.6 – Fonctionnalités et protocoles utilisés par les membres du panel de l'expérimentation PWNJUTSU pour atteindre les machines `vm2` et `vm3`

La Table 4.18 présente les réponses des membres du panel. Nous avons ensuite enrichi ces réponses avec ce que nous avons effectivement observé dans les événements enregistrés dans le jeu de données PWNJUTSU. Il est intéressant de noter que la technique préférée des participants pour satisfaire une intention de *Lateral movement* est `SSH`. Cela correspond à l'outil préféré identifié dans la Figure 4.6. Ces résultats sont d'une grande valeur puisqu'ils peuvent être considérés comme des listes de priorités pour l'implémentation de règles de détection ou pour l'application de contre-mesures.

Jeu de données PWNJUTSU

Le jeu de données PWNJUTSU est disponible au téléchargement sur le site `https://pwnjutsu.irisa.fr`. Il contient les traces (système et réseau) des 22 participants à l'expérimentation, ainsi que celles de l'environnement dit de "référence". Un moteur de recherche a également été mis en place afin de fournir un aperçu rapide des données collectées.

Techniques préférées pour des actions de *Lateral Movement* :

1. **[T1021.004] SSH**
   (préférée par 80% des participants, observée [a] pour 95%)
2. [T1021.002] SMB
3. [T1021.001] RDP
4. [T1021.006] WinRM
5. [T1021.005] VNC
6. [T1021.003] DCOM

a. n'importe quelle connexion acceptée (avec un mot de passe ou une clé privée) sur SSH.

Techniques préférées pour des actions de *Credential Access* :

1. **[T1552] Unsecured Credentials (Bash History, Creds in Files)**
   (préférée par 59% des participants, observée [a] pour 81%)
2. [T1003] OS Creds Dumping (LSASS, /etc/shadow, SAM)
3. [T1110] Bruteforce (Guessing, Spraying)
4. [T1557] MITM capture / AiTM (ARP, LLMNR)
5. [T1056.001] Input Capture (Keylogging)

a. n'importe quel événement lié au fichier `password_flag.txt` sur la machine `vm2` ou au fichier `.bash_history` sur la machine `vm3`

Techniques préférées pour des actions de *Privilege Escalation* :

1. **[T1078] Valid Accounts (default, domain, local)**
   (préférée par 68% des participants)
2. [T1548] Abuse Elevation Control Mechanism (Sudo, Bypass UAC)
3. [T1068] Vulnerability Exploitation for Privilege Escalation
4. [T1053] Scheduled Task/Job (At, Cron)
5. [T1055] Process Injection (ptrace, DLL, Proc Memory)

Techniques préférées pour des actions de *Discovery* :

1. **[T1046] Network Service Scanning**
   (préférée par 68% des participants, observée [a] pour 100%)
2. [T1016] System Network Configuration Discovery (ip, arp)
3. [T1018] Remote System Discovery (/etc/hosts, references in files)
4. [T1049] System Network Connections Discovery (netstat, ss)
5. [T1040] Network Sniffing

a. n'importe quelle tentative de connexion (`SYN`) sur le port TCP 1723, qui figure parmi le TOP 10 des ports les plus courants du scanner `nmap`. Ce port n'a pas de sens particulier dans le scénario.

TABLE 4.18 – Techniques préférées par les membres du panel PWNJUTSU.

4.7 Conclusion du quatrième chapitre

Les travaux présentés dans cette partie contribuent à étudier le *modus operandi* des acteurs opérant des campagnes d'attaques sophistiquées. L'ambition de l'expérimentation PWNJUTSU, dans laquelle 22 *Red Teamers* ont attaqué une instance similaire d'une infrastructure volontairement laissée vulnérable, est de produire un jeu de données qui puisse permettre l'étude des menaces sophistiquées. Alors qu'il n'y avait aucun langage ou aucun modèle qui permettait de décrire formellement les différents éléments manipulés au cours de cette expérimentation, notre contribution a été de proposer un cadre formel qui permet d'exprimer la progression d'un attaquant dans un réseau compromis ainsi que l'évolution de sa perception de l'environnement du SI.

Plus précisément, nous avons modélisé un attaquant selon l'état de sa connaissance et son contrôle sur le SI. Ce contrôle et cette connaissance évoluent à mesure que l'attaquant progresse dans le réseau. Nous avons également défini une sémantique opérationnelle des techniques d'attaques qui permettent de décrire cette évolution formellement. Cette sémantique nous a notamment permis de décrire formellement les différentes étapes de la campagne d'attaque conduite par un des participants à l'expérimentation PWNJUTSU. De plus, cette sémantique est suffisamment générique pour permettre de formaliser d'autres techniques d'attaques, d'autres scénarios ou encore pour permettre à des experts de la discipline de présenter des campagnes d'attaques observées.

L'autre contribution significative de ces travaux est la compilation des données collectées au cours de l'expérimentation PWNJUTSU dans un jeu de données, consistant en 16 millions d'événements et 172 Go de traces réseau. Ce jeu de données a été publié afin que l'ensemble de la communauté de la cybersécurité puisse en profiter, notamment à des fins de CTI ou pour mettre à l'épreuve les règles de détection des SIEM.

Ces travaux ont été publiés dans

Aimad Berady, Mathieu Jaume, Valérie Viet Triem Tong et Gilles Guette :
PWNJUTSU: A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns.
IEEE Transactions on Network and Service Management (TNSM), Special Issue on Recent Advances in Network Security Management, juin 2022.
DOI : 10.1109/TNSM.2022.3183476, HAL Id : hal-03694719

Conclusion générale et perspectives

L'objectif de cette thèse est de progresser dans la compréhension des menaces informatiques sophistiquées. Pour cela, nous avons focalisé nos travaux sur les Tactiques (intentions), les Techniques (moyens), les Procédures (manières) ainsi que sur les connaissances convergentes des adversaires, qui s'opposent lors d'une campagne d'attaque.

Nous apportons des éléments de réponse à cette problématique au travers de quatre contributions majeures :

- La définition du cycle de vie opérationnel d'un attaquant au cours d'une campagne, précisant les notions de phases et d'états. Nous confrontons cette lecture alternative des campagnes à deux attaques ayant été publiquement documentées.
- La spécification formelle de la notion de trace, de son origine dans les mains de l'attaquant à son exploitation par le défenseur. Nous explorons également des pistes d'optimisation de la collecte de ces traces en évaluant notre modèle sur un jeu de données public.
- La spécification formelle de la progression et de l'évolution de la connaissance d'un attaquant dans un SI compromis et la proposition d'une sémantique permettant de décrire les techniques d'attaques mises en œuvre dans le cadre d'une campagne.
- La création et la publication d'un jeu de données suite à la conduite d'une expérimentation inédite pour laquelle le panel était constitué de professionnels expérimentés de la discipline.

Phases opérationnelles d'une campagne d'attaque

Le modèle que nous proposons pour décrire les phases opérationnelles d'une campagne d'attaque permet de considérer deux aspects fondamentaux. Il s'agit de l'atteinte des objectifs finaux de façon répétée et la possible régression d'un attaquant suite à la survenue d'un événement imprévu dans la vie du SI ou suite une contre-mesure implémentée par le défenseur. À cela s'ajoute un état non technique où l'attaquant reconsidère son intérêt à

poursuivre la campagne d'attaque et donc à en supporter les coûts. Ce modèle a pu être instancié avec succès par deux campagnes d'attaques publiquement documentées.

La modélisation des cycles de vie des attaquants suivis au cours d'une campagne en considérant les phases et les états que nous avons définis dans cette thèse permettrait d'envisager des travaux dans le domaine de la *Cyber Deception*. Il s'agirait par exemple, pour un défenseur, de contraindre un attaquant à changer d'état pour l'amener à dévoiler ses positions dans le SI ou plus simplement pour augmenter considérablement le coût de l'attaque et donc le dissuader de poursuivre la campagne.

Visions confrontées des adversaires

La confrontation des visions de l'attaquant et du défenseur est possible parce qu'ils manipulent les mêmes objets du SI de la victime ; l'un pour attaquer le SI, l'autre pour y chasser l'intrus. Au cours des travaux présentés dans cette thèse, nous avons mis en avant l'importance de la trace, comme un point de jonction entre la connaissance de l'attaquant et celle du défenseur. Nous avons donc souligné l'importance, pour un défenseur, de disposer de traces pertinentes sur le SI de la victime, tout en limitant les faux positifs. Le modèle présenté dans cette thèse a pu être confronté à un jeu de données public.

Les perspectives offertes par ces travaux pourraient amener à travailler sur le calcul de similarité entre les graphes du défenseur et de l'attaquant afin d'en extraire des métriques qui pourront attester de la qualité des points de vues de l'attaquant et du défenseur. Ce genre d'expérimentations requiert l'implémentation de différents algorithmes de comparaison de graphes, afin de les évaluer. Par ailleurs, il serait également intéressant de chercher à ajuster dynamiquement les différents leviers à la disposition du défenseur et qui ont été définis au cours de ces travaux : la configuration des sondes, les règles de détection et la base de données d'IoC ; afin d'optimiser l'exhaustivité de l'espace de propagation de l'attaquant dans le SI. Pour atteindre cet objectif, il pourrait être nécessaire de définir une stratégie défensive qui prend en compte la notion de coût de déploiement et d'impact sur l'activité opérationnelle de l'entreprise de l'ajustement de ces leviers.

Connaissance et espace de propagation de l'attaquant

Cette thèse nous a également permis de progresser sur la spécification de l'évolution de la connaissance et de l'espace de propagation de l'attaquant au cours d'une campagne, à chaque exécution de procédure. Nous avons pu expérimenter cette approche permettant

de retracer les campagnes d'attaques grâce aux données collectées dans le cadre de l'expérimentation PWNJUTSU. Le formalisme défini permet également de décrire précisément l'architecture du SI de la victime.

Nous pensons que la formalisation des techniques, notamment celles présentées dans les matrices MITRE ATT&CK, pourrait, sur le long terme, contribuer à l'automatisation de certains processus coûteux mis en œuvre dans le cadre de la lutte contre les menaces modernes. Ainsi, il serait intéressant de proposer des méthodes automatiques de reconstruction des campagnes d'attaques en exploitant les journaux d'événements collectés dans le SI et en s'appuyant sur le formalisme défini dans cette thèse.

Jeu de données pour l'étude des menaces sophistiquées

L'expérimentation PWNJUTSU, forte d'un panel de 22 participants expérimentés, a permis de collecter un jeu de données que nous considérons comme étant pertinent pour analyser le comportement des attaquants progressant en phase de "propagation réseau". Ces données ont été compilées et rendues publiques au travers d'un site Internet dédié.

Le jeu de données PWNJUTSU pourrait être utilisé afin de comparer les différentes approches d'attaquants guidés par l'atteinte des mêmes objectifs. Par exemple, puisque chaque attaquant possède sa propre implémentation de ses procédures, pour les implémentations de procédures qui s'appuient sur des outils natifs des systèmes (*Living-Off-The-Land*), nous pensons qu'il serait intéressant d'étudier leurs caractéristiques afin de chercher à distinguer les procédures des attaquants de celles des administrateurs légitimes du SI. Le jeu de données PWNJUTSU pourrait également permettre d'identifier des outils offensifs ou des stratégies d'attaque par force brute (*i.e.*, *wordlists*) utilisés par les attaquants. Ainsi, il serait intéressant de caractériser et d'attribuer des artefacts de ces outils offensifs, également utilisés par les acteurs considérés comme des menaces sophistiquées.

Sources primaires

[1] ANSSI CERT-FR, « Panorama de la menace informatique 2021 », 8 mars 2022. adresse : https://www.cert.ssi.gouv.fr/cti/CERTFR-2022-CTI-002/.

[2] D. Schlette, M. Caselli et G. Pernul, « A Comparative Study on Cyber Threat Intelligence : The Security Incident Response Perspective », *IEEE Communications Surveys & Tutorials*, 2021. doi : 10.1109/comst.2021.3117338.

[3] AAA, « Does Methodological Superiority Lead the Way for SOF into Cyber Operations ? », in *Special Operations from a Small State Perspective*, Springer International Publishing, 2017. doi : 10.1007/978-3-319-43961-7_8.

[4] Rob Joyce, Chief, Tailored Access Operations, National Security Agency, *NSA TAO Chief on Disrupting Nation State Hackers*, 2016. adresse : https://www.youtube.com/watch?v=bDJb8WOJYdA (visité le 13/07/2022).

[5] L. Burita et D. T. Le, « Cyber Security and APT Groups », in *2021 Communication and Information Technologies (KIT)*, IEEE, 2021. doi : 10.1109/kit52904.2021.9583744.

[6] Y.-K. Kim, J. J. Lee, M.-H. Go et K. Lee, « Analysis of the Asymmetrical Relationships between State Actors and APT Threat Groups », in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, IEEE, 2020. doi : 10.1109/ictc49870.2020.9289506.

[7] R. Ross, « Managing Information Security Risk : Organization, Mission, and Information System View », 2011. adresse : https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=908030 (visité le 14/12/2021).

[8] M. K. Daly, « The Advanced Persistent Threat », Baltimore, MD : USENIX Association, 2009. adresse : https://www.usenix.org/legacy/event/lisa09/tech/slides/daly.pdf.

[9] B. Stojanović, K. Hofer-Schmitz et U. Kleb, « APT datasets and attack modeling for automated detection methods : A review », *Computers & Security*, 2020. doi : 10.1016/j.cose.2020.101734.

[10] E. Khaleefa et D. Abdulah, « Concept and difficulties of advanced persistent threats APT) : Survey », *International Journal of Nonlinear Analysis and Applications*, 2022. doi : 10.22075/ijnaa.2022.6230.

[11] Kaspersky. « Qu'est-ce qu'une menace persistante sophistiquée (APT) ? » (2018), adresse : https://www.kaspersky.fr/resource-center/definitions/advanced-persistent-threats (visité le 07/07/2022).

[12] M. Tatam, B. Shanmugam, S. Azam et K. Kannoorpatti, « A review of threat modelling approaches for APT-style attacks », *Heliyon*, 2021. doi : 10.1016/j.heliyon.2021.e05969.

[13] J. W. GREENERT. « Kill Chain Approach ». (2013), adresse : https://web.archive.org/web/20130613233413/http://cno.navylive.dodlive.mil/2013/04/23/kill-chain-approach-4/ (visité le 07/07/2022).

[14] E. M. Hutchins, M. J. Cloppert et R. M. Amin, « Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains », 2011.

[15] Command Five Pty Ltd, *Advanced Persistent Threats : A Decade in Review*, 2011.

[16] D. McWhorter, « APT1 : Exposing One of China's Cyber Espionage Units », 2013. adresse : https://www.mandiant.com/resources/apt1-exposing-one-of-chinas-cyber-espionage-units.

[17] P. Pols. « The Unified Kill Chain - Designing a Unified Kill Chain for analyzing, comparing and defending against cyber attacks ». (2017), adresse : https://www.unifiedkillchain.com/assets/The-Unified-Kill-Chain-Thesis.pdf (visité le 14/12/2021).

[18] MELANI :GovCERT. « APT Case RUAG - Technical Report ». (2016), adresse : https://www.govcert.ch/downloads/whitepapers/Report_Ruag-Espionage-Case.pdf (visité le 14/12/2021).

[19] F. Maymí, R. Bixler, R. Jones et S. Lathrop, « Towards a Definition of Cyberspace Tactics, Techniques and Procedures », in *2017 IEEE International Conference on Big Data*, 2017. doi : 10.1109/BigData.2017.8258514.

[20] B. E. Strom, J. A. Battaglia, M. S. Kemmerer et al., *Finding Cyber Threats with ATT&CK-Based Analytics*, 2017. adresse : https://www.mitre.org/publications/technical-papers/finding-cyber-threats-with-attck-based-analytics (visité le 14/12/2021).

[21] The MITRE Corporation. « MITRE ATT&CK™ ». (2022), adresse : https://attack.mitre.org/ (visité le 07/07/2022).

[22] P. Maynard et K. McLaughlin, « Big fish, little fish, critical infrastructure : An analysis of Phineas Fisher and the 'hacktivist' threat to critical infrastructure », 2020. arXiv : 2004.14360.

[23] F. J. Stech, K. E. Heckman et B. E. Strom, « Integrating Cyber-D&D into Adversary Modeling for Active Cyber Defense », in *Cyber Deception : Building the Scientific Foundation*, Springer International Publishing, 2016. doi : 10.1007/978-3-319-32699-3_1.

[24] M. R. Endsley, « Toward a Theory of Situation Awareness in Dynamic Systems », *Human Factors*, 1995. doi : 10.1518/001872095779049543.

[25] U. Franke et J. Brynielsson, « Cyber situational awareness - A systematic review of the literature », *Computers & Security*, 2014. doi : 10.1016/j.cose.2014.06.008.

[26] M. Pokladnik. « An Incident Handling Process for Small and Medium Businesses ». (2007), adresse : https://www.sans.org/white-papers/1791/ (visité le 05/07/2022).

[27] G. B. Dobson et K. M. Carley, « Towards Agent Validation of a Military Cyber Team Performance Simulation », in *Social, Cultural, and Behavioral Modeling*, Springer International Publishing, 2020. doi : 10.1007/978-3-030-61255-9_18.

[28] E. C. Thompson, « Threat Hunting », in *Designing a HIPAA-Compliant Security Operations Center*, Apress, Berkeley, CA, 2020, isbn : 978-1-4842-5608-4.

[29] S. Das, « Relevance of 'Red Teaming' in Strategic Military Decision-Making », *CLAWS Journal*, 2017. adresse : https://archive.claws.in/images/journals_doc/1162392053_SubhasisDas(1).pdf (visité le 08/07/2022).

[30] T. Olovsson, E. Jonsson, S. Brocklehurst et B. Littlewood, « Towards Operational Measures of Computer Security : Experimentation and Modelling », in *Predictably Dependable Computing Systems*, Springer Berlin Heidelberg, 1995. doi : 10.1007/978-3-642-79789-7_31.

[31] K. Bandla. « APT Notes ». (2022), adresse : https://github.com/aptnotes/data (visité le 22/06/2022).

[32] D. BIENSTOCK, M. DERR, J. MADELEY, T. MCLELLAN et C. GARDNER. « UNC3524 : Eye Spy on Your Email ». (2022), adresse : https://www.mandiant.com/resources/blog/unc3524-eye-spy-email (visité le 07/07/2022).

[33] THE DFIR REPORT. « Stolen Images Campaign Ends in Conti Ransomware ». (2022), adresse : https://thedfirreport.com/2022/04/04/stolen-images-campaign-ends-in-conti-ransomware/ (visité le 07/07/2022).

[34] C. YUCEL, I. CHALKIAS, D. MALLIS, E. KARAGIANNIS, D. CETINKAYA et V. KATOS, « On the Assessment of Completeness and Timeliness of Actionable Cyber Threat Intelligence Artefacts », in *Multimedia Communications, Services and Security*, Springer International Publishing, 2020. DOI : 10.1007/978-3-030-59000-0_5.

[35] M. J. M. TURCOTTE, A. D. KENT et C. HASH, *Unified Host and Network Data Set*, 2017. DOI : 10.48550/ARXIV.1708.07518. adresse : https://arxiv.org/abs/1708.07518.

[36] A. D. KENT, *Comprehensive, Multi-Source Cyber-Security Events*, Los Alamos National Laboratory, 2015. DOI : 10.17021/1179829.

[37] G. HO, M. DHIMAN, D. AKHAWE et al., *Hopper : Modeling and Detecting Lateral Movement (Extended Report)*, 2021. DOI : 10.48550/ARXIV.2105.13442.

[38] S. GIANVECCHIO, C. BURKHALTER, H. LAN, A. SILLERS et K. SMITH, « Closing the Gap with APTs Through Semantic Clusters and Automated Cybergames », in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Springer Int. Publishing, 2019. DOI : 10.1007/978-3-030-37228-6_12.

[39] A. APPLEBAUM, D. MILLER, B. STROM, C. KORBAN et R. WOLF, « Intelligent, automated red team emulation », in *Proc. of the 32nd Annual Conf. on Computer Security Applications*, ACM, 2016. DOI : 10.1145/2991079.2991111.

[40] R. RODRIGUEZ et J. L. RODRIGUEZ. « Security Datasets ». (2022), adresse : https://securitydatasets.com/ (visité le 07/07/2022).

[41] R. RODRIGUEZ. « APT29 activity from the ATT&CK evaluations ». (2020), adresse : https://github.com/OTRF/Security-Datasets/ (visité le 11/02/2022).

[42] R. Arantes, *Operationally Transparent Cyber (OpTC)*, 2021. doi : 10.21227/EDQ8-NK52. adresse : https://ieee-dataport.org/open-access/operationally-transparent-cyber-optc.

[43] M. M. Anjum, S. Iqbal et B. Hamelin, « Analyzing the Usefulness of the DARPA OpTC Dataset in Cyber Threat Detection Research », in *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*, ACM, 2021. doi : 10.1145/3450569.3463573.

[44] S. Myneni, A. Chowdhary, A. Sabur et al., « DAPT 2020 - Constructing a Benchmark Dataset for Advanced Persistent Threats », in *Deployable Machine Learning for Security Defense*, Springer International Publishing, 2020. doi : 10.1007/978-3-030-59621-7_8.

[45] A. Berady, V. Viet Triem Tong, G. Guette, C. Bidan et G. Carat, « Modeling the Operational Phases of APT Campaigns », in *CSCI 2019 - 6th Annual Conf. on Computational Science & Computational Intelligence*, 2019. doi : 10.1109/CSCI49370.2019.00023.

[46] R. Derbyshire, B. Green et D. Hutchison, « "Talking a different Language" : Anticipating adversary attack cost for cyber risk assessment », *Computers & Security*, 2021. doi : 10.1016/j.cose.2020.102163.

[47] S. M. Bellovin, « The Insider Attack Problem Nature and Scope », in *Insider Attack and Cyber Security : Beyond the Hacker*. Springer US, 2008. doi : 10.1007/978-0-387-77322-3_1.

[48] WikiLeaks. « Vault 7 : CIA Hacking Tools Revealed ». (2017), adresse : https://wikileaks.org/ciav7p1/ (visité le 08/07/2022).

[49] « Committee Releases Report Revealing New Information on Equifax Data Breach », United States House Committee on Oversight and Government Reform. (2018), adresse : https://republicans-oversight.house.gov/report/committee-releases-report-revealing-new-information-on-equifax-data-breach/ (visité le 17/12/2021).

[50] ANSSI. « Retour technique de l'incident de TV5Monde ». (2017), adresse : https://static.sstic.org/videos2017/SSTIC_2017-06-09_P09.mp4 (visité le 17/12/2021).

[51] ANSSI and Matt Suiche. « Lessons from TV5Monde 2015 Hack ». (2017), adresse : https://www.comae.com/posts/lessons-from-tv5monde-2015-hack/ (visité le 17/12/2021).

[52] A. Berady, M. Jaume, V. Viet Triem Tong et G. Guette, « From TTP to IoC : Advanced Persistent Graphs for Threat Hunting », *IEEE Transactions on Network and Service Management*, Special Issue on Latest Developments for Security Management of Networks and Services, 2021. doi : 10.1109/TNSM.2021.3056999.

[53] C. Hillier et T. Karroubi, *Turning the Hunted into the Hunter via Threat Hunting : Life Cycle, Ecosystem, Challenges and the Great Promise of AI*, 2022. doi : 10.48550/ARXIV.2204.11076.

[54] MITRE Corporation. « ATT&CK Evaluation ». (2019), adresse : https://attackevals.mitre-engenuity.org/ (visité le 08/10/2020).

[55] M. Monte, *Network attacks and exploitation*. John Wiley & Sons, 2015, isbn : 978-1-118-98723-0.

[56] OASIS Cyber Threat Intelligence. « STIX A structured language for cyber threat intelligence ». (2017), adresse : https://oasis-open.github.io/cti-documentation/stix/intro (visité le 16/03/2022).

[57] MITRE Corporation. « The MITRE Cyber Analytics Repository (CAR) ». (2018), adresse : https://car.mitre.org/ (visité le 16/03/2022).

[58] A. M. Porch. « Spoiling for a Fight : Hacking Back with the Active Cyber Defense Certainty Act ». (2020), adresse : https://www.amp.legal/images/AliceMPorch.65SDLRev467.pdf (visité le 14/12/2021).

[59] V. Mavroeidis et A. Jøsang, « Data-Driven Threat Hunting Using Sysmon », in *Proc. of the 2nd Int. Conf. on Cryptography, Security and Privacy*, ACM Press, 2018. doi : 10.1145/3199478.3199490.

[60] Y. Kurogome, Y. Otsuki, Y. Kawakoya et al., « EIGER : automated IOC generation for accurate and interpretable endpoint malware detection », in *Proc. of the 35th Annual Computer Security Applications Conf.*, ACM, 2019. doi : 10.1145/3359789.3359808.

[61] MITRE Engenuity. « APT29 Operational Flow ». (2019), adresse : https://attackevals.mitre-engenuity.org/enterprise/apt29/operational-flow (visité le 08/07/2022).

[62] Sudhakar et S. Kumar, « An emerging threat Fileless malware : a survey and research challenges », *Cybersecurity*, 2020. doi : 10.1186/s42400-019-0043-x.

[63] A. Berady, M. Jaume, V. V. T. Tong et G. Guette, « PWNJUTSU : A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns », *IEEE Transactions on Network and Service Management*, 2022. doi : 10.1109/tnsm.2022.3183476.

[64] I. Kovacevic, S. Gros et K. Slovenec, « Systematic Review and Quantitative Comparison of Cyberattack Scenario Detection and Projection », *Electronics*, 2020. doi : 10.3390/electronics9101722.

[65] ANSSI, « Recommandations de sécurité pour la journalisation des systèmes Microsoft Windows en environnement Active Directory », 28 jan. 2022. adresse : https://www.ssi.gouv.fr/guide/recommandations-de-securite-pour-la-journalisation-des-systemes-microsoft-windows-en-environnement-active-directory/.

[66] F. Pouget, M. Dacier et H. Debar, « White paper : honeypot, honeynet, honeytoken : terminological issues », *Rapport technique EURECOM*, 2003.

[67] S. Roy, N. Sharmin, J. C. Acosta, C. Kiekintveld et A. Laszka, *Survey and Taxonomy of Adversarial Reconnaissance Techniques*, 2021. doi : 10.48550/arXiv.2105.04749.

[68] S. L. Thomas et A. Francillon, « Backdoors : Definition, Deniability and Detection », in *Proceedings of the 21st International Symposium on Research in Attacks, Intrusions, and Defenses (RAID 2018)*, 2018. doi : 10.1007/978-3-030-00470-5_5.

[69] « An Examination of the EQUIFAX Cybersecurity Breach ». adresse : https://www.govinfo.gov/content/pkg/CHRG-115shrg28123/pdf/CHRG-115shrg28123.pdf.

[70] A. Berady, V. Viet Triem Tong, G. Guette et M. Jaume, « Caractérisation tactique d'un attaquant évoluant dans un réseau compromis », sér. Journée thématique du GT SSLR 2021 sur la sécurité des réseaux, 2021. adresse : https://gdr-securite.irisa.fr/wp-content/uploads/gtsslr21-reseau-AimadBerady-paper.pdf.

[71] P. N. Bahrami, A. Dehghantanha, T. Dargahi, R. M. Parizi, K.-K. R. Choo et H. H. S. Javadi, « Cyber Kill Chain-Based Taxonomy of Advanced Persistent Threat Actors : Analogy of Tactics, Techniques, and Procedures », *JIPS*, 2019. doi : 10.3745/JIPS.03.0126.

Sources secondaires

[72] P. Najafi, A. Mühle, W. Pünter, F. Cheng et C. Meinel, « MalRank : a measure of maliciousness in SIEM-based knowledge graphs », in *Proc. of the 35th Annual Computer Security Applications Conf.*, ACM, 2019. doi : 10.1145/3359789.3359791.

[73] K. Pei, Z. Gu, B. Saltaformaggio et al., « HERCULE : Attack Story Reconstruction via Community Discovery on Correlated Log Graph », in *Proc. of the 32nd Annual Conf. on Computer Security Applications*, ACM, 2016. doi : 10.1145/2991079.2991122.

[74] B. Burr, S. Wang, G. Salmon et H. Soliman, « On the Detection of Persistent Attacks using Alert Graphs and Event Feature Embeddings », in *NOMS 2020*, IEEE, 2020. doi : 10.1109/NOMS47738.2020.9110439.

[75] Z. C. Schreuders, T. Shaw, M. Shan-A-Khuda, G. Ravichandran, J. Keighley et M. Ordean, « Security Scenario Generator (SecGen) : A Framework for Generating Randomly Vulnerable Rich-scenario VMs for Learning Computer Security and Hosting CTF Events », in *2017 USENIX Workshop on Advances in Security Education (ASE 17)*, USENIX Association, 2017. adresse : https://www.usenix.org/conference/ase17/workshop-program/presentation/schreuders.

[76] S. Venkatesan, J. A. Youzwak, S. Sugrim et al., « VulnerVAN : A Vulnerable Network Generation Tool », in *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, IEEE, 2019. doi : 10.1109/milcom47813.2019.9021013.

[77] S. Liao, C. Zhou, Y. Zhao et al., « A Comprehensive Detection Approach of Nmap : Principles, Rules and Experiments », in *2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, IEEE, 2020. doi : 10.1109/cyberc49757.2020.00020.

[78] J. C. Acosta, A. Basak, C. Kiekintveld, N. Leslie et C. Kamhoua, « Cybersecurity Deception Experimentation System », in *2020 IEEE Secure Development (SecDev)*, IEEE, 2020. doi : 10.1109/secdev45635.2020.00022.

[79] S. D. D. Anton, D. Fraunholz et D. Schneider, *Investigating the Ecosystem of Offensive Information Security Tools*, 2020. doi : 10.48550/ARXIV.2012.08811.

[80] M. Pivarnikova, P. Sokol et T. Bajtos, « Early-Stage Detection of Cyber Attacks », *Information*, 2020. doi : 10.3390/info11120560.

[81] M. Rosso, M. Campobasso, G. Gankhuyag et L. Allodi, « SAIBERSOC : Synthetic Attack Injection to Benchmark and Evaluate the Performance of Security Operation Centers », in *Annual Computer Security Applications Conference*, ACM, 2020. doi : 10.1145/3427228.3427233.

[82] T. N. Sun, C. Teodorov et L. L. Roux, « Operational design for advanced persistent threats », in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems : Companion Proceedings*, ACM, 2020. doi : 10.1145/3417990.3420044.

[83] D. Everson et L. Cheng, « Network Attack Surface Simplification for Red and Blue Teams », in *2020 IEEE Secure Development (SecDev)*, IEEE, 2020. doi : 10.1109/secdev45635.2020.00027.

[84] P. Gao, F. Shao, X. Liu et al., *Enabling Efficient Cyber Threat Hunting With Cyber Threat Intelligence*, 2020. doi : 10.48550/ARXIV.2010.13637.

[85] T. Steffens, *Attribution of Advanced Persistent Threats*. Springer Berlin Heidelberg, 2020. doi : 10.1007/978-3-662-61313-9.

[86] Q. E. Hodgson, L. Ma et K. Marcinek, *Fighting shadows in the dark*. RAND, 2019, isbn : 978-1-977402-75-2.

[87] B. Ben, *The Intruder's View*. Oxford University Press, 2017, isbn : 978-0-19-068654-3.

[88] A. Kott, A. Swami et B. J. West, « The Fog of War in Cyberspace », *Computer*, 2016. doi : 10.1109/MC.2016.333.

[89] S. Roberts et R. Brown, *Intelligence-Driven Incident Response*. O'Reilly Media, 2017, isbn : 978-1-4919-3494-4.

Acronymes

ACIS *Automated Consumer Interview System.*

ANSSI Agence Nationale de la Sécurité des Systèmes d'Information.

APT *Advanced Persistent Threat.*

BCyP Bureau d'Etudes en Cyber sécurité et systèmes de Protection.

CAR *MITRE Cyber Analytics Repository.*

CERT Computer Emergency Response Team.

CTI *Cyber Threat Intelligence.*

DARPA Defense Advanced Research Projects Agency.

EDR *Endpoint Detection and Response.*

EoI *Event of Interest.*

IoC *Indicator of Compromise.*

IRSN Institut de Radioprotection et de Sûreté Nucléaire.

NIST *National Institute of Standards and Technology.*

OS *Operating System.*

PII *Personally Identifiable Information.*

RAT *Remote Access Trojan.*

SI Système d'Information.

SIEM *Security Information Event Management.*

SSL *Secure Sockets Layer*.

TTP Tactiques, Techniques et Procédures.

VPN *Virtual Private Network.*

NOTATIONS

$\mathbf{G}_A$ Graphe représentant les objets et composants impliqués dans une campagne d'attaque, tel que perçu par l'attaquant.

$\mathbf{G}_D$ Graphe représentant les objets et composants impliqués dans une campagne d'attaque, tel que perçu par le défenseur.

IoC Indicateur de compromission.

$\mathcal{A}$ Campagne d'attaque.

$\mathcal{S}$ Ensemble des secrets découverts par l'attaquant.

c Composant du SI de la victime.

y Propriété sur un objet.

ϕ Expression d'une configuration de sonde.

σ Configuration d'une sonde.

$\mathbf{D}_A$ Annuaire de l'attaquant.

$\mathbf{D}_D$ Annuaire du défenseur.

EoI Événement d'intérêt.

ev Événement.

e Exécution d'une procédure.

x *Exploit*.

τ Fonction associant un type à un rôle.

generate_IoC Fonction de génération d'indicateur de compromission.

$id_{\mathbf{x}}$ Identifiant unique d'une trace.

$\mathcal{E}$ Connaissance de l'environnement de la victime acquise par l'attaquant.

k Clé protégeant un compte utilisateur.

ℓ Niveau de privilège d'un compte utilisateur.

m Machine du SI de la victime.

μ Espace de propagation de l'attaquant.

$\perp$ Symbole désignant la contradiction utilisé ici pour indiquer l'absence de valeur.

$\mathbf{o}$ Objet.

$\mathcal{O}$ Ensemble des objets impliqués dans l'événement associé.

p Chemin sur le système de fichiers.

p Procédure.

$\models$ Relation de satisfaction entre une condition et un ensemble d'objets.

R_ϵ Ensemble des rôles pertinents pour un type d'événement donné.

$\mathbb{r}$ Rôle d'un objet.

r Règle de détection d'un SIEM.

$\mathsf{p}^\odot$ Symbole précisant que le fichier au chemin p contient un secret.

s Service.

$\mathbb{A}_\mathbf{m}$ Ensemble des comptes utilisateurs d'une machine $\mathbf{m}$.

$\mathbf{C}$ Ensemble des composants du SI de la victime.

E Ensemble d'événements.

$\mathbb{N}_\mathbf{m}$ Ensemble des machines présentes dans le voisinage réseau de la machine $\mathbf{m}$.

$\mathbf{O}$ Ensemble des objets.

$\mathbf{O}_A$ Ensemble des objets relatifs à l'attaquant.

$\mathbf{O}_D$ Ensemble des objets relatifs à la victime.

$\mathbb{P}_\mathbf{m}$ Ensemble des chemins du système de fichiers de la machine $\mathbf{m}$.

$\mathbb{R}$ Ensemble des rôles des objets.

$\mathcal{R}$ Ensemble des règles de détection d'un SIEM.

$\mathbb{S}_\mathbf{m}$ Ensemble des services fournis par une machine $\mathbf{m}$.

$\mathbf{X}$ Ensemble de *traces* induites par l'exécution d'une action.

$\mathbb{T}$ Ensemble des types d'objets existants.

$\mathfrak{E}$ Ensemble des types d'événements qui peuvent être observés.

$\mathcal{U}$ Ensemble des utilisateurs d'une machine autorisés à accéder à un fichier.

$\oplus$ Opérateur d'union entre deux graphes.

$\mathbf{t}$ *Technique*, au sens TTP du terme.

t *Timestamp.*

$\mathbf{x}$ Trace induite par l'exécution d'une action.

$\mathbf{TTP}$ Ensemble des *Techniques* existantes.

$\mathbf{TTP}_A$ Ensemble des *Techniques* maîtrisées par l'attaquant.

$\mathbf{TTP}_D$ Ensemble des *Techniques* maîtrisées par le défenseur.

$\mathbb{t}$ Type d'un objet.

ϵ Type d'événement.

$\mathbf{u}$ Utilisateur d'une machine.

Exploits Ensemble des fragments de codes permettant de déclencher une vulnérabilité sur un service.

GLOSSAIRE

Cyber Deception Manœuvre de défense active visant à tromper l'attaquant.

DevOps Pratique technique qui vise à améliorer le cycle de développement logiciel grâce à l'automatisation des tâches d'administration système liées à la qualification et au déploiement.

Threat Hunting Manœuvre tactique de chasse d'intrus opérée par les défenseurs dans un système d'information.

asset Anglicisme utilisé pour une entité intégrée dans un système d'information.

backdoor Porte dérobée, code informatique déposé par un attaquant sur un système permettant de maintenir un accès persistant à la ressource.

exploit Morceau de code qui tire profit d'une vulnérabilité logicielle ou plus globalement d'une faille de sécurité.

flag Dans un contexte de compétition de sécurité informatique, le drapeau capturé est une preuve témoignant de la progression d'une équipe.

hash Empreinte, résultat d'une fonction cryptographique de hachage.

malware Logiciel ou script malveillant, développé dans le but de nuire à un système d'information ou à y conduire des actions illégitimes.

webshell Script placé dans l'arborescence d'un site Web, qui permet à un attaquant d'exécuter des commandes sur le serveur ou d'en faire une passerelle vers le réseau.

wordlist Liste de mots compilés, collectés sur différentes sources de données et réutilisés pour inférer des chaînes de caractères utilisés sur système (mots de passe, chemins, nom de fichiers...).

rançongiciel Logiciel malveillant utilisé par les cybercriminels afin de prendre en otage les données d'un système d'information (chiffrement). La clé de déchiffrement est ensuite fournie en échange d'une rançon.

Annexe A

Annexes PWNJUTSU

A.1 Script de déploiement de l'infrastructure

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
require 'json'

file = File.read('/home/user/files/infra/vars.json')
creds = JSON.parse(file)

Vagrant.configure('2') do |config|
  config.vm.synced_folder('.', '/vagrant', type: 'nfs', disabled: true)
  # VM 1
  config.vm.define "vm1_" + "#{ENV['N_net']}" do |vm1|
    # Install
    vm1.vm.box = "metasploitable-ub1404-static"
    vm1.vm.provider :vmware_esxi do |esxi|
      esxi.esxi_hostname = creds["vcenter_hostname"]
      esxi.esxi_username = creds["vcenter_username"]
      esxi.esxi_password = creds["vcenter_password"]
      esxi.esxi_hostport = 22
      esxi.esxi_virtual_network = ["player" + "#{ENV['N_net']}"]
      esxi.esxi_disk_store = 'pwnjutsu2'
      esxi.guest_name = "n" + "#{ENV['N_net']}" + "-vm1"
      esxi.guest_memsize = '2048'
      esxi.guest_numvcpus = '1'
      esxi.guest_boot_disk_size = 50
      esxi.guest_guestos = 'ubuntu-64'
      esxi.guest_autostart = 'false'
```

```
27      end
28      # Configuration
29      vm1.vm.provision "ansible" do |ansible|
30        ansible.extra_vars = {
31          player: ENV['N_net']
32        }
33        ansible.verbose = "v"
34        ansible.playbook = "setups/setup_vm1.yml"
35      end
36    end
37
38    # VM 2
39    config.vm.define "vm2_" + "#{ENV['N_net']}" do |vm2|
40      # Install
41      vm2.vm.box = "rapid7/metasploitable3-win2k8"
42      vm2.vm.provider :vmware_esxi do |esxi|
43        esxi.esxi_hostname = creds["vcenter_hostname"]
44        esxi.esxi_username = creds["vcenter_username"]
45        esxi.esxi_password = creds["vcenter_password"]
46        esxi.esxi_hostport = 22
47        esxi.esxi_virtual_network = ["player" + "#{ENV['N_net']}"]
48        esxi.esxi_disk_store = 'pwnjutsu2'
49        esxi.guest_name = "n" + "#{ENV['N_net']}" + "-vm2"
50        esxi.guest_memsize = '4096'
51        esxi.guest_numvcpus = '1'
52        esxi.guest_boot_disk_size = 50
53        esxi.guest_autostart = 'false'
54      end
55      # Configuration
56      vm2.vm.provision "ansible" do |ansible|
57        ansible.extra_vars = {
58          player: ENV['N_net']
59        }
60        ansible.verbose = "v"
61        ansible.playbook = "setups/setup_vm2.yml"
62      end
```

```
63    end
64
65    # VM 3
66    config.vm.define "vm3_" + "#{ENV['N_net']}" do |vm3|
67      # Install
68      vm3.vm.box = "pwnjutsu/ubuntu20"
69      vm3.vm.provider :vmware_esxi do |esxi|
70        esxi.esxi_hostname = creds["vcenter_hostname"]
71        esxi.esxi_username = creds["vcenter_username"]
72        esxi.esxi_password = creds["vcenter_password"]
73        esxi.esxi_hostport = 22
74        esxi.esxi_virtual_network = ["player" + "#{ENV['N_net']}"]
75        esxi.esxi_disk_store = 'pwnjutsu2'
76        esxi.guest_name = "n" + "#{ENV['N_net']}" + "-vm3"
77        esxi.guest_memsize = '2048'
78        esxi.guest_numvcpus = '1'
79        esxi.guest_boot_disk_size = 50
80        esxi.guest_guestos = 'ubuntu-64'
81        esxi.guest_autostart = 'false'
82      end
83      # Configuration
84      vm3.vm.provision "ansible" do |ansible|
85        ansible.extra_vars = {
86          player: ENV['N_net']
87        }
88        ansible.verbose = "v"
89        ansible.playbook = "setups/setup_vm3.yml"
90      end
91    end
92
93    # Gateway
94    config.vm.define "gateway_" + "#{ENV['N_net']}" do |gw|
95      # Install
96      gw.vm.box = "debian10-static"
97      gw.vm.provider :vmware_esxi do |esxi|
98        esxi.esxi_hostname = creds["vcenter_hostname"]
```

```
99        esxi.esxi_username = creds["vcenter_username"]
100       esxi.esxi_password = creds["vcenter_password"]
101       esxi.esxi_hostport = 22
102       esxi.esxi_virtual_network = ["routers", "player" + "#{ENV['N_net']}"]
103       esxi.esxi_disk_store = 'pwnjutsu2'
104       esxi.guest_name = "n" + "#{ENV['N_net']}" + "-gateway"
105       esxi.guest_memsize = '512'
106       esxi.guest_numvcpus = '1'
107       esxi.guest_boot_disk_size = 20
108       esxi.guest_guestos = 'debian10-64'
109       esxi.guest_autostart = 'false'
110     end
111     # Configuration
112     gw.vm.provision "ansible" do |ansible|
113       ansible.extra_vars = {
114         player: ENV['N_net']
115       }
116       ansible.verbose = "v"
117       ansible.playbook = "setups/setup_gw.yml"
118     end
119   end
120 end
```

A.2 Script de *provisioning* de la machine vm1

```
1     - name: Configuration VM1
2     hosts: all
3     connection: local
4     pre_tasks:
5     - name: load vcenter informations
6       include_vars:
7         file: ../vars.json
8     - name: load flag info
9       include_vars:
10        file: ../flags/flag_{{player}}.json
11    tasks:
```

```
12        - name: set hostname & /etc/hosts
13          community.vmware.vmware_vm_shell:
14            validate_certs: "{{ vcenter_validate_certs }}"
15            hostname: "{{ vcenter_hostname }}"
16            username: "{{ vcenter_username }}"
17            password: "{{ vcenter_password }}"
18            vm_id: "n{{player}}-vm1"
19            vm_username: vagrant
20            vm_password: vagrant
21            vm_shell: /usr/bin/sudo
22            vm_shell_args: " hostnamectl set-hostname n{{player}}-vm1;sudo sed -i
    ↪  's/ubuntu/n{{player}}-vm1/g' /etc/hosts"
23
24        - name: set flag
25          community.vmware.vmware_vm_shell:
26            validate_certs: "{{ vcenter_validate_certs }}"
27            hostname: "{{ vcenter_hostname }}"
28            username: "{{ vcenter_username }}"
29            password: "{{ vcenter_password }}"
30            vm_id: "n{{player}}-vm1"
31            vm_username: vagrant
32            vm_password: vagrant
33            vm_shell: /usr/bin/sudo
34            vm_shell_args: "ls ; for i in {{flag1_home}}; do read user hash <<<
    ↪  $i; echo $hash | sudo tee /home/$user/flag.txt; sudo chown $user:users
    ↪  /home/$user/flag.txt; sudo chmod 400 /home/$user/flag.txt; echo \"-w
    ↪  /home/$user/flag.txt -p rwxa -k flag_vm1__home_${user}_flag.txt\" | sudo
    ↪  tee -a /etc/audit/audit.rules; done; for i in {{flag1}}; do read user group
    ↪  path hash <<< $i; echo $hash | sudo tee $path; sudo chown $user:$group
    ↪  $path ; sudo chmod 400 $path; path2=$(echo $path | tr '/' '_') ; echo \"-w
    ↪  $path -p rwxa -k flag_vm1_${path2}\" | sudo tee -a /etc/audit/audit.rules;
    ↪  done "
35
36        - name: config splunk
37          community.vmware.vmware_vm_shell:
38            validate_certs: "{{ vcenter_validate_certs }}"
```

```
39            hostname: "{{ vcenter_hostname }}"
40            username: "{{ vcenter_username }}"
41            password: "{{ vcenter_password }}"
42            vm_id: "n{{player}}-vm1"
43            vm_username: vagrant
44            vm_password: vagrant
45            vm_shell: /usr/bin/sudo
46            vm_shell_args: " /opt/splunkforwarder/bin/splunk enable boot-start
    ↪ --accept-license --answer-yes --no-prompt --seed-passwd REDACTED; sudo
    ↪ /opt/splunkforwarder/bin/splunk add forward-server 172.16.1.100:9997 -auth
    ↪ admin:REDACTED; sudo /opt/splunkforwarder/bin/splunk add monitor
    ↪ /var/log/auth.log -auth admin:REDACTED -follow-only True; sudo
    ↪ /opt/splunkforwarder/bin/splunk add monitor /var/log/audit/audit.log -auth
    ↪ admin:REDACTED -follow-only True; sudo /opt/splunkforwarder/bin/splunk add
    ↪ monitor /var/log/apache2/access.log -auth admin:REDACTED -follow-only True;
    ↪ sudo /opt/splunkforwarder/bin/splunk add monitor /var/log/apache2/error.log
    ↪ -auth admin:REDACTED -follow-only True; sudo
    ↪ /opt/splunkforwarder/bin/splunk add monitor
    ↪ /var/log/mysql-default/queries.log -auth admin:REDACTED -follow-only True;
    ↪ sleep 60"
47            wait_for_process: True
48            timeout: 1000
49
50        - name: set ip on eth0
51          community.vmware.vmware_vm_shell:
52            validate_certs: "{{ vcenter_validate_certs }}"
53            hostname: "{{ vcenter_hostname }}"
54            username: "{{ vcenter_username }}"
55            password: "{{ vcenter_password }}"
56            vm_id: "n{{player}}-vm1"
57            vm_username: vagrant
58            vm_password: vagrant
59            vm_shell: /usr/bin/sudo
```

```
60          vm_shell_args: " echo 'auto lo\niface lo inet loopback\n\nauto
    ↪ eth0\niface eth0 inet static\n  address 10.{{player}}.1.1/24\n  gateway
    ↪ 10.{{player}}.1.254\n' | sudo tee /etc/network/interfaces; sudo ifdown eth0
    ↪ && sudo ifup eth0"
61
62      - name: clean log
63        community.vmware.vmware_vm_shell:
64          validate_certs: "{{ vcenter_validate_certs }}"
65          hostname: "{{ vcenter_hostname }}"
66          username: "{{ vcenter_username }}"
67          password: "{{ vcenter_password }}"
68          vm_id: "n{{player}}-vm1"
69          vm_username: vagrant
70          vm_password: vagrant
71          vm_shell: /usr/bin/sudo
72          vm_shell_args: " cat /dev/null | sudo tee /var/log/audit/audit.log;
    ↪ cat /dev/null | sudo tee /var/log/auth.log"
73
74      - name: reboot vm properly
75        community.vmware.vmware_vm_shell:
76          validate_certs: "{{ vcenter_validate_certs }}"
77          hostname: "{{ vcenter_hostname }}"
78          username: "{{ vcenter_username }}"
79          password: "{{ vcenter_password }}"
80          vm_id: "n{{player}}-vm1"
81          vm_username: vagrant
82          vm_password: vagrant
83          vm_shell: /usr/bin/sudo
84          vm_shell_args: " reboot"
```

A.3 Consignes pour les participants de PWNJUTSU

Cher *hunter*,

Vous avez été sélectionné, pour vos compétences en *Red Teaming*, afin de participer à ce programme un peu particulier puisqu'il s'inscrit dans une **dynamique de recherche scientifique** portée par l'équipe CIDRE (*Confidentiality, Integrity,* Disponibilité, *Repartition*) de l'Inria (Institut national de recherche en sciences et technologies du numérique) et le BCyP (Bureau Cybersécurité et Protection Physique) de l'IRSN (Institut de radioprotection et de sûreté nucléaire). Nous vous invitons à lire attentivement les détails qui suivent avant de vous lancer.

Contexte

Pour se protéger des acteurs malveillants, les entreprises sont familières d'une alternative de défense dite "active" : la *Cyber Deception*. Cette pratique consiste à tromper l'adversaire afin de mieux le détecter pour ensuite le chasser et l'éradiquer ; ou encore de lui faire perdre du temps (augmentation du coût de l'attaque) afin de le dissuader. Parmi les techniques à la disposition des défenseurs, la plus célèbre et la plus mature à ce jour, est celle de pots de miel (*Honeypots*). Cela consiste à déployer dans un système d'information existant un dispositif factice imitant un système réel. Il peut être à basse interaction (imitation d'une simple application) ou à très haute interaction (imitation d'un réseau). Lorsqu'un utilisateur interagit avec ce dispositif, des alertes remontent à une entité de supervision (*i.e.*, un SOC) qui prendra les mesures adaptées à la situation. Cependant, lorsqu'une entité décide de mettre en place un pot de miel dans son infrastructure, elle doit faire face à deux problématiques récurrentes qui impactent le dispositif qui sera déployé : (1) celle de la représentativité et du réalisme ; et (2) celle de l'attractivité. Dans cette expérimentation, **nous cherchons à apporter des éléments de réponse à la question de l'attractivité d'un pot de miel**. C'est-à-dire l'ensemble des éléments techniques ou organisationnels qui amèneront un attaquant à s'intéresser à un composant du système d'information plutôt qu'à un autre, et en particulier au pot de miel plutôt qu'à un composant réel. Cette notion d'attractivité se concrétise sur deux axes : celui du positionnement au sein d'un réseau d'entreprise, qui est fortement dépendant de l'architecture du système d'information ; et, d'une manière plus générale, sur celui de son apparence extérieure, qui se traduira par la perception que l'attaquant aura du système auquel il fera face.

Déroulement

Après qu'un accès VPN vous ait été remis, vous pourrez vous connecter à la plateforme de jeu. Vous allez ensuite devoir compromettre des systèmes qui présentent volontairement des vulnérabilités. Le scénario se déroule en trois niveaux successifs, qui correspondent à trois machines différentes qu'il faudra compromettre. A noter qu'il vous sera nécessaire de "rebondir" par la machine du niveau *n* pour atteindre la machine du niveau *n+1*. **A chaque étape de votre exploration des *flags* pourront être découverts, notez les précieusement, dans l'ordre où vous les découvrirez, car vous devrez les restituer dans votre rapport final**. Ces *flags* témoigneront de votre cheminement au sein de l'environnement de jeu.

Il n'est pas nécessaire de relever la totalité des *flags* présents sur le système. Celui qui se trouve dans le *Working Directory* du service ou de l'utilisateur exploité suffit à témoigner de votre passage.

Tips

Adoptez une approche de *Red Teamer* : motivation et profondeur !

Des services sont présents sur plusieurs ports, n'oubliez pas la phase de *DISCOVERY*.

Les vulnérabilités sont simples et leur exploitation est évidente. Aucun piège n'est dressé.

Les machines à compromettre n'ont ~normalement~ pas accès Internet.

Lors de l'exploitation d'une vulnérabilité, dès que vous êtes en mesure d'exécuter des commandes, faites un `ls -la` (*resp.* `dir`) dans le répertoire courant, vous y trouverez le *flag* convoité.

Les *flags* sont dans fichiers `*flag.txt` et sont sous la forme `PWNJUTSU{48CgNMxrBJDVhgQx2NcJEbZcm7qa6ilt}`.

Rapport

Un seul rapport ne sera accepté par *hunter*. Contrairement aux programmes traditionnels de *Bug Bounty*, ce rapport ne devra pas préciser toutes les vulnérabilités découvertes, mais **uniquement le cheminement emprunté du point d'entrée au *flag* final**.

Par exemple :

Scan de ports sur l'IP 172.16.1.X

Exploitation du service Y sur le port Z

Découverte du *flag* AA

Élévation de privilèges par la technique BB

Découverte du *flag* CC

Rebond vers la machine DD grâce à EE

etc

La collaboration entre *hunters* introduirait des biais dans l'étude des données collectées, elle n'est donc pas autorisée.

Questionnaire

A la fin de votre action, un questionnaire vous sera transmis en réponse au rapport. Il contiendra quelques questions qui vous inviteront à préciser vos intentions et les décisions prises au cours de cette expérimentation.

Gratification

La grille des gratifications est la suivante (seul le niveau le plus haut est pris en compte) :

Low : un *flag* du niveau 1 ;

Medium : *Low* + un *flag* du niveau 2 ;

High : *Medium* + *final_flag* du niveau 3 ;

Critical : *High* + questionnaire final transmis (*Reward Bonus*).

Processus

Rapport transmis par le *hunter* : `Under review`

Flags confirmés par le *program manager* : `Accepted` , *Reward* 💰💰

Questionnaire envoyé par le *program manager* : `Ask for fix`

Questionnaire transmis par le *hunter* : `Resolved`

Reward Bonus pour atteindre le niveau de gratification *Critical* 💰💰

Outillage nécessaire

Libre à vous de choisir les outils de votre choix, sachez cependant que la distribution `Kali Linux` intègre tout ce dont vous pourriez avoir besoin.

Périmètre

Ne doivent être attaqués que les IP suivantes :

172.16.1.X

10.X.1.1

10.X.1.2

10.X.1.3
où X est votre numéro de joueur, attribué par le *program manager* à la remise des identifiants de VPN.

ATTENTION, CERTAINS SERVICES PRÉSENTENT DES VULNÉRABILITÉS DE TYPE DOS OU RISQUENT DE FAIRE *CRASH* LE SYSTÈME. A LA MANIÈRE D'UNE *RED TEAM* QUI PROGRESSE, VOUS DEVREZ FAIRE TOUT VOTRE POSSIBLE POUR GARDER L'INFRASTRUCTURE FONCTIONNELLE.

Hors périmètre

Les dispositifs de journalisation sur les systèmes et leurs configurations ne doivent pas être perturbés volontairement :

auditd

snoopy

Sysmon

SplunkForwarder

Données générées

S'agissant d'un projet à vocation scientifique, les données issues de cette expérimentation seront exploitées par les membres de l'équipe projet CIDRE (Inria Rennes).
Aucune information personnelle (adresse e-mail, pseudonyme, adresse IP...) ne sera associée au jeu de données, ni ne sera conservée.
Des communications scientifiques pourront avoir lieu pour présenter les conditions de réalisation de cette expérimentation ainsi que les résultats de cette étude, en se focalisant bien sûr sur l'ensemble des participants et **en aucun cas sur des individualités**.

Partant ?

Si vous êtes intéressé par cette aventure inédite, lancez-vous en envoyant un e-mail à l'adresse e-mail `pwnjutsu@vuln.site`. Nous vous transmettrons en réponse votre configuration VPN ainsi que votre numéro de joueur. Vous disposerez ensuite de `4 jours (96h)` pour réaliser votre action (c'est environ 48 fois plus de temps que nécessaire ^^).

Happy hunting!

A.4 Rapport informel du participant P12

Rapport PWNJUTSU

PWNJUTSU

SUBMITTED ON **MON, 10 MAY 2021**

REPORT DETAILS

SCOPE
pwnjutsu.vuln.site

SEVERITY
Critical

PAYLOAD
FINAL_PWNJUTSU{h3D9foSOHO0QAO5L2upvXAPZa1IEHgim}

TECHNICAL ENVIRONMENT
Kali

REWARD GRID

LOW	MEDIUM	HIGH	CRITICAL
€50	€200	€300	€400

BUG DESCRIPTION COMMENTS POST A RESPONSE

Bonjour et merci pour cet exercice très sympathique.

Voici la démarche que j'ai suivi :

1 - Scan nmap (1000 ports) a travers le VPN
2 - Découverte de plusieurs services
3 - Récupération de bannière et découverte de l'application continuum
3a - lancement d'un BF sur le port ssh (sans succès et pas très fonctionnel)
3b - recherche de vulnérabilités publiques sur continuum
4 - Utilisation du module metasploit pour exploiter avec succés la vuln continuum
5 - obtention d'un shell et environnement rapide de la machine

```
 PWNJUTSU{TvYSrSr6FwmMeXRVcUz6lkFQPZLBL2oj}


vagrant:$6$/Z97giSD$w1G/dmR09EnAh5hY47o/Hmb7ExAFHWwqfwsL1lvhpP41jDdF
dirmngr:*:18688:0:99999:7:::
han_solo:$1$6jIF3qTC$7jEXfQsNENuWYeO6cK7m1.:18688:0:99999:7:::
lando_calrissian:$1$Af1ek3xT$nKc8jkJ30gMQWeW/6.ono0:18688:0:99999:7:
boba_fett:$1$TjxlmV4j$k/rG1vb4.pj.z0yFWJ.ZD0:18688:0:99999:7:::
chewbacca:$1$.qt4t8zH$RdKbdafuqc7rYiDXSoQCI.:18688:0:99999:7:::
mysql:!:18688:0:99999:7:::
splunk:!:18688:0:99999:7:::

ls /home/*/*
/home/boba_fett/flag.txt
/home/han_solo/flag.txt
/home/lando_calrissian/flag.txt
/home/vagrant/flag.txt
/home/vagrant/linux.iso
cat /home/*/flag*
PWNJUTSU{dnBw8h3ksMkAmRahR856rprq4bE7TiD9}
PWNJUTSU{qDZyHEDAAPjXr1EESIefJCNKiF85FsCx}
PWNJUTSU{9lF8jhfdRkaPYKHXyMB1M8OWRez0ns2e}
PWNJUTSU{nqRI2xiCZDgvfYtyGRZRcMS4lhEuk6l2}

cd /root
```

```
cat flag.txt
PWNJUTSU{i7On8SiCUhsQpRWprWZ2nrmapDrRQXra}
```

6 - ajout d'une clé ssh (un peu de debug car mauvais format de clé et root disabled) puis connexion SSH en tant que han_solo
7 - nmap -sP directement depuis la machine compromise pour découvrir les machines suivantes
8 - 2 machines présentes d'après la table ARP mais la 10.12.1.3 ne semble pas joignable dans l'immédiat
9 - nmap sur la machine 10.12.1.2
10 - port 8080 open => low hanging fruits
11 - tomcat avec compte par défaut
12 - utilisation d'un module metasploit pour déployer un jar sur le tomcat => echec
13 - présence de axis2 => compte par defaut
14 - utilisation d'un module metasploit pour deployer une app sur axis2 et obtention d'un shell bind
15 - ajout d'un user local gomez et connexion ssh
16 - récupération d'un flag

```
PWNJUTSU{5nrQFdTGYmiuChzs7Q2BBawNnXOA4VyG}
```

17 - environnement => 172.16.1.100:9997 ?
18 - pivot ssh pour scanner la 10.12.1.3
19 - test des services open
80 => pas grand chose
21 => pas d'anonymous
22 => root/root and final flag

```
FINAL_PWNJUTSU{h3D9foSOHO0QAO5L2upvXAPZa1IEHgim}
```

A votre disposition pour tout complément d'infos.

BUG DESCRIPTION **COMMENTS** POST A RESPONSE

PWNJUTSU ON **MON, 10 MAY 2021 9:04:52 PM GMT+02:00**

NEW → UNDER REVIEW

Merci pour ce rapport.

PWNJUTSU ON MON, 10 MAY 2021 9:08:51 PM GMT+02:00

UNDER REVIEW → ACCEPTED

Nous confirmons votre cheminement jusqu'au niveau 3.

PWNJUTSU ON MON, 10 MAY 2021 9:09:00 PM GMT+02:00

ACCEPTED → ASK FOR FIX VERIFICATION

Voici le questionnaire pour conclure votre participation.

https://forms.office.com/r/yL8H6Ce9mU

PWNJUTSU ON MON, 10 MAY 2021 9:25:05 PM GMT+02:00

€400 and 50 pts rewarded

Bravo !

PWNJUTSU ON MON, 10 MAY 2021 9:25:15 PM GMT+02:00

ASK FOR FIX VERIFICATION → RESOLVED

Merci pour votre participation à l'expérimentation PWNJUTSU.

A.5 Arbre d'attaque de l'expérimentation

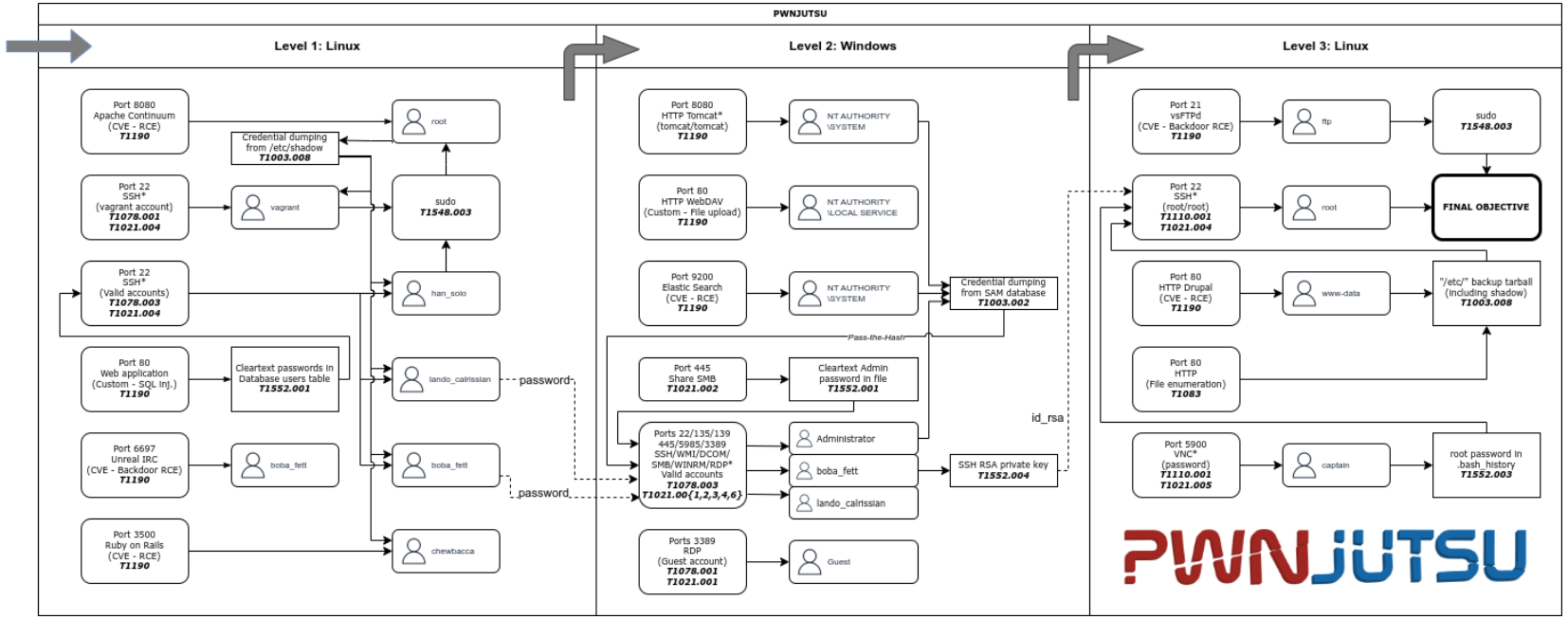

FIGURE A.1 – Arbre d'attaque du scénario de l'expérimentation PWNJUTSU proposé au panel.

DOCTORAT BRETAGNE LOIRE / MATHSTIC

CentraleSupélec

Titre : Comprendre les menaces sophistiquées.

Mot clés : APT, TTP, IOC, SIEM, Modèle, Expérimentation, Jeu de données, Connaissance

Résumé : Depuis quelques années, les experts du secteur de la cybersécurité observent une intensification d'attaques sophistiquées. Ces attaquants, qui agissent en profondeur, sont amenés à se propager dans les systèmes d'information de leurs victimes et à progresser furtivement vers leurs objectifs finaux. C'est pourquoi, des sondes doivent être disposées à des points stratégiques des systèmes d'information. Les entreprises ont également dû reconsidérer leurs mesures de cyberprotection, en mettant en place des systèmes centralisés de collecte de traces afin qu'elles puissent être exploitées dans l'éventualité d'une réponse à incident. Dans le cadre de cette thèse, les travaux que nous avons conduits ont permis de formaliser, d'un point de vue macroscopique, les différentes phases opérationnelles d'une campagne d'attaque. Parmi elles, celle de "propagation réseau" nous est apparue comme étant la plus pertinente pour détecter l'attaquant. En nous focalisant sur cette phase, nous avons ensuite mis en perspective les visions de l'attaquant et du défenseur dans le contexte d'une même campagne en confrontant leurs connaissances acquises au cours de leurs opérations respectives. Enfin, nous avons défini un modèle permettant une représentation de l'évolution de la connaissance de l'attaquant à propos du système d'information et de son espace de propagation. Ce modèle repose sur une sémantique, qui permet de spécifier formellement les techniques mises en œuvre par l'attaquant pour progresser vers ses objectifs finaux. Une expérimentation de grande envergure est venue renforcer cette contribution.

Title: Understanding sophisticated threats.

Keywords: APT, TTP, IOC, SIEM, Model, Experiment, Dataset, Knowledge

Abstract: In recent years, cybersecurity experts have observed an intensification of sophisticated threats. The attackers, acting in depth, are therefore led to propagate in their victim's information systems and progress stealthily toward their final objectives. Sensors must be deployed at strategic points in information systems. Companies also have had to reconsider their approaches to protect their information systems by setting up centralized systems to collect traces so that experts can exploit them if an incident occurs. In the context of this thesis, our work has made it possible to formalize the different operational phases of an attack campaign. Among them, "network propagation" appeared to be the most relevant for detecting the attacker. Then, we focused on this phase and put the visions of both the attacker and the defender into perspective. Finally, we defined a model allowing a representation of the attacker's propagation area and their knowledge evolution. This model wields semantics, allowing us to formally describe the techniques performed by the attacker to progress towards their final objectives. A large-scale experiment has reinforced this contribution.