Modélisation de systèmes de leurres complexes

Marwan Abbas Escribano

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2024IPPAS009

Thèse de doctorat

Modélisation de systèmes de leurres complexes

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom SudParis

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (EDIPP)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Palaiseau, le 12 avril 2024, par

Monsieur Marwan ABBAS ESCRIBANO

Composition du Jury :

Isabelle CHRISMENT Professeur, Université de Lorraine	Présidente
Ludovic MÉ Chercheur contractuel HDR, Inria	Rapporteur
Michaël HAUSPIE Maitre de Conférences HDR, Université de Lille	Rapporteur
Isabelle CHRISMENT Professeur, Université de Lorraine	Examinateur
Hervé DEBAR Professeur, Télécom SudParis	Directeur de thèse

Remerciements

Je tiens en premier lieu à remercier mon directeur de thèse Hervé Debar pour son accompagnement et ses conseils tout au long du déroulement de ma thèse, qui n'aurait pas abouti sans son aide précieuse. Je veux aussi remercier ma famille, qui m'a soutenu avant et pendant ma thèse, en toutes circonstances, sans jamais douter de moi mais en me poussant toujours de l'avant. A mes amis, qui se reconnaîtront, merci pour rien ! Et enfin, à Marguerite, merci pour tout, tout simplement.

Serment Doctoral d'Intégrité Scientifique

En présence de mes pairs. Parvenu à l'issue de mon doctorat en 2024, et ayant ainsi pratiqué, dans ma quête du savoir, l'exercice d'une recherche scientifique exigeante, en cultivant la rigueur intellectuelle, la réflexivité éthique et dans le respect des principes de l'intégrité scientifique, je m'engage, pour ce qui dépendra de moi, dans la suite de ma carrière professionnelle quel qu'en soit le secteur ou le domaine d'activité, à maintenir une conduite intègre dans mon rapport au savoir, mes méthodes et mes résultats.

Abstract

L'emploi de leurres et de techniques de déception pour la cybersécurité est très présent dans la littérature, même s'il reste relativement peu employé dans l'industrie malgré des progrès dans la virtualisation des systèmes et des architectures. Il est possible aujourd'hui de déployer des leurres pour détecter des attaquants et analyser leur procédés, mais ce déploiement se fait au niveau individuel, avec un approche restreinte : un leurre simulant un ou plusieurs services est positionné au sein d'un périmètre à défendre. Cette approche au cas par cas rend difficilement généralisable le déploiement et l'analyse de données issues des leurres.

Dans cette thèse, nous avons cherché à construire un modèle de leurre qui permet de décrire ceux-ci de façon claire et détaillée et à tester la faisabilité et l'efficacité des leurres bâtis selon celui-ci . Nous présentons en premier notre modèle ainsi que ses différentes composantes. Il se base en particulier sur la matrice MITRE ATT&CK qui nous permet une approche novatrice en construisant nos leurres à partir de possibilités d'attaque offertes aux attaquants, en simulant toute une cyberkillchain plutôt que de simples vulnérabilités. L'utilisation de ce standard permet une uniformisation de la description des systèmes de leurres que nous démontrant en décrivant plusieurs systèmes de leurres existant dans la littérature en nous basant de notre modèle. Nous avons ensuite cherché à vérifier la faisabilité de notre modèle en construisant un réseau de leurres en nous basant sur notre modèle.

Premièrement, nous avons construit un réseau de leurres en installant des services vulnérables, puis dans un second temps, en utilisant aussi des outils dédiés à la déception. Nous avons testé l'efficacité de ces leurres pour la capture de données d'attaque en les déployant dans deux contextes différents, premièrement chez un hébergeur, puis au sein de l'infrastructure de Télécom SudParis. Nous avons démontré que nos leurres sont efficaces à l'heure d'attirer des attaquants, en collectant des données cohérentes avec notre contexte de déploiement et celles trouvables dans la littérature. De plus, nous sommes capables d'obtenir des données d'analyse exploitables, ce qui démontre la capacité des leurres construits d'après notre modèle à détecter les attaquants et à permettre l'analyse de leurs procédés.

Abstract en Anglais

The use of decoys and deception techniques in cybersecurity is well documented in the literature, although it remains relatively little used in industry despite advances in system and architecture virtualization. It is possible today to deploy decoys to detect attackers and analyze their processes, but deployment is done on an individual level, with a restricted approach: a decoy simulating one or more services is positioned within a perimeter to defend. This case-by-case approach makes it difficult to generalize the deployment and analysis of decoy data.

In this thesis, we set out to build a decoy model that provides a clear and detailed description of decoys, and to test the feasibility and effectiveness of decoys based on this model. We first present our model and its various components. In particular, it is based on the MITRE ATT&CK matrix, which enables us to take an innovative approach by building our decoys on the basis of attack possibilities available to attackers, simulating an entire cyberkillchain rather than just vulnerabilities. The use of this knowledge base allows us to standardize the description of decoy systems, which we demonstrate by describing several decoy systems existing in the literature based on our model. We then sought to verify the feasibility of our model by building a decoy network based on our model.

First, we built a network of decoys by installing vulnerable services, and then, in a second phase, by also using tools dedicated to deception. We tested the effectiveness of these decoys in capturing attack data by deploying them in two different contexts, first at a hosting company, then within the Télécom SudParis infrastructure. We have demonstrated that our decoys are effective at attracting attackers, by collecting data consistent in volume and nature with our deployment context and what is found in the literature. What's more, we were able to obtain usable analysis data, demonstrating the ability of decoys built according to our model to detect attackers and enable analysis of their processes.

Table des Matières

Liste des Figures

Liste des Tables

Chapter 1

Introduction

1.1 Contexte

La numérisation croissante des activités et de la gestion des organisations publiques ou privées les rend fortement dépendantes de leurs systèmes d'information pour des éléments critiques de leur fonctionnement. Ceci contribue à faire des cyberattaques une menace plus importante que jamais, d'autant plus que leur impact porte désormais sur un périmètre élargi, concernant non seulement leurs opérations, mais aussi leur image, leur perception publique et leur crédibilité.

Au-delà de leur impact élevé, l'occurrence des cyberattaques est aussi en hausse. L'accroissement des profits qu'elles occasionnent attire des individus et des groupes plus organisés et motivés. Cette hausse de l'activité malveillante a été notablement accentuée pendant la pandémie du COVID-19, qui a eu un impact fort sur l'organisation du travail au sein de nombreuses structures, les rendant plus dépendantes du travail à distance et élargissant la surface d'attaque exposée. Il en résulte une hausse significative de l'activité malveillante mesurée pendant les années 2019 et 2020 [36, 50].

Au-delà des effets immédiatement perceptibles, l'impact des attaques réussies peut être durable. Il faut en moyenne 315 jours pour détecter une fuite de données après une attaque, et 23 jours pour retrouver un fonctionnement habituel après la compromission par un ransomware [62, 64]. D'après les réponses collectées dans cette enquête, chaque vol de données coûte en moyenne 4.35M de dollars américains, répartis au sein des opérations de détection, de signalement, de mitigation ainsi que dans la perte d'activité.

Les cyberattaques sont donc plus fréquentes, plus impactantes, et menées par des acteurs plus professionnels et motivés. A cause de ces éléments, la détection des attaques au plus tôt est un enjeu vital pour les gestionnaires des systèmes d'information (SI). L'obtention d'indices signalant les prémices d'une activité

malveillante est chaque fois plus important pour la mise en place d'une politique de cyberdéfense robuste.

Tandis que la détection d'attaque et la supervision de l'activité sur les SI devient un impératif, le périmètre de cette supervision s'élargit en même temps que la surface d'attaque exposée. De multiples canaux et technologies doivent être pris en compte, tels que les communications mobiles, le cloud ou l'émergence des "Objets connectés". En plus de ces aspects techniques, l'utilisation d'équipements ou de connexions privés par des employés effectuant leur travail à distance ou avec leur matériel présente de potentiels nouveaux accès aux attaquants qui doivent être surveillés.

Les acteurs malveillants ont appris à exploiter la dépendance des infrastructures envers des produits et des services externes. Pendant les dernières années, spécifiquement 2020 et 2021, les attaques "supply chain" (chaîne logistique) sont plus nombreuses et plus sophistiquées d'après un rapport de l'Agence Européenne pour la Cybersécurité [43]. Les attaques "supply chain" sont des opérations complexes où les acteurs malveillants contournent les mesures de sécurité en ciblant des organisations externes situées en amont de la chaîne d'approvisionnement, telles que celles qui fournissent des logiciels ou des services. Les organisations se retrouvent alors dans une situation où elles sont compromises par la simple utilisation d'outils dans lesquels elles ont confiance.

Un dernier élément qui joue aussi un rôle dans l'élargissement des surfaces d'attaque est l'essor croissant des menaces internes. Une menace interne est celle occasionnée par le membre d'une organisation possédant un accès légitime à un service, qu'il peut utiliser pour affecter l'intégrité, la disponibilité, la confidentialité ou le caractère non répudiable d'éléments du SI. Ces internes peuvent être directement membres des organisations, mais aussi être des tiers ayant accès aux systèmes tels que des prestataires de service. Des enquêtes auprès d'industriels montrent que les menaces internes ont augmenté de 44% entre 2020 et 2022 [49]. Les spécificités des risques occasionnés par les internes doivent être prises en compte lors de la mise en place de politiques de sécurité et de mécanismes de détection d'attaques.

Ces deux facteurs que sont les attaques "supply chain" et les menaces internes participent à un déplacement du périmètre de surveillance. Là où les politiques de sécurité pouvaient se baser sur un confiance des organisations envers leurs membres et leurs fournisseurs, ce n'est plus le cas. L'évolution des menaces nécessite aujourd'hui d'intégrer ces éléments au périmètre surveillé.

1.2 Déception

En réponse à l'accroissement et à l'évolution des menaces cyber sur les organisations, la mise en place de mécanismes de sécurité redondants et robustes

est une nécessité. En particulier, l'adoption de méthodes de défense en profondeur s'impose comme un impératif pour les organisations modernes qui font face à des acteurs avec des moyens et une motivation élevés. Inspirée des pratiques militaires, la défense en profondeur appliquée à la cybersécurité consiste en la mise en place d'un ensemble de systèmes de sécurité dont les périmètres d'action se chevauchent partiellement. L'objectif de cette architecture est de s'assurer qu'en cas de malfonctionnement ou de contournement par un attaquant d'un équipement ou d'une procédure de sécurité, son périmètre d'action sera compensé par la somme de ceux des autres systèmes.

Parmi les outils qui peuvent être employés pour la cybersécurité, la déception offre des possibilités intéressantes sur plusieurs champs. Tout comme la défense en profondeur, la déception est inspirée de pratiques militaires, et est définie dans la doctrine militaire française [19] telle que "Effet résultant de mesures visant à tromper l'adversaire en l'amenant à une fausse interprétation des attitudes amies en vue de l'inciter à réagir d'une manière préjudiciable à ses propres intérêts et de réduire ses capacités de riposte". On englobe donc dans la déception l'ensemble des opérations ayant pour but d'altérer la perception qu'a l'adversaire de l'état réel de nos intentions ou de la réalité des événements. Ce concept peut s'appliquer au domaine de la cybersécurité, et dans notre travail, nous nous référerons à la déception pour qualifier son application à l'informatique, et non pas son sens premier.

En cybersécurité, la déception garde son rôle principal qui est de chercher à altérer l'action de l'adversaire pour le faire agir de façon préjudiciable à son intérêt.

L'impact est double, les mesures en place agissant à la fois sur la perception qu'a l'adversaire d'une portion de son environnement, mais aussi sur sa prise de décision. Dans notre travail, nous ne considérons pas en profondeur les fondements psychologiques de la déception et nous nous restreignons aux problématiques techniques sur le sujet.

L'un des composants implémentés dans le but de décevoir des adversaires est le honeypot ou leurre, et c'est celui-ci que nous étudions dans notre travail. En pratique, il consiste en un ou plusieurs équipements informatiques mis en place non pas dans le but de présenter un intérêt légitime pour des utilisateurs, mais uniquement pour attirer l'attention de potentiels attaquants. Cela se fait généralement en introduisant volontairement ou en simulant des vulnérabilités dans la configuration ou l'implémentation des services exposés.

Leur principe assez général dans la déception permet de les employer dans une grande variété de situations. Les leurres s'adaptent selon leur conception et leur configuration à des environnements de recherche ou de production et ont deux principaux objectifs, la recherche et la détection d'attaques.

Les leurres peuvent servir dans le cadre de la recherche. Dans ce cas, leur objectif est d'attirer des attaquants dans le but de pouvoir collecter et analyser des

données à même d'éclairer sur leurs capacités et leurs outils. Grâce à cela, des méthodes d'attaque inconnues peuvent être identifiées, et les informations obtenues de cette façon peuvent venir renforcer la base de connaissances de la communauté et contribuer à l'amélioration de l'arsenal de la cybersécurité.

De façon complémentaire, les leurres peuvent servir à la détection pure dans des environnements de production. Dans ce cas là, le déploiement se fait dans le seul but de mettre en évidence la présence d'attaquants et non pas d'analyser leurs méthodes. En tant qu'outil dédié à la détection d'attaques, les leurres offrent des particularités qui leurs permettent de bien s'inscrire auprès des autres solutions dans le cadre d'une défense en profondeur efficace.

Dans un premier temps, la souplesse dans leur positionnement leur permet d'être déployés à tous les niveaux des réseaux à défendre. Ceci permet une surveillance adaptable, du plus général au plus particulier, sur toutes les positions du périmètre à défendre. Ce genre de possibilités offre l'opportunité de se prémunir contre certains types de menaces classiquement difficiles à monitorer, telles que celles occasionnées par les internes.

De par leur conception, on attend des leurres qu'ils présentent un taux de faux positifs plus faible que d'autres solutions de détection d'attaques. En effet, la mise en place d'un leurre se fait par définition via le déploiement d'un système dont l'intérêt réside dans sa seule exposition. Du fait de cette prémisse, le système ainsi exposé n'a pas d'usage légitime: tout accès à un leurre peut donc être considéré suspect du fait qu'il n'y à pas théoriquement d'accès possible pour un motif bénin. En pratique, les différentes interactions automatisées entre équipements d'un réseau, ainsi que l'activité automatique omniprésente en cas d'exposition au public nécessitent une première étape de traitement des alertes.

Ceci permet de s'affranchir d'une partie de la pollution des données obtenues dans le cadre des leurres d'analyse, et d'espérer obtenir un taux de faux positifs comparativement faible par rapport à d'autres technologies de détection d'attaques. Ces deux avantages, la qualité des données et la fiabilité des alertes, sont souvent considérés comme les principaux atouts des systèmes de détection d'attaque basés sur la déception, notamment par Nawrocki [39].

On voit donc que le déploiement de leurres peut se faire pour renforcer de façon efficace une architecture de défense en profondeur. Dans ses recommandations pour les organisations concernant la sécurité des systèmes d'information, le NIST (National Institute of Standards and Technology) recommande le déploiement de leurres dans le but "d'attirer les adversaires et détourner les attaquants des systèmes opérationnels" [46].

1.3 Motivation et Objectifs

La motivation de notre travail est la volonté de mettre en place des systèmes de déception plus efficaces et faciles à implémenter. Ceci permet d'améliorer les chances des défenseurs face aux attaquants dans le "jeu" de la cybersécurité en améliorant deux de leurs capacités : la collecte d'information et la détection d'attaques. Pour cela, il convient notamment de vaincre certaines réticences qui font obstacle au déploiement de systèmes de leurres dans beaucoup d'organisations. Nous espérons que notre travail aidera à rendre les systèmes plus compréhensibles et simplement implémentables.

A l'heure où les attaques sont plus fréquentes, plus impactantes et plus complexes, la *"course à l'information"* entre les attaquants et les défenseurs est au cœur des enjeux de la cybersécurité. Cette course apparaît du fait qu'il est bien plus difficile de se défendre contre des attaques inconnues que contre celles que l'on a déjà observé et analysé. Dans cette situation, les attaquants possèdent un avantage du fait de leur initiative, forçant les défenseurs à être dans une position réactive. De ce fait, tous les éléments permettant à ces derniers d'améliorer leurs procédés d'acquisition des informations et des connaissances se doivent d'être étudiés pour tenter de compenser ce déséquilibre.

Pour étudier les outils et les procédures des attaquants, il faut commencer par capturer des éléments d'information, dans des journaux d'observation au sein de systèmes compromis. Une fois que ceux-ci sont affectés, on peut envisager d'obtenir des informations à deux conditions. Premièrement, il faut être capable de détecter qu'une attaque a eu lieu. Il faut aussi que l'attaquant ait laissé des artefacts analysables après son passage et n'ait pas procédé à un nettoyage de ses traces. Ces deux conditions ne sont pas faciles à remplir, le temps de détection des attaques étant parfois très long, d'autant plus en cas d'opération discrète, et l'analyse pas toujours possible en cas d'effacement des traces.

Le problème avec cette méthode de travail est que l'acquisition d'information concernant les moyens et les procédures des attaquants est presque systématiquement rétrospective: on ne peut analyser que les données issues d'une compromission passée ou en cours grâce à des procédés d'analyse forensique. Ceci implique que l'acquisition d'information se fait en général à posteriori, et au prix de la compromission d'un système, compromission dont les effets peuvent aller de la nuisance au désastre.

C'est dans ce contexte que la motivation pour la mise en place de systèmes efficaces basés sur la déception et les leurres apparaît. Si les défenseurs d'un système d'information mettent en place un ensemble d'équipements factices destinés à attirer les attaquants, l'initiative de la course à l'information est renversée. Une attaque réussie contre un leurre ne signifie pas une compromission par l'attaquant mais une détection de l'attaquant et une acquisition d'informations pour les défenseurs.

Une autre motif pour le déploiement de leurres est la détection d'attaques. Dans la section suivante, nous décrivons plusieurs procédés servant à la détection des intrusions et des attaques au sein de systèmes d'information. Nous montrons aussi comment les systèmes basés sur les leurres permettent de compléter efficacement les mécanismes classiquement utilisés par les outils actuels. Par notre travail, nous cherchons à rendre plus simple et efficace le déploiement de systèmes de déception à cette finalité, permettant d'améliorer la sécurité des systèmes d'information.

Dans la pratique, les deux objectifs de détection d'attaques et d'analyse des procédés des attaquants se retrouvent dans le déploiement de honeypots. La distinction stricte entre honeypot de détection et d'analyse peut se faire théoriquement, mais a peu de sens en pratique.

Même si une motivation peut exister au vu des éléments précédents, le déploiement de leurres au sein des infrastructures de production des organisations se heurte souvent à des réticences de plusieurs ordres.

La première réticence naît de la difficulté à démontrer que l'ajout d'éléments dédiés à la déception au sein même d'un réseau de production ne dégrade pas la sécurité de celui-ci. Pour beaucoup d'utilisateurs, le principe même d'introduire volontairement dans leur périmètre un élément conçu pour être attaqué semble contre-intuitif. Du fait de méconnaissances sur le sujet ou d'absence de volonté d'investir des moyens pour mettre en place des systèmes de leurres sécurisés, beaucoup d'organisations ne perçoivent pas les avantages qu'elles en tireraient comme supérieurs aux inconvénients.

A ces réticences peuvent venir s'ajouter les différentes contraintes occasionnées par les cadres normatifs ou légaux qui peuvent peser sur les systèmes d'information au niveau de la conformité des systèmes de sécurité. En particulier dans l'Union Européenne, des problématiques liées à la gestion de données personnelles potentiellement collectées par des systèmes de leurres sont à prendre en compte. D'autres contraintes normatives relatives à l'incitation au crime [44] et à la responsabilité des administrateurs en cas d'attaque menée par rebond depuis le leurre peuvent freiner encore plus la mise en place de systèmes de déception.

De façon générale, la mise en place de leurres complexes pour la recherche ou pour l'utilisation en production au sein des réseaux des organisations nécessite de prendre en compte un ensemble de problématiques spécifiques. Ces problématiques telles que la discrétion et l'attractivité sont inhérentes au concept même de leurre, et il convient d'y répondre pour assurer le fonctionnement des systèmes. De plus, la complexité des systèmes augmente la possibilité qu'ils puissent contenir des erreurs de conception ou de configuration entraînant des vulnérabilités.

Une réponse à ces problèmes et un des principaux objectifs de notre travail est de concevoir un moyen de modéliser et de déployer des systèmes de leurres, couvrant tous les éléments de la déception, de l'architecture à la supervision, en se basant

sur le contexte de déploiement. Ceci dans le but de pouvoir déployer des leurres complexes simulant des pans entiers de réseaux avec plusieurs équipements et services. Pour rendre ce modèle attractif, nous cherchons à répondre aux trois problématiques évoquées, l'adaptation au contexte, la simplicité, la robustesse du déploiement et la capacité à détecter de l'activité malveillante.

1.4 Contributions

Notre réflexion a porté sur la conception de leurres attractifs et efficaces. Pour cela, nous nous sommes demandé comment nous pouvions intégrer le contexte de déploiement des leurres à leur conception. Nous nous sommes interrogés sur la possibilité d'intégrer dans notre conception les actions et le point de vue de l'attaquant sur nos leurres. Nous avons tout au long de notre travail gardé à l'esprit l'observabilité de nos leurres, et la façon dont les attaquants les percevaient, parfois différente de leur nature réelle. En basant notre modèle de leurres sur la matrice MITRE ATT&CK, nous nous intéressons dans la conception de nos leurres aux actions que les attaquants peuvent y effectuer.

Pour mener notre réflexion sur notre modèle de leurres, nous avons formulé trois hypothèses principales. Nous avons cherché à les évaluer via les différentes étapes de notre travail. Ces hypothèses sont les suivantes:

H1: Les leurres sont un complément efficace aux méthodes traditionnelles de détection d'attaque, notamment pour détecter certains types d'attaques qui peuvent échapper à ceux-ci.

H2: Un leurre est plus attractif pour les attaquants s'il donne l'apparence d'être proche des éléments réels de son contexte de fonctionnement, techniquement et du point de vue de l'architecture.

H3: La Matrice MITRE ATT&CK peut servir de base de travail pour l'analyse des possibilités d'action des attaquants préalable au déploiement de leurres. Elle est suffisamment exhaustive pour couvrir la majorité des cas d'attaques qui ciblent les leurres et est régulièrement mise à jour.

La principale contribution de notre travail est la mise en place d'un modèle de leurres complet, prenant en compte des problématiques d'infrastructure et de supervision et permettant la construction de leurres pour chercher à vérifier nos hypothèses H2 et H3. L'objectif de notre modèle est de décrire des leurres dédiés à la détection et à l'analyse d'attaques. Celui-ci nous offre les moyens de construire des leurres complexes basés sur plusieurs équipements virtualisés. Ceux-ci présentent aux attaquants différentes possibilités préalablement identifiées et conformes aux standards de comportement des attaquants proposés par MITRE. Cette prise en compte des comportements des attaquants est une approche innovante. Elle nous permet d'adapter au mieux les leurres déployés au contexte des différents systèmes d'information et aux spécificités des

organisations qui les accueillent. En plus de la conception du modèle, nous avons travaillé à l'implémentation d'un réseau de leurres en nous basant sur celui-ci. Nous avons pu le déployer pour tester sa capacité à attirer et à collecter des informations sur les attaquants, et à pouvoir vérifier les modalités de notre hypothèse H1. Dans cette implémentation, nous cherchons à moderniser différents travaux effectués sur le leurrage. En particulier, nous utilisons des outils plus récents, tels que la virtualisation par conteneurs et la taxonomie MITRE ATT&CK. Grâce à cela, nous pouvons mettre en place plusieurs expériences. Celles-ci nous permettent de comparer nos résultats avec près de deux décennies d'état de l'art, et de relever que le comportement des attaquants opportunistes ne semble pas avoir varié dans le temps.

1.5 Déroulé du Manuscrit

Notre travail est divisé en plusieurs chapitres.

Le Chapitre 2 présente un état de l'art sur la déception et les leurres. Pour cela, nous présentons un historique de concept et de son avancement entre les années 1980 et nos jours, présentant l'évolution de l'intérêt de la recherche et du public pour le concept de la déception. Nous définissons les principes et les concepts communément utilisés pour le travail autour des leurres, tels que le niveau d'interaction, la finalité, ainsi que des concepts importants pour notre travail tels que les vulnérabilités, la virtualisation ou la détection d'attaques.

Le Chapitre 3 présente le travail théorique autour de la conception de notre modèle de leurre. Dans celui-ci, on discute de notre modèle en particulier, des différents éléments qui ont mené à sa mise en place et des considérations à prendre en compte lors de son implémentation dans un système réel.

Le Chapitre 4 traite de l'implémentation et des choix d'architecture qui ont été effectués dans le but de réaliser un système de leurres déployables se basant sur notre modèle de leurre. Dans ce chapitre, on discute de l'architecture virtualisée de notre leurre, ainsi que des différents outils mis en place pour la supervision et la collecte de données propres à chaque composant de notre réseau de leurres.

Le Chapitre 5 porte sur les différentes expérimentations effectuées grâce au leurre implémenté comme détaillé dans le chapitre précédent. Pour chaque expérience menée, on détaille dans ce chapitre les contextes spécifiques de déploiement, les procédures adoptées, ainsi que les résultats collectés et les outils mis en place pour leur analyse.

Le Chapitre 6 porte sur l'analyse et la discussion autour des résultats collectés pendant les phases d'expérimentation décrites dans le chapitre précédent. Plusieurs analyses quantitatives et qualitatives sont présentées concernant

l'activité des attaquants collectée au sein de nos leurres, et celles-ci sont comparées entre elles ainsi qu'avec plusieurs expériences similaires trouvables dans la littérature.

Chapter 2

État de l'Art

2.1 Historique

2.1.1 La Déception avant l'informatique

La déception est un concept ancien dont l'origine se situe loin de la cybersécurité. Dans la nature, des espèces animales ou végétales adoptent des comportements ou des apparences ayant pour but de les faire passer pour autre chose que ce qu'elles sont. Ceci leur permet en général d'échapper à leurs prédateurs. Parmi les différentes stratégies rencontrées, une en particulier fait écho à notre démarche. L'automimétisme est le procédé par lequel un animal attire l'attention sur une partie moins vitale de son corps dans le but d'y focaliser de potentiels prédateurs. Ceci lui permet d'augmenter ses chances de survie en cas d'attaque. Par exemple, il peut s'être doté de caractères physiques qui donnent l'impression que sa tête est positionnée au niveau de sa queue comme dans l'exemple en figure 2.1. Ainsi, il attire prioritairement l'attention de potentiels attaquants vers une section peu ou pas critique de son corps.

Assez naturellement, ce genre de comportement a été adopté par les humains, en premier lieu dans le cadre des opérations militaires dans lesquelles le subterfuge joue une rôle essentiel. Un des exemples les plus documentés et massifs de l'usage de la déception est l'opération Fortitude menée par les Alliés pendant la seconde guerre mondiale. Cette opération avait pour objectif de brouiller l'information concernant le débarquement en Normandie en faisant croire que d'autres débarquements à d'autres endroits étaient en préparation. Pour cela, en plus des outils traditionnels de l'espionnage, de nombreux leurres sont utilisés pour faire croire à des rassemblements de troupes, sous des formes très diverses. Ainsi, des leurres sont positionnés à portée d'observation des forces allemandes, tels que des faux chars gonflables, ou des maquettes d'avions en bois [12]. On retrouve par la suite plusieurs applications à la cybersécurité entre la fin des années 80 et les années 2000. Ces approches sont souvent simples et partielles, mais innovantes

Figure 2.1: Exemple d'automimétisme. Strymon melinus donne l'impression que sa tête se situe au niveau du bout de ses ailes.

pour leur époque. Beaucoup d'entre elles préfigurent des éléments de réflexion qui seront adoptés par les systèmes modernes de leurrage.

2.1.2 Premiers leurres informatiques

En ce qui concerne la cybersécurité, le premier exemple d'usage de déception documenté peut se retrouver dans l'opération menée par Clifford Stoll en 1986 et rendue célèbre par le livre "The Cuckoo's Egg" [67]. Dans celui-ci, l'auteur explique comment il est parvenu à piéger un attaquant en monitorant précisément son activité sur un réseau sensible de son laboratoire à Berkeley. Il a mis en place un système de déception à proprement parler en créant sur le système informatique de faux départements de son laboratoire. Grâce à cela, il a été en mesure de détourner l'attention loin des éléments sensibles et finalement d'identifier l'attaquant. Dans cet exemple, la déception en informatique apparaît pour la première fois de façon documentée et on y retrouve déjà les spécificités propres au leurrage.

Une autre apparition précoce de déception en cybersécurité se trouve dans le travail de Bill Cheswick en 1997 [15]. L'approche de Stoll est opportuniste, son idée de mise en place d'un système leurrant l'attaquant intervenant après la découverte de celui-ci. A l'inverse, l'intention de Cheswick est dès le début d'exposer un système dans le but de collecter de l'information sur les attaquants. On retrouve donc dans son système un principe fondamental du leurre: son système n'a pas d'autre but que d'être attaqué et compromis.

Après ces premières études conceptuelles, une solution est développée par Fred Cohen en 1998, le "Deception Toolkit" [16]. Ce produit développé en C et en Perl propose via une dizaine de scripts de mettre en place facilement un système

de déception émulé. Il fonctionne en écoutant le trafic sur un certain nombre de ports et en répondant aux requêtes tout en journalisant les opérations, permettant d'agréger des informations sur les attaquants potentiels. Il propose avec cet outil le premier système de déploiement de leurres automatisé.

Pendant le développement de sa solution, Cohen a été confronté à une question fréquemment soulevée par les administrateurs dubitatifs face aux solutions de déception encore aujourd'hui. N'est-il pas aussi coûteux en ressources de développer une solution simulant fidèlement des vulnérabilités que de corriger les vulnérabilités existantes dans un périmètre à défendre ? A cette interrogation légitime, Cohen répond que l'intérêt pour les défenseurs d'utiliser des systèmes de déception ne réside pas uniquement dans leur capacité à capturer les actions des attaquants. Pour lui, ils contribuent à augmenter le niveau d'incertitude chez les attaquants, et donc à potentiellement les dissuader d'attaquer en exploitant leur crainte d'être pris, formant une ligne de défense autant technique que psychologique.

Après la publication de ce logiciel, plusieurs solutions commerciales ont émergé, reprenant le concept de déploiement automatisé de systèmes émulés. Souvent, des contributions nouvelles, telles que le support de protocoles, implémentations propriétaires, ou l'émulation de réseaux comprenant plusieurs machines sont apportées. On peut citer par exemple NetFacade ou BackOfficer Friendly [65]. De nombreuses solutions libres et commerciales sont répertoriées par Nawrocki [39].

Une autre étape importante dans l'étude de la déception en cybersécurité est la création du "Honeynet Project" [60] en 1999. Ce groupement de chercheurs, toujours actif, a pour objectif l'étude des attaques et des menaces en ligne, ainsi que le développement d'outils pour les contrer. Le projet de ce groupement est d'entretenir de façon continue un écosystème de leurres distribués sur plusieurs infrastructures exposées au public, dans le but d'obtenir "en direct" des tendances et des procédés d'attaque émergents. Un autre projet similaire mais aujourd'hui abandonné peut se retrouver à l'échelle européenne avec "Leurre.com" [37], actif entre 2003 et 2008.

En 2002, un leurre permet pour la première fois de détecter et d'analyser un exploit encore inconnu [39] sur une vulnérabilité répertoriée. La même année, Lance Spitzner publie un ouvrage de référence [65] sur le sujet des leurres dans lequel il propose une définition et un ensemble de classifications toujours acceptés et employés aujourd'hui, que nous décrivons en détail dans la section suivante.

2.1.3 Regain d'intérêt

Plusieurs outils nous permettent d'estimer l'évolution de l'intérêt pour les leurres dans la publication scientifique ainsi que dans la communauté en général. Concernant l'intérêt général, un indicateur intéressant peut être l'observation des

tendances de recherche sur le site de recherche en ligne Google. Bien que ce ne soit pas forcément une métrique extrêmement significative, elle nous permet d'inférer sur l'intérêt général porté par la population envers certains mots-clés et concepts. L'outil Google Trends [45] nous permet facilement d'obtenir des données concernant l'intérêt pour le mot-clé "honeypot" dans le domaine de la sécurité informatique entre 2004 et aujourd'hui dans le monde entier. En exportant les données en CSV et en appliquant une moyenne centrée au nuage de points, on obtient la figure 2.2 Bien que ces résultats brossent un portrait assez grossier des tendances, on peut remarquer un regain progressif d'intérêt pour le sujet à partir de 2010 après une baisse significative entre 2004 et 2010.

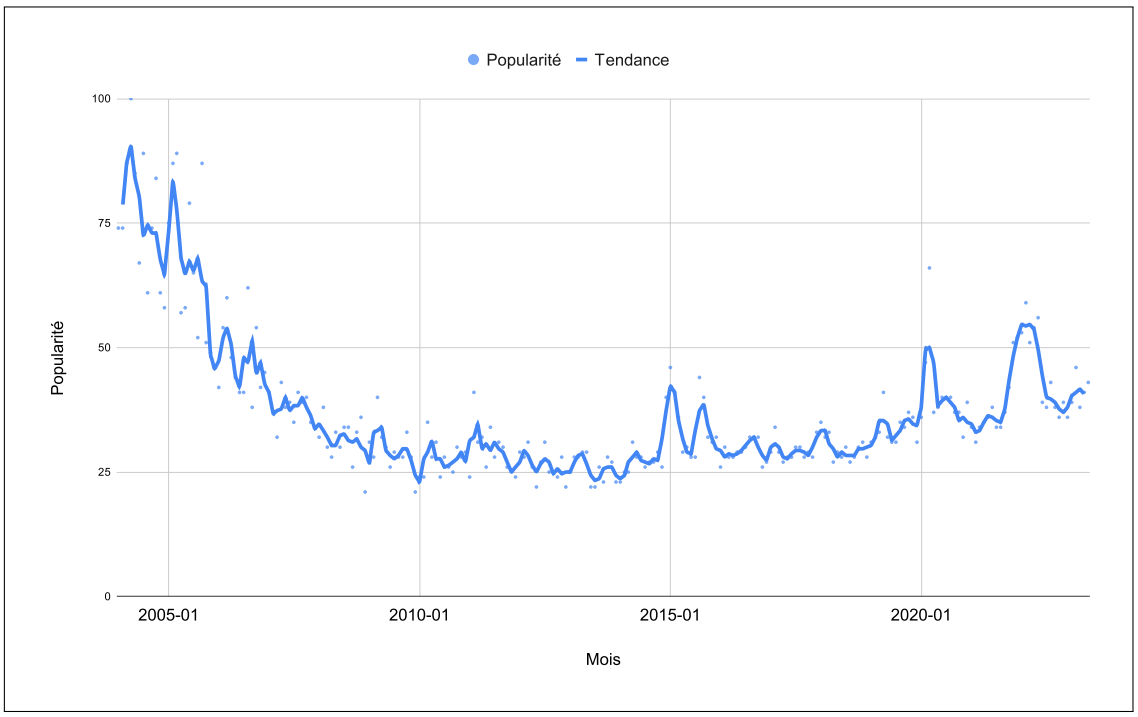

Figure 2.2: Evolution de la popularité de la recherche du mot clé "leurre" dans le domaine "Sécurité Informatique" sur google entre 2004 et 2023

En parallèle, on peut estimer l'intérêt porté plus spécifiquement par la communauté scientifique au sujet de la déception grâce à l'emploi d'outils spécialisés dans l'inventaire des publications sur un thème donné. Pour cela, nous avons utilisé l'outil DimensionIA [21], qui permet de compter le nombre de publications par année et par mot-clé. Nous avons effectué des recherches entre 2004 et 2022 pour les termes "Honeypot" et "Deception", uniquement dans les titres et abstracts, puis avons restreint nos résultats au champ de l'informatique. Nous obtenons des données au format CSV grâce auxquelles nous formons le graphique présenté à la figure 2.3. On remarque un gain d'intérêt pour la recherche intégrant le mot-clé "Deception" tout au long de la période, avec en particulier un grand bond après 2018, même si l'usage du mot "Honeypot" augmente de façon plus modérée.

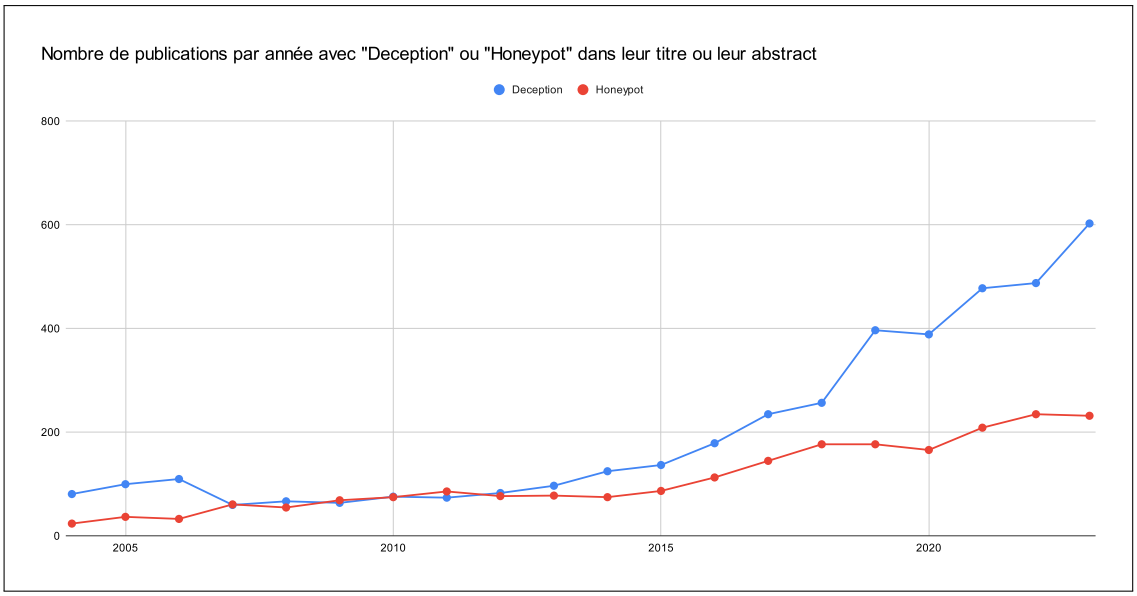

Figure 2.3: Evolution du nombre de publications dans le domaine "ANZSRC 46 Information and Computing" incluant les termes "Deception" et "leurre" dans leur titre ou leur abstract

Ces deux éléments nous permettent de discuter de façon assez informelle sur l'intérêt porté par le monde de la recherche et la société en général sur les thématiques de déception et sur les leurres en particulier sur ces vingt dernières années. On remarque une progression rapide de l'intérêt scientifique pour la déception, sans doute à cause de l'évolution des menaces et de la généralisation des outils permettant de déployer facilement des leurres, tels que la virtualisation et le développement du cloud.

2.2 Définitions et Concepts

2.2.1 Définition et classification des leurres

Bien que l'usage la précède, une définition formelle de référence des leurres souvent retenue est celle de Spitzner qui les définit tels que "une ressource informatique dont la valeur réside dans sa capacité à être sondée, attaquée ou compromise" [65]. Cette définition fondamentale met en avant que l'intérêt même du leurre est d'être exposé aux attaquants et qu'il n'a aucun autre périmètre d'utilisation que de représenter une cible. Cette finalité est primordiale, et permet à la définition d'englober un ensemble très large de ressources informatiques allant des réseaux entiers dédiés à la collecte d'information, les honeynets, à de simples éléments d'information (identifiants, mots de passe, etc) ou les honeytokens [66].

Dans la définition et dans l'usage, les leurres sont donc des ressources déployées et exposées aux attaquants dans le but de collecter des données afin de détecter leur

présence, mais aussi d'analyser leurs procédés et les outils qu'ils emploient. Ceci permet de venir renforcer les bases de connaissances existantes et d'améliorer les autres composants de la cybersécurité. De ce fait, les leurres sont un outil versatile qui s'inscrit dans l'approche de la défense en profondeur, où différentes techniques sont mises en place pour assurer la sécurité des systèmes d'information, chacune se superposant aux autres en y apportant un supplément d'efficacité ou en comblant des lacunes éventuelles.

Lorsque de tels systèmes sont déployés, ils offrent aux attaquants la possibilité d'interagir avec eux, mais la nature et la finalité de ces interactions dépendent de l'implémentation spécifique du système de leurres et de l'intention de leur propriétaire. Dans la littérature, une classification selon le type d'interaction se fait pour définir les leurres. On distingue les leurres en fonction de leur capacité d'interaction avec les attaquants [65].

Haute Interaction

On parle de leurres à haute interaction pour parler de systèmes dans lesquels les attaquants ont accès à un vrai système complet, réel ou simulé. Dans ces systèmes, les différents services exposés à l'attaquant sont réellement implémentés et non pas simulés. L'objectif est d'offrir aux attaquants la plus grande latitude possible quant à leurs possibilités d'interaction avec le leurre. On augmente ainsi la possibilité qu'ils déploient des outils et des méthodes dont l'étude apportera des connaissances précieuses.

Leur implémentation se fait souvent de la même façon que celle d'un équipement légitime, avec une couche de supervision supplémentaire. En plus de la supervision, la seule différence entre un leurre à haute interaction et un équipement de production est l'absence de cas d'usage légitime. De fait, les premiers leurres à haute interaction étaient souvent des machines non utilisées en production et reconditionnées à cette fin, comme le montrent les exemples historiques de Stoll [67]et Cheswick [15].

Du fait de la nature ouverte de la déception dans ces leurres, ils peuvent potentiellement dévoiler énormément d'information sur les pratiques des attaquants. Ainsi, les leurres à haute interaction permettent de collecter des données de grande qualité sans se restreindre à des cas d'observation pré-définis. Leur emploi offre des avantages et des inconvénients que nous pouvons lister:

Avantages

- Collecte de données de qualité, sans restriction forte du périmètre.
- Meilleure capacité à détecter des outils et méthodes inconnus.
- Réalisme poussé, difficile à identifier pour les attaquants.

Inconvénients

- Le déploiement présente un risque, les attaquants pouvant prendre la main sur le leurre, altérer son fonctionnement, l'utiliser comme point de départ pour des rebonds, etc.
- L'attaquant peut plus facilement effacer ses traces et potentiellement altérer les données collectées en échappant au leurre.
- Le coût de conception et d'entretien est très élevé par rapport à celui d'un leurre à plus faible interaction, du fait des contraintes de sécurisation et de supervision.
- L'analyse des données est plus complexe et coûteuse en temps et en moyens. Elle est aussi plus complexe à automatiser.

Ce genre d'implémentation est un exemple parfait de système avec un coût et des risques relativement élevés en comparaison avec les autres systèmes de recherche en cybersécurité, du fait de leur complexité. En contrepartie, ils offrent un haut bénéfice potentiel. Pour cette raison, il est le plus souvent utilisé dans un contexte de recherche, en dehors des environnements de production des organisations. Plusieurs travaux se basant sur le déploiement de leurres ou de réseaux de leurres à haute interaction ont été menés, en général avec l'objectif d'analyser les pratiques des attaquants. On peut citer par exemple les travaux de Nicomette et Alata en 2006 [28] et 2012 [40]. D'autres projets internationaux plus anciens tels que les projets Leurre.com [37] ou NoAH [57] existent, ainsi que d'autres travaux plus récents avec des réseaux de honeypots distribués par Agnaou [27] ou sur un leurre reproduisant une usine par Hilt [33].

Faible Interaction

A l'opposé des leurres à haute interaction, on retrouve les leurres à faible interaction. Dans ce type de leurres, les attaquants sont restreints dans leur capacité à interagir avec le leurre. Cette restriction peut se faire au niveau des moyens de communication des leurres ou de leur capacité à exécuter des commandes. Cette restriction est souvent mise en place par le remplacement d'un service réel par un service fictif émulé, qui simule tout ou partie du fonctionnement du système déployé comme leurre.

L'objectif des leurres à faible interaction est différent de celui des leurres à plus haute interaction, et les avantages et inconvénients diffèrent aussi.

Avantages

- Coût de la conception, de l'entretien et de l'analyse faibles.
- Analyse facilement automatisable.

- Facilité de déploiement sans risques ni surcoût excessif dans des environnements de production.
- Surface d'attaque déterminée: Supervision et sécurisation simplifiées.

Inconvénients

- Surface d'attaque déterminée: Restriction à la conception des données collectables.
- Données collectées de qualité moindre, limitées par le périmètre.

Ces cas d'usage les rendent beaucoup plus à même d'être utilisés en dehors du champ de la recherche, directement en tant qu'outils de détection d'attaque embarqués au sein des réseaux de production des organisations. Le concept est technologiquement mature et des solutions industrielles existent déjà [17, 18, 69], permettant de détecter les attaquants ou d'obfusquer la surface d'attaque apparente des organisations. Nawrocki [39] liste plusieurs dizaines d'implémentations de leurres à faible interaction pour la période 1997-2016.

A l'inverse des leurres à haute interaction, les leurres à faible interaction sont un bon exemple de système au coût et à la complexité relativement faibles, présentant peu de risques, mais permettant de collecter des données plutôt limitées.

En pratique, les systèmes de leurres sont construits sur un spectre de niveaux d'interaction, à mi-chemin des deux concepts présentés. Chaque incrémentation du niveau d'interaction améliore potentiellement la qualité des données collectées en contrepartie d'un coût et d'une complexité croissants. Pour les éléments se situant à mi-chemin des deux concepts, on parle de leurres à moyenne interaction. Spitzner présente un tableau récapitulant les différents niveaux d'interaction des leurres reproduit en table 2.1.

Interaction	Complexité d'installation	Complexité de maintenance	Collecte de données	Risque
Faible	Faible	Faible	Limitée	Faible
Moyenne	Variable	Variable	Variable	Moyen
Haute	Élevée	Élevée	Large	Élevé

Table 2.1: Tableau tiré de Spitzner [65] détaillant les différences d'enjeux entre les différents niveaux d'interaction des leurres.

Finalité des Leurres

Un autre élément important dans la classification des systèmes de leurre est leur finalité telle qu'elle est décrite par Mairh [38]. On peut classer les leurres en deux grandes catégories en fonction de leur finalité : les leurres de renseignement et les leurres de production.

Les premiers ont pour objectif d'acquérir des informations détaillées sur les actions des attaquants tandis que les seconds sont conçus pour être intégrés dans des réseaux de production afin de détecter des attaques ou d'analyser des

menaces. Dans les deux cas, l'objectif est d'attirer des attaquants et de collecter les données qu'ils générèrent grâce à leurs actions, mais la finalité attendue est différente dans ces deux cas. Dans le premier cas, l'objectif est de renforcer les connaissances sur les menaces cyber tandis que dans le second, l'objectif est de détecter ponctuellement les attaquants et d'améliorer la sécurité d'une infrastructure en particulier.

Comme pour ce qui est de l'interaction, il existe tout un continuum de variations entre des leurres de production purs et des leurres de recherche purs. La plupart des leurres se situent en pratique sur un axe reliant les deux concepts.

2.2.2 Détection d'Attaques

Nous avons beaucoup évoqué les leurres sous l'angle d'outils de recherche, mais ils peuvent aussi faire partie de l'arsenal de la détection d'attaques dans le cadre de la défense en profondeur. On peut considérer la définition d'une attaque donnée par la norme ISO 27001 qui la caractérise comme *"toute tentative de détruire, exposer, altérer, désactiver, voler ou obtenir un accès non autorisé à un actif, ou l'utilisation non autorisée d'un actif"* [47]. Dans le but de se prémunir contre ces attaques, les organisations développent et mettent en place des politiques visant à renforcer leur sécurité.

Un des éléments vitaux de cette politique est la détection des attaques, réalisée notamment par des IDS [2] (Intrusion Detection System pour Système de détection d'intrusions). Ces outils au nom explicite ont pour fonction de détecter des artefacts techniques ou comportementaux occasionnés par les attaques et de signaler leur présence. On distingue deux positions d'IDS et deux types de fonctionnement. Les IDS peuvent être positionnés directement sur des équipements dont ils surveillent l'activité interne, on parle alors de HIDS pour Host-based IDS. Sinon, ils peuvent être positionnés au sein du réseau dans une position d'interception du trafic, et l'on parle NIDS pour Network-based IDS. Indépendamment de leur positionnement, les IDS peuvent employer plusieurs méthodes de détection que l'on peut classifier en deux catégories, celles basées sur des correspondances, et celles basées sur des estimations statistiques. Ces deux méthodes ont des avantages et des inconvénients bien identifiés. L'avantage des IDS est leur capacité à identifier des attaques connues de façon simple, efficace et déterministe, ce qui permet d'implémenter des politiques de sécurité et de prévention d'attaques.

Malgré de très bons résultats sur certains cas d'usage, la principale faiblesse des IDS est leur incapacité à détecter de façon fiable des attaques inconnues, et plus généralement une très grande dépendance envers des bases de connaissance de qualité. Dans une optique de défense en profondeur, il est nécessaire de mettre en place des solutions qui viennent complémenter et compléter les capacités des IDS, faute de quoi, il est difficile d'assurer la cybersécurité au sein des organisations avec un coût raisonnable.

Plusieurs solutions ont été envisagées pour remédier au problème de la détection des attaques inconnues, avec des approches comportementales ou basées sur la théorie des jeux [31]. L'une des méthodes retenues est l'utilisation de leurres et de déception.

Les leurres de production orientés pour la détection d'attaques permettent de compléter efficacement les IDS basés sur les signatures en reprenant et en améliorant certains aspects des IDS basés sur l'analyse comportementale. En effet, les leurres permettent, tout comme eux, de détecter des attaques potentiellement inconnues, mais contrairement aux IDS, ils apportent une charge beaucoup moins importante en terme d'alertes et de faux positifs. De la même façon que pour les IDS, il est possible de lister un ensemble d'avantages et d'inconvénients aux leurres de production utilisés en tant qu'outils de détection d'attaques par rapport aux IDS classiques en nous basant sur les travaux de Mokube [8]. Ils présentent peu de faux positifs et ont un coût en ressources plus faible qu'une couverture d'IDS, tant dans l'entretien que dans la conception. Néanmoins, leur périmètre de surveillance est réduit et ils sont vulnérables à des actions de découverte ou d'échappement de la part des attaquants.

En plus de la détection d'attaques, les leurres de production employés comme outils de cyberdéfense peuvent contribuer autrement à assurer une sécurité robuste, en particulier contre des attaquants compétents et motivés. L'emploi de leurres peut permettre d'augmenter la surface d'attaque disponible aux attaquants, mesure efficace contre les acteurs motivés tels que les APT [29]. Cela a pour effet d'éparpiller leurs efforts et de potentiellement les rediriger vers des éléments factices, faisant "gagner du temps" à l'infrastructure réelle. Leur efficacité est plus grande encore contre ce genre d'acteurs du fait qu'ils sont capables de détecter sans difficulté l'exploitation de vulnérabilités inconnues dites "Zero Day" qui échappent presque systématiquement aux IDS classiques par signature.

2.2.3 Virtualisation

Dès que les évolutions techniques ont permis sa généralisation, la virtualisation a attiré l'intérêt des chercheurs et des industriels travaillant sur les leurres. La virtualisation est le procédé par lequel un élément logiciel émule un système d'exploitation ou un équipement matériel à proprement parler.

La virtualisation offre de nombreux avantages en termes de souplesse, de déploiement et d'entretien. Une machine peut voir son état sauvegardé et chargé à tout moment avec un impact très faible sur sa disponibilité. De plus, l'ajout d'une couche d'abstraction entre l'attaquant et le matériel permet de renforcer la sécurité des leurres et la captation des données pour la collecte d'information. Il est en effet possible mais relativement difficile pour un attaquant d'échapper à une machine virtuelle pour accéder au système hôte.

L'inconvénient de la virtualisation est qu'elle implique un surcoût en ressources informatiques du fait du procédé d'émulation logiciel: un système virtualisé sera moins performant qu'un même système tournant directement sur du matériel, même si des benchmarks réalisés par Giallorenzo [32] sur des systèmes récents montrent que l'écart de performances en 2021 est relativement faible, souvent entre 0% et 10% en fonction des applications, du fait de la généralisation de la technologie et de l'adaptation des processeurs.

Un intermédiaire récent entre le *bare-metal* et la virtualisation est le conteneur. A la différence de la machine virtuelle qui émule tous les éléments situés au dessus de l'infrastructure, incluant le système d'exploitation, les conteneurs utilisent les fonctionnalités d'isolation du noyau Linux présentes depuis la version 2.6.24 pour créer des espaces-utilisateur isolés au dessus du système d'exploitation de la machine hôte. Ceci permet d'utiliser un espace isolé, le conteneur, sans devoir s'encombrer de l'émulation du hardware et de l'ajout d'un second OS par dessus l'hyperviseur, économisant ainsi des ressources et facilitant en général la gestion et l'entretien des équipements. L'inconvénient d'une telle approche est qu'il n'est pas possible de construire des conteneurs basés sur un OS autre que linux, et avec un hyperviseur basé sur un autre OS. De même, du fait de la mise en commun de certains éléments de l'OS du système hôte, la segmentation entre un conteneur et son hyperviseur est moindre que pour une machine virtuelle complète.

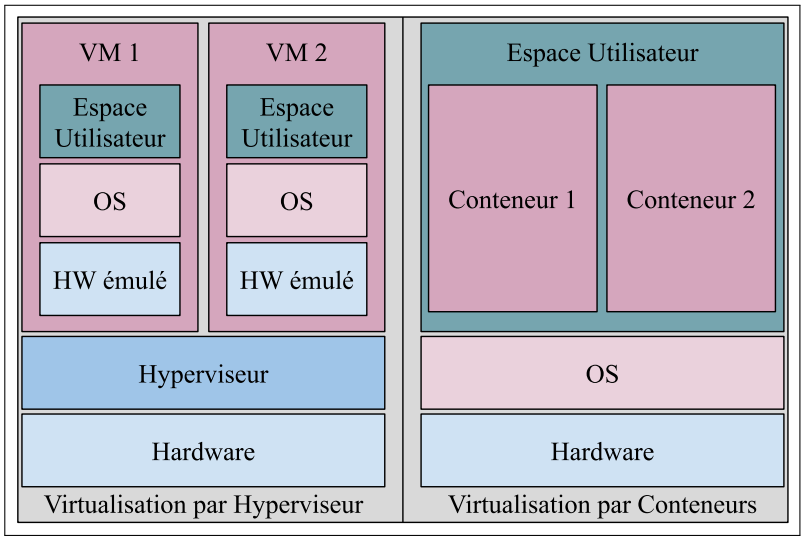

Figure 2.4: Schéma des architectures de virtualisation basées sur les hyperviseurs et sur les conteneurs, inspiré de Eder [24]. On remarque que les conteneurs n'émulent pas de hardware et n'intègrent pas leur propre système d'exploitation, allégeant conséquemment la virtualisation.

Historiquement, les travaux sur les leurres se basent surtout sur des machines virtuelles complètes, mais le développement des solutions de conteneurisation telles que Docker [23] ou LXC [48] poussent à l'apparition d'implémentations de leurres

basés sur cette technologie d'isolation.

2.2.4 Normalisation des cyberattaques

Dans l'étude des cyberattaques et des outils permettant de les analyser et de s'en prémunir, il est important de s'accorder sur des éléments de définition et de taxonomie standardisés. Deux approches de standardisation peuvent être retrouvées parmi les modèles répandus et utilisés dans la recherche et l'industrie: la standardisation des procédés, et la standardisation des détails techniques.

Une tentative d'approche standardisée des procédés d'attaque est issue des travaux de Hutchins [34] pour Lockheed Martin en 2011. Dans ceux-ci, il définit le concept de CyberKillChain (CKC) pour caractériser le parcours d'une attaque en 7 étapes standardisées. Cet enchaînement d'éléments forme la CKC et couvre toutes les étapes de l'attaque, en commençant par une phase de reconnaissance et de construction d'outils, une phase de livraison et d'installation du malware et une phase d'exploitation des éléments malveillants implantés.

Ce standard, bien que limité dans sa granularité et dans ses détails, est un bon point de départ pour l'analyse des procédés d'attaque du fait qu'il regroupe de façon globale toutes les étapes théoriques d'une attaque. Ces étapes théoriques ne représentent pas forcément toutes les procédures réelles mises en place par l'attaquant, mais le cadre général sert de point de départ pour l'analyse et la construction de nombreuses solutions de défense.

Bien qu'utile pour comprendre les procédés et les objectifs des attaquants de façon très abstraite, le modèle de la CKC reste trop général pour permettre une analyse pertinente des actions individuelles des attaquants dans l'étude des attaques réelles. Pour remédier à ces lacunes et proposer un modèle plus granulaire et proche de la réalité des attaques, MITRE à mis au point un modèle de connaissances regroupant non seulement les objectifs mais aussi les procédés employés par les attaquants: la matrice MITRE ATT&CK [4, 11]

Dans sa conception, la Matrice MITRE ATT&CK (MAM) se veut être une taxonomie exhaustive et standardisée des stratégies et des méthodes employées par les attaquants. Elle s'inscrit dans l'écosystème des standards relatifs à la cybersécurité maintenus par MITRE, qui incluent un dictionnaire pour les vulnérabilités connues, les CVE [53] ainsi qu'un dictionnaire des faiblesses répertoriées, les CWE [54]. Une dernière classification standardisée vient compléter les deux autres, celle des CAPEC[52], listant les approches employées par les attaquants pour effectuer leurs actions.

La MAM prend la forme d'un tableau qui présente trois types d'entrées: les tactiques, les techniques et les procédures.

Les tactiques sont au nombre de 14 et représentent les objectifs des attaquants.

Elles décrivent un éventail de finalités poursuivies tout au long du processus d'attaque, de la reconnaissance à l'exfiltration des données, et sont listées dans la figure 2.5. Parmi les catégories de la MAM, les tactiques sont celles qui se rapprochent le plus des éléments de la CKC, tout en détaillant plus finement les actions. Des parallèles peuvent être effectués entre les deux standards, la MAM détaillant bien plus les étapes d'exploitation et d'installation que la CKC. Le nom des tactiques se veut assez explicite, et une description détaillée de chaque tactique est disponible en ligne sur le site de la MAM [4]

Reconnaissance	Resource Development	Initial Access	Execution
Persistence	Privilege Escalation	Defense Evasion	Credential Access
Discovery	Lateral Movement	Collection	Command and Control
Exfiltration		Impact	

Figure 2.5: Liste des Tactiques de la matrice MITRE ATT&CK.

Les techniques représentent quant à elles le moyen par lequel un acteur cherche à accomplir une tactique donnée, ou bien le gain obtenu à la suite de son action. Elles représentent un ensemble général d'actions ou de procédés qui sont assignés à une tactique: pour chacune d'entre elles, une liste de techniques et de sous techniques existe. Par exemple, un attaquant cherchant à développer des ressources dans le but d'effectuer une attaque va chercher à appliquer la tactique *"Resource Development"*. Pour cela, il existe plusieurs techniques, mais il peut choisir d'acquérir un accès via un attaquant l'ayant déjà obtenu. Il utilise donc la technique *"Acquire Access"*. On peut citer quelques exemples:

- Ressource Development →Acquire Access: Achat d'un accès par un attaquant préalablement à son attaque.
- Reconnaissance →Active Scanning: Actions de reconnaissance par scan actif des éléments de la cible.
- Credential Access →Brute Force: Accès aux identifiants par brute force.
- etc.

Le troisième et dernier concept de cet ensemble concentrique est celui des procédures. Les procédures sont les implémentations spécifiques d'une technique, et représentent souvent un cas réel étudié et analysé par MITRE. Comme les techniques peuvent recouvrir plusieurs tactiques, les outils avancés des attaquants font qu'une procédure peut servir à réaliser plusieurs techniques. Par

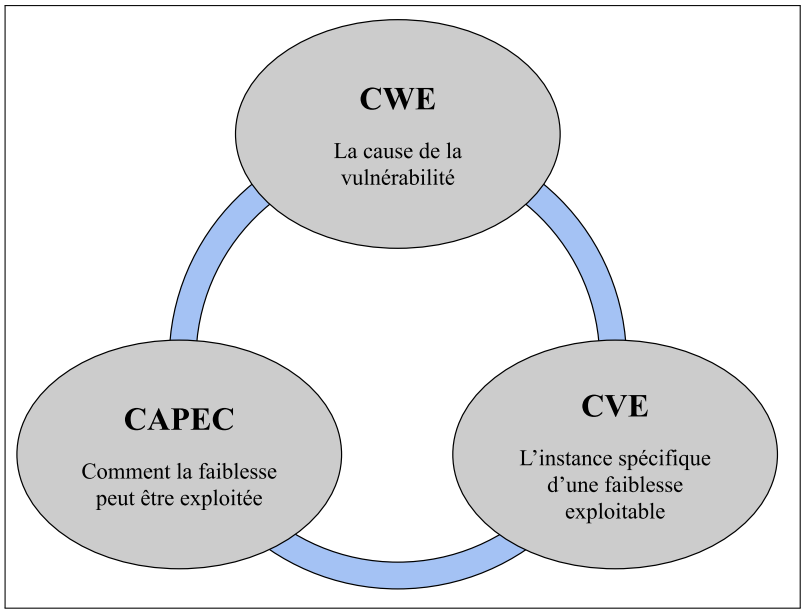

Figure 2.6: Lien entre les CAPEC, les CVE et les CWE, d'après le site web décrivant les CAPEC[52]

exemple, un outil réalisant un scan de ports et un brute-force des services d'authentification découverts couvrira à la fois la technique de scanning et celle du brute-force. Du fait de l'utilisation conjointe des différents standards de MITRE, on peut retrouver des CVE et des CWE dans la description des procédures. Ces CWE et ces CVE sont potentiellement liées à des CAPEC, qui regroupent un ensemble d'opérations effectuées par les attaquants pour exploiter les faiblesses. Ces trois éléments reliés forment un ensemble cohérent. On reprend dans la figure 2.6 le schéma explicatif provenant du site de MITRE.

On peut donc grâce à ces trois éléments propres à la MAM définir les actions des attaquants par un triplet *(Tactique ; Technique ; Procédure)*. On parle dans ce cas de TTP. Ce triplet permet de décrire les actions de façon systématique et inéquivoque, aidant au partage des informations

2.3 Positionnement par rapport à l'État de l'Art

Dans notre état de l'art, nous traitons de travaux concernant la détection d'attaques, le leurrage, la virtualisation et la standardisation des cyberattaques. Ces sujets sont importants pour cadrer nos travaux, mais notre travail de recherche ne porte pas sur tous ceux-ci.

Concernant la virtualisation, nous n'effectuons pas de travaux particuliers à son sujet. Nous nous contentons d'utiliser des éléments relativement récents, en

particulier la conteneurisation, dans le but d'implémenter des leurres. De même, nous évoquons le concept de CyberKillChain, tel que défini par Lockheed, pour offrir un contexte à la taxonomie MITRE ATT&CK.

Ces deux concepts, la matrice MITRE ATT&CK et la virtualisation par conteneurs, nous permettent de nous situer dans la continuité des travaux effectués sur le leurrage tout en y apportant des modernisations. Nous décidons d'utiliser des conteneurs plutôt que des machines virtuelles pour leur plus grande flexibilité et leur plus faible coût matériel. De la même façon, nous estimons que la taxonomie des attaques proposées par MITRE est plus pertinente que la CyberKillChain à l'heure de classifier les comportements des attaquants. Nous estimons pouvoir l'utiliser pour modéliser et implémenter un système de leurres efficace, plus à même de prendre en compte dans sa phase de conception les objectifs précis des attaquants à chaque étape de leurs opérations. Bien évidemment, du fait des contraintes et des choix d'implémentation, nous n'implémentons pas l'intégralité des potentielles attaques décrites par la matrice MITRE ATT&CK dans des leurres, mais uniquement celles qui sont pertinentes dans notre contexte de travail.

Grâce à cela, nous espérons pouvoir déployer un système de leurres et recueillir grâce à lui des données que nous pourrons comparer à celles collectées par les auteurs cités dans cette section. Ceci nous permettra de déterminer si notre approche est efficace, et de mettre l'accent sur l'évolution des pratiques des attaquants.

2.4 Outils Open Source

Dans le cadre de notre travail, nous avons rencontré de nombreuses implémentations open source. Certaines sont utilisées dans des articles que nous citons, tandis que nous en utilisons d'autres nous même dans nos implémentations. Dans cette section, nous présentons ces outils en fonction de leur catégorie.

2.4.1 Leurrage

Comme expliqué dans cette section, de nombreux leurres reposent sur des implémentations logicielles dédiées à l'émulation de certains services et de certaines vulnérabilités. Ces outils regroupent généralement deux fonctionnalités, l'émulation d'un service et la gestion de la supervision et des données collectées. En pratique, cela prend souvent la forme de logs détaillés et exhaustifs des interactions entre les services émulés et les attaquants.

Certains de ces outils peuvent se spécialiser sur l'émulation d'un service ou groupe de services spécifique. On peut parler par exemple de Cowrie [3] ou Kippo[68], des leurres dédiés à l'émulation de services SSH et Telnet. D'autres, tels que Dionaea

[22], Honeyd [61] ou QeeqBox [63] offrent une vraie plateforme permettant d'émuler plusieurs dizaines de services différents.

Nous utilisons Qeeqbox dans notre première implémentation, et dans notre seconde, nous utilisons Cowrie et Dionaea pour déployer plusieurs leurres.

2.4.2 Architecture

Avec la généralisation des technologies de virtualisation, de nombreux déploiements de leurres ont profité des opportunités qu'elles offraient, en particulier concernant la facilité de déploiement et la capacité à contenir des attaquants dans une environnement virtualisé, en limitant les effets de bords et les possibles échappements. La virtualisation offre aussi de nombreux avantages de flexibilité pour le déploiement et la maintenance des honeypots. Un même système peut être sauvegardé et déployé sur plusieurs hyperviseurs, ou redéployé à l'identique en cas de malfonctionnement ou d'altération. Un autre avantage de la virtualisation pour le leurrage est qu'elle permet de contenir les attaquants au sein d'un environnement virtualisé, réduisant les risques d'échappement du système de leurre de leur part. On peut, de façon plus générale, profiter de tous les avantages de la virtualisation présentés dans la section 2.2.3 pour construire des systèmes de leurres robustes et simples à utiliser.

Comme nous l'évoquons dans la section sur les technologies de virtualisation, on peut retrouver deux familles de technologies: la virtualisation complète et la containerisation. Dans le monde Linux, l'outil dominant est KVM[5]. Cet outil open source lié au système d'exploitation Linux permet d'utiliser au mieux les ressources de ce système pour déployer des équipements virtualisés. Un autre outil libre existe, développé par Oracle : Virtualbox[58]. Des solutions propriétaires répandues existent aussi, mais nous ne les avons pas envisagées dans notre travail. En ce qui concerne la conteneurisation, les deux solutions dominantes sont LXC[**?**] et Docker[23]. LXC pour Linux Containers permet d'exploiter des fonctionnalités d'isolation du noyau linux pour déployer des conteneurs.

Dans le but de construire des architectures virtualisées de leurres, nous utilisons dans nos travaux plusieurs outils open source que nous détaillons dans cette section. Pour la virtualisation et le déploiement de nos machines virtuelles, nous employons l'outil KVM pour Kernel Virtual Machine. De façon analogue, nous utilisons pour la virtualisation sous forme de conteneurs l'outil LXC. Afin de faciliter l'utilisation de l'outil, nous pouvons compter sur LXD [13], un outil de gestion de conteneurs construit par dessus LXC et permettant une gestion plus poussée et facile des conteneurs.

2.4.3 Supervision

Pour la supervision de nos leurres, on peut compter soit sur des outils spécifiques, soit sur les implémentations de leurres qui intègrent ces fonctionnalités. Beaucoup de chercheurs font le choix d'implémenter leurs propres solutions de supervision, en modifiant des éléments de base des systèmes qu'ils déploient pour ajouter des routines de journalisation.

De très nombreux outils et IDS existent et se basent sur des principes différents tels que nous le décrivons dans la section 2.2.2

Pour notre part, nous utilisons une supervision au niveau du réseau grâce aux fonctionnalités de journalisation de l'outil IPTABLES [55]. Nous utilisons aussi pour l'une de nos expériences un IDS système open source, Wazuh [70], qui vient compléter notre supervision en nous permettant de monitorer l'état interne de nos leurres à la recherche de modifications effectuées par les attaquants.

La plupart des travaux de la littérature ne sont pas explicites sur leur outils de gestion des données, ou utilisent des bases de données relationnelles. Dans l'idée de pouvoir suivre "en direct" les interactions des attaquants avec notre leurre, nous avons décidé de mettre en place une pile ELK pour Elasticsearch-Logstash-Kibana [25]. Cette pile logicielle permet de collecter, de formater, d'agréger et de présenter visuellement une grande quantité de données, sous forme de tableaux et de graphiques. Le format de base de données relationnelle étant néanmoins plus efficace pour l'analyse systématique des données, nous utilisons l'outil elasticdump [26] pour "extraire" nos données de notre pile ELK et les assembler dans une base de données mySQL.

Chapter 3

Modélisation des leurres

3.1 Contraintes du modèle

3.1.1 Réflexion sur le contexte des leurres

Le point de départ de notre réflexion sur la modélisation des leurres se retrouve dans les contraintes de déploiement que nous avons identifiées dans le chapitre précédent. En nous basant sur la définition fondamentale de Spitzner, nous avons réfléchi à la façon de concevoir des systèmes de leurres vulnérables, attractifs, discrets et supervisés. Pour cette conception, nous avons considéré les hypothèses de travail H2 et H3 que nous détaillons en introduction: les leurres correctement intégrés à leur environnement sont plus attractifs et nous pouvons utiliser la matrice MITRE ATT&CK comme base de travail pour cette intégration.

Cette réflexion est importante, car la vulnérabilité, l'attractivité et la discrétion d'un leurre sont intimement liées. Chacun de ces éléments doit être équilibré et influence les deux autres, ainsi que le fonctionnement général du leurre. Par exemple, un leurre peut être "trop" vulnérable s'il présente aux attaquants des failles trop visibles, connues et simples à exploiter. Ce genre de système éveillera la méfiance des attaquants les plus compétents, diminuant sa discrétion ainsi que la qualité potentielle des données collectées. En contrepartie, il attirera un grande quantité de trafic potentiellement peu pertinent, rendant plus fastidieuse l'analyse des données collectées.

Les attaquants cherchent à exploiter des équipements légitimes et évitent d'interagir avec les leurres. De ce fait, il est nécessaire que les leurres qui sont exposés à leur portée puissent passer pour des équipements légitimes. Pour cela, il est important que leur implémentation et leur architecture s'intègrent de façon harmonieuse avec leur positionnement dans le cadre de leurres isolés et avec les équipements légitimes qui les entourent dans le cadre des leurres de production.

Dans le but de concevoir des systèmes les plus attractifs et discrets possible, nous avons formulé l'hypothèse concernant le contexte que nous présentons en introduction. Un leurre déployé en considérant son contexte de fonctionnement est plus attractif pour les attaquants et plus discret. Notre premier objectif a donc été de concevoir des leurres étant capables de s'intégrer à des écosystèmes existants dans le but de tromper des attaquants sur leur nature.

L'objectif de notre démarche est de tromper les attaquants sur la nature déceptive des leurres. Pour estimer la nature d'un équipement avec lequel il interagit, un attaquant se base sur un ensemble d'artefacts et d'informations qui lui sont accessibles. Il nous a donc semblé plus important d'agir sur ces artefacts mêmes que sur la configuration interne des leurres qui pourrait les causer.

Dans notre travail, nous voulons prendre en compte à la fois la spécificité du point de vue de l'attaquant, ainsi que le type d'information qu'il collecte. En effet, au fur et à mesure des différentes étapes d'une attaque, le point de vue des attaquants évolue. Au départ uniquement externe, il s'élargit au fur et à mesure des interactions avec le leurre.

Notre modèle doit pouvoir décrire des leurres dans des contextes de déploiement et de fonctionnement très différents, qu'ils soient isolés ou intégrés au sein d'architectures plus larges.

3.1.2 Possibilités d'attaque et TTP

A partir des informations qu'il collecte sur un équipement, l'attaquant peut inférer sur sa nature déceptive ou légitime. En plus de cela, il va identifier des possibilités d'attaques qu'il va pouvoir ou non effectuer sur celui-ci. On peut résumer le processus de collecte d'information et d'identification des possibilités d'attaques en trois étapes d'observation, d'interprétation et d'action.

En considérant ceci, il nous est possible d'intégrer au sein de nos leurre des vulnérabilités ou des faiblesses en nous basant uniquement sur les possibilités d'attaque que l'on désire offrir aux attaquants. Il nous devient alors possible de renforcer l'attractivité et la discrétion de nos leurres. Il nous a semblé important d'intégrer les possibilités d'attaques offertes par un équipement lors de son analyse par un attaquant à la conception du leurre. Les vulnérabilités, et la supervision qui leur correspond, seront choisies de façon à optimiser leur visibilité par l'attaquant afin de canaliser son périmètre d'attaque.

Ainsi, en raisonnant de la sorte, nous pourrions construire des leurres en planifiant au préalable les possibilités d'attaque que nous désirons offrir aux attaquants plutôt que de sélectionner des vulnérabilités exploitables.

Il nous a semblé possible d'utiliser cette logique de possibilités d'attaque pour décrire un système de leurres du point de vue de l'attaquant, depuis l'extérieur en *boîte noire*. Pour cela, il nous était nécessaire de pouvoir décrire de façon

exhaustive l'ensemble des possibilités qu'un leurre pouvait offrir à un attaquant. C'est dans ce but que nous avons décidé d'intégrer à notre réflexion la matrice MITRE ATT&CK. Les différentes entrées de cette base de connaissances décrivent les objectifs, les moyens et même les façons d'agir spécifiques des attaquants en utilisant les TTP.

Du fait de la nature déceptive de notre travail, nous ne sommes pas intéressés par toutes les tactiques présentes dans la matrice MITRE ATT&CK.

Premièrement, certaines de ces tactiques sortent partiellement du cadre de notre travail. C'est par exemple le cas de la quasi-totalité des techniques associées aux tactiques de "Reconnaissance" et de "Ressource Development". Les opérations menées par les attaquants pour satisfaire ces tactiques sont effectuées par les attaquants au préalable, de ne sont pas observables du fait qu'elles se déroulent en dehors de notre périmètre de déception. Il nous est par exemple impossible de détecter un attaquant qui développerait un outil adapté à notre implémentation.

Concernant d'autres tactiques, leur présence est souhaitable, mais grandement contrainte. Nous voulons détecter certaines tactiques lorsqu'elles sont appliquées au sein des leurres, mais il est inacceptable qu'elles soient appliquées sur notre système de leurres entier. On peut par exemple penser à la tactique de "Defense Evasion". Il peut être pertinent d'observer quelles méthodes emploient les attaquants pour contourner les méthodes de défenses simulées dans les leurres, mais s'ils parviennent à contourner les mécanismes de défense réels du système de leurre, le système de déception lui-même risque d'être compromis. Un raisonnement similaire peut être mené pour "Exfiltration", "Impact" et "Collection": l'obtention de fausses données issues des leurres est acceptable, contrairement à l'exfiltration de données produites par notre réseau de leurres ou à l'endommagement du système en charge de la collecte des données.

Enfin, l'application de certaines tactiques sur notre réseau de leurres risque d'avoir un impact sur des éléments extérieurs à notre infrastructure, et sont donc inacceptables. Nous ne voulons pas par exemple que des commandes via la tactique "Command and Control" soient transmises sur un de nos leurres qui pourraient le pousser à affecter des systèmes à l'extérieur de notre infrastructure.

3.1.3 Progression des attaquants

L'analyse des données obtenues par un leurre offre des renseignements précieux, mais la mise en réseau de plusieurs leurres permet d'élargir et d'approfondir le périmètre exposé, et donc la variété des procédés mis en place par les attaquants.

L'accès à une machine potentiellement vulnérable est une première étape dans un processus d'attaque complexe. Dans de nombreux cas réels, une compromission

Reconnaissance	*Resource Development*	Initial Access	Execution
Persistence	Privilege Escalation	Defense Evasion	Credential Access
Discovery	Lateral Movement	Collection	~~*Command and Control*~~
Exfiltration		Impact	

Figure 3.1: Liste des Tactiques de la matrice MITRE ATT&CK intégrant en bleu les tactiques qui sortent majoritairement de notre périmètre, en orange celles que l'on veut autoriser pour les attaquants avec des contraintes très spécifiques, et en gris barré celles que nous ne voulons pas intégrer dans notre leurre.

réussie permet d'obtenir un premier accès au sein d'un réseau cible. Après ce premier accès, l'attaquant déploie un ensemble de procédures spécifiques à la reconnaissance et à la progression au sein de sa cible. En intégrant ensemble plusieurs leurres, il nous est possible d'étendre notre champ d'étude aux procédés de propagation des attaquants.

Dans le système de leurres représenté par notre modèle, les attaquants progressent dans deux niveaux de CKC. Le premier niveau est celui de la CKC individuelle propre à chaque leurre indépendant de notre système. Le second niveau est celui de la CKC générale de tout notre réseau de leurres. On s'assure de la couverture par les attaquants du plus grand nombre d'éléments de notre second niveau de CKC en organisant leur double progression. On cherche à les faire progresser au sein de notre infrastructure, en accédant à des leurres au-delà de leur point d'entrée, parallèlement à leur progression dans la CKC. Nous montrons la progression parallèle dans ces deux domaines dans la figure 3.2.

Cette progression se calque sur celle des attaquants expérimentés et motivés, tels que les APT, mènent sur des réseaux ciblés. Grâce à cette organisation, nous venons ajouter de la complexité dans notre réseau de leurres, et augmenter les possibilités d'interaction des attaquants avec nos équipements. Même si les éléments constituant le réseau de leurres sont de niveaux d'interactions variables, l'ensemble présente un haut niveau d'interaction aux attaquants du fait des possibilités d'attaques qui naissent grâce à l'agencement de plusieurs leurres, telles que le déplacement latéral ou l'exploration.

Il convient de relever que dans la littérature, il existe un type particulier de système de leurres appelé *honeynet*. Celui-ci se base sur le déploiement de plusieurs leurres distribués dans le but de créer un large périmètre de collecte de données au niveau d'un réseau. Notre architecture ne s'inscrit pas dans la même démarche. Bien que

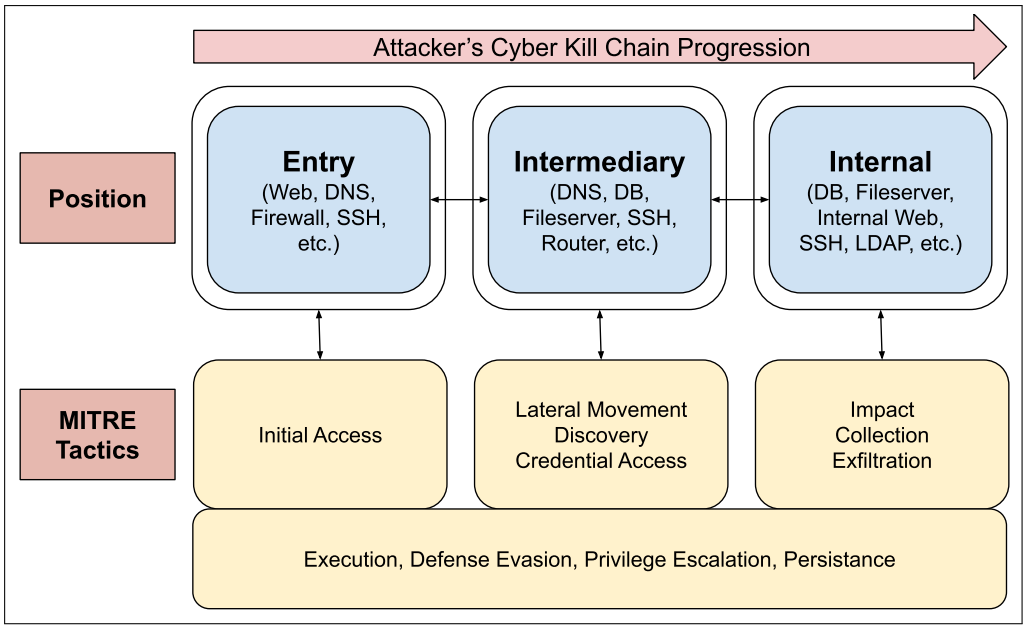

Figure 3.2: Schéma des parallèles entre la progression spatiale des attaquants et leur avancée dans la CKC. Les leurres peuvent appartenir à trois positions en fonction de leur accessibilité depuis l'extérieur du réseau de leurres. Ces positions reflètent l'organisation réelle des réseaux et la progression des attaquants sur la CKC. Cet exemple de progression est implémenté dans notre première expérience.

nous construisions un système de leurres mis en réseau, ces leurres ne sont pas indépendants. Nous ne cherchons pas simplement à élargir le périmètre exposé aux attaquants, mais à construire un cheminement logique via l'implémentation d'une CyberKillChain.

3.2 Description du modèle

Traditionnellement, les leurres sont construits en exposant un ou plusieurs éléments vulnérables dans un périmètre accessible à des attaquants. Cette approche simple permet le déploiement de leurres, mais restreint la portée de ceux-ci. Pour que les leurres construits de cette façon soient efficaces, il faut que les attaquants soient en mesure d'exploiter les vulnérabilités mises en place. Avant cela, il faut qu'ils soient capables de les détecter et de les identifier. En fonction du type de vulnérabilités déployées, du degré de complexité de leur exploitation et du niveau d'exposition des leurres, ces conditions peuvent être difficiles à satisfaire.

Plutôt que de fonder notre modèle de leurre simplement sur des vulnérabilités, nous avons décidé de le baser sur des mécanismes de processus d'attaque. Notre modèle se base sur les possibilités d'attaques que nous rendons accessibles aux attaquants par la construction d'un CyberKillChain dans notre architecture des leurres. En effet, en choisissant des paramètres de notre modèle adaptés au

contexte de déploiement, on peut mettre en place un système de leurre intégrant plusieurs éléments au sein d'un réseau de leurres dédiés. Par le choix des éléments sur nos leurres, on peut tenter de diriger l'attaquant vers un déroulé préétabli, en restreignant les tactiques et les techniques qu'il peut chercher à accomplir sur notre réseau, et en contraignant sa progression "physique" et logique au sein du réseau.

Dans la section suivante, nous décrivons les différents composants de notre modèle de leurre.

3.2.1 TTPs et Vulnérabilités

Le standard MITRE ATT&CK, définit un processus d'attaque par une tactique, une technique et une ou plusieurs procédures[4]. Ceci nous permet de définir au sein d'un leurre un ensemble de techniques et de tactiques. Dans le but d'alléger notre modèle, nous travaillons uniquement sur les tactiques et les techniques de la MITRE ATT&CK et laissons volontairement de côté les procédures. En effet, pour chaque Tactique et technique, le nombre de procédures possibles est très élevé et nous n'estimons pas avoir besoin d'une telle granularité dans notre description.

Dans la taxonomie de MITRE, certaines techniques peuvent être associées à plusieurs tactiques, donc il faut pour chaque technique spécifier la tactique. On définit donc les techniques et tactiques sous la forme de couples. Un seul leurre peut intégrer plusieurs vulnérabilités. On définit donc l'ensemble des TTP d'un leurre comme un ensemble de couples (T,t)

- $\mathbf{T}$: L'ensemble de toutes les Tactiques et techniques.
  - Défini entièrement par la MITRE ATT&CK
  - Elements de la forme (T,t) une Tactique et une technique
- $\mathcal{T}$: L'ensemble des Tactiques et techniques d'un leurre
  - $\mathcal{T} \subset \mathrm{P}(\mathbf{T})$ (Ensemble des parties de $\mathbf{T}$)
  - Elements de la forme (T,t)
- $\{TTP\}_i \subset \mathcal{T}$, ensemble des TTPs de $leurre_i$
  - $\{TTP\}_i = \bigcup_{j=0}^{J}(T_j, t_j) = \bigcup_{j=0}^{J} TTP_j$

Comme indiqué dans la section précédente, les techniques décrivent la façon dont les attaquants réalisent les objectifs que sont les tactiques. Ces techniques sont permises par des vulnérabilités qui peuvent être des vulnérabilités logicielles ou des faiblesses de configuration. Pour les exploiter, les attaquants doivent effectuer des actions qui mènent à l'exploitation des faiblesses existantes.

On peut définir ces actions comme un ensemble de CAPEC. Ces éléments représentent les actions par lesquelles des faiblesses peuvent être exploitées par les attaquants. En plus de ces CAPEC, ces actions peuvent inclure l'exploitation de CVE, mais ce n'est pas forcément le cas. Les possibilités d'attaques pour les

attaquants sont donc de deux types: ils peuvent soit abuser d'une faiblesse en employant un ou plusieurs CAPEC, ou exploiter une vulnérabilité définie par une CVE. C'est pour cela que l'on définit les tactiques des attaquants par un CAPEC et/ou une CVE.

Par exemple, une technique permise par une implémentation logicielle sensible à une injection de code sera définie par le CAPEC-242 d'injection de code ainsi que la CVE correspondante. A l'inverse, une technique exploitant un mot de passe faible sera uniquement définie par les CAPEC de brute force.

On peut donc définir les techniques de notre couple (T,t) à l'aide de ces différents éléments. Une technique sera représentée par un CAPEC seul **ou (XOR)** un CAPEC et une CVE.

- t = CAPEC $\oplus$ (CVE + CAPEC)

3.2.2 Position

Un second paramètre servant à définir nos leurres est leur position. Pour modéliser cet élément complexe de l'architecture, nous utilisons un système simple à une dimension. Notre objectif, comme pour les vulnérabilités, est de nous positionner du point de vue de l'attaquant. De ce fait, la position est relative au point de vue. Un système de leurres exposé dans un réseau interne aura un niveau d'entrée, interne et intermédiaire même s'il se situe à l'intérieur d'un réseau cloisonné. La position des leurres est totalement définie par leur accessibilité et leur exposition à l'extérieur du réseau de leurres. On définit trois niveaux de position que l'on peut détailler ainsi:

Niveau d'Entrée Les leurres situés à cette position sont directement accessibles depuis l'extérieur du réseau de leurres. Ils forment la première ligne que les attaquants peuvent sonder et compromettre dans le but d'atteindre des niveaux plus avancés. Les leurres au niveau d'Entrée sont en général accessibles depuis internet, mais ce n'est pas un impératif. Leur exposition doit se faire en accord avec le modèle de menace que l'on cherche à surveiller.

Niveau Interne Les leurres situés au niveau Interne ne sont pas accessibles ou même visibles depuis l'extérieur du réseau des leurres, à l'inverse de ceux situés au niveau d'Entrée. Ils n'exposent pas de services à l'extérieur du réseau et ne sont pas accessibles aux équipements du niveau d'Entrée.

Niveau Intermédiaire Les leurres en position Intermédiaire ont simultanément accès aux leurres des niveaux Internes et d'Entrée. Ils peuvent servir de points de rebond pour les phases de latéralisation des attaques sur le réseau de leurres.

Nous avons à un moment envisagé d'adopter une structure plus proche de celle des infrastructures réelles en intégrant un niveau de *Zone Démilitarisée* (DMZ)

pour nos leurres. Ce niveau de position aurait regroupé les leurres accessibles depuis l'extérieur mais ne permettant pas de se propager au sein de notre réseau de leurres. Les leurres étant conçus pour maximiser les informations collectées sur les attaquants dans le cadre de leur niveau d'interaction, nous avons abandonné cette idée. L'incorporation d'un niveau de DMZ aurait renforcé le réalisme des leurres que nous pourrions déployer en accord avec notre système. Néanmoins, ces leurres auraient formé des "voies sans issue" en opposition avec notre objectif d'étude des attaquants tout au long des phases d'une attaque.

Le choix des couples (T,t) ne se fait pas au hasard mais est fortement lié à la position des leurres. Ils doivent être choisis judicieusement pour inciter les attaquants à progresser à la fois dans l'espace et dans leur CKC. Par exemple, on peut envisager d'intégrer des vulnérabilités permettant une Tactique de type "accès initial" sur des leurres positionnés au niveau d'Entrée. De la même façon, on peut permettre des techniques de "déplacement latéral" sur des leurres de niveau intermédiaire pour aider à la propagation des attaquants.

En plus des couples (T,t), la position d'un leurre impose de nombreuses contraintes sur sa configuration. L'adressage est une première contrainte, une exposition à l'extérieur étant nécessaire pour les leurres au niveau d'Entrée. La supervision doit être adaptée au positionnement en plus du couple (T,t), en particulier la supervision générale présentée dans les paragraphes suivants.

Les valeurs de position ne sont pas uniques. Ainsi, dans une architecture complexe, on peut envisager plusieurs leurres d'entrée, ne permettant pas un même accès à tous les éléments du réseau. Certains leurres intermédiaires ne sont accessibles qu'à partir de certains leurres d'entrée. Pour cette raison, on peut envisager que certains leurres, pour pouvoir être exposés aux attaquants, se doivent de répondre à certains prérequis. Ces prérequis peuvent se référer à un autre leurre. Par exemple, un leurre accessible uniquement via des privilèges d'administrateur sur un autre leurre, aura comme prérequis de position la compromission et l'élévation de privilèges sur le leurre qui permet d'y accéder. Le cas échéant, on peut définir les prérequis des leurres comme une liste d'éléments suffisants ou nécessaires à la possibilité d'accès des attaquants. Cette notion n'a de sens que pour les systèmes de leurres complexes, intégrant plusieurs composants connectés et interdépendants.

Les prérequis peuvent être de deux types: les prérequis techniques, et ceux non techniques. Les prérequis techniques sont du même type que ceux décrits dans ce paragraphe. Par exemple, si les attaquants ne peuvent accéder à un leurre que par un autre leurre, alors la compromission de ce premier leurre est un prérequis technique. Néanmoins, la volonté de l'attaquant de se propager d'un premier leurre vers le suivant est un exemple de prérequis non technique. Les prérequis non techniques entrent souvent dans le champ de la déception pure, voir de la psychologie des attaquants. Leurs intentions et leur curiosité sont des éléments difficiles à quantifier et à prévoir contrairement à des éléments plus techniques et palpables. Ainsi, sans nous y attarder dans notre travail, nous nous devons

de reconnaitre l'existence de ces prérequis non techniques et les intégrer à notre modèle pour couvrir les cas de figure où ils sont nécessaires.

On peut donc formaliser l'ensemble des positions pour un leurre de la façon suivante:

- $\mathcal{P}$: L'ensemble des positions possibles pour un leurre
  - $\mathcal{P}$ = Entrée, Intermédiaire, Interne
- $\mathcal{PR}$: L'ensemble des Prérequis d'accès à un leurre

3.2.3 Supervision

La supervision est nécessaire pour collecter des données à partir des actions des attaquants. En plus de cela, une supervision robuste permet d'assurer une certaine intégrité du système de leurres. En effet, lors du déploiement du réseau de leurres, les services exposés sont déterminés par les couples (T,t) présentés aux attaquants et par les possibilités de supervision. Ainsi, on n'implémente que ce que l'on peut surveiller. Du fait de l'objectif des leurres, les outils de supervision ont pour seule finalité la collecte de données. Il n'intègrent en aucun cas des fonctionnalités de prévention d'attaque ni d'outils d'analyse de données.

Dans notre modèle, la supervision couvre deux dimensions: une supervision du réseau et une supervision des systèmes. Premièrement, toutes les communications au niveau du réseau de leurres, en interne et avec l'extérieur, sont surveillées. Deuxièmement, l'état interne des différents leurres du réseau et les changements qui y sont apportés sont aussi supervisés. Pour répondre à ces deux missions, on distingue deux mécanismes de supervision:

La Supervision Générale (SG) assure l'observation et la journalisation des événements sur réseau de leurres. Elle sert de base pour la surveillance, la journalisation d'événements et la collecte de données au sein de celui-ci. Elle est chargée de la supervision de toutes les métriques communes aux leurres.

La Supervision Ciblée (SC) assure la supervision de CAPEC ou de CVE spécifiques au sein des leurres eux-mêmes. Cette supervision dédiée sur chaque leurre s'assure que les détails spécifiques à l'exploitation de ces éléments sont consignés. Cette supervision peut prendre beaucoup de formes. Il peut s'agir de configurations additionnelles de la supervision générale ou d'éléments dédiés.

Ainsi, la supervision d'un réseau de leurres est composée d'un système général chargé de la surveillance de l'infrastructure et du trafic sur le réseau. En plus de cette supervision générale, chaque leurre du réseau intègre ses propres fonctionnalités de surveillance ciblées et spécifiques déterminées par les couples (T,t) et la position du leurre. Nous pouvons définir un élément de supervision comme une conjonction de capteurs et de règles. Les capteurs sont des éléments

qui permettent à la supervision de collecter des métriques pertinentes sur des systèmes. Les règles sont l'ensemble des logiques qui permettent d'isoler certaines des métriques collectées par les capteurs en fonction des besoins de supervision. On peut décomposer les règles en deux éléments elles aussi : les prémisses et les conclusions. Une règle est un lien logique qui tire une conclusion sur une métrique collectée à partir d'une prémisse, et est dépendante du capteur qui collecte la métrique. Cette conclusion peut être binaire (alerte/pas d'alerte) ou plus complexe (classification). Bien évidemment, ces deux composantes de la supervision sont intimement liées. Les règles doivent prendre en considération les métriques collectées par les capteurs, qui sont eux mêmes soumis à des contraintes d'architecture et d'implémentation propres à chaque infrastructure.

Indépendamment du couple (T,t) et des choix de supervision, il est important que chaque élément vulnérable soit monitoré par au moins un des éléments de la supervision. Pour représenter cela, on définit une fonction de supervision, qui associe à chaque couple (technique,supervision) 1 si la supervision permet de détecter la technique et 0 si ce n'est pas le cas. Ceci nous permet de nous assurer de la couverture de notre supervision, et de quantifier le degré de redondance de nos solutions. Pour chaque technique d'un leurre, la somme des fonctions de supervision doit être supérieure ou égale à un. Une valeur supérieure à un indique une redondance des supervisions.

On peut résumer les éléments de notre supervision sous la forme suivante:

- $\mathcal{S}$: Ensemble de toutes les supervision possibles
  - $\mathcal{S}$ = Capteurs(Regles)
  - Sup: Fonction de supervision
  - Sup: ($\mathcal{T}$,$\mathcal{S}$) $\rightarrow$ 0,1
  - Sup: (t,s) $\mapsto$ 1 si s detecte t, sinon 0.

Dans notre modèle, ces éléments constituent des composants de la supervision générale ou de la supervision ciblée de nos leurres. Il est important de remarquer que même si ces deux composantes sont disjointes dans notre modèle, il est possible en pratique qu'elles soient liées, voire qu'elles soient deux composantes d'une même implémentation technique.

3.2.4 Présentation du modèle

A partir des éléments précedents, on peut assembler une version complète de notre modèle de leurre:

- **Leurre = TTP,Position,SupervisionLeurre**
- $\mathcal{P}$: L'ensemble des positions possibles pour un leurre
  - $\mathcal{P}$ = Entrée, Intermédiaire, Interne
  - $\mathcal{PR}$: L'ensemble des Prérequis d'accès à un leurre

  - $Pre_i \in \mathcal{PR}$: L'ensemble des prérequis d'un leurre i
  - Position = Pos + Pre_i avec Pos $\in \mathcal{P}$

- **T**: L'ensemble de toutes les Tactiques et techniques.
  - Défini entièrement par la MITRE ATT&CK
  - Elements de la forme (T,t) une Tactique et une technique

- $\mathcal{T}$: L'ensemble des Tactiques et techniques d'un leurre
  - $\mathcal{T} \subset$ P(**T**) (Ensemble des parties de **T**)
  - Elements de la forme (T,t)
  - t = CAPEC $\oplus$ (CVE + CAPEC)

- $\mathcal{S}$: Ensemble de toutes les supervision possibles
  - $* \ \mathcal{S}$ = Capteurs(Regles)

- Sup: Fonction de supervision
  - Sup: ($\mathcal{T}$,$\mathcal{S}$) $\rightarrow$ 0,1
  - Sup: (t,s) $\mapsto$ 1 if s detects t, else 0.

- $\{TTP\}_i \subset \mathcal{T}$, ensemble des TTPs de $leurre_i$
  - $\{TTP\}_i = \bigcup_{j=0}^{J}(T_j, t_j) = \bigcup_{j=0}^{J} TTP_j$

- $SupervisionLeurre$ = SG $\cup$ SC
  - avec SG et SC $\subset \mathcal{S}$ (Supervision Générale et Ciblée)
  - $\forall \ TTP_j \in \{TTP\}_i$, Sup(TTP_j,SG) + Sup(TTP_j,SC) ≥ 1

Un exemple visuel d'un leurre d'après notre modèle peut se retrouver dans la figure 3.3

3.2.5 Modélisation des leurres existants

Afin d'inscrire notre modèle dans la continuité des travaux que nous présentons dans la section précédente, nous avons tenté de représenter par notre modèle plusieurs leurres employés. En adoptant une approche chronologique, nous pouvons observer les évolutions dans les choix d'architecture et d'implémentation. Les leurres que nous étudions sont orientés vers le service SSH. Les principales différences apparaissent donc dans la supervision. Du fait des spécificités d'implémentation et des détails fournis dans les publications, tous les champs du modèle peuvent ne pas être caractérisés finement. Nous avons néanmoins tenté de faire coïncider au mieux les informations disponibles avec notre approche de modélisation.

Leurre de Alata[28] - 2006 Les leurres développés par Alata exposent un serveur SSH à jour, avec des identifiants faibles, et sont monitorés grâce à une

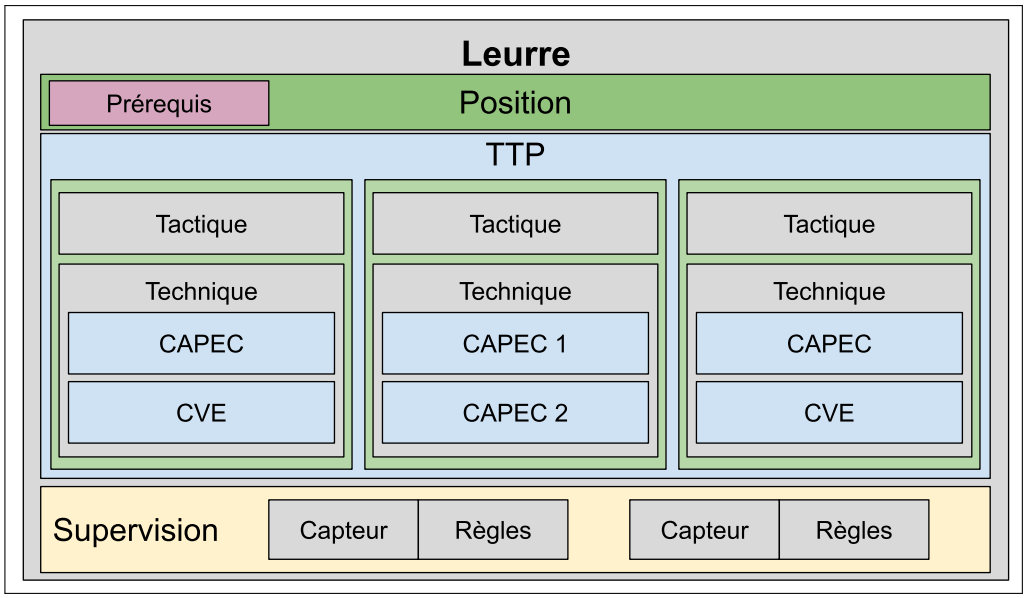

Figure 3.3: Différents composants possibles de notre modèle de leurre détaillés. Le leurre est défini par une position, un ensemble de couples (Tactique, technique) groupés sous appellation *TTP* ainsi que plusieurs éléments de supervision couvrant la Supervision Générale et la Supervision Ciblée.

modification des fonctions des drivers. Dans le but de pouvoir journaliser l'ensemble des frappes entrées par les attaquants, Alata a modifié les fonctions TTY_READ et TTY_WRITE pour y ajouter une routine de journalisation: les entrées des attaquants sont stockées dans la mémoire du noyaux, traitées et enregistrées dans une base SQL. On peut les représenter par notre modèle.

$Honeypot_{al} = (TTP_{al}, Pos_{al}, SupervisionLeurre_{al})$

Pos_{al} = Entrée

$TTP_{al} = (T_{al}, t_{al})$ = (Initial Access, Valid Accounts)

t_{al} = CAPEC-49: Password Brute Force

$SupervisionLeurre_{al} = SC_{al}$: tty_read et tty_write modifiés, traitement et base SQL

Leurre de Nicomette[40] - 2011 L'implémentation de Nicomette met en parallèle plusieurs leurres virtualisés, et intègre une surveillance des appels système en plus de la simple surveillance des commandes entrées. Les auteurs précisent aussi qu'ils ont pu identifier des CVE existantes sur la version du services SSH qu'ils déploient, mais que ces vulnérabilités permettent un contournement de l'authentification qui n'empêche pas les actions des attaquants d'être journalisées et analysées.

$Honeypot_{ni} = (TTP_{ni}, Pos_{ni}, SupervisionLeurre_{ni})$

Pos_{ni} = Entrée

$TTP_{ni} = (T_{ni}, t_{ni})$ = (Initial Access, Valid Accounts)

t_{ni} = CAPEC-49: Password Brute Force et CVE non listées

$SupervisionLeurre_{ni} = SC_{ni}$: Modifications Kernel pour journaliser les commandes, les exécutions et les identifiants (tty driver, do_execve, nouvel appel système)

Leurre de Koniaris[35] - 2013 Le leurre de Koniaris, à la différence des deux autres décrits dans cette section, n'intègre pas de service vulnérable réel, mais une implémentation logicielle dédiée au leurrage, Kippo, qui émule un service SSH.

$Honeypot_{ko} = (TTP_{ko}, Pos_{ko}, SupervisionLeurre_{ko})$

Pos_{ko} = Entrée

$TTP_{ko} = (T_{ko}, t_{ko})$ = (Initial Access, Valid Accounts)

t_{ko} = CAPEC-49: Password Brute Force et CVE non listée

$SupervisionLeurre_{ko} = SC_{ko}$: Implémentation Kippo Dédiée.

Leurre de Melese[1] - 2016 Le leurre de Melese emploie Kippo comme celui de Koniaris, mais aussi Dionaea pour présenter des services autres que le SSH.

$Honeypot_{me} = (TTP_{me}, Pos_{me}, SupervisionLeurre_{me})$

Pos_{me} = Entrée

$TTP_{me} = (T_{me1}, t_{me1}), (T_{me2}, t_{me2})$

(T_{me1}, t_{me1}) = (Initial Access, Valid Accounts)

t_{me1} = CAPEC-49: Password Brute Force

(T_{me2}, t_{me2}) = (Collection, Data from Network Shared Drive)

t_{me2} = CAPEC-643: Identify Shared Files/Directories on System

$SupervisionLeurre_{me} = SC_{me}$: Implémentations Kippo et Dionaea dédiées

L'utilisation de notre modèle permet de distinguer les différents leurres en fonction de leur mode de fonctionnement et de leur choix d'implémentation technique. Dans le cadre de notre travail, nous avons cherché à tester la pertinence de notre modèle en implémentant plusieurs leurres en nous basant sur celui-ci. Dans la section suivante, nous détaillons les leurres que nous avons construit.

3.2.6 Apports du modèle

Le modèle de leurre que nous avons conçu présente plusieurs intérêts.

Premièrement, il représente un moyen générique de décrire les différents systèmes de leurre existants. Nous démontrons cela en modélisant les différents systèmes que nous déployons, mais aussi en l'appliquant à plusieurs leurres de la littérature. Ce modèle offre l'intérêt de recouvrir les différents éléments du leurre, la partie exposée aux attaquants et la partie cachée qui sert à la supervision. En intégrant à sa conception des bases de connaissances standardisées, telles que la MITRE ATT&CK, les CVE ou les CWE, nous assurons la compréhension des différents éléments utilisés par les leurres.

A la base de sa création, on retrouve l'idée de se baser sur une analyse de risques pour assurer un déploiement efficace de leurres adaptés à leur contexte. Dans cette optique, notre modèle de leurre à l'avantage d'être généraliste. Dans l'absolu, il est possible d'appliquer notre modèle de leurre à un équipement légitime supervisé et exposé à des attaquants. Du fait de l'utilisation d'un même cadre pour décrire des leurres et des éléments potentiellement légitimes, il est possible d'utiliser notre modèle pour décrire à la fois les leurres et les éléments de leur écosystème.

Chapter 4

Implémentation des leurres

4.1 Implémentations

Dès la conception de notre modèle, nous avons gardé à l'esprit l'impératif de le rendre implémentable. Nous cherchons à pouvoir identifier la pertinence de notre modèle, il est donc important pour nous de pouvoir implémenter un prototype s'en inspirant. Une fois ce réseau de leurres implémenté, nous l'avons exposé pour évaluer sa capacité à collecter des données.

Cette expérimentation s'est faite en deux temps. Nous avons effectué deux expériences avec deux déploiements et deux implémentations différentes. Dans ce chapitre, nous présentons les deux choix d'implémentations différents que nous avons retenus, en les motivant. Lors de la seconde implémentation, les apprentissages tirés des données de la première expérience ont été utiles pour améliorer notre seconde expérience.

Les deux implémentations présentent des points communs. Dans les deux cas, nous avons tenté de construire au sein d'une même machine virtuelle un réseau de leurres virtualisé d'après notre modèle. Cette machine virtuelle intègre en plus des leurres tous les éléments en charge de la supervision et de la gestion des données. Elle inclut de plus deux interfaces réseau réelles. L'une d'entre elles est exposée aux attaquants, tandis que la seconde sert à l'administration à distance et à l'accès aux données.

Dans les implémentations, on peut distinguer deux plans d'architecture. Le premier plan est celui relatif à l'architecture réseau et à la supervision des leurres. Il intègre tous les éléments relatifs à la Supervision Générale et à la gestion des données. Le second plan est celui des leurres à proprement parler, et il intègre tous leurs composants ainsi que ceux de leur Supervision Ciblée. Un schéma de cette architecture est présenté dans la figure 4.1

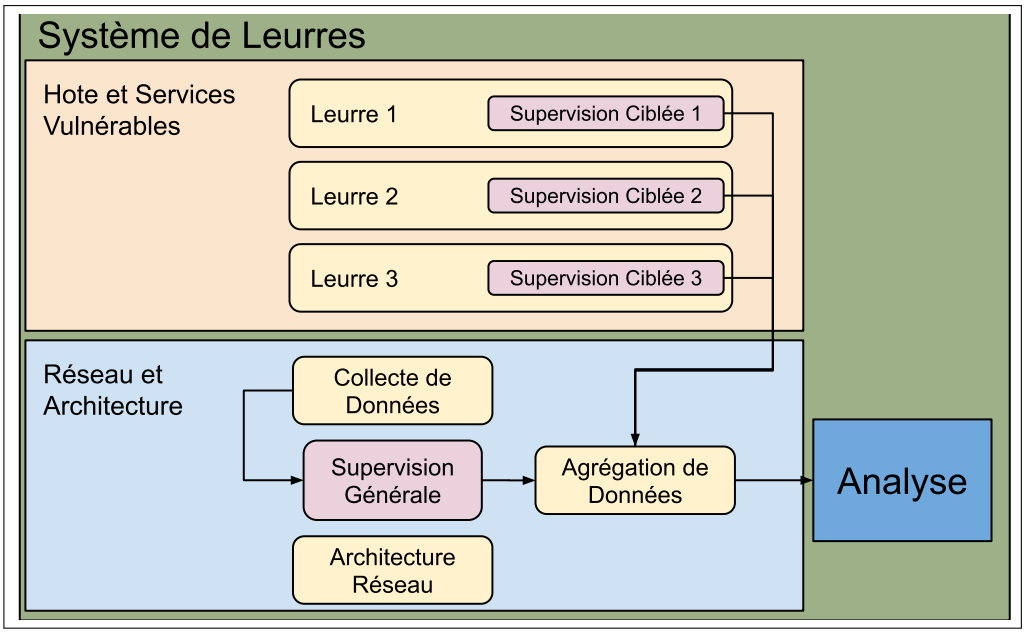

Figure 4.1: Niveaux d'architecture de notre implémentation de leurres. La Supervision Générale est intégrée au niveau "Network and Architecture", avec ses composantes et ses contraintes, notamment concernant l'agrégation des données pour l'analyse. A l'opposé, la configuration des leurres liée à leur Supervision Ciblée appartiennent au niveau "Hosts and Vulnerable Services".

4.2 Première Implémentation

4.2.1 Description Générale

La première implémentation de notre réseau de leurres a été envisagée dès la conception de notre modèle. Nous avons voulu implémenter un réseau de leurres de test en implémentant et en déployant un prototype. Ce prototype a été conçu en envisageant les couples (T,t) les plus susceptibles d'attirer les attaquants.

Notre architecture est virtualisée, pour permettre une gestion simple du déploiement. Ceci nous permet aussi de déployer facilement des clones de notre expérience pour la répéter ou la remettre à zéro en cas de problème. Dans notre implémentation, nous utilisons une machine virtuelle qui intègre les éléments de supervision générale et les outils d'agrégation de données. Au sein de cette machine virtuelle, nous avons implémenté chaque leurre sous la forme d'un conteneur LXC. Cette architecture nous permet d'avoir un fonctionnement relativement économe en ressources.

Grâce à la virtualisation de notre infrastructure et à l'utilisation d'un seul leurre de niveau d'Entrée, notre implémentation fonctionne sur un serveur dédié généraliste. Les prérequis matériels sont abordables: deux interfaces réseau, une adresse IP publique, un seul processeur et 8Go de mémoire vive. Grâce à cela,

notre expérience est déployable dans n'importe quel serveur dédié avec les ressources suffisantes.

Notre implémentation inclut trois leurres. Chacun de ces leurres est positionné à une position différente, avec un leurre en Entrée, un leurre Intermédiaire et un leurre Interne. Ces leurres sont organisés conformément aux contraintes imposées par notre modèle. Les attaquants n'ont accès depuis l'extérieur du réseau qu'au leurre en Entrée, puis au leurre Intermédiaire depuis celui-ci et enfin, au leurre Interne par un second rebond. Parmi les différentes stratégies d'implémentations de réseaux de leurres définies par Scottberg [42], notre architecture est un "Hacker Zoo" où l'intégralité de notre environnement est dédié à la déception. Afin de former ce réseau, nous utilisons un bridge virtuel permettant de créer un réseau propre à nos leurres à l'intérieur de notre machine virtuelle. En limitant les connectivités entre les leurres via des restrictions de pare-feu, nous simulons notre réseau. Par exemple, tout le trafic entre notre leurre d'Entrée et notre leurre Interne est bloqué, obligeant les attaquants à se propager par rebond.

D'un point de vue technique, notre machine virtuelle contenant les différents leurres fonctionne sous la version 20.04 du système d'exploitation Ubuntu. Elle est hébergée par Oracle VirtualBox Manager et sa gestion se fait via un accès en bureau distant. Pour cela, nous exposons aussi sur notre serveur un service RDP, mais sa configuration est sécurisée, et il ne fait pas partie du périmètre exposé aux attaquants pour notre expérience. Néanmoins, sa présence peut servir à jauger l'attractivité du service par rapport à ceux exposés dans le leurre.

En ce qui concerne la supervision générale propre à tous les leurres, nous utilisons trois composants. Le premier est IPTables, grâce auquel nous pouvons journaliser les entêtes de tous les paquets IP qui entrent dans notre réseau de leurre. Ceci nous permet d'observer l'activité au niveau du réseau. De plus, nous intégrons un contrôleur d'EDR wazuh qui pourra servir à centraliser les différentes données générées par les supervisions ciblées de nos leurres. Un dernier élément de notre supervision générale est un conteneur avec une pile ELK (Elasticsearch-Kibana-Logstash). Cet outil d'agrégation et de présentation de logs nous permet de rassembler et d'analyser nos données de façon centralisée, quelle que soit leur provenance dans notre supervision.

Une architecture technique de notre implémentation est représentée dans la figure 4.2.

4.2.2 Détails des leurres

Nous avons intégré au sein de notre machine virtuelle trois leurres intégrés dans leur propre conteneur LXC. Dans les trois cas, nous avons utilisé pour la création des conteneurs une image de base correspondant à Ubuntu 20.04 serveur, comme pour la machine virtuelle hôte.

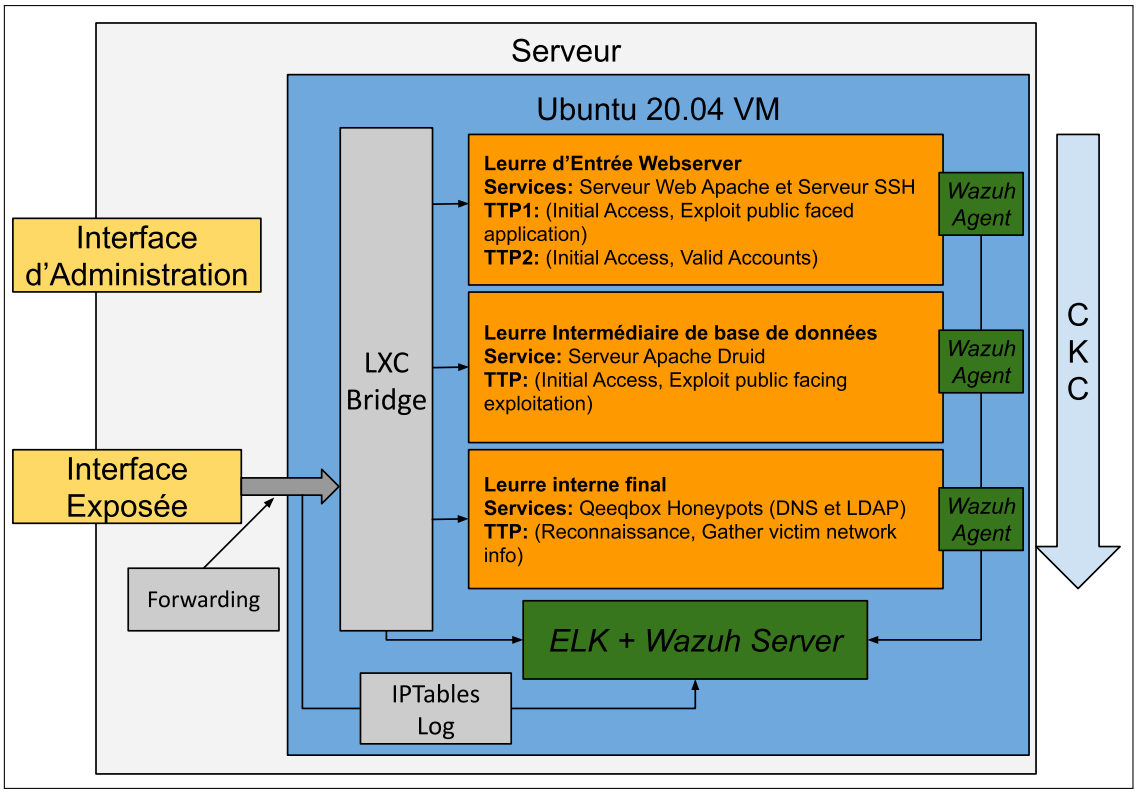

Figure 4.2: Schéma de notre implémentation. En jaune, les deux interfaces permettent de connecter le réseau de leurres à l'extérieur. En Orange, on représente les trois leurres sous la forme de conteneurs LXC. En gris, on retrouve les éléments de réseau qui permettent le fonctionnement de notre implémentation. En vert, on représente les éléments des supervisions générales et spécifiques. On représente aussi sur le schéma le sens de progression des attaquants.

Le premier leurre se situe en position d'Entrée. Il expose à l'extérieur du réseau de leurre deux services. Le premier service est une version vulnérable du serveur web Apache (httpd). Le second service est une version à jour d'un serveur OpenSSH, mais configuré avec un mot de passe faible. Le second leurre situé en position Intermédiaire expose un serveur de base de données Apache Druid vulnérable. Le dernier leurre, positionné au niveau Interne, intègre une implémentation de honeypot à faible interaction configurée pour simuler un service LDAP et DNS.

Nous décrivons nos leurres en détail avec notre modèle dans chaque section qui leur est dédiée.

Leurre Web-SSH

Le premier leurre, en position d'entrée, propose une tactique d'accès initial. Cette technique permet aux attaquants d'obtenir un point d'accès au leurre, mais aussi au réseau de leurres en général. Deux tactiques sont employables par les attaquants

sur ce leurre d'entrée, une pour chaque service exposé. Il est aussi possible pour les attaquants d'effectuer des attaques de type "déni de service" sur notre réseau de leurres via le service web.

Le service web fonctionne grâce au démon http d'Apache [59] à la version 2.4.49. Cette version est vulnérable à la faille CVE-2021-41773 [14], activement exploitée à l'heure de notre déploiement. Cette faille de criticité élevée (CVSS 7.5) permet aux attaquants d'accéder à des éléments de l'hôte du serveur web en employant des procédures de "path traversal". En configurant volontairement le serveur web de façon adaptée, nous pouvons ouvrir l'accès aux attaquants à tous les autres fichiers du système.

Le serveur SSH est déployé quant à lui avec une version à jour (à l'époque du déploiement) de OpenSSH. La faiblesse de cette implémentation provient du fait que nous avons volontairement fait le choix d'un mot de passe faible. Pour cela, nous nous sommes basés sur les travaux de Maxime Alay-Eddine sur Richelieu [9]. Ce projet compile un classement des 20000 mots de passe les plus communs utilisés en France en 2019. Dans les premiers points de la liste, on retrouve des mots de passe constitués de séquences classiques (123456, azerty, etc.), puis plusieurs mots usuels. Pour notre implémentation, nous avons utilisé le 6eme mot de passe le plus courant, "*marseille*", avec le nom d'utilisateur usuel "*user*".

Nous avons spécifiquement choisi ce mot de passe pour plusieurs raisons. En premier lieu, ce mot de passe nous a semblé être un fort marqueur linguistique français, nous permettant d'évaluer si la localisation des leurres peut avoir un impact sur le choix des attaquants. De plus, ce mot de passe nous semblait facilement trouvable par des attaquants, mais moins accessible que les premiers de la liste. Nous espérions donc trouver un équilibre entre attractivité et volume d'activité des attaquants. En effet, nous craignions de ne pas être capables d'assurer la stabilité de notre leurre et d'analyser nos données de façon pertinente en cas d'influx trop massif d'attaquants.

Pour la supervision de ce leurre, nous utilisons à la fois notre système de supervision ciblée et de supervision générale. La supervision générale nous permet de collecter des données au niveau du réseau concernant les tentatives d'attaques, en particulier les IP sources et les ports ciblés. Pour la supervision ciblée, nous utilisons un agent Wazuh configuré pour transmettre les logs générés par les démons SSH et HTTP à notre agrégateur ELK central. Nous avons donc une visibilité sur l'interface avec l'extérieur et sur l'état des applications exposées.

Nous pouvons représenter notre leurre avec notre modèle ainsi:

- Honeypot = (TTP_i , Position , SupervisionLeurre)
- Position = Pos $\wedge$ PR
- Pos = Entrée
- PR = $\emptyset$
- $TTP_1 = (T_1, t_1)$ = (Initial Access, Valid Accounts)

  - t_1 = CAPEC-49: Password Brute Force
- $TTP_2 = (T_2, t_2)$ = (Initial Access, Exploit Public Facing Application)
  - t_2 = CAPEC-126: Path traversal $\wedge$ CVE-2021-41773
- SupervisionLeurre = SC $\wedge$ SG
  - SC = Wazuh IDS Agent, journalisation des commandes
  - SG = IPTables Logging, ELK Stack

Leurre Intermédiaire

Le second leurre de notre architecture propose aux attaquants un moyen de se propager au sein de notre réseau de leurres. Notre travail sur l'implémentation de notre modèle a été réalisé à la fin de l'année 2021. C'est pendant cette période que l'existence d'une vulnérabilité critique avec un score CVSS de 10/10 sur la bibliothèque *Apache Log4J*, la CVE-2021-44228 [56] à été rendue publique. Cette bibliothèque est employée par de très nombreuses applications dans des domaines et des périmètres très variés, et la vulnérabilité permet aux attaquants de l'exploiter pour exécuter du code arbitraire. Il nous a semblé pertinent à ce moment-là de déployer un leurre intégrant cette bibliothèque vulnérable. Ceci nous permettait de profiter au mieux de l'effet d'annonce et de l'opportunisme des attaquants pour renforcer l'attractivité de notre système. Nous avons donc décidé de mettre en place un leurre permettant aux attaquants d'exécuter du code arbitraire dans le but de lui permettre de se propager dans notre réseau de leurres.

Parmi les différents produits intégrant la bibliothèque vulnérable dans son fonctionnement, nous avons décidé de déployer dans un leurre le logiciel de base de données Apache Druid, dans sa version précédant la correction de la vulnérabilité. Ce logiciel, bien que moins connu que les autres, était à l'époque listé dans les éléments vulnérables à la CVE.

Pour la supervision ciblée, nous avons utilisé un agent Wazuh similaire à celui du premier leurre pour rediriger les logs de Druid vers notre agrégateur de données. Ceci nous permet de couvrir les actions des attaquants sur le leurre tout en ayant une approche unifiée de notre supervision ciblée, facilitant la configuration et l'entretien de notre système.

Nous pouvons représenter notre leurre avec notre modèle ainsi:

- Honeypot = (TTP_i , Position , SupervisionLeurre)
- Position = Pos $\wedge$ PR
- Pos = Intermédiaire
- PR = Emission de commandes depuis le leurre d'entrée
- $TTP_1 = (T_1, t_1)$ = (Initial Access, Exploit Public Facing Application)
  - t_1 = CAPEC-267: Leverage Alternate Encoding $\wedge$ CVE-2021-44278
- SupervisionLeurre = SC $\wedge$ SG
  - SC = Wazuh IDS Agent
  - SG = IPTables Logging, ELK Stack

Leurre Interne

Le leurre interne est le plus simple des trois qui composent notre réseau de leurres. Il se base sur le même socle de base que les deux autres en utilisant Ubuntu 20.04 comme OS. Là où les deux autres leurres exposent aux attaquants des services réels et vulnérables, ce dernier leurre n'expose pas de système réel. En effet, ce leurre se base sur un outil open-source dédié à la déception, "qeeqbox honeypots" développé par qeeqbox et dont le code source est disponible sur Git-Hub [63]. Cet outil développé en python permet de simuler un fonctionnement minimal de 30 services différents. Pour chacun de ces services simulés, l'outil écoute sur le port idoine, répond aux tentatives de connexion et journalise un ensemble de valeurs.

Nous avons opté pour cette approche plus simple que celle des deux autres leurres car du fait de notre architecture, les attaquants arrivant à notre leurre de niveau interne ont déjà interagi avec les leurres précédents. Nous avons considéré que les deux premiers leurres seraient ciblés par les attaquants dans le but de gagner l'accès à des ressources internes. Ce dernier leurre représente ces ressources internes, et en quelque sorte le "point d'arrivée" des attaquants au sein de notre réseau de leurres. De ce fait, il n'y a pas de compromission possible pour ces leurres, et nous les considérons plutôt comme une sorte de "ligne d'arrivée" pour les attaquants. Si nous sommes capables de détecter leur interaction avec ce leurre, c'est qu'ils ont parcouru entièrement le réseau de leurres qui leur était exposé.

Parmi les multiples choix d'implémentation simulées, nous avons configuré notre outil pour présenter de faux services DNS et LDAP. Nous avons considéré que ces deux services sont assez universels pour s'intégrer à notre architecture de leurres de façon pertinente. De plus, les informations qui peuvent être extraites de leur compromission sont en général pertinentes pour les attaquants qui cherchent à mener des attaques complexes. Ceci vient donc renforcer l'attractivité de ces faux services déployés.

Du fait que ces leurres ne sont pas des systèmes réel mais des outils dédiés, ils intègrent leurs propres fonctions de supervision. L'ensemble des échanges entre attaquant et leurre est journalisé. Nous utilisons Filebeat, un outil associé à la suite ELK, pour transmettre les logs générés à notre agrégateur. Nous n'avons pas besoin pour ce leurre d'utiliser d'agent Wazuh comme pour les autres. En effet, pour les deux premiers leurres, il est nécessaire de traiter et de formater les données issues des logs, là où l'outil de qeeqbox fournit directement des données formatées, ingérables par ELK.

On peut représenter notre leurre avec notre modèle ainsi:

- Honeypot = (TTP_i , Position , SupervisionLeurre)
- Position = Pos $\wedge$ PR
- Pos = Interne
- PR = Emission de commandes depuis le leurre intermédiaire
- $TTP_1 = (T_1, t_1)$ = (Reconnaissance, Gather Victim Network Information)

Limites de la première Expérience	Solution apportées
Identifiants peu attractifs	Usernames et mots de passe multiples. Choix d'identifiants moins "français" et plus anglophones
Adresse IP peu attractive	Déploiement à TSP, avec une adresse IP académique connue
Emergence de vulnérabilités nouvelles	Services simulés moins évolutifs que les vrais services
Supervision du trafic sortant	Supervision complète des commandes via proxy

Table 4.1: Description des limites identifiées dans la première expérience et des solutions qui y sont apportées dans la conception de la deuxième expérience.

  - t_1 = CAPEC-54: Query System for Information (DNS et LDAP)
- SupervisionLeurre = SC $\wedge$ SG
  - SC = Agent IDS Wazuh, Implémentation dédiée à la déception
  - SG = IPTables Logging, ELK Stack

4.3 Limites de la première implémentation

A la suite de notre analyse que nous détaillons dans le chapitre 5, nous avons tiré des enseignements concernant notre premier déploiement. Nous remarquons que nos leurres sont relativement peu attractifs, pour plusieurs raisons évoquées dans les sections suivantes, et que notre supervision est imparfaite. Nous dressons dans la table 4.1 les différentes limites de notre première expérience ainsi que des potentielles solutions qui pourraient y être apportées.

Premièrement, les identifiants choisis pour notre leurre n'ont pas été attractifs. Nous avons fait le choix de considérer des identifiants francophones, ce qui n'est pas pertinent au vu des tentatives effectuées par les attaquants. De plus, du fait de l'utilisation d'un service SSH légitime, il ne nous a pas été possible d'apporter beaucoup de granularité dans nos identifiants. Il ne nous est pas possible, par exemple, d'attribuer plusieurs mots de passe à un seul identifiant comme nous pourrions le faire avec un service dédié à la déception.

Deuxièmement, notre adresse IP attribuée à un hébergeur public ne présentait aucune attractivité particulière pour les attaquants. Nous n'avions pas non plus de visibilité sur le trafic sortant de notre réseau de leurres, ce qui est rétrospectivement un mauvais choix d'implémentation. En dernier lieu, nos services vulnérables étant fixés ainsi que leur supervision, nous n'étions pas forcément en capacité de détecter l'exploitation de vulnérabilités préalablement inconnues qui auraient émergé au cours de notre expérience. Ce dernier point est moins critique que les trois autres du fait que les attaquants n'ont pas compromis les éléments vulnérables.

A la lumière de ces différents constats, nous avons conçu et implémenté une seconde expérimentation pour chercher à corriger les défauts et à compléter l'approche de notre première implémentation.

4.4 Seconde Implémentation

4.4.1 Description Générale

Notre seconde implémentation a été conçue après le déploiement et l'analyse des résultats de la première implémentation.

La principale différence de contexte entre nos deux expériences est que la première a été déployée au sein d'un serveur dédié loué auprès d'un hébergeur public. Il n'y avait donc rien de remarquable ou d'attractif à notre localisation. Pour la seconde expérience, nous avons pu déployer notre système de leurre directement au sein de l'infrastructure de Télécom SudParis. L'adresse IP par laquelle les attaquants ont accès aux leurres en position d'entrée est facilement identifiable par WHOIS comme appartenant à l'établissement.

Nous avons pris cette décision de déploiement en espérant renforcer l'attractivité de notre réseau de leurres. En effet, plusieurs établissements d'éducation supérieure avaient été victimes d'attaques pendant les années 2021 et 2022. Nous étions aussi informés de première main du fait que Télécom SudParis avait été la cible de plusieurs tentatives d'attaques récentes. Nous avons donc considéré que les IP marquées comme appartenant à cet établissement attireraient plus d'attaquants. En plus de cela, nous attendions des attaquants plus motivés et qualifiés que ceux ciblant des serveurs anonymes hébergés publiquement.

Malgré ce contexte différent, nous conservons le même paradigme de conception. Nous cherchons à implémenter un réseau de leurres suivant notre modèle en choisissant des couples "tactique;technique" bien adaptés à nos contraintes afin de les exposer aux attaquants. De même, pour les raisons énoncées dans la section précédente, nous avons conservé notre approche virtuelle, avec des leurres sous la forme de conteneurs hébergés au sein d'une machine virtuelle. Les ressources nécessaires pour assurer le fonctionnement de notre réseau de leurres sont grossièrement les mêmes que pour la première expérience. Pour des raisons techniques d'hébergement, nous avons remplacé notre hyperviseur Oracle VirtualBox par Kernel-Based Virtual Machine (KVM). Nous avons aussi mis à jour notre système d'exploitation vers la version *Long Time Support* suivante, passant de Ubuntu Server 20.04 à 22.04. Cette mise à jour s'est aussi faite pour les nouveaux conteneurs intégrant les leurres.

Dans cette implémentation, nous avons optimisé notre conception au vu des résultats de notre première expérience. Par exemple, nous avons remarqué lors de notre premier déploiement que les attaquants ont très majoritairement ignoré le service web au niveau de notre leurre d'entrée pour se focaliser sur le service d'authentification SSH. C'est pourquoi nous avons décidé de nous concentrer sur celui-ci lors de ce déploiement.

Pour cette seconde implémentation, nous avons opté, dans certains de nos leurres, pour des solutions dédiées à la déception plutôt qu'à des services réels vulnérables.

Ce choix est motivé par plusieurs raisons.

Premièrement, le coût de mise en place de ce genre de services est bien moindre. Il suffit de les installer, et ils intègrent en général tous les éléments nécessaires à leur fonctionnement, de leur déploiement à leur supervision. Il n'y a donc pas de mesures particulières à prendre concernant la supervision ciblée des leurres simulés autre que relier leur système de supervision intégré à notre agrégateur de données.

A l'inverse, les systèmes réels nécessitent souvent d'être installés depuis du code source archivé, avec des paramètres de compilation ou de configuration volontairement erronés. Là où il est facile de trouver de la documentation expliquant comment sécuriser les services et les applications, l'inverse est plus difficile. Il n'est pas toujours évident de rendre les services vulnérables en ne focalisant qu'un ensemble restreint de TTP et en gardant la main sur la supervision ciblée.

Deuxièmement, le coût en ressources des systèmes simulés est inférieur à celui des systèmes réels. Ceci nous permet de déployer plus de leurres dans une même machine réelle, ou de déployer notre réseau de leurres dans des équipements moins puissants.

Un dernier avantage des leurres dédiés par rapport aux services réels vulnérables et supervisés existe. Lorsqu'un leurre est mis au point avec un service et une version spécifique, il représente une instantanée dans le temps de l'implémentation logicielle choisie. Après un certain temps, il est possible que des nouvelles vulnérabilités soient identifiées sur cette même version. Celles-ci peuvent permettre aux attaquants de déployer des techniques et des tactiques qui ne sont pas observables par la supervision choisie lors du déploiement. Il peut être alors compliqué de pallier à ces vulnérabilités sans affecter celles plus anciennes que l'on cherche à exposer. Cette variabilité peut rendre complexe les déploiements de leurres sur le long terme.

Ce problème se pose moins avec les implémentations dédiées au leurrage qui évoluent moins dans le temps, et qui exposent une plus petite surface d'attaque. En contrepartie, l'absence d'émergence de nouvelles vulnérabilités peut porter atteinte au réalisme des leurres simulés.

Les services dédiés à la déception ont néanmoins plusieurs inconvénients. Premièrement, ils peuvent sembler moins réalistes que les leurres réels lorsqu'ils sont examinés par un attaquant expérimenté. Pour palier à ce problème, nous avons décidé de ne pas utiliser ce genre d'outils pour nos leurres au niveau d'entrée, mais uniquement pour ceux intermédiaires et internes. Ainsi, si nous éveillons la suspicion d'un attaquant, il sera déjà à l'intérieur de notre réseau de leurres.

Un second inconvénient des outils de leurrage, qui rejoint un peu le premier, est qu'ils sont parfois connus et identifiables par les outils utilisés par les attaquants.

Par exemple, un de nos outils était reconnu en tant que leurre par les fonctionnalités de détection de service de Nmap. Au vu de l'utilisation omniprésente de cet outil de scan, nous avons dû apporter des modifications à notre leurre pour empêcher cette détection. Celles-ci sont décrites dans la section concernant les détails des leurres.

Pour toutes ces raisons, nous avons opté pour des systèmes simulés pour deux de nos leurres intermédiaires. En ce qui concerne notre leurre d'entrée, nous avons une conception à mi-chemin entre le service réel et le service simulé, via l'utilisation d'un proxy SSH que nous détaillons dans la section suivante. Nous sommes conscients que ce choix d'implémentation représente un retour en arrière concernant le réalisme de nos leurres. Néanmoins, ces changements sont relativement transparents du point de vue des attaquants, et nous espérons que les choix judicieux d'architecture et de services exposés permettront de compenser la potentielle perte d'attractivité des leurres.

On peut retrouver un schéma de notre architecture dans la figure 4.3 et un schéma plus détaillé de son implémentation technique dans la figure 4.4

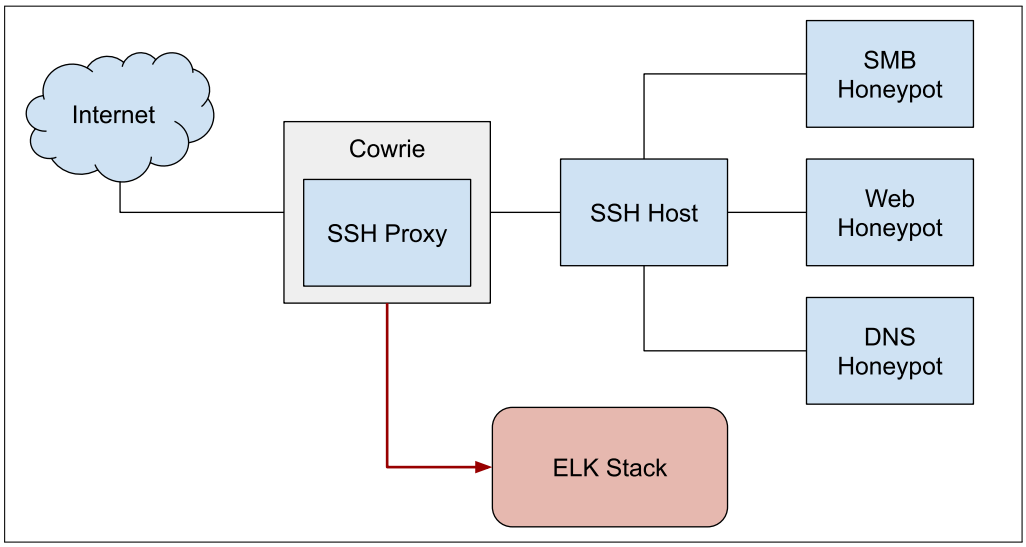

Figure 4.3: Architecture générale de la seconde implémentation du système de leurres. Nous avons conservé de la première implémentation le service web, le serveur DNS simulé ainsi que la pile ELK pour l'agrégation des données. Nous avons ajouté un nouveau leurre SSH ainsi qu'un leurre SMB simulé.

4.4.2 Détail des leurres

Nous avons intégré dans cette seconde implémentation de notre modèle quatre leurres fonctionnant au sein de quatre conteneurs LXC distincts. Parmi ces leurres, nous avons directement réutilisé les leurres WEB et DNS que nous avions déployé lors de notre première expérience. Nous détaillons dans cette section les deux autres que nous avons ajoutés.

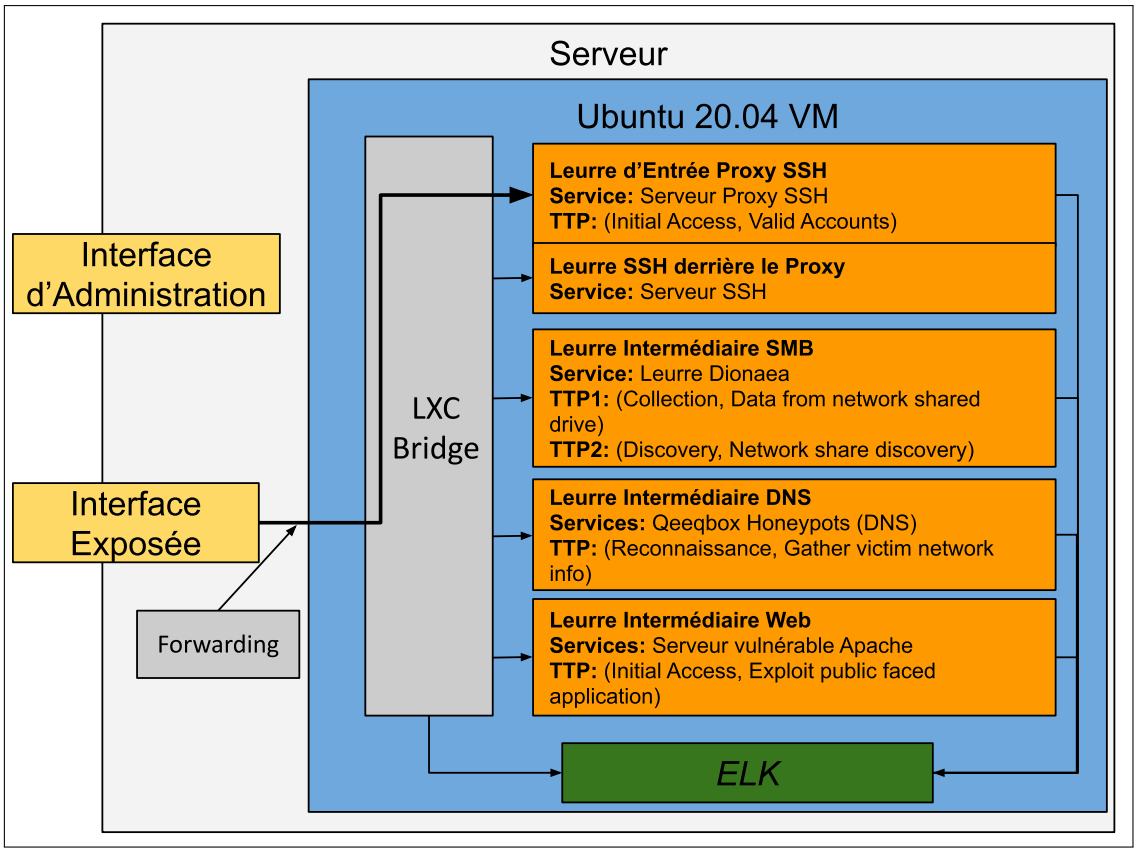

Figure 4.4: Schéma de notre implémentation. En jaune, les deux interface permettent de connecter le réseau de leurres à l'extérieur. En Orange, on représente les leurres sous la forme de conteneurs LXC. En gris, on retrouve les éléments de réseau qui permettent le fonctionnement de notre implémentation. En vert, on représente les éléments de supervision, en particulier la pile ELK qui permet la centralisation des données.

Notre premier leurre en position d'entrée expose via un proxy un serveur SSH configuré avec plusieurs noms d'utilisateur et mots de passe faibles. Nous avons fait ce choix en remarquant que lors de notre première expérience, les attaquants avaient très majoritairement ignoré notre serveur web, se focalisant presque exclusivement sur le SSH. Nous exposons à la suite de cela trois leurres au niveau intermédiaire. Nous limitons ainsi la profondeur du déplacement latéral de l'attaquant, mais nous augmentons sa liberté au sein de notre réseau de leurre. Ceci nous permet d'analyser quelles sont ses cibles privilégiées à l'intérieur de notre réseau. Sur ces trois leurres, deux sont réutilisés de la première expérience, et nous avons ajouté un nouveau leurre simulant un service SMB exposé.

Pour nos trois nouveaux conteneurs, nous sommes partis d'une image de base intégrant Ubuntu Server 22.04, la version à jour au moment de notre expérience.

Leurre SSH

Comme pour la première expérience, le leurre d'entrée propose une tactique d'accès initial via un serveur SSH avec des identifiants vulnérables. Comme pour le premier cas, l'accès initial est offert pour le leurre, mais aussi pour le réseau tout entier. La tactique accessible par les attaquants est *"Valid Accounts"*, via un brute-force du mot de passe.

On peut représenter notre leurre avec notre modèle ainsi:

- Honeypot = (TTP_i , Position , SupervisionLeurre)
- Position = Pos $\wedge$ PR
- Pos = Entrée
- PR = $\emptyset$
- $TTP_1 = (T_1, t_1)$ = (Initial Access, Valid Accounts)
  - t_1 = CAPEC-49: Password Brute Force
- SupervisionLeurre = SC $\wedge$ SG
  - SC = Journalisation par proxy Cowrie
  - SG = ELK Stack

En ce qui concerne son implémentation, ce leurre est bâti autour de deux composants. Le premier est le conteneur intégrant le serveur SSH en lui-même. Dans celui-ci, l'attaquant pourra mettre en place toutes les commandes qu'il désire, et utiliser des techniques de rebond pour accéder aux autres équipements. Cet équipement n'est néanmoins pas accessible depuis l'extérieur du réseau. Pour y avoir accès, l'attaquant doit passer par un proxy, sous la forme d'un serveur Cowrie [3], qui est installé sur un second conteneur dédié. Cet outil open source permet de déployer simplement un proxy SSH pour rediriger les commandes des attaquants vers le système réel. Il permet de plus d'avoir une journalisation détaillée des commandes et des tentatives d'authentification sous la forme de fichiers JSON. Dans la mesure où les attaquants accèdent aux leurres à l'intérieur de notre réseau via rebond sur le leurre SSH, la journalisation du proxy joue un rôle central dans la supervision générale.

En plus d'améliorer la supervision du leurre, l'utilisation d'un proxy séparé du service réel offre l'avantage de permettre une remise à zéro plus facile du leurre. En effet, pour permettre une analyse pertinente, il est important que les attaquants soient confrontés au même leurre indépendamment du moment de leur connexion. Du fait que les attaquants précédents ont eu accès au leurre, ils on pu le modifier, altérant l'état dans lequel les attaquants futurs le rencontreront. Pour remédier à cela, nous avons mis en place un mécanisme de remise à zéro des conteneurs. De façon périodique, le conteneur intégrant le service SSH réel se remet à un état initial. On utilise pour cela les fonctionnalités de snapshot et de rétablissement offertes par LXC. Cette solution n'est pas entièrement satisfaisante, mais c'est celle que nous avons choisie. Une autre possibilité aurait pu être de créer un conteneur dédié à chaque connexion sur le proxy, mais nous craignions de cette façon de surcharger notre équipement, et

donc de ne pas pouvoir assurer l'exposition continue de notre leurre.

Un avantage qu'offre le proxy par rapport à un démon SSH réel est qu'il permet de configurer avec finesse les couples user:password acceptés sans avoir à créer de nombreux comptes utilisateurs. La configuration permet par exemple d'accepter un couple User:Password, mais aussi n'importe quel mot de passe avec un user défini, ou l'inverse (*:Password ou User:*) On pourrait le configurer pour qu'il accepte toutes les tentatives de connexion, mais cela aurait un impact fort sur le réalisme et l'attractivité du leurre, et risquerait de surcharger nos capacités de monitoring et d'analyse. Ce choix d'implémentation à des conséquences importantes. En effet, beaucoup d'attaquants expérimentés font volontairement le choix d'espacer les différentes étapes de leurs opérations. Ceci leur permet d'échapper plus facilement à la supervision et au mécanismes de leurre. Nous acceptons donc le fait de ne pas pouvoir capturer d'attaques "lentes" pour pouvoir assurer la stabilité de notre environnement.

Nous utilisons IPTables pour rediriger l'intégralité du trafic du port standard 22 vers le port 2222 où écoute le proxy Cowrie. Ceci nous permet de ne pas affecter l'utilisation des ports standard et d'assurer que notre proxy SSH est le plus réaliste et accessible possible en s'exposant sur le port 22.

La supervision du proxy est directement intégrée à Cowrie. Nous utilisons simplement Filebeat pour transmettre à notre pile ELK une version traitée de la journalisation de base de Cowrie.

On peut retrouver un schéma du fonctionnement du proxy dans la figure 4.5.

Nous avons configuré des identifiants valides pour notre proxy. Tout attaquant entrant ces identifiants a accès à notre leurre SSH. Pour cela, les attaquants doivent entrer n'importe quelle combinaison des noms d'utilisateur et des mots de passe suivants:

Noms d'Utilisateur : admin, test, user, root
Mots de Passe : marseille, azertyuiop, bonjour, Password123

Nous avons choisi des noms d'utilisateur usuels, souvent présents dans les systèmes, ainsi qu'une combinaison de mots de passe clairement francophones et un anglophone très faibles.

Leurre SMB

Pour la mise en place du leurre SMB, comme pour les autres leurres, nous avons été confrontés au même choix que pour les autres. Est-il plus judicieux d'exposer aux attaquants un service réel monitoré, ou un outil dédié à la déception ?

Dans le premier cas, on obtient un réalisme optimal et on assure une forte attractivité pour les attaquants opportunistes ou expérimentés. Néanmoins, cette approche nécessite d'identifier clairement une vulnérabilité à exposer et à

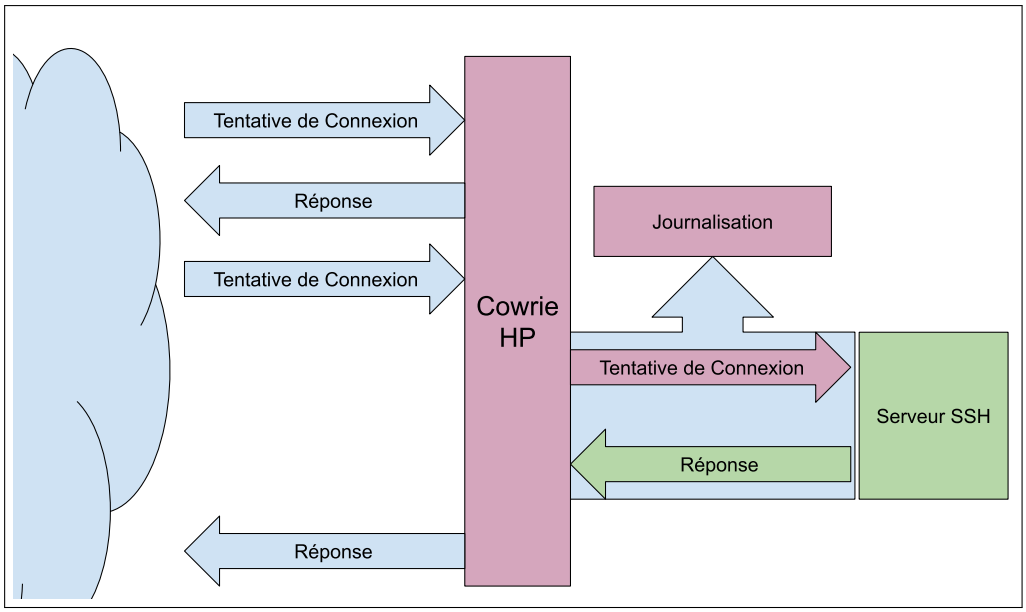

Figure 4.5: Description schématique du fonctionnement de Cowrie. Le proxy gère la phase d'authentification en journalisant les données entrées par le client. Une fois l'utilisateur authentifié, le proxy journalise puis transmet les commandes au serveur intégrant le démon ssh et renvoie à l'attaquant les réponses.

superviser. En faisant ainsi, on s'expose à introduire des vulnérabilités qui nous seraient passées inaperçues.

A l'inverse, on peut choisir d'intégrer un outil dédié à la déception et au déploiement de leurres. Cette solution offre beaucoup de facilités pour l'intégration, la supervision et la robustesse des leurres.

Dans le cadre de notre implémentation, nous avons longtemps hésité entre un leurre intégrant un démon SMB réel, en version 4.15.0 vulnérable à la CVE-2021-44142 et le déploiement de Dionaea, outil dédié à la déception. Nous avons finalement opté pour l'utilisation de Dionaea, en nous basant sur son utilisation dans plusieurs travaux de recherche par le passé.

Pour l'implémentation de ce leurre, nous installons l'outil Dionanea au sein d'un de nos conteneurs. Nous cherchons à réduire le périmètre de notre leurre, prévu pour simuler de nombreux services, tout en soignant la supervision. On désactive donc tous les autres modules que le module "SMB".

L'utilisation de cet outil pose néanmoins un problème concernant la discrétion des leurres. Lorsque l'on utilise l'outil de scan nmap pour identifier le type de service SMB exposé, on remarque que le leurre est identifié en tant que tel. On peut voir ceci dans la figure 4.6.

Cet élément est assez disqualifiant, rendant notre leurre immédiatement identifiable et donc évitable par les attaquants. Pour essayer de remédier à cette

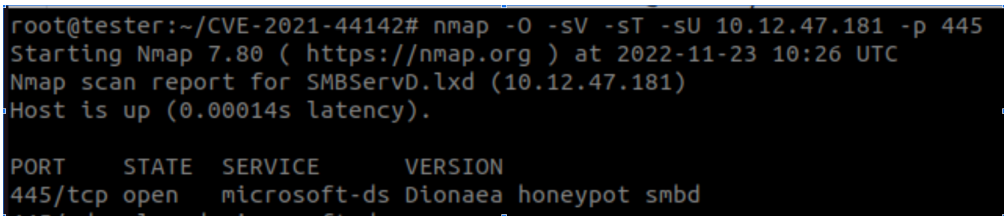

Figure 4.6: Sortie de la commande NMAP cherchant à identifier la nature du service exposé sur le port 445 de notre conteneur. On constate que la version détectée est clairement identifiée comme un leurre par l'outil, dévoilant la nature de notre équipement.

situation, nous avons cherché à identifier comment nmap et les nombreux outils qui en dépendent identifient les services. Cette identification se fait par correspondance entre les éléments retournés par les services et une base de connaissances intégrée dans nmap.

Il nous fallait donc modifier le comportement de Dionanea au moment de l'établissement de la connexion en SMB pour le rendre différent du comportement attendu par nmap. Nous avons identifié dans le code source de Dionaea les chaînes de caractère à transmettre au client lors de l'établissement d'une connexion et avons modifié cette chaîne de caractères de façon à la rendre similaire mais différente. De cette façon, nmap n'est plus capable d'identifier la version de notre service SMB.

De la même façon que Cowrie, Dionaea est théoriquement capable de produire des logs au format JSON. On peut donc utiliser Filebeat pour formater et transmettre le contenu de ces logs vers SLK, comme pour notre leurre SSH. En pratique, cette méthode de journalisation manque de stabilité. Nous employons donc une méthode différente en journalisant nos données sous la forme de base MySQL. A intervalles réguliers, on extrait les données de cette base pour les retranscrire en JSON et les transmettre vers notre agrégateur de données.

On peut représenter notre leurre avec notre modèle ainsi:

- Honeypot = (TTP_i , Position , SupervisionLeurre)
- Position = Pos $\wedge$ PR
- Pos = Intermédiaire
- PR = Accès au leurre d'entrée
- $TTP_1 = (T_1, t_1)$ = (Collection, Data from Network Shared Drive)
- $TTP_1 = (T_2, t_2)$ = (Discovery, Network Share Discovery)
  - $t_1 = t_2$ = CAPEC-643: Identify Shared Files/Directories on System
- SupervisionLeurre = SC $\wedge$ SG
  - SC = Journalisation des commandes au niveau du leurre d'entrée
  - SG = ELK Stack

Supervision et autres leurres

Pour la supervision, nous utilisons à nouveau un conteneur dédié intégrant une pile ELK. Cet outil nous permet de visualiser les données en temps réel et d'accéder à nos données en ligne. La configuration de ce conteneur est assez proche de celle de la première expérience. Les données des différentes sources de supervision générale sont ingérées et présentées.

Pour assurer la continuité du fonctionnement de notre leurre, et la sécurité des données collectées, nous avons intégré au sein de notre machine virtuelle deux scripts qui s'exécutent régulièrement. Le premier script s'exécute de façon aléatoire toutes les quelques minutes pour restaurer le leurre SSH à une snapshot initiale définie, comme décrit dans la section sur ce leurre. Le second script est exécuté tous les jours à minuit. Son objectif est de créer quotidiennement une version exportée du conteneur ELK, qui vient chaque jour écraser la précédente. Ceci permet de la télécharger et de se prémunir contre un risque hypothétique d'échappement complet et de perte d'information sur le conteneur ELK. Les mises à jour ne sont pas incrémentales, chaque sauvegarde écrasant la précédente. En complément de ce script, il faut mettre en place sur un autre équipement une méthode pour récupérer les sauvegardes de façon régulière en passant par l'interface d'administration. Ceci nous permet de nous assurer que même en cas de dysfonctionnement catastrophique de notre leurre, la perte de données n'excède pas 24 heures.

Concernant les autres leurres, les leurres Web et DNS sont les mêmes que pour notre première implémentation, uniquement avec une position différente, et avec une supervision assurée par le proxy SSH.

Chapter 5

Analyse des Résultats

5.1 Résultats de la première expérience

5.1.1 Généralités

Le déploiement de notre première expérience a duré 17 jours, entre le 20 juin et le 6 juillet 2022. Pendant cette période de temps, notre système agrégeant les différentes supervisions a enregistré 1.5 million d'événements. Les différents événements sont composés d'un bruit constant d'activité ponctué de pics plus importants.

Ce nombre d'événements est relativement cohérent avec d'autres chiffres obtenus par des expériences passées. On remarque par exemple que McCaughey [51] et Ryandy [41] collectent respectivement 23000 et 27400 évènements par jour en 2017 et 2020, là où nous en collectons 88000 avec trois services exposés. Nos résultats sont néanmoins plus éloignés d'autres expériences. Il est difficile de comparer de façon immédiate ces chiffres, car les moyens de supervision sont plus ou moins verbeux. Dans certains systèmes, un événement représente une attaque, tandis que dans d'autres, une attaque peut soulever un grand nombre d'évènements. Néanmoins, l'ordre de grandeur de nos résultats semble cohérent avec d'autres travaux.

Le premier constat de notre expérience est qu'aucun attaquant n'a compromis notre leurre d'entrée. Il n'y a pas eu d'accès initial ni de rebond effectué à l'intérieur de notre réseau de leurres. C'est pour cela que toutes les données que nous sommes capables d'analyser sont celles issues des tentatives de connexion à notre leurre d'entrée, exposant un service SSH et un serveur Web vulnérable.

Au niveau de notre supervision, chaque événement collecté correspond à une nouvelle connexion TCP ou à l'impact de celle-ci sur le leurre d'entrée. Notre supervision collecte aussi l'entête IP et TCP des connexions effectuées, offrant une information précieuse pour l'analyse des 1492319 évènements.

IP	Nombre d'événements	Attribution
61.128.X.X	113610 - 7.6%	Chinanet-1
51.142.X.X	38480 - 2.6%	Microsoft (UK)
116.131.X.X	22794 - 1.5%	China United Networks
54.38.X.X	21265 - 1.4%	OVH (France)
89.248.X.X	18830 - 1.3%	Recyber
61.177.X.X	15268 - 1%	Chinanet-2

Table 5.1: Liste des adresses IP représentant plus de 1% du trafic collecté. Pour chaque adresse, on liste le nombre d'événements et l'attribution présentée dans les registres WHOIS.

Acteur	Nombre d'événements	Nombre d'IPs
Chinanet-1	113 610	1
Recyber	42 381	33
Microsoft (UK)	38 480	1
Internet Solutions & Innovations LTD (ISI LTD)	26 834	8
China United Networks	22 794	1
OVH (France)	21 256	1
Chinanet-2	15 268	1
Censys	14 122	182

Table 5.2: Liste des acteurs identifiés en fonction de leurs réserves d'adresses IP et représentant plus de 1% des événements. On remarque deux types d'acteurs: ceux qui concentrent leurs activités dans quelques adresses, et ceux qui diluent leur activité en utilisant parfois plus d'une centaine d'adresses différentes.

Notre analyse de données se fait en trois étapes. Premièrement, nous discutons de la distribution temporelle des attaques ainsi que des adresses IP d'origine. Ensuite, nous étudions les ports et les services ciblés par les attaquants au sein de notre leurre. Dans un troisième temps, nous analysons spécifiquement les tentatives de connexion au service SSH puis discutons de l'extraction d'IoC pertinents grâce à notre expérience.

5.1.2 Acteurs et Attaquants

Une première donnée intéressante de notre supervision concerne les adresses IP sources des attaques. Parmi tous nos événements, nous avons pu extraire 35971 adresses IP uniques, avec des niveaux d'activité variables. Certaines sont très actives, tandis que d'autres n'apparaissent que quelques fois. Nous listons dans la table 5.1 les six adresses les plus actives, représentant chacune plus de 1% des événements, et leurs attributions. Nous sommes capables d'identifier parmi les adresses actives que certaines appartiennent aux mêmes plages et aux mêmes organisations. On peut supposer qu'elles sont utilisées par une même entité pour mener des attaques à une échelle plus importante via plusieurs machines ou interfaces. On liste ces acteurs dans la table 5.2.

On remarque immédiatement une adresse IP très dominante dans notre jeu de données. L'adresse attribuée à l'acteur "China Telecom CHINANET Chongqing Province Network" est de loin la plus présente dans nos événements. Il semble que cette adresse soit liée à un datacenter de la région du Chongquing en Chine. Les acteurs les plus représentés sont principalement des hébergeurs et des fournisseurs d'accès. Ceux-ci sont assez prévisibles, et nous nous attendions à recevoir du trafic

malveillant depuis ce genre d'organisations. A l'inverse, la présence de RECYBER et de Censys dans nos huit plus gros acteurs est plus surprenante.

En analysant l'entrée WHOIS des adresses IP appartenant à RECYBER, nous trouvons le texte suivant dans le champ "remarques":

> *This net-block is not trying to hack you, we are only scanning for LEGIT purposes ONLY. This scanning is done by multiple security organizations. Please use https://www.recyber.net/opt-out to have your ip-address and/or netblock/as number white-listed and excluded from this project. If you have any further questions please contact email@recyber.net*

Ceci semble indiquer que cet acteur n'est pas un attaquant à proprement parler. RECYBER se présente comme une organisation qui scanne automatiquement internet à finalité de recherche et de renseignement. On peut néanmoins s'interroger sur la nécessité d'effectuer près de 40000 tentatives pour scanner ou cartographier un seul équipement exposant une seule adresse. De son côté, Censys se présente comme une plateforme de "Internet Intelligence for threat hunting and exposure management", et semble remplir une mission proche de celle de RECYBER. On remarquera que là où ces derniers offrent une option pour ne pas subir de scans, Censys ne propose pas cette option.

La distribution temporelle des attaques montre que leur répartition est relativement homogène pour un bruit de fond constant ponctué de pics d'activité. Ces pics d'activité sont essentiellement du fait de 8 acteurs que nous avons isolé. On représente dans la figure 5.1 le nombre de d'événements par jour liés à ces gros acteurs comparé à celui de toutes les autres sources combinées.

On peut se focaliser uniquement sur les attaques menées par ces "gros acteurs", et on présente leur répartition dans la figure 5.2. Les agissements sont de deux types. Certains acteurs, en particulier ceux qui se disent "non malveillants", diluent leur activité, avec un nombre très régulier de tentatives chaque jour. A l'inverse, les acteurs Chinant-1, Microsoft (UK) et China United Networks concentrent toute leur activité sur un ou deux jours, avec un volume très important en comparaison aux autres.

Les adresses IP les plus actives appartiennent donc soit à des fournisseurs d'accès, soit à des hébergeurs ou à des organisations effectuant des scans massifs d'Internet. Il est difficile de déterminer si les adresses appartenant au même fournisseur sont utilisées par les mêmes acteurs. Néanmoins, la similitude des schémas d'attaque semble l'indiquer. Les organismes de recherche affirment que leur trafic n'est généré qu'à des fins scientifiques, mais parmi la grande majorité des scans, nous observons des tentatives plus ciblées sur les ports SSH. Cette activité implique au moins quelques tentatives d'exploitation d'une configuration non sécurisée. Quelle que soit sa motivation, le trafic généré par des organismes de recherche prétendument

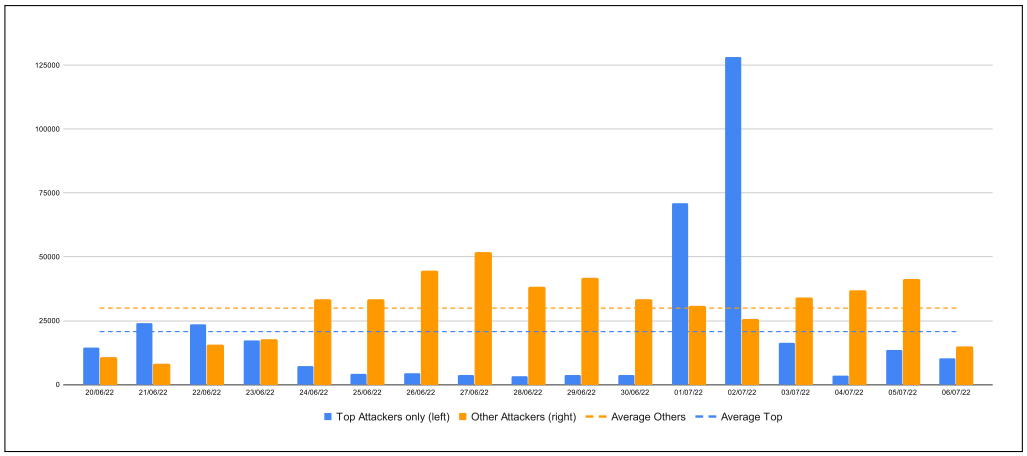

Figure 5.1: Distribution comparée du nombre d'événements par jour liés aux 8 acteurs les plus actifs (Top) et aux autres sources (Other). On remarque que les pics les plus élevés d'activité sont du fait des acteurs les plus actifs, et qu'ils représentent souvent une partie importante du trafic journalier.

légitimes représente un pourcentage significatif des nouvelles connexions que nous avons enregistrées. Les discussions en ligne sur les forums dédiés aux professionnels de la sécurité semblent indiquer que beaucoup d'opérationnels considèrent le trafic de ces scanneurs légitimes comme malveillant et le bloquent au niveau des pare-feu.

5.1.3 Services Ciblés

Dans cette section, nous nous focalisons sur l'analyse des ports et des services ciblés par les attaquants. Cette focalisation nous permet d'inférer sur leurs intentions et leurs objectifs. Du fait que nous supervisons directement au niveau de l'IP, nous avons une visibilité complète sur l'activité des attaquants, même sur les ports où aucun service n'écoute.

En premier lieu, il nous est possible de lister les ports les plus ciblés. On le fait dans la table 5.3. Comme la table l'indique, les services associés à l'authentification à distance sont majoritairement ciblés, essentiellement par des attaques de type brute-force. On mesure 56% du trafic dirigé vers les ports 3389 et 22. En ce qui concerne ce volume de données, il est cohérent avec les mesures effectuées par les anciennes expériences citées. Comme indiqué dans la section d'implémentation, le serveur RDP était configuré de façon sécurisée, avec une version à jour, afin de réduire notre surface vulnérable au seul serveur SSH. Même si plusieurs CVE liées à RDP ont été publiées pendant le déroulé de notre expérience, on ne détecte aucune activité liée à celles-ci dans notre supervision. La majorité du trafic lié à RDP est lié à une activité opportuniste.

En observant l'activité sur les différents ports des 8 attaquants les plus actifs, nous sommes capables de définir trois profils d'attaquants et d'activité:

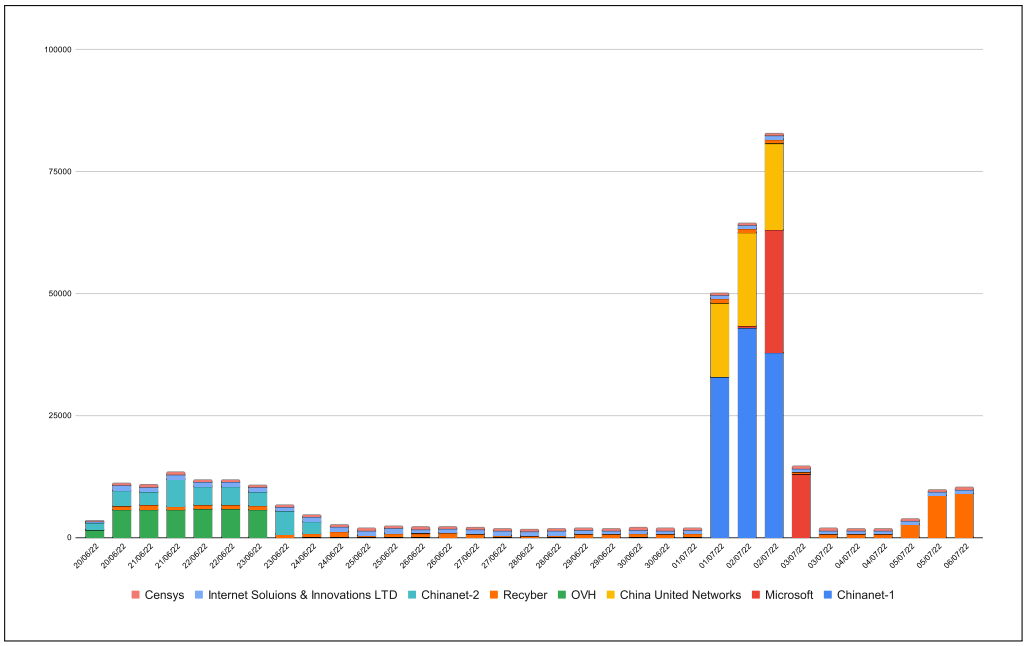

Figure 5.2: Distribution du nombre d'événements par demi-journée liés aux 8 acteurs les plus actifs uniquement. On remarque que certains sont actifs tout au long ou presque de la durée de l'expérience, tandis que d'autres focalisent leurs attaques sur un ou deux jours, avec un très haut volume journalier.

Brute forceurs RDP Deux des trois acteurs les plus actifs, Chinanet-1 et Microsoft UK, effectuent de nombreuses tentatives sur le port 3389 associé au port RDP. Ils montrent toutes les apparences d'une attaque en force brute cherchant à identifier les valeurs d'authentification. Ils n'y sont pas parvenus.

Brute forceurs SSH Les adresses liées aux acteurs China United Network, OVH et Chinanet-2 ont un comportement similaire à celles décrites au-dessus, mais elles se focalisent sur le port 22 associé au protocole SSH. Eux non plus n'y sont pas parvenus.

Scanneurs Les trois derniers acteurs de notre top 8 effectuent des campagnes de scan. Elles ciblent un grand nombre de ports différents, avec peu de

Port	Nombre d'événements	Service associé
3389	228 482	Remote Desktop Protocol
22	222 306	Secure SHell
445	71 592	Microsoft Active Directory/SMB
23	8 010	Telnet
6379	7 660	Redis Server
2222	6 136	Alternative SSH
80	3 341	HTTP
8080	2 149	HTTP
443	2 664	HTTPS
1433	2 156	SQL Server

Table 5.3: Nombre d'événements par port pour les dix ports les plus ciblés, ainsi que leur service associé. Les ports associés à des services d'authentification distante tels que RDP et SSH sont massivement ciblés par rapport à tous les autres.

tentatives pour chacun. Leur objectif n'est pas aussi évident à identifier que pour les autres. Sont-ils motivés par un motif légitime de recherche comme ils l'affirment, ou cherchent-ils des informations pour poursuivre avec des attaques ciblées? Les expériences sur le déploiement de leurres montrent souvent ce genre de comportement, mais les scanneurs qui nous ciblent couvrent énormément de ports. Dacier [20] rapporte 188 ports analysés en 2012, là où nos scanneurs en ciblent plusieurs dizaines de milliers.

Les données sur les ports et la classification des acteurs montrent que les tactiques que nous avons voulu mettre en place pour les attaquants, *Accès Initial* et *Reconnaissance*, sont exploitées. Les techniques que nous pouvons identifier sont *Active Scanning* pour la reconnaissance. Pour l'accès initial, *Brute-force* et *Valid Accounts* sont présents. Nous constatons donc un usage effectif par les attaquants des TTPs que nous cherchions à permettre.

On remarque aussi que les brute forceurs sont entièrement focalisés sur le service qu'ils ciblent, ignorant tous les autres ports exposés.

En analysant les données au niveau du réseau, on peut constater un comportement curieux. Là où la majorité du trafic TCP est originaire de ports source aléatoires dans la plage dédiée, certains attaquants se cantonnent à un petit nombre de ports source. Ceci semble être la conséquence de l'utilisation de *masquerading*, impliquant que les opérations sont effectuées par plusieurs machines derrière un même équipement réseau. Par exemple, presque tout le trafic provenant des scans de RECYBER sont originaires des ports 46194 et 43304.

Ces résultats nous permettent de pointer le fait que dans l'implémentation de leurres, les éléments supposément hors du périmètre d'analyse le sont de fait. A moins de mettre en place des procédures de segmentation totales, ce qui peut être rendu impossible par les spécificités techniques d'implémentation. Ceci entraîne donc la nécessité, comme dans notre cas, de superviser la supervision dans les cas où elle ne peut totalement être segmentée des services exposés.

5.1.4 Bruteforce SSH

Concernant la supervision des attaques sur le SSH, nous collectons les noms d'utilisateur employés. Du fait de la conception même du démon SSH, il nous était impossible de journaliser les entrées de mots de passe. Parmi les près de 300k événements liés au SSH, nous parvenons à identifier 24782 noms d'utilisateurs unique. Le nombre de noms d'utilisateurs différents est proche de celui d'expériences plus anciennes, telles que celles de Alata [28]. Une différence fondamentale réside dans le fait qu'il a obtenu 249k événement en 180 jours, là ou l'on dépasse ce nombre en 17 jours. Parmi les 40k noms d'utilisateurs employés chez Alata, 340 réussissent à se connecter, là où aucun attaquant n'y est parvenu dans notre expérience. Les noms d'utilisateurs employés plus de mille fois sont listés dans la table 5.4 et ils sont comparés à d'autres expériences.

Username	Notre Expérience	Melese 2016[1]	Koniaris 2013[35]	Nicomette 2011[40]	Alata 2006[28]
root	157713 - 63.3%	17553 - 53.7% (1)	10523 - 45.2% (1)	85426 - 15.5% (1)	34251 - 13.8% (1)
admin	17133 - 6.9	3409 - 10.4% (2)	207 - 0.9% (5)	15556 - 2.8% (2)	4007 - 1.6% (2)
user	2524 - 1%	1037 - 3.2% (4)	121 - 0.5% (7)	2165 - 0.4% (5)	1247 - 0.5% (4)
test	2492 - 1%	740 - 2.3% (6)	273 - 1.2% (4)	5615 - 1% (3)	3109 - 1.3% (3)
ubuntu	1972 - 0.8%	-	-	-	-
1111	1366 - 0.5%	-	-	-	-
oracle	1173 - 0.5%	-	319 - 1.4% (3)	1479 - 0.3% (7)	870 - 0.4% (8)
ftpuser	1074 - 0.4%	352 - 1.1% (10)	-	-	-
postgres	1029 - 0.4%	-	120 - 0.5% (8)	-	834 - 0.3% (9)

Table 5.4: Nombre de tentatives et rang des noms d'utilisateurs dans plusieurs expériences avec des leurres SSH.

Là où notre leurre était configuré avec le nom d'utilisateur user, nous remarquons qu'il n'a été testé que dans 1% des tentatives des attaquants, ce qui est une explication possible pour l'absence de succès des attaquants. Néanmoins, les principaux noms d'utilisateurs testés par les attaquants sont ceux auxquels ont pouvait s'attendre. Ils sont majoritairement soit des noms usuels présents sur beaucoup de machines (root, admin, test, etc.), soit des noms de produits ou de services (ubuntu, oracle, etc.). On remarque une dominance très claire du nom *root*.

Nous pouvons constater dans nos données un comportement curieux et inattendu. En général, les attaquants qui emploient le bruteforce utilisent des attaques par dictionnaire. Pour cela, ils choisissent un ou plusieurs noms d'utilisateurs et énumèrent pour celui-ci un ensemble de mots de passe contenus dans une liste, le dictionnaire. Parmi nos attaquants, nous constatons que certaines adresses IP, plutôt que d'énumérer les mots de passe pour un même nom d'utilisateur, semblent énumérer les noms d'utilisateur eux-mêmes. Ils se contentent de faire un essai de connexion pour chaque username. Nous avons au début suspecté une méthode pour identifier les noms d'utilisateurs valides. En effet, certains services d'authentification mal conçus délivrent un message différent en cas d'utilisateur inconnu ou d'identification invalide, permettant de découvrir les noms d'utilisateur légitimes. Nous avons donc testé notre implémentation de OpenSSH et ce n'est pas le cas pour celle-ci. Notre meilleure hypothèse pour expliquer ce comportement reste donc l'erreur humaine. Il est possible qu'un utilisateur ait mal exécuté une commande dans un outil dédié au bruteforce, provoquant une énumération du champ erroné. Cette hypothèse, couplée au type d'attaque le plus présent, est un indicateur du faible niveau de technicité et de motivation de la part des attaquants ciblant notre leurre SSH.

En analysant les détails des deux attaques de bruteforce provenant des réseaux Chinanet, on peut remarquer que les attaques contre les leurres RDP et SSH sont relativement simultanées. Nous présentons la temporalité des deux attaques dans la figure 5.3. Nous ne pouvons pas affirmer qu'il s'agit d'une attaque coordonnée par un même acteur avec deux ensembles d'IP, mais la proximité des événements et l'intensité des attaques peut le laisser penser.

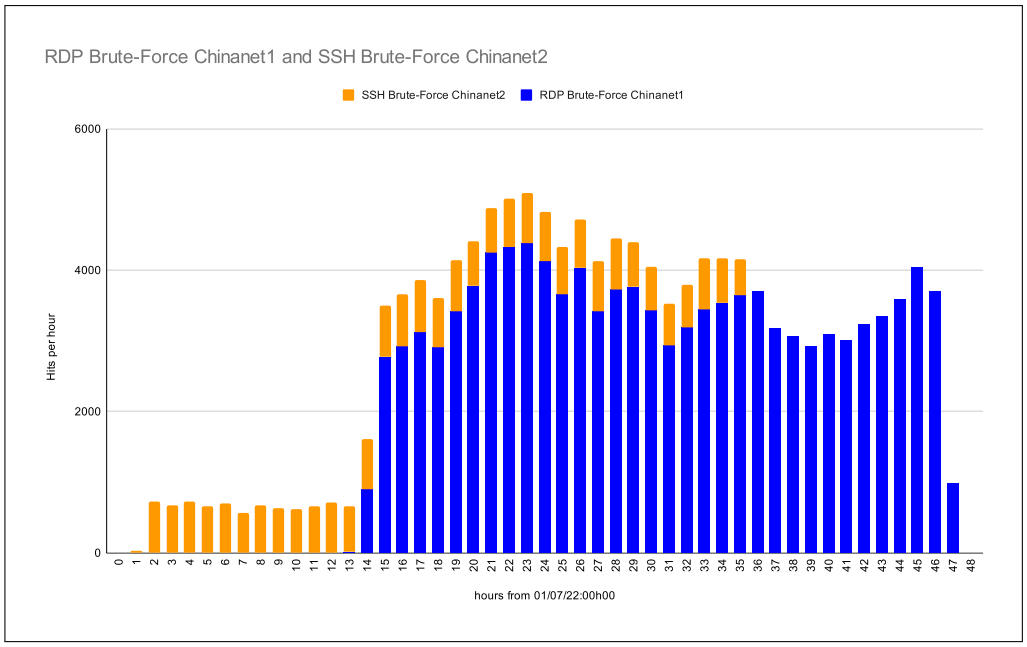

Figure 5.3: Distribution du nombre d'événements de bruteforce heure par heure de la part des IP attribuées à Chinanet-1 et Chinanet-2. On remarque un grand nombre d'événements dans les 48h suivant le premier juillet 2022.

5.1.5 Collecte d'indices de compromission

Un objectif secondaire de notre expérience est de tester la capacité de notre système de leurres à générer des indices de compromission. Pour cela, nous espérions collecter un maximum d'éléments distincts de la part des attaquants. En pratique, nous ne sommes parvenus à collecter que des adresses IP, qui sont des indices de compromissions de faible valeur, bien que valables.

Pour cela, nous avons cherché à analyser les différentes adresses IP que nous avons pu collecter. Ceci nous a permis de déterminer si elles sont connues par la communauté et catégorisées comme malveillantes, ou si nous sommes les premiers à les détecter. Nous avons donc utilisé l'outil communautaire Virustotal [6] pour analyser ces adresses IP. Ce site est un agrégateur qui permet d'analyser des fichiers, des IP ou d'autres indicateurs à l'aide de 94 (à l'heure de notre analyse) outils et antivirus. Il catégorise les adresses en quatre classes: *unknown, suspicious, malicious et harmless.* Nous comptabilisons comme des résultats positifs les classifications suspicious et malicious. Nous analysons les cinq adresses IP les plus actives se livrant au bruteforce sur notre leurre, et nous présentons les résultats de ces analyses dans la table 5.5. Celle-ci montre que seuls les acteurs chinois sont identifiés comme malveillants, mais uniquement par une très faible minorité des outils d'analyse. Il semble peu judicieux de considérer ces adresses comme malveillantes à la seule vue de ces analyses, tant les résultats sont peu affirmatifs. Ceci semble indiquer que l'emploi d'adresses IP en tant qu'indice de compromission n'est pas très efficace. L'usage par les

Addresses IP	Attribution	Résultat et Commentaires (07/2022)	Résultat et Commentaires (12/2023)
61.128.X.X	Chinanet1	1/94 - 1 Malware	0/88 - No detection
51.142.X.X	Microsoft	0/94 - No detection	0/88 - No detection
116.131.X.X	China United Networks	4/94 - 3 Malware, 1 Malicious	1/88 - Malicious
54.38.X.X	OVH	0/94 - No detection	0/88 - No detection
61.177.X.X	Chinanet2	4/94 - 1 Malware, 3 Malicious	5/88 - 1 Malware, 4 Malicious

Table 5.5: Résultats de l'analyse des adresses IP des brute-forcers sur Virustotal. Ces adresses IP ne sont pas identifiées comme malveillantes par les outils.

attaquants d'adresses variables, notamment de fournisseurs d'accès ou de data centers, permet à leur empreinte de rester sous les seuils de détection. Du fait de la volatilité des adresses IP en tant qu'indices de compromission, nous comparons les résultats obtenus lors de notre analyse avec ceux obtenus lors de notre rédaction pour vérifier leur évolution dans le temps, comme indiqué dans la table 5.5.

5.1.6 Discussion de la première expérience

Bien que nous ayons obtenu des résultats intéressants, nous sommes capables d'identifier plusieurs limites à notre déploiement pour expliquer ses défauts.

La principale limite de notre expérience est l'absence de compromission de notre leurre par des attaquants. L'analyse de nos données montre que malgré les failles que nous avons intégré, aucun attaquant n'a réussi à gagner un accès initial dans notre leurre d'entrée, et par conséquence dans notre réseau. Ceci est du à plusieurs facteurs. Premièrement, notre choix d'identifiants n'était pas le plus judicieux. Le nom d'utilisateur que nous avons choisi n'a pas été essayé par les attaquants qu'un millier de fois, ce qui est très faible pour une attaque par bruteforce. Concernant notre mot de passe, notre choix, bien que motivé, s'est avéré sans doute trop géographiquement restrictif. Peut-être que les attaquants ne se préoccupent pas de la localisation des serveurs à l'heure de tester des mots de passe. Le choix de plusieurs mots de passe, dont certains moins marqués comme "français" aurait sans doute été plus judicieux. Il convient donc pour une seconde expérience de mettre en place plus de noms d'utilisateurs attractifs, et des mots de passe plus "internationaux".

Une autre raison de cette absence de compromission peut se retrouver dans le fait que notre adresse IP est peu attractive. Nous avons fait héberger notre expérience chez un hébergeur, et rien ne permettait à notre équipement d'apparaître plus intéressant aux yeux des attaquants que n'importe quel autre équipement accessible en ligne. Pour notre deuxième expérience, il faudrait mettre en place notre leurre dans un périmètre plus attractif.

Une limite importante de notre supervision concerne le fait que nous observons le trafic entrant, mais pas le trafic sortant. Néanmoins, du fait qu'aucun attaquant n'est parvenu à accéder à notre réseau de leurres, nous ne sommes pas directement impactés par cette limite.

Une autre limite de notre expérience vient du fait que la version des services exposés dans les leurres est fixée pendant la période de son déploiement. De ce fait, de nouvelles vulnérabilités peuvent émerger pendant la phase d'expérimentation qui ne sont pas couvertes par notre supervision. Nous n'avons pas remarqué de nouvelles vulnérabilités émergentes, mais il faut tenir compte de ce fait pour de potentiels déploiements plus longs.

Une dernière limitation peut être trouvée dans le volume de données collectées. Même s'il est conséquent et permet une analyse manuelle, 1,5 million d'événements ne sont pas suffisants pour effectuer une analyse automatisée efficace. Le volume a été limité par le fait que notre expérience était dimensionnée pour un certain volume de données. Nous avions peu d'indications sur le volume quotidien moyen que nous recevrions, et nous craignions de surcharger notre système en cherchant à générer plus de données.

D'autres travaux pourraient être effectués sur le même modèle de leurres, à quelques différences près. Nous pourrions rendre plus faible la sécurité du leurre d'entrée de gamme, en choisissant des identifiants plus simples et anglophones. Des améliorations supplémentaires pourraient être apportées à la supervision, notamment en surveillant les services supplémentaires ou le trafic sortant.

Toutes ses limites ont été prises en compte dans la mise au point de notre seconde expérience, dont nous décrivons les conclusions dans la section suivante.

Un autre élément important que nous pouvons tirer de l'analyse de notre première expérience est le poids de l'exposition du protocole RDP à nos attaquants. En effet, nous n'avions pas prévu d'intégrer ce protocole à notre leurre. Néanmoins, c'est le protocole qui était fourni par l'hébergeur pour l'administration de notre serveur distant et nous avons remarqué qu'il attirait une grande part des d'attaquants, dans des proportions similaires au service SSH. Ceci nous pousse à considérer que les contraintes propres au déploiement et à l'implémentation des leurres font partie du périmètre du leurre de façon contrainte. Il n'est pas possible d'échapper aux contraintes posées par les implémentations pratiques, et il convient donc de le prendre en compte dans la supervision. Dans le cadre de notre seconde expérience, nous avons essayé de réduire cette empreinte d'implémentation en ne nous basant pas sur le protocole RDP, mais en restreignant au mieux notre périmètre.

5.2 Résultats de la deuxième expérience

5.2.1 Généralités

Notre seconde expérience est déployée au sein de notre propre infrastructure à Télécom sud Paris. Elle fonctionne pendant plusieurs mois, mais un premier état des lieux des données collectées a été fait au bout de deux mois d'exposition, entre le 15 mars et le 15 mai 2023. Pendant cette période de temps, notre système de supervision a collecté 900319 événements, desquels nous avons pu extraire 180255

session SSH et 225206 tentatives d'authentification.

Il est important de relever que du fait de maintenances et pour des raisons techniques, notre système de leurres a été interrompu au cours de l'expérience 5 jours entre le 22 et le 26 mars. Deux autres interruptions ont eu lieu deux jours entre le 30 avril et le 1er mai, et une autre 7 jours entre le 3 et le 9 mai. Nous pouvons affirmer que ces interruptions ne sont pas imputables à des actions d'attaquants, mais uniquement à des opérations d'entretien de l'infrastructure qui nous hébergeait.

Comme indiqué dans la section concernant l'implémentation, nous utilisons une pile ELK pour centraliser nos données de supervision. Cette solution est efficace à l'heure de visualiser et de gérer des grands volumes de données. Néanmoins, elle a un coût en ressources élevé et nécessite de maîtriser son propre langage de requêtes pour y effectuer des analyses poussées. Du fait de notre volume d'événements, il nous a été possible d'exporter nos données à des formats standards, tels que JSON, CSV ou SQL, plus simples à analyser avec des outils classiques. Pour cela, nous avons utilisé l'outil Elastic-Dump [26] pour exporter nos données au format JSONL, un format de texte enrichi ou chaque ligne est un JSON valide. Le volume de ces logs représente 1.2Go de données.

Nous effectuons les étapes dans le traitement de nos données:

- Export de notre base Elasticsearch au format JSONL
- Conversion de notre base JSONL en base SQL
- Retrait des messages superflus ou redondants de la négociation SSH
- Retrait des messages invariants, par exemple ceux relatifs à notre supervision, identiques pour chaque entrée collectant des données d'attaquant.
- Retrait des données générées par notre propre activité, lors de tests effectués au déploiement et lors du suivi de notre expérience.

La première étape de notre analyse de données à été le nettoyage de nos données brutes. Chaque session SSH initiée par un attaquant sur notre proxy laisse plusieurs messages dans notre supervision, et ils ne sont pas tous utiles pour notre analyse. Par exemple, nous avons décidé d'ignorer les spécificités propres à la négociation d'algorithmes cryptographiques, ou les notifications de fermeture de session. Nous avons donc supprimé les lignes de log correspondant à ces événements. Nous avons aussi choisi de supprimer les éléments de log invariants entre toutes les alertes, en particulier ceux issus de notre supervision. Dans le but de pouvoir effectuer des analyses adaptées, nous avons conservé les champs d'information suivants:

Event Type (eventid) : Description du type d'événement collecté par la supervision (login, tentative de connexion, clôture de connexion, etc.)

Source IP (src_ip) : Adresse IP de l'attaquant effectuant la tentative de connexion

	eventid	src_ip	message	session	src_port	timestamp ▾¹
	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre
1	cowrie.session.connect	49.88.112.65	New connection: 49.88.112.65:15757 (157.159.11.139:22) [session: c1bb001c507d]	c1bb001c507d	15757	2023-03-17 23:34:56.98
2	cowrie.client.version	49.88.112.65	Remote SSH version: SSH-2.0-PUTTY	c1bb001c507d	-	2023-03-17 23:34:56.99
3	cowrie.client.kex	49.88.112.65	SSH client hassh fingerprint: 1616c6d18e845e7a01168a44591f7a35	c1bb001c507d	-	2023-03-17 23:34:57.24
4	cowrie.login.failed	49.88.112.65	login attempt [root/987321] failed	c1bb001c507d	-	2023-03-17 23:34:58.60
5	cowrie.login.failed	49.88.112.65	login attempt [root/adamadam] failed	c1bb001c507d	-	2023-03-17 23:34:59.85
6	cowrie.login.failed	49.88.112.65	login attempt [root/ch@ng3m3] failed	c1bb001c507d	-	2023-03-17 23:35:01.11
7	cowrie.session.closed	49.88.112.65	Connection lost after 5 seconds	c1bb001c507d	-	2023-03-17 23:35:02.37

Figure 5.4: Exemple de logs concernant une session, visualisés dans un outil de gestion de bases de données.

Source Port (src_port) : Port d'émission du trafic reçu
Log Message (message) : Message du log contenant les détails de l'alerte
Session Identifier (session) : Identifiant alphanumérique unique à chaque session, permettant de les distinguer individuellement.
Timestamp : Date et heure de l'entrée du log.

Un exemple des données obtenues et visualisables peut se trouver dans la figure 5.4

Une fois ces données restantes isolées, nous en retirons 230Mo de données, sous la forme d'une base MySQL facilement traitable et partageable avec la communauté.

Nous effectuons l'analyse des données obtenues en deux phases. Premièrement, nous analysons quantitativement les différents champs de données. Ensuite, nous analysons qualitativement les données et les alertes soulevées par les attaquants, tout particulièrement ceux qui sont parvenus à accéder à notre réseau de leurres.

5.2.2 Analyse Quantitative

IP et sessions

La première étape de notre étude quantitative concerne les sessions et les adresses IP des attaquants. A partir des 900k événements, nous collectons environs 180k sessions SSH pour 225k tentatives de connexion avec un couple d'identifiants. Ces variations entre les chiffres peuvent s'expliquer, du fait que chaque session SSH laisse plusieurs traces dans les logs (initialisation, négociation du protocole, etc.). De plus, chaque session peut inclure plusieurs tentatives d'authentification. Parmi ces connexions, on constate immédiatement que 206 se soldent par une authentification réussie, ce qui donne un taux de succès de 0.09% pour les attaquants.

Ces sessions sont créees par 3010 adresses IP uniques que nous sommes capables d'identifier. Une analyse statistique rapide nous montre que ces adresses effectuent plusieurs tentatives d'authentification, entre 0 et plus de 27000. En moyenne, chaque adresse effectue 75 tentatives d'authentification. Le nombre de tentatives a

IP Address	Login Attempts and percentages	Attribution	Bloc
124.52.XX.XX	27518 - 12.22 %	LG POWERCOMM Korea	-
61.177.XX.XX	17947 - 7.97 %	Chinanet	bloc 1
218.88.XX.XX	17758 - 7.89 %	Chinanet	bloc 2
61.177.XX.XX	14777 - 6.56 %	Chinanet	bloc 1
61.177.XX.XX	13565 - 6.02 %	Chinanet	bloc 1
218.92.XX.XX	11636 - 5.17 %	Chinanet	bloc 2
61.177.XX.XX	9096 - 4.04 %	Chinanet	bloc 1
185.246.XX.XX	4622 - 2.05 %	w1n ltd Sweden	-
182.138.XX.XX	4080 - 1.81 %	Chinanet	-
49.88.XX.XX	3860 - 1.71 %	Chinanet	-

Table 5.6: Adresse et nombre de tentatives des 10 acteurs les plus actifs ayant ciblé notre expérience.

néanmoins un écart type de 833 et une médiane de 10. On peut donc considérer que ce nombre est très distribué. Beaucoup d'adresses s'attardant très peu sur notre système, et peu d'entre elles investissant des efforts de bruteforce plus significatifs. Une autre source d'étonnement est le faible nombre de tentatives d'une majorité d'attaquants. Notre analyse montre que 75% des attaquants effectuent moins de 30 tentatives d'authentification.

Nous sommes assez surpris par le fait qu'un nombre significatif d'IP, environ un quart, n'effectuent aucune tentative d'authentification. Parmi les adresses qui effectuent toutes les étapes pour la mise en place d'une session SSH, 796 sur 3010 ne tentent aucune authentification, se contentant de couper leur connexion.

A partir de nos observations, nous pouvons distinguer les attaquants qui ciblent notre système de leurre en trois catégories:

Les Opportunistes détectent notre réseau de leurres et essayent de s'authentifier entre une et quelques dizaines de fois, avant de se désintéresser.

Les Scanneurs initialisent une connexion SSH valide, puis se déconnectent sans tenter de s'identifier.

Les Brute-forceurs détectent notre leurre et effectuent un grand nombre de tentatives de connexion dans le but de déterminer les identifiants valides.

Parmi ces adresses IP, comme pour notre première expérience, nous pouvons identifier les 10 adresses les plus actives et nous les présentons dans la table 5.6.

Dans ce top 10, on reconnait assez vite deux blocs d'adresses IP, 61.177.172-3.XX et 218.92.0.XX. Pour ces deux plages, nous détectons respectivement 13 et 22 adresses, qui contribuent pour 29% et 15% des tentatives d'authentification. On remarque aussi que des adresses appartenant au premier bloc ont été identifiées dans notre première expérience, en particulier un adresse liée à *Chinanet* dans la table 5.1.

Parmi ces adresses de notre top 10, 8 sont attribuées à des organismes situés en Chine, une en Corée et une en Suède.

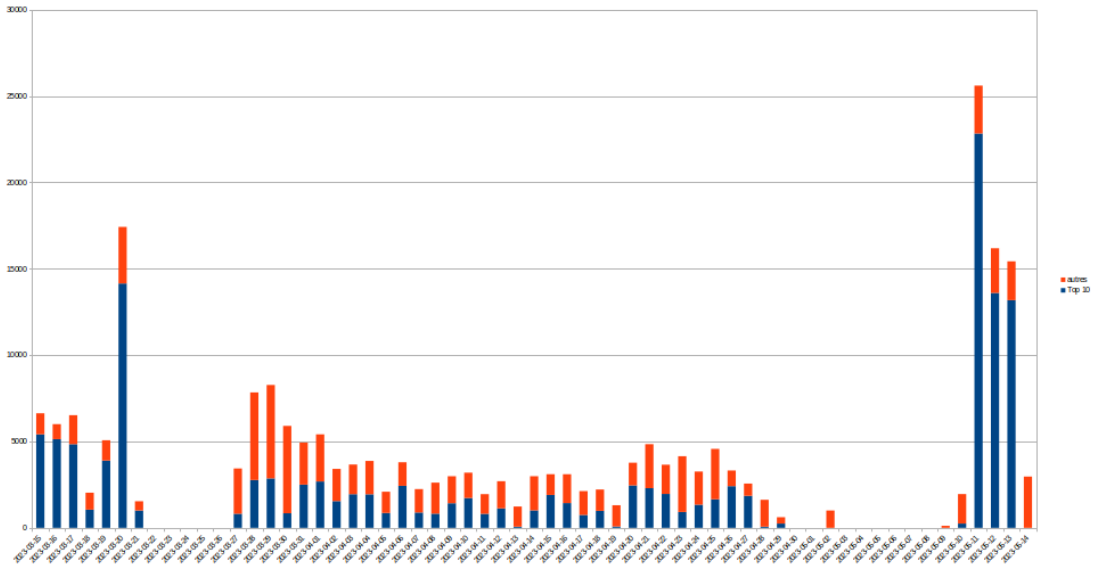

Figure 5.5: Analyse temporelle du nombre d'attaques par jour. On distingue dans la figure les attaques des 10 adresses les plus actives de celles effectuées par les autres adresses.

Analyse temporelle

Nous sommes capables d'analyser la distribution des attaques dans le temps. Afin de distinguer plus clairement leur activité du bruit ambiant, nous séparons le trafic provenant des 10 adresses les plus actives du reste. Nous présentons la répartition des attaques dans la figure 5.5. On y remarque une répartition stable du bruit de fond avec des pics ponctuels d'activité. Dans nos données, on constate les trois interruptions que nous évoquons dans l'introduction de la section.

Noms d'utilisateur

Notre leurre SSH a attiré des attaquants qui ont tenté d'y accéder en s'authentifiant à l'aide d'un nom d'utilisateur et d'un mot de passe. Nous avons journalisé ces deux champs et nous analysons dans cette sous-section les noms d'utilisateur. Parmi les 225k tentatives de connexion, nous pouvons isoler 5330 noms d'utilisateur différents. Ils sont utilisés par les attaquants entre 1 et 162840 fois, avec une distribution très déséquilibrée. En moyenne, chaque nom d'utilisateur est utilisé 42 fois, mais la médiane du nombre d'utilisation est à 2, et 75% des noms d'utilisateurs sont utilisés 4 fois ou moins.

Nous listons dans la table 5.7 les 10 noms d'utilisateurs les plus employés par les attaquants. On remarque immédiatement que le nom "root" est très majoritairement employé, représentant presque trois quarts des tentatives.

Ces données concernant les noms d'utilisateur sont cohérentes, à la fois avec notre première expérience et avec des travaux similaires. La répartition des usernames

Username	Nombre de tentatives	Pourcentage des tentatives
root	162840	72.31 %
admin	8841	3.93 %
user	5703	2.53 %
user2	2317	1.03 %
ubuntu	2230	0.99 %
test	1815	0.81 %
oracle	1646	0.73 %
pi	1174	0.52 %
postgres	1090	0.48 %
ubnt	940	0.42 %

Table 5.7: Détail des 10 noms d'utilisateurs les plus utilisés par les attaquants, avec une très claire dominance de "root".

Username	Tentatives	Première Expérience	Melese 2016[1]	Koniaris 2013[35]	Nicomette 2011[40]	Alata 2006[28]
root	162840 - 72.3%	157713 - 63.3% (1)	17553 - 53.7% (1)	10523 - 45.2% (1)	85426 - 15.5% (1)	34251 - 13.8% (1)
admin	8841 - 3.9%	17133 - 6.9% (2)	3409 - 10.4% (2)	207 - 0.9% (5)	15556 - 2.8% (2)	4007 - 1.6% (2)
user	5703 - 2.5%	2524 - 1% (3)	1037 - 3.2% (4)	121 - 0.5% (7)	2165 - 0.4% (5)	1247 - 0.5% (4)
user2	2317 - 1%	257 - 0.1% (34)	-	-	-	-
ubuntu	2230 - 1%	1972 - 0.8% (5)	-	-	-	-
test	1815 - 0.8%	2492 - 1% (4)	740 - 2.3% (6)	273 - 1.2% (4)	5615 - 1% (3)	3109 - 1.3% (3)
oracle	1646 - 0.7%	1173 - 0.5% (7)	-	319 - 1.4% (3)	1479 - 0.3% (7)	870 - 0.4% (8)
pi	1174 - 0.5%	538 - 0.2% (15)	-	-	-	-
postgres	1090 - 0.5%	1029 - 0.4% (9)	-	120 - 0.5% (8)	-	834 - 0.3% (9)
ubunt	940 - 0.4%	520 - 0.2% (17)	834 - 2.6% (5)	-	-	-

Table 5.8: Nombre de tentatives et rang des noms d'utilisateurs employés par les attaquants dans plusieurs déploiements.

et la prédominance de certains confirment nos suppositions concernant les profils des attaquants. Une majorité d'entre eux énumère un faible nombre de noms d'utilisateurs avant de se désintéresser de notre leurre. A l'inverse, un faible nombre d'entre eux effectuent des attaques massives sur un faible nombre de noms.

Les quatre noms d'utilisateur que nous avons choisi comme valides (root, admin, user et root) représentent environ 80% des tentatives. Nous pouvons donc considérer que, contrairement à notre première expérience, notre choix de nom d'utilisateur est judicieux dans l'objectif d'attirer des attaquants.

Comme pour la première expérience, nous pouvons comparer nos résultats avec plusieurs éléments de la littérature dans la table 5.10. Les résultats semblent encore une fois relativement cohérents, avec une dominance de usernames usuels et de noms de services ou d'applications. Nous comparons ces résultats avec notre première expérience, avec Melese en 2016; Koniaris en 2013 et Nicomette en 2011. Ceci nous permet de faire une comparaison sur 11 ans et dans des contextes de déploiement différents. On constate des similarités dans les résultats, le nom "root" semblant être toujours le choix privilégié des attaquants.

Nous pouvons analyser les noms d'utilisateur spécifiquement utilisés par nos 10 adresses IP les plus actives listées dans la table 5.6. On constate que 9 d'entre eux n'utilisent que le nom "root" pour leurs tentatives. L'adresse 185.246.XX.XX énumère quant à elle 86 noms différents. Ceci confirme la concentration des brute forceurs, énumérant de nombreux mots de passe pour peu de noms d'utilisateur.

Username	Première Expérience	Seconde Expérience	Melese 2016[1]
root	157713 - 63.3%	162840 - 72.3% (1)	17553 - 53.7% (1)
admin	17133 - 6.9%	8840 - 3.9% (2)	3409 - 10.4% (2)
user	2524 - 1%	5703 - 2.5% (3)	1037 - 3.2% (4)
test	2492 - 1%	1815 - 0.8% (6)	740 - 2.3% (6)
ubuntu	1972 - 0.8%	2230 - 1% (5)	-
1111	1366 - 0.5%	21 - 0% (237)	-
oracle	1173 - 0.5%	1646 - 0.7% (7)	-
ftpuser	1074 - 0.4%	691 - 0.3% (11)	352 - 1.1% (10)
postgres	1029 - 0.4%	1090 - 0.5% (9)	-

Table 5.9: Nombre de tentatives et rang des noms d'utilisateurs employés par les attaquants dans plusieurs déploiements, en utilisant notre première expérience comme clé de comparaison

Username	Melese 2016[1]	Seconde Expérience	Première Expérience
root	17553 - 53.7%	162840 - 72.3% (1)	157713 - 63.3% (1)
admin	3409 - 10.4%	8841 - 3.9% (2)	17133 - 6.9% (2)
bnet	3301 - 10.1%	0	-
user	1037 - 3.2%	5703 - 2.5% (3)	2524 - 1% (3)
ubnt	834 - 2.6%	940 - 0.4% (10)	-
test	740 - 2.3%	1815 - 0.8% (6)	2492 - 1% (4)
support	672 - 2.1%	238 - 0.1% (30)	-
guest	647 - 2%	562 - 0.3% (14)	-
ftp	360 - 1.1%	210 - 0.1% (34)	-
ftpuser	352 - 1.1%	691 - 0.3% (11)	1074 - 0.4% (8)

Table 5.10: Nombre de tentatives et rang des noms d'utilisateurs employés par les attaquants dans plusieurs déploiements, en utilisant Melese [1] comme clé de comparaison

Mots de passe

Concernant les mots de passe, leur analyse indique une bien plus grande variété d'usage. Là où nous comptions environs 5000 noms d'utilisateur, nous relevons l'usage de 46069 mots de passe différents. Contrairement à ces premiers, la répartition des mots de passe est bien plus aplatie. Ils sont testés entre 1 et 4815 fois, avec une moyenne à 4.89 et une médiane et un troisième quartile à 2.

La dispersion des valeurs est très grande. Il n'est pas possible d'isoler un mot de passe clairement dominant. Seulement trois d'entre eux représentent plus de 1% des tentatives. Nous pouvons remarquer comparativement que 10.6% des tentatives d'authentification sont effectuées avec le même username et mot de passe.

Les mots de passe les plus utilisés sont listés dans la table 5.11. On remarque qu'ils sont très prévisibles, en général des mots de passe usuels ou des séquences de caractères cohérentes numériquement ou typographiquement (ex:qwerty). Nous constatons aussi que beaucoup d'attaquants font des tentatives d'authentification avec un champ de mot de passe vide.

Il est possible de comparer nos résultats avec certains des travaux avec lesquels nous comparons nos noms d'utilisateur. Toutes les expériences n'incluent pas de supervision pour ce champ, uniquement celles des Melese et de Koniaris. Nous présentons le comparatif dans la table 5.12, et nous inversons les clés de la table dans les deux tables suivantes 5.13 et 5.14. On retrouve dans plusieurs expériences les mots de passe usuels que nous rencontrons dans la nôtre.

Mots de passe	Tentatives	Pourcentage
123456	4815	2.14 %
1234	2253	1.00 %
1	1813	0.81 %
123	1802	0.80 %
(No Password)	1592	0.71 %
password	1589	0.71 %
admin	1581	0.70 %
12345	1231	0.55 %
root	1125	0.50 %
12345678	1090	0.48 %
123456789	923	0.41%
test	895	0.40 %
Password	817	0.36%
qwerty	598	0.27%
ubuntu	547	0.24%
guest	542	0.24%
toor	539	0.24%
1234567	517	0.23%
Admin123	496	0.22%
admin123	456	0.20%

Table 5.11: Détails des 20 mots de passe les plus employés par les attaquants.

Mot de passe	Tentatives	Melese [1]	Koniaris [35]
123456	4815 - 2.1%	712 - 2.2% (4)	1507 - 6.5% (1)
1234	2253 - 1%	688 - 2.1% (6)	338 - 1.5% (4)
1	1813 - 0.8%	-	-
123	1802 - 0.8%	-	213 - 0.9% (6)
(no password)	1592 - 0.7%	-	-
password	1589 - 0.7%	705 - 2.2% (5)	430 - 1.9% (2)
admin	1581 - 0.7%	1564 - 4.8% (2)	-
12345	1231 - 0.6%	-	319 - 1.4% (5)
root	1125 - 0.5%	880 - 2.7% (3)	425 - 1.8% (3)
12345678	1090 - 0.5%	-	-

Table 5.12: Nombre de tentatives de mot de passe effectuées par les attaquants, comparées avec les expériences de Melese et Koniaris.

Mot de passe	Melese [1]	Nos Tentatives	Koniaris [35]
admin	1564 - 4.8%	1581 - 0.7% (7)	-
root	880 - 2.7%	1125 - 0.5% (9)	425 - 1.8% (3)
123456	712 - 2.2%	4815 - 2.1% (1)	1507 - 6.5% (1)
password	705 - 2.2%	1589 - 0.7% (6)	430 - 1.9% (2)
1234	688 - 2.1%	2253 - 1.0% (2)	338 - 1.5% (4)
ubnt	548 - 1.7%	281 - 0.1% (35)	-
support	444 - 1.4%	154 - 0.1% (82)	-
test	415 - 1.3%	895 - 0.4% (12)	166 - 0.7% (7)
user	413 - 1.3%	341 - 0.2% (30)	-

Table 5.13: Nombre de tentatives de mot de passe effectuées par les attaquants, comparées avec les expériences de Melese et Koniaris en utilisant Melese [1] pour clé.

Mot de passe	Koniaris [35]	Nos Tentatives	Melese [1]
123456	1507 - 6.5%	4815 - 2.1% (1)	712 - 2.2% (4)
password	430 - 1.9%	1589 - 0.7% (6)	705 - 2.2% (5)
root	425 - 1.8%	1125 - 0.5% (9)	880 - 2.7% (3)
1234	338 - 1.5%	2253 - 1.0% (2)	688 - 2.1% (6)
12345	319 - 1.4%	1231 - 0.6% (8)	-
123	213 - 0.9%	1802 - 0.8% (4)	-
test	166 - 0.7%	895 - 0.4% (12)	415 - 1.3% (9)
abc123	129 - 0.6%	320 - 0.1% (31)	-
changeme	110 - 0.5%	107 - 0.1% (131)	-
111111	102 - 0.4%	425 - 0.2% (23)	-

Table 5.14: Nombre de tentatives de mot de passe effectuées par les attaquants, comparées avec les expériences de Melese et Koniaris en utilisant Koniaris [35] pour clé.

Chaîne de caractères	Nb de variants	Nb de tentatives
123	6263	63709
admin	611	10917
password	212	4441
root	371	5100
test	182	3238
qwerty	182	1976

Table 5.15: Nombre et usage des mots de passe incluant une chaîne de caractères fixe.

Nous pouvons aussi analyser la longueur des mots de passe qui sont employés par les attaquants. On remarque alors une moyenne à 7.45 et une médiane à 8 caractères. Ces longueurs sont similaires à celles obtenues par Koniaris en 2013. Ces mots de passe possèdent donc en moyenne une entropie entre 25 et 52 bits selon qu'ils emploient uniquement des chiffres ou des lettres, des chiffres, des minuscules et majuscules ainsi que des symboles spéciaux. Dans les deux cas, ces chiffres sont en dessous des recommandations de sécurité usuellement édictées par des organisations telles que l'ANSSI [10] se situant au-dessus de 100 bits. On peut donc considérer que dans notre expérience, les attaquants ciblent principalement des mots de passe faibles.

5.2.3 Analyse Qualitative

La section précédente consistait en une analyse quantitative des différents champs d'information collectés. Dans cette section, nous analysons en détail et qualitativement les informations que nous avons collectées avec notre leurre.

Variations de Mots de passe

Un marqueur fort d'énumération automatique de mots de passe par les attaquants est l'usage de variants d'un même mot de passe. On constate par exemple l'usage de plusieurs altérations d'un même mot de passe. Ces altérations sont par exemple le remplacement d'un caractère par un autre typographiquement similaire, un "a" par un "@" ou un "o" par un "0". Pour identifier l'étendue du phénomène, nous avons recherché des chaînes de caractère fixées au sein des mots de passe collectés, sans nous préoccuper de la casse. Ainsi, nous avons pu quantifier le nombre de mots de passe incluant une même chaîne de caractères avec des ajouts. Nous montrons les trouvailles les plus pertinentes dans la table 5.15. On remarque par exemple que près d'un tiers des mots de passe utilisés incluent la chaîne "123", et des mots tels que "password" ou "admin" sont très répandus.

Sessions Valides

Parmi les 225206 tentatives d'authentification effectuées par les attaquants, 206 se sont soldées par un succès. Tous ces succès ont été réalisés en utilisant 5 couples user:password listés dans la table 5.16. On observe une dominance claire des mots

Identifiants Valides	Nombre de Connexions
root:Password123	77
admin:Password	54
root:azertyuiop	44
admin:Password123	25
root:bonjour	6

Table 5.16: Identifiants valides utilisés pour les authentifications des attaquants.

Sessions	Procédures	Tactiques MITRE
28	Téléchargement et Exécution du malware XORDDoS	Execution, Command and Control
9	Téléchargement d'un binaire obfusqué lié aux binaires Mirai/RapperBot	Command and Control
3+1	Procédure de téléchargement et exécution du malware DOTA3 + 1 interrompue	Execution, Impact
9	cat /bin/echo;	Reconnaissance
1	Acquisition d'information sur le hardware (CPU et RAM)	Reconnaissance
2	Version obfusquée du malware Shellbot avec des commentaires en portugais	Execution, Command and Control

Table 5.17: Description des différentes procédures effectuées par les attaquants.

de passe “root” et “admin” qui s'explique par leur prédominance parmi les noms d'utilisateur utilisés. Ces succès ont été obtenus par 99 adresses IP différentes. Sur celles-ci, 91 ne se sont connectées qu'une fois tandis que 8 adresses se connectent plusieurs fois avec des identifiants valides, de deux à 28 fois dans notre expérience. On remarque aussi que 57 des 99 connexions sont effectuées depuis les plages d'adresses 61.177.173.XX et 61.177.172.XX. La première connexion valide a eu lieu assez tôt, environ 1h50min après le début de notre déploiement et de notre exposition sur internet.

Commandes et Procédés

Bien que 206 connexions à notre leurre aient été réussies, la majorité de celles-ci n'ont pas entraîné d'actions de la part des attaquants. En effet, 54 connexions se sont soldées par des commandes entrées tandis que pour 152 d'entre elles, les attaquants n'en entrent aucune. Pendant ces connexions, des séquences de commandes ont été transmises. Ces séquences de commandes sont rarement uniques, et en les analysant au cas par cas, nous parvenons à identifier 6 procédures distinctes, réalisées entre 1 et 28 fois chacune. Dans la table 5.17, on liste ces séquences en détaillant pour chacune d'elles les techniques de la MITRE ATT&CK correspondantes.

Nous pouvons observer immédiatement que les attaquants qui entrent des commandes se concentrent sur des opérations rapides et automatisées. On constate qu'aucun d'entre eux ne tente de se déplacer dans le réseau de leurres par mouvement latéral. Ils semblent satisfaits d'avoir eu accès au leurre SSH d'entrée et d'avoir compromis un seul équipement. Ce comportement semble cohérent avec l'approche opportuniste que l'on peut attribuer à nos attaquants. Presque toutes leurs opérations se concentrent sur des activités de faible complexité et sur une exploitation immédiate de l'hôte. Par exemple, les attaquants ajoutent notre équipement à un botnet ou installent des logiciels de minage de cryptomonnaies. Le contenu ou la nature de notre équipement ne semble pas avoir d'importance pour nos attaquants, qui sont uniquement intéressés par le vol de ressources matérielles.

Tactiques MITRE ATT&CK	Description
Reconnaissance	Collecte d'informations sur le Hardware
Credentials Acces	Tentatives d'accès par Bruteforce
Command and Control	Installation de malware lié à des botnets
Impact	Vol de ressources hardware, notamment pour le minage de cryptomonnaies
Ressource Development	Téléversement de Malware

Table 5.18: Liste et description des tactiques employées par les attaquants sur notre leurre d'entrée.

En analysant les actions des attaquants grâce à la matrice MITRE ATT&CK, nous observons qu'ils ne cherchent à accomplir que les 5 tactiques que l'on liste dans la table 5.18.

Ces Tactiques couvrent une portion de la matrice MITRE ATT&CK. Sur les 14 tactiques existantes, nous en rencontrons 5. Nous ne nous attendions pas à rencontrer toutes les tactiques dans notre expérience, du fait que nous avons intégré une cyberkillchain déterminée à notre réseau de leurres. Nous discutons des différentes tactiques dans la table 5.19

L'attaquant qui charge un malware que nous identifions comme Rapperbot s'est aussi connecté pour n'entrer que la commande "cat /bin/echo;". D'après plusieurs analystes [7, 30], cette manœuvre permet aux attaquants de lire les entêtes ELF du binaire afin de collecter de l'information sur l'architecture de la machine cible.

Réputation des Adresses IP

En étudiant les adresses IP des attaquants, nous nous sommes demandés si nous étions face à des adresses connues par la communauté pour leurs activités malveillantes. Pour répondre à cette question, nous utilisons, comme pour la première expérience, les fonctionnalités de recherche du site Virustotal[6]. En recherchant des données sur 10 adresses IP, nous calculons un taux de suspicion en divisant le nombre de détections positives par le nombre de vendeurs présents sur le site. On obtient des résultats relativement bas, entre 0 et 14%. Le bruteforce étant une activité très bruyante, nous nous attendions à des résultats plus élevés. En moyenne, nous avons obtenu 7.7% de signalements par adresse, ce qui est faible à nos yeux. De plus, deux adresses représentant plus de 20k tentatives de connexions ne sont pas identifiées par les outils d'analyse. Nous présentons les résultats dans la table 5.20

Les adresses IP étant des indicateurs relativement volatiles, il faut les considérer comme des IoC assez faibles. Néanmoins, nous sommes capables d'identifier pour la plupart des IP que nous détectons un écart entre leur réputation et leur comportement. Parmi les 3010 adresses que nous identifions, une centaine ont un taux de suspicion nul.

Tactiques MITRE ATT&CK	Attendue	Présente	Commentaire
Reconnaissance	✓	✓	Notre système étant exposé sur internet, nous nous attendions a attirer des scanneurs analysant nos ports exposés. Nous les retrouvons dans notre analyse. Nous n'avons pas d'information sur de potentielles recherches complémentaires effectuées au sujet de notre infrastructure par les attaquants.
Ressource Development	✗	✗	Les différentes techniques de cette tactique se déroulent en dehors de notre infrastructure. Du fait de notre supervision, nous ne nous attendions pas à détecter ce genre d'activité, et notre expérience confirme nos attentes.
Initial Access	✓	✓	Nos leurres exposés aux attaquants permettent un accès initial à la fois au leurre d'entrée mais aussi à notre réseau de leurres entier. Nous attendions un accès initial par l'utilisation de comptes valides pour notre leurre d'entrée, et par l'exploitation de vulnérabilités pour notre leurre Web. Nous ne constatons que le premier.
Execution	✓	✗	Nous nous attendions, lors de notre déploiement, à découvrir des attaquants exécutant des malwares ou des scripts sur nos leurres après y avoir gagné accès. Nous constatons que malgré plusieurs malwares versés dans notre leurre, il n'y a pas de tentatives de les executer ou de mettre en place des routines d'exécution automatiques (systemd, cron, etc.)
Persistence	✓	✓	Par le versement d'outils de contrôle à distance associés à des botnets, les attaquants mettent en place des techniques permettant d'assurer la persistance de leur accès.
Privilege Escalation	✗	✗	Nous ne constatons aucune tentative d'élévation de privilèges. En effet, les attaquants tentent majoritairement de se connecter via des comptes nommés "root" ou "admin" et ne semblent pas chercher à augmenter leurs privilèges.
Defense Evasion	✓	✗	Nous nous attendions à observer des tentatives d'évasion des défenses, tel que des tentatives de suppression des historiques ou de prospection à la recherche d'outils de supervision. Nous ne détectons aucune tentative de cela.
Credential Access	✓	✓	C'est la tactique que nous nous attendions le plus à retrouver, de par les attaques par force brute effectuées par les attaquants. Nous détectons comme prévu de nombreuses tentatives.
Discovery	✓	✗	Nous nous attendions à ce que les attaquants pénétrant dans notre réseau de leurres par notre leurre d'entrée tentent de découvrir notre réseau en prévision de mouvement latéraux. Nous ne constatons néanmoins aucune pratique de ce genre, les attaquants s'arrêtant à notre leurre d'entrée et ne montrant aucun intérêt pour la propagation interne.
Lateral Movement	✓	✗	De même que pour la tactique précédente, nous nous attendions à retrouver des tentatives de mouvement latéral témoignant de la volonté des attaquants de se propager dans notre réseau de leurres. Néanmoins, nous ne détectons aucune tentative de cela.
Collection	✗	✗	Nous ne constatons aucune tentative de collection de données, et nous n'en attendions pas du fait du type de leurres déployés dans notre réseau.
Command and Control	✓	✓	Nous nous attendions à collecter des indices d'installation d'outils et de transmission de commandes par les attaquants à notre leurre. Dans les données que nous collectons, nous ne sommes pas capable d'identifier de communications, mais nous pouvons relever l'installation d'outils qui auraient pu permettre à de telles communications de s'effectuer. Il est possible que cela soit du à notre politique de remise à zéro des leurres.
Exfiltration	✗	✗	Nous n'avons pas simulé la présence de données dans notre réseau de leurres, et de ce fait, nous ne nous attendions pas à détecter de tentatives d'exfiltration. Nous n'en détectons pas, comme prévu.
Impact	✓	✓	Nous nous attendions à constater des attaques sur nos leurres, en particulier des tentatives de changement de mots de passe ou de comptes, ainsi que des défacements potentiels, en particulier par des rançongiciels. C'est d'ailleurs pour cela que nous avions mis en place un mécanisme de remise à zéro des leurres. Malgré cela, nous ne constatons aucune tentative d'impact autre que le vol de ressources matérielles, qui n'est pas effectif mais que les attaquants ont tenté de mettre en place.

Table 5.19: Détail des tactiques employées par les attaquants

Addresse	Attribution	Taux de suspicion
124.52.XX.XX	LG POWERCOMM Korea	10.43
61.177.XX.XX	Chinanet	13.79 %
218.88.XX.XX	Chinanet	0 %
61.177.XX.XX	Chinanet	11.49 %
61.177.XX.XX	Chinanet	9.19 %
218.92.XX.XX	Chinanet	9.19 %
61.177.XX.XX	Chinanet	10.34 %
185.246.XX.XX	w1n ltd	12.64 %
182.138.XX.XX	Chinanet	0 %
49.88.XX.XX	Chinanet	10.34 %

Table 5.20: Analyse Virustotal des 10 adresses IP les plus actives. On calcule pour chacune le pourcentage de vendeurs les détectant comme suspicieuses ou malveillantes.

Comportement des attaquants

En analysant les comportements des attaquants, en particulier concernant le bruteforce, nous constatons des comportements remarquables. Certains attaquants ne font que des tentatives avec des identifiants valides. Ceci implique soit une coïncidence très improbable, soit un transfert d'information, soit l'emploi de plusieurs adresses IP pour les attaques. Certaines adresses seraient donc dédiées au bruteforce, et d'autres uniquement à l'exploitation une fois les identifiants obtenus, par eux-mêmes ou auprès d'autres acteurs. En particulier, un attaquant ne fait qu'une tentative de connexion, entre une séquence de commandes et n'interagit plus avec cette adresse IP.

D'autres attaquants ont des comportements contre-intuitifs. L'un d'entre eux par exemple entre les identifiants valides dans sa première tentative, puis entre 30 identifiants invalides le lendemain. De même, certains utilisateurs continuent leur bruteforce après avoir trouvé des identifiants valides. On peut supposer alors que soit les attaquants tentent de collecter plusieurs jeux d'identifiants, soit l'automatisation de l'attaque est peu perfectionnée.

Plusieurs adresses IP différentes, toutes liées au malware Dota3 dans notre leurre, présentent exactement le même comportement. Ils effectuent d'abords une attaque bruteforce classique, en énumérant des identifiants jusqu'à trouver une paire valable. Après cela, ils tentent de se connecter avec les identifiants *345gs5662d34:345gs5662d34* et *root:3245gs5662d34*, ce qui se solde par des échecs. La chaîne de caractères *345gs5662d34* est connue de la communauté. On la retrouve dans les résultats de plusieurs déploiements de leurres mais il n'y a pas de consensus sur l'origine de ce mot de passe. Il ne semble correspondre à aucune valeur par défaut, et une hypothèse serait que cette chaîne correspond à un mot de passe usuel dans une langue étrangère écrit avec un clavier en alphabet latin. Après cela, ils effectuent une nouvelle tentative avec les identifiants valides qu'ils ont déjà obtenu.

Nous pensions au début que ce comportement s'expliquerait par l'utilisation d'un même dictionnaire pour les attaques en bruteforce. Néanmoins, l'analyse des tentatives montre que ce n'est pas le cas. Les deux couples d'identifiants qu'ils utilisent sont connus par la communauté, ayant déjà été détectés dans d'autres systèmes de déception. Nous pouvons supposer que ces identifiants sont utilisés par certains attaquants après la compromission pour créer un compte servant de "porte dérobée", leur permettant de sécuriser les systèmes qu'ils compromettent et d'y retourner.

Du fait de sa réinitialisation périodique, nous nous attendions à ce qu'un attaquant observant notre leurre pendant une période de temps prolongée puisse identifier sa nature. Nous nous attendions aussi à ce que des attaquants, voyant l'effet de leur compromission effacé par notre réinitialisation, ne se reconnectent pas à notre leurre, sa nature apparaissant clairement. Malgré cela, nous constatons que certains attaquants s'obstinent, répétant leurs opérations jusqu'à 56 fois pour l'un

d'entre eux, malgré les réinitialisations régulières.

Contrairement à ce que nous pouvions attendre de la part des attaquants opportunistes que nous attirons, nous ne constatons aucun comportement destructif. Aucun des attaquants qui parvient à accéder à notre leurre ne tente de le défacer, d'y supprimer des données ou d'installer de malware de type ransomware.

5.2.4 Discussion de la deuxième expérience

Comme l'analyse de nos données le montre, les attaquants ne se sont intéressés qu'à notre leurre d'entrée. Ils n'ont pas cherché à effectuer de mouvement latéral au sein de notre architecture. Ceci semble indiquer que nous n'avons réussi à attirer que des attaquants relativement peu motivés et déterminés, plutôt motivés par l'appât du gain opportuniste. Leur seule activité remarquable est le minage de cryptomonnaies et l'ajout de notre leurre à des réseaux de botnets, et ils ne montrent aucun intérêt pour d'autres attaques, du vol de données à l'extorsion. Nous attendions un intérêt plus poussé de la part des attaquants, du fait de notre IP identifiable comme appartenant à une entité académique. En effet, en 2022 et 2023, plusieurs établissements d'enseignement supérieur Français ont été victimes d'attaques, et nous espérions profiter de cette tendance pour attirer des attaquants motivés.

Ce travail se voulait être une amélioration du travail de notre première expérience concernant la mise en pratique de notre modèle de leurres. Notre implémentation est plus robuste, mieux supervisée, plus de données sont collectées et l'analyse est donc plus pertinente. De plus, le contexte de déploiement de notre réseau de leurres est plus intéressant que celui de notre première expérience.

Notre réseau de leurres a attiré des attaquants dans des proportions similaires à celles de notre première expérience et d'autres déploiements dans la littérature. Ceci semble confirmer l'attractivité des leurres construits selon notre modèle.

En plus de cela, l'analyse des attaques par bruteforce nous permet de comparer nos résultats avec ceux de la littérature sur près de deux décennies. Du fait du succès des attaquants, nous sommes capables d'analyser leur méthode d'exploitation après la phase d'accès initial, identifiant plusieurs classes de malware, du cryptominer à l'agent de botnet.

5.2.5 Limites

Bien que ces résultats soient satisfaisants, plusieurs limites persistent, et nous les identifions ici pour présenter des pistes de travaux futurs.

En premier lieu, nous n'avons attiré que des attaquants opportunistes et peu motivés. Nous ne trouvons aucun indice d'intérêt pour autre chose que nos ressources matérielles ou nos identifiants dans les actions des attaquants. Nos

mesures de réinitialisation rendent impossible pour eux d'effectuer des attaques de longue durée. Malgré cela, nous ne trouvons aucun indice indiquant que les attaquants tentent ce genre d'attaques. Notre routine de réinitialisation est une limite évidente de notre expérience, et il est nécessaire d'explorer d'autres pistes de robustesse pour d'autres déploiements à venir.

Une autre limite potentielle est le choix de notre adresse IP. Là où nous considérions notre attribution à un établissement scolaire comme un avantage pour l'attractivité, il est aussi possible que ce choix ait repoussé des attaquants experts. Ceux-ci ont pu se douter de la nature déceptive de notre système au vu de sa localisation.

Un approche permettant d'améliorer la collecte de données avec notre expérience serait de la déployer en parallèle dans plusieurs localisations. Du fait de la nature virtualisée de notre système, son partage et son déploiement est facile, et cela permettrait de maximiser la surface d'attaque présentée aux attaquants.

5.3 Résultats complémentaires de la deuxième expérience

5.3.1 Généralités

Dans le but de cadrer notre travail, nous avons limité dans le temps l'exposition de notre leurre. Nous avons analysé les résultats de notre seconde expérience après deux mois d'exposition. Néanmoins, nous n'avons pas démantelé notre expérience après cette période de temps. Nous avons laissé tourner notre expérience, et nous sommes capables d'analyser les résultats complémentaires deux mois après, entre le 15 mai et le 19 juillet. Ceci nous permet d'avoir une perspective de plus long terme sur les actions des attaquants. Au 19 juillet, nous avons collecté 161725 événements pour 308593 sessions SSH et 440752 tentatives d'authentification. Nous constatons un quasi doublement des données collectées depuis notre dernière analyse. Nous pouvons affirmer que sur la période entière, le nombre d'attaques croît de façon relativement linéaire avec le temps.

Nous ne présenterons pas à nouveau les détails du traitement préliminaire. Celui-ci est le même que pour l'analyse précédente, à cela près que le volume de données représente 2.3Go.

Nous reprenons la même structure d'analyse que pour la session précédente.

5.3.2 Analyse Quantitative

IP et sessions

Entre mai et juillet, nous passons à 1.6M d'événements pour 310k sessions et 440k tentatives d'authentification. Le nombre de tentatives est quasiment doublé, bien

IP Address	Login Attempts and percentages	Attribution
124.52.XX.XX	27518 - 6.24 %	LG POWERCOMM Korea
61.177.XX.XX	17947 - 4.07 %	Chinanet
218.88.XX.XX	17758 - 4.03 %	Chinanet
218.92.XX.XX	15292 - 3.47 %	Chinanet
61.177.XX.XX	14777 - 3.35 %	Chinanet
61.177.XX.XX	13565 - 3.08 %	Chinanet
218.92.XX.XX	11636 - 2.64 %	Chinanet
185.246.XX.XX	11045 - 2.51 %	w1n ltd Sweden
61.177.XX.XX	9096 - 2.06 %	Chinanet
81.68.XX.XX	8063 - 1.83 %	Tencent CSCL

Table 5.21: Adresse et nombre de tentatives des 10 acteurs les plus actifs ayant ciblé notre expérience.

que les autres valeurs croissent moins rapidement. De 206 connexions avec succès, nous passons à 736, avec un doublement du taux de succès qui passe à 0.17%. Ceci peut signifier que les attaquants sont plus efficaces, ou que des attaquants possédant déjà l'information se connectent plus fréquemment.

Nous détectons aussi 2201 nouvelles adresses IP, ce qui porte notre total à 5211 adresses depuis lesquelles ont lieu des connexions à notre leurre. En moyenne, les adresses IP effectuent 10 connexions de plus que lors de l'analyse précédente, mais aucune nouvelle adresse n'est plus active que celle qui l'était en mai. Nous constatons toujours que 75% des attaquants montrent un intérêt très superficiel pour notre leurre, tentant moins de 30 authentifications par adresse. Nous observons aussi la même tendance de 1577 attaquants à initialiser une session SSH sans tenter aucune authentification.

Nous ne sommes pas capables d'identifier de nouveau profil de comportement d'attaquant à partir des nouvelles données. Nous pouvons néanmoins actualiser notre tableau regroupant les 10 adresses IP les plus actives dans la table 5.21.

Parmi ces adresses, nous retrouvons 8 de celles qui dominaient l'analyse précédente. On peut constater que les deux nouvelles adresses de ce top 10, 218.92.XX.XX et 81.61.XX.XX sont totalement absentes de nos données collectées en mai. Parmi les adresses que nous avions déjà collecté, seule 185.246.XX.XX effectue de nouvelles tentatives d'authentification, passant de 4622 à 11045.

Analyse temporelle

Comme pour notre analyse précédente, nous sommes capables de représenter le nombre d'attaques sur notre leurre SSH par jour. Ceci nous permet de visualiser les tendances et la répartition des attaques. Nous retrouvons cette répartition dans la figure 5.6. Cette répartition est plus lissée que sur la première période, et sans interruptions.

Noms d'utilisateur

Bien que le nombre de tentatives de connexion ait doublé, le nombre de noms d'utilisateurs différents employés par les attaquants n'a augmenté que

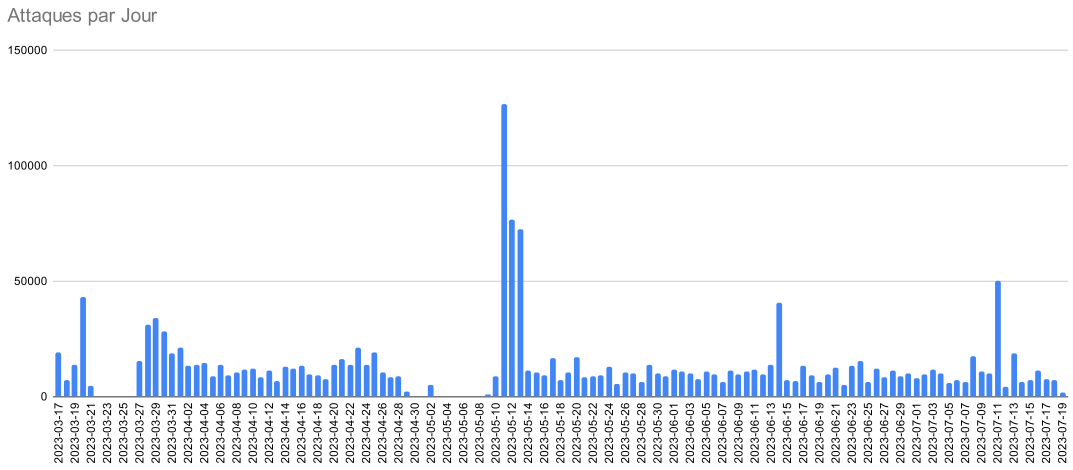

Figure 5.6: Analyse temporelle du nombre d'attaques par jour. On remarque pour la seconde partie de notre expérience, entre mai et juillet, une distribution très lisse des attaques, avec deux pics d'une journée.

Username	Nombre de tentatives	Pourcentage des tentatives	Evolution dans l'ordre
root	321040	72.8%	-
admin	22105	5%	-
user	13928	3.2%	-
user2	7537	1.7%	-
ubuntu	4217	1%	-
test	3135	0.7 %	-
oracle	3017	0.7%	-
pi	2441	0.5 %	-
ubnt	2044	0.5%	+1
debian	1922	0.4%	+3
postgres	1864	0.4%	-2

Table 5.22: Détail des 11 noms d'utilisateurs les plus utilisés par les attaquants, avec des résultats presque similaires à ceux de notre analyse précédente

marginalement. Dans notre troisième analyse, nous passons de 5330 à 6015 nouveaux identifiants. Nous observons que l'usage des noms d'utilisateur reste presque exactement le même, comme l'indique la table 5.22. On remarque que la répartition et l'usage des noms d'utilisateur les plus courants est resté extrêmement stable, au pourcent près. Ceci peut être expliqué par le fait que ces noms effectivement sont les plus utilisés par les attaquants. Une autre explication peut provenir du fait que les attaquants de la deuxième phase de notre expérience sont les mêmes acteurs, utilisant les mêmes dictionnaires d'attaque depuis des adresses IP différentes.

Du fait de la grande similarité entre nos résultats, nous ne présentons pas à nouveau la comparaison avec de précédentes expériences. Néanmoins, les analyses et les conclusions de la section précédente s'appliquent encore ici.

Mots de passe	Tentatives	Pourcentage
123456	9954	2.2%
(No Password)	8314	1.9%
1234	4654	1.00%
admin	3559	0.8%
1	3540	0.8%
123	3403	0.8%
password	3368	0.8%
root	2902	0.7%
12345	2743	0.6%
test	2307	0.5%

Table 5.23: Détails des 10 mots de passe les plus employés par les attaquants.

Mots de passe

Pour les mots de passe, les différences sont encore une fois très faibles entre nos deux analyses en mai et juillet. Nous passons de 46069 mots de passe distincts à 48686, soit une différence négligeable. Assez logiquement, au vu du nombre de tentatives, la moyenne du nombre de tentatives a environ doublé, passant de 4.9 à 9.1. Les valeurs de médiane et de quartiles n'évoluent pas. Nous constatons dans cette deuxième période de temps la même répartition très aplanie des mots de passe employés par les attaquants. On peut mettre à jour notre table des usages comme pour les noms d'utilisateur, dans la table 5.23.

De même que pour les noms d'utilisateur, les valeurs sont relativement identiques, le nombre de tentatives ayant doublé de façon uniforme. On retrouve dans un ordre différent mais dans des proportions similaires les mots de passe déjà identifiés lors de notre précédente analyse. L'analyse de la complexité des mots de passe employés montre des résultats très similaires, la moyenne du nombre de caractères variant d'un centième uniquement.

5.3.3 Analyse Qualitative

La section précédente consistait en une analyse quantitative des différents champs d'information collectés. Dans cette section, nous analysons en détail et qualitativement les informations que nous avons collectées avec notre leurre.

Sessions Valides

Lors de cette seconde analyse, nous sommes capables d'identifier 736 authentification valides contre 206 fin mai. Le nombre de tentatives a plus ou moins doublé tandis que le nombre de succès à plus que triplé. Les attaquants sont donc devenus plus efficaces à l'heure de choisir des identifiants valides, ou bien ont utilisé des identifiants valides qu'ils connaissaient déjà. Un nouveau couple user:password qui n'avait pas encore été identifié a permis à un attaquant d'accéder au leurre. Les autres ont utilisé les mêmes couples isolés dans notre analyse précédente. Nous listons les informations sur les authentifications valides dans la table 5.24. Nous remarquons l'usage du nom d'utilisateur "test", qui avait été jusqu'à présent infructueux pour les attaquants. Les connexions réussies

Identifiants Valides	Nombre de Connexions
root:Password123	574
root:azertyuiop	70
admin:Password	54
admin:Password123	30
root:bonjour	7
test:Password123	1

Table 5.24: Identifiants valides utilisés pour les authentifications des attaquants.

Sessions	Procédures	Tactiques MITRE
56	Téléchargement et Exécution du malware XORDDoS	Execution, Command and Control
11	Téléchargement d'un binaire obfusqué lié aux binaires Mirai/RapperBot	Command and Control
3+1	Procédure de téléchargement et exécution du malware DOTA3 + 1 interrompue	Execution, Impact
9	cat /bin/echo;	Reconnaissance
4	Acquisition d'information sur le hardware (CPU et RAM)	Reconnaissance
10	Version obfusquée du malware Shellbot avec des commentaires en portugais	Execution, Command and Control
1	Acquisition d'information sur le hardware et les connexions ouvertes	Reconnaissance

Table 5.25: Description des différentes procédures effectuées par les attaquants.

proviennent de 329 adresses IP différentes au lieu de 99, nombre cohérent avec le triplement des connexions réussies.

Là où les plages d'adresses 61.177.173.XX et 61.177.172.XX représentaient plus de la moitié des connexions réussies en mai, on ne constate aucune nouvelle connexion depuis celles-ci. Elles ne sont même plus les plages les plus actives, avec la plage 141.98.11.XX représentant 84 connexions valides.

Commandes et Procédés

Comme pour la première analyse, la grande majorité des authentifications réussies ne mènent à aucune saisie de commandes de la part des attaquants. Nous pouvons à nouveau supposer que ceci peut être dû à deux facteurs. Soit ils ont deviné la nature déceptive de notre leurre, soit ils ne sont intéressés que par la collecte d'identifiants et non l'exploitation. Sur nos 736 sessions, 96 incluent des commandes. Comme pour notre première analyse, ces commandes ne sont pas uniques, et nous pouvons isoler 8 procédures distinctes, soit 2 de plus que lors de notre première analyse. Nous listons les procédures dans la table 5.25.

Comme pour notre première analyse, nous constatons que les attaquants se concentrent sur des opérations rapides et automatiques. Nous ne détectons toujours pas de tentative de déplacement latéral au sein de notre réseau et pouvons confirmer le caractère opportuniste des attaquants.

Un nouvel attaquant a un comportement remarquable. Il collecte de l'information sur le CPU et sur la RAM, comme d'autres, mais il vérifie aussi le contenu du fichier "/proc/net/tcp" en y cherchant des indices de connexion ouverte avec une adresse IP spécifique, adresse non reconnue comme étant malveillante au moment de notre analyse.

5.3.4 Suite de la deuxième expérience

Le fait de continuer à collecter des données de notre deuxième expérience après une première analyse nous permet de projeter nos résultats sur un semestre entier. Sur cette durée étendue, nous constatons une continuité dans notre analyse. Le comportement des attaquants est stable sur cette période de temps.

Comme pour notre première analyse, nous détectons presque uniquement des attaquants opportunistes, dont nous sommes facilement capables d'identifier les procédures et les intentions. Même sur une période de temps étendue, nous ne constatons toujours pas de déplacement latéral ni de tentatives d'attaques plus complexes que l'exploitation immédiate des ressources matérielles.

Globalement, cette nouvelle analyse vient renforcer les conclusions que nous tirons dans la section précédente.

Chapter 6

Discussion des Résultats

6.1 Modèle et implémentation

La conception et le déploiement de nos deux expériences cherchaient à remplir plusieurs objectifs. Premièrement, nous cherchions à tester notre modèle de leurres, et à déterminer s'il était possible de construire un réseau réel de leurres à partir de celui-ci.

En plus de la validation de notre modèle, nous travaillons aussi à partir d'un état de l'art. Des expériences relativement similaires à celles que nous déployons sont trouvables dans la littérature, entre la fin des années 2000 et la fin des années 2010. Néanmoins, nos expériences se veulent une actualisation des expériences passées.

Nous avons pu intégrer de nouveaux composants dans l'implémentation et la conception. Par exemple, les expériences passées se basaient techniquement sur des équipements réels ou des machines virtuelles complètes. De notre côté, nous avons fait le choix technique de l'usage de conteneurs pour la virtualisation. De même, nous sommes capables d'intégrer à notre réflexion, lors de la conception, la taxonomie MITRE ATT&CK. En faisant cela, nous disposons d'un outil de modélisation et de représentation des attaques qui faisait défaut à beaucoup de travaux précédents. Ceux-ci pouvaient se baser sur des concepts plus simples tels que la CyberKillChain de Lockheed Martin, mais grâce à notre travail, une approche plus approfondie des comportements des attaquants peut être intégrée à la conception de leurres. En particulier, notre approche intégrant la matrice MITRE ATT&CK nous permet de concevoir nos leurres en prenant en compte le point de vue de l'attaquant, rendant la déception plus efficace.

Dans notre travail de modélisation et d'implémentation, nous avions la volonté d'élargir le périmètre des leurres pour permettre de collecter des données plus importantes, et de meilleure qualité. Du fait que nous étions intéressés par des attaquants motivés au comportement expert, nous avons cherché à mettre en place

plusieurs niveaux parallèles de CyberKillChain dans nos leurres. Dans nos deux expériences, nous implémentons deux domaines de cyberkillchains parallèles. La première est la cyberkillchain générale de notre réseau de leurre, et la seconde la cyberkillchain propre à chaque élément de notre réseau de leurres.

Ainsi, un attaquant se devait, s'il voulait progresser dans notre leurre et en compromettre le plus d'éléments possibles, de parcourir les éléments de CyberKillChain que nous y avions intégrés. Il commençait par accéder à nos leurres en niveau d'entrée, puis il pouvait gagner un accès lui permettant d'accomplir un déplacement latéral vers nos autres leurres, et ainsi de compromettre d'autres fonctionnalités de notre infrastructure simulée. Ainsi, il devait parcourir la cyberkillchain de l'ensemble de notre réseau de leurres.

Néanmoins, pour parvenir à compromettre individuellement chacun des composants du réseau de leurres, et ainsi pouvoir y progresser, les attaquants se devaient de collecter des informations et accéder à ces composants individuels, avec un niveau de privilège suffisant. Il devait donc parcourir un ensemble de cyberkillchains propre à chacun des leurres qui composent notre réseau de leurres au complet.

Par ce double parcours de chaîne d'attaque, nous cherchions à rapprocher au mieux notre implémentation d'une architecture réelle afin d'exposer aux attaquants un environnement de déception le plus efficace possible.

En actualisant les méthodes de conception et d'implémentation, nous nous attendions à pouvoir détecter des attaques absentes dans les travaux précédents. Ces éléments, couplés à notre choix de localisation pour les leurres, nous ont fait espérer que nous pourrions attirer des attaquants motivés et compétents. Nous espérions pouvoir, grâce à nos analyses, actualiser les résultats sur le profil et les approches privilégiées par les attaquants.

6.2 Synthèse des Analyses

Nos résultats montrent que notre modèle et notre implémentation modernisés ont su attirer des attaquants dans des proportions similaires à celles de la littérature. De ce fait, on peut affirmer que, quantitativement, notre expérience attire les attaquants et permet de collecter des informations de façon similaire à celles de la littérature. Concernant l'aspect qualitatif des attaques collectées et analysées, nous constatons une grande similarité entre nos résultats et ceux de la littérature. C'est donc le constat suivant que nous pouvons faire. Nous avons cherché et réussi à actualiser les méthodes de déception pour la collecte de données, mais nos analyses montrent que les attaquants ont peu changé dans leurs intérêts et leurs pratiques.

Concernant notre idée d'exposition de *double CyberKillChain*, nous constatons un relatif échec dans nos deux expériences. Dans aucun des cas d'attaques que

nous détectons les attaquants ne cherchent à progresser au sein de notre réseau de leurres. Malgré les mesures que nous avons prises lors de notre seconde expérience pour améliorer notre implémentation, les attaquants se contentent de compromettre nos leurres d'entrée, lorsqu'ils arrivent à y accéder. Ainsi, de nos deux CyberKillChains exposées en parallèle, seules celles relatives aux leurres individuels, et en particulier aux leurres en niveau d'entrée, sont exploitées par les attaquants.

Notre objectif est donc paradoxalement rempli. Nous sommes capables d'affirmer que nos leurres attirent les attaquants avec une efficacité similaire à ceux présentés dans la littérature. De plus, les données que nous collectons sont de qualité, et permettent l'identification des procédés et des outils utilisés par les attaquants. Néanmoins, nous ne sommes pas capables d'identifier des attaquants avec des profils innovants ou différents de ceux que les leurres précédents parvenaient à attirer. En particulier, nous n'attirons pas d'attaquants motivés ou expérimentés, tels que des APT, au sein de notre leurre. De plus, parmi les différents comportements que nous espérions pouvoir analyser, nous ne sommes pas capables de collecter des données témoignant de tentatives de déplacement latéral.

Ce constat nous permet de défendre le fait que les attaques techniques ciblant les infrastructures exposées sur internet ont assez peu varié dans les dernières années. Les attaquants restent essentiellement des opportunistes dont l'objectif principal est de tirer un profit rapide de leur exploitation. L'évolution des pratiques se remarque dans la nature des exploitations. Nous constatons par exemple qu'en plus de la constitution de botnets, le vol de ressources passe beaucoup par l'installation de mineurs de cryptomonnaies, qui n'étaient pas répandus il y a une dizaine d'années.

Notre système de leurres est innovant et efficace, mais l'écosystème des attaquants que nous cherchons à attirer ne semble pas avoir beaucoup varié, adoptant des techniques nouvelles pour remplir des objectifs similaires.

6.3 Travaux Futurs

Concernant notre travail, nous envisageons deux grands axes de continuation qui nous permettraient de pousser plus avant nos résultats et nos recherches.

De façon immédiate, une première piste de travail concerne l'extension du déploiement de notre système de leurres, potentiellement avec une politique de remise à zéro des composants moins stricte, afin de tenter d'attirer et d'identifier des attaquants plus motivés. En effet, nous pensons que notre système est en capacité de détecter des attaquants plus intéressants que ceux opportunistes que nous détectons. L'extension du déploiement peut se faire dans le temps, mais aussi avec le clonage de notre machine virtuelle sur d'autres infrastructures, afin de mettre en place un réseau distribué de leurrage réparti sur plusieurs

sites.

Un autre axe de travail pertinent pourrait porter sur la complexité des systèmes de leurre. Pendant notre étude de l'état de l'art et notre conception de systèmes de leurre, nous sommes confrontés de façon assez intuitive au concept de complexité des leurres. Cette notion recouvre un ensemble d'éléments, regroupant à la fois le niveau d'interaction possible avec les attaquants, mais aussi le nombre de services simulés, la finesse de la simulation, etc. Cette complexité, difficilement quantifiable, peut sembler à première vue corrélée avec le périmètre d'exposition du leurre. Un leurre plus complexe, au sens où il permet plus d'interactions par des attaquants sur plus de services ou avec plusieurs niveaux d'architecture présente un périmètre de leurrage plus élargi qu'un leurre plus simple.

Au-delà de ces considérations instinctives, un travail pourrait être mené sur la possibilité de quantifier, par la prise en compte des différents facteurs qui l'influencent, la complexité d'un leurre. Cette quantification pourrait permettre de comparer différentes conceptions et implémentations de leurres avec un approche standardisée.

Cette notion de complexité peut aussi de prime abord sembler liée à la capacité d'un leurre à sembler réaliste, et à attirer des attaquants dont la suspicion aurait pu être éveillée par des leurres plus simples. De même, on peut assez immédiatement concevoir que la complexité d'un système de leurres a un impact sur sa conception et son implémentation, soulevant des contraintes et des surcoûts qu'il convient de prendre en compte.

Un travail complémentaire pourrait donc porter sur la définition précise de ce concept intuitif de complexité d'un leurre, ainsi que sur une tentative de quantification standardisée de celle-ci. Si les objectifs de ces travaux étaient atteints, alors il serait possible de relier expérimentalement la complexité et l'efficacité des leurres, en fonction de comment ces concepts seraient définis. On pourrait alors pour chaque contexte de déploiement borner la complexité des systèmes de leurres, les rendant suffisamment complexes pour répondre aux impératifs de déception, mais assez peu pour ne pas être dépassés par les contraintes imposées.

Bibliography

[1] Honeypot system for attacks on ssh protocol. *International Journal of Computer Network and Information Security*. doi: 10.5815/ijcnis.2016.09.03.

[2] Cengage learning.

[3] Cowrie, 2022. URL `https://www.cowrie.org/`.

[4] Matrix - Enterprise | MITRE ATT&CK®, 2022. URL `https://attack.mitre.org/versions/v11/matrices/enterprise/`.

[5] Kernel virtual machine, 2023. URL `https://linux-kvm.org/page/Main_Page`.

[6] Virustotal, 2023. URL `https://www.virustotal.com/gui/home/upload`.

[7] "0x00pf". Iot malware droppers (mirai and hajime), 2017. URL `https://0x00sec.org/t/iot-malware-droppers-mirai-and-hajime/1966`.

[8] Mokube. Adams. Honeypots: concepts, approaches, and challenges. In *Proceedings of the 45th annual southeast regional conference*, 2007. doi: 10.1145/1233341.1233399.

[9] Maxime Alay-Eddine. Richelieu, 2023. URL `https://github.com/tarraschk/richelieu`.

[10] ANSSI, 2021. URL `https://www.ssi.gouv.fr/guide/recommandations-relatives-a-lauthentification-multifacteur-et-aux-mots-de-passe/`.

[11] Strom. Applebaum. Mitre att&ck: Design and philosophy, 2020. URL `https://attack.mitre.org/docs/ATTACK_Design_and_Philosophy_March_2020.pdf`.

[12] Barbier. *D-Day deception: Operation fortitude and the Normandy invasion*. Stackpole Books, 2005.

[13] Canonical. Linux containers, 2023. URL `https://ubuntu.com/lxd`.

[14] CERT-FR, 2021. URL `https://www.cert.ssi.gouv.fr/avis/CERTFR-2021-AVI-759/`.

[15] Cheswick. An evening with berferd in which a cracker is lured, endured, and studied. In *Proc. Winter USENIX Conference, San Francisco*, pages 20–24, 1992.

[16] Cohen. The deception toolkit homepage, 1998. URL `http://all.net/dtk/`.

[17] Commvault. Metallic.io threat wise, 2023. URL `https://metallic.io/threatwise-cyber-deception`.

[18] CounterCraft. Countercraft, 2023. URL `https://www.countercraftsec.com/`.

[19] France Armée Centre de doctrine d'emploi des forces Division doctrine. *Glossaire français / anglais de l'armée de terre (ex TTA 106) : recueil de termes, sigles, signes et symboles conventionnels militaires*. Etablissement de diffusion, d'impression et d'archive du Commissariat des Armées, 2013.

[20] Dacier. Pouget. Debar. Attack processes found on the internet. 2004.

[21] Dimensions. Dimensions.ai, 2023. URL `https://app.dimensions.ai/`.

[22] DinoTools. Dionaea, 2021. URL `https://github.com/DinoTools/dionaea`.

[23] Docker. Docker, 2023. URL `https://www.docker.com/`.

[24] Eder. Hypervisor-vs. container-based virtualization. volume 1, 2016.

[25] Elastic. Elasticsearch. 2023. URL `https://www.elastic.co`.

[26] Elasticsearch-dump. Elasticsearch-dump, 2023. URL `https://github.com/elasticsearch-dump/elasticsearch-dump`.

[27] Agnaou. Abou. et al. Automated technique to reduce positive and negative false from attacks collected through the deployment of distributed honeypot network. *International Journal of Computer Science and Information Security*, 2016.

[28] Alata. Nicomette. et al. Lessons learned from the deployment of a high-interaction honeypot. In *2006 Sixth European Dependable Computing Conference*, 2006. doi: 10.1109/EDCC.2006.17.

[29] Alshamrani. Myneni. et al. A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. *IEEE Communications Surveys & Tutorials*, 2019. doi: 10.1109/COMST.2019.2891891.

[30] Ben Said. Biondi. et al. Detection of Mirai by Syntactic and Semantic Analysis. 2017. URL `https://inria.hal.science/hal-01629040`.

[31] Diamantoulakis. Dalamagkas. et al. Game theoretic honeypot deployment in smart grid. *Sensors*, 2020. doi: 10.3390/s20154199.

[32] Giallorenzo. Mauro. et al. Virtualization costs: Benchmarking containers and virtual machines against bare-metal. *SN Computer Science*, 2021. doi: 10.1007/s42979-021-00781-8.

[33] Hilt. Maggi. et al. Caught in the act: Running a realistic factory honeypot to capture real threats. *Trend Micro, Shibuya City, Japan, White Paper*, 2020.

[34] Hutchins. Cloppert. et al. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 2011.

[35] Koniaris. Ioannis. et al. Analysis and visualization of ssh attacks using honeypots. In *Eurocon 2013*, 2013. doi: 10.1109/EUROCON.2013.6624967.

[36] Lallie. Shepherd. et al. Cyber security in the age of covid-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic. *Computers & Security*, 105, 2021. doi: 10.1016/j.cose.2021.102248.

[37] Leita. Pham. et al. The Leurre.com Project: Collecting Internet Threats Information Using a Worldwide Distributed Honeynet. In *2008 WOMBAT Workshop on Information Security Threats Data Collection and Sharing*, Amsterdam, 2008. IEEE. doi: 10.1109/WISTDCS.2008.8.

[38] Mairh. Abhishek. et al. Honeypot in network security: A survey. In *Proceedings of the 2011 international conference on communication, computing & security*, 2011. doi: 10.1145/1947940.1948065.

[39] Nawrocki. Marcin. et al. A Survey on Honeypot Software and Data Analysis, 2016. URL `http://arxiv.org/abs/1608.06249`.

[40] Nicomette. Kaâniche. et al. Set-up and deployment of a high-interaction honeypot: experiment and lessons learned. *Journal in Computer Virology*, 2011. doi: 10.1007/s11416-010-0144-2. URL `http://link.springer.com/10.1007/s11416-010-0144-2`.

[41] Ryandy. Lim. et al. Xt-pot: Exposing threat category of honeypot-based attacks. In *Proceedings of the 2021 International Conference on Engineering and Information Technology for Sustainable Industry*, 2020. doi: 10.1145/3429789.3429868.

[42] Scottberg. Yurcik. et al. Internet honeypots: protection or entrapment? In *IEEE 2002 International Symposium on Technology and Society (ISTAS'02). Social Implications of Information and Communication Technology. Proceedings (Cat. No.02CH37293)*, 2002. doi: 10.1109/ISTAS.2002.1013842.

[43] European Union Agency for Cybersecurity. Valeros. Malatras. et al. *ENISA threat landscape for supply chain attacks*. European Network and Information Security Agency, 2021. doi: doi/10.2824/168593.

[44] Girard. The honeypot stings back: Entrapment in the age of cybercrime and a proposed pathway forward. *Chicago Journal of International Law*, 2023.

[45] Google. Google trends, 2023. URL `https://trends.google.com/trends/explore?cat=314&date=all&q=honeypot&hl=fr`.

[46] Joint Task Force Interagency Working Group. *Security and Privacy Controls for Information Systems and Organizations*. Revision 5 edition, 2020. doi: 10.6028/NIST.SP.800-53r5. URL `https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf`.

[47] ISO/IEC. Norme iso 27001, 2013.

[48] linuxcontainers. Linux containers, 2023. URL `https://linuxcontainers.org/`.

[49] Ponemon Institute LLC. 2022 cost of insider threats global report. URL `https://www.proofpoint.com/us/resources/threat-reports/cost-of-insider-threats`.

[50] Chigada. Madzinga. Cyberattacks and threats during covid-19: A systematic literature review. *South African Journal of Information Management*, 23(1), 2021. ISSN 1560-683X. doi: 10.4102/sajim.v23i1.1277.

[51] McCaughey. Deception using an ssh honeypot. Technical report, Naval Postgraduate School Monterey United States, 2017.

[52] MITRE. Common attack pattern enumeration and classification, 2023. URL `https://capec.mitre.org`.

[53] MITRE. Common vulnerabilities and exposures, 2023. URL `https://cve.mitre.org/`.

[54] MITRE. Common weaknesses enumeration, 2023. URL `https://cwe.mitre.org/`.

[55] netfilter. Iptables, 2023. URL `https://www.netfilter.org/projects/iptables/index.html`.

[56] NIST, 2021. URL `https://nvd.nist.gov/vuln/detail/CVE-2021-44228#range-9066479`.

[57] NoAH. European network of affined honeypots, 2015. URL `https://web.archive.org/web/20150201032813/http://www.fp6-noah.org/`.

[58] Oracle. Virtualbox, 2023. URL `https://www.virtualbox.org/`.

[59] Apache HTTP Server Project, 2023. URL `https://httpd.apache.org/`.

[60] The Honeynet Project. The honeynet project, 1999. URL `https://www.honeynet.org/`.

[61] Provos. A virtual honeypot framework. 2004.

[62] PurpleSec. Purplesec 2022 Cyber Security Statistics Trends & Data. URL `https://purplesec.us/resources/cyber-security-statistics/`.

[63] qeeqbox, 2023. URL `https://github.com/qeeqbox/honeypots`.

[64] Ponemon Institute LLC. IBM Security. Cost of a Data Breach Report 2020. URL `https://www.ibm.com/downloads/cas/RZAX14GX`.

[65] Spitzner. *Honeypots: tracking hackers*. Addison-Wesley, Boston Munich, 1. print edition, 2003. ISBN 978-0-321-10895-1.

[66] Spitzner. Honeytokens: The other honeypots, 2005.

[67] Stoll. *The cuckoo's egg: tracking a spy through the maze of computer espionage*. Doubleday, 1st ed edition. ISBN 978-0-385-24946-1.

[68] Upi Tamminen. Kippo, 2014. URL `https://github.com/desaster/kippo`.

[69] Thinkst. Thinkst canary, 2023. URL `https://canary.tools/`.

[70] Wazuh. Wazuh, 2023. URL `https://wazuh.com/`.

ECOLE DOCTORALE

Titre : titre (en français)

Mots clés : Cybersécurité, Leurrage, Honeypot, Déception

Résumé : L'emploi de leurres et de techniques de déception pour la cybersécurité est très présent dans la littérature, même s'il reste relativement peu employé dans l'industrie malgré des progrès dans la virtualisation des systèmes et des architectures. Il est possible aujourd'hui de déployer des leurres pour détecter des attaquants et analyser leur procédés, mais se déploiement se fait au niveau individuel, avec un approche restreinte : un leurre simulant un ou plusieurs services est positionné au sein d'un périmètre à défendre. Cette approche au cas par cas rend difficilement généralisable le déploiement et l'analyse de données issues des leurres. Dans cette thèse, nous avons cherché à construire un modèle de leurre qui permet de décrire ceux-ci de façon claire et détaillée et à tester la faisabilité et l'efficacité des leurres bâtis selon celui-ci . Nous présentons en premier notre modèle ainsi que ses différentes composantes. Il se base en particulier sur la matrice MITRE ATT&CK qui nous permet une approche novatrice en construisant nos leurres à partir de possibilités d'attaque offertes aux attaquants, en simulant toute une cyberkillchain plutôt que de simples vulnérabilités. Nous avons ensuite cherché à vérifier la faisabilité de notre modèle en construisant un réseau de leurres en nous basant sur notre modèle, puis avons testé l'efficacité de ces leurres pour l'analyse de données d'attaque en les déployant dans deux contextes différents. Nous avons démontré que nos leurres sont efficaces à l'heure d'attirer des attaquants et d'obtenir des données d'analyse exploitables.

Title : titre (en anglais)

Keywords : Cybersécurity, Decoy, Honeypot, Deception

Abstract : The use of decoys and deception techniques in cybersecurity is well documented in the literature, although it is not widespread used in industry despite advances in system and architecture virtualization. It is possible today to deploy decoys to detect attackers and analyze their processes, but deployment is done on an individual level, with a restricted approach : a decoy simulating one or more services is positioned within a perimeter to be defended. This case-by-case approach makes it difficult to generalize the deployment and analysis of decoy data. In this thesis, we set out to build a decoy model that provides a clear and detailed description of decoys, and to test the feasibility and effectiveness of decoys based on this model. We first present our model and its various components. In particular, it is based on the MITRE ATT&CK matrix, which enables us to take an innovative approach by building our decoys from attack possibilities offered to attackers, simulating an entire cyberkillchain rather than just vulnerabilities. We then sought to verify the feasibility of our model by building a network of decoys based on our model, and tested the effectiveness of these decoys for analyzing attack data by deploying them in two different contexts. We demonstrated that our decoys are effective in attracting attackers and obtaining exploitable analysis data.

Institut Polytechnique de Paris
91120 Palaiseau, France