Pouvez-vous détecter le piège? Détection de pots de miel face à l'évolution des tactiques d'évasion

Can You Spot the Trap? Honeypot Detection in the Face of Evolving Evasion Tactics

Mathis Durand[1,*], Alexandre Dey[2], Yvon Kermarrec[3] and Marc-Oliver Pahl[1]

[1]IRISA, IMT Atlantique, Cyber CNI, Rennes, France
[2]Airbus CyberSecurity, Cyber CNI
[3]Lab-STICC, IMT Atlantique, Brest, France

Abstract

Honeypots are important tools in network security, offering a means to detect, deflect, and analyze malicious activity. However, their effectiveness relies on remaining undetected. High-skill attackers are aware of honeypots and develop detection and evasion tactics. This paper presents a systematization of knowledge of the current state of honeypot detection, providing an overview of techniques employed in both laboratory and real-world settings.

Keywords

Honeypot, Honeypot identification, Honeypot detection, Anti-honeypot techniques

Résumé

Les pots de miel sont des outils importants pour la sécurité des réseaux, car ils permettent de détecter, de détourner et d'analyser les activités malveillantes. Toutefois, leur efficacité repose sur le fait qu'ils ne soient pas détectés. Les attaquants les plus avancés connaissent le concept de pots de miel et mettent au point des tactiques de détection et d'évasion. Cet article présente une systématisation des connaissances sur l'état actuel de la détection des pots de miel, en donnant un aperçu des techniques employées en laboratoire et dans le monde réel.

1. Introduction

Les pots de miel ont gagné en importance pour la sécurité des systèmes d'information en apportant de nouvelles connaissances sur les attaquants. Le nombre d'articles de recherche publiés chaque année avec les mot-clés *honeypot* et *security* est en constante augmentation depuis 2001 d'après Scopus. Les *pots de miel* sont des outils qui imitent des systèmes pour attirer des attaquants. Les pots de miel permettent ainsi de collecter des données ou de dévier des attaques tout en garantissant la sécurité du véritable système. Seulement, la force du pot de miel réside dans sa discrétion. Des attaquants précautionneux vont chercher à éviter ce type d'environnement afin d'attaquer un système véritablement sensible. Dans ce cas précis, un pot de miel identifié comme tel par un attaquant perd tout intérêt. Pour contrer une détection par les attaquants de pots de miel, il est crucial de comprendre comment détecter un pot de miel.

Cet article est la première systématisation des connaissances sur les méthodes de détection de pots de miel. Néanmoins, de nombreuses techniques sont déjà décrites dans la littérature.

Cette systématisation des connaissances permet de :

- proposer une taxonomie des pots de miel relative à leur détection pour identifier leurs vecteurs de détection;

C&ESAR'25: Computer & Electronics Security Application Rendezvous, Nov. 19-20, 2025, Rennes, France
*Corresponding author.
mathis.durand@imt-atlantique.fr (M. Durand); alexandre.dey@airbus.com (A. Dey); yvon.kermarrec@imt-atlantique.fr (Y. Kermarrec); marc-oliver.pahl@imt-atlantique.fr (M. Pahl)
0009-0002-3649-5490 (M. Durand); 0000-0002-7006-9113 (A. Dey); 0009-0005-9418-8976 (Y. Kermarrec); 0000-0001-5241-3809 (M. Pahl)

CEUR Workshop Proceedings ISSN 1613-0073
published 2025-12-22

Table 1
Travaux connexes

Références	Année	Taxonomie	Stratégie		Type		Origine	
			Évasion	Détection	Empreinte	Comportement	Laboratoire	Attaque
Chen et al. [1]	2008	◉	✓		✓	✓		✓
Lauren et al. [2]	2017	◉		✓	✓		✓	
Afianian et al. [3]	2019	◉	✓				✓	✓
Tay et al. [4]	2023	◉		✓	✓		✓	
Cet article	2025	●	✓	✓	✓	✓	✓	✓

- proposer une taxonomie de la détection de pots de miel pour structurer les différentes techniques de détection de pots de miel;
- analyser les différentes techniques de détection de pots de miel décrites dans la littérature;
- proposer une synthèse des techniques de détection de pots de miel ainsi que les mitigations suggérées.

Cette systématisation des connaissances est construite de la manière suivante. La section 2 présente les travaux connexes. La section 3 présente la méthode utilisée pour cette systématisation des connaissances. La section 4 présente une taxonomie des techniques de détection. La section 5 présente une analyse des techniques de détection. La section 6 présente une synthèse de la revue de la littérature et leurs mitigations.

2. Travaux connexes

Cette section présente les articles de la littérature ayant un but proche de cet article. Cette section développe les points de convergence et de divergence des différents travaux et souligne la nécessité d'une systématisation des connaissances.

La table 1 récapitule les travaux connexes. Les travaux connexes peuvent proposer une taxonomie (complète ● ou partielle ◉), décrire les stratégies (évasion ou détection), se restreindre sur au type de détection (par empreinte ou selon le comportement) et décrire des méthodes utilisées par des défenseurs ou des attaquants. Néanmoins les travaux identifiés n'offrent pas de systématisation des connaissances. [1, 3] proposent une analyse des techniques d'évasion découvertes dans des attaques ou en laboratoire. [2, 4] proposent une analyse des techniques de détection basées sur l'empreinte du pot de miel établies en laboratoire. Les paragraphes ci-dessous décrivent les principales contributions des travaux connexes.

Un *malware* peut avoir différents comportements selon le système où il est exécuté [1]. Chen et al. analysent 6,222 échantillons de malware et décrivent leur changement de comportement. Exposés à un environnement virtuel, autour de 4% des échantillons présente moins d'actions suspectes contre 39.9% exposés à un environnement avec des *debuggers*. Dans ces conditions, Chen et al. [1] développent une taxonomie des vecteurs de détection utilisables par un malware.

Lauren et al. [2] proposent une taxonomie sur les techniques de détections de pots de miel. Leur méthodologie distingue deux groupes de pots de miel: les pots de miel à basse et moyenne interactivité et les pots de miel à haute interactivité. Les pots de miel à basse et moyenne interactivité sont plutôt concernés par les techniques de prise d'empreinte du réseau et de l'application. Tandis que, les pots de miel à haute interactivité sont visés par les techniques de prise d'empreinte du système et de l'analyse des opérations.

Afianian et al. [3] proposent une taxonomie sur les techniques d'évasion utilisées par les *malware*. Les techniques d'évasion présentées concernent plutôt les environnements d'analyse que les pots de

miel. Cependant, ces techniques proviennent de logiciels malveillants et renseignent des méthodes utilisées par de réels attaquants.

Tay et al. [4] complètent la taxonomie de Lauren et al. en ajoutant plusieurs catégories. Leur article propose une revue des techniques de détection des pots de miel implémentés pour les *Smart Grids*. Ces techniques se concentrent sur la détection par l'empreinte du pot de miel et sur les techniques proposées en laboratoire. Ce papier propose une taxonomie des techniques de détection avec une catégorie et une sous-catégorie des techniques présentées.

Les travaux connexes ne regroupent pas les techniques de détection ou d'évasion de pots de miel utilisées par les défenseurs ou par les attaquants.

3. Méthodologie

Cet article apporte des réponses aux questions de recherche suivantes:

- Quelle est la taxonomie des techniques de détection des pots de miel?
- Quelles sont les techniques de détection utilisées par les défenseurs?
- Quelles sont les techniques de détection utilisées par les attaquants?
- Les techniques de détection utilisées par les défenseurs sont-elles comparables aux techniques des attaquants?

Proposer de nouvelles techniques de détection de pot de miel pose un problème éthique et moral. Plusieurs articles consacrent explicitement une section pour discuter des aspects éthiques [5, 6, 7]. D'autres articles proposent des contre-mesures en plus de la technique de détection [8]. C'est pourquoi, cet article propose une synthèse dédiée aux mitigations des techniques de détection pour le défenseurs.

Cette systématisation des connaissances présente la méthodologie suivante:

1. Une identification des blocs de construction des pots de miel focalisée sur leur détection:
   Une architecture de référence des pots de miel est proposée. Un point de vigilance est accordé aux propriétés d'un pot de miel responsables des mécanismes de détection par un utilisateur.
2. Une construction d'une taxonomie basée sur les travaux de la littérature:
   Cette taxonomie introduit les notions centrales dans la caractérisation d'une technique de détection de pot de miel. Elle propose le vocabulaire et les paramètres permettant de comparer des techniques de détection.
3. Une revue des techniques de détection de pots de miel issues de la littérature:
   Les mot-clés pour sélectionner les articles de la littérature sont *"honeypot detection"*, *"honeypot identification"*, *"honeypot fingerprinting"*, ou *"anti-honeypot"*. Les articles sélectionnés sont principalement en anglais. Néanmoins, un article en français a été intégré dans cette revue car ce dernier introduit deux techniques de détection utilisées par des attaques réelles [9]. Cette revue inclut certaines techniques d'évasion. Ces techniques d'évasion ne permettent pas de détecter un pot de miel mais offrent les mêmes conséquences: empêcher un logiciel malveillant de s'exécuter en dehors de leur cible et rendre un pot de miel inefficace.
4. Une synthèse des techniques existantes avec leurs mitigations.
   Cette synthèse permet d'illustrer les points traités par la littérature et les mitigations des techniques de détection.

4. Taxonomies

La littérature propose différentes taxonomies pour les pots de miel et pour les techniques de détection. Les différentes taxonomies rencontrées peuvent être complètes ou partielles, être présentées explicitement ou implicitement. La section section 4 présente de manière explicite une agglomération des taxonomies rencontrées dans la littérature.

Cette section définit deux taxonomies. La section 4.1 présente une taxonomie des pots de miel restreinte à la détection de pot de miel. Cette taxonomie décrit les propriétés d'un pot de miel, notées h1-6, qui ont un rôle dans sa détection. La section 4.2 présente une taxonomie des techniques de détection des pots de miel, notées c1-3 (propriété du contexte de la technique) ou t1-4 (propriété de la technique elle-même). Cette taxonomie permet de comparer différentes techniques de détection ou d'évasion et de les classifier.

Une taxonomie offre trois points forts:

1. exhaustivité: c'est-à-dire, toutes les techniques décrites dans la littérature peuvent être caractérisées par la taxonomie.
2. précision: c'est-à-dire, décrire à elle seule une technique de détection ou d'évasion de pots de miel.
3. concision: c'est-à-dire, comparer synthétiquement deux techniques entre elles.

Néanmoins, un champ de la taxonomie proposée reste mal défini. Le champ Cible (c1) est trop vaste pour être parfaitement exhaustif. Cependant ce champ offre un niveau de détail important pour décrire le cas d'usage (Cible (c1)). Ce champ reste donc une propriété-clé de cette taxonomie. La table 2 récapitule les catégories de la taxonomie des techniques de détection de pot de miel mentionnées ci-dessus.

4.1. Taxonomie des pots de miel restreint à leur détection

La taxonomie présentée dans les prochains paragraphes reposent sur trois articles de la littérature [10, 11, 12].

4.1.1. Objectif (h1)

Les pots de miel peuvent avoir différents objectifs :

- collecter des données sur les attaques [10],
- détecter une intrusion [11],
- détourner les attaques vers des composants moins sensibles [10],
- faire perdre aux attaquants du temps et de l'énergie [10].

Malgré les différents objectifs possibles, un pot de miel doit convaincre un éventuel attaquant de sa légitimité ou de sa valeur pour maximiser son impact du pot de miel.

4.1.2. Scénario (h2)

Un scénario définit ce quoi doit imiter le pot de miel. Cela comprend un domaine d'application, une activité, et un rôle. Le domaine d'application d'un pot de miel peut être très varié [12]: Technologie d'Information (IT), l'Internet des Objets (IoT), l'Internet Industriel des Objets (IIoT), les Réseaux Électriques Intelligents (Smart Grid), Technologie Opérationnelle (OT). Le domaine d'application influe sur ce qu'un attaquant attend d'un service légitime et de son activité, et donc sur ce qu'un pot de miel doit imiter.

Simuler de l'activité dans un pot de miel permet de renforcer sa crédibilité auprès d'un acteur extérieur. Cette activité peut prendre différente origine: appels système, simulation d'utilisateur, traffic réseau entre le pot de miel et le système légitime. Une machine de production sans aucune interaction humaine est suspecte.

Le rôle d'un pot de miel est une propriété qui peut être prise en compte. Une machine seule qui joue le rôle de services normalement présents sur des machines distinctes peut éveiller des soupçons chez un attaquant.

4.1.3. Niveaux de ressources (h3)

Les pots de miel peuvent être déployés avec différents niveaux de ressources: pot de miel virtuel, pot de miel physique ou pot de miel hybride. Il est courant de voir des pots de miel basés sur des machines virtuelles, mais les pots de miel peuvent être déployés sur des machines réelles [12]. En fonction du niveau de ressources, des techniques de détection sont possibles. Un pot de miel virtuel peut être détecté via des artefacts de virtualisation.

4.1.4. Niveau d'interaction (h4)

Le niveau d'interaction distingue les réseaux de pots de miel. De bas à haut, il caractérise la quantité d'interaction qu'un attaquant peut effectuer [10]. Les pots de miel à basse interaction, par exemple un service simple, collectent moins de données sur l'attaquant, mais sont plus faciles à implémenter et à sécuriser [12]. Les pots de miel à haute interaction, par exemple des machines réelles permettant l'accès à leurs systèmes d'exploitation [12], requièrent plus d'entretien, mais collectent des données sur le comportement de l'attaquant à plusieurs niveaux, tels que la propagation sur le réseau ou l'exploration du système [11]. La littérature fournit également des pots de miel à moyenne interaction comme compromis [11], par exemple la simulation d'un terminal de commandes.

4.1.5. Code (h5)

Franco et al. [12] et Ilg et al. [11] accordent de l'importance à la disponibilité du code. Lorsqu'un logiciel est dit *open source*, cela signifie que le code a été publié et est publiquement disponible pour modification ou implémentation [12]. Les pots de miel *open source* permettent une analyse approfondie des autres groupes de recherche [11] et des améliorations de la part d'autres développeurs [12]. Néanmoins, les pots de miel *open source* peuvent être étudiés par les attaquants afin de mieux les détecter.

4.1.6. Actions de l'attaquant (h6)

Pour des raisons de sécurité, un pot de miel ne peut pas laisser un utilisateur effectuer n'importe quelles actions [13]. Sinon, le pot de miel pourrait être utilisé comme un nouveau vecteur d'attaque par l'attaquant.

4.2. Taxonomie des techniques de détection

La taxonomie présentée dans les prochains paragraphes reposent sur onze articles de la littérature [13, 14, 15, 16, 17, 18, 7, 19, 20, 4, 21].

4.2.1. Cible (c1)

Un premier élément de la taxonomie d'une technique de détection de pot de miel est sa cible. Une technique de détection peut cibler un type de pot de miel en particulier, comme:

- une catégorie de pots de miel: par exemple, pot de miel cyber-physique[20],
- un type de pot de miel: par exemple, pot de miel à haute interaction [17],
- un pot de miel en particulier: par exemple, honeyd [15].

Cette cible peut être détermiée par la technique de détection elle-même ou être généralisable.

4.2.2. Lieux de déploiement (c2)

Une technique de détection a aussi différents lieux de déploiement.

- côté *serveur* (c2.1): la technique est utilisée dans le potentiel pot de miel pour l'identifier [18],

- côté *client* (c2.2): la technique est utilisée dans une machine tierce et cherche à interagir avec le potentiel pot de miel pour l'identifier [21],
- côté *réseau* (c2.3): la technique est utilisée dans une machine tierce et cherche à identifier le potentiel pot de miel en analysant son interaction avec un client légitime [17].

4.2.3. Origine (c3)

Une technique a une origine. Cette origine peut être un laboratoire (c3.1) comme une méthode d'évaluation [19] ou un attaquant réel (c3.2) comme une partie d'un logiciel malveillant [14].

4.2.4. Catégorie (t1)

Une technique de détection est caractérisée par sa catégorie de détection. Tay et al. proposent plusieurs catégories de détection [4]:

- *Structure* (t1.1): repose sur la topologie et les composants du réseau du pot de miel [17],
- *Temps* (t1.2): repose sur les latences du réseau [14], l'horloge du pot de miel [15],
- *Opérations* (t1.3): repose sur les opérations effectuées et les flux d'informations,
- *Matériel* (t1.4): repose sur l'identification des composants [16],
- *Environnement* (t1.5): repose sur les données du système, des fichiers, des logiciels [18], et du mode d'exposition du pot de miel [7],

4.2.5. Vecteur (t2)

Une technique de détection repose sur un vecteur de détection pour identifier le pot de miel. Un vecteur de détection est une propriété du pot de miel qui permet la détection. Ce vecteur est défini par la taxonomie des pots de miel restreinte à leur détection section 4.1.

4.2.6. Source (t3)

Une technique de détection a une source de donnée pour la détection. Cette source peut être d'origine:

- *réseau* (t3.1): par exemple, intégrité des paquet réseau [13],
- *système* (t3.2): par exemple, temps de réponse [14], utilisateurs [18], logiciels installés,
- *applicative*(t3.3): par exemple, les valeurs retournées par l'application [19].

4.2.7. Type (t4)

Une technique de détection peut être basée sur l'empreinte (t4.1) ou sur le comportement (t4.2) du pot de miel. Une technique basée sur l'empreinte (t4.1) cherchent des éléments qui ne sont pas suspects mais qui sont généralement attribués à des pots de miel: e.g. réponse identique à un pot de miel *open source* [7]. Une technique basée sur le comportement (t4.2) cherchent des éléments qui sont suspects sans certifier que la machine suspecte soit un pot de miel: e.g. une évolution anormale des réservoirs d'essence [19].

5. Revue des techniques de détection et d'évasion de pots de miel

Cette section offre une revue de la littérature des techniques de détection et d'évasion des pots de miel issues des laboratoires et des attaquants. Chacune des sous-section regroupe des techniques de détection ayant des vecteurs similaires.

La table 3 et la table 4 décrivent les techniques présentées dans le parties ci-dessous. Les articles qui introduisent des techniques provenant d'attaques sont présentés en gris. La table se repose sur la taxonomie des techniques de détection de pot de miel décrite dans la section 4.2. Les techniques de cette table sont décrites dans les sous-sections suivantes et sont regroupées selon leur vecteur (t2) de détection.

Groupe	Propriété	Valeur	Références
(c) Contexte	(c1) Cible	*Liste non exhaustive*	-
		Pot de miel à haute interaction	[18, 22, 17]
		Pot de miel CPS	[16, 20]
		Pot de miel HTTP	[23]
		Pot de miel virtuel	[23]
		Pot de miel *open-source*	[5, 7, 22]
		...	...
	(c2) Localisation	(c2.1) Côté Serveur	[14, 18, 16, 22, 13]
		(c2.2) Côté Client	[13, 24, 6, 25, 5, 19, 7, 26, 27, 8 20, 23, 15, 21, 28]
		(c2.3) Côté Réseau	[17]
	(c3) Origine	(c3.1) Laboratoire	[14, 18, 13, 7, 27, 24, 6, 25, 5, 19 26, 8, 20, 23, 22, 17, 21, 15]
		(c3.2) Attaque	[16, 14, 28]
(t) Technique	(t1) Catégorie	(t1.1) Structure	[17]
		(t1.2) Temps	[15, 14]
		(t1.3) Opérations	[13, 24, 6, 25, 5, 19, 26, 8, 20, 23 28, 18, 22]
		(t1.4) Matériel	[16, 18]
		(t1.5) Environnement	[7, 27, 21, 19, 14, 18, 13]
	(t2) Vecteur	(t2.0) Divers	-
		(t2.1) Objectif	[14, 18, 13, 17]
		(t2.2) Scénario	[7, 19, 28, 18]
		(t2.3) Niveau de Ressource	[21, 14, 18, 15]
		(t2.4) Intégration Physique	[19]
		(t2.5) Code	[7, 27, 16, 24, 6, 25, 5, 19, 26, 8 20, 23, 22]
		(t2.6) Actions permises	[14, 18, 13]
	(t3) Source	(t3.1) Réseau	[13, 18, 7, 25, 26, 8, 17, 19, 28]
		(t3.2) Application	[7, 27, 24, 6, 25, 5, 19, 20, 23, 15]
		(t3.3) Système	[14, 18, 27, 16, 25, 22, 13, 21]
	(t4) Type	(t4.1) Empreinte	[14, 7, 27, 16, 24, 6, 25, 5, 19, 26 13, 17, 18]
		(t4.2) Comportement	[18, 13, 8, 20, 23, 22, 14, 21, 15, 7 28]

Table 2
Taxonomie des techniques de détection ou d'évasion de pots de miel

5.1. Techniques reposant sur l'objectif (t2.1) du pot de miel

Étudier les outils d'un service [14, 13] ou les équipements du réseau [17] peut donner des indices sur l'objectif (t2.1) de ce service.

La détection de pot de miel peut se baser sur les indices de collecte de données [13]. Dornseif et al. démontrent qu'il est possible, sans privilèges particuliers, de détecter Sebek v2.1.6 responsable de la collecte d'information. Un attaquant peut alors de choisir de désactiver Sebek s'il en est capable ou abandonner son attaque dans le cas contraire.

En plus de la journalisation, la présence de *debuggers* peut être utilisée par un logiciel malveillant pour modifier son comportement [14]. Si un système possède des *debuggers*, alors ce système cherche à étudier des programmes et aura peu de valeur pour l'attaquant. Si un système ne possède pas de *debugger*, alors ce système a des chances d'être une machine avec de la valeur pour l'attaquant.

Analyser l'environnement réseau du pot de miel peut permettre de déceler des indices sur la nature d'un pot de miel [17]. Les pots de miel à haute interaction proposent un réseau complet pour récolter des données sur des attaques complexes. Ces pots de miel assurent aussi leur sécurité à travers un pare-feu et un système de détection d'intrusion. Wenda et al. propose une méthode plus stable que

Table 3
(Partie 1) Récapitulatif des articles présentants au moins une technique de détection ou d'évasion de pot de miel (en gris, les articles avec des techniques provenant d'attaques)

Réf. & Année	Description	Vecteur (t2)	Cible (c1)	Origine (c3)
Dornseif et al.[13] 2004	Étudie l'altération des paquets et les limitations de traffic	Actions permises	*Honeywall* restrictif	Lab.
	Détecte la présence de Sebek v2.1.6 ou antérieure	Objectif	Pot de miel basé sur Sebek (v2.1.6 or less)	Lab.
Holz et al.[14] 2005	Détecte une adresse MAC de VMWare et la backdoor de VMWare	Niveau de ressource	Pot de miel sur VMware	Agobot
	Détecte User Mode Linux au sein d'un système *host*	Niveau de ressource	Pot de miel sur UML	Lab.
	Détecte *jail* et *chroot* dans un système	Actions permises	Any	Lab.
	Détecte les système plus lent	Niveau de ressource	Pot de miel virtuel	Lab.
	Détecte les *debuggers*	Objectif	Bac à sable	Agobot
Fu et al.[15] 2006	Étudie la latence de réponse	Niveau de ressource	Pot de miel basé sur Honeyd	Lab
Boulaiche et al.[8] 2008	Etudie le protocole ARP	Code	Pot de miel basé sur Honeyd	Lab.
Wenda et al.[17] 2011	Détecte la présence de composants de honeynet	Objectif	Pot de miel à haute interaction	Lab.
Falliere et al.[16] 2011	Attaque un pot de miel selon des propriétés du système	Code	Pot de miel CPS	Stuxnet
Hayatle et al.[18] 2012	Décrit la virtualisation comme un critère de détectabilité	Niveau de ressource	Pot de miel virtuel	Lab.
	Étudie les logiciels manquants dans un système	Code	Pot de miel à haute interaction	Lab.
	Étudie l'activité des utilisateurs au sein du pot de miel	Scénario	Pot de miel à haute interaction	Lab.
	Étudie la facilité de compromission du pot de miel	Actions permises	Pot de miel	Lab.
	Étudie la capacité du pot de miel à envoyer du traffic vers l'extérieur	Actions permises	Pot de miel	Lab.
Aguirre et al.[24] 2014	Étudie les codes d'erreur à des requêtes HTTP	Code	Pot de miel HTTP	Lab.

l'analyse temporelle ou la prise d'empreinte de réponse de pots de miel vis-à-vis de la latence du réseau.

5.2. Techniques reposant sur le scénario (t2.2) du pot de miel

La méthode d'exposition [7] ainsi que ce qui est exposé [19, 27] d'un pot de miel peut être déterminant. L'activité au sein du pot de miel peut aussi rendre un pot de miel inefficace [18, 28, 9].

Les pots de miel ont des propriétés qui sont partagées d'un prototype à l'autre. Zhang et al. [27] mesurent que les pots de miel ont un plus grand nombre de ports ouverts qu'une machine classique. Les pots de miel ont aussi des réponses identifiables à des commandes spécifiques. Les pots de miel ont souvent une IP publique identifiable puisqu'elles ne sont pas enregistrées dans *Alexa*.

Zamiri et al. étudient les ports et services qui sont exposés sur une même machine [19]. Si trop de services ou des services normalement présents sur des machines différentes sont présents sur la même machine, alors la machine est suspecte.

Srinivasa et al. [7] complètent une méthode basée sur l'empreinte du pot de miel avec une méthode qui prend en compte l'hébergement du pot de miel. Typiquement, deux propriétés sont analysées: le fournisseur d'accès internet (HTTPS/HTTP/SSH/FTP/Telnet) et l'hébergeur cloud (Modbus/S7). Ces

Table 4
(Partie 2) Récapitulatif des articles présentants au moins une technique de détection ou d'évasion de pot de miel (en gris, les articles avec des techniques provenant d'attaques)

Réf. & Année	Description	Vecteur (t2)	Cible (c1)	Origine (c3)
Vetterl et al.[6] 2018	Étudie la configuration d'un service	Code	Pot de miel à basse interaction	Lab.
Huang et al.[25] 2019	Identifie un service à travers ses réponses avec du *machine learning*	Code	Pot de miel HTTP, FTP, et SMTP	Lab.
Morishita et al.[5] 2019	Étudie la configuration d'un service et ses réponses FTP	Code	Pot de miel *open source*	Lab.
Surnin et al.[22] 2019	Analyse les réactions aux commandes shell	Code	Pot de miel à haute interaction	Lab.
	Analyse des propriété-types	Code	Pot de miel *open source*	Lab.
Rrushi et al.[28] 2019	Une attaque peut baser sa méthode de propagation sur l'activité des machines d'un réseau	Scénario	Pot de miel inactif	*malware* Savvy
Zamiri et al.[19] 2019	Étudie une sélection de propriétés type de Conpot	Code	Pot de miel ICS	Lab.
	Utilise PLCScan pour déterminer les aspects manquants	Code	Pot de miel ICS	Lab.
	Étudie l'état du système et sa vraisemblance	Intégration physique	Pot de miel ICS	Lab.
	Étudie la cohérence des applications exposées sur une même machine	Scénario	Pot de miel ICS	Lab.
Guihéry et al.[9] 2020	Activité du pot de miel réaliste	Scénario	Pot de miel à haute interaction	Zebrocy, BaneChan
Sun et al.[20] 2020	Détecte les erreurs de comportements par *fuzzy testing*	Code	Pot de miel Cyber-Physique	Lab.
Srinivasa et al.[7] 2021	Cherche les propriétés courantes de pot de miel	Code	Pot de miel configuré par défaut	Lab.
	Analyse les conditions d'exposition de pot de miel	Scénario	Pot de miel déployé publiquement	Lab.
Franzen et al. [26] 2022	Étudie la pile TLS	Code	Pot de miel RDP	Lab.
	Crée des requêtes provoquant des réponses spécifiques	Code	Pot de miel RDP, Dionaea et Impacket (SMB)	Lab.
Chen et al. [23] 2023	Détecte les écarts de réponse dans les erreurs	Code	Pot de miel HTTP	Lab.
Zhang et al.[27] 2023	Compare certaines propriétés du système avec celle de pots de miel communs	Code	Pot de miel commun	Lab
Zhu et al.[21] 2024	Teste la mémoire d'un PLC	Niveau de ressource	Pot de miel OT	Lab.

données sont obtenues avec des sondeurs tiers comme Shodan [29] ou Censys [30].

Hayatle et al. suggèrent que l'activité des utilisateurs et des comptes de services et un indice de poids forts dans la détection de pots de miel [18].

Des techniques d'évasion existent sans avoir pour objectif de détecter les pots de miel particulièrement. Basée sur les *malware* Saavy, une méthode de propagation utilise les échanges de données entre plusieurs machines pour trouver une nouvelle cible [28]. Si une machine légitime est infectée, il faudra alors que le pot de miel communique avec la machine victime pour être éventuellement ciblée par le *malware*.

Un pot de miel à haute interaction doit pouvoir simuler de l'activité en son sein s'il veut déclencher certaines attaque [9]. Guihéry et al. mentionnent deux chevaux de Troie qui se déclenchent selon

l'activité du système. Le cheval de Troie Zebrocy s'exécute à la fermeture d'un document. Le cheval de Troie BaneChan compte un nombre de clics de souris avant de déclencher son attaque. Ces deux stratégies permettent d'évader des environnements de détonation et d'analyse s'ils ne sont pas assez convaincants.

5.3. Techniques reposant sur le niveau de ressource (t2.3) du pot de miel

Pour un malware, détecter qu'il est exécuté dans une machine virtuelle est un signe d'une potentielle introspection au sein d'une *sandbox* [18]. Certaines techniques basées sur des anomalies temporelles (t1.2) étudient les latences [14] et régularités [15] dans les opérations pour discerner une machine réelle d'une machine virtuelle. D'autres techniques utilisent le matériel (t1.5) d'un système, virtuel ou physique, pour identifier la nature du système [14, 16] Il est aussi possible d'analyser les outils présents dans le système pour déterminer sa nature [31]. Enfin, des erreurs peuvent apparaître dans les outils de virtualisation sans être présentes dans les versions physiques [21].

Kemkes et al. [32] décrit un ensemble de méthodes qui peuvent être employées par les malware afin de détecter que leur environnement d'exécution est une machine virtuelle. Le *Malware Behavior Catalog* (MBC) entretenu par l'organisme MITRE [33] contient également une liste de méthodes employées par les malware pour détecter des les machines virtuelles.

Utiliser des systèmes virtuels peut être détecté [14]. Holz et al. expliquent qu'un système virtualisé possède plus d'opération qu'un système physique ce qui ajoute un délai. Cette mécanique peut alors être utilisée pour déceler un système virtuel. Holz et al. donnent l'exemple d'une commande illégale: un pot de miel hébergé sur VMWare a besoin de plus de trois fois plus de temps qu'un système physique cette commande. Cette différence est expliquée par les échanges entre VMWare et le système d'accueil pour déterminer l'origine de l'erreur produite par la commande illégale.

En 2006, Honeyd présente des artéfacts d'émulation qui ont des répercussions sur la latence des réponses [15]. Typiquement, les réponses d'Honeyd ont une latence qui est un multiple de 10ms. Ceci permet alors de distinguer des systèmes réels des systèmes virtuels.

Certains composants sont identifiables facilement: e.g. les machines virtuelles déployées via VMWare peuvent être identifiées par l'adresse MAC de l'interface réseau [14]. De plus, VMWare possède une porte dérobée pour les entrées/sorties. Cette porte dérobée permet à VMWare d'interagir avec la machine virtuelle et elle est utilisée par *Agobot* pour détecter un système indésirable, e.g. un environnement de détonation de *malware*.

En premier lieu, il est possible de comparer les spécificités du matériel dans l'environnement d'exécution avec celles normalement disponibles pour les machines physiques. Les périphériques des machines virtuelles (cartes réseaux, cartes graphiques, claviers, souris, etc.) sont souvent des moyens fiables pour identifier que l'environnement d'exécution est une machine virtuelle et quel est l'hyperviseur le cas échéant. Par exemple, par défaut, les cartes réseaux VirtualBox ont souvent `08:00:27` comme identifiant constructeur dans l'adresse MAC (i.e., les trois premiers octets de l'adresse). En listant tous les périphériques, il est aussi possible de vérifier si certains contiennent des noms connus d'hyperviseur ou technologies de virtualisation (e.g., *qemu*, *vmware*, *vbox*, etc.). Cette analyse des périphériques peut aussi être menée en analysant les pilotes installés, comme par exemple ceux pour les périphériques VirtIO dans le cas de l'hyperviseur KVM [31].

Au-delà de la détection, certains *malware* peuvent évader un système indésirable en cherchant une propriété précise d'un système réel [16]. Falliere et al. montrent que *Stuxnet* possède une étape de vérification de la version du PLC ciblé. Dans ce cas de figure, le *malware* ne détecte pas le pot de miel. Cependant, si le malware est exécuté dans un pot de miel ou une *sandbox*, l'attaque ne sera pas initiée car la vérification échouera. Nous parlons alors d'évasion plutôt que de détection.

Certains malwares cherchent également à contacter les moyens de communication entre le système virtualisé et l'hôte. Par exemple, en exécutant l'instruction assembleur IN, qui permet de communiquer avec un port d'entrée sortie, sur le port `0x5658` (VX en héxadécimal), au sein d'une machine virtuelle VMware, la valeur `VMXh` sera retournée.

Enfin, au sein des systèmes d'exploitation des machines virtualisées il est possible de trouver des artefacts indiquant le type de l'hyperviseur. Ces artefacts peuvent être des fichiers (e.g., `"C:\system32\drivers\VBoxMouse.sys"` étant un fichier contenant un des drivers VirtualBox sur la machine invité), les clés de registre (e.g., la clé `"\HKLM\SOFTWARE\VMware, Inc.\VMware Tools (VMWARE)"` indique un hyperviseur VMware), des processus ou services (e.g., `"xenservice.exe"` pour les hyperviseurs Xen).

HoneyJudge est un outil qui teste la mémoire d'un *Automate programmable industriel* (PLC) pour déterminer si c'est un pot de miel [21]. Un PLC est constitué de trois mémoires: la mémoire du système d'opération, la mémoire des utilisateurs, et la mémoire des processus physiques. *HoneyJudge* compare la mémoire de PLC suspects avec la mémoire de PLC légitimes. Dans la mémoire du système d'opération,*HoneyJudge* utilise les conflits sémantiques et la structure de la mémoire. Dans la mémoire des utilisateurs, *HoneyJudge* utilise la gestion de la mémoire et les fonctions logiques. Dans la mémoire des processus physiques,*HoneyJudge* utilise le bruit des processus et les dynamiques des processus. Ainsi, *HoneyJudge* permet d'identifier *Conpot*, *HoneyPLC*, *ICSpot*, *ICSpotPlus*, et *Idling PLC* comme pots de miel tandis que d'autres outils comme *nmap* et *PLCScan* ne détectent que *Conpot*.

5.4. Techniques reposant sur l'intégration physique (t2.4) du pot de miel

Un service qui utilise un modèle afin de prétendre échanger avec le monde physique est un service suspect [19]. Un pot de miel complexe peut chercher à modéliser une situation précise. Gaspot modélise une station d'essence et en particulier ses réservoires d'essence [34]. Zamiri et al. expliquent que le modèle de ces réservoires peut être un vecteur de détection pour les attaquants [19]. Par exemple, un niveau des réservoires baisse de manière déterministe est suspect.

5.5. Techniques reposant sur le code (t2.5) du pot de miel

Implémenter tous les aspects d'une machine ou d'un service peut être couteux en temps et en argent. Des techniques basées sur la gestion des erreurs cherchent à étudier les fonctionnalités et erreurs qui n'ont pas été anticipées par les développeurs de pots de miel [8, 18, 19, 20, 23]. De plus, utiliser des pots de miel *open source* peut être un point fort mais aussi un point faible. Un pot de miel *open source* peut être analysé pour en déterminer une empreinte qui permettra d'identifier toutes ses copies [24, 6, 25, 5, 19, 7, 26, 27].

Honeyd présente plusieurs anomalies dans le protocole ARP [8]. Avec un simple ping, un pot de miel basé sur *Honeyd* avec une configuration par défaut peut être détecté. Par exemple, un *Honeyd* avec une configuration par défaut renvoie réponse en *broadcast* à une requête ARP. En temps normal, un système répond directement à la machine à l'origine de la requête plutôt qu'en *broadcast*. Hayatle et al. étudient les logiciels et composants présents au sein d'une machine [18]. Si certains éléments critiques ne sont pas présents, un attaquant pourrait comprendre que la machine n'est pas réellement utilisée et donc identifiée comme pot de miel. Un pot de miel industriel comme Conpot n'implémente pas tous les aspects du protocole Siemens [19]. Zamiri et al. suggèrent que PLCScan permet de lister les commandes qui ne sont pas implémentées dans Conpot. Étudier les commandes non implémentées peut alors permettre de distinguer service réel et pot de miel. Plus récemment, le *secure fuzzy testing* permet de détecter des pots de miel cyber-physiques par les erreurs qu'ils renvoient [20]. Leur approche utilise un réseau neuronal convolutif du modèle d'apprentissage profond. Leur approche suit un entraînement hors-ligne pour effectuer une détection de pots de miel industriel en ligne. En plus du protocole ARP, d'autres protocoles peuvent être étudier pour détecter la présence d'un pot de miel comme les pots de miel RDP. Franzen et al. décrivent une méthode de détection basée sur l'analyse de la pile TLS Un ensemble de paquets de sondage basé sur une anomalie (*PSA*) [23] peut être utilisé pour identifier des pots de miel. L'approche est basée sur un réseau neuronal convolutif sur deux jeux de service HTTP: un jeu avec des services légitimes comme Baidu ou Google, et un jeu avec des pots de miel comme HellPot ou NodePot. L'approche est évaluée sur le jeu de pots de miel avec une configuration légèrement différente de la configuration utilisée à l'entraînement.

Un pot de miel peut être détecté en fonction des erreurs qu'il crée [24]. Aguirre et al. proposent une procédure d'identification de pots de miel HTTP à basse interaction. La procédure est testée d'abord sur des pots de miel connus, puis est utilisée pour détecter des pots de miel en ligne. Ainsi, Aguirre et al. compare des services suspects avec des pots de miel commun. Un pot de miel peut être caractérisé par son manque de personnalisation [6]. Vetterl et al. montrent que des pots de miel des services SSH, Telnet, HTTP peuvent non seulement être détectés mais aussi être identifiés avec leur version. Huang et al. propose une méthode d'identification de pots de miel suivant trois sources de données: réseau (t3.1), application (t3.2), système (t3.3) [25]. Les donnés collectées concernent des paramètres liés aux services HTTP, FTP, et SMTP. Leur méthode utilise du *machine learning* pour identifier le service et atteint une précision de 0.9 et une bonne généralisation pour la technique d'apprentissage de forêt d'arbres décisionnels. Morishita et al. proposent une méthode pour étudier la détectabilité des pots de miel *open source* exposés sur Internet et leur durée de vie. Leur méthode de détection se base sur la signature des pots de miel à travers leur bannière, leur réponse HTTP et leur réponse aux erreurs. Ces trois vecteurs de détection sont comparés avec Shodan [29] et Censys [30] qui permettent eux aussi d'évaluer les adresses IP suspectes. Ainsi, plus de 19,000 pots de miel *open source* exposés en ligne peuvent être détectés à cause de leur configuration par défaut. En guise de mitigation, Morishita et al. suggèrent de remplacer les bannières des pots de miel par des bannières standardes, personnaliser les fichiers HTML pour le contenu Web, et de corriger la gestion des erreurs pour les services SSH pour un résultat plus proche de la réalité. L'étude de la configuration des pots de miel touche aussi les pots de miel industriel comme Conpot [19]. Zamiri et al. proposent une signature de Conpot. Cette signature est constituée des bannières par défaut qui comprennent adresse MAC, identifiant, numéro de série, ou nom du produit. Zamiri et al. expliquent que ces paramètres sont décrits par Shodan [29] et Censys [30]. Srinivasa et al. [7] proposent une méthode de détection de pot de miel basé sur son empreinte. Leur méthode repose sur l'analyse de réponse d'un service à différente requête. Si les réponses obtenues correspondent à celle d'un pot de miel, le service est identifié comme tel. La méthode inclut différents modules de détection présents dans la littrature [35, 6, 5]: un scan de port, un test de la bannière, une analyse de la réponse HTTP, un test des dépendances de librairies, et une analyse de réponse de commands statiques. Néanmoins, leur méthode introduit deux nouveaux modules: une analyse de l'établissement de connexion et un test de certificat SSL/TLS. Franzen et al. utilise la méthode sur des pots de miel RDP et SMB [26]. En forgeant correctement certaines requêtes, un attaquant peut identifier un pot de miel avec les réponses obtenues.

5.6. Techniques reposant sur les actions permises (t2.6) vis-à-vis du pot de miel

Un pot de miel est un équipement qui se veut vulnérable pour attirer des attaquants. Néanmoins, un pot de miel ne doit pas être un vecteur d'attaque supplémentaire. Des comportements étranges comme des règles de sécurité supplémentaire naissent de ce paradoxe [13, 14, 18] D'autre part, si le pot de miel est trop facilement compromis, l'attaquant pourrait se méfier [18]

Un *honeywall* joue le rôle de pare-feu pour un composant d'un pot de miel ou pour son réseau de pots de miel. Dornseif et al. [13] expliquent que le *honeywall* peut avoir des règles pour restreindre les actions de l'attaquant. En particulier, les règles du honeywall peuvent chercher à éviter qu'un attaquant compromette le pot de miel ou l'utilise comme machine-rebond. Ces restrictions sont distinguées en deux cas: une limite de paquets sortant du pot de miel et l'altération des paquest pour transformer des requêtes dangereuses en requêtes bénines. Il en découle alors deux techniques. La première consiste à ouvrir une connexion depuis le pot de miel et envoyer de multiples paquets pour voir s'il y a un bloquage. Cette première technique est aussi suggérée par Hayatle et al. [18]. La seconde consiste à étudier l'altération des paquets émis.

En outre, une sécurité particulière peut éveiller des soupçons [14]. Holz et al. montrent qu'un attaquant peut détecter la présence d'une prison ou d'un *chroot* au sein d'un système d'exploitation. La prison et le *chroot* permettent de restreindre un utilisateur à une partie très réduite d'un système. Sans installation préalable, même les commandes les plus simples ne sont pas présentes dans une prison ou d'un *chroot*. Ceci limite au maximum les actions permises à un attaquant.

Table 5
Synthèse des techniques de détections avec leurs mitigations (en gris, les cibles avec des techniques provenant d'attaques)

Cible (c1)	Catégorie (t1)	Vecteur (t2)	Origine (c3)	Mitigation
Pot de miel CPS	Opérations	Code	Lab. [20, 19]	Implémenter les fonctionnalités manquantes
		Niveau de ressource	Lab. [21]	
		Intégration Physique	Lab. [19]	Introduire du bruits [19]
	Matériel	Scénario	Stuxnet [16]	
	Environnement	Code	Lab. [19]	
Pot de miel HTTP	Environnement	Code	Lab. [23]	Prévoir les erreurs typiques
Honeywall restrictif	Opérations	Actions permises	Lab. [13]	
Honeyd	Opérations	Code	Lab. [8]	Ajouter 3 modules [8]: allocation d'adresse IP, générateur d'adresse MAC, traducteur d'adresse MAC
	Temps	Niveau de ressource	Lab. [15]	
Pot de miel à haute interaction	Structure	Scénario	Lab. [17]	
	Environnement	Code	Lab. [18]	
	Opérations	Scénario	Lab. [18]	
			Zebrocy[9]	Interagir avec des documents
			BaneChan[9]	Clics de souris
Pot de miel virtuel	Temps	Niveau de ressource	Lab. [14]	
Pot de miel basé sur Sebek (v2.1.6)	Opérations	Objectif	Lab. [13]	Sebek v2.1.7 ou plus [13]
Pot de miel basé sur UML	Environnement	Niveau de ressource	Lab. [14]	
Pot de miel basé sur VMWare	Matériel	Niveau de ressource	Agobot [14]	Modifier les adresses MAC
Pot de miel open-source	Environnement	Code	Lab. [24, 6, 25, 5, 19, 7, 26, 27]	Modifier les réponses et bannières
Toute cible	Opérations	Actions permises	Lab. [14, 18]	
		Scénario	*malware* Savvy[28]	Émettre du traffic
		Objectif	Agobot [14]	Masquer les intentions réelles du pot de miel
	Environnement	Sénario	Lab. [18]	Rendre le pot de miel moins facile à compromettre

En plus de l'analyse des traffics sortants malveillants, une sécurité trop basse peut aussi éveiller des soupçons [18].

6. Synthèse

Cette section propose une synthèse des techniques de détection et d'évasion de pot de miel avec les mitigations suggérées par la littérature.

La table 5 récapitule les techniques de détection et d'évasion selon les propriétés ciblées d'un pot de miel avec leur mitigation. Dans cette table, les lignes grises ont des techniques de détection provenant

d'attaque réelles décrites dans la littérature. Bien que toutes les catégories soient représentées: Opérations (15 sur 33 techniques), Environnement (13 sur 33 techniques), Matériel (2 sur 33 techniques), Temps (2 sur 33), et Structure (1 sur 33); la table 2 et la table 5 montrent que les techniques liées aux catégories Opérations et Environnement sont prédominantes. En outre, la littérature présente en majorité des techniques de détection produite en laboratoire (c3.1) (27 sur 33) contre 6 techniques provenant de logiciels malveillants (c3.2). Une parties des techniques de détection sont proposées avec leur mitigation (9 sur 33).

Il est crucial de masquer l'objecif réel d'un pot de miel. Cela concerne les techniques de *debugging* [14] ou de journalisation [13].

La scénario dans lequel un pot de miel est implémenté est déterminant pour ne pas éveiller les soupçon d'un attaquant. Cela concerne la difficulté de compromission [18], le rôle que joue le pot de miel au sein d'un réseau [28], ou l'activité présente dans le pot de miel [9].

Les traces de virtualisation peut aussi être un indice pour les attaquants. Cette propriété est détectable via des techniques variées [32]: adresse MAC connue [14], latence trop régulière [15] ou trop importante [14], pilote installés [31], erreur de virtualisation [21], ou processus-type de machines virtuelles [14] La virtualisation reste un indice subjectif puisqu'une machine industrielle peut être virtualisée. Il est alors nécessaire de choisir une machine qui corresponde au scénario sélectionné.

Les attaquants peuvent aussi analyser leurs actions et les impacts sur le monde physique. Si le modèle est incohérent ou suspect [19], l'attaquant pourrait abandonner son attaque. Utiliser un jumeau numérique ou un système réel pourrait être envisagé pour convaincre un attaquant.

De nombreuses techniques de détection de pots de miel utilisent le code de pot de miel. Les techniques se répartissent en deux familles: les techniques reposant sur une implémentation incorrecte [8] ou sur une configuration par défaut [27]. Il est nécessaire de personnaliser la configuration d'un pot de miel *open source*. Aussi, des implémentations de pots de miel peuvent être développées pour générer moins d'erreurs propres aux pots de miel.

Certains attaquants savent qu'un pot de miel ne peut pas répéter une attaque. C'est pourquoi certaines techniques reposent sur l'analyse de la sécurité qui est déployée dans un pot de miel [18]. Un pot de miel devrait alors présenter une sécurité suffisamment forte qui l'empêche de produire une attaque tout en garantissant un cas crédible pour l'attaquant.

7. Conclusion

Cet article présente une systématisation des connaissances des techniques de détection de pot de miel. Cet article propose une taxonomie des techniques de détection de pot de miel en section 4 et une analyse des techniques de détection étudiées en laboratoire et utilisées par les attaquants en section 5. La taxonomie et l'analyse donnent des clés de compréhension et des pistes pour améliorer le réalisme des pots de miel vis-à-vis les attaquants avancés.

Un pot de miel possède plusieurs propriétés sur lesquelles un attaquant peut reposer une technique de détection: Objectif (t2.1), Scénario (t2.2), Niveau de Ressource (t2.3), Intégration Physique (t2.4), Code (t2.5) ou Actions Permises (t2.6). Les défenseurs des sytèmes d'information utilisent principalement les indices de virtualisation (t2.3), la gestion des erreurs et les similarités d'un pot de miel avec un pot de miel *open source* (t2.5) pour évaluer le réalisme d'un prototype de pot de miel. Toutefois, les attaquants semblent plutôt s'intéresser aux outils (t2.1) du pot de miel, à son activité (t2.2), et aux indices de virtualisation (t2.3).

Cependant, très peu d'articles mentionnent explicitement les techniques issues des attaquants. Ce phénomène a un fort impact sur les stratégies que les défenseurs peuvent adopter. Étudier les techniques réellement utilisées est une piste cruciale pour améliorer le réalisme des pots de miel.

Remerciements

Cette recherche est financée par la chaire industrielle de recherche Cybersécurité des Infrastructures

Critiques (CyberCNI).

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatBotChatApp in order to: Draft structure. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] X. Chen, J. Andersen, Z. M. Mao, M. Bailey, J. Nazario, Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware, in: 2008 IEEE international conference on dependable systems and networks with FTCS and DCC (DSN), IEEE, 2008, pp. 177–186.

[2] S. Laurén, V. Leppänen, S. Rauti, J. Uitto, A survey on anti-honeypot and anti-introspection methods, Recent Advances in Information Systems and Technologies 2 (2017) 11–13.

[3] A. Afianian, S. Niksefat, B. Sadeghiyan, D. Baptiste, Malware dynamic analysis evasion techniques: A survey, ACM Computing Surveys (CSUR) 52 (2019) 1–28.

[4] V. Tay, X. Li, D. Mashima, B. Ng, P. Cao, Z. Kalbarczyk, R. K. Iyer, Taxonomy of fingerprinting techniques for evaluation of smart grid honeypot realism, in: 2023 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), IEEE, 2023, pp. 1–7.

[5] S. Morishita, T. Hoizumi, W. Ueno, R. Tanabe, C. Gañán, M. J. Van Eeten, K. Yoshioka, T. Matsumoto, Detect me if you… oh wait. an internet-wide view of self-revealing honeypots, in: 2019 IFIP/IEEE symposium on integrated network and service management (IM), IEEE, 2019, pp. 134–143.

[6] A. Vetterl, R. Clayton, Bitter harvest: Systematically fingerprinting low-and medium-interaction honeypots at internet scale, in: 12th USENIX Workshop on Offensive Technologies (WOOT 18), 2018.

[7] S. Srinivasa, J. M. Pedersen, E. Vasilomanolakis, Gotta catch'em all: a multistage framework for honeypot fingerprinting.(2021), arXiv preprint cs.CR/2109.10652 (2021).

[8] A. Boulaiche, K. Adi, Honeyd detection via abnormal behaviors generated by the arpd daemon., in: SECRYPT, 2008, pp. 65–71.

[9] F. Guihéry, A. Siffer, J. Paillard, Beezh: une plateforme de détonation réaliste pour l'analyse des modes opératoires d'attaquants (2020).

[10] P. Lackner, How to mock a bear: Honeypot, honeynet, honeywall & honeytoken: A survey, volume 2, Science and Technology Publications, Lda, 2021, pp. 181–188. doi:10.5220/0010400001810188.

[11] N. Ilg, P. Duplys, D. Sisejkovic, M. Menth, Survey of contemporary open-source honeypots, frameworks, and tools, Journal of Network and Computer Applications (2023) 103737.

[12] J. Franco, A. Aris, B. Canberk, A. S. Uluagac, A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems, IEEE Communications Surveys & Tutorials 23 (2021) 2351–2383.

[13] M. Dornseif, T. Holz, C. N. Klein, Nosebreak-attacking honeynets, 2004.

[14] T. Holz, F. Raynal, Detecting honeypots and other suspicious environments, in: Proceedings from the sixth annual IEEE SMC information assurance workshop, IEEE, 2005, pp. 29–36.

[15] X. Fu, W. Yu, D. Cheng, X. Tan, K. Streff, S. Graham, On recognizing virtual honeypots and countermeasures, in: 2006 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing, IEEE, 2006, pp. 211–218.

[16] N. Falliere, L. O. Murchu, E. Chien, et al., W32. stuxnet dossier, White paper, symantec corp., security response 5 (2011) 29.

[17] D. Wenda, D. Ning, A honeypot detection method based on characteristic analysis and environment detection, in: 2011 International Conference in Electrics, Communication and Automatic Control Proceedings, Springer, 2011, pp. 201–206.
[18] O. Hayatle, A. Youssef, H. Otrok, Dempster-shafer evidence combining for (anti)-honeypot technologies, Information Security Journal: A Global Perspective 21 (2012) 306–316.
[19] M.-R. Zamiri-Gourabi, A. R. Qalaei, B. A. Azad, Gas what? i can see your gaspots. studying the fingerprintability of ics honeypots in the wild, in: Proceedings of the fifth annual industrial control system security (icss) workshop, 2019, pp. 30–37.
[20] Y. Sun, Z. Tian, M. Li, S. Su, X. Du, M. Guizani, Honeypot identification in softwarized industrial cyber–physical systems, IEEE Transactions on Industrial Informatics 17 (2020) 5542–5551.
[21] H. Zhu, M. Liu, B. Chen, X. Che, P. Cheng, R. Deng, Honeyjudge: A plc honeypot identification framework based on device memory testing, IEEE Transactions on Information Forensics and Security (2024).
[22] O. Surnin, F. Hussain, R. Hussain, S. Ostrovskaya, A. Polovinkin, J. Lee, X. Fernando, Probabilistic estimation of honeypot detection in internet of things environment, in: 2019 International Conference on Computing, Networking and Communications (ICNC), IEEE, 2019, pp. 191–196.
[23] X. Chen, B. Lu, R. Sun, M. Jiang, Honeypot detection method based on anomalous requests response differences, in: Proceedings of the 2023 6th International Conference on Electronics, Communications and Control Engineering, 2023, pp. 109–117.
[24] E. Aguirre-Anaya, G. Gallegos-Garcia, N. S. Luna, L. A. V. Vargas, A new procedure to detect low interaction honeypots, International Journal of Electrical and Computer Engineering 4 (2014) 848.
[25] C. Huang, J. Han, X. Zhang, J. Liu, Automatic identification of honeypot server using machine learning techniques, Security and Communication Networks 2019 (2019) 2627608.
[26] F. Franzen, L. Steger, J. Zirngibl, P. Sattler, Looking for honey once again: Detecting rdp and smb honeypots on the internet, in: 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE, 2022, pp. 266–277.
[27] Y.-J. Zhang, W.-J. Liu, K.-N. Guo, Y.-M. Kang, Identification of ssh honeypots using machine learning techniques based on multi-fingerprinting, in: 2023 IEEE 6th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), volume 6, IEEE, 2023, pp. 1376–1381.
[28] J. Rrushi, Honeypot evader: Activity-guided propagation versus counter-evasion via decoy os activity, in: Proceedings of the 14th IEEE International Conference on Malicious and Unwanted Software, 2019.
[29] Shodan. honeypot or not?, https://honeyscore.shodan.io/, ???? Accessed: 2025-08-06.
[30] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, J. A. Halderman, A search engine backed by internet-wide scanning, in: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, 2015, pp. 542–553.
[31] Y. Benditovich, Y. Vugenfirer, V. Rozenfeld, L. Prosek, V. Prutyanov, V. Chulak, G. McRae, K. Kostiuk, benyamin-codez, M. Dráb, bish22ah, S. Bykov, A. Li, zjmletang, B. Salman, ivellioscolin, 6-dehan, M. Kedzierski, Twentylives, L. You, A. Odaki, J. Kohler, PolloLoco, H. Xiang, G. Lamm, M. Janiszewski, kfir manor, 54shady, yach-yf, xuxi0513, virtio-win/kvm-guest-drivers-windows, https://github.com/virtio-win/kvm-guest-drivers-windows, 2025. URL: https://github.com/virtio-win/kvm-guest-drivers-windows.
[32] P. Kemkes, Evaluation of current virtual machine detection methods, in: 14. GI FG SIDAR Graduierten-Workshop über Reaktive Sicherheit, 2019, p. 15.
[33] MITRE, Malware behavior catalog, ???? URL: https://github.com/MBCProject/mbc-markdown.
[34] K. Wilhoit, S. Hilt, The gaspot experiment: Unexamined perils in using, blackhat (2015).
[35] Z. Durumeric, E. Wustrow, J. A. Halderman, {ZMap}: Fast internet-wide scanning and its security applications, in: 22nd USENIX Security Symposium (USENIX Security 13), 2013, pp. 605–620.