Mise en place d’un honeypot avec OpenCanary

Projet personnel

1. Présentation générale du projet

1.1 Contexte du projet

Ce projet correspond au **dernier projet personnel réalisé** dans le cadre de mon apprentissage en informatique, en complément des projets scolaires du BTS SIO option SLAM. Il ne s’agit pas d’un projet imposé, mais d’une démarche volontaire visant à approfondir mes connaissances en **sécurité informatique** et en **analyse des comportements réseau**.

L’objectif principal était de mettre en place un système simple mais réaliste permettant d’observer et de comprendre comment se manifestent certaines tentatives d’intrusion sur un réseau local.

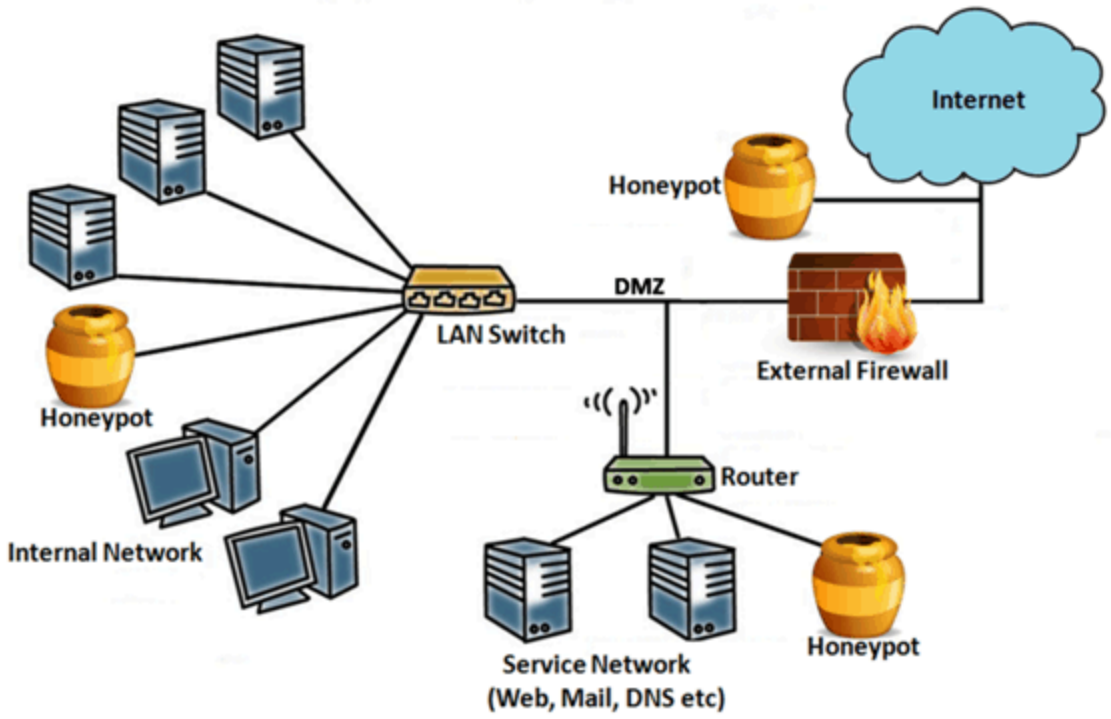
1.2 Définition d’un honeypot

Un **honeypot** est un système informatique volontairement exposé sur un réseau afin d’attirer des attaquants potentiels.
Contrairement à un serveur classique, il n’a **pas vocation à fournir un service réel** à des utilisateurs légitimes.

Son rôle principal est de :

- détecter des tentatives d’accès non autorisées,
- enregistrer les actions effectuées par un attaquant,
- collecter des informations utiles à l’analyse de la menace.

Toute interaction avec un honeypot est considérée comme **suspecte**, car aucun utilisateur normal n’est censé s’y connecter.



1.3 Intérêt d’un honeypot en cybersécurité

L’utilisation d’un honeypot présente plusieurs intérêts :

- il permet de **détecter rapidement des comportements anormaux**,
- il fournit des logs détaillés exploitables pour l’analyse,
- il n’impacte pas les systèmes de production,
- il sert d’outil pédagogique pour comprendre les méthodes d’attaque courantes.

Dans un contexte d’apprentissage, le honeypot est particulièrement pertinent car il permet d’observer concrètement ce qui est souvent vu de manière théorique en cours.

1.4 Choix de la solution OpenCanary

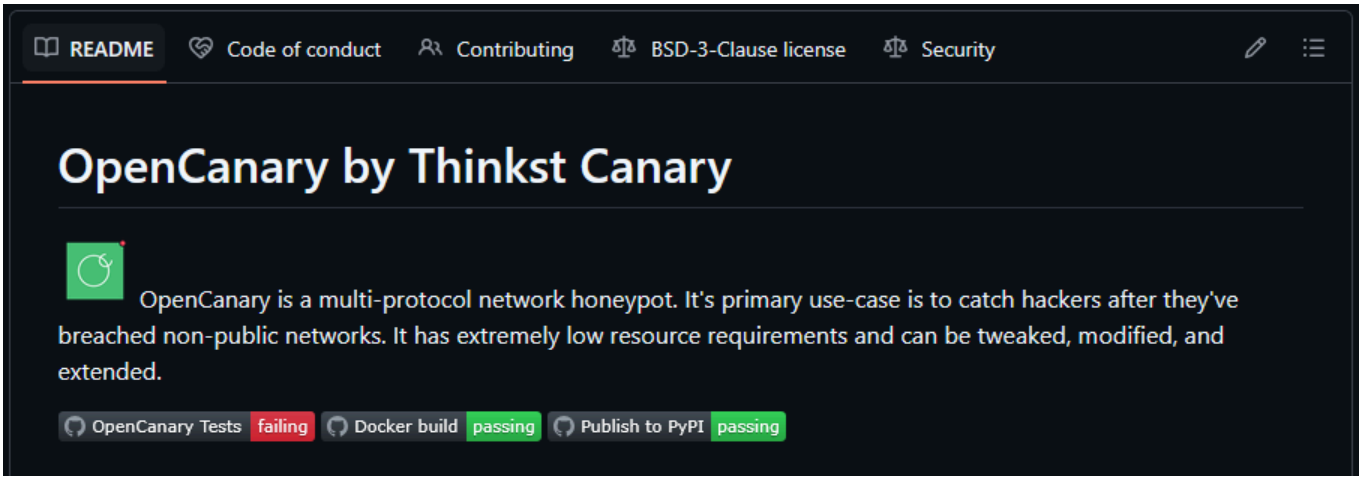
Pour ce projet, la solution **OpenCanary** a été retenue.
Il s’agit d’un honeypot léger, modulaire et facile à déployer sur un système Linux.

Les raisons de ce choix sont les suivantes :

- configuration simple via un fichier JSON,
- possibilité d’activer uniquement les services souhaités,
- génération de logs structurés au format JSON,

- outil largement utilisé à des fins pédagogiques.

OpenCanary permet ainsi de se concentrer sur l'essentiel : la **détection et l'analyse des tentatives d'intrusion**, sans complexité inutile.



1.5 Objectif de cette documentation

Cette documentation a pour but de :

- présenter le fonctionnement général d'un honeypot,
- décrire l'installation et la configuration d'OpenCanary,
- expliquer les choix techniques réalisés,
- montrer comment analyser les événements détectés.

Elle sert également de support de présentation pour mon **portfolio**, afin de valoriser un projet personnel orienté sécurité informatique.

---

2. Objectifs du projet et périmètre technique

2.1 Objectifs généraux du projet

L'objectif principal de ce projet est de mettre en place un **honeypot fonctionnel** afin d'observer et d'analyser des tentatives d'accès non autorisées sur un réseau.

Ce projet ne vise pas à bloquer ou contrer une attaque, mais uniquement à :

- détecter des connexions suspectes,

- enregistrer les actions réalisées,
- analyser les informations collectées.

Il s’agit donc d’une approche **passive**, centrée sur l’observation et la compréhension des comportements malveillants.

2.2 Objectifs techniques

Sur le plan technique, les objectifs fixés étaient les suivants :

- déployer un honeypot sur une machine Linux,
- simuler un service SSH accessible sur un port non standard,
- simuler un service HTTP accessible sur un port dédié,
- centraliser les événements dans un fichier de logs,
- exploiter ces logs pour analyser les tentatives détectées.

Chaque objectif a été volontairement limité afin de garder une configuration claire et maîtrisée.

2.3 Services volontairement exposés

Deux services ont été activés dans le cadre de ce projet :

Service SSH

- Port utilisé : **2222**
- Objectif : détecter les tentatives de connexion SSH (scan, login, brute force basique)
- Intérêt : le protocole SSH est une cible fréquente des attaques automatisées

Service HTTP

- Port utilisé : **8080**
- Objectif : simuler la présence d’un service web
- Intérêt : observer les accès HTTP et les scans de ports web

Ces services ne donnent accès à **aucun contenu réel** et servent uniquement de points d’observation.

2.4 Périmètre volontairement limité

Le périmètre du projet a été volontairement restreint afin de rester cohérent avec un projet personnel :

- environnement local (réseau privé),
- une machine attaquante,
- une machine honeypot,
- pas d’exposition directe à Internet.

Ce choix permet de se concentrer sur le fonctionnement du honeypot sans introduire de risques supplémentaires.

2.5 Approche pédagogique retenue

Ce projet a été conçu avec une logique d’apprentissage progressif :

- comprendre ce qu’est un honeypot avant de le déployer,
- limiter les services pour mieux analyser les résultats,
- privilégier la lecture et l’interprétation des logs,
- éviter toute configuration inutile ou non comprise.

Cette approche permet de relier directement la pratique aux notions vues en cours, notamment en réseau et en sécurité informatique.

---

3. Architecture du projet et environnement utilisé

3.1 Vue d’ensemble de l’architecture

L’architecture mise en place pour ce projet repose sur un environnement simple composé de **deux machines distinctes** reliées sur le même réseau.
Cette séparation permet de reproduire un scénario réaliste dans lequel un attaquant tente d’accéder à un système distant.

Le honeypot est volontairement isolé sur sa propre machine afin d’éviter toute interaction avec un système de production réel.

3.2 Description des machines utilisées

Machine attaquante

- Système d'exploitation : Windows
- Rôle : simuler un attaquant ou un utilisateur malveillant
- Outils utilisés :
  - client SSH en ligne de commande
  - navigateur web pour les tests HTTP
  - commandes réseau de base (ping)

Cette machine permet de générer différentes tentatives d'accès vers le honeypot.

Machine honeypot

- Système d'exploitation : Linux
- Rôle : héberger le honeypot OpenCanary
- Services simulés :
  - SSH sur le port 2222
  - HTTP sur le port 8080
- Fichiers importants :
  - fichier de configuration OpenCanary
  - fichier de logs des événements détectés

Cette machine ne fournit aucun service réel et n'a pour but que la détection et l'enregistrement des interactions.

3.3 Organisation réseau

Les deux machines se trouvent sur le même réseau local, ce qui permet :

- une communication directe entre les systèmes,
- des tests rapides sans configuration complexe,
- une analyse claire des flux réseau.

Les ports ouverts sur la machine Linux sont strictement limités aux services simulés par OpenCanary.

3.4 Flux de communication

Les flux réseau observés dans ce projet sont les suivants :

- Windows → Linux : tentative de connexion SSH sur le port 2222
- Windows → Linux : requêtes HTTP sur le port 8080
- Linux → système de fichiers : écriture des événements dans les logs

Chaque tentative est immédiatement interceptée par OpenCanary et enregistrée sans permettre d'accès réel.

3.5 Emplacements clés sur la machine Linux

Afin de mieux comprendre l'architecture logicielle, certains emplacements sont essentiels :

- fichier de configuration :
  - `/etc/opencanaryd/opencanary.conf`
- fichier de logs :
  - `/var/log/opencanary.log`
- service OpenCanary :
  - lancé et arrêté via la commande `opencanaryd`

Cette organisation claire facilite la maintenance, la lecture des logs et les modifications de configuration.

3.6 Justification des choix d'architecture

L'architecture retenue répond à plusieurs objectifs :

- simplicité de mise en place,
- lisibilité des échanges réseau,
- séparation claire des rôles,
- absence de risque pour un système réel.

Elle constitue une base solide pour comprendre le fonctionnement d'un honeypot avant d'envisager des architectures plus complexes.

---

4. Installation et mise en place d’OpenCanary

4.1 Préparation de l’environnement Linux

Avant l’installation d’OpenCanary, la machine Linux doit être correctement préparée.
Les opérations suivantes sont réalisées avec un compte disposant des droits administrateur.

La mise à jour du système permet de garantir la compatibilité des paquets installés et d’éviter des problèmes de dépendances.

```
sudo apt update sudo apt upgrade -y
```

Cette étape assure que le système est à jour avant l’installation du honeypot.

4.2 Installation d’OpenCanary

OpenCanary est installé à partir des dépôts disponibles sur le système Linux.

```
sudo apt install opencanary -y
```

Une fois l’installation terminée, le service OpenCanary est disponible sur la machine, mais n’est pas encore opérationnel tant qu’aucune configuration valide n’est définie.

4.3 Installation de l’outil jq

Les logs générés par OpenCanary sont au format JSON.
Afin de faciliter leur lecture et leur analyse, l’outil **jq** est installé.

```
sudo apt install jq -y
```

`jq` permet de formater, filtrer et exploiter facilement les fichiers JSON depuis la ligne de commande.

4.4 Vérification de l’installation

Après l’installation, il est important de vérifier que les outils sont bien fonctionnels.

```
opencanaryd --help jq --version
```

Ces commandes permettent de confirmer :

- que le service OpenCanary est correctement installé,
- que l'outil jq est disponible pour l'analyse des logs.

4.5 Emplacement des fichiers importants

Une fois OpenCanary installé, plusieurs fichiers et répertoires sont utilisés dans le cadre du projet :

- fichier de configuration principal :
  - `/etc/opencanaryd/opencanary.conf`
- fichier de logs :
  - `/var/log/opencanary.log`
- service OpenCanary :
  - contrôlé via la commande `opencanaryd`

Ces emplacements seront utilisés tout au long du projet pour la configuration, les tests et l'analyse.

4.6 Principe de fonctionnement après installation

À ce stade :

- OpenCanary est installé,
- aucun service n'est encore actif tant que la configuration n'est pas définie,
- le honeypot n'écoute aucun port.

La prochaine étape consiste donc à **configurer OpenCanary**, en définissant précisément les services à simuler et le mode de journalisation utilisé.

---

5. Configuration d'OpenCanary et choix d'une configuration minimale

---

5.1 Principe de la configuration OpenCanary

OpenCanary fonctionne à partir d'un fichier de configuration au format **JSON**.
Ce fichier permet de définir précisément :

- les services à activer,
- les ports utilisés,
- les paramètres réseau,
- la gestion des logs.

Par défaut, OpenCanary propose une configuration complète intégrant de nombreux services. Dans le cadre de ce projet, ce fonctionnement n'a pas été retenu.

5.2 Suppression de la configuration par défaut

Le fichier de configuration initial fourni par OpenCanary a été **entièrement supprimé**.

Ce choix a été volontaire et motivé par plusieurs raisons :

- éviter l'activation de services non compris ou inutiles,
- limiter la surface d'exposition du honeypot,
- garder une configuration lisible et maîtrisée,
- se concentrer uniquement sur les services réellement étudiés.

Cette approche permet de mieux comprendre chaque paramètre utilisé et d'éviter toute configuration automatique non maîtrisée.

5.3 Emplacement du fichier de configuration

Le fichier de configuration utilisé dans ce projet est situé à l'emplacement suivant :

`/etc/opencanaryd/opencanary.conf`

Ce fichier est lu par OpenCanary à chaque démarrage du service.

5.4 Configuration minimale utilisée

La configuration retenue active uniquement les services nécessaires au projet.

```
{ "device.node_id": "opencanary-1", "device.listen_addr": "0.0.0.0",
"ssh.enabled": true, "ssh.port": 2222, "http.enabled": true, "http.port": 8080,
```

```
"logger": { "class": "PyLogger", "kwargs": { "handlers": { "file": { "class": "logging.FileHandler", "filename": "/var/log/opencanary.log" } } } } }
```

5.5 Explication des paramètres principaux

Identité du honeypot

- `device.node_id` : identifiant du honeypot
  Il permet de reconnaître la machine dans les logs, notamment dans un environnement comprenant plusieurs honeypots.
- `device.listen_addr` : adresse d'écoute
  La valeur `0.0.0.0` indique que le honeypot écoute sur toutes les interfaces réseau disponibles.

Service SSH

- `ssh.enabled` : active le service SSH simulé
- `ssh.port` : définit le port d'écoute du service

Le port **2222** a été choisi afin d'éviter le port SSH standard (22) et d'observer plus facilement les tentatives de connexion ciblées.

Service HTTP

- `http.enabled` : active le service HTTP simulé
- `http.port` : définit le port d'écoute du service

Le port **8080** est couramment utilisé pour les services web alternatifs, ce qui en fait une cible intéressante pour les scans réseau.

Journalisation des événements

- `logger.class` : type de logger utilisé
- `FileHandler` : écriture des logs dans un fichier
- `filename` : emplacement du fichier de logs

Les événements sont enregistrés au format **JSON**, ce qui permet une analyse simple et automatisable.

5.6 Validation de la configuration

Une fois le fichier enregistré :

- la configuration est prête à être utilisée,
- aucun service n'est encore actif tant que le honeypot n'est pas démarré,
- une erreur de syntaxe dans le fichier empêcherait OpenCanary de se lancer.

La prochaine étape consiste donc à démarrer le honeypot et à vérifier que les services configurés sont bien actifs.

---

6. Démarrage du honeypot et vérification du fonctionnement

6.1 Activation du honeypot

Après la mise en place de la configuration minimale, le honeypot est lancé à l'aide de la commande suivante :

```
opencanaryd --start
```

Une fois le service démarré, OpenCanary commence à écouter les ports configurés et à enregistrer toute interaction entrante.

6.2 Tentative de connexion depuis la machine Windows

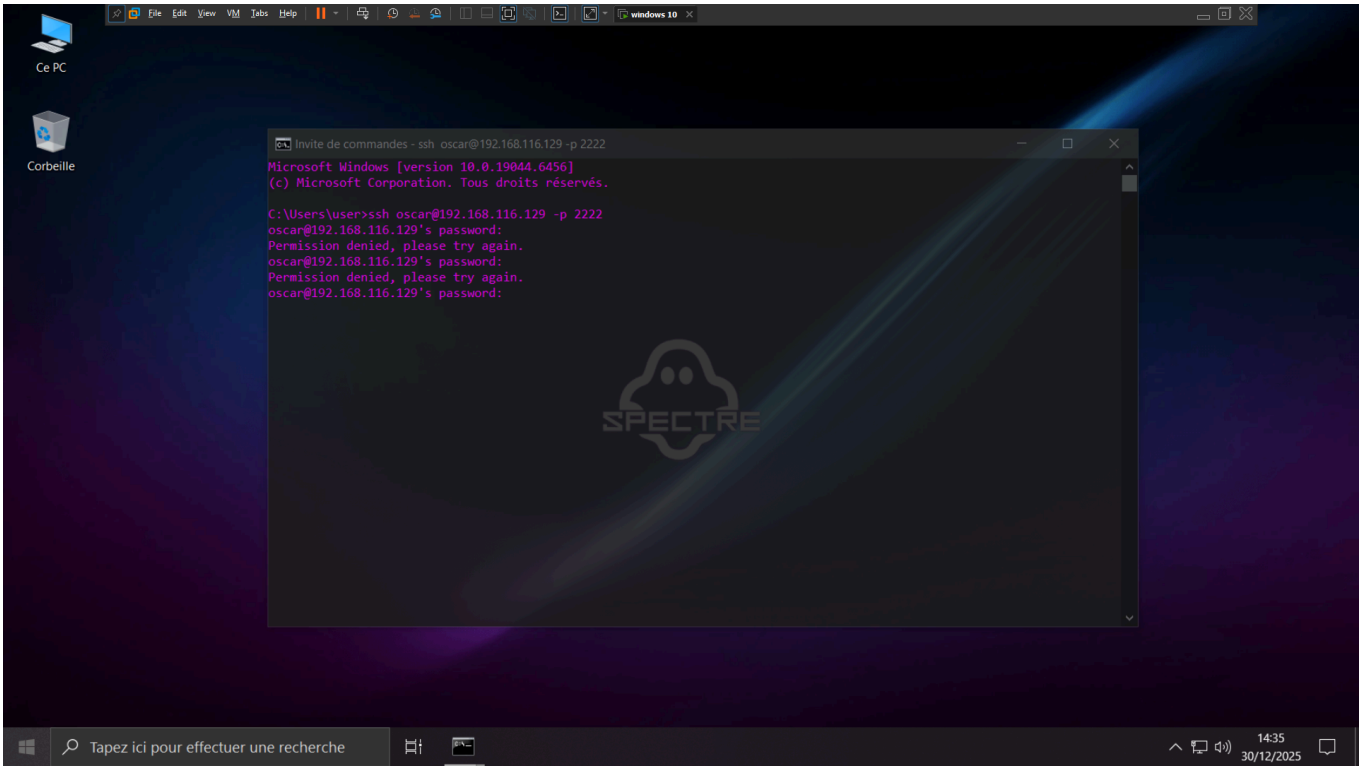
Depuis la machine Windows, une tentative de connexion SSH est effectuée vers la machine Linux hébergeant le honeypot.

```
ssh test@192.168.116.129 -p 2222
```

La connexion est refusée avec le message *permission denied*.
Ce comportement est normal : le service SSH simulé par le honeypot ne permet **aucun accès**

réel au système.

Cette tentative correspond à une action typique d'un attaquant cherchant à accéder à un service distant.

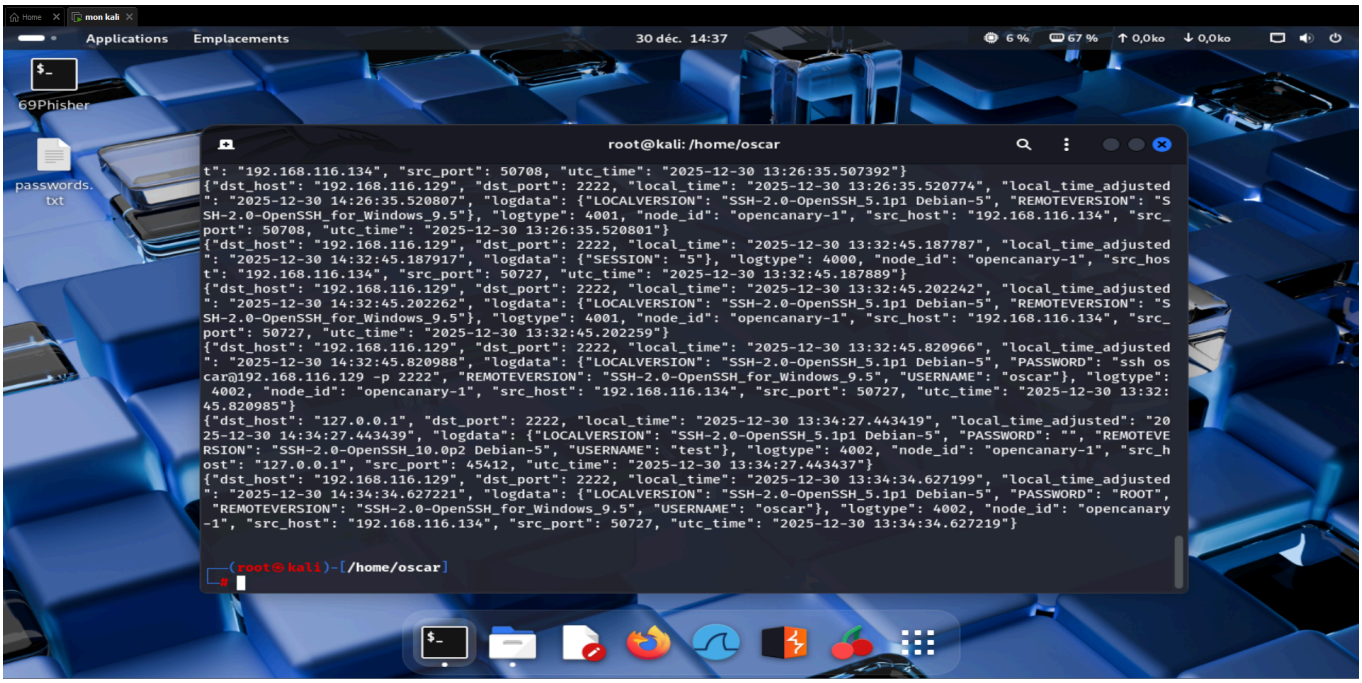


6.3 Enregistrement de la tentative dans les logs

Bien que l'accès soit refusé côté attaquant, la tentative de connexion est **correctement détectée et enregistrée** par OpenCanary.

Les informations relatives à cette connexion apparaissent dans le fichier de logs du honeypot, notamment :

- l'adresse IP source,
- le port ciblé,
- le service concerné (SSH),
- l'horodatage de la tentative.



Cette correspondance entre la tentative de connexion et son apparition dans les logs confirme le bon fonctionnement du honeypot.

6.4 Validation du fonctionnement du honeypot

Cette phase de test permet de valider que :

- la machine Windows peut atteindre le honeypot,
- les tentatives de connexion sont refusées comme prévu,
- chaque tentative est bien enregistrée dans les logs,
- le honeypot joue correctement son rôle de système de détection.

Le honeypot est désormais pleinement opérationnel et prêt pour une analyse plus approfondie des événements détectés.

---

7. Analyse des logs et exploitation avec jq

7.1 Emplacement et rôle des fichiers de logs

OpenCanary enregistre l'ensemble des événements détectés dans un fichier de logs situé à l'emplacement suivant :

`/var/log/opencanary.log`

Ce fichier contient les traces de toutes les interactions avec le honeypot, notamment les tentatives de connexion SSH refusées observées lors des tests précédents.

7.2 Lecture brute des logs

Une première analyse peut être réalisée en affichant directement le contenu du fichier de logs.

```
cat /var/log/opencanary.log
```

Cette commande permet de vérifier que les événements sont bien enregistrés après chaque tentative de connexion.

7.3 Lecture structurée avec jq

Les logs étant au format JSON, leur lecture brute peut être difficile.
L'outil `jq` permet d'afficher ces données de manière lisible et structurée.

```
jq . /var/log/opencanary.log
```

Chaque événement est alors présenté sous forme de clés et valeurs clairement identifiables, facilitant l'analyse.

7.4 Types d'informations collectées

Les événements enregistrés par OpenCanary contiennent notamment :

- la date et l'heure de la tentative,
- le service ciblé (SSH ou HTTP),
- l'adresse IP source,
- le port de destination,
- des informations liées à la tentative (exemple : identifiant SSH utilisé).

Ces données permettent de comprendre précisément ce qu'un attaquant tente de faire, même lorsque l'accès est refusé.

7.5 Intérêt de l’analyse des logs

L’analyse des logs constitue un point central du fonctionnement d’un honeypot.
Elle permet de :

- confirmer la détection des tentatives d’accès,
- conserver une trace exploitable dans le temps,
- analyser les comportements observés sans interagir avec l’attaquant.

Cette étape valide le rôle du honeypot comme outil de **détection et d’observation**, et non comme système de défense active.

---

8. Synthèse des tests réalisés

8.1 Récapitulatif des actions effectuées

Les tests réalisés dans ce projet reposent sur un scénario simple mais représentatif :

- une machine Windows joue le rôle de machine attaquante,
- une machine Linux héberge le honeypot OpenCanary,
- des tentatives de connexion sont effectuées vers des services simulés.

Les actions principales ont été :

- tentative de connexion SSH sur un port non standard,
- refus d’accès côté attaquant,
- enregistrement automatique de la tentative dans les logs du honeypot.

8.2 Correspondance tentative / détection

Chaque tentative effectuée depuis la machine Windows génère :

- un retour négatif côté attaquant (*permission denied*),
- une entrée correspondante dans les logs du honeypot.

Cette correspondance permet de valider que :

- le honeypot intercepte correctement les connexions,

- aucune interaction n’est perdue,
- les événements sont tracés de manière fiable.

8.3 Comportement attendu d’un honeypot

Le comportement observé est conforme au fonctionnement attendu d’un honeypot :

- aucune ouverture de session réelle,
- aucune réponse fonctionnelle au service simulé,
- uniquement de la détection et de la journalisation.

Cela confirme que le système joue correctement son rôle d’outil d’observation.

---

9. Apports du projet et compétences mobilisées

9.1 Compétences techniques mises en œuvre

Ce projet personnel a permis de mobiliser plusieurs compétences techniques, notamment :

- compréhension du fonctionnement des services réseau (SSH, HTTP),
- manipulation d’un système Linux en ligne de commande,
- gestion et modification de fichiers de configuration,
- analyse de logs au format JSON,
- compréhension des principes de détection en sécurité informatique.

9.2 Intérêt pédagogique du projet

La mise en place d’un honeypot permet de relier directement :

- la théorie vue en cours (réseau, sécurité),
- à une application concrète et observable.

Ce projet apporte une vision plus réaliste des mécanismes de détection utilisés en cybersécurité, sans complexité excessive.

---

10. Limites du projet

10.1 Limites techniques

Le projet présente volontairement certaines limites :

- environnement limité à un réseau local,
- absence d’exposition directe à Internet,
- analyse manuelle des logs,
- nombre réduit de services simulés.

Ces limites sont assumées et cohérentes avec un projet personnel d’apprentissage.

10.2 Sécurité et précautions

Le honeypot ne contient :

- aucune donnée sensible,
- aucun service réel,
- aucun accès possible au système.

Cela garantit qu’aucune exploitation réelle ne peut être réalisée à partir de ce dispositif.

---

11. Pistes d’amélioration possibles

Plusieurs évolutions pourraient être envisagées :

- activation d’autres services OpenCanary (FTP, SMB, base de données),
- mise en place d’alertes automatiques,
- centralisation des logs vers un outil d’analyse,
- exposition contrôlée sur Internet pour observer des attaques réelles.

Ces améliorations n’ont pas été mises en place afin de conserver un périmètre maîtrisé.

---

12. Conclusion

Ce projet personnel de mise en place d'un honeypot avec OpenCanary m'a permis de comprendre concrètement le rôle d'un honeypot en sécurité informatique.
Il illustre une approche simple mais efficace de la détection d'intrusion, basée sur l'observation et l'analyse des logs.

Ce travail s'inscrit comme un complément pertinent aux projets réalisés dans le cadre du BTS SIO, et démontre un intérêt personnel pour les thématiques liées à la sécurité des systèmes et des réseaux.

URLs utiles: https://github.com/thinkst/opencanary
I found a hacker in my network with this tool | thinkst/opencanary honeypot - YouTube
Qu'est-ce qu'un Honeypot ? Signification, types, avantages et plus | Fortinet