



UNIVERSITÉ MOHAMMED V
- RABAT
ÉCOLE NATIONALE SUPÉRIEURE
D'INFORMATIQUE ET D'ANALYSE DES SYSTÈMES

RAPPORT DE STAGE 2 ÈME ANNÉE
- GÉNIE LOGICIEL -

Automated Hardening of Virtual Machines using Scripts

Réalisé par :
Chaymae BOUAZZA

Encadré par :
M. Zakaria YACOUBI

Remerciements

Je souhaite tout d'abord exprimer ma profonde gratitude à l'entreprise **Atlas Cloud Services** pour m'avoir offert l'opportunité de réaliser ce stage au sein de ses équipes. Cette expérience a constitué pour moi une étape essentielle dans mon parcours, en m'apportant un cadre professionnel riche en apprentissages et en échanges. Je remercie chaleureusement l'ensemble des collaborateurs d'Atlas pour leur accueil, leur disponibilité et l'esprit de partage dont ils ont fait preuve.

J'adresse mes remerciements les plus sincères à mon encadrant, **M. Zakaria Yacoubi**, pour son accompagnement constant, sa patience et ses conseils avisés qui ont grandement contribué à la réussite de ce projet. Son suivi rigoureux, sa disponibilité permanente et sa capacité à transmettre ses connaissances avec clarté et pédagogie ont fait de ce stage une véritable expérience d'apprentissage et de perfectionnement. Sa confiance et son soutien m'ont permis de progresser tant sur le plan technique que personnel.

Je tiens également à remercier **M. Ouissame Damiani**, pour sa confiance, sa vision et son soutien tout au long de cette expérience. Mes remerciements s'adressent aussi à **M. El mahdi FRIMANE** et **M. Ahmed Jadani**, pour leur collaboration précieuse, leurs échanges enrichissants, leur aide tout au long du stage.

Enfin, je voudrais exprimer toute ma reconnaissance à l'**École Nationale Supérieure d'Informatique et d'Analyse des Systèmes (ENSIAS)**, pour la qualité de l'enseignement dispensé et pour l'accompagnement offert aux étudiants. Je remercie l'ensemble du corps professoral pour leur engagement et leur soutien continu, qui ont constitué une base solide pour la réussite de ce stage.

Résumé

Dans le cadre de mon stage chez **Atlas Cloud Services**, j'ai travaillé sur l'automatisation du durcissement des machines virtuelles Linux et Windows à l'aide de scripts et d'outils comme Ansible. L'objectif principal de ce projet était de renforcer la sécurité des VMs en appliquant les bonnes pratiques définies par les standards CIS et ANSSI, tout en assurant un processus reproductible et auditable.

Le projet comprend le développement de scripts modulaires permettant de réaliser un cycle complet *Audit* → *Remédiation* → *Validation*, avec génération de rapports détaillés et conservation des preuves (logs, etc..). Ces scripts automatisent le durcissement des configurations réseau, des pare-feux, des politiques d'authentification et mots de passe, ainsi que la désactivation de services non essentiels, tout en garantissant l'idempotence et la sécurité des modifications appliquées. Une attention particulière a été portée à la **documentation** de chaque script, incluant son fonctionnement, ses paramètres, et les résultats attendus, afin d'assurer une compréhension claire et une maintenance facilitée.

Cette expérience m'a permis de mettre en pratique mes compétences en administration système, scripting Bash et PowerShell, ainsi qu'en automatisation avec Ansible, tout en travaillant sur un projet concret d'entreprise. Elle illustre ma capacité à concevoir, documenter et déployer des solutions automatisées, sécurisées et conformes aux standards internationaux, adaptées à un environnement cloud et virtualisé.

Mots clés : Durcissement, Sécurité, Machines virtuelles, Automatisation, CIS, ANSSI.

Abstract

During my internship at **Atlas Cloud Services**, I worked on automating the hardening of Linux and Windows virtual machines using scripts and tools such as **Ansible**. The main objective of this project was to enhance VM security by applying best practices defined by CIS and ANSSI standards, while ensuring a reproducible and auditable process.

The project involved developing modular scripts that implement a full *Audit → Remediation → Validation* cycle, with detailed reports and evidence collection (logs, diffs, configurations). These scripts automate the hardening of network configurations, firewalls, authentication and password policies, and the disabling of unnecessary services, while guaranteeing idempotence and safety of applied changes. Special attention was given to the **documentation** of each script, describing its functionality, parameters, and expected results, to ensure clarity and facilitate maintenance.

This experience allowed me to apply my skills in system administration, Bash and PowerShell scripting, and automation with Ansible, while working on a real enterprise project. It demonstrates my ability to design, document, and deploy automated, secure solutions compliant with international standards, suitable for cloud and virtualized environments.

Keywords : Hardening, Security, Virtual Machines, Automation, CIS, ANSSI.

Introduction générale

Dans un contexte où la sécurité des infrastructures informatiques et des environnements virtualisés devient un enjeu stratégique pour les entreprises, il est essentiel de mettre en place des solutions automatisées permettant de renforcer la protection des systèmes et des données. Les machines virtuelles, largement utilisées dans les environnements cloud et hybrides, nécessitent des configurations sécurisées conformes aux standards internationaux tels que CIS et ANSSI afin de limiter les risques d'intrusion et de vulnérabilités.

Cependant, la configuration manuelle et le durcissement des VMs peuvent s'avérer fastidieux, sujets à erreurs et difficiles à reproduire sur de multiples machines. Cela peut entraîner des incohérences dans les paramètres de sécurité et compliquer la maintenance et l'audit des systèmes.

C'est dans ce contexte que s'inscrit mon projet de stage, qui a consisté à développer des scripts automatisés pour le durcissement des machines virtuelles Linux et Windows, complétés par des playbooks Ansible. Ces outils permettent d'appliquer de manière systématique et sécurisée les bonnes pratiques, tout en générant des rapports d'audit et en conservant des preuves pour chaque étape (*Audit → Remédiation → Validation*). Une documentation détaillée de chaque script a également été réalisée afin de faciliter la compréhension, la maintenance et l'évolution des solutions mises en place.

L'objectif principal de ce projet est de fournir une solution complète et automatisée pour sécuriser les machines virtuelles, tout en démontrant mes compétences en administration système, scripting, automatisation et sécurité informatique.

Table des matières

| | |
|--|-----------|
| Remerciements | 1 |
| Résumé | 2 |
| Abstract | 3 |
| Introduction générale | 4 |
| 1 Présentation de l'organisme d'accueil | 10 |
| 1.1 Introduction | 11 |
| 1.2 Historique et Vision | 11 |
| 1.3 Mission et Valeurs | 12 |
| 1.4 Infrastructures et Data Center | 12 |
| 1.5 Produits et Services | 13 |
| 1.6 Partenariats Stratégiques | 14 |
| 1.7 Conclusion | 15 |
| 2 Présentation du projet de stage | 16 |
| 2.1 Introduction | 17 |
| 2.2 Contexte général | 17 |
| 2.3 Problématique | 17 |
| 2.4 Objectifs du projet | 18 |
| 2.5 Conclusion | 18 |
| 3 Analyse et Conception | 19 |
| 3.1 Intoduction | 20 |
| 3.2 Analyse des besoins | 20 |

| | | |
|----------|--|-----------|
| 3.2.1 | Besoins fonctionnels | 20 |
| 3.2.2 | Besoins non-fonctionnels | 21 |
| 3.3 | Étude des standards de sécurité | 21 |
| 3.3.1 | Analyse des CIS Benchmarks | 22 |
| 3.3.2 | Analyse des recommandations ANSSI | 22 |
| 3.3.3 | Mapping des standards et priorités | 23 |
| 3.4 | Architecture de la solution | 23 |
| 3.4.1 | Vue d'ensemble architecturale | 23 |
| 3.4.2 | Architecture des scripts | 23 |
| 3.4.3 | Structure détaillée des dossiers Ansible | 26 |
| 3.4.4 | Flux de traitement | 28 |
| 3.5 | Conclusion | 29 |
| 4 | Réalisation | 30 |
| 4.1 | Introduction | 31 |
| 4.2 | Outils utilisés | 31 |
| 4.2.1 | Bash | 31 |
| 4.2.2 | Ansible | 31 |
| 4.2.3 | WSL (Windows Subsystem for Linux) | 32 |
| 4.2.4 | Lynis | 32 |
| 4.2.5 | SMTP / Gmail | 32 |
| 4.2.6 | Autres outils | 33 |
| 4.3 | Structure des scripts et Résultats | 33 |
| 4.3.1 | Phase 1 : Packages, Chrony, TMOUT, umask (CIS 1.2, 2.3, 5.4.3) . | 33 |
| 4.3.2 | Phase 2 : Firewall & Network (CIS 4, 3) | 35 |
| 4.3.3 | Phase 3 : Disable Unused Services (CIS 2 : 2.1, 2.2) | 37 |
| 4.3.4 | Phase 4 : PAM & Password Policies (CIS 5 : 5.3, 5.4.1) | 40 |
| 4.3.5 | Phase 5 : Privilege Escalation (CIS 5.2) | 44 |
| 4.3.6 | Phase 6 : System Maintenance (CIS 7 : 7.1, 7.2) | 44 |
| 4.3.7 | Configuration Réseau avec Netplan | 46 |
| 4.4 | Automatisation avec Crontab | 48 |

| | |
|---|-----------|
| 4.5 Automatisation avec Ansible | 49 |
| 4.5.1 Architecture de la solution | 49 |
| 4.5.2 Workflow d'exécution | 49 |
| 4.5.3 Démonstration pratique | 51 |
| 4.5.4 Bénéfices de l'automatisation | 55 |
| Conclusion Générale | 57 |

Table des figures

| | | |
|------|---|----|
| 1.1 | Logo ACS | 11 |
| 1.2 | Logos de l'OCP et de l'UM6P | 11 |
| 1.3 | Certifié Tier III et Tier IV | 12 |
| 1.4 | Data Center d'Atlas Cloud Services | 13 |
| 1.5 | Partenaires stratégiques d'Atlas Cloud Services | 14 |
| 3.1 | Architecture générale de la solution de durcissement | 23 |
| 3.2 | Architecture générale des scripts de durcissement | 24 |
| 3.3 | Architecture du module Password Policies | 25 |
| 3.4 | Architecture du durcissement des fichiers et des comptes utilisateurs | 25 |
| 3.5 | Architecture Ansible | 27 |
| 3.6 | flux du traitement du durcissement | 29 |
| 4.1 | Logo Bash | 31 |
| 4.2 | Logo Ansible | 31 |
| 4.3 | Logo WSL | 32 |
| 4.4 | Logo Lynis | 32 |
| 4.5 | SMTP via Gmail pour notifications | 32 |
| 4.6 | Résultat de l'exécution automatique des scripts via Crontab | 48 |
| 4.7 | Architecture générale de la solution Ansible | 49 |
| 4.8 | Workflow de durcissement automatisé | 50 |
| 4.9 | Démarrage de l'exécution du playbook Ansible | 51 |
| 4.10 | Traitements de la section Firewall (UFW) | 52 |
| 4.11 | Vérification manuelle de la configuration UFW | 53 |
| 4.12 | Règles de pare-feu configurées automatiquement | 53 |
| 4.13 | Fichiers temporaires créés par Ansible | 54 |

| | |
|---|----|
| 4.14 Exécution de l'audit de sécurité Lynis | 54 |
| 4.15 Rapport de sécurité Lynis reçu par email | 55 |

Chapitre 1

Présentation de l'organisme d'accueil

1.1 Introduction

Dans ce chapitre, nous présentons l'organisme qui a accueilli mon stage **Atlas Cloud Services**, en détaillant son historique, sa mission, son organisation, ses infrastructures, ses produits et services, ainsi que ses partenariats stratégiques. Cette présentation permet de situer le contexte professionnel dans lequel s'inscrit mon projet de durcissement automatisé des machines virtuelles.



FIGURE 1.1 – Logo ACS

1.2 Historique et Vision

Atlas Cloud Services est la concrétisation d'une vision nationale visant à renforcer la souveraineté digitale du pays. Fruit d'un partenariat entre l'OCP, leader mondial de l'industrie du phosphate, et l'Université Mohammed VI Polytechnique (UM6P), Atlas Cloud Services ambitionne de catalyser la transformation digitale de l'écosystème national et régional.



FIGURE 1.2 – Logos de l'OCP et de l'UM6P

1.3 Mission et Valeurs

La mission principale d'Atlas Cloud Services est d'accélérer la transformation digitale des institutions et entreprises marocaines en leur proposant des services Cloud fiables, performants et conformes aux standards internationaux.

Les valeurs fondamentales de l'entreprise sont :

- **Fierté** : contribuer à faire du Maroc un acteur majeur en matière de services Data Center Cloud.
- **Responsabilité** : prioriser la flexibilité, la sécurité, la connectivité et la performance énergétique.
- **Agilité** : rester à l'écoute des besoins des clients et des évolutions technologiques.
- **Ambition** : déployer le leadership d'Atlas sur l'ensemble des offres technologiques hébergées au Maroc.

1.4 Infrastructures et Data Center

Atlas Cloud Services dispose d'un Data Center situé au Tech Park de Benguérir, conçu pour allier sécurité, performance et flexibilité. Certifié Tier III et Tier IV par l'Uptime Institute, il répond aux standards internationaux les plus stricts.



FIGURE 1.3 – Certifié Tier III et Tier IV

Caractéristiques principales :

- Surface totale : 2000 m² (4 data halls de 500 m²)
- PUE : 1.5
- Alimentation : 5 MW IT Load
- Refroidissement : Freecooling avec ventilateurs EC

Ces caractéristiques démontrent l'importance accordée à la performance, à la sécurité et à l'efficacité énergétique du Data Center. La surface généreuse et la répartition en plusieurs halls permettent une flexibilité maximale dans l'organisation des équipements. Le faible PUE reflète l'optimisation énergétique, tandis qu'une alimentation de 5 MW garantit une puissance suffisante pour les charges IT critiques. Le système de refroidissement Freecooling, associé à des ventilateurs EC, assure un maintien optimal de la température tout en minimisant l'impact environnemental.



FIGURE 1.4 – Data Center d'Atlas Cloud Services

1.5 Produits et Services

Atlas Cloud Services propose une gamme complète de services Cloud et Data Center, conçus pour accompagner la transformation numérique des entreprises et institutions marocaines. Parmi les offres disponibles :

- **Hébergement Cloud Souverain** : Solutions d'infrastructure Cloud locales, garantissant la souveraineté des données et la conformité aux réglementations marocaines.
- **Services IaaS, PaaS et SaaS** : Plateformes flexibles pour le déploiement d'applications, le stockage de données et la gestion des ressources informatiques.

- **Solutions de stockage et de sauvegarde sécurisée** : Services adaptés aux besoins de continuité d'activité et de protection des données sensibles.
- **Assistance technique 24/7 et services *Hands & Eyes*** : Support opérationnel disponible en permanence pour assurer la disponibilité et la performance des infrastructures.
- **Interconnexion et connectivité optimisée** : Services tels que *Cross Connect* et *Meet Me Room* pour faciliter les échanges de données et la connectivité entre différents réseaux.

Ces services sont conçus pour offrir aux entreprises marocaines des solutions Cloud performantes, sécurisées et conformes aux exigences locales, contribuant ainsi à la souveraineté numérique du pays.

1.6 Partenariats Stratégiques

Atlas Cloud Services collabore avec des leaders technologiques mondiaux : Red Hat, SAP, IBM, Nutanix, VMware et HCLSoftware.

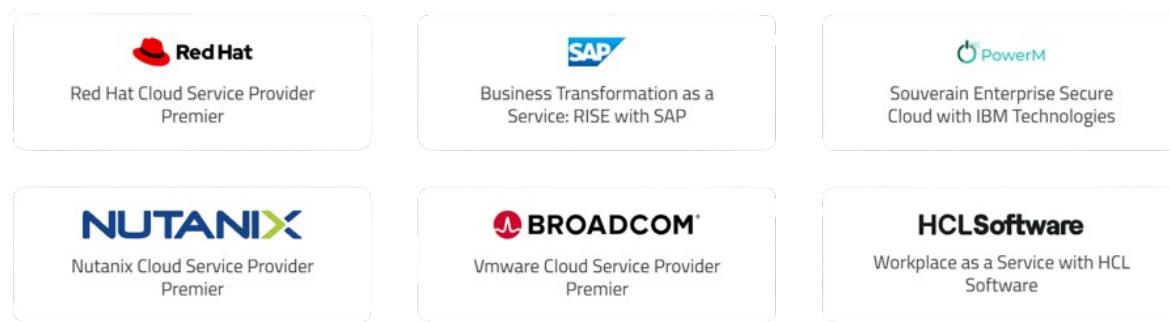


FIGURE 1.5 – Partenaires stratégiques d'Atlas Cloud Services

Ces partenariats permettent à Atlas Cloud Services d'offrir des solutions Cloud fiables, sécurisées et conformes aux standards internationaux, contribuant ainsi à la souveraineté numérique du Maroc et à l'optimisation des services numériques pour les entreprises et administrations.

1.7 Conclusion

Ce chapitre a permis de présenter Atlas Cloud Services, son historique, sa mission, ses valeurs, son infrastructure, ses produits et services ainsi que ses partenariats stratégiques. Cette présentation fournit le cadre organisationnel et technique dans lequel mon projet de durcissement automatisé des machines virtuelles a été réalisé.

Chapitre 2

Présentation du projet de stage

2.1 Introduction

Dans ce chapitre, nous présentons de manière détaillée notre projet de durcissement automatisé des machines virtuelles. Nous commencerons par établir le contexte dans lequel s'inscrit ce projet au sein d'Atlas Cloud Services. Nous identifierons ensuite la problématique spécifique liée aux défis de sécurisation des infrastructures virtualisées dans un environnement multi-cloud. Enfin, nous définirons les objectifs stratégiques et techniques que nous visons à atteindre. Ce chapitre constitue le fondement de notre approche DevSecOps pour l'automatisation de la sécurité.

2.2 Contexte général

Le projet se positionne dans le cadre des activités d'Atlas Cloud Services et répond aux enjeux internes de l'entreprise, notamment en matière de sécurité, de gestion des environnements Cloud, d'automatisation des processus et de standardisation des configurations des machines virtuelles.

Il vise à fournir une solution permettant d'auditer et de durcir les systèmes Windows et Linux de manière reproductible et efficace, contribuant ainsi à l'amélioration continue des infrastructures et des services proposés par Atlas Cloud Services.

2.3 Problématique

Dans un environnement où les systèmes sont de plus en plus complexes et critiques, le respect des bonnes pratiques de sécurité devient un enjeu majeur. Cependant, appliquer manuellement les configurations de durcissement sur chaque machine virtuelle entraîne plusieurs difficultés :

- Risques d'erreurs humaines lors des configurations répétitives,
- Manque d'homogénéité entre les serveurs,
- Perte de temps lors des déploiements et mises à jour,
- Difficultés à garantir la conformité continue aux standards de sécurité (CIS Benchmarks, ANSSI).

Ces limites soulignent la nécessité d'une approche plus efficace et centralisée pour assurer un durcissement fiable et systématique des environnements.

2.4 Objectifs du projet

L'objectif principal de ce projet est de mettre en place une solution de durcissement automatisé des machines virtuelles, intégrée aux environnements VMware et Nutanix, afin d'assurer un haut niveau de sécurité et de conformité. Les objectifs spécifiques sont :

- Étudier et appliquer les bonnes pratiques de durcissement selon les standards (CIS, ANSSI),
- Développer des scripts Bash et PowerShell permettant l'audit et la remédiation des configurations de sécurité,
- Mettre en place une solution d'automatisation centralisée avec **Ansible** pour déployer ces configurations de manière cohérente et reproductible,
- Intégrer la solution dans un processus industrialisé afin de faciliter la gestion et le suivi des environnements durcis.
- Rédiger une documentation claire pour assurer la réutilisabilité des scripts.

2.5 Conclusion

Ce projet s'inscrit dans une démarche DevSecOps visant à intégrer la sécurité dès les phases de conception et de déploiement des infrastructures. En automatisant le processus de durcissement, nous contribuons non seulement à l'amélioration de la posture sécuritaire d'Atlas Cloud Services, mais également à l'optimisation des processus opérationnels. L'approche choisie, combinant développement de scripts spécialisés et orchestration Ansible, permet d'adresser les problématiques identifiées tout en assurant la scalabilité et la maintenabilité de la solution. Ce chapitre établit les fondements conceptuels qui guideront la suite de notre démarche, notamment les phases d'analyse, de conception et d'implémentation qui seront détaillées dans les chapitres suivants.

Chapitre 3

Analyse et Conception

3.1 Introduction

Ce chapitre présente la démarche d'analyse et de conception adoptée pour développer notre solution de durcissement automatisé. Nous détaillerons l'étude des besoins, l'analyse des standards de sécurité, la conception architecturale de la solution, ainsi que les choix technologiques effectués.

3.2 Analyse des besoins

3.2.1 Besoins fonctionnels

Le projet répond à plusieurs besoins fonctionnels essentiels pour assurer la sécurité et l'automatisation des environnements virtuels :

1. **Durcissement automatisé multi-plateforme** : Le système doit permettre le durcissement automatisé des machines sous Linux (Ubuntu, CentOS) et Windows Server. Les configurations de sécurité doivent être appliquées de manière automatique et conforme aux standards reconnus, garantissant ainsi une homogénéité et une fiabilité sur toutes les plateformes.
2. **Conformité aux standards de sécurité** : Le projet doit intégrer les recommandations des CIS Benchmarks et respecter les guidelines ANSSI pour les systèmes français. Il inclut des mécanismes d'audit et de validation afin d'assurer que les machines respectent en permanence les normes de sécurité définies.
3. **Orchestration centralisée** : La solution prévoit une gestion centralisée des déploiements via Ansible, permettant d'exécuter les scripts simultanément sur plusieurs machines. Un inventaire dynamique doit également être géré pour faciliter le suivi et le contrôle des systèmes.
4. **Scripts modulaires et réutilisables** : Les scripts sont conçus de manière modulaire, organisés par domaine de sécurité, et peuvent être réutilisés sur différents environnements. Ils offrent un paramétrage flexible pour s'adapter aux besoins spécifiques de chaque infrastructure.

5. **Création de templates Golden Image :** Le projet inclut la génération d'images de référence durcies selon les standards de sécurité. Ces templates sont réutilisables pour les futurs déploiements et permettent de valider et certifier les images de base, garantissant ainsi une mise en production rapide et sécurisée.

3.2.2 Besoins non-fonctionnels

Le projet répond également à des besoins non-fonctionnels qui garantissent la qualité, la sécurité et la maintenabilité de la solution :

1. **Performance :** Les scripts de durcissement doivent s'exécuter de manière optimisée, avec une capacité à traiter plusieurs machines en parallèle afin de réduire les temps d'exécution globaux.
2. **Fiabilité :** Les scripts doivent être robustes et inclure une gestion d'erreurs efficace. Des mécanismes de rollback doivent être prévus en cas d'échec, et des logs détaillés doivent être générés pour faciliter le debugging.
3. **Maintenabilité :** Le code doit être documenté et structuré selon une architecture modulaire. Il doit permettre une mise à jour facile des règles de sécurité et l'évolution des scripts sans complexité.
4. **Sécurité :** L'exécution des scripts doit se faire dans un environnement sécurisé. Les credentials et accès sensibles doivent être protégés afin de garantir la confidentialité et l'intégrité des systèmes.

3.3 Étude des standards de sécurité

Cette section présente l'analyse des principaux standards de sécurité appliqués dans le cadre du projet de durcissement des systèmes Linux, en se basant sur les recommandations CIS Benchmarks et ANSSI. L'objectif est d'assurer une conformité optimale et de renforcer la sécurité des environnements virtuels automatisés.

3.3.1 Analyse des CIS Benchmarks

Pour les systèmes Linux, les CIS Benchmarks proposent deux niveaux de recommandations :

- **Niveau 1** : Recommandations de base, faible impact sur les performances.
- **Niveau 2** : Recommandations avancées, adaptées aux environnements nécessitant une sécurité élevée.

Les principaux domaines couverts et intégrés dans le projet sont :

- **Initial Setup (Section 1)** : Gestion des packages et synchronisation du temps (CIS 1.2, 2.3) — Phase 1.
- **Services (Section 2)** : Désactivation des services inutiles côté serveur et client (CIS 2.1, 2.2) — Phase 3.
- **Network Configuration (Section 3)** : Sécurisation réseau, configuration firewall et blocage des modules non nécessaires (CIS 3, 4) — Phase 2.
- **Access, Authentication (Section 5)** : Politiques de mot de passe, verrouillage des comptes, privilèges sudo et prévention des escalades (CIS 5.2, 5.3, 5.4.1) — Phases 4 et 5.
- **System Maintenance (Section 6)** : Permissions des fichiers système, cohérence des comptes et groupes, maintenance et mises à jour sécurisées (CIS 7.1, 7.2) — Phase 6.

Chaque domaine a été traité via des **scripts modulaires automatisés**, d'abord en Bash, puis orchestrés avec Ansible pour permettre un déploiement rapide et reproductible sur toutes les machines Linux.

3.3.2 Analyse des recommandations ANSSI

Les recommandations ANSSI pour Linux complètent les CIS Benchmarks en renforçant la sécurité selon les pratiques nationales, notamment pour les administrations françaises. Elles garantissent :

- La conformité légale et réglementaire.
- Le renforcement des configurations critiques pour les systèmes de production.

- L'intégration dans une approche automatisée de durcissement.

3.3.3 Mapping des standards et priorités

Pour clarifier les priorités et l'application des standards dans le projet, le mapping ci-dessous résume les domaines traités pour Linux :

| Domaine | CIS Linux | Priorité |
|----------------------------|--------------------------|----------|
| Gestion des comptes | Section 5 | Critique |
| Configuration réseau | Section 3 | Élevée |
| Services système | Section 2 | Élevée |
| Packages et temps système | Sections 1, 2.3 | Élevée |
| Mot de passe et priviléges | Sections 5.2, 5.3, 5.4.1 | Critique |
| Maintenance système | Sections 6, 7.1, 7.2 | Élevée |

Ce mapping reflète l'ensemble des **phases couvertes dans le projet** et sert de référence pour l'implémentation automatisée et la vérification de la conformité.

3.4 Architecture de la solution

3.4.1 Vue d'ensemble architecturale

La solution de durcissement adoptée est centrée sur l'automatisation pour les systèmes Linux (Ubuntu) et repose sur l'orchestration via Ansible ainsi que des scripts Bash modulaires. L'architecture globale se présente comme suit :

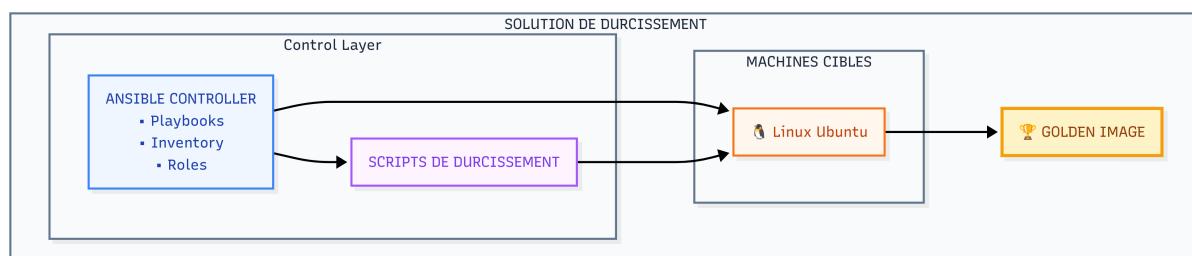


FIGURE 3.1 – Architecture générale de la solution de durcissement

3.4.2 Architecture des scripts

La solution adopte une structure modulaire pour faciliter la maintenance, la réutilisabilité et l'automatisation. Chaque module correspond à un domaine spécifique de sécurité

et contient des scripts d'audit et de remédiation.

Organisation générale des scripts Le diagramme ci-dessous présente la structure principale des scripts de durcissement :



FIGURE 3.2 – Architecture générale des scripts de durcissement

Zoom sur les politiques de mot de passe Pour plus de clarté, la structure interne du module *password-policies* est détaillée dans le diagramme ci-dessous :

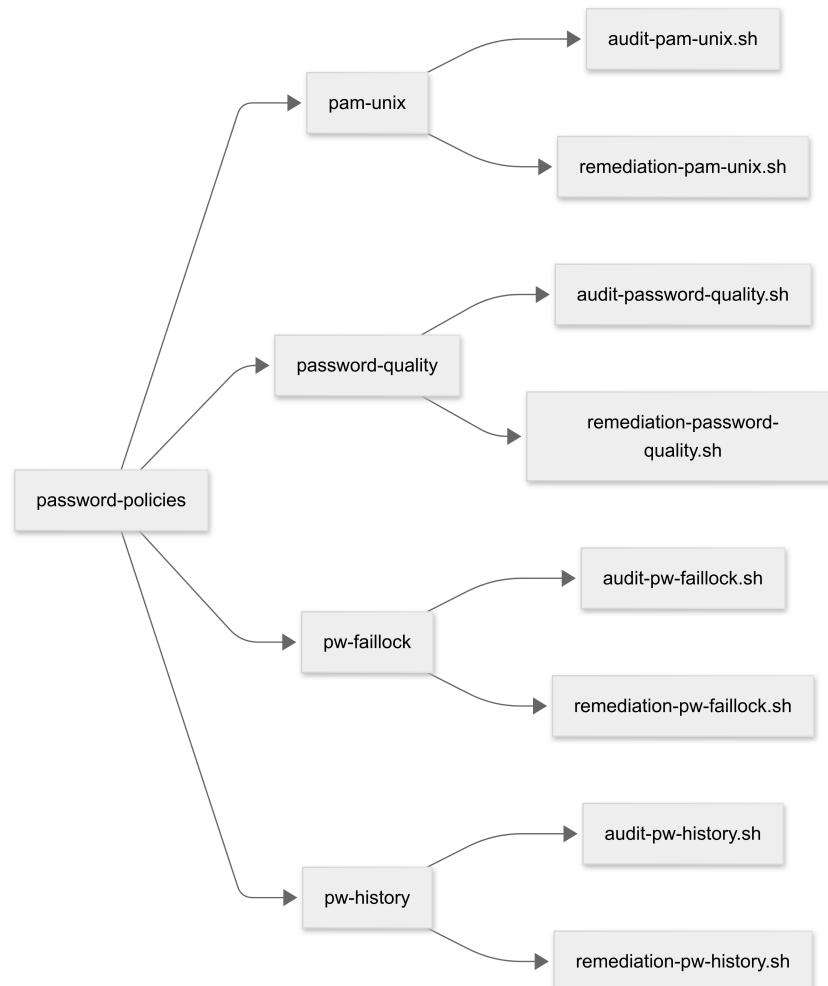


FIGURE 3.3 – Architecture du module Password Policies

Zoom sur le durcissement du système de fichiers Le module *System Maintenance* est illustré dans le diagramme suivant :

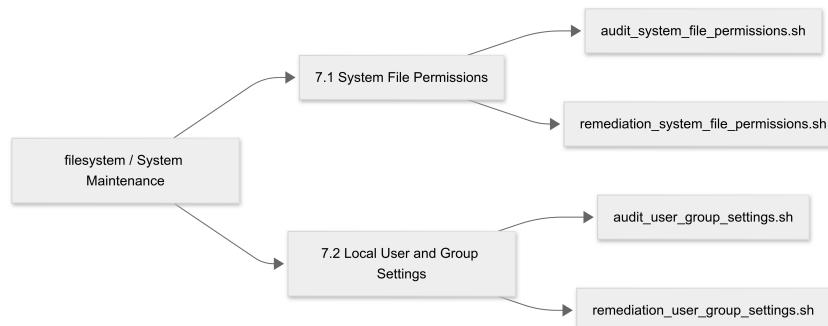


FIGURE 3.4 – Architecture du durcissement des fichiers et des comptes utilisateurs

3.4.3 Structure détaillée des dossiers Ansible

L'organisation des fichiers et dossiers Ansible suit une structure standardisée pour faciliter la maintenance et l'orchestration :

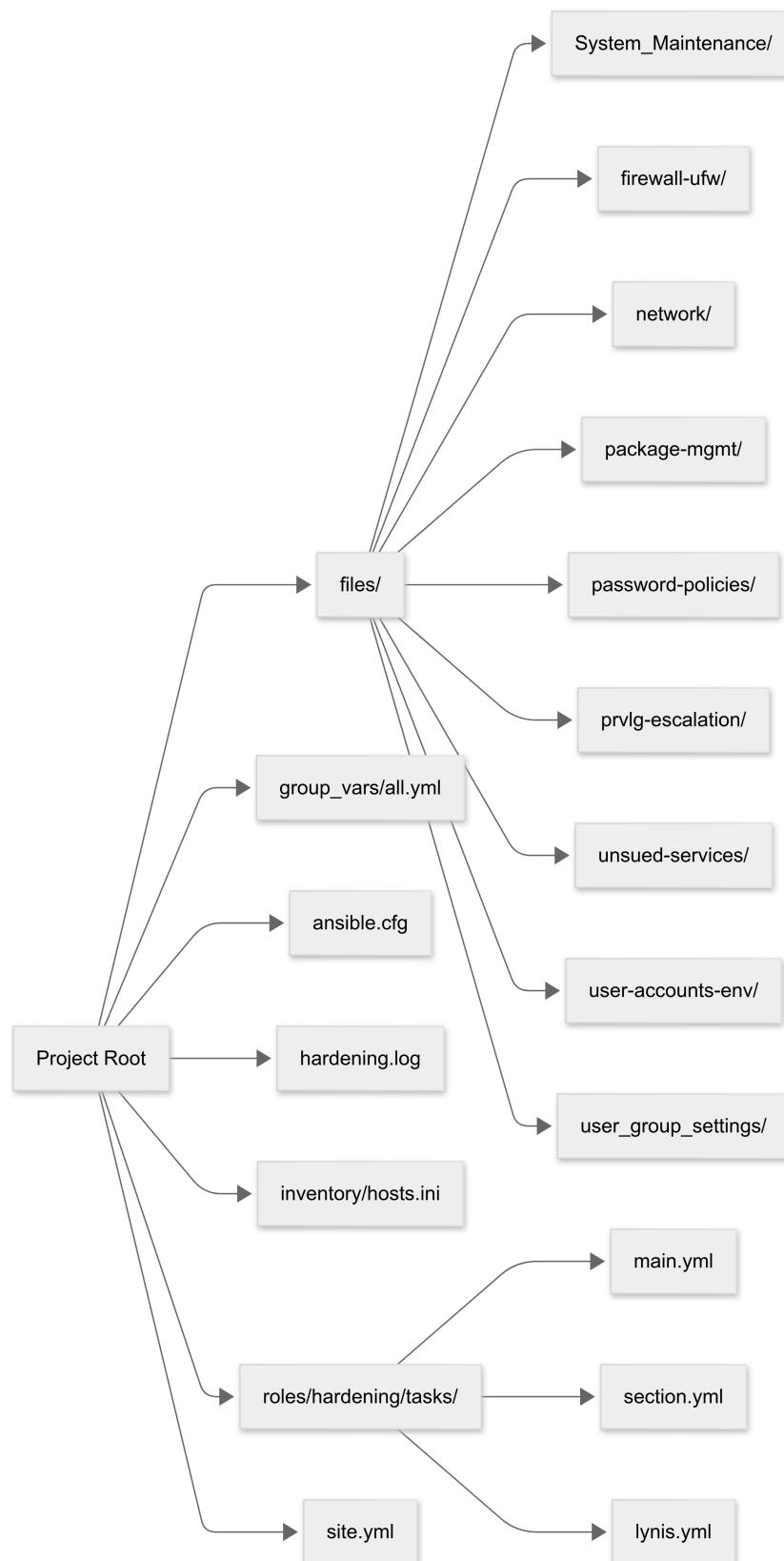


FIGURE 3.5 – Architecture Ansible

Configuration et inventaire

- **ansible.cfg** : Configuration globale d'Ansible (timeouts, logs, privilèges)

- **inventory/hosts.ini** : Définition des groupes de machines et leurs paramètres de connexion
- **group_vars/all.yml** : Variables partagées entre tous les hosts

Orchestration

- **site.yml** : Playbook principal qui orchestre l'exécution de tous les modules de durcissement
- **roles/hardening/** : Rôle Ansible contenant la logique d'orchestration des scripts
Cette architecture modulaire permet une exécution flexible (audit seul, remédiation ciblée) et facilite la maintenance des différents domaines de sécurité.

3.4.4 Flux de traitement

Le processus de durcissement automatisé se déroule en quatre étapes principales :

1. Initialisation

- Lecture de l'inventaire Ansible
- Validation de la connectivité aux machines cibles
- Chargement des variables de configuration

2. Analyse pré-durcissement

- Audit de l'état initial du système
- Identification des configurations à modifier
- Création d'un rapport de baseline

3. Exécution du durcissement

- Application séquentielle des scripts par domaine (utilisateurs, services, réseau, fichiers)
- Vérification après chaque étape
- Gestion des erreurs et rollback si nécessaire

4. Validation post-durcissement

- Audit de conformité final
- Archivage des logs d'exécution



FIGURE 3.6 – flux du traitement du durcissement

La solution reste évolutive et peut intégrer ultérieurement d'autres modules ou une chaîne CI/CD pour la conformité continue.

3.5 Conclusion

Cette phase d'analyse et de conception nous a permis de définir une architecture robuste et modulaire pour notre solution de durcissement automatisé. L'approche choisie, combinant scripts spécialisés et orchestration Ansible, répond aux besoins identifiés tout en respectant les contraintes techniques et de sécurité. La modularité de l'architecture facilite la maintenance et l'évolution de la solution, tandis que l'utilisation d'Ansible comme orchestrateur garantit la reproductibilité et la scalabilité des déploiements. Cette conception constitue la base solide sur laquelle s'appuiera la phase d'implémentation détaillée dans le chapitre suivant.

Chapitre 4

Réalisation

4.1 Introduction

Ce chapitre présente la mise en œuvre pratique de notre solution de durcissement (hardening) d'un système Ubuntu 22.04 LTS conforme aux recommandations CIS. L'implémentation a été réalisée de manière modulaire et automatisée, en utilisant des scripts Bash, la planification via cron, la configuration réseau avec netplan, et une intégration partielle avec Ansible.

4.2 Outils utilisés

4.2.1 Bash



FIGURE 4.1 – Logo Bash

Bash est utilisé pour l'écriture des scripts d'audit et de remédiation automatisés. Tous les scripts de durcissement du système ont été développés en Bash pour garantir une compatibilité maximale avec Ubuntu 22.04 LTS et permettre une exécution facile via cron.

4.2.2 Ansible



FIGURE 4.2 – Logo Ansible

Ansible est utilisé pour automatiser le déploiement et la remédiation des configurations sur plusieurs machines simultanément. Les playbooks Ansible permettent de reproduire

facilement les modifications de sécurité sur un parc de VM.

4.2.3 WSL (Windows Subsystem for Linux)



FIGURE 4.3 – Logo WSL

WSL permet d'exécuter un environnement Linux complet sur une machine Windows. Dans ce projet, il a été utilisé comme nœud de contrôle pour Ansible afin de gérer et orchestrer le durcissement des VM Linux depuis Windows.

4.2.4 Lynis



FIGURE 4.4 – Logo Lynis

Lynis est un scanner de sécurité open-source utilisé pour auditer la sécurité des systèmes Linux et Unix. Il fournit un score de conformité basé sur les recommandations CIS et ANSSI, et permet d'identifier les failles et les configurations non sécurisées.

4.2.5 SMTP / Gmail



FIGURE 4.5 – SMTP via Gmail pour notifications

SMTP est utilisé pour l'envoi d'emails de notifications automatisées après l'exécution des scripts de durcissement. Grâce à l'intégration avec Gmail, les administrateurs reçoivent les rapports de conformité et les scores Lynis en temps réel.

4.2.6 Autres outils

| Outil | Description |
|---------|--|
| Cron | Permet la planification automatique des scripts pour exécuter les audits et remédiations périodiquement. |
| Netplan | Outil de configuration réseau sur Ubuntu 22.04 pour gérer les interfaces, passerelles et DNS. |
| UFW | Gestion simplifiée du firewall pour sécuriser les ports et le trafic réseau. |
| Vim | Éditeurs de texte pour modifier les fichiers de configuration système. |

4.3 Structure des scripts et Résultats

La mise en œuvre des scripts de durcissement a été effectuée de manière automatisée pour respecter les recommandations CIS Ubuntu 22.04 LTS. Pour des raisons de lisibilité, seuls des exemples de scripts sont présentés, tandis que tous les résultats d'audit et de remédiation sont synthétisés dans des tableaux.

4.3.1 Phase 1 : Packages, Chrony, TMOUT, umask (CIS 1.2, 2.3, 5.4.3)

Objectifs

Cette phase vise à assurer que :

- Tous les paquets critiques sont à jour et les mises à jour automatiques configurées.
- Le service Chrony est installé, actif et configuré pour la synchronisation horaire.

- L'environnement par défaut des utilisateurs respecte les politiques TMOUT et umask.

Résultats

Package Management & Time Synchronization

| Contrôle ID | Description du contrôle | Avant l'assainissement | Après remédiation | Résultat |
|-------------|---|---|----------------------------|----------|
| 1.2.2.1 | S'assurer que les mises à jour et les correctifs sont appliqués | ✗ Mises à jour en attente détectées | Le système est à jour | PASS |
| 1.2.2.1 | Mises à niveau non surveillées installées | ✓ Installées | ✓ Installées | PASS |
| 2.3.3 | Chrony installé et configuré | ✗ Chrony inactive / configuration manquante | ✓ Chrony active et activée | PASS |

Default User Environment

| Contrôle ID | Contrôle CIS | Contrôle | Audit avant remédiation | Audit après remédiation | Résultat |
|-------------|--|--------------------------|---|--|----------|
| 5.4.3.1 | S'assurer que nologin n'est pas listé dans /etc/shells | nologin dans /etc/shells | nologin n'est pas listé | nologin n'est pas répertorié | PASS |
| 5.4.3.2 | S'assurer que le délai d'attente par défaut du shell utilisateur (TMOUT) est configuré | Valeur de TMOUT | ✗ TMOUT non configuré | ✓ TMOUT configuré dans <code>/etc/profile.d/99-tmout.sh</code> | PASS |
| 5.4.3.3 | S'assurer que l'umask de l'utilisateur par défaut est configuré | Valeur de l'umask | ✗ umask incorrectement défini dans <code>/etc/login.defs</code> | ✓ umask correctement configuré dans <code>/etc/profile.d/50-systemwide_umask.sh</code> | PASS |

Exemple de script type

Pour illustrer la structure des scripts utilisés dans cette phase, voici un extrait pour l'audit de Chrony :

```
#!/bin/bash

# audit-chrony.sh

# Vérifie que Chrony est installé et actif

if ! command -v chronyd &> /dev/null; then
    echo "[FAIL] Chrony n'est pas installé"
else
    systemctl is-active --quiet chronyd
    if [ $? -eq 0 ]; then
        echo "[PASS] Chrony est actif"
    else
        echo "[WARN] Chrony installé mais inactif"
    fi
fi
```

4.3.2 Phase 2 : Firewall & Network (CIS 4, 3)

Objectifs

- Configurer correctement le pare-feu UFW pour sécuriser les connexions entrantes et sortantes.
- Vérifier l'état des interfaces réseau et désactiver les modules ou services inutiles (IPv6, Bluetooth, etc.).

Résultats

Firwall - UFW

| Contrôle ID | Description CIS | Audit avant remédiation | Audit après la remédiation | Résultat de l'audit |
|-------------|--|--|---|---------------------|
| 4.1.1 | S'assurer que ufw est installé | ufw est installé | ufw est installé | PASS |
| 4.1.2 | Vérifier que iptables-persistent n'est pas installé avec ufw | iptables-persistent n'est pas installé | iptables-persistent n'est pas installé | PASS |
| 4.1.3 | S'assurer que le service ufw est activé | Activé au démarrage : activéActivement actif : inactifStatut : inactif | Activé au démarrage : enabledCurrently active : activeStatus : active | PASS |
| 4.1.4 | S'assurer que le trafic de bouclage ufw est configuré | [WARN] Les règles de bouclage ne sont pas correctement configurées | Les règles de bouclage existent | PASS |
| 4.1.5 | S'assurer que les connexions sortantes ufw sont configurées | [WARN] Les règles de sortie ne sont pas configurées | Des règles de connexion sortante existent | PASS |

Netwrok

| Contrôle ID | Contrôle | Audit avant remédiation | Audit après remédiation | Résultat de l'audit |
|-------------|---|---|---|---------------------|
| 3.2.1 | S'assurer que le module noyau dccp n'est pas disponible | [PASS] non chargé ; [WARN] non bloqué via modprobe ; [WARN] non blacklisté | [PASS] non chargé ; [PASS] bloqué via modprobe ; [PASS] sur liste noire | Remédié |
| 3.2.2 | S'assurer que le module noyau tipc n'est pas disponible | [PASS] non chargé ; [WARN] non bloqué via modprobe ; [WARN] non blacklisté | [PASS] non chargé ; [PASS] bloqué via modprobe ; [PASS] blacklisté | Remédié |
| 3.2.3 | S'assurer que le module noyau rds n'est pas disponible | [PASS] non chargé ; [WARN] non bloqué via modprobe ; [PASS] sur liste noire | [PASS] non chargé ; [PASS] bloqué via modprobe ; [PASS] blacklisté | Remédié |
| 3.2.4 | S'assurer que le module noyau sctp n'est pas disponible | [PASS] non chargé ; [WARN] non bloqué via modprobe ; [WARN] non blacklisté | [PASS] non chargé ; [PASS] bloqué via modprobe ; [PASS] blacklisté | Remédié |

4.3.3 Phase 3 : Disable Unused Services (CIS 2 : 2.1, 2.2)

Objectifs

- Identifier et désactiver tous les services serveur et client inutiles afin de réduire la surface d'attaque.

Résultats

Services - Client

| Contrôle de l'ID | Contrôle | Audit avant remédiation | Audit après remédiation | Résultat de l'audit après remédiation |
|------------------|------------|-------------------------|-------------------------|---------------------------------------|
| 2.2.1 | client nis | ✗ Non installé | ✓ Non installé | PASS |
| 2.2.2 | rsh-client | ✗ Non installé | ✓ Non installé | PASS |
| 2.2.3 | talk | ✗ Non installé | ✓ Non installé | PASS |
| 2.2.4 | telnet | ⚠ Installé | ✓ Non installé | REMEDIATED |
| 2.2.5 | ldap-utils | ✗ Non installé | ✓ Non installé | PASS |
| 2.2.6 | ftp | ⚠ Installé | ✓ Non installé | REMEDIATED |

Services - Serveur

| Contrôle CIS | Contrôle | Audit avant remédiation | Audit après remédiation | Résultat |
|--------------|---|--|---|----------|
| 2.1.1 | S'assurer qu'autofs n'est pas installé | [OK] autofs n'est pas installé | [OK] autofs n'est pas installé | PASS |
| 2.1.2 | S'assurer que avahi-daemon n'est pas installé/désactivé | [WARNING] avahi-daemon est activé et actif | [OK] avahi-daemon n'est pas installé | PASS |
| 2.1.3 | S'assurer que le serveur DHCP n'est pas installé | [OK] idc-dhcp-server n'est pas installé | [OK] idc-dhcp-server n'est pas installé | PASS |
| 2.1.4 | Vérifier que le serveur DNS (bind9) n'est pas installé | [OK] bind9 n'est pas installé | [OK] bind9 n'est pas installé | PASS |
| 2.1.5 | S'assurer que dnsmasq n'est pas installé | [OK] dnsmasq n'est pas installé | [OK] dnsmasq n'est pas installé | PASS |
| 2.1.6 | Vérifier que le serveur FTP (vsftpd) n'est pas installé | [OK] vsftpd n'est pas installé | [OK] vsftpd n'est pas installé | PASS |

| | | | | |
|--------|---|---|---|------|
| 2.1.7 | Vérifier que le serveur LDAP (slapd) n'est pas installé | [OK] slapd n'est pas installé | [OK] slapd n'est pas installé | PASS |
| 2.1.8 | S'assurer que le serveur IMAP/POP3 (dovecot) n'est pas installé | [OK] dovecot-imapd/pop3d n'est pas installé | [OK] dovecot-imapd/pop3d n'est pas installé | PASS |
| 2.1.9 | S'assurer que le serveur NFS n'est pas installé | [OK] nfs-kernel-server n'est pas installé | [OK] nfs-kernel-server n'est pas installé | PASS |
| 2.1.10 | Vérifier que le serveur NIS (ypserv) n'est pas installé. | [OK] ypserv n'est pas installé | [OK] ypserv n'est pas installé | PASS |
| 2.1.11 | Vérifier que CUPS n'est pas installé/désactivé | Cups est activé et actif [WARNING] | [OK] cups n'est pas installé | PASS |
| 2.1.12 | S'assurer que rpcbind n'est pas installé | [OK] rpcbind n'est pas installé | [OK] rpcbind n'est pas installé | PASS |
| 2.1.13 | S'assurer que rsync n'est pas installé/désactivé | [INFO] rsync est installé | [OK] rsync n'est pas installé | PASS |

| | | | | |
|--------|--|------------------------------------|--|------|
| 2.1.14 | S'assurer que Samba n'est pas installé | [OK] samba n'est pas installé | [OK] samba n'est pas installé | PASS |
| 2.1.15 | S'assurer que SNMP n'est pas installé | [OK] snmpd n'est pas installé | [OK] snmpd n'est pas installé | PASS |
| 2.1.16 | Vérifier que le serveur TFTP n'est pas installé | [OK] tftpd-hpa n'est pas installé | [OK] tftpd-hpa n'est pas installé | PASS |
| 2.1.17 | Vérifier que le serveur proxy (squid) n'est pas installé | [OK] squid n'est pas installé | [OK] squid n'est pas installé | PASS |
| 2.1.18 | Vérifier que le serveur web (apache2) n'est pas installé | [OK] apache2 n'est pas installé | [OK] apache2 n'est pas installé | PASS |
| 2.1.19 | S'assurer que xinetd n'est pas installé | [OK] xinetd n'est pas installé | [OK] xinetd n'est pas installé | PASS |
| 2.1.20 | S'assurer que X11 n'est pas installé | [INFO] xserver-common est installé | [OK] xserver-common n'est pas installé | PASS |

4.3.4 Phase 4 : PAM & Password Policies (CIS 5 : 5.3, 5.4.1)

Objectifs

- Appliquer les politiques PAM, qualité des mots de passe, historique et verrouillage des comptes après échec, etc ..

Résultats

pam-unix

| Contrôle ID | Vérifier | Audit avant remédiation | Audit après remédiation | Résultat |
|-------------|--|-------------------------|-------------------------|------------|
| 5.3.3.4.1 | pam_unix utilisé en commun-* | ✓ Utilisé | ✓ Utilisé | PASS |
| 5.3.3.4.1 | présence de l'option nullok | ✗ Présente | ✓ Non présent | REMEDIATED |
| 5.3.3.4.2 | présence de l'option remember | ✓ Non présent | ✓ Non présent | PASSER |
| 5.3.3.4.3 | Hachage de mot de passe fort (sha512/yescrypt) | ✓ Configuré | ✓ Configuré | PASS |
| 5.3.3.4.4 | présence de l'option use_authtok | ✓ Présent | ✓ Présent | PASS |

qualité du mot de passe

| Contrôle ID | Contrôle | Avant la remédiation | Après la remédiation |
|-----------------------------|-------------------------|----------------------|----------------------|
| Qualité du mot de passe PAM | | | |
| 5.3.3.2.7 | pam_pwquality.so activé | ✓ Configuré | ✓ Configuré |
| 5.3.3.2.1 | difok | ✗ Not found | ✓ difok = 2 |
| 5.3.3.2.2 | minlen | ✗ Non trouvé | ✓ minlen = 14 |
| 5.3.3.2.3 | dcredit | ✗ Non trouvé | ✓ dcredit = -1 |
| 5.3.3.2.3 | ucredit | ✗ Non trouvé | ✓ ucredit = -1 |
| 5.3.3.2.3 | lcredit | ✗ Non trouvé | ✓ lcredit = -1 |
| 5.3.3.2.3 | ocredit | ✗ Non trouvé | ✓ ocredit = -1 |
| 5.3.3.2.6 | dictcheck | ✗ Non trouvé | ✓ dictcheck = 1 |

| Politiques de mots de passe fantômes | | | |
|--------------------------------------|---|---|--|
| 5.4.1.1 | PASS_MAX_DAYS dans /etc/login.defs | ✗ Non conforme | <input checked="" type="checkbox"/> 365 |
| 5.4.1.1 | PASS_MAX_DAYS par utilisateur | ✗ Hors limites / manquant | <input checked="" type="checkbox"/> 1-365 appliqué |
| 5.4.1.2 | PASS_MIN_DAYS dans /etc/login.defs | ✗ Non conforme | <input checked="" type="checkbox"/> 1 |
| 5.4.1.2 | PASS_MIN_DAYS par utilisateur | ✗ Hors limites / manquant | <input checked="" type="checkbox"/> ≥1 appliqué |
| 5.4.1.3 | PASS_WARN_AGE dans /etc/login.defs | ✗ Non conforme | <input checked="" type="checkbox"/> 7 |
| 5.4.1.3 | PASS_WARN_AGE par utilisateur | ✗ Hors limites / manquant | <input checked="" type="checkbox"/> ≥7 appliqué |
| 5.4.1.4 | ENCRYPT_METHOD | ✗ Non conforme | <input checked="" type="checkbox"/> YESCRYPT |
| 5.4.1.5 | Valeur par défaut INACTIVE pour les nouveaux utilisateurs | ✗ Non conforme | <input checked="" type="checkbox"/> 45 |
| 5.4.1.5 | INACTIF par utilisateur | ✗ Hors limites / manquant | <input checked="" type="checkbox"/> 0-45 appliqué |
| 5.4.1.6 | Utilisateurs avec dates futures de changement de mot de passe | PAS d'utilisateurs avec des dates futures | <input checked="" type="checkbox"/> Aucun utilisateur avec des dates futures |

pam-faillock

| ID de contrôle | Description de la vérification | Avant la remédiation | Après remédiation | Notes / Détails de la configuration |
|----------------|--|----------------------------------|---------------------------------|---|
| 5.3.3.1.1 | S'assurer que le paramètre <code>deny</code> est défini dans le fichier <code>/etc/security/faillock.conf</code> | ✗ pam_faillock non configuré | deny = 5 (valide ≤5) | Maximum de 5 tentatives de connexion échouées avant le verrouillage |
| 5.3.3.1.1 | Assurer le <code>refus</code> dans <code>/etc/pam.d/common-auth</code> | ✗ pam_faillock non configuré | Pas de valeur de refus invalide | Pas de valeur de deny invalide codée en dur |
| 5.3.3.1.2 | S'assurer que <code>unlock_time</code> est configuré | pam_faillock n'est pas configuré | ✓ unlock_time = 900 (valide) | Temps de verrouillage de 15 minutes, non codé en dur dans les profils PAM |
| 5.3.3.1.3 | Vérifier les paramètres de verrouillage de la racine (<code>even_deny_root / root_unlock_time</code>). | ✗ pam_faillock non configuré | ✓ root_unlock_time = 900 | Verrouillage de la racine appliqué, >=60 secondes, non codé en dur dans les profils PAM |

pam-history

| Contrôle ID | Vérifier | Avant la remédiation | Après remédiation | Statut |
|-------------|---|----------------------|-------------------|---------|
| 5.3.3.3.1 | pam_pwhistory.so dans <code>/etc/pam.d/common-password</code> | Manquant | Présent | Corrigé |
| 5.3.3.3.1 | valeur de <code>rappel</code> ≥ 5 | Manquant | 24 | Fixe |
| 5.3.3.3.2 | option <code>enforce_for_root</code> | Manquante | Présent | Corrigé |
| 5.3.3.3.4 | option <code>use_authok</code> | Manquant | Présente | Corrigé |

4.3.5 Phase 5 : Privilege Escalation (CIS 5.2)

Objectifs

- Assurer la bonne configuration de sudo, journalisation, authentification et restrictions su.

Résultat

| Contrôle ID | Vérification Description | Avant la remédiation | Après la remédiation | Statut |
|-------------|--|---|---|--------|
| 5.2.1 | S'assurer que sudo est installé | sudo est installé | sudo est installé | PASSER |
| 5.2.2 | S'assurer que les commandes sudo utilisent PTY | Les valeurs par défaut use_pty ne sont pas manquantes | /etc/sudoers : Valeurs par défaut use_pty | PASS |
| 5.2.3 | S'assurer que le fichier journal sudo existe | Le fichier journal sudo n'est PAS configuré | /etc/sudoers : Par défaut logfile="/var/log/sudo.log" | PASSER |
| 5.2.4 | S'assurer que les utilisateurs doivent fournir un mot de passe pour sudo | Toutes les commandes sudo requièrent un mot de passe | Toutes les commandes sudo nécessitent un mot de passe | PASS |
| 5.2.5 | S'assurer que la réauthentification n'est pas désactivée globalement | La réauthentification est obligatoire | La réauthentification est obligatoire | PASS |
| 5.2.6 | S'assurer que le délai d'authentification sudo est configuré | le délai d'attente (timestamp_timeout) n'est pas explicitement défini | /etc/sudoers : Par défaut timestamp_timeout=15 | PASS |

4.3.6 Phase 6 : System Maintenance (CIS 7 : 7.1, 7.2)

Objectifs

- Vérifier les permissions des fichiers système critiques.

- Contrôler l'existence et l'exactitude des comptes et groupes système.

Résultats

autorisations fichier système

| ID de contrôle | Description du contrôle | Résultat de l'audit | Notes / Détails de la configuration |
|----------------|--|--|---|
| 7.1.1 | S'assurer que /etc/passwd a les bonnes permissions. | PASSER | 644, propriétaire : root, groupe : root |
| 7.1.2 | Vérifier que /etc/passwd- a les bonnes permissions. | <input checked="" type="checkbox"/> PASS | 644, propriétaire : root, groupe : root |
| 7.1.3 | Vérifier que /etc/group a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 644, propriétaire : root, groupe : root |
| 7.1.4 | Vérifier que /etc/group- a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 644, propriétaire : root, groupe : root |
| 7.1.5 | Vérifier que /etc/shadow a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 640, propriétaire : root, groupe : shadow |
| 7.1.6 | Vérifier que /etc/shadow- a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 640, propriétaire : root, groupe : shadow |
| 7.1.7 | S'assurer que /etc/gshadow a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 640, propriétaire : root, groupe : shadow |
| 7.1.8 | Vérifier que /etc/gshadow- a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 640, propriétaire : root, groupe : shadow |
| 7.1.9 | Vérifier que /etc/shells a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 644, propriétaire : root, groupe : root |
| 7.1.10 | S'assurer que /etc/security/opasswd a les bonnes permissions | <input checked="" type="checkbox"/> PASS | 600, propriétaire : root, groupe : root |

réglages groupe utilisateur

| Contrôle ID | Contrôle Description | Résultat de l'audit | Notes / Détails de la configuration |
|-------------|--|--|---|
| 7.2.1 | S'assurer que les comptes dans /etc/passwd utilisent des mots de passe masqués | <input checked="" type="checkbox"/> PASS | Tous les comptes utilisent déjà des mots de passe masqués |
| 7.2.2 | S'assurer que les champs de mots de passe du fichier /etc/shadow ne sont pas vides | <input checked="" type="checkbox"/> PASS | Aucun champ de mot de passe vide n'a été trouvé |
| 7.2.3 | Vérifier que tous les groupes dans /etc/passwd existent dans /etc/group | <input checked="" type="checkbox"/> PASS | Tous les GID référencés dans <code>/etc/passwd</code> existent dans <code>/etc/group</code> . |
| 7.2.4 | S'assurer que le groupe fantôme est vide | <input checked="" type="checkbox"/> PASS | Le groupe fantôme n'a pas de membres. |
| 7.2.5 | S'assurer qu'il n'y a pas d'UID en double | <input checked="" type="checkbox"/> PASS | Aucun UID en double n'a été trouvé |
| 7.2.6 | S'assurer qu'il n'y a pas de GID en double | <input checked="" type="checkbox"/> PASS | Aucun GID en double n'a été trouvé |
| 7.2.7 | S'assurer qu'il n'y a pas de noms d'utilisateur en double | <input checked="" type="checkbox"/> PASS | Aucun nom d'utilisateur en double n'a été trouvé |
| 7.2.8 | S'assurer qu'il n'y a pas de noms de groupe en double | <input checked="" type="checkbox"/> PASS | Aucun nom de groupe en double n'a été trouvé |

4.3.7 Configuration Réseau avec Netplan

Objectif

Cette partie illustre la configuration réseau automatique de l'interface via un script Netplan. L'objectif est de définir l'adresse IP statique, la passerelle et les serveurs DNS pour l'interface réseau.

Script Netplan

Résultat attendu

- L'interface réseau est correctement configurée avec IP, passerelle et DNS.
- La connectivité réseau est testable via des pings ou des services internes.

```
#!/bin/bash

if [ "$#" -ne 5 ]; then
    echo "Usage: $0 <interface> <ip_address/CIDR> <gateway> <dns1> <dns2>"
    exit 1
fi

INTERFACE="$1"
IPADDR="$2"
GATEWAY="$3"
DNS1="$4"
DNS2="$5"

cat <<EOF | sudo tee /etc/netplan/01-custom.yaml > /dev/null
network:
    version: 2
    renderer: networkd
    ethernets:
        $INTERFACE:
            addresses:
                - $IPADDR
            routes:
                - to: 0.0.0.0/0
                  via: $GATEWAY
            nameservers:
                addresses: [$DNS1, $DNS2]
EOF

echo "[INFO] Applying Netplan configuration..."
sudo netplan apply
```

4.4 Automatisation avec Crontab

Pour automatiser l'exécution des scripts de durcissement, nous avons utilisé Crontab en conjonction avec le script principal `start-hardening.sh`.

Script principal Le script `start-hardening.sh` permet de lancer l'ensemble des modules de durcissement selon deux modes : `audit` et `remediate`.

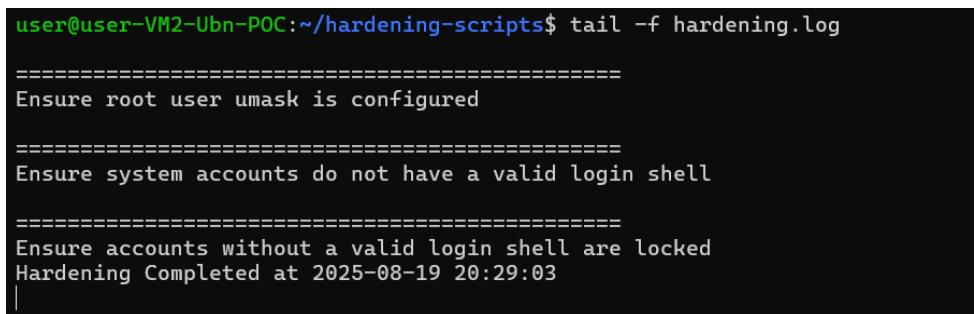
- Le mode `audit` effectue une vérification de l'état du système et des configurations selon les bonnes pratiques CIS. - Le mode `remediate` applique les corrections nécessaires pour rendre le système conforme aux recommandations.

Le script centralise la journalisation des erreurs dans `/var/log/hardening_errors.log` et affiche les heures de début et de fin d'exécution. Cette organisation modulaire facilite la maintenance et l'ajout de nouveaux modules si nécessaire.

Exemple de Crontab

```
# Exécution du script principal en mode audit toutes les 2 minutes
*/2 * * * * /home/user/hardening-scripts/start-hardening.sh --mode audit
```

Le screenshot ci-dessous montre que les scripts de durcissement s'exécutent automatiquement selon le calendrier défini dans Crontab.



```
user@user-VM2-Ubn-POC:~/hardening-scripts$ tail -f hardening.log
=====
Ensure root user umask is configured
=====
Ensure system accounts do not have a valid login shell
=====
Ensure accounts without a valid login shell are locked
Hardening Completed at 2025-08-19 20:29:03
|
```

FIGURE 4.6 – Résultat de l'exécution automatique des scripts via Crontab

4.5 Automatisation avec Ansible

4.5.1 Architecture de la solution

L'automatisation du processus de durcissement a été réalisée avec Ansible, permettant une orchestration centralisée des scripts de sécurité sur l'ensemble des machines virtuelles cibles.

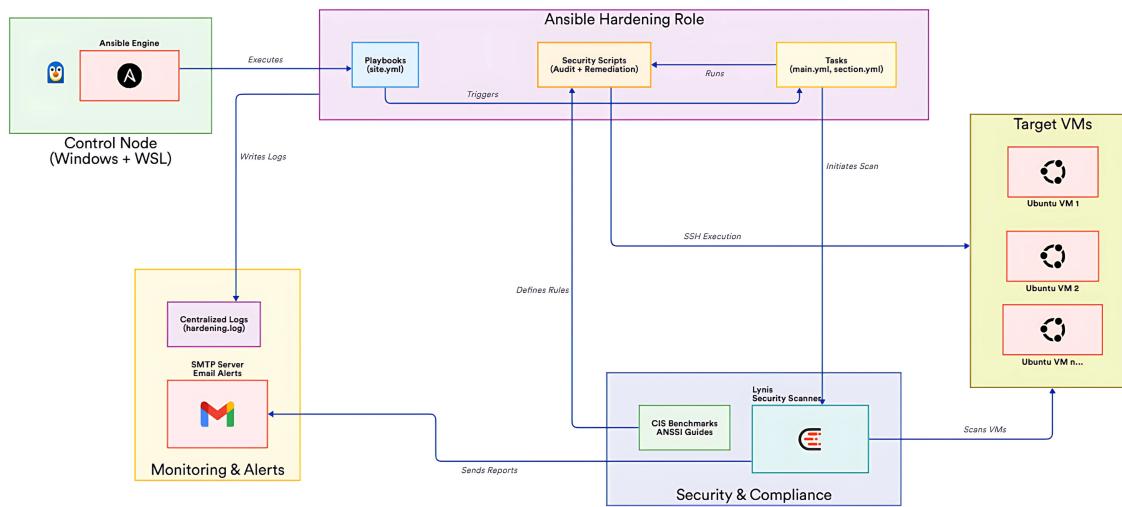


FIGURE 4.7 – Architecture générale de la solution Ansible

Composants principaux

- **Nœud de contrôle** : Machine Windows avec WSL hébergeant Ansible
- **Rôle de durcissement** : Ensemble de tâches orchestrant l'exécution des scripts
- **Scripts de sécurité** : Scripts modulaires d'audit et de remédiation
- **Machines cibles** : VMs Ubuntu à durcir
- **Outils de compliance** : Lynis pour l'évaluation de sécurité
- **Monitoring** : Logs centralisés et notifications SMTP

4.5.2 Workflow d'exécution

Le processus d'automatisation suit un workflow structuré garantissant la traçabilité et la fiabilité des opérations.

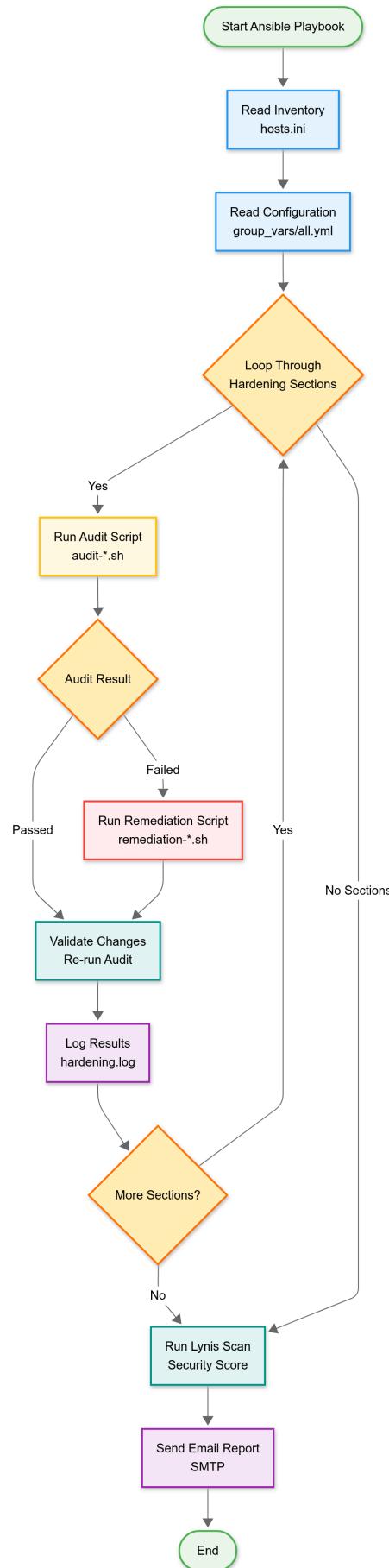


FIGURE 4.8 – Workflow de durcissement automatisé

Cycle d'exécution des scripts

Pour chaque section de durcissement, Ansible suit un cycle rigoureux :

1. **Audit** : Vérification de l'état actuel de sécurité
2. **Évaluation** : Analyse des résultats d'audit
3. **Remédiation** : Application des corrections si nécessaire
4. **Validation** : Re-audit pour confirmer l'application des mesures
5. **Logging** : Enregistrement des résultats dans les logs centralisés

4.5.3 Démonstration pratique

Exécution du playbook

L'exécution d'Ansible montre le traitement séquentiel des différentes sections de durcissement :

```
chay@DESKTOP-PHIAR0M:~/hardening_ansible$ ansible-playbook site.yml
PLAY [CIS Hardening] ****
TASK [Gathering Facts] ****
ok: [server1]
TASK [hardening : Pick the hardening checks] ****
ok: [server1]
TASK [hardening : Run each section] ****
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=firwall_ufw)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=package_chrony)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=package_general)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=pam_unix)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=pw_quality)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=pw_faillock)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=pw_history)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=prvlg_escalation)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=sys_file_permissions)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=sys_user_group)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=unused_client_services)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=unused_server_services)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=default_env)
included: /home/chay/hardening_ansible/roles/hardening/tasks/section.yml for server1 => (item=root_system_accounts)

TASK [hardening : Compute audit and remediation script paths] ****
ok: [server1]
TASK [hardening : [AUDIT] Firewall (UFW)] ****
ok: [server1]
```

FIGURE 4.9 – Démarrage de l'exécution du playbook Ansible

La figure 4.9 illustre le démarrage du playbook avec :

- Collecte des informations système (Gathering Facts)
- Sélection des vérifications de durcissement
- Traitement de chaque section (firewall_ufw, package_general, etc.)

Traitement d'une section de durcissement

L'exemple du pare-feu UFW démontre le cycle complet audit-remédiation-validation :

```

TASK [hardening : Show audit outcome] *****
ok: [server1] => {
  "msg": [
    "Section: firewall_ufw - Firewall (UFW)",
    "RC: 0",
    "Non-compliant? False"
  ]
}

TASK [hardening : [REMEDiate] Firewall (UFW)] *****
skipping: [server1]

TASK [hardening : [VALIDATE] Firewall (UFW)] *****
ok: [server1]

TASK [hardening : Final Compliance after remediation] *****
ok: [server1]

TASK [hardening : Print final result] *****
ok: [server1] => {
  "msg": [
    "After remediation - non-compliant? False"
  ]
}

TASK [hardening : Compute audit and remediation script paths] *****
ok: [server1]

TASK [hardening : [AUDIT] Password - pwquality] *****
ok: [server1]

TASK [hardening : Determine non-compliance] *****
ok: [server1]

```

FIGURE 4.10 – Traitement de la section Firewall (UFW)

Dans cet exemple (Figure 4.10) :

- **Audit initial** : Détection de non-conformité (RC : 0, Non-compliant : False)
- **Remédiation** : Exécution automatique du script de correction
- **Validation** : Confirmation de la conformité après remédiation

Vérification manuelle des changements

Les scripts exécutés modifient effectivement la configuration système :

```
Last login: Fri Aug 29 23:17:37 2025 from 10.212.134.200
user@user-VM2-Ubn-POC:~$ sudo ufw disable
[sudo] password for user:
Firewall stopped and disabled on system startup
user@user-VM2-Ubn-POC:~$ sudo ufw status
Status: inactive
user@user-VM2-Ubn-POC:~$ |
```

FIGURE 4.11 – Vérification manuelle de la configuration UFW

4.1.5 Configure outbound/inbound connections

```
user@user-VM2-Ubn-POC:~/hardening-scripts/Firewall-ufw$ vim remediation-ufw.sh
user@user-VM2-Ubn-POC:~/hardening-scripts/Firewall-ufw$ sudo ufw disable
Firewall stopped and disabled on system startup
user@user-VM2-Ubn-POC:~/hardening-scripts/Firewall-ufw$ ufw status
ERROR: You need to be root to run this script
user@user-VM2-Ubn-POC:~/hardening-scripts/Firewall-ufw$ sudo ufw status
Status: inactive
user@user-VM2-Ubn-POC:~/hardening-scripts/Firewall-ufw$ sudo ufw status
Status: active

To                         Action      From
--                         --          --
22                         ALLOW       Anywhere
Anywhere on lo              ALLOW       Anywhere
Anywhere                   DENY        127.0.0.0/8
80                         ALLOW       Anywhere
443                        ALLOW       Anywhere
306                        ALLOW       Anywhere
22/tcp                      ALLOW       Anywhere
22 (v6)                    ALLOW       Anywhere (v6)
Anywhere (v6) on lo          ALLOW       Anywhere (v6)
Anywhere (v6)               DENY        ::1
80 (v6)                    ALLOW       Anywhere (v6)
443 (v6)                   ALLOW       Anywhere (v6)
306 (v6)                   ALLOW       Anywhere (v6)
22/tcp (v6)                ALLOW       Anywhere (v6)
```

FIGURE 4.12 – Règles de pare-feu configurées automatiquement

Les figures 4.11 et 4.12 confirment que :

- Le pare-feu UFW est désormais actif
- Les règles de sécurité ont été appliquées conformément aux standards CIS
- La configuration persiste après l'exécution d'Ansible

Gestion temporaire des scripts

Ansible assure une gestion propre des scripts sur les machines cibles :

```
user@user-VM2-Ubn-POC:/tmp$ ls
ansible_ansible.legacy.command_payload_a6d4a5lv
ansible_ansible.legacy.command_payload_b5don6wr
ansible_ansible.legacy.command_payload_j6vvtde3
ansible_ansible.legacy.command_payload_x_qgz1h1
prvlg_escalation_audit.sh
prvlg_escalation_remediate.sh
snap-private-tmp
systemd-private-089fe0f0bf894962bf2e3897f0a84104-chrony.service-IQrRIX
systemd-private-089fe0f0bf894962bf2e3897f0a84104-fwupd.service-5902N3
systemd-private-089fe0f0bf894962bf2e3897f0a84104-ModemManager.service-Gizc4n
systemd-private-089fe0f0bf894962bf2e3897f0a84104-power-profiles-daemon.service-tLT9Vp
systemd-private-089fe0f0bf894962bf2e3897f0a84104-switcheroo-control.service-RmI8BV
systemd-private-089fe0f0bf894962bf2e3897f0a84104-systemd-logind.service-9DT1GR
systemd-private-089fe0f0bf894962bf2e3897f0a84104-systemd-oomd.service-uAzB2b
systemd-private-089fe0f0bf894962bf2e3897f0a84104-systemd-resolved.service-mtsdDJ
systemd-private-089fe0f0bf894962bf2e3897f0a84104-upower.service-GKimoI
VMwareDnD
vmware-root_790-2965972456
```

FIGURE 4.13 – Fichiers temporaires créés par Ansible

La figure 4.13 montre les fichiers temporaires créés dans `/tmp/` pendant l'exécution, incluant :

- Les scripts d'audit et de remédiation transférés
- Les fichiers de payload Ansible temporaires
- Une gestion automatique du nettoyage après exécution

Évaluation de sécurité avec Lynis

Après le durcissement, Lynis évalue le niveau de sécurité global :

```
TASK [Ensure Lynis is installed] ****
*****
changed: [server1]

TASK [Run Lynis audit] ****
*****
ok: [server1]

TASK [Extract hardening index] ****
*****
ok: [server1]

TASK [Email Lynis score + report] ****
*****
[
```

FIGURE 4.14 – Exécution de l'audit de sécurité Lynis

Notification automatique des résultats

Le système envoie automatiquement un rapport de sécurité par email :

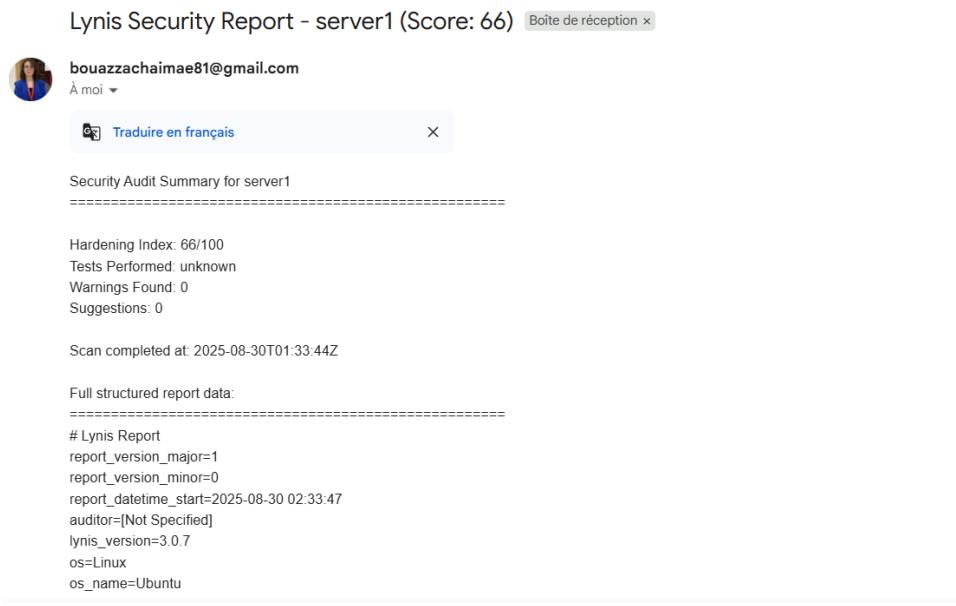


FIGURE 4.15 – Rapport de sécurité Lynis reçu par email

Le rapport email (Figure 4.15) contient :

- **Score de durcissement** : 66/100 pour server1
- **Métriques de sécurité** : Nombre de tests, avertissements, suggestions
- **Horodatage** : Traçabilité temporelle de l'audit
- **Données structurées** : Informations détaillées pour analyse

4.5.4 Bénéfices de l'automatisation

Centralisation et scalabilité

L'intégration Ansible transforme les scripts individuels en une plateforme centralisée capable de traiter simultanément plusieurs machines virtuelles, éliminant les interventions manuelles répétitives.

Traçabilité et auditabilité

Chaque exécution génère des logs détaillés et des rapports de compliance, facilitant les audits de sécurité et la démonstration de conformité aux standards.

Proactivité

Les notifications automatiques par email permettent une réaction rapide aux problèmes de sécurité, réduisant la fenêtre d'exposition aux vulnérabilités.

Conclusion

Dans ce chapitre, nous avons présenté la mise en place et l'orchestration de l'automatisation des configurations de sécurité à l'aide d'Ansible. L'approche adoptée a permis d'exécuter de manière centralisée l'ensemble des scripts d'audit et de remédiation, garantissant ainsi une homogénéité dans l'application des politiques de sécurité sur les machines cibles.

L'intégration d'outils de vérification, tels que **Lynis**, a permis de mesurer l'efficacité des configurations appliquées. Les rapports générés et transmis par mail, contenant notamment le score de sécurité obtenu, constituent une preuve tangible de l'amélioration du niveau de conformité et facilitent le suivi des évolutions dans le temps.

Cette orchestration par Ansible, offre donc une solution fiable, reproductible et maintenable pour le durcissement automatisé des systèmes Linux et Windows, en accord avec les recommandations des benchmarks CIS et de l'ANSSI. Elle constitue une étape clé dans la sécurisation des infrastructures cloud et virtualisées.

Conclusion Générale

Dans le cadre de notre projet, nous avons visé à développer une solution automatisée pour le durcissement et la sécurisation des systèmes Ubuntu, en conformité avec les standards CIS. L'objectif principal était de concevoir des scripts modulaires permettant d'auditer et de remédier efficacement les configurations système, la gestion des utilisateurs, les politiques de mot de passe, la sécurité réseau et le contrôle des services inutiles.

Pour atteindre cet objectif, nous avons mis en place une architecture de scripts organisée par phases et modules, facilitant la maintenance, la réutilisabilité et l'automatisation complète des tâches de sécurité. Nous avons également intégré un mécanisme de planification via **cron** et une configuration réseau automatisée via **netplan**, ce qui permet une gestion cohérente et reproductible des systèmes.

Par ailleurs, l'utilisation d'**Ansible** nous a permis de centraliser le déploiement et l'exécution des scripts sur plusieurs machines cibles, garantissant la cohérence de l'application des politiques de sécurité à grande échelle.

Cette expérience nous a permis de renforcer nos compétences en administration système, en scripting Bash et en automatisation avec Ansible, tout en appliquant concrètement les bonnes pratiques de sécurité. Le projet pose les bases pour un environnement plus sécurisé et facilement maintenable, tout en ouvrant la possibilité d'étendre l'automatisation à d'autres distributions Linux et à des environnements cloud à l'avenir.

Nous sommes fiers du résultat final, qui contribue non seulement à renforcer la sécurité et la conformité des systèmes, mais aussi à simplifier la gestion et la maintenance des environnements informatiques.

Bibliographie

- [1] Center for Internet Security (CIS). *CIS Benchmarks : System Security Configuration Guides*. Disponible sur : <https://www.cisecurity.org/cis-benchmarks/>
- [2] Kyle Rankin. *Linux Hardening in Hostile Networks : Server Security from TLS to Tor*. No Starch Press, 2017. Disponible sur : <https://nostarch.com/linuxhardening>
- [3] Red Hat. *Ansible Documentation : Automation for DevOps*. Disponible sur : <https://docs.ansible.com/ansible/latest/index.html>
- [4] Canonical Ltd. *Ubuntu Server Guide : Security and Hardening*. Disponible sur : <https://ubuntu.com/server/docs/security-hardening>
- [5] Linux-PAM Project. *Pluggable Authentication Modules (PAM) Documentation*. Disponible sur : <http://www.linux-pam.org/>
- [6] Jeff Geerling. *Ansible for DevOps : Server and Configuration Management for Humans*. Leanpub, 2021. Disponible sur : <https://www.ansiblefordevops.com/>
- [7] Daniel J. Barrett, Richard E. Silverman, Robert G. Byrnes. *Linux Security Cookbook*. O'Reilly Media, 2003.
- [8] Debian Project. *CronHowto : Using Cron for Scheduled Tasks*. Disponible sur : <https://wiki.debian.org/CronHowto>
- [9] GeeksforGeeks. *Crontab in Linux with Examples*. Disponible sur : <https://www.geeksforgeeks.org/linux-unix/crontab-in-linux-with-examples/>