Institut Universitaire de Technologie
Aix-Marseille Université

RAPPORT DE STAGE

de fin de deuxième année
Bachelor Universitaire de Technologie
Spécialité Réseaux et Télécommunications
parcours cybersécurité

Admin DevOps Ansible : Durcissement Infra sur Debian 12 à partir des recommandations CIS

Tony KALAIJI

Edenred France

Responsable entreprise : Guillaume BORY

Responsable académique : Ivan MADJAROV

2024

Table des matières

1 Introduction

Durant mon stage au sein de l'entreprise Edenred France, au service Cloud dirigé par Guillaume Bory, j'ai eu l'opportunité de travailler sur un projet crucial de sécurisation et de durcissement des systèmes informatiques. Edenred, une entreprise de renommée internationale spécialisée dans les solutions de paiement, accorde une grande priorité à la sécurité et à la conformité de ses systèmes. Ce projet consistait principalement à renforcer la sécurité de l'infrastructure en durcissant les configurations des serveurs à partir des recommandations du Center for Internet Security (CIS).

L'objectif de mon stage était de mettre en place des configurations de sécurité robustes pour tendre vers la conformité avec les normes de sécurité du CIS, tout en contribuant au déploiement automatisé à grande échelle.

Mes missions incluaient l'implémentation de scripts Ansible pour durcir les systèmes Debian 12, la gestion des tickets sur Jira, la réalisation de tests de sécurité avec GitLab CI/CD, et la documentation des procédures de sécurité. J'ai également participé aux réunions d'équipe pour évaluer et valider les changements apportés à l'infrastructure.

Ce rapport de stage est structuré en plusieurs sections pour offrir une vue complète de mon expérience et des travaux réalisés. Il commence par une introduction qui présente le contexte et les objectifs du stage, suivie d'une présentation de l'entreprise et du service au sein duquel j'ai travaillé. Ensuite, le rapport décrit le cadre technique du projet, les objectifs spécifiques et le cahier des charges. La section principale détaille le travail réalisé, les méthodes et choix techniques, ainsi que les problèmes rencontrés et leurs solutions. Enfin, le rapport conclut par une évaluation des résultats obtenus et des contributions, suivie des remerciements, d'un glossaire des termes techniques et d'une bibliographie des références utilisées.

2 Présentation

2.1 Présentation de l'entreprise

Edenred est une entreprise leader dans les solutions de paiement à destination des entreprises, des employés et des commerçants. Elle propose des services diversifiés comme les titres-restaurant, les cartes cadeaux, les solutions de gestion des frais professionnels, et bien d'autres. Les clients d'Edenred incluent des entreprises de toutes tailles, des institutions publiques et des particuliers. Son offre inclut des solutions variées telles que les cartes et ticket restaurant, les cartes cadeaux, les solutions de mobilité, ainsi que des services de gestion des avantages sociaux.

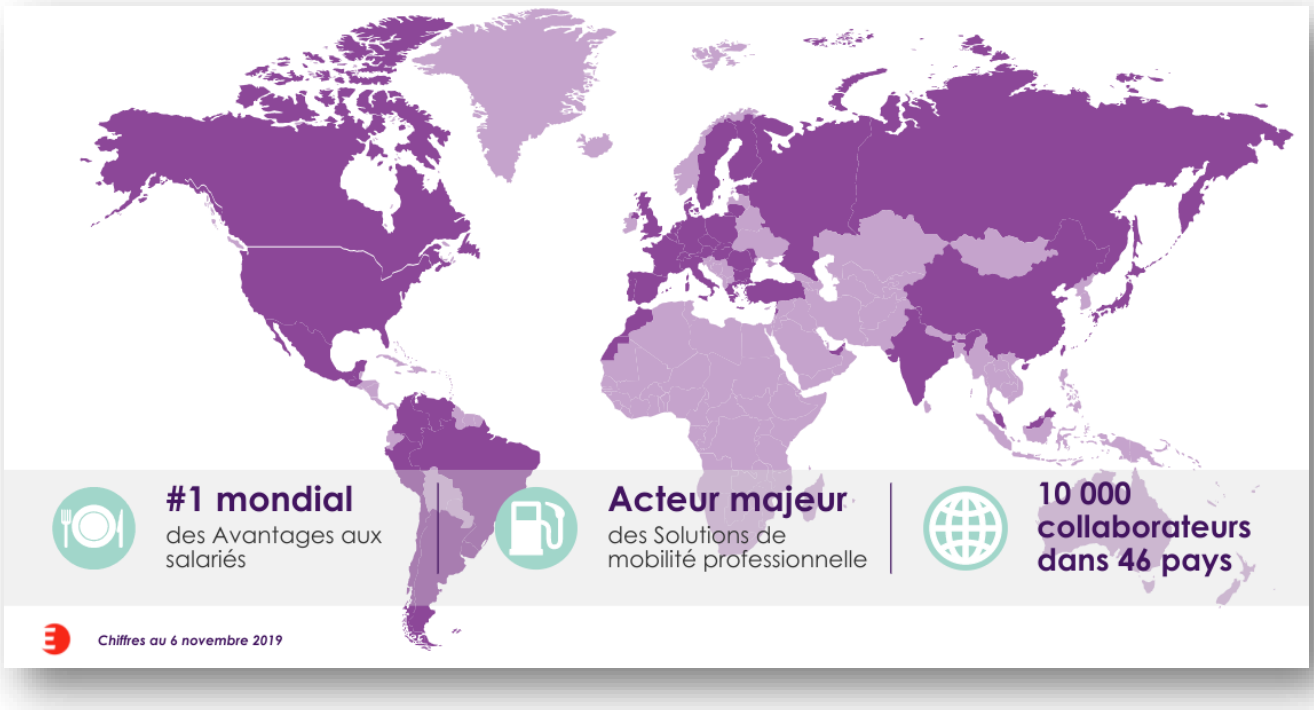

Figure 1 : Ceci est une carte de l'influence d'Edenred dans le monde

Edenred dévoile sa raison d'être

Figure 2 : Ceci est une vidéo de présentation de l'entreprise

En quelques chiffres le groupe c'est dix mille employés dans 46 pays, et les solutions d'Edenred c'est plus de cinq millions de salariés bénéficiaires rien qu'en France, renforçant leur pouvoir d'achat et leur bien-être au travail. Les commerçants bénéficient également de l'afflux de clients grâce aux solutions de paiement Edenred, augmentant ainsi leur chiffre d'affaires.

La culture d'entreprise d'Edenred valorise l'innovation et la collaboration. Les employés sont encouragés à travailler en équipe via Teams et à être flexibles, que ce soit en présentiel sur site ou en télétravail dans un environnement de travail dynamique. L'organisation informatique d'Edenred France est structurée sur trois sites principaux : **Malakoff**, **Saint-Martin-d'Hères** et **Lille**. Ces sites collaborent pour fournir des services cohérents et efficaces à travers tout le territoire.

L'organisation des services d'Edenred est structurée autour de plusieurs divisions et services spécialisés pour répondre aux besoins de ses clients. Les principaux départements incluent : le développement commercial qui est chargé de la prospection, de la gestion des relations avec les clients et de la vente des solutions Edenred. Les ressources humaines qui sont responsables de la gestion du recrutement et de la formation des employés. Les technologies de l'information qui ont la charge du développement, de la maintenance et de la sécurisation des plateformes numériques et des systèmes de paiement. Le service client qui assure le support et l'assistance aux clients, garantissant une expérience utilisateur optimale. Le marketing et communication qui développe les stratégies de communication et de promotion des produits Edenred, renforçant la notoriété de la marque. Et finance et administration qui gère les aspects financiers, la comptabilité et les opérations administratives de l'entreprise.

2.2 Présentation du service au sein de l'entreprise

Au sein d'Edenred France, j'ai intégré le **service <u>Cloud</u>** de Guillaume BORY**,** dédié à la gestion des infrastructures dans le Cloud, à l'orchestration de containers, et à la sécurisation des infrastructures informatiques. Ce service est crucial pour assurer le bon fonctionnement et la sécurité des systèmes d'information de l'entreprise.

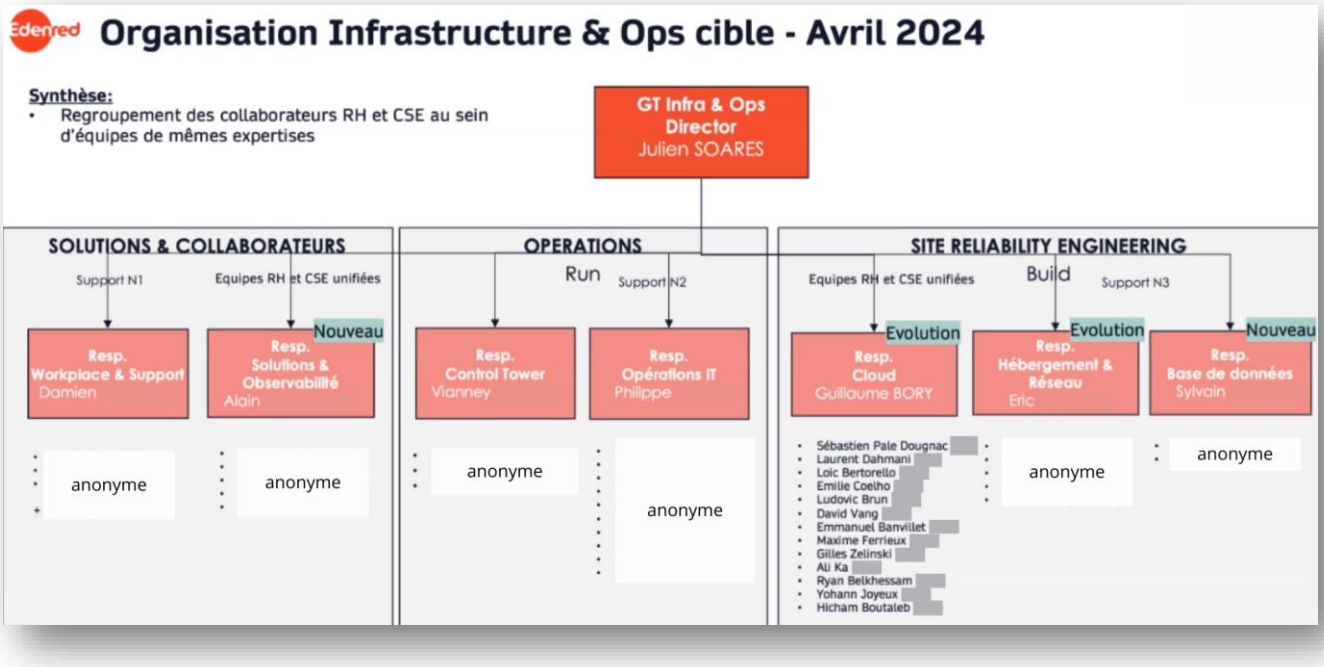

Figure 3 : Ceci est l'organigramme partiellement anonymisé du pôle Infrastructure & Opérations d'avril 2024

3 Présentation du cadre technique général du sujet

3.1 Définition des objectifs du stage

Le stage que j'ai effectué au sein du service Cloud d'Edenred France avait pour principal objectif de renforcer la sécurité de l'infrastructure informatique de l'entreprise. Le parc de serveurs est majoritairement basé sur CentOS 7. J'ai été intégré au projet de migration sur la partie préparation et durcissement. Mon rôle consistait à préparer et sécuriser l'environnement pour permettre la migration des serveurs de CentOS 7 vers Debian 12. Pour ce faire, je me suis appuyé sur les recommandations de sécurité du CIS (Center for Internet Security) afin de garantir que les nouveaux systèmes répondent aux normes de sécurité les plus élevées.

L'un des principaux objectifs était de réaliser la transition des systèmes d'exploitation des serveurs de CentOS 7 vers Debian 12. Avec l'arrêt du support de CentOS 7, le choix de Debian 12 comme nouvel OS Linux standard s'est marqué par les avantages suivants : les packages Debian ont une empreinte plus petite, ils n'embarquent que le strict nécessaire et ainsi offrent moins de surface d'attaque ; la migration d'une version d'OS à une autre est une simple formalité ; Debian est la distribution Linux la plus stable pour un usage en production.

Un autre objectif central était l'implémentation des recommandations de sécurité du CIS pour Debian Linux 12 sur l'ensemble du parc informatique. Les recommandations CIS sont des standards de sécurité reconnus qui permettent de protéger les systèmes contre les vulnérabilités courantes. Pour atteindre cet objectif, j'ai dû lire et comprendre les recommandations, puis les impémenter en écrivant des scripts Ansible qui permettront le déploiement de ces recommandations sur l'ensemble du parc de serveur géré par l'équipe, soit environ 400 serveurs.

L'automatisation des configurations de sécurité était également un objectif clé. Utiliser Ansible pour automatiser le processus de configuration des systèmes selon les recommandations CIS permettait d'assurer une application cohérente et répétable des configurations de sécurité, réduisant ainsi les erreurs humaines et améliorant l'efficacité opérationnelle. Les activités associées comprenaient le développement de playbooks Ansible, la création de pipelines d'intégration continue (CI) dans GitLab, et la validation et le déploiement des configurations automatisées.

La mise en place de pipelines d'intégration continue (CI) pour valider les modifications de code de configuration avant leur déploiement en production était un autre objectif majeur. Les pipelines CI permettent de détecter les erreurs de configuration et les problèmes de sécurité avant qu'ils n'affectent les systèmes de production. Le durcissement des systèmes ajoute de la complexité, ce qui peut parfois inciter les collègues à retirer une configuration gênante. Pour éviter les erreurs ou les retours en arrière involontaires, j'ai écrit des tests automatisés et exécuté des tests de déploiement et de vérification des configurations.

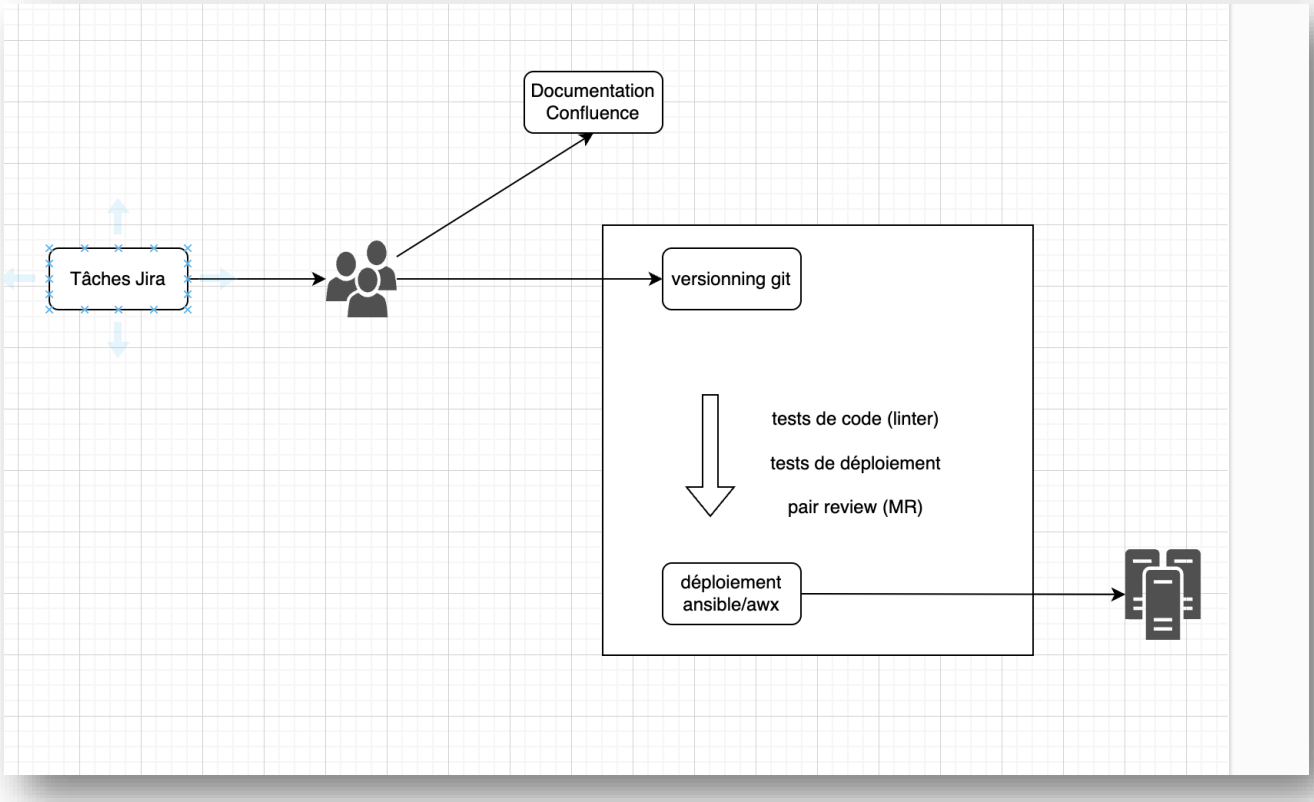

Figure 4 : Ceci est le schéma de l'usine logicielle

Enfin, la documentation faisait partie des objectifs du stage. Une documentation claire et détaillée est essentielle pour la continuité des opérations et pour permettre aux autres membres de l'équipe de comprendre et de reproduire les configurations. Les activités comprenaient la rédaction de guides et de tutoriels, la mise à jour des documents de configuration dans la base documentaire Confluence, et la présentation des résultats aux équipes concernées.

3.2 Contexte théorique et historique

Dans le contexte actuel où les cyberattaques sont de plus en plus fréquentes et sophistiquées, la sécurité des systèmes d'information est une priorité absolue pour les entreprises. Les recommandations du CIS fournissent un cadre détaillé et éprouvé pour sécuriser les systèmes informatiques contre une variété de menaces. L'adoption de ces recommandations permet non seulement de renforcer la sécurité, mais aussi de garantir la conformité, qui sont essentielles pour les entreprises manipulant des données sensibles, notamment dans les secteurs financiers et transactionnels.

Au fil des années, CentOS, une distribution Linux dérivée de Red Hat Enterprise Linux, a été largement adoptée par les entreprises pour sa stabilité et son support à long terme. Cependant, en décembre 2020, Red Hat a annoncé que CentOS 7, atteindrait sa fin de vie le 30 juin 2024, beaucoup plus tôt que prévu. Cette annonce a incité de nombreuses entreprises, y compris Edenred, à rechercher des alternatives pour garantir la continuité de leurs opérations sans compromettre la sécurité.

La fin de la distribution en tant que serveur a posé des défis significatifs pour les entreprises qui dépendaient de cette distribution pour leurs opérations critiques. En réponse à cette situation, Edenred a choisi en 2023 de migrer vers Debian 12, une autre distribution Linux reconnue pour sa stabilité, sa sécurité et son vaste support communautaire. Debian est réputée pour son engagement envers la libre distribution et sa gestion rigoureuse des paquets, ce qui en fait une option idéale pour les environnements de production sensibles.

L'automatisation des configurations de sécurité à l'aide d'Ansible s'inscrit dans le cadre théorique de la gestion moderne des infrastructures informatiques, connue sous le nom de DevOps. DevOps est une approche qui combine les pratiques de développement logiciel (Dev) et des opérations IT (Ops) pour améliorer la capacité des entreprises à fournir des applications et des services à grande vitesse. L'automatisation joue un rôle clé dans DevOps, permettant une gestion efficace et reproductible des configurations système.

3.3 Présentation du cahier des charges

Le cahier des charges du projet de stage chez Edenred a été la recommandation CIS de 2024. Ce document définissait les objectifs, les compétences techniques requises, ainsi que les moyens techniques à utiliser pour mener à bien ce projet.

Compétences techniques requises

Pour réaliser ce projet, plusieurs compétences techniques étaient nécessaires :

- **Administration système** : Il faut une connaissance approfondie des systèmes d'exploitation Linux, en particulier CentOS et Debian.
- **Automatisation avec Ansible** : Une capacité à écrire, tester et déployer avec Ansible pour automatiser les configurations de sécurité et les tâches administratives.
- **Gestion des versions et CI/CD** : Être familiarisé avec GitLab pour le versionnement du code et la mise en place de *pipelines* d'intégration continue (CI) et de déploiement continu (CD).
- **Sécurité informatique** : Une compréhension des concepts de sécurité informatique, y compris les recommandations du CIS.
- **Script Shell** : Des compétences en scripting Bash pour automatiser les tâches et vérifier les configurations de sécurité.

Compétences à acquérir

Au cours du stage, plusieurs compétences techniques supplémentaires étaient à acquérir pour compléter celles déjà requises :

- **Implémentation des recommandations CIS** : Apprendre les recommandations CIS spécifiques à Debian 12 et leur application pratique.
- **Utilisation de Jinja2** : Des compétences en *templating* avec Jinja2 pour créer des configurations dynamiques et flexibles sur les serveurs de CentOS 7 à Debian 12 dans les playbooks Ansible.
- **Gestion d'inventaires avec AWX** : L'utilisation d'AWX pour orchestrer les déploiements et gérer les inventaires de systèmes de manière centralisée et efficace.

Moyens techniques utilisés

Pour atteindre les objectifs définis dans le cahier des charges, plusieurs moyens techniques ont été mis à disposition :

- **Ansible** : Utilisé pour l'automatisation des configurations système. Ansible permet d'écrire des playbooks en YAML pour appliquer de manière cohérente et répétée les configurations de sécurité.
- **GitLab** : Plateforme utilisée pour le versionnage du code, la gestion des branches, et la mise en place de pipelines CI/CD. GitLab CI/CD a permis de tester automatiquement les modifications de code avant leur déploiement en production.
- **AWX** : Version open-source d'Ansible Tower, utilisée pour gérer les inventaires et orchestrer les déploiements. AWX offre une interface utilisateur facilitant la planification des tâches et la gestion des accès.
- **Scripts Bash** : Utilisés pour les tâches d'automatisation spécifiques, telles que la vérification des configurations de sécurité et l'application des modifications nécessaires.
- **Documentation CIS** : Recommandations du Center for Internet Security pour Debian 12 de 2024, fournis un cadre détaillé pour durcir la sécurité des systèmes.
- **Environnements de développement et de test** : Serveurs de test et de préproduction configurés pour permettre la validation des configurations avant leur déploiement sur les systèmes de production.
- **La suite Atlassian (Jira et Confluence) :** Jira est un logiciel qui sert à piloter et organiser l'activité d'une équipe informatique, en créant et distribuant des tâches sous forme de ticket. Confluence est un logiciel de documentation qui sert généralement de référentiel de documentation technique pour les équipes.

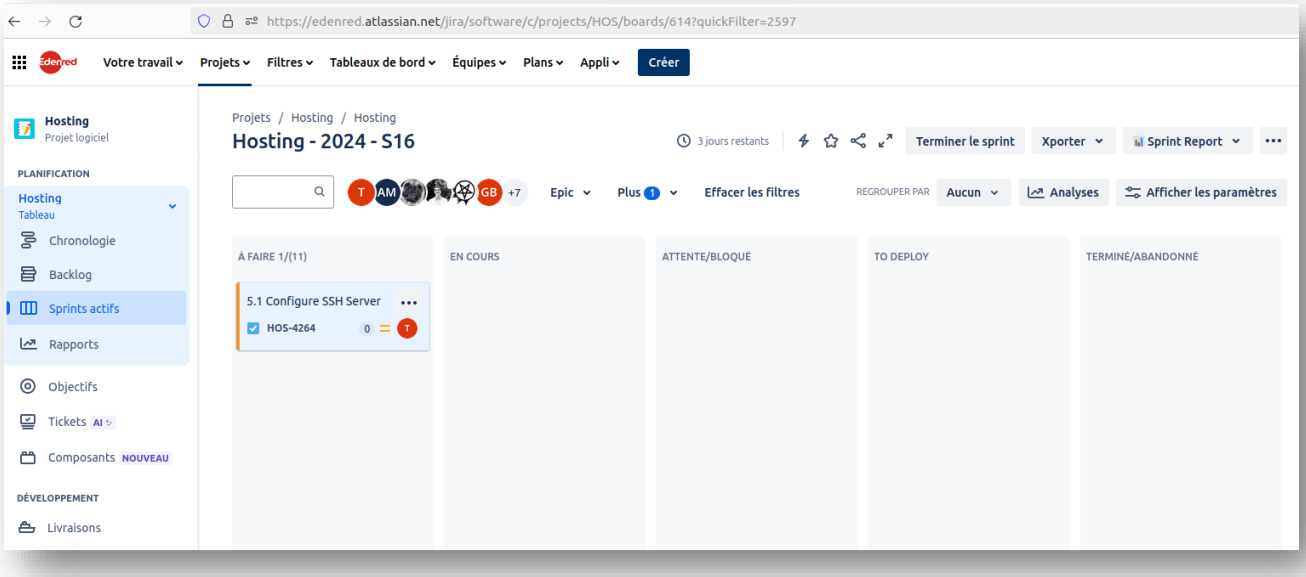

Figure 4 : Ceci est l'interface de gestion de tickets Jira avec dans la colonne "à faire" mon premier ticket

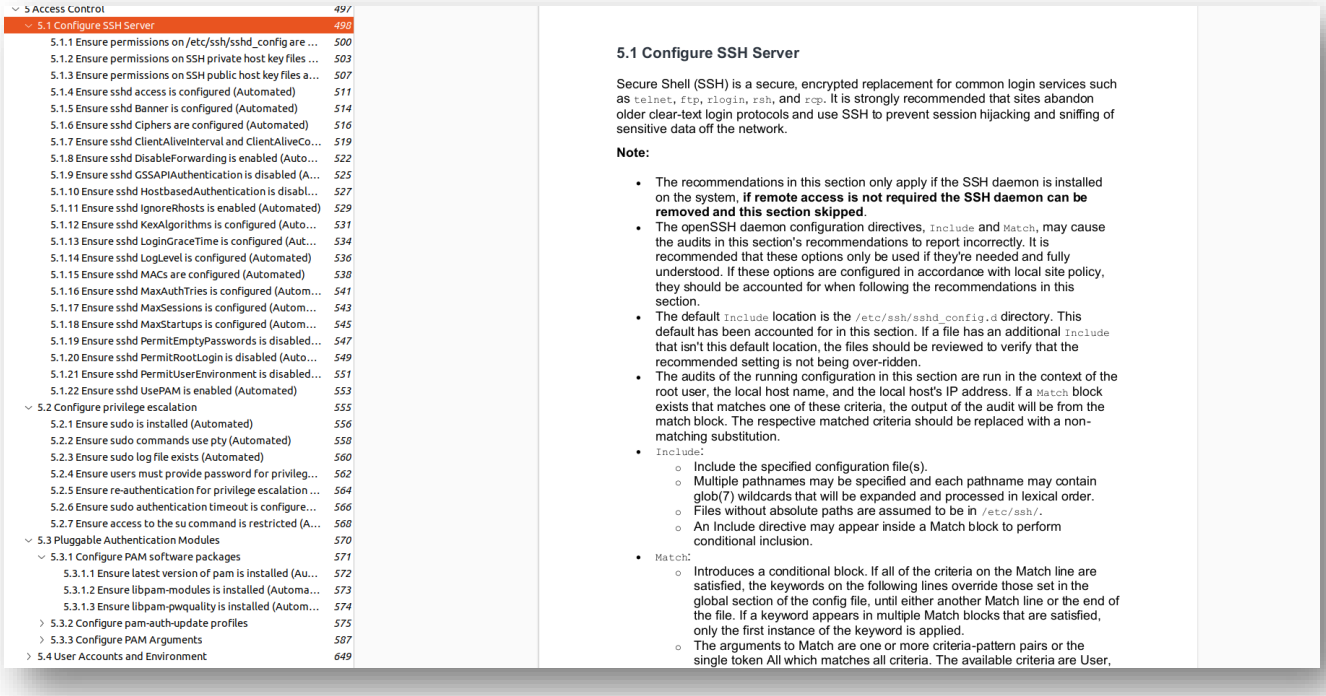

Figure 5 : Ceci est la page 498 de la recommandation CIS associé à mon premier ticket Jira

4 Présentation du travail réalisé

4.1 Justification des méthodes et des choix techniques

Les méthodes et choix techniques du projet de durcissement de la sécurité des systèmes Debian 12 chez Edenred France reposaient sur les recommandations du CIS.

Pour réaliser le travail en conformité avec le cahier des charges, plusieurs choix techniques et méthodologiques ont été effectués en se basant sur les objectifs du projet, les meilleures pratiques du secteur, de l'équipe et les outils disponibles. Voici un aperçu détaillé des méthodes et des choix techniques adoptés :

- Utilisation d'Ansible pour l'automatisation : Ansible est utilisé comme principal outil d'automatisation en raison de sa simplicité, de sa flexibilité et de sa large adoption dans l'administration système.
- Ansible permet d'écrire des scripts (*playbooks*) en YAML, facilitant ainsi la lecture et la maintenance du code. De plus,
- Ansible est sans agent, ce qui réduit la charge sur les machines clientes et simplifie le déploiement.

Adoption des recommandations CIS pour le durcissement : Les recommandations du Center for Internet Security sont des standards largement reconnus pour sécuriser les systèmes d'exploitation. En suivant ces recommandations, nous assurons que notre infrastructure est protégée contre les vulnérabilités courantes et conforme aux meilleures pratiques de sécurité. Les recommandations CIS pour Debian Linux 12 ont été rigoureusement appliquées pour garantir un niveau élevé de sécurité.

Utilisation du *benchmark* CIS développé par OVH : Le script de benchmark CIS développé et rendu disponible en OpenSource par OVH a été intégré pour tester l'ensemble des audits de sécurité et fournir un score de conformité. Ce score permet de visualiser les points à corriger et d'orienter les efforts d'amélioration. En utilisant ce benchmark, nous avons pu identifier rapidement les configurations non conformes et apporter les ajustements nécessaires pour améliorer la sécurité globale. Cet outil s'est avéré précieux pour évaluer et améliorer continuellement la posture de sécurité de l'infrastructure.

Utilisation de GitLab pour le versionnage et l'intégration continue : GitLab est utilisé pour gérer le versionnage du code et automatiser les tests grâce à ses fonctionnalités d'intégration continue (CI). Chaque modification de code passe par une série de tests automatiques pour vérifier la syntaxe, déployer les modifications sur des environnements de test et vérifier leur bon fonctionnement. Cette approche assure que les modifications sont fiables et ne perturbent pas l'existant.

Utilisation de AWX pour la gestion des inventaires et des déploiements : AWX a été utilisé pour gérer les inventaires et orchestrer les déploiements. AWX offre une interface utilisateur conviviale, permet de planifier des tâches et de gérer les rôles et les accès, facilitant ainsi l'administration des configurations à grande échelle.

Structuration des playbooks par rôle et composant : Pour faciliter la maintenance et la réutilisabilité des configurations, les playbooks Ansible ont été structurés par rôle et composant. Par exemple, des rôles spécifiques pour Apache, MySQL, et SSH ont été définis dans des répertoires distincts. Cette approche modulaire permet d'appliquer des configurations spécifiques à des groupes de machines tout en conservant une structure claire et organisée.

Utilisation de Jinja2 pour le *templating* : Jinja2, un moteur de templating intégré à Ansible, a été utilisé pour créer des configurations dynamiques et flexibles. Grâce à Jinja2, il est possible de personnaliser les fichiers de configuration en fonction des variables d'inventaire, ce qui permet de gérer efficacement les différences entre les environnements des deux systèmes d'exploitation.

Mise en place d'une stratégie d'immutabilité : La stratégie d'immutabilité vise à assurer que les configurations soient toujours en état conforme, et ce, de manière automatique. En écrivant des modifications de sécurité avec des playbooks Ansible qui vérifient et appliquent les configurations nécessaires à chaque exécution.

Tous ces choix techniques, tels que l'utilisation d'Ansible, GitLab, AWX, les recommandations CIS, et le benchmark OVH, ont permis de répondre efficacement aux exigences du cahier des charges. Ils ont amélioré la sécurité et la fiabilité de l'infrastructure tout en facilitant la gestion et la maintenance des configurations à grande échelle. Grâce à ces méthodes, nous avons pu atteindre un haut niveau de conformité et garantir un environnement sécurisé et robuste pour l'entreprise.

4.2 Description du travail accompli pour la mise en œuvre des solutions

Pour illustrer mon travail accompli durant ce stage, j'ai sélectionné les éléments les plus représentatifs des tickets que j'ai remaniés.

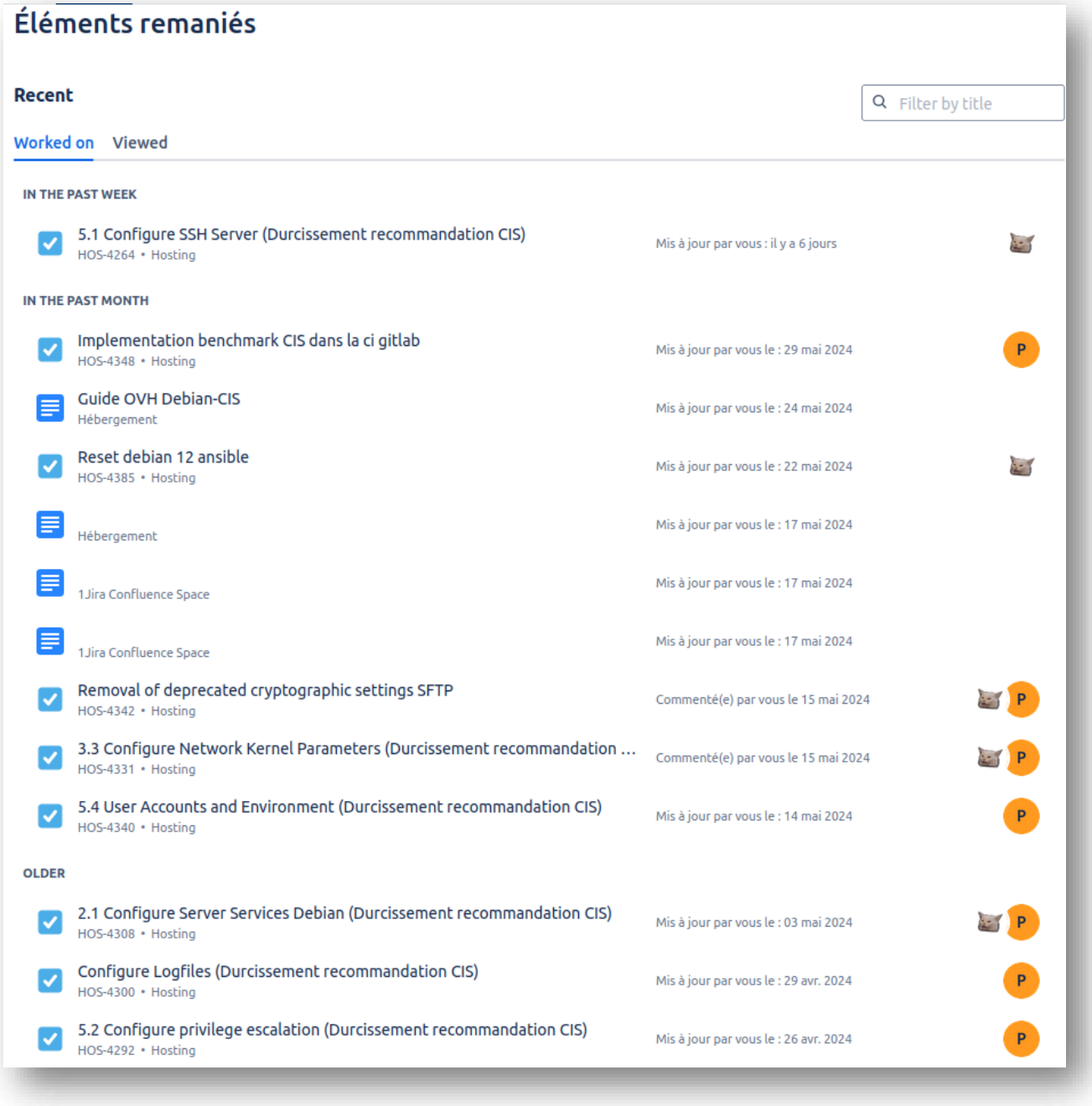

Figure 6 : Ceci est l'ensemble des tickets que j'ai réalisé qui est listé sur Jira

Création et gestion du ticket Jira : afin d'offrir une vue d'ensemble claire et détaillée, je vais me concentrer sur un service bien connu des Réseaux et Télécommunications (R&T) : le service SSH. Le 16 avril 2024, j'ai créé mon premier ticket Jira pour la configuration du service SSH selon les recommandations CIS pour Debian Linux 12. Ce ticket, numéroté pour référence, a été le point de départ de mon travail sur la sécurisation du service SSH.

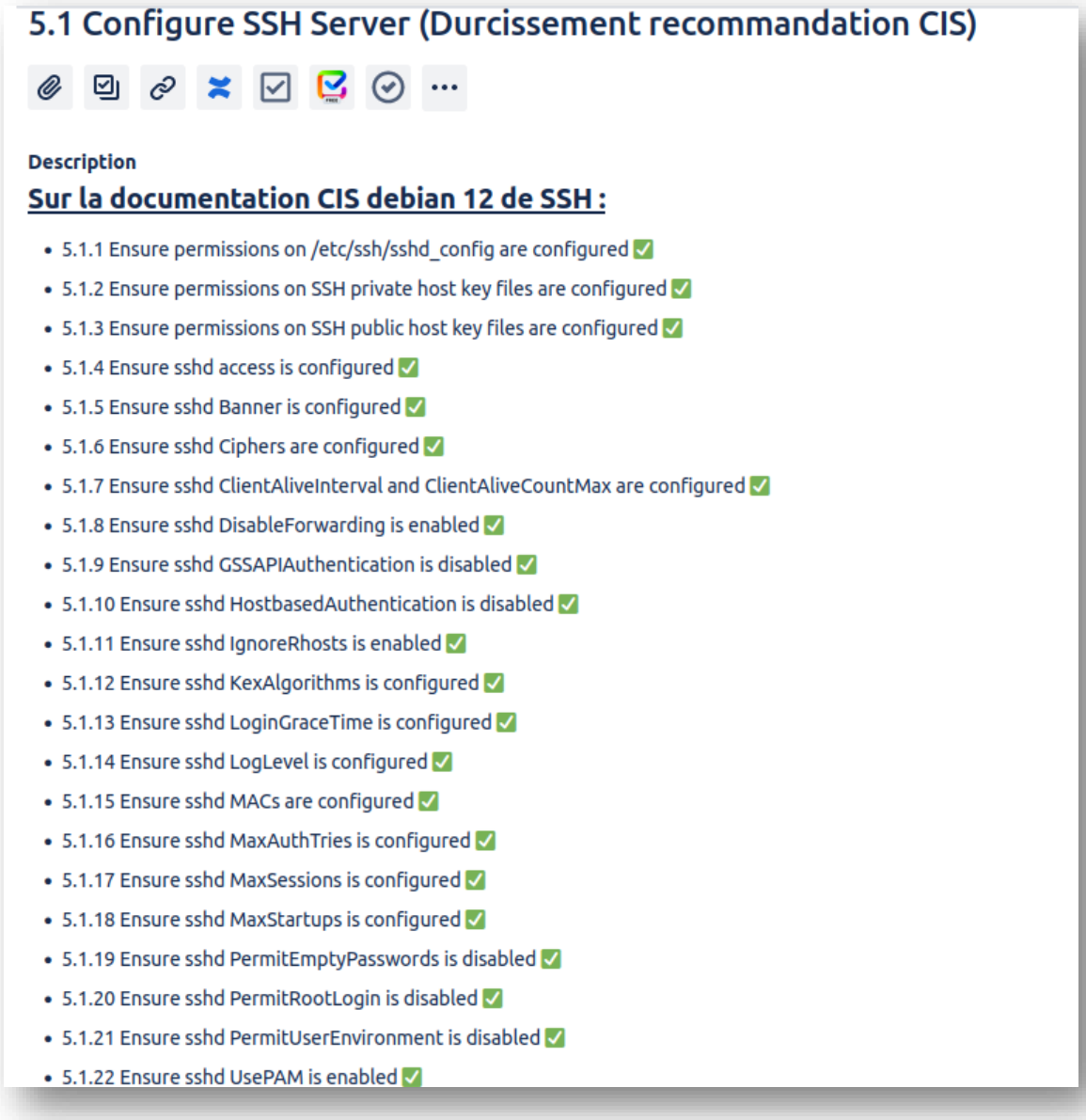

Figure 7 : Ceci est le ticket Jira associé à la configuration du serveur SSH sur la recommandation CIS

Développement et versionnage du code : pour traiter ce ticket, j'ai créé une nouvelle branche GitLab depuis Visual Studio Code portant le même nom que le ticket sur Jira. Cette branche m'a permis de travailler de manière isolée sur les modifications nécessaires sans affecter la branche principale, nommée "master". Chaque modification de code que j'effectuais était Commit et proposé à la branche principale master.

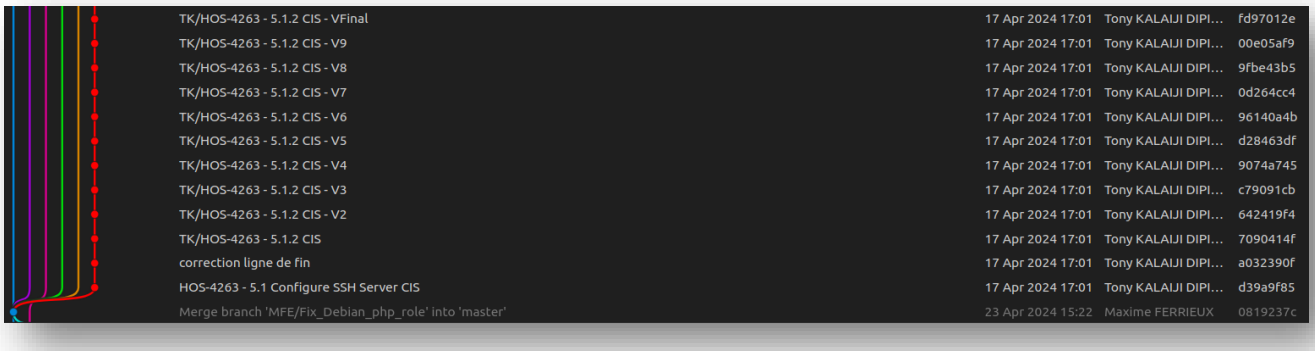

Figure 8 : Ceci est le Git Graph de toutes les modifications et MR ajouté

Explication du code ajouté : voici une explication point par point du code ajouté pour assurer que les permissions sur les fichiers de clés privées SSH sont correctement configurées selon les recommandations CIS pour Debian Linux 12.

```
62 +
63 + # CIS Debian Linux 12 - 5.1.2 Ensure permissions on SSH private host key files are configured
64 + - name: Trouver tous les fichiers de clés privées SSH dans /etc/ssh
65 +   ansible.builtin.find:
66 +     paths: /etc/ssh
67 +     patterns: '*_key'
68 +     file_type: file
69 +   register: ssh_private_keys
70 +
71 + - name: Définir les permissions et le groupe des fichiers de clés privées SSH
72 +   ansible.builtin.file:
73 +     path: "{{ fichier_private.path }}"
74 +     owner: root
75 +     group: "{{ ssh_config.keys_group_owner | default('root') }}"
76 +     mode: '0600'
77 +   loop: "{{ ssh_private_keys.files }}"
78 +   loop_control:
79 +     loop_var: fichier_private
80 +   notify:
81 +     - Test de sshd
82 +     - Redémarrage de sshd
83 +
84 + # CIS Debian Linux 12 - 5.1.3 Ensure permissions on SSH public host key files are configured
```

Figure 9 : Ceci est un code que j'ai ajouté qui assure que les autorisations sur les fichiers de clés privées SSH sont configurées.

- **Trouver tous les fichiers de clés privées SSH dans /etc/ssh :**

La tâche "Trouver tous les fichiers de clés privées SSH dans /etc/ssh" (ligne 64) : Cette ligne indique que l'objectif de la tâche est de trouver tous les fichiers de clés privées SSH dans le répertoire /etc/ssh.

Le module ansible.builtin.find (ligne 65) : Ce module recherche des fichiers ou des répertoires dans un chemin spécifié.

La directive register: ssh_private_keys (ligne 69) : Enregistre le résultat de la recherche dans une variable ssh_private_keys.

- **Définir les permissions et le groupe des fichiers de clés privées SSH :**

La tâche "Définir les permissions et le groupe des fichiers de clés privées SSH" (ligne 71) : Indique que cette tâche est destinée à définir les permissions et le groupe des fichiers de clés privées SSH.

Le module ansible.builtin.file (ligne 72) : Utilisé pour gérer les permissions des fichiers et des répertoires.

La directive loop: "{{ ssh_private_keys.files }}" (ligne 77) : Indique que cette tâche doit être exécutée en boucle pour chaque fichier trouvé par la tâche find. La variable {{ ssh_private_keys.files }} contient la liste des fichiers trouvés.

Le loop_control (ligne 78) : Utilisé pour contrôler le comportement de la boucle.

Le loop_var: fichier_private (ligne 79) : Définit le nom de la variable pour chaque élément dans la boucle. Ici, chaque fichier sera référencé par fichier_private.

Le notify (ligne 80) : Déclenche des *handlers* à la fin de cette tâche si des modifications sont effectuées.
Comme le test de sshd : Indique que le service sshd doit être testé après modification.
Et redémarrage de sshd : Indique que le service sshd doit être redémarré après modification.

Tests et validation : une fois le code finalisé, il était testé sur des machines de test spécifiquement configurées pour Debian 12, nommées “debian12-ansible”. Ces tests visaient à s'assurer que les configurations appliquées fonctionnaient correctement et amélioraient la sécurité sans introduire de problème de fonctionnement.

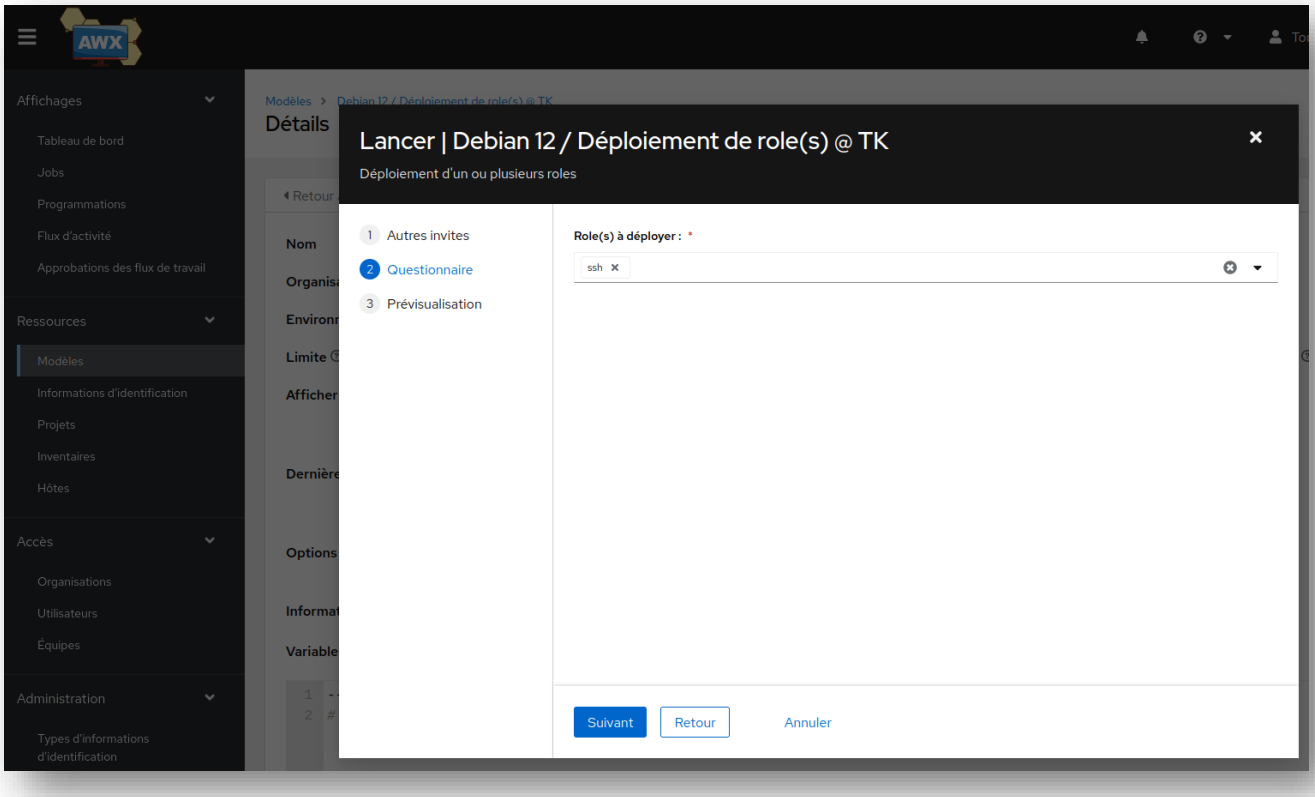

Figure 10 : Ceci est l’interface AWX pour déployer le rôle SSH sur la machine de test ”debian12-ansible"

Revue et Proposition de Merge Request (MR) : Après la phase de test, le code devait être approuvé par au moins deux membres de l'équipe avant d'être intégré à la branche principale master. Ce processus obligatoire de validation garantissait que les modifications respectaient les standards de qualité et de sécurité de l'entreprise.
À chaque étape de mon développement, je proposais une Merge Request (MR) pour intégrer les modifications dans la branche principale. Ce processus de MR permettait une revue de code approfondie par d'autres membres de l'équipe, garantissant ainsi la qualité et la sécurité des modifications apportées.

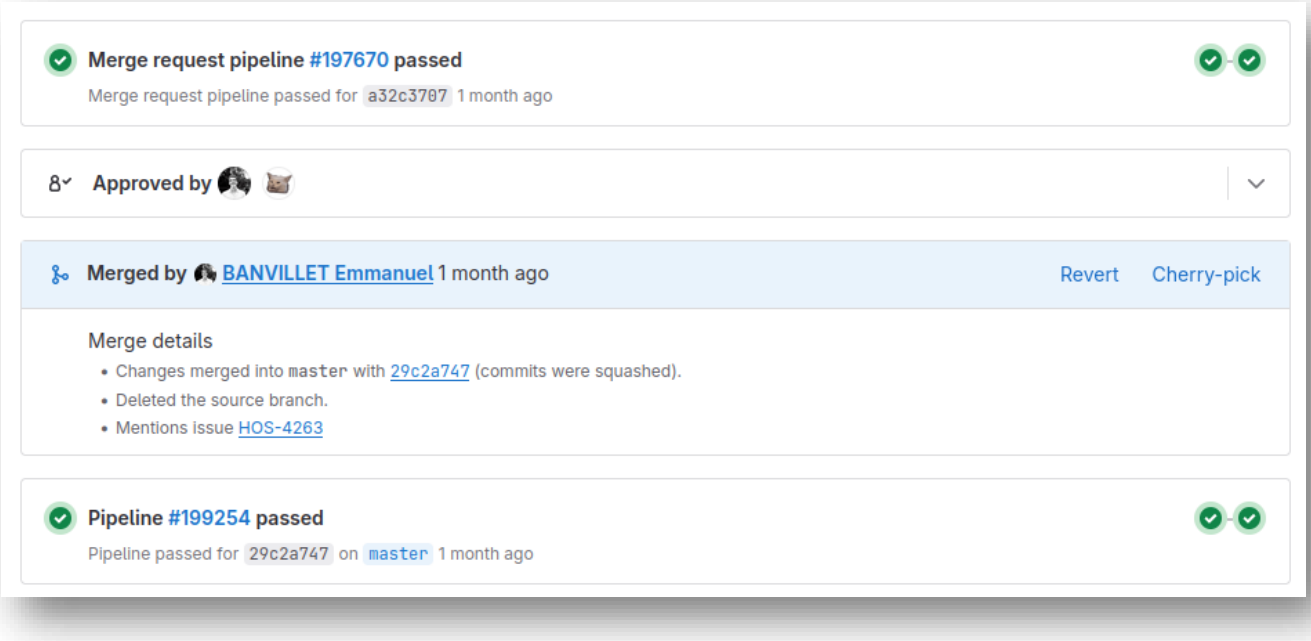

Figure 11 : Ceci est l'approbation du code par 2 membres

Déploiement à grande échelle : une fois le code intégré dans la branche principale "master", un membre de l'équipe disposant des permissions nécessaires appliquait le code à l'ensemble des machines clientes du parc Edenred France, comprenant plus de 400 machines. Ce déploiement à grande échelle assurait que toutes les machines étaient mises à jour avec les configurations de sécurité les plus récentes, garantissant ainsi une protection uniforme et renforcée sur l'ensemble du parc informatique.

Cette méthodologie de travail, impliquant la création de tickets détaillés, le développement en branches isolées, les propositions de MR, les tests rigoureux, la revue par les développer de l'équipe, et le déploiement à grande échelle, a permis de mettre en œuvre des solutions de manière structurée et sécurisée.

Mon plus gros projet durant ce stage

Maintenant je vais vous parler de la réalisation de mon plus grand projet chez Edenred, j'ai mis en œuvre l'intégration du benchmark CIS d'OVH pour Debian 12 dans la CI GitLab. Ce projet comportait plusieurs étapes clés, décrites ci-dessous.

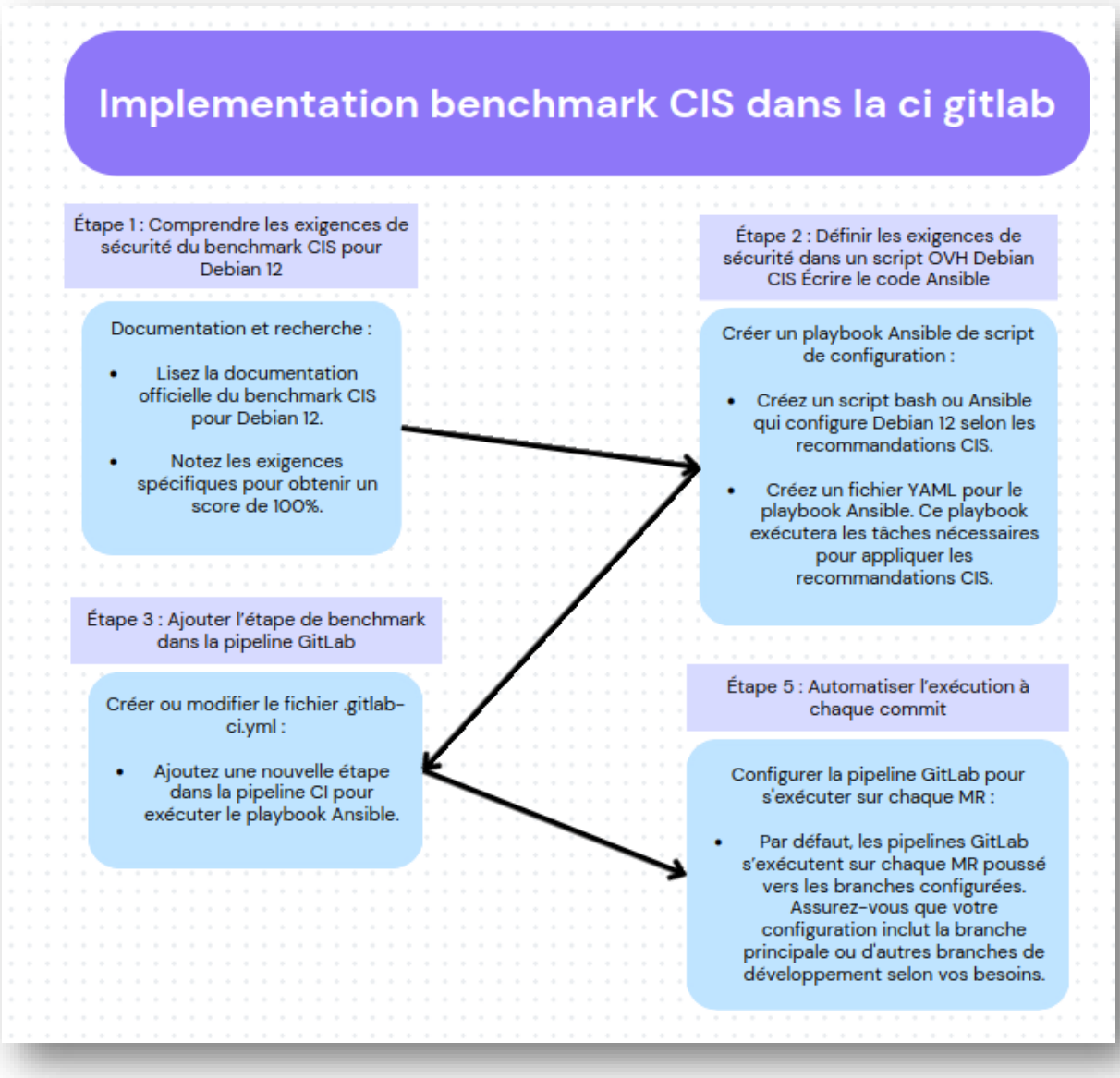

Figure 12 : Ceci est le plan d'action initial pour ce projet

Plan d'Action

1. Compréhension des Exigences de Sécurité du Benchmark CIS :
   - Analyse approfondie des exigences de sécurité définies par le benchmark CIS pour Debian 12.

2. Définition des Exigences de Sécurité dans un Script et Écriture du Code Ansible
   - Développement d'un script détaillant les exigences de sécurité.
   - Écriture du code Ansible pour automatiser le déploiement et la configuration des scripts de sécurité.

3. Ajout de l'Étape du Benchmark dans GitLab CI
   - Intégration de l'étape de benchmark dans le pipeline GitLab CI.

4. Automatisation de l'Exécution à Chaque Commit
   - Configuration pour que l'exécution du benchmark soit automatique à chaque commit.
   - Mise en place d'un système pour afficher un code de sortie si le score n'atteint pas 100%.

Procédure pour Atteindre un Score de 100% au Benchmark CIS

Pour garantir un score de 100% au benchmark CIS, j'ai suivi cette procédure rigoureuse :

1. <u>Préparation et déploiement Initial</u>
   - Attente de la modification ajoutant mes derniers codes de hardening Ansible.
   - Demande à quelqu'un de l'équipe d'une machine Debian vierge.
   - Lancement d'une simulation de déploiement sur la machine vierge.

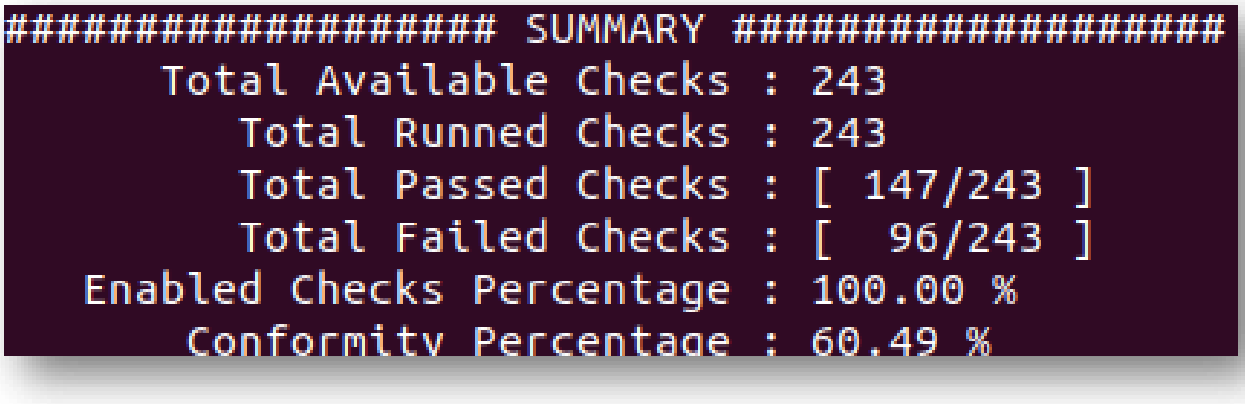

Figure 8 : Ceci est la sortie du score du benchmark OVH CIS

2. Identification et désactivation des modules échouant
   - Exécution du script CIS d'OVH pour identifier les audits en échec.
   - Écriture d'un script Bash (update_status.sh) pour désactiver les audits échouant en modifiant les fichiers de configuration du benchmark.

```bash
#!/bin/bash

# Chemin absolu du fichier list_excludes_cis.txt
LIST_CIS_FILE="/tmp/debian-cis/list_excludes_cis.txt"

# Vérifie si le fichier list_excludes_cis.txt existe
if [ ! -f "$LIST_CIS_FILE" ]; then
    echo "Le fichier $LIST_CIS_FILE est introuvable."
    exit 1
fi

# Boucle sur chaque ligne du fichier list_excludes_cis.txt
while IFS= read -r filename; do
    # Vérifie si le fichier etc/conf.d/$filename existe
    if [ -f "/tmp/debian-cis/etc/conf.d/$filename" ]; then
        # Utilise sed pour remplacer status=audit par status=disabled dans le fichier
        sed -i 's/status=audit/status=disabled/g' "/tmp/debian-cis/etc/conf.d/$filename"
        echo "Modifié: /tmp/debian-cis/etc/conf.d/$filename"
    else
        echo "Le fichier /tmp/debian-cis/etc/conf.d/$filename est introuvable."
        exit 1
    fi
done < "$LIST_CIS_FILE"
```

Figure 9 : Ceci est le script update_status.sh

3. Vérification du Score de Conformité
   - Développement du script Bash (check_benchmark.sh) pour vérifier le score de conformité.
   - Ecriture du code ansible pour l'exécution des scripts pour s'assurer d'un score de 100%.

```bash
#!/bin/bash

# Exécuter le benchmark OVH CIS
result=$(bin/hardening.sh --audit)

# Extraire le pourcentage de conformité
conformity_percentage=$(echo "$result" | grep -oP 'Conformity Percentage\s+:\s+\K[0-9.]+')

# Vérifier si le pourcentage est de 100%
if [ "$conformity_percentage" == "100.00" ]; then
  echo "Le score est de 100%."
  exit 0
else
  echo "Le score n'est pas de 100%."
  echo "$result" | grep "Check Failed"
  echo "$result"
  exit 1
fi
```

Figure 10 : Ceci est le script check_benchmark.sh

```yaml
# Déploiement du Benchmark CIS de OVH pour tester le score.
# Si le résultat != 100, un exit code de 1 sera affiché.
#

- name: Installation benchmark Debian CIS
  ansible.builtin.git:
    repo: 'https://github.com/ovh/debian-cis.git'
    dest: '/tmp/debian-cis'
    version: 'master'
    update: false

- name: Configuration de CIS
  ansible.builtin.shell: |
    cd /tmp/debian-cis
    cp debian/default /etc/default/cis-hardening
    sed -i "s#CIS_LIB_DIR=.*#CIS_LIB_DIR='$(pwd)'/lib#" /etc/default/cis-hardening
    sed -i "s#CIS_CHECKS_DIR=.*#CIS_CHECKS_DIR='$(pwd)'/bin/hardening#" /etc/default/cis-hardening
    sed -i "s#CIS_CONF_DIR=.*#CIS_CONF_DIR='$(pwd)'/etc#" /etc/default/cis-hardening
    sed -i "s#CIS_TMP_DIR=.*#CIS_TMP_DIR='$(pwd)'/tmp#" /etc/default/cis-hardening
    bin/hardening.sh --create-config-files-only
  changed_when: false

- name: Ajout du script update_status.sh
  ansible.builtin.copy:
    src: update_status.sh
    dest: /tmp/debian-cis
    owner: root
    group: root
    mode: '0700'

- name: Ajout du script list_excludes_cis.txt
  ansible.builtin.copy:
    src: list_excludes_cis.txt
    dest: /tmp/debian-cis
    owner: root
    group: root
    mode: '0600'

- name: Execution update_status.sh
  ansible.builtin.shell:
    cmd: ./update_status.sh
    chdir: /tmp/debian-cis
  changed_when: false

- name: Ajout du script check_benchmark
  ansible.builtin.copy:
    src: check_benchmark.sh
    dest: /tmp/debian-cis
    owner: root
    group: root
    mode: '0700'

- name: Execution check_benchmark.sh
  ansible.builtin.shell:
    cmd: ./check_benchmark.sh
    chdir: /tmp/debian-cis
  changed_when: false
```

Figure 11 : Ceci est le code final sur ansible du Déploiement du Benchmark CIS de OVH pour tester le score, si le résultat est différent de 100%, un exit code retour de 1 sera affiché

Pour ce projet, j’ai rédigé un guide sur le gestionnaire de documentation de l’entreprise : Confluence, intitulé "Guide OVH Debian-CIS". Il décrit en détail les étapes suivies pour intégrer et automatiser le benchmark CIS dans l'environnement de développement et de déploiement continu. Il explique comment j’ai utilisé et configuré des scripts de sécurité pour renforcer les systèmes. Il inclut des instructions pour l'installation rapide, la configuration des scripts, et l'automatisation des vérifications de sécurité avec GitLab CI pour garantir un score de conformité de 100%.

Dans le futur, si une configuration était modifiée, modifiant une étape du durcissement, alors le score calculé par le script serait inférieur à 100%, empêchant ainsi la contribution de code, et donc le déploiement de cette régression en termes de sécurité.

J’ai mis ce document en **Annexe de ce rapport de stage**.

4.3 Analyse des problèmes rencontrés et solutions adoptées

Au cours de mon stage, j'ai rencontré plusieurs défis techniques liés à la mise en œuvre des recommandations de sécurité CIS pour Debian Linux 12. Voici une analyse des principaux problèmes rencontrés et les solutions adoptées pour les surmonter.

Incompatibilité des scripts CIS avec certaines configurations existantes

Analyse : Les recommandations de sécurité CIS fournissent des lignes directrices générales qui ne prennent pas toujours en compte les spécificités de l'infrastructure existante. Dans le cas d'Edenred, certaines configurations spécifiques ne correspondaient pas directement aux recommandations standard, entraînant des conflits et des erreurs lors de l'application des scripts CIS.

Solution : Pour résoudre ce problème, j'ai modifié les scripts Ansible pour adapter les recommandations CIS aux spécificités de l'infrastructure d'Edenred. J'ai utilisé des conditions dans les *playbooks* pour gérer les exceptions et assurer une application flexible et contextuelle des configurations de sécurité. Par exemple, certaines règles de permissions ont été ajustées pour ne pas affecter les fichiers critiques propres à l'infrastructure d'Edenred.

Échecs récurrents des tests de vérification syntaxique ansible-lint

Analyse : ansible-lint est un outil essentiel pour garantir que les playbooks Ansible respectent les meilleures pratiques et les standards de codage. Cependant, les tests ansible-lint échouaient fréquemment en raison de problèmes de style et de syntaxe, retardant le processus de validation et de déploiement des configurations.

Solution : Pour minimiser les échecs des tests ansible-lint, j'ai installé l'outil localement sur ma machine de développement. Cela m'a permis de tester les configurations avant de soumettre les merge requests (MR). En détectant et en corrigeant les erreurs de style et de syntaxe en amont, j'ai pu réduire significativement le nombre de problèmes lors de la revue de code et accélérer le processus de validation.

Difficulté à atteindre un score de 100% de conformité

Analyse : Obtenir un score de conformité de 100% aux recommandations CIS est un objectif ambitieux mais crucial pour garantir la sécurité optimale des systèmes. Cependant, certaines configurations échouaient systématiquement, rendant difficile l'atteinte de ce score parfait.

Solution : Pour surmonter cette difficulté, j'ai développé des scripts supplémentaires pour désactiver temporairement les modules non conformes. Ces scripts utilisaient des commandes pour automatiser la correction des configurations échouées. Parallèlement, j'ai effectué des vérifications manuelles des configurations problématiques pour identifier les causes profondes des échecs et ajuster les scripts en conséquence. Cette approche hybride d'automatisation et de vérification manuelle a permis d'améliorer progressivement le score de conformité et de s'approcher de l'objectif de 100%.

4.4 Présentation des résultats obtenus et contributions du stagiaire

Au cours de mon stage chez Edenred, j'ai contribué de manière significative à plusieurs aspects du projet de renforcement de la sécurité de l'infrastructure informatique. Voici une présentation détaillée des résultats obtenus et de mes contributions principales.

Une migration possible vers Debian 12 sécurisé : la migration de CentOS 7 à Debian 12 peut être fait dans un environnement durci, permettant de se conformer aux nouvelles normes de sécurité et de répondre à la fin de vie de CentOS. Cette migration sera assurée par une meilleure prise en charge des fonctionnalités de sécurité modernes et a aligner l'infrastructure sur les standards actuels.

Implémentation des recommandations CIS : les recommandations de sécurité du CIS pour Debian Linux 12 ont été appliquées systématiquement sur l'ensemble du parc informatique. L'application de ces recommandations renforce la sécurité des systèmes, réduisant les vulnérabilités et augmentant la résilience contre les cyberattaques.

Automatisation des configurations de sécurité : les configurations de sécurité ont été automatisées à l'aide de *playbooks* Ansible, assurant une application cohérente et répétable. L'automatisation a amélioré l'efficacité opérationnelle, réduit les erreurs humaines et permis une gestion plus facile des configurations à grande échelle. J'ai développé et modifié des scripts Ansible pour adapter les recommandations CIS aux spécificités de l'infrastructure d'Edenred. Ces scripts ont permis une application flexible et contextuelle des configurations de sécurité, assurant une protection optimale tout en respectant les particularités de l'infrastructure.

Score de conformité amélioré : grâce aux scripts supplémentaires et aux vérifications manuelles, le score de conformité aux recommandations CIS a été considérablement amélioré. Un score de conformité élevé garantit que l'infrastructure répond aux standards de sécurité les plus stricts, protégeant ainsi les données sensibles et les opérations critiques. C'est au total plus de 10% de la sécurité qui est améliorée selon le benchmark d'OVH.

Documentation et transfert de connaissances : j'ai rédigé des guides et des tutoriels détaillés sur les procédures et les configurations mises en place, et j'ai mis à jour les documents de configuration dans Confluence. La documentation claire et détaillée a facilité la compréhension et la reproduction des configurations par les autres membres de l'équipe, assurant une continuité opérationnelle.

5 Conclusion

Mon stage chez Edenred, réalisé dans le cadre de la fin de la deuxième année du BUT R&T, parcours cybersécurité, a été une expérience très enrichissante et formatrice.

Durant ce stage, j'ai travaillé sur le projet de durcissement de l'infrastructure informatique en utilisant Ansible pour mettre en œuvre les recommandations CIS sur des systèmes Debian 12. Sous la supervision de Guillaume Bory, responsable de l'équipe Cloud, j'ai contribué à plusieurs aspects du projet, améliorant la sécurité et l'efficacité de l'infrastructure d'Edenred.

Les principaux objectifs de mon stage étaient de durcir Debian 12, en réponse à la fin de vie annoncée de CentOS, et d'automatiser les configurations de sécurité à l'aide de *playbooks* Ansible. L'automatisation des configurations de sécurité a permis de réduire les erreurs humaines et d'assurer une application uniforme des configurations. L'implémentation de pipelines CI/CD dans GitLab a permis de valider les modifications de code avant leur déploiement en production, garantissant une stabilité accrue et une gestion plus efficace des configurations. De plus, grâce aux scripts supplémentaires et aux vérifications manuelles, le score de conformité aux recommandations CIS a été considérablement amélioré. J'ai également rédigé un guide et une documentation détaillés dans Confluence, facilitant ainsi la compréhension et la reproduction des configurations par les autres membres de l'équipe.

Bien que la majorité des objectifs aient été atteints, certains défis subsistent. La recommandation CIS faisant plus de 1000 pages, il était compliqué pour moi de traiter tous les points pour atteindre un score de conformité de 100%. De plus, ce score ne pouvait pas être atteint car la recommandation ne prend pas en compte la configuration d'origine des serveurs. Les perspectives d'évolution incluent l'automatisation complète de la vérification de conformité CIS, ainsi que l'intégration de nouvelles technologies de sécurité pour anticiper les futures menaces.

Ce stage m'a permis d'acquérir des compétences techniques précieuses en administration système, en automatisation avec Ansible, et en sécurité informatique. J'ai également développé une meilleure compréhension de la gestion de projets en environnement professionnel et de l'importance de la documentation. La formation reçue à l'IUT a été fondamentale pour aborder les aspects techniques et théoriques du stage.

6 Remerciements

Je remercie grandement ma tante Céline Dipietrangelo sans qui ce stage n’aurait pas pu se réaliser.

Je tiens à exprimer ma gratitude à toutes les personnes qui ont contribué à la réussite de mon stage au sein de l'équipe d'Edenred France, à Saint-Martin-d'Hères. Leur soutien, leurs conseils et leur expertise ont été essentiels à l'aboutissement de mon projet et à mon développement professionnel.

Un immense merci à Loïc Bertorello, Vincent Jacquin, Jérôme Glenat, Emilie Coelho, Aurélien Muscio. Merci également à Ludovic Brun pour son expertise en Ansible qui m'a grandement aidé sur plusieurs sujets. Maxime Ferrieux, merci pour ta présentation de Terraform. Guillaume Bory, merci pour tes conseils avisés et ta méthodologie de travail. Et enfin, un grand merci à Emmanuel Banvillet. Tu as pris le temps de m'expliquer l'ensemble des outils, le fonctionnement de l'entreprise, les bonnes pratiques et même des bons plans à Grenoble. Ton aide a été inestimable, tant sur le plan professionnel que personnel.

Je remercie chaleureusement toute l'équipe pour leur accueil, leur collaboration et l'environnement de travail positif et motivant que vous avez su créer. Travailler avec vous a été une expérience extrêmement enrichissante et agréable. Grâce à votre soutien, j'ai pu acquérir de nouvelles compétences et vivre une très bonne expérience professionnelle.

Merci à toutes pour cette opportunité !

7 Glossaire

Hardening : Processus de sécurisation d'un système en réduisant sa surface d'attaque (durcissement)
CIS : (Center for Internet Security) Un ensemble de pratiques et de recommandations de sécurité
CI : Continuous Integration
CD : Continuous Deployment
BUT : Bachelor Universitaire de Technologie
IUT : Institut Universitaire de Technologie
R&T : Réseaux et Télécommunications
SSH : Secure Shell
DevOps : Approche culturelle et technique qui fusionne le développement logiciel (Dev) et les opérations informatiques (Ops) pour améliorer la collaboration, l'efficacité et la qualité des logiciels.
AWX : AWX est une interface utilisateur web open source pour Ansible
YAML : YAML (Yet Another Markup Language) est un format de sérialisation de données
MR : Merge Request
JIRA : Outil de gestion de projet et de suivi des tickets développé par Atlassian
Confluence : Outil de collaboration et de gestion de documentation développé par Atlassian
Ansible : Plateforme logicielle libre pour la configuration et la gestion des ordinateurs
AWX : Interface web pour gérer et automatiser des déploiements Ansible
Branche : Version du code source pour le développement sans affecter la branche principale
Merge request : demande de fusion d'une branche de code dans une autre
Master : La branche master est la branche principale d'un dépôt Git

Commit : Un commit est une opération Git qui enregistre des modifications dans un dépôt

8 Bibliographie

CIS (2024). CIS Debian Linux 12 Benchmark. Center for Internet Security.

GitLab (2024). GitLab CI/CD Documentation. GitLab.

Ansible (2024). Ansible Documentation. Ansible.

Red Hat (December 8, 2020). CentOS Linux 8 End-of-Life Announcement. Red Hat.

Edenred (2024). Internal Security Policies and Procedures. Edenred France.

OVH (2024). OVH CIS Benchmark Tool Documentation. OVH.