Cahier des Charges – Automatisation du Patch Management avec AWX et Ansible

Introduction

Dans un environnement informatique en constante évolution, la gestion des mises à jour (**patch management**) est une composante essentielle de la sécurité et de la stabilité des infrastructures IT. Un système non mis à jour est une cible de choix pour les cyberattaques et peut être source d'instabilité, mettant en péril la disponibilité des services et la conformité réglementaire.

Face à cette problématique, il devient impératif d'adopter une solution d'automatisation robuste et centralisée pour garantir l'application régulière et contrôlée des correctifs sur l'ensemble des plateformes : **serveurs Linux et Windows, équipements réseau, et infrastructures virtualisées (ESXi, OpenStack, Kubernetes, Proxmox).**

Ce projet vise à concevoir et déployer une **solution de gestion automatisée des correctifs**, reposant sur **AWX et Ansible**, permettant d'orchestrer les mises à jour de manière sécurisée, efficace et traçable. L'objectif est de proposer une **architecture flexible et évolutive**, capable de s'intégrer dans des environnements hétérogènes tout en répondant aux exigences de sécurité et de conformité.

Conception et Déploiement de la Solution

L'architecture technique repose sur plusieurs composants interconnectés qui assureront une gestion centralisée des mises à jour. AWX, qui est l'interface web open-source pour Ansible, sera déployé sur une instance **Kubernetes Single Node**, garantissant ainsi flexibilité et scalabilité. Cette instance facilitera l'exécution et la gestion des **playbooks Ansible**, scripts automatisant les tâches de mise à jour.

Le stockage des correctifs sera assuré par un **référentiel Linux** capable d'héberger les mises à jour nécessaires pour **les systèmes Linux, Windows et les équipements réseau**. Ce référentiel servira de dépôt centralisé, évitant ainsi la dépendance à des sources externes et garantissant une distribution rapide et contrôlée des correctifs.

Pour gérer et versionner les **playbooks Ansible**, un **dépôt Git** sera mis en place avec **Gitea**, une alternative légère et auto-hébergée à GitLab ou GitHub. Gitea facilitera la collaboration entre les administrateurs et assurera un suivi précis des modifications et des mises à jour des scripts d'automatisation. Il sera déployé sur une machine virtuelle dédiée, optimisée pour l'exécution de Docker, simplifiant ainsi son installation et sa maintenance.

L'un des défis majeurs du projet réside dans la gestion de l'**authentification et des permissions**. Pour garantir un accès sécurisé aux serveurs Windows, AWX sera **intégré à**

Active Directory (AD), permettant ainsi une gestion centralisée des droits d'accès et une authentification sécurisée via les comptes existants. L'inventaire des machines et équipements à mettre à jour sera géré dynamiquement grâce aux **modules d'inventaire dynamique d'Ansible**, permettant ainsi une détection automatique des hôtes et un suivi en temps réel de leur état.

Infrastructure Technique et Déploiement

L'ensemble de la solution sera déployé dans un **environnement virtualisé basé sur Proxmox**, garantissant un isolement optimal des composants et facilitant les tests avant un déploiement en production. Chaque machine virtuelle aura un rôle bien défini, optimisant ainsi la gestion des ressources et simplifiant les opérations de maintenance.

Le **serveur AWX**, installé sur une machine virtuelle sous **Ubuntu 22.04**, constituera le cœur du système. Il sera responsable de l'exécution des tâches Ansible et de la coordination des mises à jour. L'orchestration d'AWX sera assurée par **Kubernetes**, offrant ainsi une flexibilité accrue en matière de déploiement et de scalabilité. Une base de données PostgreSQL stockera l'ensemble des configurations et des historiques d'exécution.

Le **référentiel des correctifs** sera hébergé sur une autre instance Linux, dotée d'un espace de stockage conséquent pour conserver les mises à jour nécessaires aux différentes plateformes. L'accès se fera via HTTP, FTP et Samba, permettant ainsi une distribution fluide des fichiers vers les systèmes concernés.

Quant à **Gitea**, il sera installé sur une machine virtuelle distincte, hébergeant un serveur Docker, ce qui simplifiera son déploiement et sa mise à jour. Son rôle principal sera de stocker les **playbooks Ansible**, facilitant ainsi la gestion des versions et la collaboration entre les administrateurs système.

Enfin, un **contrôleur Active Directory sous Windows Server 2019** sera mis en place pour gérer l'authentification des serveurs Windows et centraliser les droits d'accès. Cette intégration permettra une meilleure traçabilité des actions effectuées et renforcera la sécurité de l'infrastructure.

Automatisation et Workflows d'Exécution

L'automatisation des mises à jour s'appuiera sur des **workflows spécifiques** pour chaque type de plateforme, garantissant ainsi une gestion optimisée et adaptée aux particularités de chaque environnement.

Mise à Jour des Serveurs Linux

Les serveurs Linux seront mis à jour via Ansible en utilisant les gestionnaires de paquets natifs (`apt`, `dnf`, `yum`). Le processus commencera par une détection des mises à jour disponibles, suivie d'une vérification des dépendances et d'un contrôle de compatibilité. Les correctifs

seront ensuite appliqués de manière contrôlée, avec une option permettant de planifier les redémarrages en fonction des contraintes de production.

Mise à Jour des Serveurs Windows

La gestion des correctifs Windows s'effectuera via **WinRM**, permettant une exécution distante sécurisée. Après authentification via Active Directory, les mises à jour seront téléchargées depuis le référentiel interne, puis appliquées selon un calendrier défini. Un mécanisme de reporting assurera un suivi précis des mises à jour installées et des éventuelles erreurs rencontrées.

Mise à Jour des Équipements Réseau

Les équipements réseau, tels que les routeurs et switches Cisco, nécessitent une approche différente. Une première phase consistera à sauvegarder la configuration actuelle des équipements, afin de pouvoir effectuer un rollback en cas de problème. Ensuite, les nouvelles images seront téléchargées et appliquées, avec une vérification post-déploiement garantissant le bon fonctionnement des équipements après la mise à jour.

Mise à Jour des Plateformes Virtualisées (ESXi, OpenStack, Kubernetes, Proxmox)

L'approche ici consistera à utiliser les API de chaque plateforme pour orchestrer les mises à jour de manière non intrusive. Dans le cas d'ESXi, par exemple, les hôtes seront mis en mode maintenance avant l'application des correctifs, et un mécanisme de test post-mise à jour sera mis en place pour éviter toute interruption de service.

Sécurité et Contrôle d'Accès

L'aspect **sécurité** sera primordial dans ce projet, en garantissant une gestion stricte des accès et une traçabilité complète des actions effectuées. L'authentification centralisée via **Active Directory** renforcera le contrôle d'accès, et AWX permettra d'assigner des **rôles spécifiques** aux administrateurs, limitant ainsi les privilèges à ce qui est strictement nécessaire.

Les connexions SSH et WinRM seront sécurisées avec une gestion rigoureuse des clés et certificats. De plus, un système de **sauvegarde automatique des configurations** garantira une récupération rapide en cas d'incident.

Déploiement et Configuration des Machines Virtuelles

Chaque VM sera dimensionnée en fonction de son rôle, garantissant une répartition équilibrée des ressources.

Serveur AWX (Ansible Tower Open Source)

- **OS :** Ubuntu 22.04 LTS
- **CPU :** 8 vCPU
- **RAM :** 16 Go
- **Stockage :** 100 Go SSD
- **Rôle :**
  - Hébergement d'AWX pour l'orchestration des mises à jour
  - Exécution des **playbooks Ansible**
  - Interface web pour la gestion et le monitoring des tâches
  - Intégration avec Active Directory (authentification et gestion des droits)
- **Déploiement d'AWX via Kubernetes Single Node** :
  - **Kubernetes installé avec kubeadm**
  - Utilisation d'un **Ingress Controller** pour l'accès web sécurisé
  - Base de données **PostgreSQL 14** hébergée en local (VM)
  - AWX Operator pour le déploiement et la gestion des mises à jour

Serveur Gitea (Dépôt Git Auto-hébergé)

- **OS :** Ubuntu 22.04 LTS
- **CPU :** 4 vCPU
- **RAM :** 8 Go
- **Stockage :** 200 Go SSD (stockage des playbooks et logs)
- **Rôle :**
  - Gestion des **playbooks Ansible**
  - Versionnement des scripts et des fichiers de configuration
  - Intégration avec AWX pour une exécution automatisée des tâches
- **Déploiement de Gitea via Docker** :
  - Gitea exécuté dans un conteneur Docker
  - Base de données **MariaDB** ou **PostgreSQL** pour la persistance des données
  - Exposition via **Nginx Reverse Proxy** sécurisé avec **Let's Encrypt**

Référentiel des Correctifs (Serveur Linux)

- **OS :** Debian 12
- **CPU :** 4 vCPU
- **RAM :** 16 Go
- **Stockage :** 2 To HDD (stockage des mises à jour)

- **Rôle :**
  - Hébergement des **patchs Linux, Windows, ESXi, équipements réseau (Cisco, Juniper, etc.)**
  - Accès via **HTTP, FTP et SMB** pour une distribution rapide
  - Synchronisation automatique avec les sources officielles et stockage des mises à jour validées
- **Configuration spécifique :**
  - Serveur **Apache/Nginx** pour l'accès web
  - Service **Samba** pour le déploiement des correctifs Windows
  - Mirroring des dépôts officiels (Ubuntu, CentOS, Windows Update Catalog)

Contrôleur Active Directory (Authentification Windows)

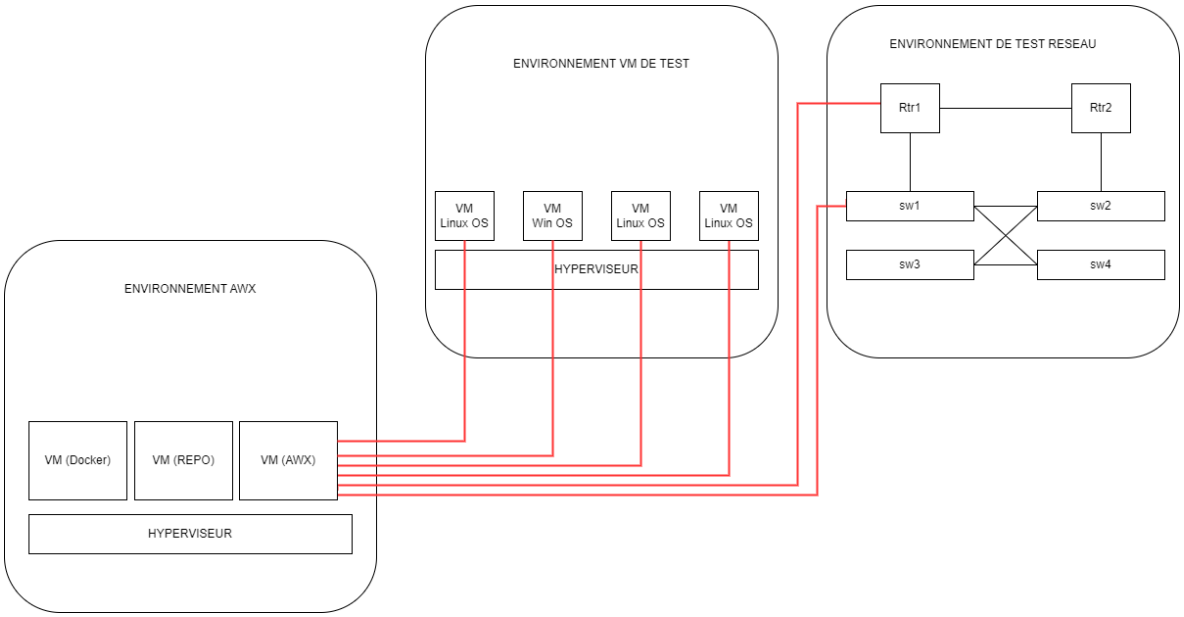
- **OS :** Windows Server 2019
- **CPU :** 4 vCPU
- **RAM :** 8 Go
- **Stockage :** 100 Go SSD
- **Rôle :**
  - Gestion des comptes et des groupes pour l'authentification sur AWX
  - Centralisation des droits d'accès pour les serveurs Windows
  - Intégration avec AWX via Kerberos/WinRM

Serveurs de Test pour l'Application des Patches

Pour valider les mises à jour avant leur déploiement en production, des serveurs de test seront configurés :

- **Serveur Linux Test :** Ubuntu 22.04, 2 vCPU, 4 Go RAM, 40 Go SSD
- **Serveur Windows Test :** Windows Server 2019, 2 vCPU, 4 Go RAM, 60 Go SSD
- **Équipement Réseau Test :** GNS3 avec des images Cisco/Juniper pour tester les mises à jour des firmwares

Schéma de principe :



Présentation du Bootcamp

Bootcamp Intensif : Automatisation du Patch Management sur Infrastructure Hybride

Ce programme intensif de trois mois s'adresse aux professionnels de l'informatique souhaitant acquérir une expertise approfondie dans l'automatisation du patch management sur des infrastructures hétérogènes. Il couvre l'ensemble des processus nécessaires à la gestion efficace des mises à jour sur des environnements variés comprenant des serveurs Linux et Windows, des équipements réseau ainsi que des plateformes cloud et de virtualisation telles que Kubernetes, OpenStack et Proxmox.

L'objectif est d'apporter aux participants une méthodologie rigoureuse et des compétences pratiques pour concevoir, déployer et administrer une solution d'automatisation basée sur **AWX** (version open-source d'Ansible Tower), **Ansible** et **Gitea** pour la gestion centralisée des configurations et des workflows d'automatisation. Une attention particulière sera portée à l'intégration des référentiels de correctifs, à l'inventaire dynamique et à la supervision des opérations via des outils de monitoring et d'audit.

À l'issue de ce bootcamp, chaque participant sera en mesure de concevoir et mettre en place un **environnement de gestion automatisée des mises à jour**, incluant :

- **L'installation et la configuration complète d'une infrastructure sur Proxmox** en vue d'héberger les différents services nécessaires à l'automatisation du patch management.

- **L’utilisation avancée d’Ansible et AWX** pour orchestrer les mises à jour des serveurs et équipements réseau, en tenant compte des meilleures pratiques en matière de sécurité et de fiabilité.
- **La mise en place d’un référentiel Git (Gitea)** afin de versionner et centraliser les playbooks Ansible, garantissant ainsi une gestion cohérente et auditable des configurations.
- **Le déploiement d’un référentiel centralisé de correctifs** pour stocker et distribuer les mises à jour des systèmes Linux, Windows et des équipements réseau.
- **L’implémentation de workflows automatisés** couvrant divers environnements technologiques, qu’il s’agisse de serveurs classiques, d’infrastructures cloud ou de plateformes de virtualisation comme Kubernetes et OpenStack.
- **La supervision et la sécurisation de l’ensemble des opérations** grâce à des outils d’audit et de monitoring permettant de garantir la conformité des mises à jour et d’anticiper les éventuelles anomalies.

Prérequis et Compétences Requises

Afin de maximiser l’efficacité de la formation et garantir une progression fluide, il est recommandé aux participants de posséder un certain niveau de connaissances dans les domaines suivants :

1. Maîtrise des systèmes d’exploitation

Une expérience préalable avec **Linux** (Debian/Ubuntu, Red Hat/CentOS) est un atout majeur. La compréhension de la ligne de commande, la gestion des packages ainsi que la gestion des permissions et des services système sont essentielles pour naviguer efficacement dans l’environnement Unix. De même, des notions de base sur **Windows Server** (versions 2016/2019), notamment sur **Active Directory**, **WinRM** et la gestion des mises à jour via WSUS ou SCCM, permettront d’intégrer les serveurs Windows dans le processus d’automatisation.

2. Compréhension des concepts réseaux et de la virtualisation

Les participants doivent être à l’aise avec les notions fondamentales de **TCP/IP, DNS, DHCP, VLAN et SSH**, qui sont essentielles pour assurer la communication et l’administration à distance des machines cibles. Une première expérience avec des hyperviseurs comme **Proxmox, VMware ou VirtualBox/KVM** est fortement conseillée, notamment pour la gestion des ressources virtuelles. Par ailleurs, une connaissance des **équipements réseau Cisco, Juniper, Mikrotik ou Fortinet** facilitera l’automatisation des mises à jour et la gestion des configurations.

3. Notions de DevOps et d’automatisation

La maîtrise des bases d’**Ansible** est un atout important, notamment la compréhension des **playbooks, rôles et modules**. Une familiarité avec **Git, GitHub ou GitLab** est également indispensable pour le stockage et la gestion du code d’automatisation. De plus, des notions sur **Docker et Kubernetes**, en particulier la gestion des conteneurs et le déploiement d’applications, seront essentielles pour comprendre le fonctionnement et l’installation d’AWX sur un cluster Kubernetes.

4. Aptitudes en scripting et en gestion des mises à jour

Une bonne capacité à écrire et comprendre des scripts en **Bash, PowerShell ou Python** sera un atout considérable pour personnaliser et adapter les processus d'automatisation. De plus, une compréhension des enjeux liés aux **mises à jour de sécurité et à la gestion des vulnérabilités** est essentielle pour assurer un patch management efficace et conforme aux exigences des entreprises.

Un Accompagnement Progressif pour Assurer une Montée en Compétences

Ce bootcamp est conçu pour accompagner les participants, quel que soit leur niveau initial, avec une **mise à niveau progressive** sur les compétences fondamentales en début de formation. Ainsi, même si certaines notions restent floues au départ, le programme permet d'acquérir rapidement les bases nécessaires avant d'aborder des sujets plus avancés.

L'approche pédagogique alterne **cours théoriques, démonstrations pratiques, exercices encadrés et projets concrets**, garantissant ainsi une **assimilation progressive et efficace des compétences**. L'objectif est que chaque participant puisse, à l'issue du programme, **mettre en œuvre de manière autonome une infrastructure de patch management automatisée et évolutive**.

Ce bootcamp représente une opportunité unique pour les professionnels de l'IT souhaitant développer leur expertise en **automatisation, DevOps et gestion des mises à jour à grande échelle**. La demande pour ces compétences est en forte croissance, et cette formation vous positionnera comme un acteur clé dans l'administration et la sécurisation des infrastructures modernes.

Ce **programme intensif de trois mois** est conçu pour **les administrateurs systèmes** souhaitant maîtriser l'**automatisation du patch management** dans des environnements **Linux et Windows**. Il couvre **Proxmox, Docker, Gitea, Kubernetes, Ansible et AWX**, permettant aux participants de mettre en place un **système automatisé de gestion des mises à jour**.

Chaque étudiant bénéficie de 12 heures de coaching personnalisé réparties sur trois mois (**4 heures par mois**), garantissant un suivi rigoureux et un accompagnement technique structuré.

Mois 1 : Mise en place de l'infrastructure et des outils fondamentaux

Ce premier mois est consacré à l'installation et la configuration des **outils de virtualisation et de gestion des conteneurs**, qui constitueront la base du projet.

Semaine 1 – Installation et configuration de Proxmox

🕮 Chapitres à suivre :

- **Installation et mise à jour de Proxmox**
  - 2.1 Installation de Proxmox (5 min)
  - 2.2 Correction installation de Proxmox (15 min)
  - 2.3 Mise à jour de Proxmox (10 min)
  - 2.5 Découverte de l'interface d'administration (15 min)
- **Configuration réseau minimale pour les VMs**
  - 3.1 Le réseau de type bridge (10 min)

🛠 Exercice pratique :

- Installer **Proxmox** sur un serveur ou une VM
- Configurer un **réseau Bridge** pour assurer la connectivité des VMs

---

Semaine 2 – Création et administration des machines virtuelles

🕮 Chapitres à suivre :

- **Création des machines virtuelles Linux et Windows**
  - 6.1 Créer une machine virtuelle sur Proxmox (3 min)
  - 6.2 Créer une machine virtuelle avec KVM (22 min)
- **Gestion du stockage des VMs**
  - 4.1 Les volumes LVM (Volume Group - VG) (18 min)
  - 4.2 LVM Thin : gestion avancée du stockage (12 min)

🛠 Exercice pratique :

- Créer les machines virtuelles suivantes :
  - **VM Linux pour AWX**
  - **VM Linux pour le référentiel de patchs**
  - **VM Linux pour Gitea (installer docker et déploie Gitea avec docker-compose)**
  - **VMS Windows pour tester le patch management**

🔑 Bonus : Authentification et snapshots (si nécessaire)

- **Intégration avec Active Directory (pour AWX & Windows)**
  - 5.3 Connecter un Active Directory à Proxmox (11 min)
- **Gestion des snapshots et sauvegarde**
  - 9.5 Gestion des snapshots (8 min)

🛠 Exercice pratique :

- Créer **des snapshots avant l'installation d'AWX et Gitea**

Semaine 3 – Administration système sous Linux et Windows Server

🕮 Chapitres à suivre :

https://teachmemore.fr/courses/administrer-ubuntu/

https://teachmemore.fr/courses/windows-server-lessentiel/

- **Administration Linux**
  - 2.1 Découvrir le shell de Linux (7 min)
  - 2.2 Les différents répertoires racines (18 min)
  - 2.3 Les commandes de base de Linux (20 min)
  - 2.5 Commandes "LS, MKDIR, CD, PWD" (31 min)
  - 2.6 Commandes "LN, CAT, LESS, MORE" (23 min)
  - 2.7 Rechercher des fichiers avec FIND (16 min)
  - 2.9 Rechercher des chaînes avec GREP (7 min)
  - 2.14 Gestion des permissions avec CHMOD (17 min)

Exercice : Créer le référentiel de dépôt Linux pour stocker les binaires et les paques !

- **Installation et configuration de Windows Server**
  - Installation de Windows Server (13 min)
  - Explication du rôle d'un **contrôleur de domaine** (7 min)
  - Installation et configuration d'**Active Directory**
  - Création et gestion des **unités organisationnelles et groupes** (7 min)
  - Creation des utilisateurs

🛠 Exercice pratique :

- Manipulation des **commandes Linux essentielles** sur un serveur Ubuntu
- Installation et configuration d'un **contrôleur de domaine Windows**
- Connexion d'une machine Windows au **domaine Active Directory**

Semaine 4 – Conteneurisation avec Docker et gestion du code avec Gitea

🕮 Chapitres à suivre :

https://teachmemore.fr/courses/docker-lessentiel-pour-les-sysadmin-et-reseaux/

- **Introduction à Docker**
  - C'est quoi Docker ? (27 min)
  - Installer Docker (10 min)
- **Gestion des conteneurs et stockage**
  - Manipuler les images Docker (14 min)
  - Créer et gérer des conteneurs (26 min + 6 min)

    - Supprimer des conteneurs (6 min)
    - Manipuler les volumes (34 min)
- **Docker Compose et gestion avancée**
    - Explication de Docker Compose (11 min)
    - Déploiement d'une application avec Docker Compose (35 min)
- **Gitea et gestion du code**
    - Installation de Gitea sur Docker
    - Création et gestion des **dépôts Git**

🛠 Exercice pratique :

- Déployer un **serveur Gitea avec Docker Compose**
- Créer un **dépôt Git et versionner les playbooks Ansible**

Mois 2 : Automatisation avec Ansible et AWX (ansible pour les réseaux et ansible pour Linux)

Ce deuxième mois se concentre sur la mise en place de l'**automatisation des patchs** avec **Ansible et AWX**.

Introduction à Kubernetes.

https://teachmemore.fr/courses/kubernetes-2023-formation-complete-pour-devops-admins/

Reviser la partie 1 à 7 sauf la partie 5 (Maxmimum 2 à 3 semaines)

Semaine 5 – Introduction à Ansible et installation d'AWX

Réviser tous ce cours : https://teachmemore.fr/courses/ansible-pour-les-administrateurs-systemes-linux/ (deux semaines)

🛠 Exercice pratique :

- Installer **AWX avec Kubernetes**
- Exécuter un **premier playbook Ansible**

Semaine 6 – Gestion des inventaires et intégration avec Gitea

📖 **Chapitres à suivre :**

- Création et gestion des **inventaires sous AWX**
- Synchronisation des **playbooks avec Gitea**
- Gestion des **droits et utilisateurs sous AWX**

🛠 **Exercice pratique :**

- Configurer un **inventaire dynamique des équipements réseau**

- Création de **Job Templates sous AWX**
- Automatisation des **mises à jour des routeurs et des switchs y compris des configurations**
- Exécution des **workflows d'automatisation**

- Déploiement des **patchs automatisés sur plusieurs machines**

- Sécurisation des **communications avec HTTPS et certificats TLS (traefik pour reverse proxy)**
- Intégration d'**Active Directory avec AWX**

🛠 **Exercice pratique :**

- Configurer l'**authentification AD-DS sous AWX**

Mois 3 : Validation du projet et perfectionnement

- **Semaine 9 – Optimisation des workflows et gestion avancée des correctifs**
- **Semaine 10 – Résolution des problèmes et tests avancés**
- **Semaine 11 – Finalisation du projet et intégration des compétences**
- **Semaine 12 – Soutenance et validation du projet**

📌 **Objectif final :**
Déployer une infrastructure complète intégrant **Proxmox, AWX, Kubernetes et Gitea** pour **automatiser la gestion des correctifs** sur un parc de machines **Linux et Windows**. 🚀