

## Objectifs pédagogiques

À l'issue de ce TP, vous devez être capables de :

- Écrire un script de tests de non-régression pour un service web simple.
- Vérifier automatiquement la présence d'un paramètre de sécurité (en-tête HTTP).
- Intégrer ces tests dans un pipeline CI/CD.
- Comprendre comment un pipeline peut protéger contre des régressions fonctionnelles et sécurité.

## Contexte du TP

Vous travaillez dans l'équipe DevOps chez Corell Industries, petite entreprise de production de cargo interstellaire, qui édite une page web statique (par exemple index.html) est déployée sur une VM Azure avec un serveur web (Apache ou Nginx).

Votre responsable sécurité vous demande de **renforcer la configuration du serveur web** (durcissement) en ajoutant au moins un **paramètre de sécurité** (par exemple un en-tête HTTP de sécurité).

Problème : à chaque modification de configuration, il y a un risque de **régression** :

- la page peut devenir inaccessible (erreur 403/500),
- le contenu peut être modifié,
- ou le paramètre de sécurité peut ne pas être correctement appliqué.

Votre mission est donc de :

1. **Automatiser des tests de non-régression** pour vérifier que la page reste accessible et correcte.
2. **Ajouter des tests de sécurité** pour vérifier que le nouveau paramètre de sécurité est bien en place.
3. Intégrer ces tests dans un **pipeline CI/CD** (par exemple via GitHub Actions ou Azure DevOps) qui s'exécute automatiquement.

## Description simplifiée de l'architecture

- **VM Azure :**
  - OS : Linux (Ubuntu par exemple)
  - Serveur web : Apache ou Nginx
  - Application : simple page index.html (ex. "Bienvenue sur mon site sécurisé").
- **Dépôt Git :**
  - Contient au minimum :
    - le fichier index.html (ou un placeholder),
    - un répertoire tests/ où vous mettrez vos scripts de tests,
    - le fichier de configuration du pipeline (.github/workflows/\*.yml pour GitHub Actions, ou azure-pipelines.yml pour Azure DevOps).
- **Le pipeline CI que vous allez construire :**
  - sera déclenché à chaque push sur la branche,
  - exécutera vos tests de non-régression et vos tests de sécurité contre l'URL publique de la VM Azure.

## Ressources fournies :

### Préparation : définir la cible de test

1. Vérifiez que votre VM Azure est accessible
2. Dans votre dépôt Git, créez un fichier tests/config.sh

### Script de tests de non-régression (fonctionnels)

1. Créez un script tests/check\_functional.sh qui vérifie au minimum :
  - o que la page principale renvoie un **code HTTP 200**,
  - o que le **contenu** contient une chaîne attendue (ex : "Hello World").
2. Vérifiez que le script fonctionne **en local** (depuis votre machine) :

### Modification d'un paramètre de sécurité sur le serveur web

1. Sur la VM Azure, modifiez la configuration de votre serveur web pour ajouter au moins **un en-tête de sécurité HTTP**.  
Par exemple :
  - o X-Frame-Options: DENY
  - o X-Content-Type-Options: nosniff
  - o Content-Security-Policy: default-src 'self';
2. Vérifiez manuellement que :
  - o la page est toujours accessible,
  - o l'en-tête de sécurité ajouté est bien présent dans la réponse.

### Script de tests de sécurité (non-régression sécurité)

1. Créez un script tests/check\_security.sh qui vérifie :
  - o que la page renvoie toujours **200**,
  - o que l'en-tête de sécurité attendu est bien présent dans la réponse,
  - o éventuellement que sa **valeur** est correcte (ex. DENY pour X-Frame-Options).
2. Vérifiez que ce script fonctionne en local et qu'il **échoue** si vous modifiez ou supprimez l'en-tête sur le serveur.

---

## Intégration dans un pipeline CI/CD

### Choix de la plateforme CI/CD :

- o Si vous utilisez **GitHub** : créez un fichier .github/workflows/security-regression.yml.
- o Si vous utilisez **Azure DevOps** : créez un fichier azure-pipelines.yml.

### Le pipeline doit :

- être déclenché au moins sur les **push** sur la branche principale,
- exécuter une machine Linux (Ubuntu),
- exécuter les deux scripts :
  - o tests/check\_functional.sh
  - o tests/check\_security.sh.