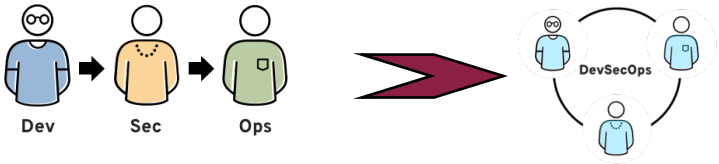
POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

LOG8100: DEVSECOPS - OPÉRATIONS ET DÉV. LOGICIEL SÉCUR

Titre du Projet : Intégration d'un pipeline de déploiement sécurisée dans Kubernetes



Remarque:

Les travaux pratiques constituent une partie importante du cours de LOG8100 et ont pour objectif de vous permettre d'appliquer les notions et les **principes de DevSecOps** étudiés dans les séances de cours.

Les travaux pratiques vous permettront d'élaborer des stratégies de test et d'utiliser les différents outils disponibles pour analyser et évaluer la qualité des logiciels en termes de sécurité.
A travers les TP vous pouvez acquérir une expérience pratique précieuse pour comprendre et traiter les vulnérabilités de sécurité des applications dans un environnement sûr et contrôlé.

La note des 3 travaux pratiques et du projet représente 50% de la note globale du cours. Il est donc vivement recommandé de les aborder avec sérieux, en faisant appel à votre créativité et à votre esprit critique, afin de les réussir au mieux

Planification des tâches et dates de remise :
Ce travail comporte trois parties, la répartition des tâches et les dates de remise seront les suivantes :

	Tâches du projet	Pondération	Date de remise
Partie 1	Planification et configuration de l'infrastructure	10%	**7 décembre, 2023, 23:59**
Partie 2	Analyse de sécurité et déploiement de l'application sur un le cluster K8S	10%	

Partie 3	Surveillance, autocorrection et mitigation des vulnérabilités	10%	

La collaboration avec vos collègues est permise pendant et en dehors des séances de laboratoire. Cependant, veuillez noter que les règles concernant le plagiat restent en vigueur en tout temps

Il est recommandé d'utiliser les outils mentionnés dans l'énoncé, mais vous avez également la possibilité d'explorer d'autres outils.

Objectif du TP2 :
Le projet DevSecOps consiste à mettre en place un pipeline d'intégration et de déploiement continus (CI/CD) pour une application OWASP dans un environnement Kubernetes (K8S). L'objectif est de garantir la sécurité tout au long du processus de développement et de déploiement et que l'application déployée sur un cluster Kubernetes avec un minimum de vulnérabilités de sécurité.

Les étapes principales du projet sont les suivantes :

1. Configuration d'un cluster Kubernetes (K8S) à l'aide d'Ansible et Terraform sur deux machines virtuelles. Vous pouvez utiliser vagrant pour créer les VM https://app.vagrantup.com/centos/boxes/7 (os recommandé CentOS)
2. Effectuer une analyse et de sécurité Terraform (Terraform TFLint, Checkov, Terrascan, Terraform fmt)
3. Mise en place d'un GitLab runner pour automatiser les builds.
4. Sélectionner une application OWASP hors ligne.
5. Effectuer l'analyse statique du code source qui évalue la qualité du code et identifie les problèmes de sécurité potentiels (SonarQube, OWASP ZAP)
6. Créer une image Docker de l'application OWASP sélectionnée hors ligne et publier l'image de l'application sur Docker Hub.
7. Déployer l'application OWASP conteneurisée depuis Docker Hub sur le cluster K8S créé.
8. Effectuer des scans de sécurité pour l'application et les conteneurs dans le cluster en utilisant des outils de scan de conteneurs.
9. Intégrer des outils de surveillance tels que Grafana et Prometheus pour surveiller les performances et la sécurité.
10. Configurer des alertes sur Slack ou Teams en cas de détection de vulnérabilités.
11. Mettre en place un mécanisme d'autocorrection et de mitigation des vulnérabilités détectées.

Tâche	Outils
Configuration d'un cluster Kubernetes (K8S) à l'aide d'Ansible et de Terraform sur deux machines virtuelles	- Ansible - Terraform - Minikube (pour le développement et les tests locaux)
Effectuer une analyse et de sécurité Terraform	Terraform TFLint, Checkov, Terrascan, Terraform fmt
Mise en place d'un GitLab Runner pour automatiser les builds	- GitLab Runner
Sélectionner une application OWASP hors ligne	- OWASP WebGoat - OWASP Juice Shop
Créer une image Docker de l'application OWASP sélectionnée hors ligne et publier l'image de l'application sur Docker Hub	- Docker (outil principal) – Docker Hub (pour le stockage des images)
Déployer l'application OWASP containerisée depuis Docker Hub sur le cluster K8S créé	- Kubectl (outil de ligne de commande pour Kubernetes)

Effectuer des scans de sécurité pour l'application et les conteneurs dans le cluster en utilisant les outils suivants	- Clair (pour l'analyse de vulnérabilités des conteneurs) - Trivy (scanner de vulnérabilités des conteneurs) - OWASP ZAP (pour le test de sécurité des applications web)
Intégrer des outils de surveillance tels que Grafana et Prometheus pour surveiller les performances et la sécurité	- Grafana (pour la visualisation des données) - Prometheus (pour la collecte de métriques) - Kube-state-metrics (pour les métriques spécifiques à Kubernetes)
Configurer des alertes sur Slack ou Teams en cas de détection de vulnérabilités	- Slack (intégration de webhooks pour les alertes) - Microsoft Teams (intégration de webhooks pour les alertes)
Mettre en place un mécanisme d'autocorrection et de mitigation des vulnérabilités détectées	- GitLab CI/CD (pour automatiser les actions en réponse aux vulnérabilités) - Contrôleurs d'admission Kubernetes (pour automatiser les politiques de sécurité)

Livrables attendus :

1- Le code source de votre dépôt (Partager avec moi votre dépôt gitlab)
2- Un rapport PDF qui présente l'exécution des différentes étapes (stages), en expliquant en détail les étapes nécessaires (maximum 10 pages)

Références:

1. K8S:
    - https://kubernetes.io/docs/tasks/tools/
    - https://registry.terraform.io/providers/scott-the-programmer/minikube/latest/docs/resources/cluster
2. Ansible: https://docs.ansible.com/ansible/latest/index.html
3. Terraform: https://developer.hashicorp.com/terraform/tutorials/getting-started
4. GitLab runner : https://docs.gitlab.com/runner/install/
5. Bonne pratique pour la configuration de K8S: https://kubernetes.io/docs/concepts/configuration/overview/
6. Analyse des Conteneurs : Clair (Outil Clair sur GitHub)
7. Surveillance/Monitoring: https://grafana.com/docs/
8. Collecte de Métriques : https://prometheus.io/docs/introduction/overview/
9. Configuration des Alertes : https://prometheus.io/docs/alerting/latest/alertmanager/
10. Mécanisme d'Autocorrection et de Mitigation : https://kubernetes.io/docs/concepts/security/pod-security-admission/

Autres références :

1. **Minikube**:

Minikube is a popular tool for setting up a single-node Kubernetes cluster on your local machine. It's a lightweight option for learning and development.

2. **K3d (K3s in Docker):**

K3d is a tool that allows you to run K3s, a lightweight Kubernetes distribution, in Docker containers. It's a great choice for local testing and learning.

3. **Kind (Kubernetes in Docker):**

KinD is another tool that enables you to run Kubernetes clusters inside Docker containers. It's designed for testing, development, and CI/CD environments.

4. **MicroK8s:**

MicroK8s is a snap-based Kubernetes distribution for local development. It's easy to install and manage, making it suitable for learning and experimentation.

5. **Docker Desktop (with Kubernetes Enabled):**

If you're already using Docker Desktop, you can enable Kubernetes support in the settings. This allows you to run a Kubernetes cluster on your local machine.

6. **Vagrant with VirtualBox:**

You can use Vagrant to provision virtual machines with VirtualBox and then deploy a Kubernetes cluster on those virtual machines. This approach offers more flexibility but requires a bit more setup.

Partie 1 - Planification et Configuration de l'Infrastructure (10%) :
- Choisissez une application web simple.
- Mettez en place un dépôt de contrôle de version (par exemple, Git) pour l'application.
- Choisissez un outil CI/CD (par exemple, Jenkins, GitLab CI ou GitHub Actions) et créez un nouveau projet pour automatiser le processus de déploiement.
- Planifiez et créez un cluster Kubernetes en utilisant Terraform.
- Documentez les étapes du pipeline et la configuration du cluster Kubernetes dans un fichier README.md.

Partie 2 - Analyse de Sécurité et Déploiement (10%) :
- Intégrez un outil d'analyse de sécurité statique de l'application (SAST) dans le pipeline pour analyser le code.
- Intégrez un outil d'analyse de conteneurs dans le pipeline pour scanner les images Docker à la recherche de vulnérabilités.
- Configurez le pipeline pour exécuter automatiquement ces analyses de sécurité à des étapes appropriées.
- Déployez l'application sur le cluster Kubernetes en utilisant Ansible pour la gestion de la configuration.

Partie 2 - Automatisation de la Sécurité et Rapports (10%) :
- Mettez en place des contrôles de sécurité automatisés dans le pipeline pour empêcher le déploiement de code présentant des vulnérabilités de sécurité critiques.
- Intégrez un outil de rapports de sécurité pour générer des rapports de sécurité.
- Configurez le pipeline pour notifier l'équipe de développement en cas de découverte de vulnérabilités et pour échouer le pipeline en cas de problèmes critiques.
- Démontrez le pipeline en déployant l'application sur le cluster Kubernetes et montrez comment les vérifications de sécurité sont effectuées à différentes étapes.