DevSecOps
Rapport Final

Khadidja MEHDI – Antoine LE CALLOCH – Valentin NASONE

Introduction

Dans le cadre du cours de Security & Privacy dispensé par Christian Delettre, nous avons entrepris un projet DevSecOps visant à mettre en place un déploiement continu et une intégration continue pour une application PHP. Cette application, initialement conçue pour afficher une image et un secret, nous a servi de point de départ pour explorer et mettre en pratique les principes DevSecOps, de bout en bout, depuis la phase de développement jusqu'à l'environnement de production.

1. Défis rencontrés et solutions apportées

La transition vers une approche DevSecOps a représenté un de nos défis majeurs, surtout pour une équipe avec des membres ayant peu d'expérience préalable dans ce domaine. Pour surmonter cette difficulté, nous avons adopté une approche progressive. Nous nous sommes documentés pas à pas sur le concept et une des membres de l'équipe avec le plus de connaissances sur le sujet a partagé son expérience.

De plus, en tant qu'étudiants, nous avons été confrontés au défi de trouver des outils efficaces et peu coûteux, tout en étant faciles à intégrer dans notre pipeline CI/CD. Les outils que nous utilisons généralement en entreprise (GitLab CI, Vault, etc.) n'étaient pas disponibles en raison du coût. Nous avons donc dû apprendre à utiliser de nouveau outils open source, qui présentent parfois certaines limitations en termes de fonctionnalités. Il a été parfois compliqué de trouver des réponses à nos questions car la communauté de développeurs utilisant ces outils était assez restreinte.

Enfin, nous avons rencontré des différences de comportement entre l'exécution des commandes via la pipeline et sur nos machines locales, notamment en ce qui concerne l'évaluation incorrecte de certaines commandes via le remote SSH. Pour réussir à diagnostiquer les problèmes, il a fallu effectuer de nombreux tests via la pipeline, ce qui s'est avéré assez long. De plus, en créant un compte Circle CI nous disposons de crédits et donc d'un nombre de lancement de pipeline limité. Nous avons donc dû créer plusieurs comptes pour remédier à ce problème.

2. Pipeline

Voici des schémas qui récapitule du *workflow* de notre pipeline. Une description plus détaillé du fonctionnement de la pipeline est disponible dans le fichier **setup-project-guide.md**.

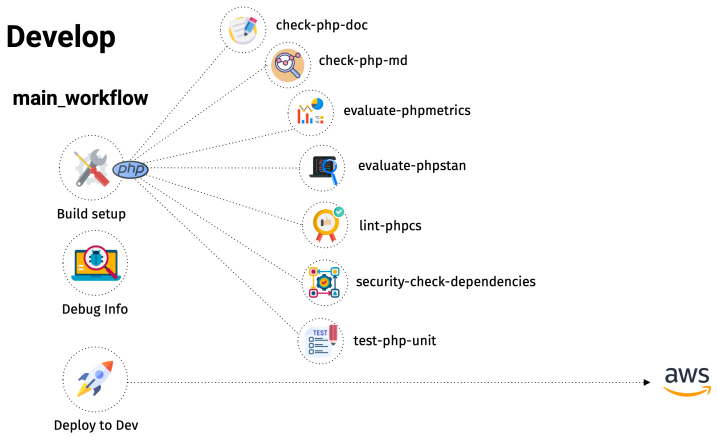

Figure 1 : Main workflow, Develop branch

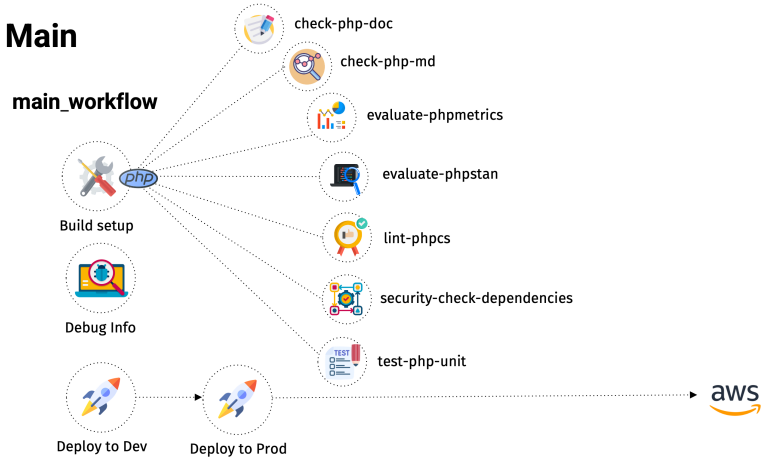

Figure 2 : Main workflow, Main branch

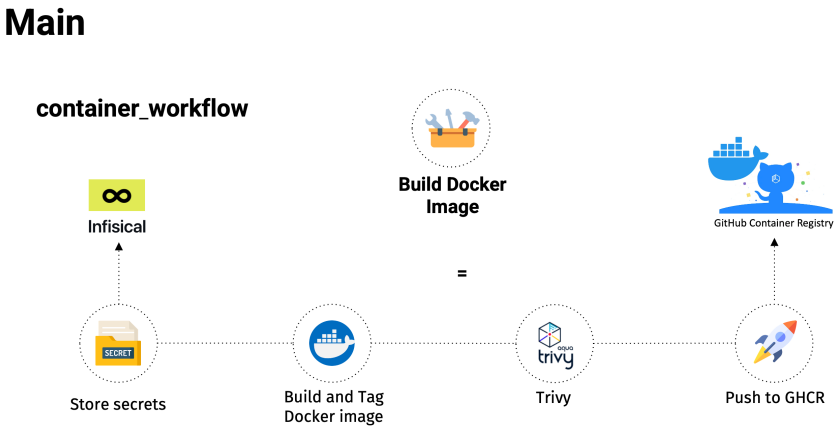

Figure 3: Container workflow, Main & Develop Branches

3. Sécurité

Configuration du Repository et Compte GitHub

Pour sécuriser notre repository GitHub, nous l'avons rendu privé et limité l'accès aux membres invités uniquement. De plus, nous avons protégé les branches et exigé des validations via des pull requests avant toute fusion de code.

Fiabilité de l'image Docker

Pour sécuriser nos images Docker, nous avons intégré Docker Trivy, un scanner de vulnérabilités efficace. Trivy analyse nos conteneurs et leurs dépendances, détectant les failles connues. Son intégration simple et sa base de données constamment mise à jour en font un outil idéal pour garantir la sécurité de nos déploiements Docker.

Renforcement de la sécurité de l'infrastructure

La sécurité de notre instance EC2 peut être renforcée de plusieurs manières pour garantir une protection optimale. Premièrement, plutôt que de s'appuyer sur des clés SSH pour le déploiement sur les différents environnements, nous pourrions envisager l'utilisation de comptes de service. Cette méthode offrirait une gestion des accès plus sécurisée et contrôlée. De plus, il est crucial de limiter l'accès aux ports de l'application, en particulier le port SSH, pour réduire la surface d'attaque potentielle. Un autre aspect important de la sécurisation de notre infrastructure est le scan régulier des instances avec AWS Security Scan, permettant de détecter et de corriger proactivement les vulnérabilités.

Concernant la protection contre les attaques OWASP et DDoS, nous utilisons pour l'instant des groupes de sécurité fonctionnent à un niveau similaire à celui d'un pare-feu de réseau, en contrôlant le trafic entrant et sortant basé sur des règles prédéfinies. Cependant, pour une protection plus spécifique contre les attaques applicatives et les DDoS, l'implémentation d'un pare-feu applicatif web (WAF) serait effectivement judicieuse. Un WAF peut offrir une couche supplémentaire de sécurité en filtrant, surveillant et bloquant le trafic HTTP vers et depuis une application web.

4. Conclusion

En conclusion, ce projet a été extrêmement enrichissant à plusieurs niveaux. Il nous a permis de découvrir et de maîtriser les principes du DevSecOps, ainsi que de nous familiariser avec de nouveaux outils et technologies. Cette expérience nous a également offert l'opportunité d'améliorer notre polyvalence et notre capacité à nous adapter à de nouveaux outils/environnements de travail.

Bien que nous soyons satisfaits des résultats obtenus, nous aurions aimé pouvoir explorer davantage certaines idées si nous avions disposé de plus de temps et moins de charge de

travail parallèle. Parmi les pistes d'amélioration envisagées, nous aurions souhaité mettre en place une infrastructure plus complexe, intégrant des éléments tels que Kubernetes, Helm pour la gestion des déploiements, et l'utilisation de services account pour les déploiements sur AWS.

En dépit de ces ambitions non réalisées, nous sommes reconnaissants d'avoir eu l'opportunité de participer à ce projet et d'avoir suivi ce cours. Cette expérience nous a permis de développer de nouvelles compétences et de renforcer notre compréhension des bonnes pratiques en matière de développement logiciel et de sécurité informatique.