SUSE

Guide

# 10 étapes pour automatiser la sécurité des conteneurs dans le pipeline CI/CD

# 10 étapes pour automatiser la sécurité des conteneurs dans le pipeline CI/CD

## Comment intégrer la sécurité dans le pipeline Kubernetes

## Introduction

L'application des exigences de sécurité et de conformité dans les pipelines modernes cloud native peut s'avérer complexe. La surface d'attaque accrue des infrastructures de conteneurs rend la sécurité encore plus importante, mais les équipes de sécurité et DevOps ne peuvent pas se permettre de ralentir le pipeline avec des processus manuels.

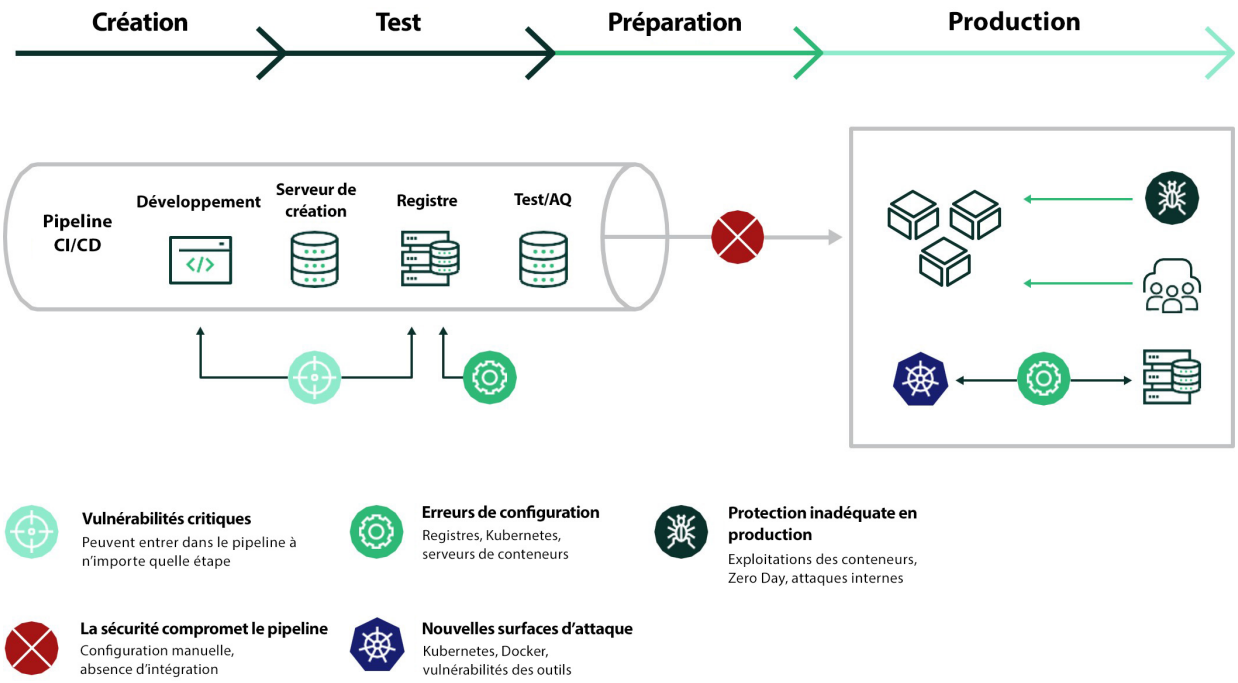
Un large éventail de fonctionnalités de sécurité cloud native est nécessaire pour réussir l'intégration de la sécurité dans le pipeline CI/CD, y compris dans l'environnement de production. Pour mettre en oeuvre une approche de sécurité en couches pour les conteneurs, commencez par analyser les images des conteneurs afin de détecter les possibles vulnérabilités et problèmes de conformité. Continuez ensuite vers la droite avec des protections en temps réel du réseau, des conteneurs et des serveurs.

L'analyse des vulnérabilités sans application de correctifs virtuels submergera les équipes avec des données non exploitables, et la sécurité « run-time » sans gestion des vulnérabilités et de la conformité dans le pipeline augmentera le risque d'attaques préjudiciables dans l'environnement de production.

Ce guide aborde les problèmes de sécurité auxquels sont confrontées les équipes DevOps et de sécurité, et identifie dix points d'intégration où la sécurité des conteneurs peut être automatisée dans le pipeline.

# La surface d'attaque du pipeline CI/CD

Avant d'examiner l'automatisation de la sécurité des conteneurs, voyons pourquoi la sécurité est aussi critique pour les conteneurs et les pipelines Kubernetes. Dès le début du pipeline CI/CD en production, il existe des points d'attaque où la sécurité peut être compromise. Le schéma ci-dessous fournit un résumé des principaux problèmes de sécurité qui préoccupent constamment les équipes DevOps et de sécurité.




**Vulnérabilités critiques**
Peuvent entrer dans le pipeline à n'importe quelle étape

**Erreurs de configuration**
Registres, Kubernetes, serveurs de conteneurs

**Protection inadéquate en production**
Exploitations des conteneurs, Zero Day, attaques internes

**La sécurité compromet le pipeline**
Configuration manuelle, absence d'intégration

**Nouvelles surfaces d'attaque**
Kubernetes, Docker, vulnérabilités des outils

# La résolution appropriée de ces problèmes de sécurité doit être l'objectif principal de l'intégration de la sécurité dans le pipeline

## Vulnérabilités critiques

Des vulnérabilités peuvent être introduites dans la phase de création, dans le registre et même dans l'environnement de production. Toutes les vulnérabilités ne disposent pas de correctifs et des vulnérabilités peuvent être détectées même après l'analyse de la phase de création, dans les images des registres ou des conteneurs déjà déployés.

## La sécurité compromet le pipeline

L'ajout d'étapes de configuration manuelles pour des raisons de sécurité ou l'introduction de points de contrôle de processus pour les examens de sécurité peut ralentir, voire arrêter l'intégration continue et le lancement d'applications nouvelles ou mises à jour. Cela réduit les avantages commerciaux issus d'un pipeline CI/CD. Les équipes de sécurité ne peuvent pas se permettre d'empêcher l'entreprise d'atteindre ses objectifs.

## Erreurs de configuration

Les outils cloud native, notamment le développement, les registres et Kubernetes, sont à la fois nouveaux et complexes, ce qui entraîne des erreurs de configuration qui compromettent la sécurité. Bien que l'apprentissage et la formation soient une solution, un audit continu des configurations doit être effectué pour s'assurer qu'un environnement configuré en toute sécurité à un moment ne s'ouvre pas aux attaques après la mise à jour suivante.

## Nouvelles surfaces d'attaque

L'infrastructure stratégique, telle que l'orchestrateur de conteneurs Kubernetes, CRI-O d'exécution ou containerd, et l'infrastructure sous-jacente, comme les registres, sont toutes deux des surfaces d'attaque. Les pirates informatiques ont désormais compris que l'explosion des conteneurs en production en fait des cibles parfaites pour lancer des attaques Zero Day. L'évolution rapide et le manque d'expérience opérationnelle généralisée des outils cloud native impliquent qu'ils ne seront pas testés et protégés contre les attaques pendant de nombreuses années.

## Protection inadéquate en production

De bonnes méthodes de sécurité incluent des analyses rigoureuses des vulnérabilités, des audits de conformité et le renforcement des infrastructures de production. Cependant, aucune analyse ni préparation n'éliminera le risque d'attaques Zero Day, internes et de phishing. Si vous devez protéger des données sensibles ou des ressources précieuses, la sécurité d'exécution dotée d'une protection approfondie du réseau et des périphériques (conteneurs, serveur) est indispensable.

# 10 étapes pour l'automatisation de la sécurité des conteneurs – Résumé

Dans ce document, nous allons examiner en détail dix points d'intégration qui peuvent servir de guide pour les 10 étapes pour l'automatisation de la sécurité des conteneurs. Bien qu'il y en ait beaucoup plus de dix, celles-ci sont considérées comme des étapes essentielles, car elles sont simples et efficaces pour améliorer la sécurité. Voici le résumé de ces étapes, suivi d'un schéma de référence de pipeline.

1. **Analyse de la phase de création**
   Les images de conteneur doivent être analysées pour détecter les vulnérabilités et les violations de conformité pendant la phase de création. Ainsi, l'étape de création pourra être arrêtée (mise en échec) pour imposer la correction ou le traitement. L'intégration est facilitée par des plug-ins et des extensions pour les outils courants tels que Jenkins, CircleCI, Azure DevOps, Gitlab, Bamboo, etc.

2. **Analyse des registres**
   Une fois que les images ont passé l'analyse de la phase de création, elles sont placées dans des registres pour y être également analysées. De nouvelles vulnérabilités peuvent être découvertes ou introduites après la migration des images dans les registres. Les résultats de l'analyse des registres peuvent également être liés aux contrôles d'admission (voir n° 7 ci-dessous) pour empêcher le déploiement d'images non autorisées ou vulnérables.

3. **Analyse de la production, bancs d'essai et audits de conformité CIS**
   L'analyse et les audits doivent s'étendre aux environnements de préparation et de production avec des analyses de vulnérabilité d'exécution et l'exécution de bancs d'essai CIS pour Kubernetes et Docker, ainsi que des contrôles de conformité personnalisés.

4. **Création de rapports sur les risques et gestion des vulnérabilités**
   Bien que la gestion des environnements à haut risque et l'interprétation des rapports de gestion des vulnérabilités impliquent généralement un examen manuel, les alertes et la corrélation peuvent être effectuées automatiquement pour accélérer et faciliter l'évaluation et le traitement.

5. **Politique de sécurité sous forme de code**
   La définition et le déploiement des règles de sécurité pour les nouvelles applications peuvent être automatisés sous forme de code, de la même manière que Kubernetes prend en charge les manifestes de déploiement dans les fichiers yaml standard. Ainsi, les workloads nouveaux ou mis à jour sont automatiquement protégés lorsqu'ils sont déployés en production.

6. **Apprentissage comportemental des applications**
   L'apprentissage comportemental est une technique importante permettant de caractériser automatiquement le comportement des applications, comme les connexions réseau, les protocoles utilisés, les processus requis et l'activité d'accès aux fichiers autorisée. Nous verrons plus tard comment cela peut fonctionner avec la politique de sécurité sous forme de code (n° 5) pour automatiser la sécurité d'exécution du développement à la production.

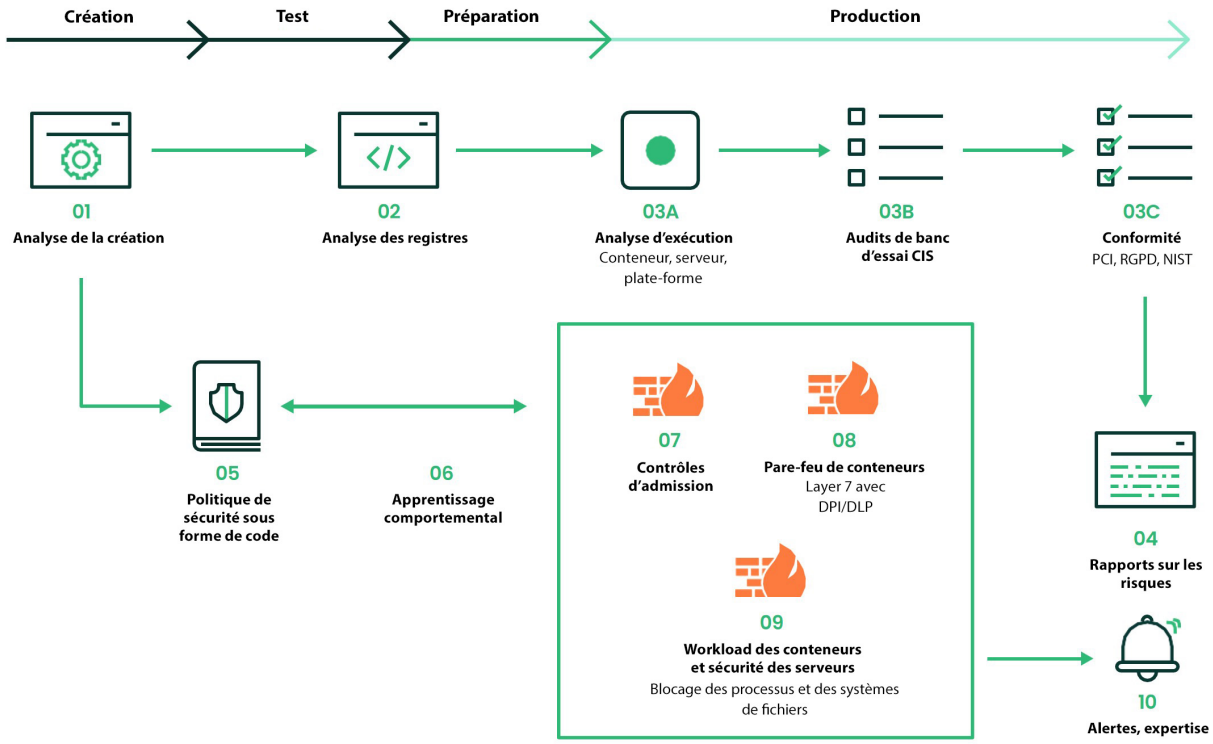
7. **Contrôles d'admission**
   Les contrôles d'admission constituent une passerelle importante entre le pipeline CI/CD et l'environnement de production. Une fois les règles établies, les déploiements vulnérables ou non autorisés peuvent être automatiquement évités.

8. **Pare-feu du réseau de conteneurs**
   Un pare-feu de conteneurs Layer 7 (couche Application) appliquera automatiquement la segmentation du réseau en bloquant les attaques réseau et les connexions non autorisées. La création et la maintenance de ces règles réseau de liste blanche peuvent être automatisées sous forme de code (n° 5) à l'aide de l'apprentissage comportemental (n° 6).

9. **Workload des conteneurs et sécurité des serveurs (périphériques)**
   Les conteneurs et les serveurs doivent faire l'objet d'une surveillance continue afin de détecter tout comportement suspect ou non autorisé, tel que les processus et l'activité des fichiers. Ces activités peuvent être bloquées automatiquement. La création et la maintenance de ces règles de liste blanche peuvent être automatisées sous forme de code (n° 5) à l'aide de l'apprentissage comportemental (n° 6).

10. **Alertes, réponses et expertises**
    Enfin, des alertes, des réponses et des expertises automatisées peuvent être lancées en cas d'activités suspectes. Il peut s'agir de la mise en quarantaine des conteneurs, de captures de paquets et de notifications personnalisées envoyées aux systèmes de gestion des dossiers.

## 10 étapes pour une sécurité complète du cycle de vie



## Qui doit mettre en oeuvre ces étapes ?

Les 10 étapes résumées ci-dessus constituent un excellent point de départ pour l'automatisation de la sécurité. Mais qui doit être responsable de chacune de ces étapes ? Bien évidemment, la réponse est « ça dépend ». Bien que les nombreux rôles impliqués dans le pipeline puissent inclure les développeurs, les DevOps, les SRE (ingénieurs en fiabilité de site)/opérations, la conformité et la sécurité, chaque organisation aura une approche et un processus différents pour la gestion de la sécurité.

**Prenons une vue simplifiée de deux rôles importants : DevOps (qui inclut DevSecOps) et la sécurité. Ces considérations peuvent permettre de déterminer qui est responsable de chaque étape :**

En fin de compte, les équipes de sécurité sont chargées de protéger les ressources critiques, en particulier dans un environnement de production. Cependant, le mouvement « Shift-Left » visant à intégrer la sécurité dans le pipeline en amont signifie que les DevOps doivent comprendre les concepts de sécurité et maîtriser certains aspects de l'automatisation de la sécurité.

Bien que les équipes de sécurité soient souvent formées à des concepts tels que la segmentation du réseau, les pare-feu, la sécurité des périphériques, l'IDS/IPS et la réponse aux incidents, elles ne connaissent souvent pas aussi bien les outils modernes d'orchestration du cloud tels que les conteneurs et Kubernetes. Elles devront apprendre lesquels de ces concepts de sécurité traditionnels doivent être appliqués dans les déploiements cloud native, et comment ils doivent être mis en oeuvre.

Même si les équipes DevOps deviennent rapidement des spécialistes des pipelines CI/CD automatisés et de la gestion des outils d'orchestration tels que Kubernetes, elles manquent souvent de compréhension et d'expérience en ce qui concerne les technologies de sécurité et les meilleures pratiques dans les environnements de production nécessaires pour contrer les attaques constantes sur l'infrastructure et les applications.

De nouveaux outils cloud native tels que les registres, les orchestrateurs (Kubernetes), les maillages de services et d'autres outils Open Source introduisent de nouvelles surfaces d'attaque qui sont inconnues des équipes de sécurité ou DevOps.

Les nouveaux concepts tels que « l'infrastructure sous forme de code » et « la politique de sécurité sous forme de code » nécessiteront une collaboration entre les équipes DevOps et de sécurité afin de permettre l'automatisation des pipelines dans un environnement sécurisé.

En général, les premières étapes (1-4) relèvent de la responsabilité des équipes DevOps et de conformité, tandis que les dernières étapes (8-10) relèvent des équipes des opérations et de sécurité. Les étapes intermédiaires (5-7) relient le pipeline CI/CD et l'environnement de production, la politique de sécurité sous forme de code et les contrôles d'admission étant gérés par les équipes DevOps sous la supervision de l'équipe de sécurité.

## Comment l'intégration doit-elle être effectuée ?

**Il existe des outils et des techniques permettant d'automatiser l'ensemble des 10 étapes et plus encore, de plug-ins à des scripts simples. De nombreuses automatisations sont déjà intégrées à des outils tels que SUSE NeuVector pour simplifier l'application de la sécurité.**

| Plug-ins et extensions | Intégrations incorporées | API REST | Notifications |
|---|---|---|---|
| Les automatisations de sécurité les plus fréquemment utilisées disposent de plug-ins ou d'extensions, comme un plug-in Jenkins pour l'analyse de la phase de création. | Les outils modernes cloud native tels que Kubernetes offrent des ressources permettant des intégrations de sécurité telles que les contrôles d'admission, les CRD ou l'application RBAC.<br>Ils peuvent être exploités par des outils de sécurité tels que SUSE NeuVector afin de fournir un ensemble complet de fonctions de sécurité intégrées automatiquement à l'orchestrateur. | Pour toute intégration qui doit être hautement personnalisée ou qui n'est pas prise en charge par un plug-in, une API REST complète permet de créer des scripts pour les automatisations de sécurité. Par exemple, les analyses, les règles de politique, la gestion des utilisateurs et chaque fonction configurable de SUSE NeuVector peuvent être contrôlées via l'API REST. | Les alertes et les notifications peuvent être automatisées par le biais de méthodes traditionnelles telles que les événements SYSLOG ou les déclencheurs modernes comme les webhooks. Ils peuvent non seulement déclencher des alertes et des notifications, mais aussi ouvrir des dossiers et déclencher des réponses telles que des mises en quarantaine et des captures de paquets. |

Il existe des outils et des techniques permettant d'automatiser l'ensemble des 10 étapes et plus encore, de plug-ins à des scripts simples. De nombreuses automatisations sont déjà intégrées à des outils tels que SUSE NeuVector pour simplifier l'application de la sécurité.

Dans la plupart des cas, une configuration simple pour la personnalisation de la politique de sécurité est requise au départ, et peut même être automatisée via les API REST. Une fois la configuration effectuée, les analyses, le monitoring, l'application, la mise à jour et les alertes peuvent être automatisés.

# Étapes 1 à 4 : Gestion des vulnérabilités et de la conformité de bout en bout

**La gestion des vulnérabilités de bout en bout doit suivre le workflow Évaluer, Hiérarchiser, Agir et Améliorer, comme indiqué ci-dessous.**


## Les principales étapes de l'automatisation dans le pipeline CI/CD comprennent les éléments suivants :

1. **Déclencheurs d'analyse des vulnérabilités de la phase de création**
   A. Utilisez des plug-ins, des extensions ou des API REST pour appliquer l'analyse lors de la création d'images afin d'identifier les images vulnérables dès le début du pipeline. Par exemple, SUSE NeuVector fournit un plug-in Jenkins, un ORB CircleCI, des extensions Azure DevOps, une API REST et d'autres moyens d'imposer des analyses de la phase de création. Les images peuvent également être annotées par le développeur ou l'équipe responsable afin de faciliter les alertes ou la création de rapports.
   B. Les alertes concernant les vulnérabilités critiques détectées doivent être envoyées à l'équipe de gestion de la conformité et éventuellement au développeur responsable de l'image. Elles peuvent être filtrées afin de traiter en priorité les problèmes de vulnérabilité urgents pour lesquels il existe un correctif. Si l'image est annotée avec le nom du développeur, les alertes sont facilitées et plus précises. Reportez-vous à la section ci-dessous sur l'application de correctifs virtuels pour une présentation de la gestion des vulnérabilités sans correctif disponible qui doivent être autorisées à exister en production.

2. **Analyse des registres automatisée**
   A. Surveillez en permanence les images dans chaque registre utilisé pour les environnements de préparation et de production. Les résultats des analyses en couches peuvent faciliter la recherche par les développeurs du package ou de la bibliothèque vulnérable en examinant les commandes de création pour chaque couche. Les images peuvent également être annotées par le développeur ou l'équipe responsable afin de faciliter les alertes ou la création de rapports.
   B. Les alertes peuvent être gérées de la même manière que les analyses de la phase de création, où seules les vulnérabilités critiques avec des correctifs disponibles sont envoyées aux développeurs pour traitement. Si l'image est annotée avec le nom du développeur, les alertes sont facilitées et plus précises. Bien que l'analyse et les alertes doivent être automatisées, certains processus manuels peuvent être nécessaires au départ jusqu'à ce que l'intégration soit testée et achevée.

3. **Analyse et audits d'exécution (production)**
   A. Surveillez en permanence les images dans chaque registre utilisé pour les environnements de préparation et de production. Les résultats des analyses en couches peuvent faciliter la recherche par les développeurs du package ou de la bibliothèque vulnérable en examinant les commandes de création pour chaque couche. Les images peuvent également être annotées par le développeur ou l'équipe responsable afin de faciliter les alertes ou la création de rapports.
   B. Exécutez en continu des bancs d'essai et des contrôles de conformité CIS personnalisés.
   C. Appliquez les exigences de conformité du secteur telles que le pare-feu et la segmentation pour PCI et d'autres normes de confidentialité. Reportez-vous aux sections de ce document sur la politique de sécurité sous forme de code, l'apprentissage comportemental et la segmentation du réseau pour savoir comment automatiser la conformité pour l'application du pare-feu du réseau.
   D. Les alertes concernant les vulnérabilités d'exécution critiques avec des correctifs disponibles, ainsi que certains bancs d'essai CIS, tels que les conteneurs s'exécutant en tant que racine, doivent être envoyées à l'équipe DevOps pour examen.

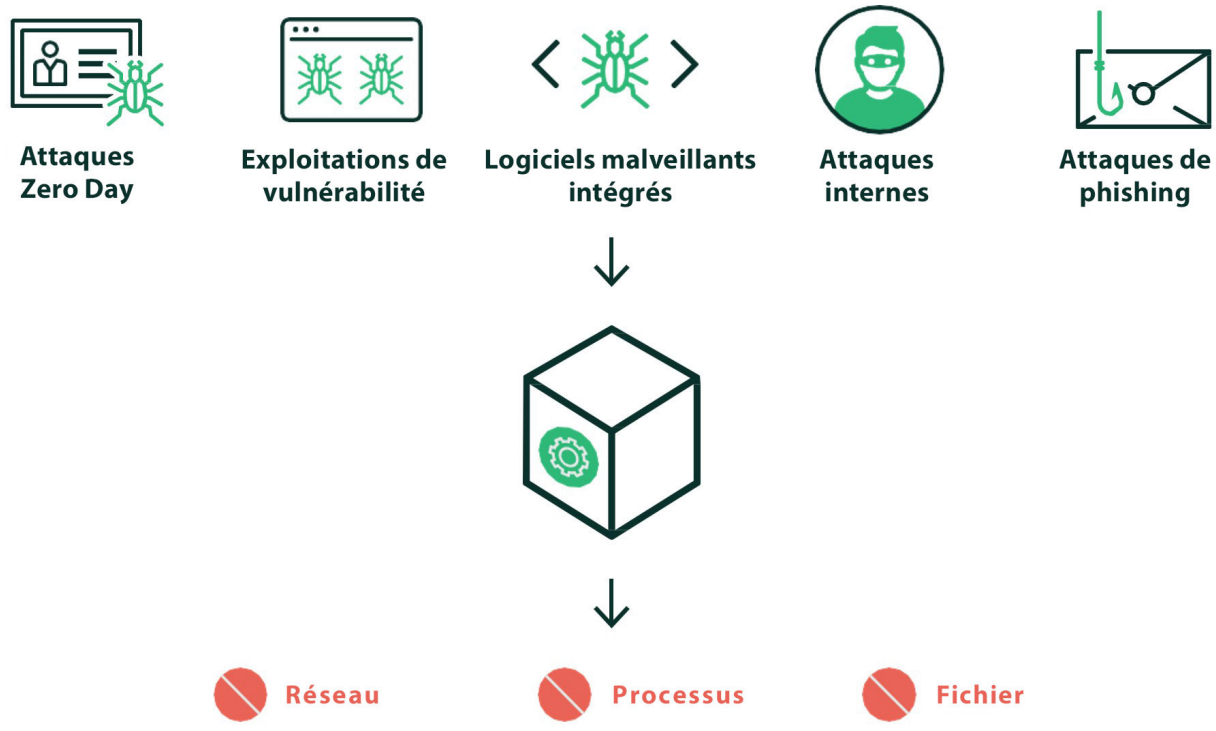
4. **Analyse, triage et corrélation de « l'impact » des vulnérabilités en production**
   A. Mettez en corrélation les vulnérabilités avec les images, les conteneurs et les serveurs. Évaluez les ressources de production affectées par les problèmes critiques de vulnérabilités et de conformité.
   B. Mettez en oeuvre « l'application de correctifs virtuels » pour atténuer les risques liés à la production.
   C. Les alertes concernant les ressources non protégées présentant des vulnérabilités critiques doivent être envoyées aux équipes DevOps et de sécurité. Les ressources pour lesquelles des correctifs virtuels sont appliqués peuvent être rétrogradées ou leur priorité peut être abaissée, car elles présentent un risque d'exploitation moindre.

## Qu'est-ce que l'application de correctifs virtuels ?

L'application de correctifs virtuels permet aux équipes de sécurité de « corriger virtuellement » une vulnérabilité dans un conteneur ou un serveur d'exécution, sans avoir à mettre à jour ou à remplacer la ressource d'exécution par une version corrigée. En fait, elle protège la ressource d'exécution (conteneur, serveur, etc.) contre une tentative d'exploitation de la vulnérabilité.

La meilleure façon de procéder est de caractériser et de placer sur liste blanche automatiquement tous les comportements des conteneurs d'applications, tels que les connexions réseau, les processus et l'activité des fichiers, puis de les verrouiller. Le workload ou le serveur est ensuite « corrigé virtuellement » et toute tentative d'exploitation crée une connexion réseau, un processus ou un accès aux fichiers non autorisé qui peut être bloqué.

L'application de correctifs virtuels protège contre les exploitations de vulnérabilité, les logiciels malveillants intégrés, les attaques Zero Day et les attaques internes et de phishing.




Lecture supplémentaire | Consultez cet article pour en savoir plus sur la gestion des vulnérabilités de bout en bout et l'application de correctifs virtuels

# Étapes 5 à 6 : Politique de sécurité sous forme de code et apprentissage comportemental

L'automatisation de l'analyse des vulnérabilités est un excellent point de départ pour sécuriser les conteneurs. Ce qui est encore plus difficile et important, c'est l'automatisation de la création de politiques de sécurité pour protéger les workloads d'applications en production. Les politiques de sécurité d'exécution, en particulier les règles de pare-feu, nécessitaient jusqu'à présent une configuration manuelle dans les infrastructures de datacenter existantes.

Cependant, dans les déploiements cloud modernes, l'utilisation de ressources personnalisées Kubernetes pour déclarer une politique de sécurité d'applications à n'importe quel stade du pipeline fournit une solution à ce problème.

Les développeurs, les DevOps et les équipes de sécurité peuvent utiliser des définitions de ressources personnalisées (CRD) pour automatiser et maintenir les politiques de sécurité d'exécution. Les CRD peuvent également être utilisées pour appliquer des politiques de sécurité globales sur plusieurs clusters Kubernetes.

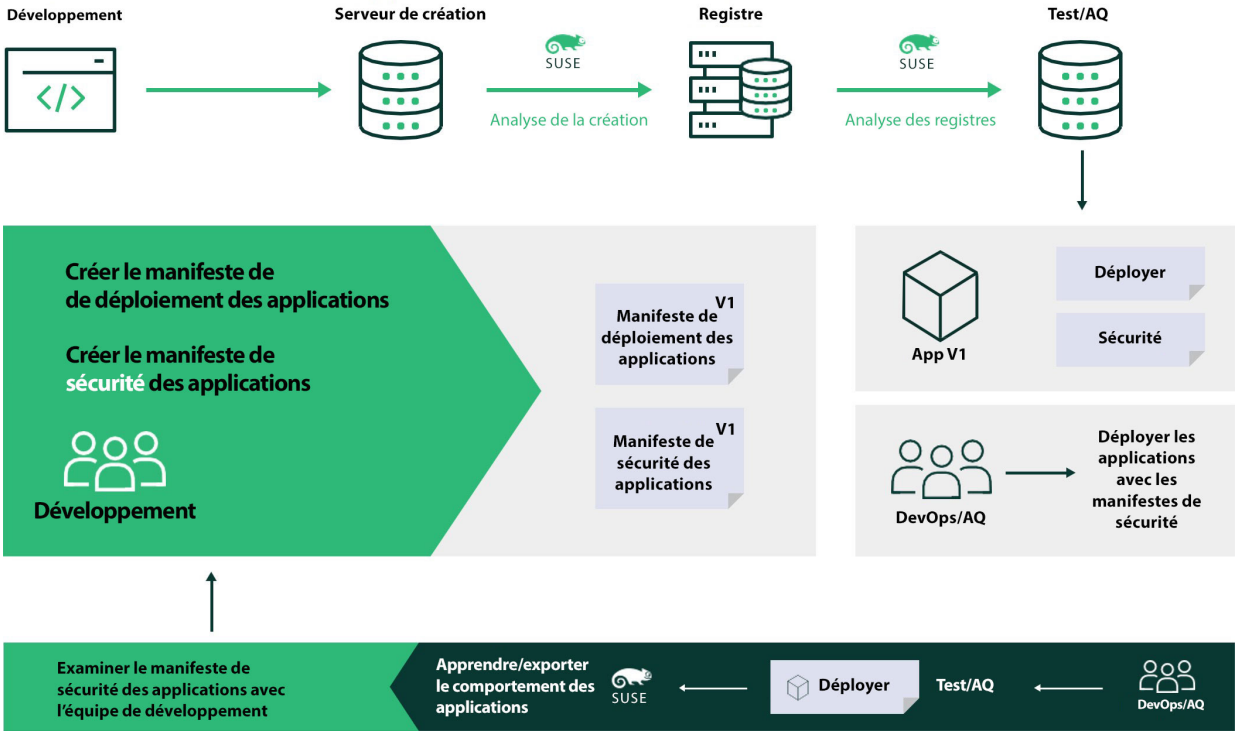
L'apprentissage comportemental est une technique utile pour apprendre et caractériser le comportement d'une application afin de rédiger automatiquement la CRD de la politique de sécurité. Cette CRD peut ensuite être examinée par les équipes de développement, DevOps et de sécurité et modifiée selon les besoins, puis autorisée en tant que politique de sécurité pour l'application. Enfin, ce « code » CRD peut être enregistré dans le système de gestion des changements avant d'être déployé en production.

## Les principales étapes de l'automatisation dans le pipeline CI/CD comprennent les éléments suivants :

5. **Politique de sécurité sous forme de code et apprentissage comportemental**
   A. Saisissez le comportement « autorisé » des applications dans les fichiers yaml CRD standard. Le comportement doit inclure les connexions réseau et les protocoles autorisés, y compris les entrées/sorties, les processus et l'activité d'accès aux fichiers. Les politiques de sécurité qui ne sont pas liées à des applications spécifiques, telles que les règles de sécurité globales, peuvent également être exprimées dans des CRD distinctes.
   B. Créez des étapes d'examen, d'approbation et de réintégration dans le pipeline afin que la politique de sécurité sous forme de code représente l'autorité ultime pour les règles de sécurité de production.
   C. Déployez les CRD de la politique de sécurité dans l'environnement de production avant le déploiement des applications nouvelles ou mises à jour, afin de garantir la sécurité des nouveaux workloads dès leur exécution.

6. **Utilisation de l'apprentissage comportemental pour automatiser la politique de sécurité**
   A. Dans les environnements de test, d'assurance qualité et/ou de préparation, exécutez les applications et leurs suites de tests associées, et utilisez l'apprentissage comportemental pour capturer le comportement autorisé des applications sous la forme de règles de sécurité réseau, de processus, de fichiers et autres.
   B. Exportez les règles en tant que CRD pour examen et utilisez-les à l'étape 5 ci-dessus.

**Le schéma ci-dessous montre le workflow utilisant l'apprentissage comportemental pour générer la CRD de la politique de sécurité pour une nouvelle application.**




La partie supérieure du schéma montre le workflow dans lequel les développeurs créent l'application, le manifeste de déploiement des applications et l'image, qui passe par un processus d'analyse automatisé dans les étapes 1 et 2. Parallèlement, l'équipe DevOps/AQ exécute l'application (en bas du schéma, de droite à gauche) et utilise l'apprentissage comportemental de SUSE NeuVector pour créer le manifeste de sécurité (la CRD) à examiner avec l'équipe de développement. Après avoir été approuvée et testée, la CRD est enregistrée dans le système de gestion des changements et déployée dans l'environnement de production pour protéger la nouvelle application contre les attaques d'exécution.

# Qu'est-ce qu'une définition de ressources personnalisées (CRD) SUSE NeuVector ?

**La CRD SUSE NeuVector permet de créer une politique de sécurité sous forme de code avec des fichiers yaml natifs Kubernetes. La CRD définit le comportement autorisé des applications, par exemple :**

| Connexions réseau | Processus de conteneur | Activité des fichiers de conteneur |
|---|---|---|
| Il s'agit notamment de règles Layer 7 basées sur les protocoles d'application. Les règles peuvent appliquer la segmentation est-ouest des applications ainsi que les règles d'entrée et de sortie. | Les processus du conteneur d'applications sont mis sur liste blanche afin que les processus non autorisés puissent être détectés. | L'activité des fichiers avec des applications spécifiques sur liste blanche peut être définie. |

**En outre, d'autres configurations liées à la sécurité SUSE NeuVector peuvent être déclarées :**

| Groupes | État de protection des applications (conteneur) | Autres configurations |
|---|---|---|
| Des regroupements logiques de conteneurs ou d'autres objets peuvent être créés dans SUSE NeuVector, et des règles peuvent être appliquées à ces groupes. | L'état de protection des applications, à savoir Découvrir, Surveiller ou Protéger. | Des paramètres supplémentaires peuvent être configurés et ce format peut être étendu pour en ajouter d'autres dans les versions futures. Ils sont facilement appliqués et mis à jour, par exemple à l'aide des commandes « kubectl apply -f new_policy.yaml ». |

Les CRD sont contrôlées automatiquement via RBAC par Kubernetes, de sorte que seuls les utilisateurs disposant d'un accès approprié (administrateur avec un espace de noms ou de cluster) seront autorisés à créer ou modifier les ressources de la politique de sécurité sous forme de code.

Lecture supplémentaire | Consultez cet article pour en savoir plus sur la mise en oeuvre de la politique de sécurité sous forme de code.

## Étapes 7 à 9 : Protection complète d'exécution

Une fois en production, les conteneurs, leurs serveurs et l'orchestrateur doivent être protégés contre les attaques. La protection complète d'exécution doit inclure les contrôles d'accès aux fichiers et aux processus de conteneur, le monitoring des serveurs et, surtout, la sécurité réseau avec une inspection approfondie des paquets. Par le passé, les règles de pare-feu du réseau et les politiques de sécurité des périphériques nécessitaient une personnalisation manuelle importante, mais cela peut être évité pour les pipelines automatisés modernes.

La sécurité d'exécution commence par les contrôles d'admission sous forme de sentinelle de l'environnement de production, et passe par l'analyse d'exécution et les contrôles de conformité à la prévention des attaques en temps réel.

### Les principales étapes de l'automatisation dans le pipeline CI/CD comprennent les éléments suivants :

7. **Utilisation de contrôles d'admission pour empêcher les déploiements vulnérables ou non autorisés**
   A. Utilisez des critères tels que les résultats d'analyse de vulnérabilité, les espaces de noms, les registres et les propriétés de conteneur (par exemple, s'exécutant en tant que racine) pour contrôler les déploiements.
   B. Bien que la création de ces règles puisse être automatisée ou manuelle, l'application et les alertes doivent être automatisées.
8. **Déploiement d'un pare-feu de conteneurs de couche 7 pour automatiser la segmentation et la détection des menaces**
   A. Automatisez le blocage, la mise en quarantaine, la capture de paquets et les alertes en cas d'attaques réseau et de violations de segmentation.
   B. Utilisez la politique de sécurité sous forme de code pour automatiser la création et les mises à jour des règles.
9. **Utilisation de contrôles de sécurité des périphériques pour les conteneurs et les serveurs**
   A. Automatisez le blocage et les alertes pour les processus et les activités de fichiers suspects dans les conteneurs et les serveurs.
   B. Utilisez la politique de sécurité sous forme de code pour automatiser la création et les mises à jour des règles.

## Qu'est-ce que la segmentation du réseau de conteneurs ?

SUSE NeuVector fournit un véritable pare-feu de conteneurs Layer 7 cloud native qui effectue automatiquement la segmentation du réseau. Grâce à l'apprentissage comportemental, les connexions et les protocoles d'application utilisés entre les services sont détectés et des règles de liste blanche destinées à les isoler sont automatiquement créées. Cela signifie que la segmentation des conteneurs est simple et automatisée, et qu'elle n'a pas besoin de connaître les connexions au préalable ni de réaliser la création et la maintenance manuelles des règles de segmentation.

Avec une solution de segmentation des conteneurs Layer 7 cloud native, les workloads peuvent être segmentés même s'ils s'exécutent sur le même serveur ou dans le même cluster. Cela est particulièrement utile pour répondre aux exigences de conformité aux normes de l'industrie telles que la norme PCI DSS.

## Le modèle ultime de sécurité dans le cloud : la segmentation des conteneurs par workload

En fin de compte, pour offrir aux entreprises la plus grande flexibilité possible en matière de lancement rapide et d'utilisation optimale des ressources, la segmentation des conteneurs doit être appliquée à chaque pod et suivre les workloads d'application à mesure de leur évolution et de leurs déplacements dynamiques. Dans cet article sur la vision du micropérimètre, Gary Duan, CTO de SUSE NeuVector, présente une vision de la sécurité du cloud dans laquelle le périmètre de protection entoure le workload, même lorsqu'il est en mouvement dans les clouds hybrides.

Lecture supplémentaire | Consultez cet article pour en savoir plus sur l'utilisation des contrôles d'admission pour empêcher les déploiements de conteneurs non autorisés et vulnérables.

Lecture supplémentaire | Consultez cet article pour en savoir plus sur les modèles de segmentation des conteneurs, y compris pour la conformité PCI.

## Étape 10 : Alertes, réponse et expertises

De nombreuses automatisations de cette étape sont relativement bien comprises et mises en oeuvre pour d'autres infrastructures. Par exemple, la création de rapports d'événements vers les systèmes SIEM pour la capture d'alertes et d'expertises. Dans un environnement basé sur des conteneurs, certaines actions sont plus difficiles, comme le lancement de captures de paquets uniquement sur des conteneurs suspects, ou bien l'identification appropriée du pod, de l'espace de noms ou du service d'application par son nom de déploiement Kubernetes, afin d'identifier où se produit une attaque.

### Les principales étapes de l'automatisation dans le pipeline CI/CD comprennent les éléments suivants :

10. **Automatisation des alertes et des réponses en temps réel**
    - A. Générez des alertes spéciales pour les événements de sécurité critiques.
    - B. Placez en quarantaine les conteneurs suspects.
    - C. Lancez des captures de paquets à des fins d'enquêtes et d'expertise.
    - D. Intégrez les outils de gestion des dossiers pour résoudre les violations de sécurité et de conformité.

SUSE NeuVector offre plusieurs mécanismes d'automatisation et d'intégration à l'étape 10. Il s'agit notamment de la sortie d'événements SYSLOG, d'alertes webhook personnalisées, de la capture et de la mise en quarantaine automatisées des paquets, et d'une API REST pour des automatisations personnalisées et scriptées. Un tableau de bord Grafana et un exporter Prometheus sont également pris en charge.

## Résumé

### Prenez les mesures nécessaires pour automatiser la sécurité

La voie vers un pipeline entièrement automatisé avec des contrôles de sécurité intégrés jusqu'à la production commence par quelques étapes simples. L'étape parmi les dix que vous choisirez pour commencer dépend de votre progression dans le processus d'automatisation de votre pipeline. L'essentiel est de commencer par les actions qui peuvent être effectuées facilement et celles qui sont les plus importantes, puis de créer une feuille de route pour une sécurité entièrement automatisée.

Par exemple, l'analyse de la phase de création et l'analyse des registres sont relativement simples à automatiser et offrent de bons résultats. Les réponses de sécurité d'exécution automatisées, telles que les alertes, le blocage, la mise en quarantaine et la capture de paquets, sont également simples à appliquer. En revanche, la nécessité d'une politique de sécurité sous forme de code peut être plus complexe, tant du point de vue de la création technique que du test des workflows, que du point de vue des processus humains. Pour réussir, les équipes DevOps et de sécurité doivent s'accorder sur les rôles, les responsabilités, les processus du pipeline et le rôle que les développeurs jouent dans le déploiement de la politique de sécurité sous forme de code.

La bonne nouvelle, c'est que la mise en oeuvre d'un petit sous-ensemble des étapes décrites dans ce guide permettra d'améliorer l'automatisation des pipelines. En outre, les organisations pourront évoluer et atteindre l'objectif d'automatisation des pipelines avec l'intégration de la sécurité.

## Étapes suivantes

### Vous souhaitez obtenir plus d'informations ?

Contactez SUSE NeuVector à l'adresse https://NeuVector.com pour obtenir plus d'articles concernant la sécurité des conteneurs sur notre blog. Vous pourrez également planifier une démonstration de la plate-forme de sécurité des conteneurs SUSE NeuVector, y compris le pare-feu de conteneurs Layer 7.

SUSE

2023

SUSE Software Solutions
Germany GmbH

Frankenstraße 146
90461 Nürnberg
Allemagne

www.suse.com

Pour en savoir plus, contactez SUSE
aux numéros suivants :

+1 800 796 3700 (États-Unis/Canada)

+49 (0)911-740 53-0 (International)

# Innover partout