CCN-CERT

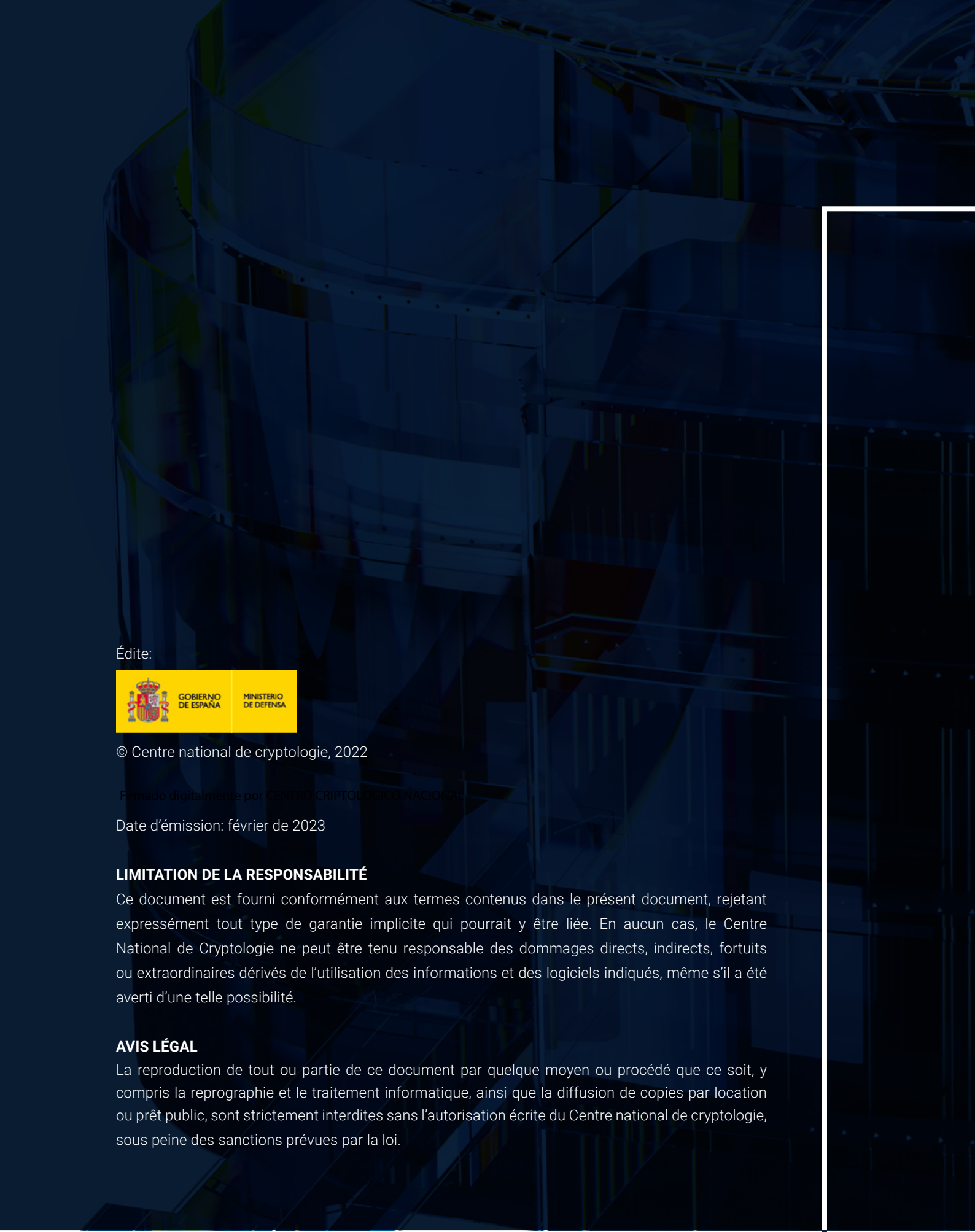
BP/27

Recommandations de sécurité de Kubernetes

RAPPORT DE BONNES PRATIQUES

SEPTEMBRE 2022

cn-cert centro criptológico nacional
CCN centro criptológico nacional



Édite:

GOBIERNO DE ESPAÑA MINISTERIO DE DEFENSA

Firmado digitalmente por CENTRO CRIPTOLÓGICO NACIONAL

Date d'émission: février de 2023

Indice

1. Introduction

L'avènement de la technologie de conteneurisation (Docker) a fourni un moyen de créer, déployer et mettre à l'échelle rapidement des applications et des services ou d'effectuer le déploiement de logiciels.

Avec le soutien de grandes entreprises telles que Google, RedHat ou Microsoft, entre autres, elle est devenue une option récurrente lors de la mise en œuvre de solutions pour les entreprises, que ce soit de manière locale « on-premise » ou avec un hébergement dans le cloud. Il s'agit d'une technologie en constante évolution, qui répond à la nécessité de générer des microservices de plus en plus critiques pour le bon fonctionnement d'une entreprise ou d'une entité, d'atteindre des structures de haute disponibilité avec redondance des ressources et de protéger les données traitées par les services hébergés.

Face à cette progression, et alors que se forgent des structures de plus en plus complexes, il devient nécessaire de générer une orchestration de tous les ensembles de conteneurs, donnant naissance à la figure de Kubernetes.

Un Kubernetes, un mot issu du grec signifiant « timonier » ou « pilote », est chargé de gérer la coordination des déploiements, la supervision des services et la mise à l'échelle des ressources. En bref, la gestion de tous les services ou microservices générés par une architecture distribuée des systèmes.

Kubernetes est une plateforme permettant d'orchestrer des conteneurs, de gérer la coordination, la surveillance et la mise à l'échelle des ressources dans une architecture distribuée. La technologie de conteneurisation (Docker) est récurrente dans les entreprises.

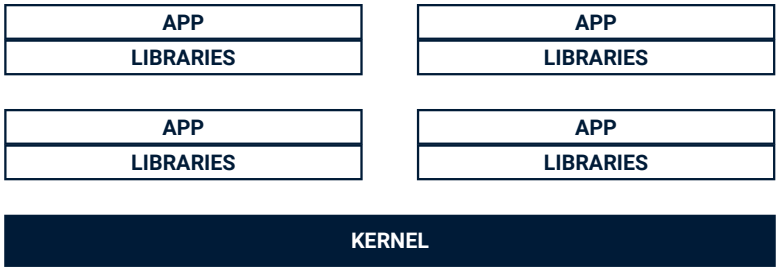


Illustration 1 - Structure du conteneur

2. Kubernetes

Kubernetes est une plateforme open source conçue pour gérer des grappes de services et d'applications déployées dans des conteneurs, qui peuvent être hébergés sur site ou dans le nuage. Sa principale fonctionnalité est la gestion centralisée et l'automatisation des charges de travail.

Développé par Google et les ingénieurs Craig McLuckie et Brendan Burns, et lancé en 2014, le projet est géré par la Cloud Native Computing Foundation (CNCF), une section de la Linux Foundation.

Kubernetes est souvent appelé « k8s », qui est une abréviation des huit lettres de Kubernetes comprises entre le « k » initial et le « s » final.

L'utilisation de cette technologie, qui peut être mise en œuvre sous une forme virtualisée, offre plusieurs possibilités de flexibilité dans sa configuration et différentes couches de sécurité.

Kubernetes est une plateforme de gestion de clusters d'applications et de services de contenu, offrant flexibilité et sécurité lorsqu'elle est utilisée de manière virtualisée.

2.1 Versions

Kubernetes offre une cadence de publication majeure de trois (3) mises à jour mineures par cycles annuels, soutenue par des correctifs de sécurité pendant environ un an après la publication.

Les versions de Kubernetes sont exprimées par une numérotation divisée en trois blocs et séparée par des chiffres.

- **Premier bloc.** Il s'agit de la version principale ou majeure de la version de Kubernetes.
- **Deuxième bloc.** Représentée comme la version mineure ou secondaire.
- **Troisième bloc.** Indique le numéro de version du correctif appliqué à l'application Kubernetes.

REMARQUE : des informations supplémentaires sur le cycle de vie des produits sont disponibles sur le lien suivant :

https://Kubernetes.io/releases/release/

2.2 Composants

Un cluster Kubernetes peut être défini comme un ensemble de différents nœuds, qui peuvent être plusieurs machines virtuelles ou physiques. Ces nœuds sont classés en deux catégories :

- Plan de contrôle.
- Nœuds de travail (worker).

2.2.1 Plan de contrôle Kubernetes

Il s'occupe de la gestion de l'écosystème Kubernetes, et prend les décisions globales pour le cluster, en réagissant à tous les événements déclenchés par les différents nœuds, comme la réponse à la création d'une réplique de tâche.

Les composants du plan de contrôle sont les suivants.

- **Kube-apiserver :** gère l'interaction avec les différents utilisateurs du système par le biais d'une interface de contrôle du système, en gérant les demandes adressées au cluster Kubernetes. Le serveur API Kubernetes détermine quelles requêtes sont valides et comment les traiter.
- **ETCD :** Il s'agit d'une classe de stockage de données non volatiles, où sont sauvegardées toutes les informations relatives à la configuration du cluster Kubernetes.
- **Kube-sheduler :** c'est là que les priorités de gestion des clusters sont ordonnées et planifiées à l'aide d'API à partir de la ligne de commande.
- **Kube-controller-manager :** responsable de la révision des processus indépendants des composants de l'écosystème Kubernetes. Ces composants sont :
  - **Contrôle de la réplication.**
  - **Contrôleur d'extrémité (services et pods).**
  - **Contrôleur de jetons et comptes services (namespaces).**
  - **Contrôleur de nœuds**
- **Cloud-controller-manager.** Il est chargé d'exécuter les contrôleurs qui interagissent avec le nuage. Il peut interagir avec des équilibreurs de charge, servir de médiateur pour les mises à jour ou les suppressions à partir de différents nœuds de travail, ou configurer les chemins de réseau.

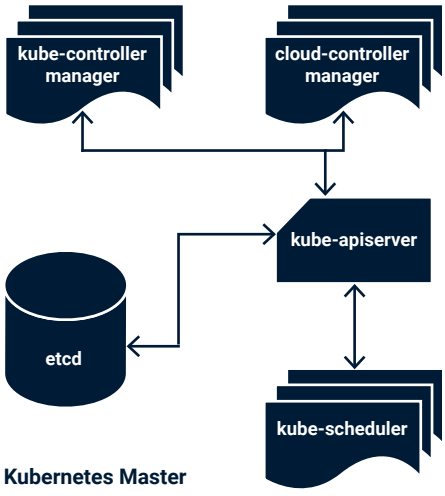


Illustration 2 - Structure du nœud maître.

2.2.2 Nœuds de travail

Aux premiers jours du développement de Kubernetes, le nœud de travail était appelé « minion » et pouvait être constitué d'une ou plusieurs machines, virtuelles ou physiques, selon le type de cluster.

Chaque nœud de travail est contrôlé par un « nœud maître », qui contient tous les outils nécessaires à la gestion d'un ou plusieurs pods.

Un pod, ou des pods, est un regroupement logique utilisé par Kubernetes pour effectuer une gestion plus uniforme d'un ensemble d'un ou plusieurs conteneurs hébergeant des services ou des applications, qui sont proposés par Kubernetes comme solution de produit final.

Les conteneurs de pod partagent les ressources du système en termes d'emplacement de l'espace disque et des noms de réseau (espaces de noms) qui leur sont donnés pour les identifier.

Si plusieurs pods partagent une fonctionnalité, pour une gestion efficace et homogène, Kubernetes regroupe tous ces pods dans une unité logique appelée « services », facilitant ainsi, par exemple, l'attribution des IP d'accès externes, ou facilitant l'équilibrage de la haute disponibilité.

Les composants d'un nœud de travail sont énumérés ci-dessous :

- **Kubelet :** chaque nœud d'un cluster Kubernetes possède un agent, qui est chargé de vérifier l'exécution des pods et de le communiquer au plan de contrôle.
- **Kube-proxy :** agent en charge des règles du réseau ainsi que de la redirection des connexions vers l'hôte hébergeant le cluster.
- **Environnement d'exécution des conteneurs :** il s'agit de l'exécution des conteneurs eux-mêmes dans l'environnement Kubernetes, qui prend en charge différentes technologies de conteneurisation, telles que Docker, containerd, cri-o ou rktlet. Il exécute essentiellement toute implémentation de conteneurisation capable de prendre en charge une interface d'exécution Kubernetes (Kubernetes CRI).

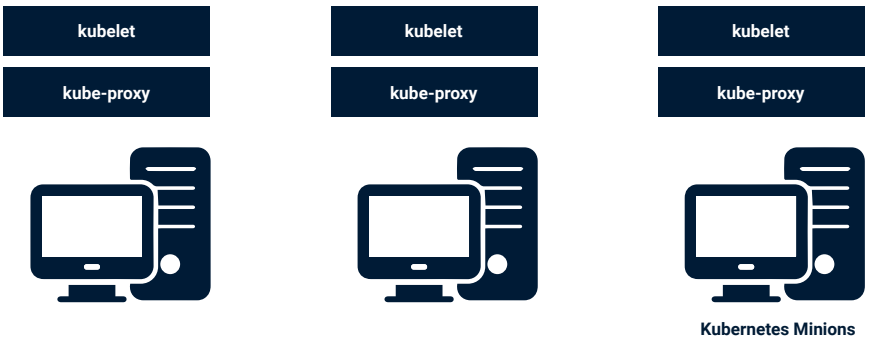


Illustration 3 - Structure de nœud worker.

2.3 Architecture

L'architecture utilisée par Kubernetes est une architecture « maître-esclave » entre les nœuds maîtres et les nœuds de travail.

Les serveurs ou équipements d'un cluster comprennent généralement un ou plusieurs « nœuds », qui peuvent à leur tour être hébergés sur un ou plusieurs serveurs physiques ou virtuels sur site ou dans le nuage.

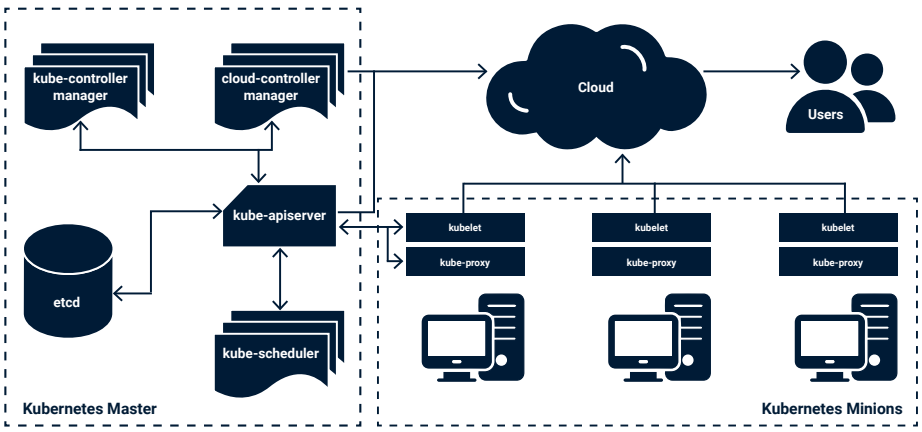


Figure 4 - Architecture d'un cluster Kubernetes

2. Kubernetes

Les clusters Kubernetes, lorsqu'ils sont hébergés dans le cloud, doivent être placés auprès de fournisseurs de services certifiés Kubernetes, qui géreront une partie de l'infrastructure de l'environnement Kubernetes. Il convient de noter que la plupart des configurations par défaut utilisées par ces fournisseurs de services ne s'accompagnent généralement pas de politiques de sécurité inébranlables, mais privilégient plutôt la fonctionnalité et l'accès au service.

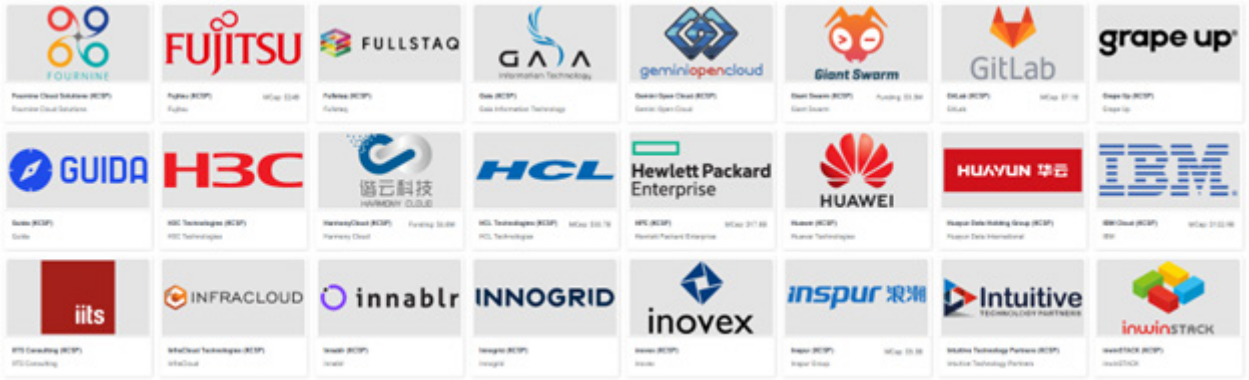


Illustration 5 - Exemple de prestataires de services certifiés.

NOTE : Pour obtenir une liste complète des CSP (Certified Service Providers), veuillez consulter le site pour plus d'informations.

https://Kubernetes.io/partners/#kcsp

Une organisation qui a besoin de disposer de dispositifs de sécurité supplémentaires et de les contrôler peut choisir d'établir des services Kubernetes via des serveurs locaux « on- premise ».

3. Sécurité de Kubernetes

Une organisation peut avoir toutes ses données hébergées sur un cluster Kubernetes, des informations qui peuvent être convoitées par les cybercriminels.

De même, la puissance de calcul des ordinateurs qui hébergent les différents processus et services Kubernetes peut être usurpée pour faire usage de techniques d'extraction de cryptomonnaies par des méthodes de minage.

Une autre méthode d'agression par un cybercriminel est le déni des services publics offerts par les différentes solutions Kubernetes d'une organisation, provoquant des temps d'arrêt, voire des pertes financières.

La sécurité de Kubernetes peut être divisée en trois points :

La sécurité dans Kubernetes se divise en trois points : protection des données, prévention de l'utilisation abusive des ressources et prévention des attaques par déni de service.

- **Éléments qui composent le système et son infrastructure.**
- **Configurations des composants Kubernetes.**
- **Incidents dans les temps d'exécution.**

3.1 La sécurité des infrastructures

3.1.1 Virtualisation

L'utilisation de la technologie de virtualisation pour faire fonctionner un système de conteneurs peut fournir des couches supplémentaires de sécurité pour l'exploitation de différentes solutions et services. Un tel système se voit attribuer une « fraction » des ressources totales du serveur physique, de sorte qu'il ne peut accéder à l'ensemble des ressources de l'hôte où il est hébergé, évitant ainsi un déni total du système. L'isolation des conteneurs est assurée.

La virtualisation des systèmes offre la possibilité d'effectuer des « points de restauration », en cas de nécessité de revenir à un point dans le temps antérieur à une configuration appliquée au système.

3.1.2 Protection basée sur le noyau

L'utilisation d'outils du noyau, tels que « seccomp », peut être utilisée pour limiter les appels effectués par un conteneur sur le système, réduisant ainsi l'exposition à un attaquant potentiel.

Un autre outil pour la protection du noyau du système est SElinux. L'utilisation du contexte de sécurité fourni par SElinux offre au conteneur une protection contre les changements non désirés et un niveau plus élevé d'audit de sécurité.

3.1.3 Politiques et segmentation du réseau

Les conteneurs d'un pod partagent une adresse IP et un port, ces adresses IP doivent être déclarées dans les fichiers de configuration « localhost » afin d'établir des connexions.

Sachant que les échanges d'informations entre les pods et les services se font par le biais de communications réseau, la segmentation du réseau est un point à prendre en compte, afin d'établir des connexions entre les différents éléments de manière sécurisée, en isolant les conteneurs et les services nécessaires.

Il faut mettre en œuvre une politique de réseau correcte, en la délimitant par l'utilisation de pare-feu et le cryptage du trafic autorisé pour la protection du trafic de données entre les différents points du système.

Le cryptage des communications doit être effectué à l'aide de certificats TLS, dans leurs versions TLS 1.2 ou TLS 1.3, versions de protocoles considérées comme sûres.

Le tableau suivant présente les principaux paramètres de configuration pour les communications chiffrées par certificat.

Fichier	Paramètre	Valeur	Remarques
kube-apiserver.yaml	etcd-certfile	[chemin]	Connexion cryptée entre APIserver et etcd
kube-apiserver.yaml	etcd-keyfile	[chemin]	Connexion cryptée entre APIserver et etcd
kube-apiserver.yaml	Fichier tls-cert	[chemin]	Utilisation du trafic crypté par le serveur d'API
kube-apiserver.yaml	tls-private-key-file	[chemin]	Utilisation du trafic crypté par le serveur d'API
etcd.yaml	auto-tls	false	Éviter d'utiliser des certificats auto-signés
etcd.yaml	peer-client-file	[chemin]	Configuration TLS pour etcd
etcd.yaml	peer-key-file	[chemin]	Configuration TLS pour etcd
etcd.yaml	peer-client-cert-auth	true	Authentification sécurisée etcd
etcd.yaml	peer-auto-tls	false	Éviter d'utiliser des certificats auto-signés

Fichier	Paramètre	Valeur	Remarques
etcd.yaml	trusted-ca-file	[chemin]	Utilisation de l'autorité de certification
10-kubeadm.conf	Fichier tls-cert	[chemin]	Utilisation du trafic crypté pour le Worker Node
10-kubeadm.conf	tls-private-key-file	[chemin]	Utilisation du trafic crypté pour le Worker Node
10-kubeadm.conf	RotateKubeletServerCertificate	[chemin]	Rotation des certificats pour le serveur kubelet

Tableau 1 - Configurations TLS.

Pour créer une politique de réseau, un plug-in d'interface de conteneur (CNI), qui prend en charge les API de NetworkPolicy, est nécessaire.

Chaque pod du cluster Kubernetes possède sa propre IP privée dans le cluster et peut être traité comme un véritable ordinateur où un port est attribué à l'adresse IP (socket). Ces sockets peuvent héberger des ressources et l'accès aux applications d'une organisation, il faut donc un élément qui assure la stabilité des éléments du réseau entre les pods Kubernetes, car, dans un processus, par exemple, de changement ou de mise à jour des pods, ces adresses IP peuvent être modifiées entraînant une perte de connexion ou une perte d'accès à une ressource.

Kubernetes résout ces pertes de connexion potentielles dues au changement d'IP en unifiant les adresses IP en ensembles logiques de pods grâce à des services qui sont étiquetés.

La manière d'accéder à ces services depuis une source externe à Kubernetes se fait généralement par le biais de « NodePorts », en accordant de manière aléatoire un port à une IP, qui peut être configurée de manière statique.

Pour une segmentation adéquate du réseau, les modules complémentaires CNI de Kubernetes peuvent utiliser des nodeports ou des pare-feu, ainsi que des équilibreurs de charge. De cette façon, la segmentation du réseau limite la surface d'attaque d'un cybercriminel.

3.1.4 Espaces de noms des conteneurs

Les « espaces de noms » sont un moyen de délimiter les ressources Kubernetes en attribuant une étiquette à une portée de ressources de cluster, formant ainsi une unité sur laquelle un ensemble d'autorisations RBAC (contrôle d'accès basé sur les rôles) et de politiques de réseau peuvent être appliquées.

Il existe trois espaces de noms par défaut :

- **Système Kube. Utilisé pour les composants Kubernetes.**
- **Kube-public. Utilisé pour étiqueter les ressources publiques.**
- **Par défaut. Ce sont des ressources pour les utilisateurs.**

3.1.5 Politiques de ressources

Une bonne pratique de gestion consiste à limiter l'utilisation des ressources matérielles physiques pour éviter un débordement des ressources du serveur Kubernetes.

Les restrictions peuvent être effectuées sur une base par conteneur ou par espace de nom. Les restrictions sont faites en créant des fichiers de politique avec l'extension « yaml » et en les appliquant à l'aide de la commande « kubectl ».

À titre d'exemple, une création de politiques de limitation des ressources et la manière de les appliquer dans un cluster Kubernetes sont présentées.

Étape	Description
1.	Créer un espace de nom pour que les ressources soient isolées du reste de votre cluster. `$ kubectl create namespace quota-mem-cpu-name`

2.	Créer le fichier de configuration de la politique avec l'extension « yaml » [policy.yaml]
	`apiVersion : v1`
	`Type : ResourceQuota`
	`métadonnées :`
	`  nom : mem-cpu-demo`
	`spéc :`
	`  dur :`
	`  requests.cpu : « 1 »`
	`  Mémoire des demandes : 1Gi`
	`  limites.cpu : « 2 »`
	`  Mémoire limite : 2Gi`

3.	Appliquer la politique de limitation des ressources avec la commande suivante. `$ kubectl apply -f [policy.yaml] --namespace=quota-mem-cpu-name`
4.	Il est possible de vérifier l'application de la politique avec la commande suivante. `$ kubectl get quota --namespace=quota-mem-cpu-name`

3.1.6 Fortification du plan de contrôle

C'est le cœur de Kubernetes, il gère les conteneurs, les différentes tâches de maintenance, ainsi que la gestion des informations sensibles du cluster, ce qui en fait un point sensible pour fortifier l'ensemble de l'écosystème Kubernetes.

Les communications doivent être cryptées à l'aide des protocoles TLS (versions 1.2 et 1.3), disposer d'un contrôle d'accès basé sur les rôles (RBAC) et de méthodes d'authentification forte avec une complexité suffisante des mots de passe.

Une autre mesure est le contrôle d'accès au réseau des communications effectuées par l'API Kubernetes, en limitant au moyen de pare-feu les types de connexions et leur adressage, afin d'établir une utilisation correcte de l'API avec la plus grande fiabilité possible.

Vous trouverez ci-dessous un tableau présentant les principaux ports utilisés par l'API Kubernetes, ainsi que leurs protocoles.

Nœud de composant	Orientation trafic	Protocole	Ports
Kube-controller-manager	Entrant	TCP	10257
Kube-scheduler (planificateur de tâches)	Entrant	TCP	10259
API Kubernetes	Entrant	TCP	6443
API etcd	Entrant	TCP	2379-2380
API kubelet	Entrant	TCP	10250

Tableau 2 - Trafic réseau de l'API Kubernetes.

Les fichiers de configuration locaux qui gèrent les composants du nœud de contrôle doivent bénéficier d'une gestion des autorisations et des accès des utilisateurs.

Fichier	Permis	Propriétaire
kube-apiserver.yaml	644 ou plus restrictif	root:root
kube-controller-manager.yaml	644 ou plus restrictif	root:root
kube-scheduler.yaml	644 ou plus restrictif	root:root
etcd.yaml	644 ou plus restrictif	root:root
Configurations CNI	644 ou plus restrictif	root:root
BBDD_etcd	644 ou plus restrictif	root:root
admin.conf	644 ou plus restrictif	root:root
scheduler.conf	644 ou plus restrictif	root:root
controller-manager.conf	644 ou plus restrictif	root:root
certificats pki (extension crt)	640	root:root
clés certifiées pki (clé d'extension)	400	root:root

Tableau 3 - Adhésion et autorisations de fichiers do nœud de contrôle.

Remarque: L'emplacement par défaut des fichiers « yaml » est /etc/kubernetes/manifests/. L'emplacement par défaut des fichiers « conf » est "/etc/kubernetes/". Pour plus d'informations, voir:

https://Kubernetes.io/docs

3.1.7 La fortification des nœuds de travail

Comme pour le nœud maître, un contrôle d'accès au réseau doit être mis en place pour chaque nœud de travail du cluster Kubernetes. En identifiant la direction du trafic réseau de chaque nœud, vous pouvez limiter l'accès aux nœuds à l'aide d'un pare-feu en les configurant via leurs sockets.

Les ports et services utilisés par les nœuds de travail Kubernetes sont indiqués ci-dessous.

Nœud de composant	Orientation du trafic	Protocole	Ports
Services de nœuds	Entrant	TCP	30000-32767
API kubelet	Entrant	TCP	10250

Tableau 4 - Trafic réseau du nœud de travail.

Les fichiers de configuration locaux qui gèrent les composants du nœud de contrôle doivent bénéficier d'une gestion des autorisations et des accès des utilisateurs.

Fichier	Permis	Propriétaire
10-kubeadm.conf	644 ou plus restrictif	root:root
Fichier « kubeconfig » de kube-proxy	644 ou plus restrictif	root:root
kubelet.conf	644 ou plus restrictif	root:root

Tableau 5 - Composition du nœud de travail et autorisations de fichiers.

NOTE : Pour plus d'informations, veuillez consulter les liens suivants :

https://Kubernetes.io/docs/admin/kube-proxy/

https://Kubernetes.io/docs/admin/kubelet/

https://Kubernetes.io/docs/tasks/administer-cluster/kubelet-config-file/

3.1.8 Informations sensibles

Les informations sensibles dans Kubernetes (secrets), telles que les mots de passe ou les certificats, sont placées pour être utilisées dans les configurations dans des fichiers « yaml », des images de conteneurs ou des variables d'environnement, sachant qu'il est plus sûr d'utiliser leur contenu comme un fichier, et non comme une variable, puisque l'accès peut être régulé par les permissions et les utilisateurs dans un fichier.

Ces données sensibles sont stockées par Kubernetes en clair, il les encode en base64 et peut les crypter en configurant son chiffrement de données dans l'API du serveur.

L'activation des paramètres de cryptage du serveur API est présentée ci-dessous à titre d'exemple.

Fichier	Paramètre à configurer	Valeur
kube-apiserver.yaml	--encryption-provider-config	/[emplacement fichier]/secrets.yaml

Tableau 6 - Configuration d'Apiserver.

NOTE : Pour plus d'informations, veuillez consulter le lien suivant.

https://Kubernetes.io/docs/tasks/administer-cluster/encrypt-data/.

3.1.9 Mises à jour

Au fil du temps, les logiciels et les applications de l'environnement Kubernetes sont susceptibles de présenter des failles de sécurité en raison de l'évolution continue des attaques et de l'apparition de vulnérabilités et d'expositions à la sécurité. C'est pourquoi il peut être nécessaire de les mettre à jour et/ou d'appliquer des correctifs, quel que soit le support sur lequel ils sont déployés, qui doit à son tour appliquer les mises à jour et/ou les correctifs de sécurité publiés par leurs fournisseurs respectifs.

3. Sécurité de Kubernetes

Kubernetes offre une cadence de publication majeure de trois (3) mises à jour mineures par cycles annuels, soutenue par des correctifs de sécurité pendant environ un an après la publication.

Les versions de Kubernetes sont exprimées par une numérotation divisée en trois blocs et séparée par des chiffres.

- **Premier bloc.** Il s'agit de la version principale ou majeure de la version de Kubernetes.
- **Deuxième bloc.** Représentée comme la version mineure ou secondaire.
- **Troisième bloc.** Indique le numéro de version du correctif appliqué à l'application Kubernetes.

Lorsqu'une application subit une mise à jour, et notamment lorsqu'il s'agit d'une mise à jour de version majeure, il est généralement nécessaire de redémarrer le cluster afin d'appliquer correctement les modifications, ce qui entraîne une interruption des services déployés par la solution Kubernetes. Pour éviter ces interruptions de service, il est parfois nécessaire d'installer un système de haute disponibilité afin d'obtenir une redondance des services.

Il convient de noter que, dans un environnement de production, l'application de changements majeurs aux applications à la suite d'une mise à jour peut rendre l'application mise à jour incompatible avec sa configuration avant la mise à jour. Les mises à jour doivent être appliquées au préalable dans un environnement de pré-production ou de test, en vérifiant le bon fonctionnement des applications déployées, et une fois que leur bon fonctionnement a été corroboré, appliquer les mises à jour et les changements de configuration à l'environnement de production.

3.2 Sécurité des conteneurs et pods

3.2.1 Contrôle d'accès aux pods

L'accès à un cluster Kubernetes se fait par l'authentification des utilisateurs. Pour éviter les attaques par usurpation d'identité et pouvoir effectuer des appels non autorisés à l'API Kubernetes, l'authentification des utilisateurs prend en charge différents mécanismes d'accès qui ne sont pas activés par défaut.

Il y a une différenciation à faire dans les utilisateurs d'un système Kubernetes, avec deux types de comptes :

- **Comptes de service.**
- **Comptes nominatifs pour les utilisateurs.**

Les pods effectuent chacun des requêtes et des appels à l'API Kubernetes. Cette demande est faite avec un compte de service attaché au pod, ce qui permet de faire l'appel de « contrôleur d'admission » de « ServiceAccount ». Un compte de service doit être associé à chaque espace de noms, sinon le gestionnaire d'admission attachera un compte par défaut à l'espace de noms.

Les comptes de service doivent être créés individuellement et recevoir des autorisations spécifiques afin de délimiter leurs fonctions.

Les comptes de service par défaut ne doivent pas être utilisés, car un ensemble de permissions leur est attribué par défaut. De même, lorsqu'un compte par défaut est créé, il reçoit un jeton lui permettant d'accéder à l'API de manière incontrôlée.

Pour empêcher la création de jetons pour des comptes de service par défaut et pour empêcher l'attribution de permissions, Kubernetes dispose d'un mécanisme de contrôle à cet effet.

Fichier	Paramètre à configurer	Valeur
/var/run/secrets/Kubernetes.io/serviceaccount/token	automountServiceAccountToken	faux

Tableau 7 - Configuration de l'accès aux pods.

Pour les utilisateurs ou administrateurs nommés, une méthode d'authentification spécifique doit être appliquée. Les méthodes d'authentification dans le système Kubernetes sont les suivantes :

- **Certificats X509**
- **Jetons Bootstrap**
- **Jetons OpenID**

Vous devez disposer d'une politique de mots de passe et de méthodes d'authentification d'une complexité suffisamment élevée pour empêcher tout accès involontaire, en évitant les demandes ou formes d'accès anonymes. Dans les versions 1.6 et ultérieures de Kubernetes, les demandes anonymes sont activées par défaut dans le système.

Le refus d'accès anonyme doit être configuré dans le serveur API.

Fichier	Paramètre à configurer	Valeur
/etc/Kubernetes/manifests/kube-apiserver.yaml	auth. anonyme	faux

Tableau 8 - Configuration de l'accès à Apiserver.

3.2.2 Contrôle d'accès basé sur les rôles

Le contrôle d'accès basé sur les rôles (RBAC) est utilisé pour gérer l'accès des utilisateurs aux ressources du cluster.

Vous devez avoir le contrôle RBAC activé dans l'API afin de pouvoir contrôler et gérer correctement les ressources du cluster. Pour initialiser le contrôle d'accès basé sur les rôles, exécutez la commande suivante.

```
$ kube-apiserver --authorization-mode=RBAC
```

Dans une politique de sécurité de rôle d'accès médiocre, les politiques « AlwaysAllow » apparaissent. Ces paramètres de configuration doivent être examinés et les marges de la politique de rôle d'accès correcte définies au niveau de l'opérabilité de l'utilisateur au sein du cluster, en imposant toujours une politique de moindre privilège sur l'accès au système.

Deux types de permissions peuvent être établis :

- **Rôles.** Définit les permissions pour un « espace de noms » particulier.
- **Rôles des clusters.** Définit les permissions sur toutes les ressources du cluster, indépendamment des « espaces de noms ».

Les rôles sont utilisés pour ajouter des permissions, ils n'ont pas de règles de refus. Le serveur API Kubernetes refusera les autorisations qui ne sont pas explicitement autorisées.

Une fois que les rôles ou les groupes de rôles ont été créés, « RoleBinding » est utilisé pour les associer à un utilisateur ou à un groupe.

« RoleBinding » ou « ClusterRoleBinding » est utilisé lorsque vous souhaitez gérer un ensemble de permissions similaires pour différents espaces de noms.

Les privilèges attribués aux utilisateurs, aux groupes et aux comptes de service doivent respecter le principe du moindre privilège, en accordant des autorisations uniquement pour effectuer une tâche, ainsi que les rôles.

3.2.3 Conteneurs immuables

Dans le cas où l'un des conteneurs est compromis et qu'un attaquant en a pris le contrôle, il pourrait supprimer des fichiers, créer des scripts ou même modifier les paramètres de configuration d'un composant du conteneur.

Tous les scénarios ci-dessus peuvent être évités en verrouillant le système de fichiers du conteneur par Kubernetes, mettant le système dans un état de lecture seule. Cependant, ce mode peut entraîner un comportement anormal dans le fonctionnement du conteneur, donc s'il est mis en œuvre, une étude préalable doit être réalisée afin d'obtenir un résultat de fonctionnement optimal.

Fichier	Paramètre à configurer	Valeur
/etc/Kubernetes/manifests/kube-apiserver.yaml	readOnlyRootFilesystem	vrai

Tableau 9 - Configuration de l'accès à Apiserver.

3.2.4 Création de conteneurs

Une image de conteneur peut être créée à partir de zéro, ou elle est généralement téléchargée à partir d'un référentiel et des modifications peuvent être apportées sur la base de cette image.

Lorsque des images sont téléchargées à partir de référentiels, ces référentiels doivent provenir de sources fiables.

Vous pouvez configurer l'environnement Kubernetes pour restreindre le déploiement vers le cluster des images de conteneurs qui ne possèdent pas de signature numérique provenant d'une source de confiance.

Lorsqu'on insère une image dans l'écosystème Kubernetes, il faut vérifier qu'elle ne contient pas de bibliothèques dépréciées, de vulnérabilités connues, ni de configurations qui violent la sécurité des ports ou des autorisations non sécurisées.

3.2.5 Sécurité de pods

Kubernetes dispose de deux mécanismes pour sécuriser les pods de cluster. Ces mécanismes sont :

- **Admission de sécurité de pod.**
- **Politiques de sécurité de pod (PSP).**

Le « Pod Security Admission », une politique de sécurité mise en œuvre par Kubernetes, est une politique de sécurité par défaut à partir de la version 1.23. Les pods sont classés (basique, restreint ou privilégié), ce qui permet une mise en œuvre plus simple que dans PSP.

Lorsque la fonction « Admission de sécurité de pod » est activée, les espaces de noms peuvent être configurés pour définir un ensemble de modes de contrôle d'admission pour l'utilisation de la sécurité dans le pod et par espace de noms.

Kubernetes définit un ensemble de balises permettant de définir des niveaux normalisés de sécurité des pods.

Mode	Description
Appliquer	Toute police en infraction sera rejetée par le pod.
Auditer	Les violations de la politique seront appliquées, mais elles seront enregistrées par les pistes d'audit du système.
Avertir	Les violations de la politique seront notifiées à l'utilisateur, mais les changements dans le système seront appliqués.

Tableau 10 - Étiquettes de politique de sécurité.

Dans les politiques de sécurité de pod (PSP), désormais obsolètes depuis la version 1.21, il valide la création des pods et met à jour les requêtes au PAI du système en utilisant un ensemble de règles pour leur gestion.

3.2.6 Sécurité dans les environnements Kubernetes

La sécurité de l'environnement qui héberge la solution est tout aussi importante que celle du cluster Kubernetes lui-même. L'outil de gestion de cluster en ligne de commande est « kubectl », qui peut être déployé sur les systèmes d'exploitation Windows, linux et macOS.

REMARQUE : des informations supplémentaires sur l'installation et le support du système d'exploitation sont disponibles sur le lien suivant :

https://Kubernetes.io/es/docs/tasks/tools/_print/

Si la conteneurisation hébergée par un hyperviseur est utilisée, la virtualisation limite l'utilisation du matériel, plutôt que le système d'exploitation sur lequel Kubernetes serait déployé.

L'isolation par hyperviseur est plus sûre que l'isolation traditionnelle par conteneur, car elle permet de séparer davantage les différents nœuds et de limiter les ressources du système à celles allouées par l'hyperviseur.

Dans une solution basée sur le noyau linux, l'outil seccomp peut être utilisé pour limiter les appels système des conteneurs, réduisant ainsi la surface d'attaque sur le noyau. Des outils similaires en termes de fonctionnalité peuvent être Selinux ou APParmor.

Certaines solutions de moteur de conteneur offrent une isolation entre l'application du conteneur et le noyau de l'hôte appelée « SandBox », confinant les ressources dans un espace virtuel isolé.

3.3 Incidents dans les temps d'exécution

3.3.1 Conteneurs no « root »

Par défaut, certains services dans les conteneurs sont exécutés avec l'utilisateur « root », un utilisateur ayant accès à toutes les ressources du système. Dans beaucoup de ces tâches, il n'est pas nécessaire d'utiliser cet utilisateur administrateur, mais tout autre utilisateur ayant une configuration correcte d'accès aux ressources du cluster peut le faire.

Un conteneur peut être configuré de manière à ce que certaines tâches ne soient pas exécutées par l'utilisateur « root ». Ce paramètre par défaut est généralement défini au moment de la création de l'image de conteneur.

Une alternative dans Kubernetes est d'utiliser un pod avec le paramètre « SecurityContext » défini sur « runAsUser », en spécifiant que l'identifiant de l'utilisateur n'est pas « 0 », puisque cet identifiant appartient à l'utilisateur « root ».

3.3.2 Sécurité à jetons pour les comptes de service

Par défaut, Kubernetes accorde un jeton « secret » à un compte de service lors de la création d'un pod. Ce jeton est accordé au sein du pod au moment de l'exécution du service qu'il exécute.

Toutes les applications dans les conteneurs ne nécessitent pas un accès direct aux comptes de service. Si une application est compromise par un attaquant, les intégrations et l'accès aux jetons peuvent être compromis et utilisés pour accéder à l'API du système.

Il convient d'étudier la nécessité d'accorder automatiquement un jeton à un compte de service. Pour éviter de créer des jetons pour des comptes de service par défaut et d'attribuer des autorisations, Kubernetes dispose d'un mécanisme de contrôle à cet effet.

Fichier	Paramètre à configurer	Valeur
/var/run/secrets/Kubernetes.io/serviceaccount/token	automountServiceAccountToken	false

Tableau 11 - Configuration de l'accès.

Si, par nécessité, cette mesure de sécurité ne peut pas être mise en œuvre, par exemple dans le cadre du provisionnement des authentifications aux services externes, il convient d'appliquer des politiques de contrôle d'accès basées sur les rôles, en réduisant au minimum les privilèges de pod au sein du cluster.

3.3.3 Pistes d'audit

Les journaux d'audit Kubernetes permettent de suivre les informations relatives à la sécurité de votre système. Les entrées de journal sont générées pour contrôler le plus d'informations possible sur les événements qui se produisent sur le système. Ces informations sont cruciales dans les environnements critiques pour déterminer qui viole les politiques de sécurité et quelles mesures ont été prises. L'audit n'apporte pas de sécurité supplémentaire à votre système, mais il peut être utilisé pour découvrir les violations des politiques de sécurité utilisées sur le système.

Ces violations peuvent être évitées par des mesures de sécurité supplémentaires qui, avec l'étude des événements collectés dans les journaux du système, aident à l'élaboration, telles que des politiques de contrôle d'accès basées sur les rôles.

Les journaux d'audit surveillent l'activité du cluster. Des journaux chronologiques sont tenus sur les questions de sécurité et sur les activités réalisées par les utilisateurs du système et d'autres composants, permettant ainsi de vérifier les faits d'un événement particulier.

Les administrateurs du système doivent mettre en œuvre une méthode de journalisation et de surveillance des ressources et de leur accès par tout utilisateur.

Certains des points les plus importants pour la surveillance de Kubernetes sont :

- **Appels API**
- **Mesures de la performance**
- **Consommation des ressources**
- **Trafic réseau**
- **Modification de l'image et du conteneur**
- **Changements de privilèges**
- **Création et modification du planificateur de tâches.**
- **Modification de la hiérarchie des fichiers.**

3.3.4 Logs du système

La capture des journaux d'événements dans le cadre des audits de sécurité doit être effectuée régulièrement, en tenant compte des historiques de journaux collectés précédemment et en vérifiant par recoupement les changements significatifs entre ces fichiers, ainsi qu'en effectuant une analyse des raisons pour lesquelles les changements ont été effectués. Si l'on détecte une consommation excessive des ressources du système qui constitue un changement significatif par rapport aux audits précédents, par exemple, cela peut indiquer un accès non autorisé d'un attaquant à la puissance de calcul du système.

Si l'organisation dispose d'un collecteur d'événements centralisé (syslog), d'un système de gestion des informations et des événements de sécurité (SIEM), la transmission de ces journaux de sécurité peut aider à détecter les anomalies des clusters au plus près du temps d'exécution.

Lorsque des informations ou des enregistrements de journaux sont transférés vers un système centralisé, la connexion doit être sécurisée à l'aide de certificats, tels que TLS versions 1.2 et 1.3, afin de garantir l'intégrité des informations transférées d'un point à un autre.

Lorsque vous utilisez un serveur de logs externe à Kubernetes, vous devez configurer le forwarder de logs avec un accès en append-only au stockage externe. Cela protège les journaux stockés en externe contre la suppression ou l'écrasement à partir du cluster.

3.3.5 Journaux d'audit de Kubernetes

Dans le cluster Kubernetes, chaque fois qu'une demande est faite par le front-end kubi-apiserver, un journal des événements est produit, soit par une demande d'un utilisateur, d'une application ou par le plan de contrôle. Lorsqu'un événement est enregistré, le kube-apiserver recherche la première correspondance dans ses politiques d'audit pour effectuer la classification de l'événement, pas par défaut.

Les fichiers de politique de journalisation sont stockés au format « yaml », et sont utilisés pour définir les règles et spécifier quel niveau d'audit doit être enregistré pour un événement qui correspond à une politique précédemment créée.

```yaml
apiVersion: audit.k8s.io/v1 # Esto es obligatorio.
kind: Policy
# No generar eventos de auditoría para las peticiones en la etapa RequestReceived.
omitStages:
  - "RequestReceived"
rules:
  # Registrar los cambios del pod al nivel RequestResponse
  - level: RequestResponse
    resources:
    - group: ""
      # Los recursos "pods" no hacen coincidir las peticiones a cualquier sub-recurso de pods,
      # lo que es consistente con la regla RBAC.
      resources: ["pods"]
  # Registrar "pods/log", "pods/status" al nivel Metadata
  - level: Metadata
    resources:
    - group: ""
      resources: ["pods/log", "pods/status"]

  # No registrar peticiones al configmap denominado "controller-leader"
  - level: None
    resources:
    - group: ""
      resources: ["configmaps"]
      resourceNames: ["controller-leader"]
```

Figure 6 - Extrait de la politique du registre Kubernetes.

NOTE: Pour de plus amples informations sur ce point, veuillez consulter le lien suivant.

https://Kubernetes.io/es/docs/tasks/debug-application-cluster/audit/

Pour qu'une politique ou une règle soit considérée comme valide, elle doit avoir l'un des niveaux d'audit gérés par le cluster Kubernetes.

- **AUCUN.** Les événements qui exécutent la règle de journalisation ne sont pas journalisés.
- **METADATA.** Les métadonnées de la demande, comme l'horodatage, sont enregistrées, mais pas la demande elle-même, ni la réponse à la demande.
- **RESQUEST.** Les métadonnées et la requête qui déclenchent l'exécution de l'événement sont enregistrées, mais la réponse ne l'est pas. Ceci ne s'applique pas aux demandes de non-ressources.
- **RESQUESTRESPONSE.** Les métadonnées, la demande et la réponse sont enregistrées. Ceci ne s'applique pas aux demandes sans recours.

Le niveau maximal des pistes d'audit est « RESQUESTRESPONSE », fournissant le maximum d'informations disponibles en cas d'événement à surveiller.

Les journaux ayant le niveau maximal de « RESQUESTRESPONSE » peuvent entraîner la capture d'objets « secrets » où des informations sensibles, telles que des mots de passe, peuvent être stockées en cryptage base64. En outre, elle génère une quantité considérable d'enregistrements, notamment dans les clusters en production, ce qui peut entraîner des incidents de manque d'espace.

Il convient d'établir une étude, pour une mise en œuvre ultérieure, des politiques dans les journaux d'audit du cluster Kubernetes de l'organisation, en oscillant les politiques avec les différents niveaux d'audit et avec une surveillance spéciale des processus ou des appels critiques pour le bon fonctionnement de l'application.

Kube-apiserver, dispose d'une série de fonctionnalités et de paramètres de configuration pour le traitement, le stockage et la rotation des journaux ou des pistes d'audit générés par les événements de la politique de sécurité de Kubernetes. « Webhook » est un backend et un endpoint http, que kubernetes invoquera au moment de l'exécution d'un événement, configuré pour envoyer à une API http externe les logs ou les informations souhaitées.

NOTE: Pour de plus amples informations sur ce point, veuillez consulter le lien suivant.

https://Kubernetes.io/docs/tasks/debug/debug-cluster/audit/

Fichier	Paramètre	Valeur	Remarques
kube-apiserver.yaml	audit-log-path	[chemin]	Emplacement des pistes d'audit
kube-apiserver.yaml	audit-log-maxage	[Valeur en jours]	Rotation des pistes d'audit
kube-apiserver.yaml	audit-log-maxbackup	[Valeur en jours]	Conservation des dossiers d'audit
kube-apiserver.yaml	audit-log-maxsize	[Valeur en méga-octets]	Taille maximale des pistes d'audit

Tableau 12 - Configuration du journal.

3.3.6 Journal des conteneurs et des nœuds de travail

Kubelet effectue la journalisation d'audit sur chaque nœud individuellement dans chaque conteneur, en gérant les journaux, en les stockant et en les faisant tourner selon les paramètres de la politique.

Kubelet, est un outil de console de commande géré par la commande kubectl.

Commande	Paramètre	POD	Remarques
kubectl	--namespace [nom]	logs [POD]	Affichage des enregistrements des conteneurs

Tableau 13 - Commande kubectl.

3.3.7 Audit par le Seccomp

Tous les types de journaux mentionnés ci-dessus peuvent être complétés par des journaux d'audit des appels au noyau qui se produisent. À cette fin, l'application « seccomp » peut être utilisée pour auditer les appels de conteneurs dans l'écosystème Kubernetes.

Seccomp est désactivé par défaut, en l'activant et en le configurant, il est capable de limiter les appels système par les conteneurs, réduisant ainsi la surface d'attaque du système. Une autre fonctionnalité consiste à enregistrer ou à consigner tous les appels faits au noyau.

Fichier	Paramètre	Valeur	Remarques
Kubelet-config.yaml	SeccompDefault	true	Activation seccomp

Tableau 14 - Commande kubectl.

Les configurations du seccomp se font par profils, en autorisant, refusant ou enregistrant les appels. Vous devrez configurer les profils seccomp, avec les paramètres qui répondent le mieux aux besoins de votre organisation.

Fichier	Paramètre	Remarques
/var/lib/kubelet/seccomp/[profil]	[configuration seccomp pour un pod]	Activation et configuration de seccomp

Tableau 15 - Configuration de seccomp.

Avec l'aide de seccomp, il est possible d'identifier les appels système nécessaires aux opérations standardisées du cluster, et donc d'utiliser les modèles d'opérations de pod pour identifier tout modèle de comportement anormal et identifier les activités malveillantes.

3.3.8 Détection des menaces

L'utilisation de collecteurs de journaux, tels que journald, syslog et rsyslog, accélère la collecte d'informations sur le système et aide à trouver des modèles pour la détection de menaces pour le système.

La surveillance continue des journaux et des ressources système au moment de l'exécution est un outil très puissant pour comprendre le fonctionnement de Kubernetes et détecter les anomalies dans les processus, et ainsi pouvoir déterminer si des agents malveillants ont attaqué ou attaquent le système. Pour de telles tâches, l'utilisation d'outils externes tels qu'un SIEM peut être d'une grande aide pour une organisation, non seulement en termes de sécurité, mais aussi en termes d'opérabilité du système, puisqu'il est capable de visualiser en temps réel la gestion des ressources par Kubernetes.

4. Liste de vérification

Criticité	Description
Élevée	L'ordinateur hôte qui héberge l'application Kubernetes doit avoir son système mis à jour, tous les correctifs de sécurité publiés par le fabricant du produit appliqués, et son cycle de vie en vigueur.
Élevée	Les dernières mises à jour et les derniers correctifs de sécurité publiés par le fabricant doivent être installés sur Kubernetes.
Élevée	Les connexions entre les nœuds et vers les nœuds sont cryptées par des certificats.
Élevée	Les communications sont cryptées à l'aide des protocoles TLS dans les versions 1.2 ou 1.3.
Élevée	Les fichiers de configuration du « plan de contrôle » ne peuvent être modifiés que par des utilisateurs administrateurs.
Élevée	Les certificats et leurs générateurs de clés dans Kubernetes appartiennent à l'utilisateur « root » et au groupe « root ».
Élevée	Seuls les utilisateurs administrateurs peuvent accéder aux certificats et à leurs clés de générateur Kubernetes.
Élevée	Les fichiers de configuration de « nœuds de travail » ne peuvent être modifiés que par des utilisateurs administrateurs.

4. Liste de vérification

Criticité	Description
Élevée	La connexion à l'environnement Kubernetes n'est établie que par les utilisateurs authentifiés.
Élevée	Les journaux d'audit sont configurés dans Kubernetes.
Élevée	Les conteneurs ont un contrôle d'accès basé sur les rôles et des espaces de noms de conteneurs sont définis.
Moyenne	Les fichiers de configuration du « plan de contrôle » appartiennent à l'utilisateur « root » et au groupe « root ».
Moyenne	Les fichiers de configuration des « nœuds de travail » doivent appartenir à l'utilisateur « root » et au groupe « root ».
Moyenne	Les informations sensibles appelées « secrets » sont cryptées.
Moyenne	Les comptes de service des pods (ServiceAccount) sont configurés avec des autorisations spécifiques pour effectuer leurs tâches.
Moyenne	Les politiques de sécurité pop (PSP) sont configurées et actives.
Moyenne	Les comptes de service sont configurés pour ne pas recevoir automatiquement un jeton d'authentification.
Moyenne	Les journaux d'événements sont configurés dans Kubernetes en fonction des niveaux d'audit.
Moyenne	Une politique de stockage des journaux d'audit a été configurée.

Criticité	Description
Moyenne	Les journaux d'appels du noyau sont configurés dans Kubernetes.
Faible	Les pistes d'audit sont stockées sur un support externe.
Faible	Les ressources du cluster Kubernetes sont configurées avec une segmentation basée sur les quotas.
Faible	Si les conteneurs ne doivent pas être modifiés, ils sont considérés comme « immuables ».

5. Décalogue de recommandations

Voici dix recommandations de sécurité pour l'utilisation de Kubernetes.

Dix recommandations pour Kubernetes

1. Il est recommandé de **toujours utiliser la dernière version stable** avec les dernières mises à jour recommandées par le fabricant, éliminant ainsi les vecteurs d'attaque avec les remèdes appliqués par le fabricant.

2. **L'utilisation d'algorithmes cryptographiques sécurisés** est recommandée. L'utilisation d'un algorithme de cryptage robuste et fiable élimine la possibilité d'interception des messages du système.

3. Il est recommandé d'**utiliser des sources officielles et/ou de confiance** pour déployer des images de conteneurs. L'environnement Kubernetes gère des conteneurs qui remplissent certaines fonctions pour une organisation. Ces conteneurs peuvent être créés par l'organisation ou téléchargés à partir de sources officielles et fiables pour être déployés sur le système.

4. Il est recommandé de **toujours utiliser des protocoles sécurisés (TLS)**, pour sécuriser les communications de bout en bout, afin d'éviter l'interception des informations.

5. **L'utilisation de politiques de mots de passe forts et complexes** est recommandée, avec une longueur minimale de caractères, en plus de l'utilisation de lettres majuscules, de symboles minuscules et de chiffres, ainsi qu'une période de validité du mot de passe.

6. **L'utilisation d'images minimales et actualisées** est recommandée. L'utilisation d'un nombre minimal d'images permet de consommer moins de ressources et de réduire le vecteur d'attaque.

7. Il est recommandé d'**évaluer les privilèges utilisés par les conteneurs.** Le principe de la fonctionnalité minimale et du privilège minimal doit être respecté.

8. **La ségrégation physique du réseau est** recommandée. Optimisation des ressources et segmentation des éléments du réseau, limitant l'accès aux informations et empêchant la propagation des incidents de sécurité.

9. Il est recommandé **de séparer les rôles et les autorisations des utilisateurs.** Les besoins d'exécution au sein de Kubernetes et les besoins d'accès aux informations gérées sont définis pour chaque utilisateur.

10. Il est recommandé de **documenter la plateforme Kubernetes.** L'infrastructure Kubernetes peut être très importante, étant donné qu'elle héberge plusieurs nœuds de contrôle et de travail. La documentation du système permet d'identifier rapidement les conteneurs inutilisés ou sous-utilisés, ce qui permet d'optimiser les ressources et de mieux gérer le système. Il est conseillé de mettre à jour cette documentation à chaque modification pertinente apportée au système.

CCN centro criptológico nacional

CCN-CERT centro criptológico nacional

6. Glossaire

Terme	Description
K8s	Abréviation de Kubernetes.
Pod	Un ensemble d'un ou plusieurs conteneurs, et une spécification de la façon d'exécuter ces conteneurs.
Gestionnaire de contrôleur de nuage (CSP)	Composant facultatif utilisé pour les déploiements basés sur le cloud. Le contrôleur de nuage interagit avec le fournisseur de services en nuage (CSP) pour gérer les équilibreurs de charge et les réseaux virtuels pour le cluster.
ETCD	Unité de stockage persistante où toutes les informations sur l'état du cluster sont sauvegardées.
SELinux	Module de sécurité pour le noyau Linux.
AppArmor	Module de sécurité pour le noyau Linux.
RBAC	Contrôle d'accès basé sur les rôles.
Seccomp	Une fonction du noyau Linux qui vous permet de limiter le nombre d'appels système qu'un processus peut faire.
API	Interface de programmation d'applications.
CSP	Fournisseur de services certifié.
CA	Organisme de certification.
TLS	Protocole cryptographique utilisé dans les réseaux. Sécurité de la couche de transport.
HTTPS	Protocole de transfert hypertexte sécurisé.
Docker ou conteneur	Projet open source qui automatise le déploiement d'applications.
on-premise	Installation locale de logiciels ou de matériel.
Plan de contrôle	Un ensemble de composants s'exécutant sur un seul nœud du cluster Kubernetes et contrôlant divers aspects du cluster.
Nœuds de travail	Ensemble de machines exécutant les tâches demandées et assignées par le plan de contrôle.
Kube-apiserver	Il est chargé de l'interaction avec les différents utilisateurs du système Kubernetes.

6. Glossaire

Terme	Description
Kube-sheduler	Il ordonne et planifie les priorités dans la gestion des groupes.
Kube-controller-manager	Passe en revue les processus de Kubernetes.
Pod	Un ensemble d'un ou plusieurs conteneurs déployés sur un seul nœud. C'est l'objet le plus petit et le plus simple de Kubernetes.
Kubectl	Interface de ligne de commande où vous pouvez gérer votre cluster Kubernetes.
Kubelet	Application de chaque nœud, qui communique avec le plan de contrôle.
Kube-proxy	Proxy réseau Kubernetes.
Yaml	Format de sérialisation des données lisible par l'homme, inspiré de différents langages de programmation.
NodePort	Accès aux nœuds par un port.
Namespace ou espace de noms	Clusters virtuels sauvegardés par un même cluster physique.
Cluster	Systèmes informatiques distribués reliés entre eux, généralement par un réseau à haut débit, et se comportant comme s'ils étaient un seul serveur.
Secrets	Objets secrets dans Kubernetes pour stocker et gérer des informations sensibles telles que des mots de passe, des jetons OAuth et des clés ssh.
hyperviseur	Moniteur de la machine virtuelle
journal	L'enregistrement séquentiel dans un fichier ou une base de données de tous les événements affectant un processus particulier.
Front-end	Interface utilisateur.
Back-end	Serveur ou moteur de services.

CCN
centro criptológico nacional

CCN-CERT
centro criptológico nacional

www.ccn.cni.es

www.ccn-cert.cni.es

oc.ccn.cni.es