Formation Kubernetes 1 Aout 2022

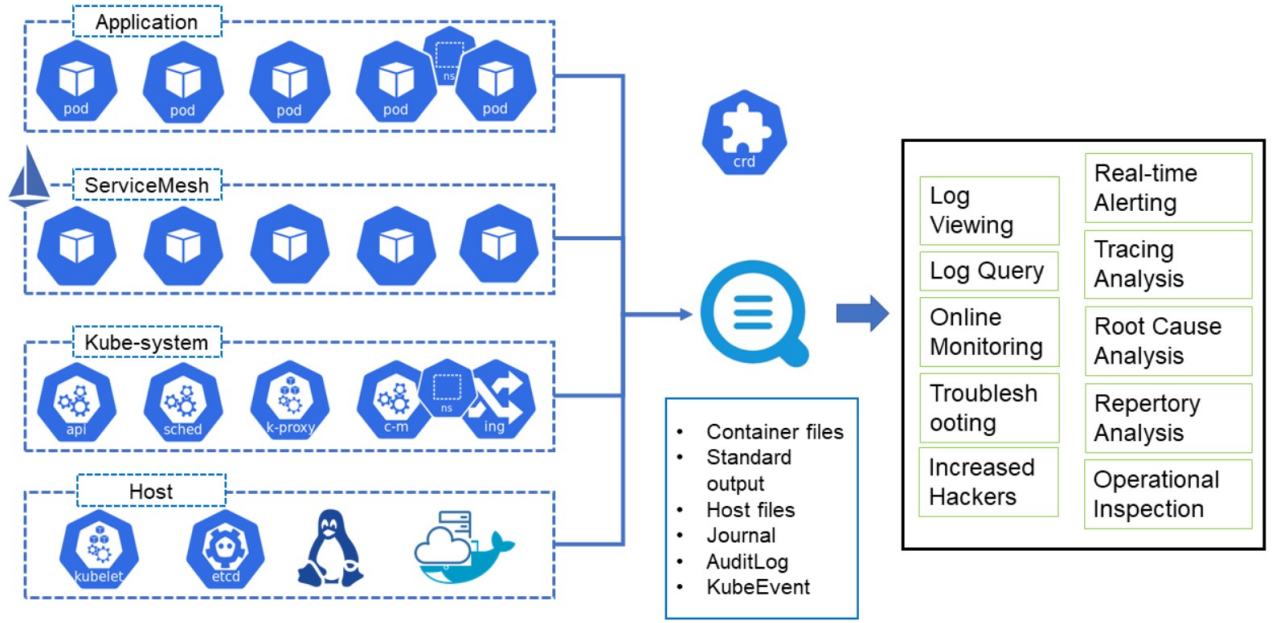
Ihab ABADI / UTOPIOS

Sommaire

- Logging et dépannage
- ressources personnalisées
- Sécurité
- Haute disponibilité
- Infrastructure RKE2

Kubernetes – Logging et dépannage

- Logging en Kubernetes peut être au niveau de :
  - Des conteneurs.
  - Des nœuds.
  - Du Cluster



Kubernetes – Logging et dépannage - Conteneurs

- Chaque application qui est exécutée dans un conteneur d'un pod peut écrire des logs stdout et stderr.
- Ces logs sont collectés par les kubelets et sauvegardés au niveau du worker.
- Pour récupérer ces logs, Kubernetes fournit l'api logs.
- Kubectl logs <pod_name>

Kubernetes – Logging et dépannage – nœuds

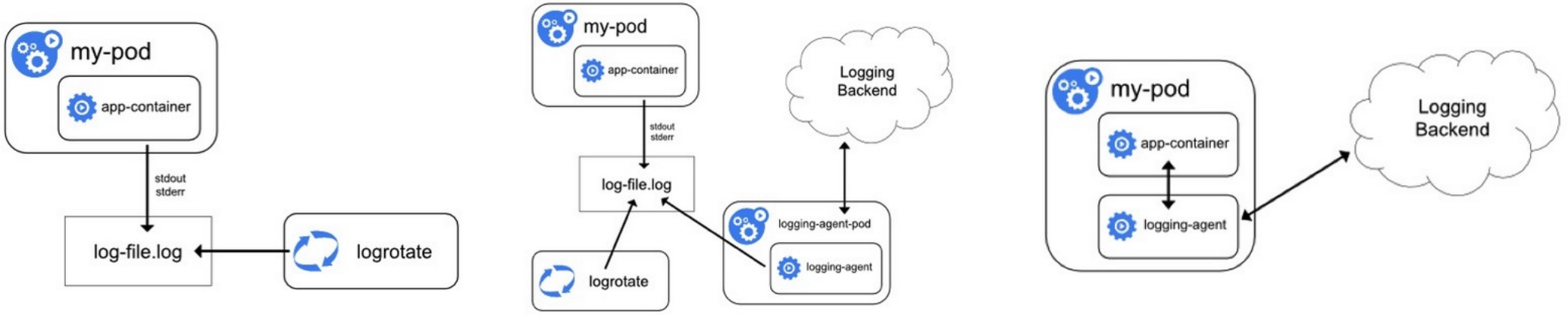
- Kubelet délègue la sauvegarde des logs au pods au runtime container.
- Chaque nœud enregistre les logs réseaux (kube-proxy).
- Les logs des conteneurs par exemple sont /var/log/container.
- Le reste des logs dans /var/log

Kubernetes – Logging et dépannage – Cluster

- Au niveau du cluster, nous pouvons récupérer les logs de plusieurs composants:
- *kube-apiserver*
  - /var/log/kube-apiserver.log
- *kube-scheduler*
  - /var/log/kube-scheduler.log
- *Etcd*
  - *Etcd peut être exécuter dans un container au niveau du controller plan, Etc exposte le port 4001, on peut récupérer les métric etcd à l'aide du end points /metric.*
- *Events Kubernetes.*

Kubernetes – Logging et dépannage

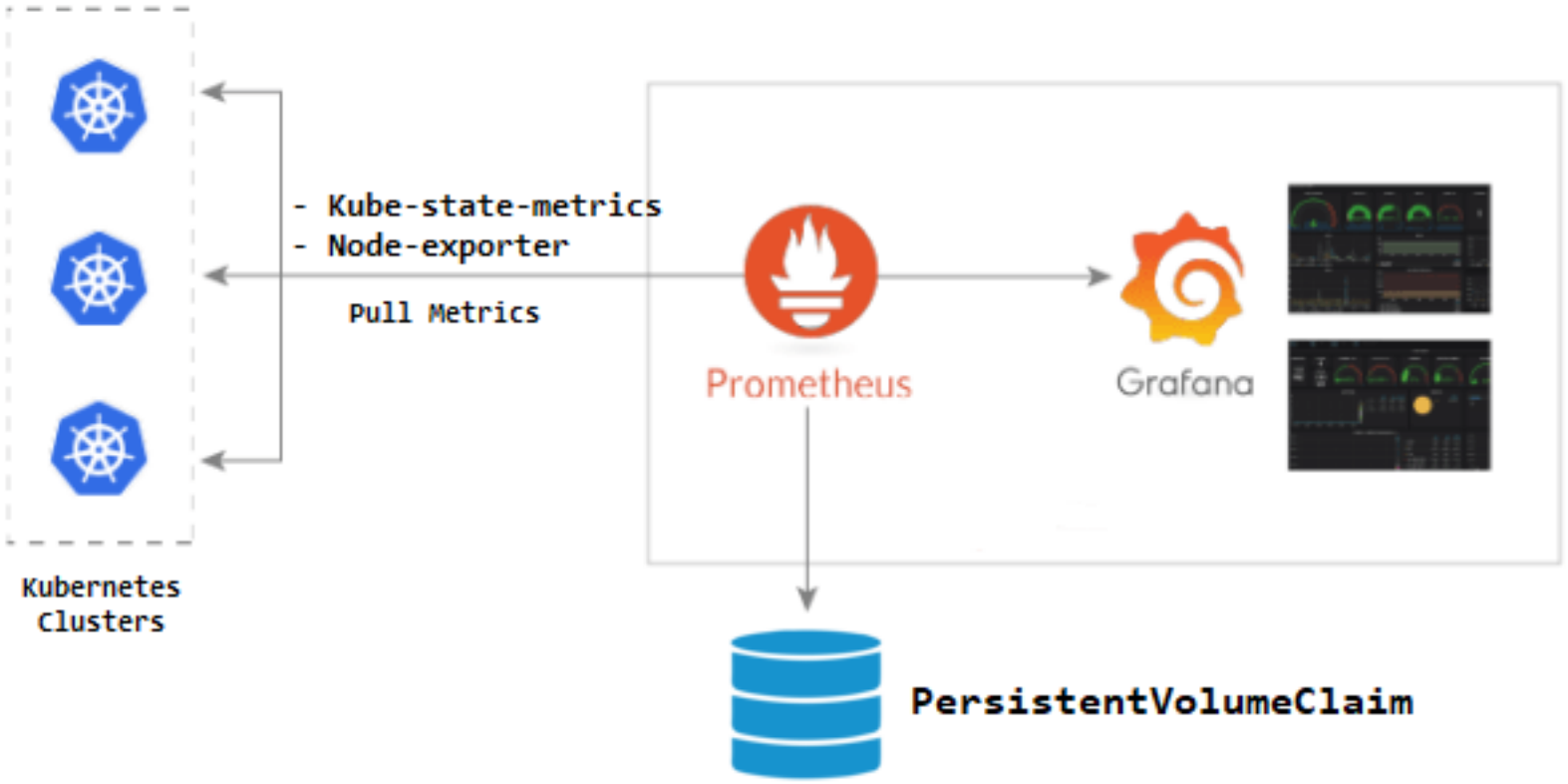
- Pour collecter les différents logs et métriques nous pouvons utiliser
- Un agent sur chaque nœud (worker et controller plane).
- Une application sidecar.
- Une application externe avec un mécanisme de pull par api.



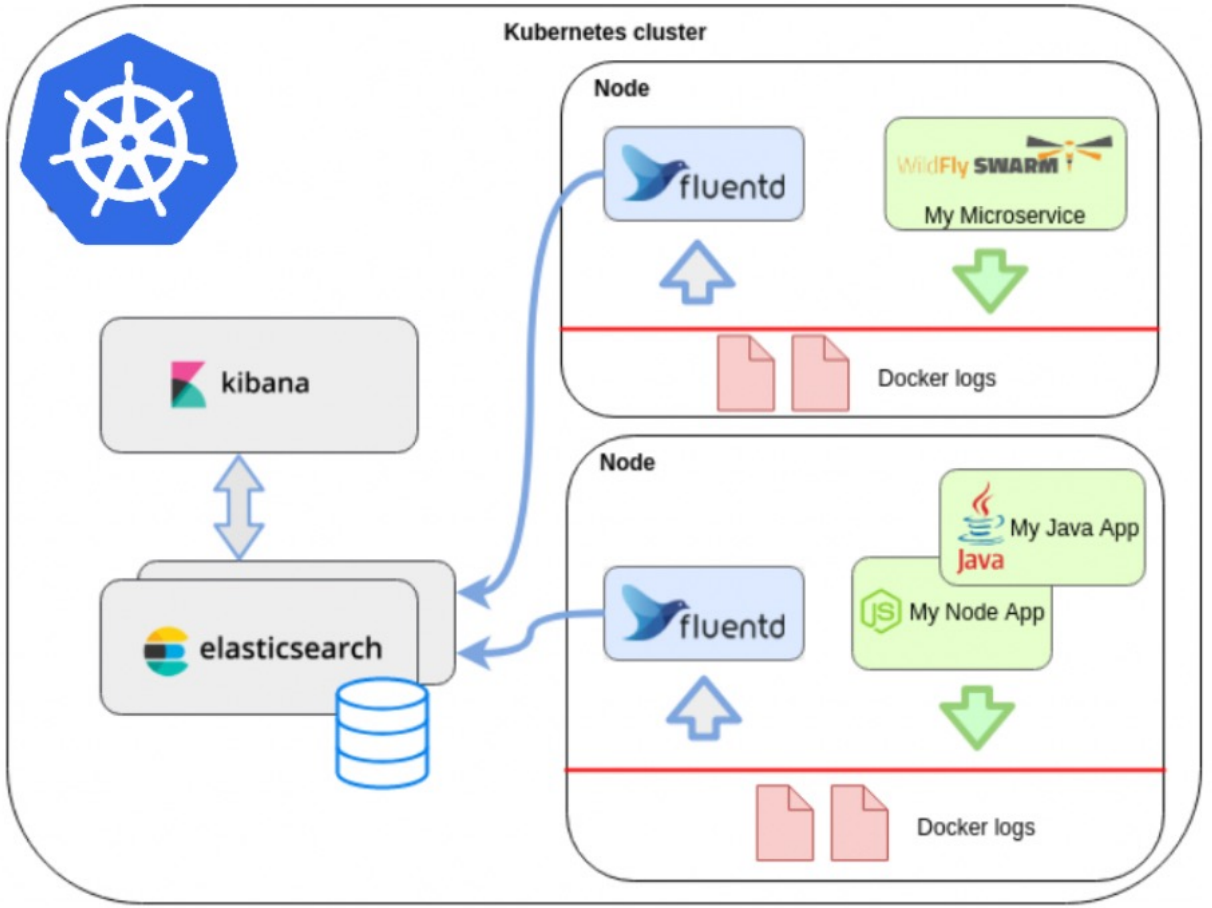
Kubernetes – Logging et dépannage – Outils

- Nous pouvons utiliser une multitude d'outils pour visualiser les logs
  - Kubernetes Dashboard
  - Stack ELK
  - Prometheus
  - Fluentd
  - Fluent bit
  - Data Dog
- …

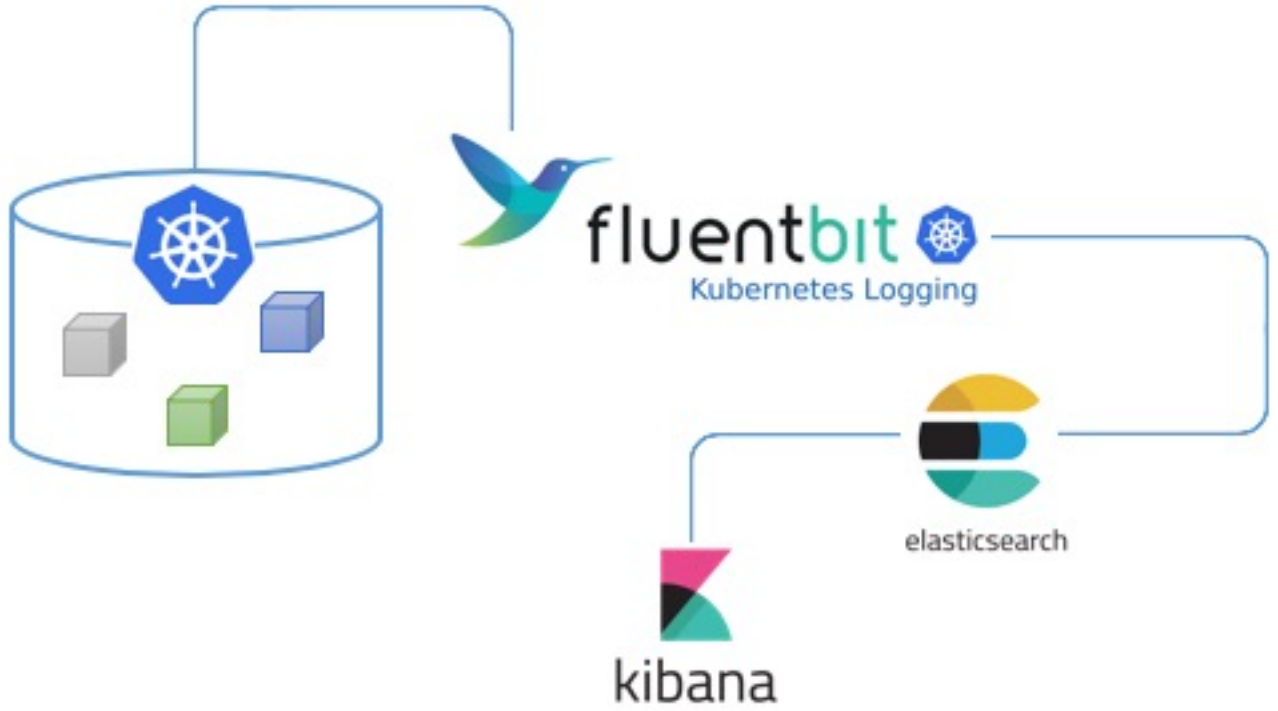
Kubernetes – Logging et dépannage – Prometheus



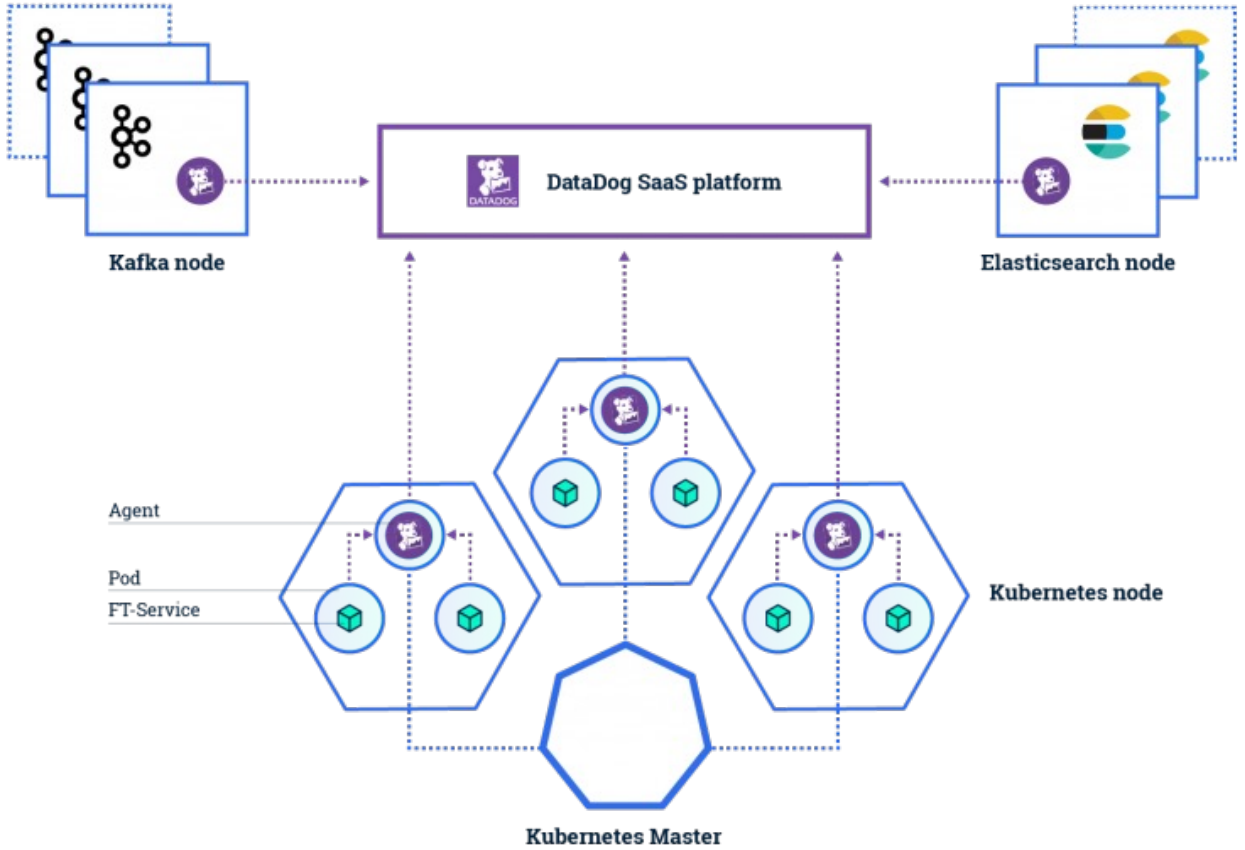
Kubernetes – Logging et dépannage – fluentd



Kubernetes – Logging et dépannage – fluent bit



Kubernetes – Logging et dépannage – data dog



Kubernetes – Logging et dépannage – events

- Les events sont les différentes actions réalisées dans notre cluster.
- Les events permettent d'identifier les actions à prendre pour le dépannage de notre cluster.
- Ils existent une multitude d'events
  - Nous pouvons utiliser l'api events pour accéder aux events, ajouter..
- Kubectl get events

Kubernetes – Logging et dépannage – events

- À l'aide des events et les logs, nous pouvons procéder aux dépannages d'applications.
- Quelques commandes api pour aider au dépannage:
  - Kubectl cluster-info
  - kubectl get componentstatus
  - Kubectl get events
  - kubectl describe pod

Kubernetes – Logging et dépannage – events

- Quelque events :
- **CrashLoopBackOff:** se produit lorsqu'un pod démarre, se bloque, redémarre, puis se bloque à nouveau
- **ImagePullBackOff**: se produit lorsque le nœud est incapable de récupérer l'image
- **Evicted events**, qui peuvent se produire lorsqu'un nœud détermine qu'un pod doit être expulsé ou résilié pour libérer des ressources (CPU, mémoire, etc.). Lorsque cela se produit, K8s est censé reprogrammer le pod sur un autre nœud.
- **FailedMount / FailedAttachVolume:** lorsque les pods nécessitent un volume ou un stockage persistant, cet événement les empêche de démarrer si le stockage n'est pas accessible.
- **FailedSchedulingEvents:** lorsque le planificateur n'est pas en mesure de trouver un nœud pour exécuter vos pods.
- **NodeNotReady:** lorsqu'un nœud ne peut pas être utilisé pour exécuter un pod en raison d'un problème sous-jacent.

Exercice

EXERCICE

- Démarrer le pod exercice-1.yml
- En utilisant les events, essayer d’apporter une correction pour faire fonctionner le pod

Exercice

EXERCICE

- Nous avons réaliser une application vote en micro-service.
- Le fichier script.sh permet de créer les différents ressources nécessaires pour déployer notre application sur un cluster kubernetes.
- Après le déploiement, l'application n'est pas totalement opérationnelle.
- À l'aide des events et les logs de l'application, essayez d'apporter des solutions pour rendre l'application operationnelle.

Ressources personnalisées

- Kubernetes permet de créer des ressources personnalisées de deux façon différentes.
- CRD
- Api Aggregation.

Ressources personnalisées - CRD

- Les définitions de ressources personnalisées (CRD) sont des extensions d'API Kubernetes qui peuvent définir de nouveaux types d'objets
- Les CRD sont eux-mêmes un type d'objet Kubernetes. Vous les créez de la même manière que n'importe quelle autre ressource, en écrivant un fichier YAML et en l'appliquant à votre cluster.
- Le CRD est prêt à être utilisé lorsque vous voyez Type: Established vers la fin de la sortie de la commande.
- Kubernetes acceptera les demandes adressées au point de terminaison API du CRD

Ressources personnalisées – Api Aggregation

- La couche d'agrégation permet de fournir des implémentations spécialisées pour les ressources personnalisées en écrivant et en déployant notre propre serveur d'API.
- Le serveur d'API principal délègue les demandes à votre serveur d'API pour les ressources personnalisées que vous gérez, les rendant disponibles à tous ses clients.

Exercice

EXERCICE

- Créer une ressource personnalisée qui permet de créer des objets de types website.

Kubernetes - Sécurité

- Les éléments à prendre en compte lorsque vous sécurisez vos conteneurs Kubernetes :
- **Configurations par défaut**
- **Communication avec l'API**
- **Exécution de conteneur**
- **Images**
- **Sécurité des hôtes**
- **Communications pod à pod**

Kubernetes – Sécurité – Méthodes authentification

- Chaque technique d'authentification Kubernetes sert le même objectif de base :
- valider l'identité de l'utilisateur ou du service qui émet la demande d'authentification pour déterminer si l'accès doit être accordé.
- Après l'authentification, les actions d'un utilisateur doivent être autorisées, généralement via un processus d'octroi d'accès basé sur les rôles.

Kubernetes – Sécurité – X.509 Certificates

- L'une des méthodes d'authentification les plus simples dans Kubernetes consiste à utiliser un certificat X.509 pour vérifier l'identité de l'utilisateur émettant une requête.
- Pour utiliser l'authentification X.509, vous devez d'abord créer une clé privée et une demande de signature de certificat, ou CSR, pour le compte d'utilisateur que vous souhaitez authentifier.

Kubernetes – Sécurité – Service Account

- L'authentification par service account utilise des jetons de porteur signés pour valider les demandes d'authentification.

Kubernetes – Sécurité – Autre méthodes

- **Static password files**
- **OpenID Connect Tokens & Identity Provider**
- **Authentication Proxy**
- **Webhook Token Authentication**

Kubernetes – Haute disponibilité

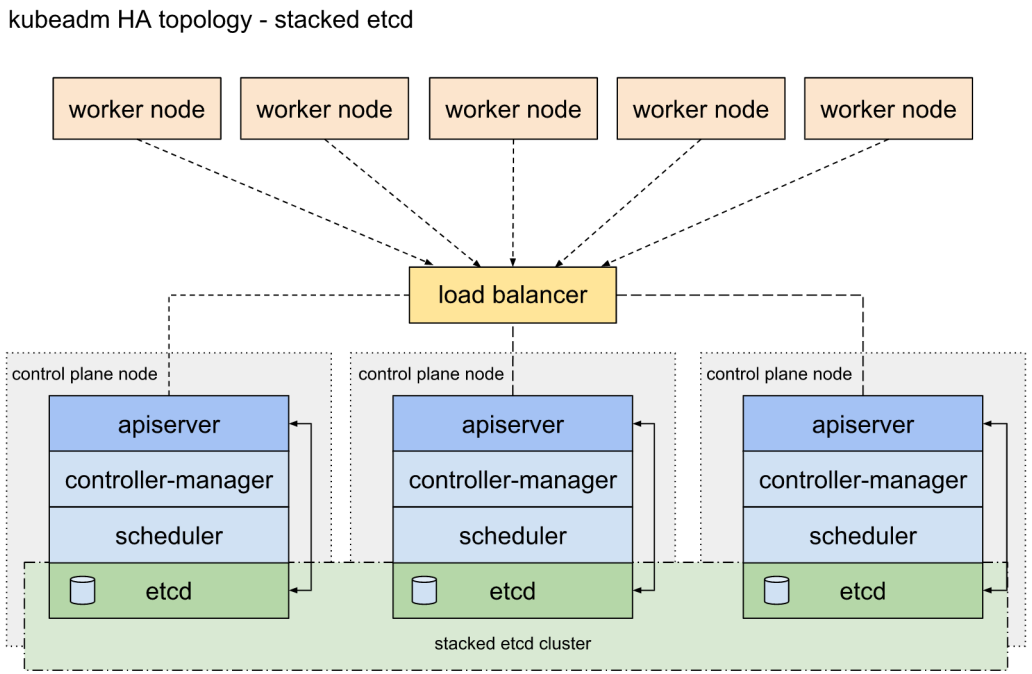
- Il existe trois éléments essentiels à la mise en œuvre de la haute disponibilité :
- Déterminez le niveau de disponibilité souhaité dans votre application. Le niveau d'indisponibilité acceptable varie en fonction de l'application et des objectifs commerciaux.
- Déployez votre application avec un plan de contrôle redondant. Le plan de contrôle gère l'état du cluster et aide à garantir que vos applications sont disponibles pour les utilisateurs.
- Déployez votre application avec un plan de données redondant. Cela signifie répliquer les données sur chaque nœud du cluster.

Kubernetes – Haute disponibilité

- La haute disponibilité de Kubernetes se présente sous deux formes : les clusters actifs-actifs et actifs-passifs.
- Les clusters actifs-actifs offrent une haute disponibilité en exécutant plusieurs copies de services sur tous les nœuds du cluster.
- Les clusters actifs-passifs utilisent des nœuds de sauvegarde qui sont inactifs la plupart du temps, avec une seule copie du service en cours d'exécution à un moment donné, mais si nécessaire, ils peuvent être rapidement mis à l'échelle pour gérer la charge.
- La façon de créer un cluster hautement disponible consiste à utiliser le plan de contrôle Kubernetes, qui est responsable de la planification des conteneurs pour les ressources. Kubernetes empêche les points de défaillance uniques en utilisant des contrôleurs de réplication afin que les défaillances de tout composant matériel ou logiciel n'affectent pas l'ensemble du cluster.

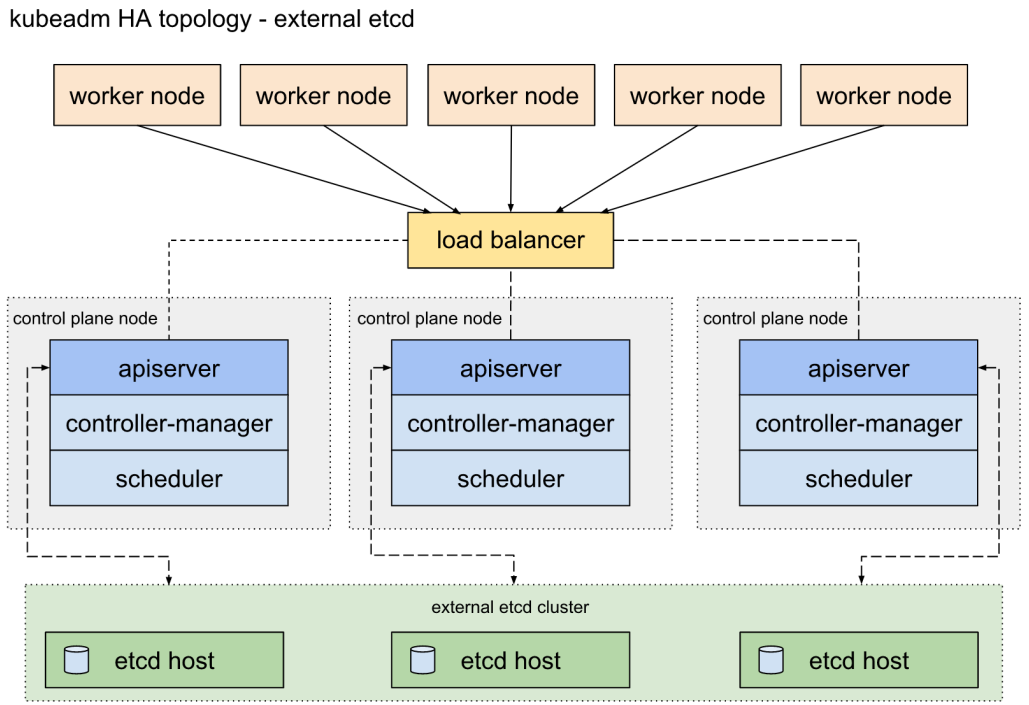
Kubernetes – Haute disponibilité - Stratégie

- Pour répondre aux besoins d'haute disponibilité, nous pouvons utiliser les base de données etcd empilées.



Kubernetes – Haute disponibilité - Stratégie

- Pour répondre aux besoins d'haute disponibilité, nous pouvons utiliser les base de données etcd externes.



Kubernetes – Haute disponibilité - Tools

- Kubeadm
- KubeKey