Orchestration et outil k8s

Objectifs

- Notion d’orchestrateur
- Présentation du produit Kubernetes (k8s)
- Premières commandes

Retour sur ISS

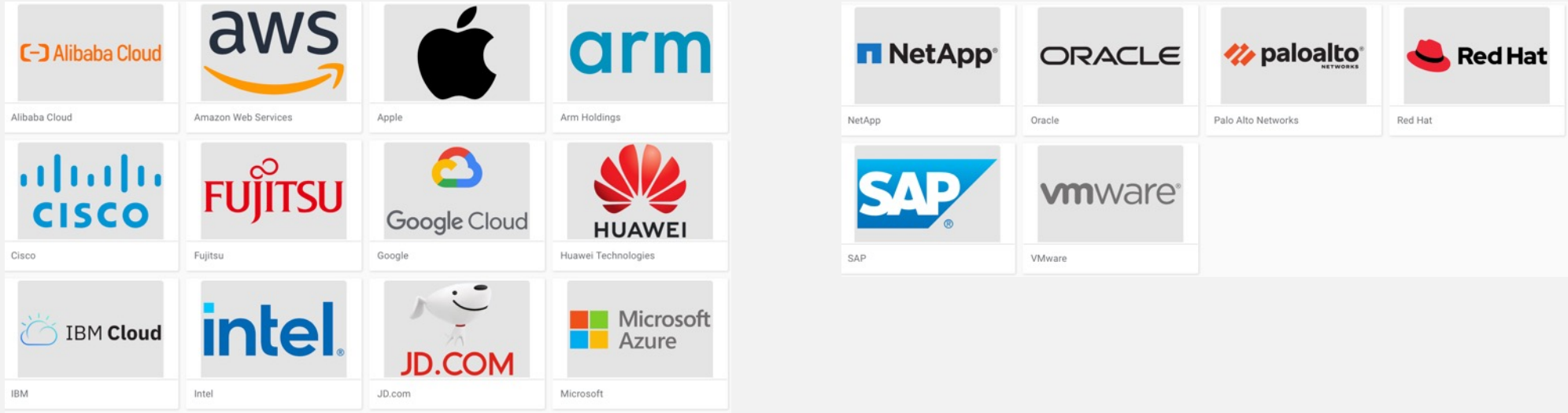
- Vous avez vu vu comment :
  - Créer une image docker
  - Créer un conteneur
  - Les faires « parler » ensemble
- Les opérations sont faites principalement à la main : on crée chaque conteneur en ligne de commande
- Si vous voulez les assembler à travers un fichier de configuration, vous avez du utiliser un orchestrateur que vous avez manipulé avec docker-compose
- L'utilisation **simultanée de plusieurs conteneurs** à mis en lumière la notion de dépendance :
  - Un conteneur peut **dépendre d'un autre** (ex. Wordpress dépend de MySQL)
  - Un conteneur doit **exister avant l'autre** (ex. MySQL avant le démarrage Wordpress

Orchestrateur

- Un orchestrateur permet :
  - Coordonner et gérer les ressources logicielles et matérielles
- Dans notre cas :
  - À partir d'une description, il va faire son possible pour exécuter la description (système dynamique)
  - À partir d'une description modifiée, il va adapter les ressources
- La description :
  - Création de ressources et dépendances : validation des ressources disponibles
  - Ordonnancer la création des conteneurs par rapport aux dépendances
  - Valider que les ressources créées sont dans l'état demandé

kubernetes

- Projet open source originaire de Google(1) :
  - Le nom vient du grec kubernesis : piloter / gouverner
  - Souvent simplifié en k8s (débute par k, 8 lettres, termine par s)
  - Basé sur Borg 2004, produit en version 1.0 en 2015(2)
  - Fait parti des projets du Cloud Native Computing Foundation (CNCF)

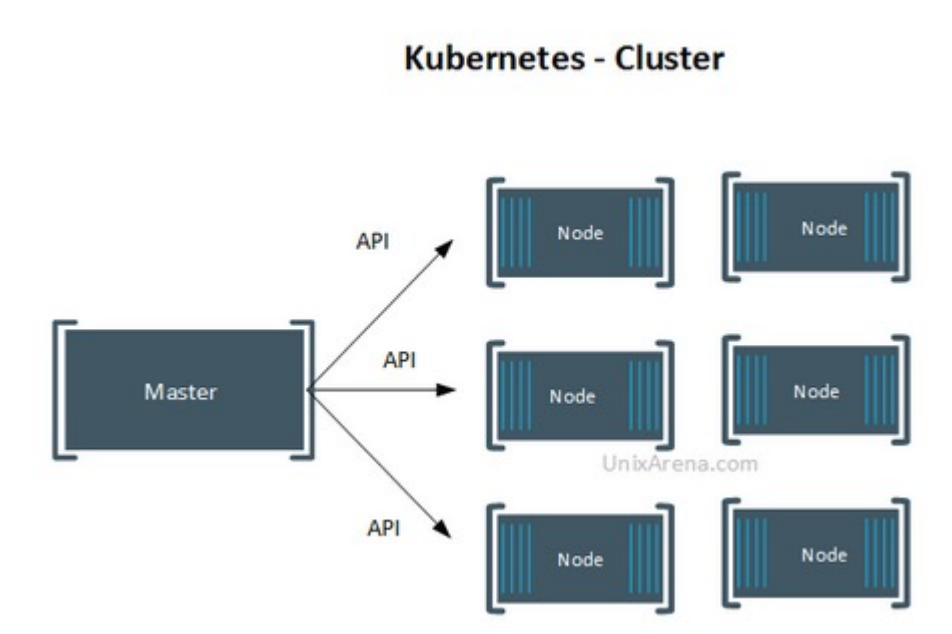


(1) https://kubernetes.io

(2) https://research.google/pubs/pub43438/

Notions et vocabulaire k8s

- Nœud (serveur) : physique ou virtuel
  - Maitre : gestion des nœuds de calcul, s'assure que les configurations sont respectées au mieux
  - Nœud de calcul : nœud où s'exécute un ensemble de « pods »
- Grappe kubernetes (cluster) : ensemble de nœuds dont des nœuds maitres et des nœuds de calcul



Les différents types de nodes

```
$ kubectl get nodes
NAME      STATUS    ROLES     AGE       VERSION
kube-01   Ready     master    32m       v1.13.4
kube-02   Ready     <none>    24m       v1.13.4
kube-03   Ready     <none>    24m       v1.13.4
```

Master

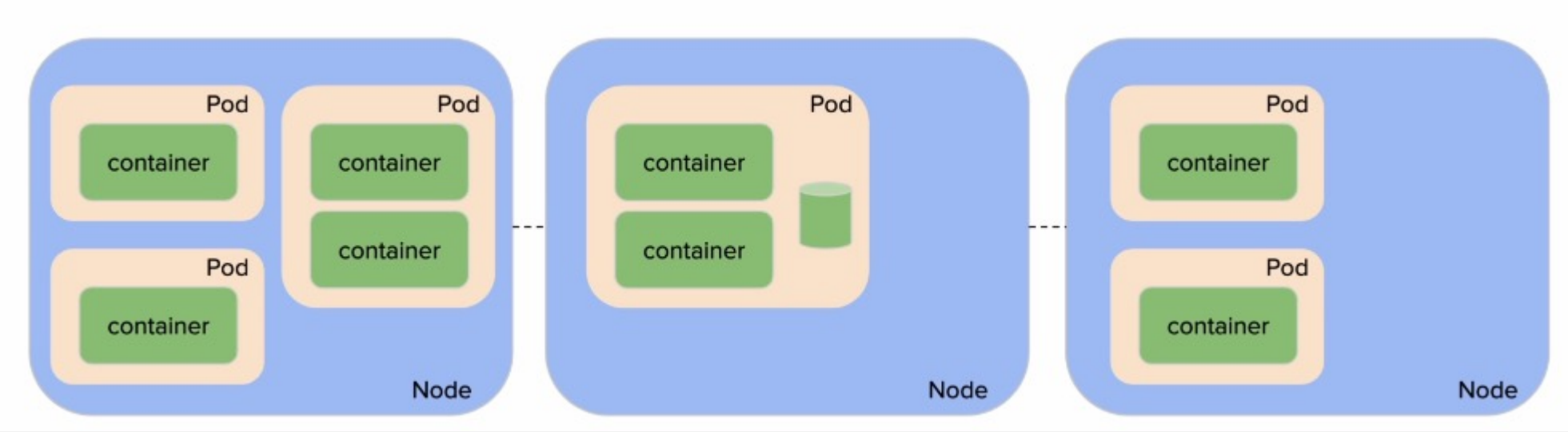
- Responsables de la gestion du cluster (« Control plane »)
- Exposent l'API server
- Planifient les Pods sur les nodes du cluster

Worker /Node

- Nodes sur lesquels sont lancés les Pods applicatifs
- Communiquent avec le Master
- Fournissent les ressources de calcul aux Pods

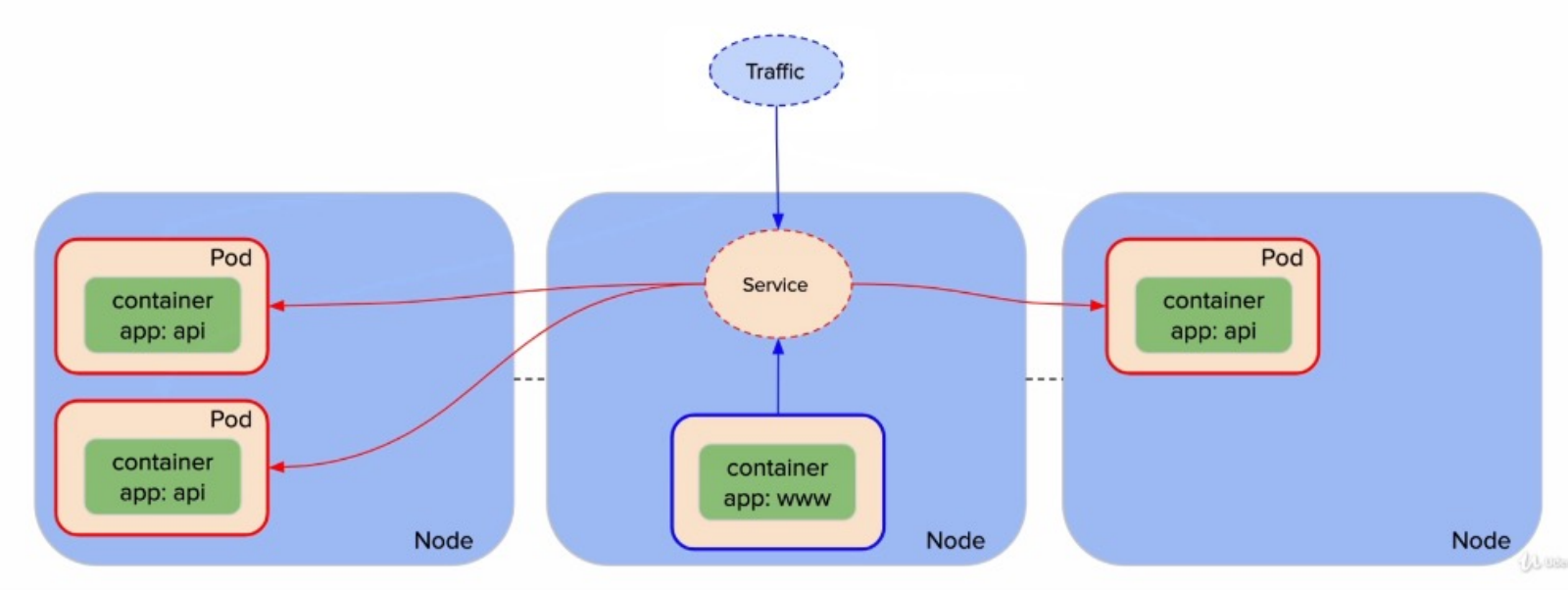
Notions et vocabulaire k8s

- Pods : pierre angulaire de k8s
  - Plus petite unité applicative qui tourne sur un cluster Kubernetes
  - Ensemble cohérent de conteneurs : 1 à n conteneurs
    - Nom d'une instance dans kubernetes
  - Un groupe de containers qui partagent réseau/stockage



Notions et vocabulaire k8s

- Service :
  - Expose les applications des Pods à l'intérieur ou à l'extérieur du cluster
  - Abstraction qui permet de définir un ensemble logique de pods et une méthode d'accès à ces pods
  - Les pods ne sont pas accessibles de l'extérieur du cluster

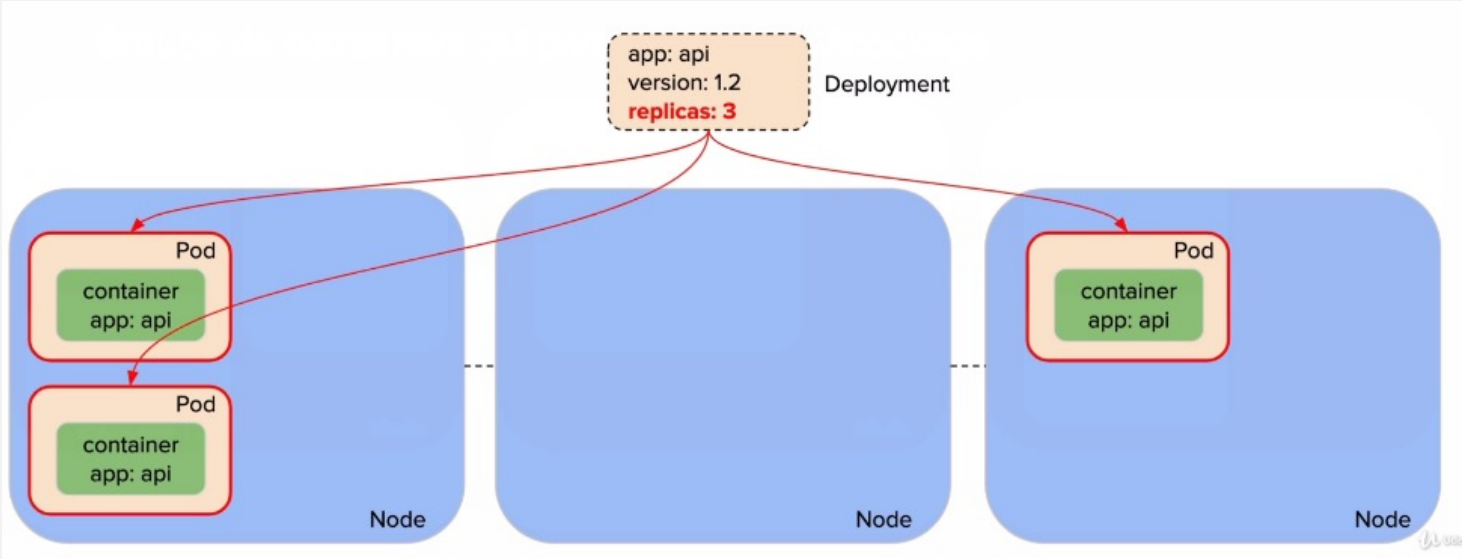


Notions et vocabulaire k8s

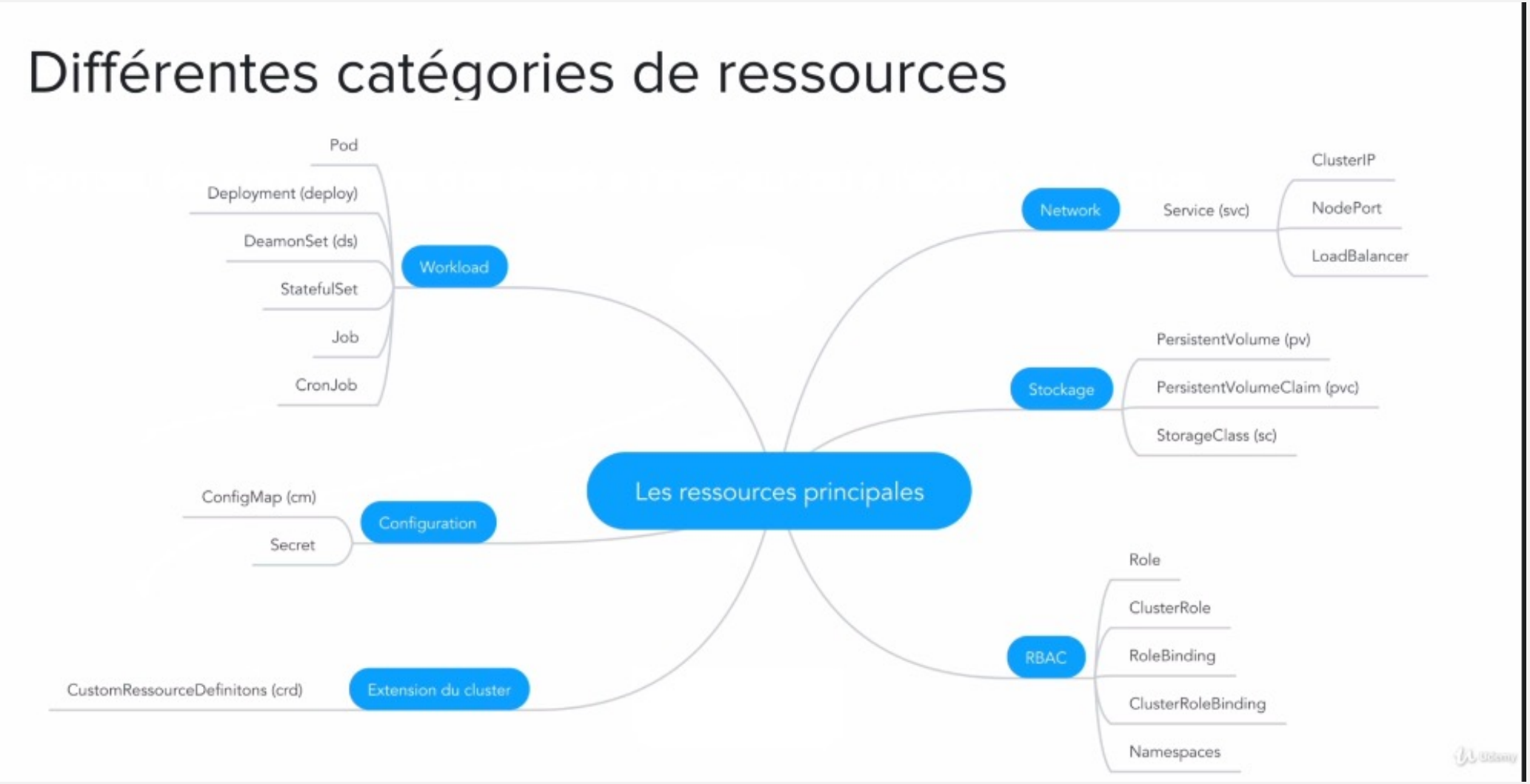
- Volumes
  - Même principe que pour docker : assurer la persistance
- Secret
  - Gestion de données sensibles comme des mots de passes
  - On peut les utiliser dans un fichier de configuration à la place d'écrire directement un mot de passe par exemple
- Configuration (config-maps)
  - Simplifie le déploiement sur différent environnement (ex. : développement, test, intégration, production)
- Namespaces
  - Sorte de partition dans k8s
  - La sécurité peut-être appliquée par namespace

Notions et vocabulaire k8s

- Déploiement (deployments)
  - Permet de gérer un ensemble de **Pods** identiques (mise à jours /rollback)
  - Gestion de la configuration d'un ensemble de ressources (pods, services, etc.) : création / lecture / modification / suppression (CRUD)
  - Possibilité de faire des mises à l'échelle
    - Verticale : ajout de ressources matérielles
    - Horizontale : ajout de pods
- Etc.

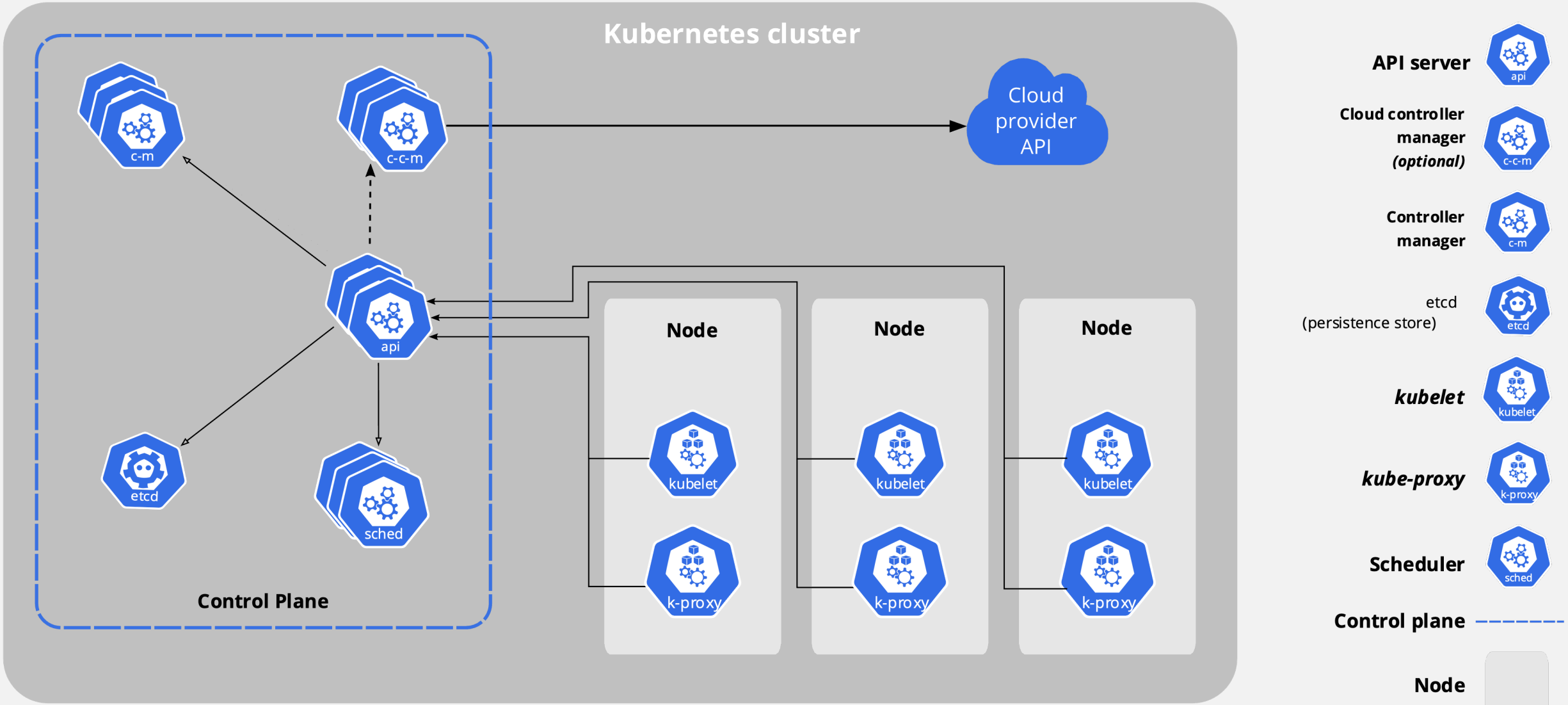


Types de ressources



Source : Intoduction à Kubernetes, Luc Juggery, Udemy.

Architecture de kubernetes



https://kubernetes.io/docs/concepts/overview/components/

Passons à quelques commandes

- Client kubernetes : kubectl
- kubectl get <type ressource>[,type ressource]* :
  - kubectl get nodes : affiche les informations sur les nœuds du cluster
  - kubectl get pods : affiche les pods
  - kubectl get pods --all-namespaces : affiche les pods de tous les namespaces
- kubectl run <nom> --image=image [--env="key=value"] [--port=port] [--dry-run=server|client] [--overrides=inline-json] [--command] -- [commande] [args...]

Passons à quelques commandes

- kubectl exec <nom> [-it] -- [commande] [args...]
- kubectl delete [type ressource] [pod name] :
  - kubectl delete pods <nom du pod> : supprime un pod
- kubectl api-resources : liste des ressources disponibles dans la grappe k8s
- kubectl create -f <nom fichier> : crée les ressources du fichier
- kubectl apply -f <nom fichier> : modifie / crée les ressources du fichier
- kubectl delete -f <nom fichier> : supprime les ressources du fichier

Passons à quelques commandes

- https://kubernetes.io/fr/docs/reference/kubectl/cheatsheet/ : Auto-complétion
  - Bash :
    - source <(kubectl completion bash)
    - echo 'source <(kubectl completion bash)' >>~/.bashrc
  - ZSH :
    - source <(kubectl completion zsh)
    - echo 'source <(kubectl completion zsh)' >>~/.zshrc
  - PowerShell :
    - kubectl completion powershell | Out-String | Invoke-Expression
    - kubectl completion powershell >> $PROFILE

Références

- Site officiel : https://kubernetes.io/fr/
- CheatSheet : https://kubernetes.io/docs/reference/kubectl/cheatsheet/
- Commandes kubectl : https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands
- Si vous êtes intéressés par les statistiques : https://k8s.devstats.cncf.io/d/12/dashboards?orgId=1&refresh=15m