malabo
ICT Advisory

Scheda Kubernetes

Kubernetes è un sistema **open-source** per l'automazione del deployment, scalabilità, e gestione di applicativi in containers, consentendo la loro orchestrazione e gestione.

L'obiettivo di Kubernetes è di facilitare la realizzazione di microservizi in ambiti virtualizzati e containerizzati, tipicamente in cloud, ma anche on-premises e in ambienti cloud ibridi, aggiornabili più volte al giorno in maniera sicura ed immediata, ossia senza down time, operanti 24 ore su 7 giorni. Kubernetes facilita quindi che le applicazioni containerizzate operino dove e quando si vuole, e che trovino tutte le risorse e gli strumenti ICT necessari perché possano operare.

Kubernetes è in pratica uno strumento per lo sviluppo e la gestione, in maniera sicura ed efficace, di software secondo la logica DevOps, che integra i processi di progetto sviluppo del codice software (Dev) con quelli della sua gestione operativa in produzione (Ops): consente quindi di gestire la complessità di DevOps in contesti terziarizzati e virtualizzati,

Inizialmente sviluppato ed adottato da Google, funziona con molti sistemi di containerizzazione, compreso Docker. Il sito ufficiale di riferimento è https://kubernetes.io/it/. Sulla base di questo standard i principali fornitori mondiali hanno rilasciato le loro soluzioni, in particolare Amazon EKS, Azure AKS, Google GKE.

La fig.1 schematizza l'architettura di un cluster Kubernetes evidenziando i principali elementi:

- Il **Master** è il coordinatore e supervisore del cluster con i vari nodi. Il suo ruolo principale è di orchestrare i nodi, non di eseguire container applicativi. Per motivi di affidabilità il Master può essere replicato su sistemi diversi. Nel suo ambito:
    - API server: rende disponibili le API, Application Program Interface di Kubernetes, in logica REST, verso lo stato del cluster. Esso rappresenta l'unico canale di coordinamento e controllo, sia per i nodi che per gli operatori/amministratori
    - Control Manager: controlla che lo stato attuale del sistema coincida con il desired state
    - Scheduler: assegna il carico di lavoro specificato dal desired state sui nodi che compongono il cluster
    - Etcd: mantiene lo stato del sistema. I componenti del control plane sono stateless e fanno riferimento, tramite *kube-apiserver*, allo stato mantenuto in etcd.
- Il **Nodo**, chiamato anche "worker", esegue il carico di lavoro secondo quanto definito dal Master. Per poter eseguire i carichi di lavoro, il nodo deve disporre di un container runtime come Docker o rkt**.**
    - Il kubelet è il componente di control plane che controlla le risorse e gestisce il carico di lavoro su un singolo nodo. Kubelet controlla il container runtime**;** mantiene una comunicazione col master e interviene costantemente sul nodo al fine di raggiungere e mantenere il desired state.
    - cAdvisor: collezziona ed aggrega informazioni sul container "running" di un dato nodo, quali utilizzo CPU, memoria, file, uso della rete.
    - Kube-proxy: inoltra il traffico fra i nodi secondo la configurazione delle regole di networking sugli stessi. Tramite il proxy viene resa trasparente la gestione dell'accesso ai service.

- Gli **oggetti di base,** come **Pod**, costituiscono l'unità elementare eseguibile su un nodo del cluster. Un pod raggruppa dei container che condividono le risorse e che vengono eseguiti sullo stesso nodo. Il pod si occupa di astrarre rete e storage al fine di poter essere spostato e replicato facilmente sui nodi del cluster, permettendo una forte scalabilità orizzontale, in particolare alle applicazioni orientate ai microservizi.
- Il **servizio** (Service) definisce come esporre dei pod su una rete interna o esterna. Il service definisce un nome che viene risolto dal DNS interno al cluster con uno dei pod ad esso associati.

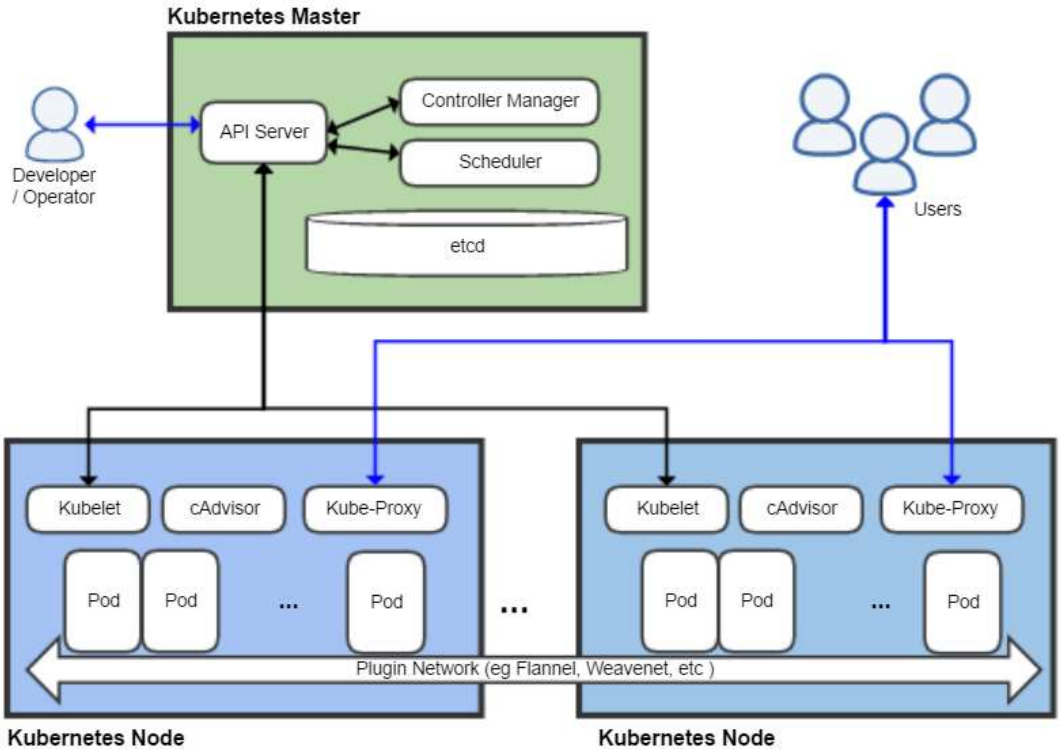

Fig. 1 Architettura Kubernetes (fonte: Wikipedia)