Operativ Systemer 5

Lavet af: Vivek Misra

Container Orchestrators

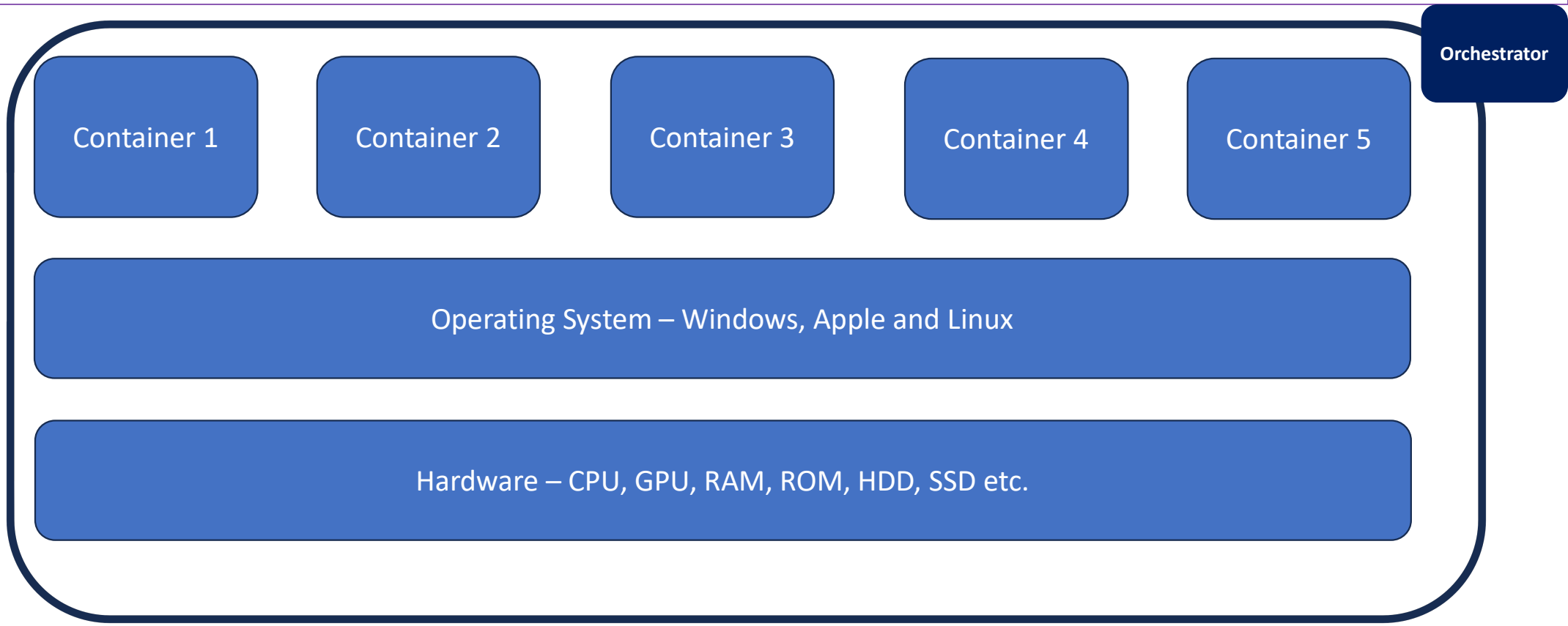
Vi kommer til at introducere jer til betydning af Containere i sammenhæng med Kubernetes.

Containers og Orchestrator

- Når vi snakker om Containers fra den virkelige verden, så plejer vi at referere til en fragtskib der indeholder forskellige varer.
- Men når det kommer til den teknologiske verdenen, så betragter vi Containere som en helt verden, der indeholder forskellige tekniske features og elementer.
- Hvorimod hvis man snakker om Orchestrator, kan det anses som en Gulvvagt der holder styr på de forskellige containers behov.
  - Eksempelvis hvis Mobile Banking kræver mere RAM, så vil Orchestratoren allokere mere ressource til selve Mobile Banking.
  - Eksempel på Orchestrator kan være Kubernetes, der er kendt for at være effektiv og skalerbar ift. Allokering af ressourcer til forskellige forbrug.
- På næste side viser vi en strukturel opbygning af Containers og Orchestrator.

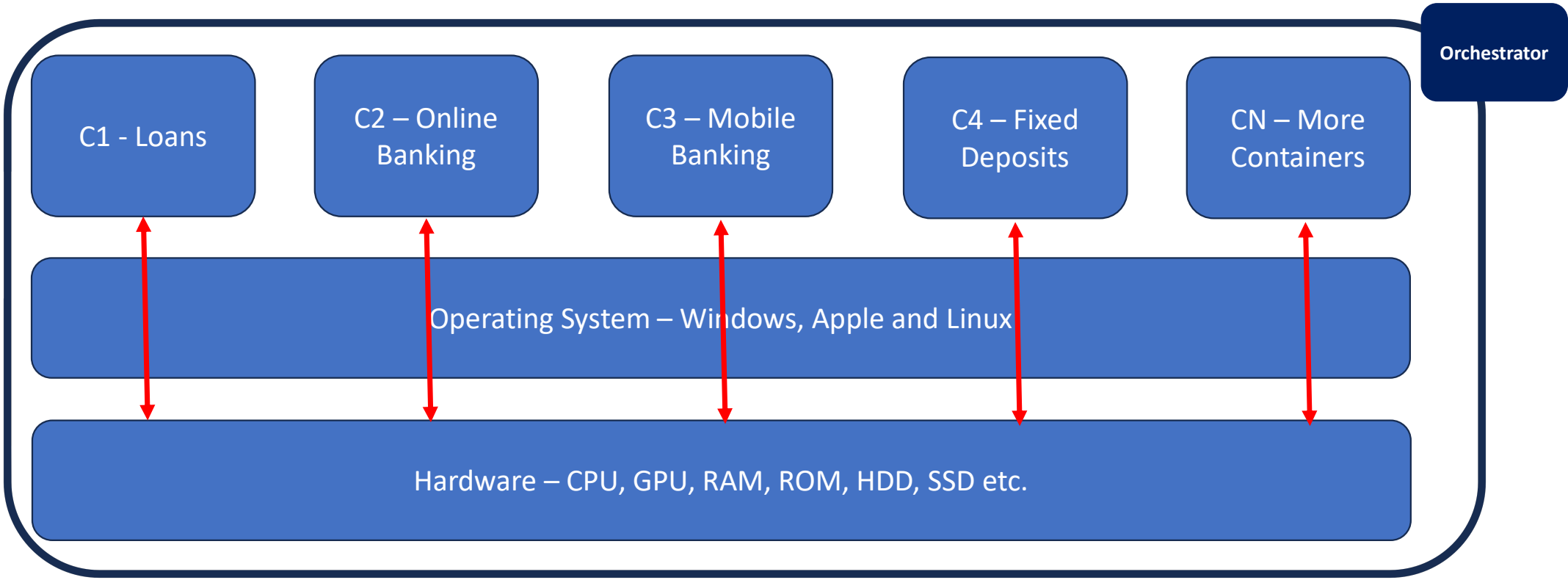
Containers og Orchestrator

- Her viser vi en strukturel opbygning af Containers og Orchestrator.



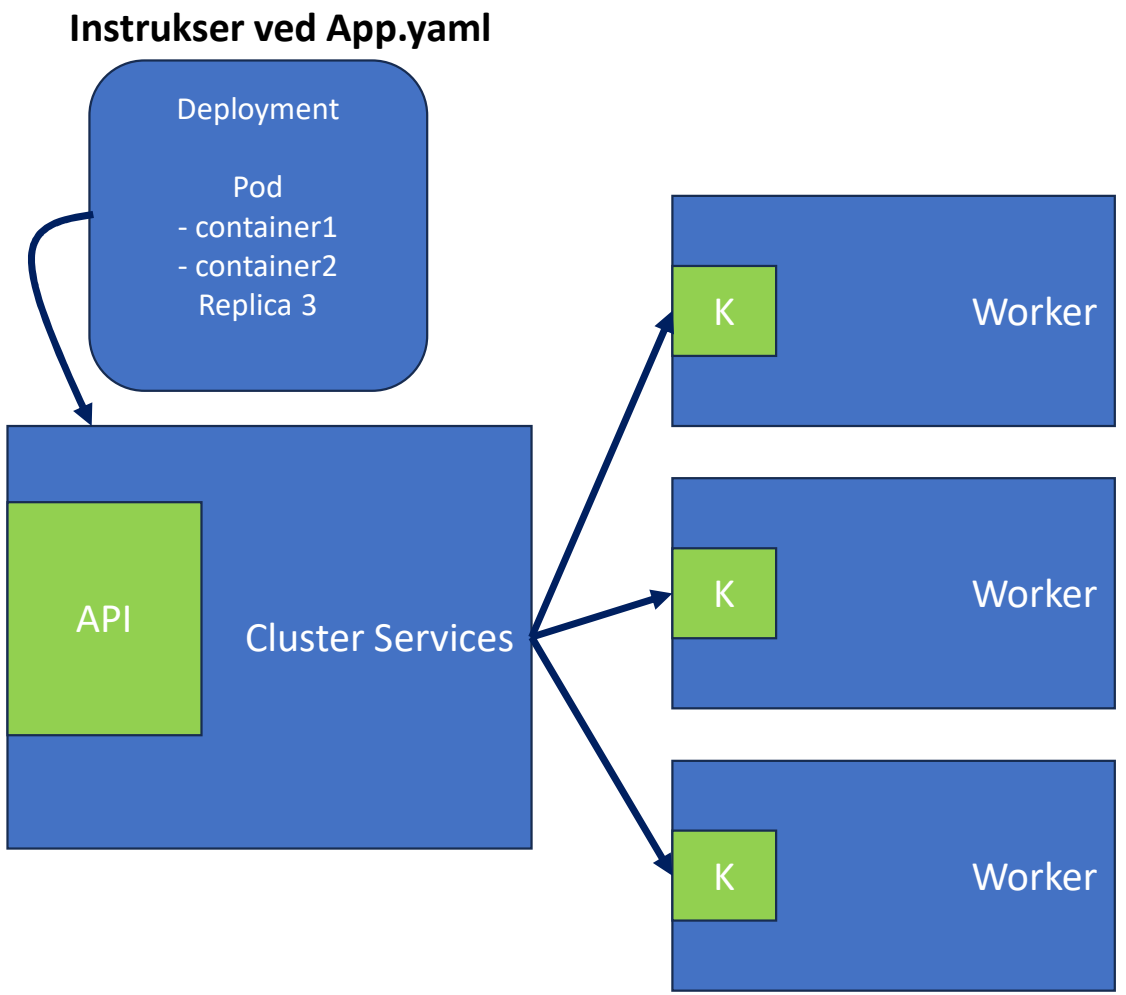
Containers og Orchestrator

- Her viser et eksempel med Banken og de forskellige muligheder den tilbyder.
- Vi kan se, at Orchestratoren holder styr på Konfigurationen og Management.



Kubernetes

- Som vi nævnte tidligere var Kubernetes en Container Orchestrator.
  - Hensigten med Kubernetes var, at kunne danne en ”ønsket tilstand”, eller en ”desired state” således at man kunne holde styr på forskellige containere og deres arbejde.
  - Vi kan se, at de forskellige workloads er defineret gennem workers. Workers er initially dem som hoster ”containers” og har en kublet der skaber forbindelse med clusteren ved Kubernetes.
  - Den ”desired state” som vi ønsker at have, er defineret gennem en ”.yml” fil.
  - Under ”.yml” filen kan det ses, at vi laver forskellige instrukser som eksempelvis ”pod” der forklarer hvor mange containers der skal køre i Kubernetes.



Kubernetes

- Lige nu har vi kun vist det basale struktur af Kubernetes.
  - Nu viser vi den rigtige og komplekse struktur, hvor der findes de overordnede komponenter sammen med API-Kaldet ved Clusteret.
  - Vi kan se, at der findes forskellige komponenter ved Kubernetes strukturen som vi vil fremhæve på næste side.

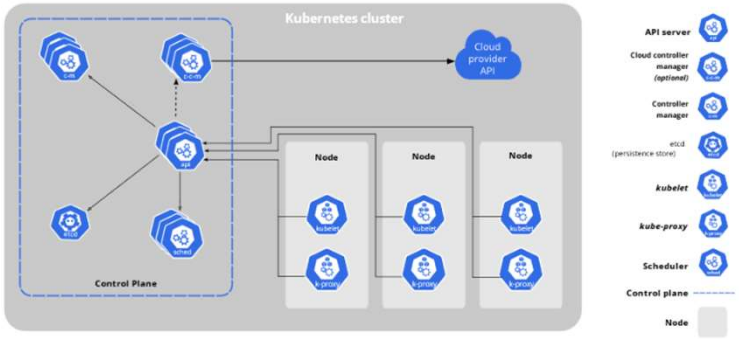


Figure: © Kubernetes

Kubernetes

- Nu forklarer vi om Kubernetes - Komponenter:
  - **Control Plane:** Det kan betragtes som hjernen af containersystemet. Den håndtere og styrer alt hvad der sker i systemet – som eksempelvis start eller stoppe en container.
  - **Scheduler:** Det kan betragtes som ”organisatoren”. Scheduleren bestemmer når en maskine eller noden skal køre en ny container baseret på ledige ressourcer.
  - **Controller Manager:** Det er ligesom en ”Supervisor” som holder styr på systemets tilstand og sørger for at alt kører som det skal. Den holder styr på forskellige aspekter af systemet.
  - **Cloud Controller Manager:** Det er en specialiseret Controller Manager beregnet for at holde styr på cloud enviroment. Den inteagerer med cloud providerens API-Kald til at kontrollere ressource som eksempelvis virtuelle maskiner og lagring.
  - **ETCD:** Den ses som en database, der gemmer alt information om clusteren. Det holder styr på ting som eksempelvis, hvilken node er en del af hvilket cluster. Samt også nodens nuværende status.

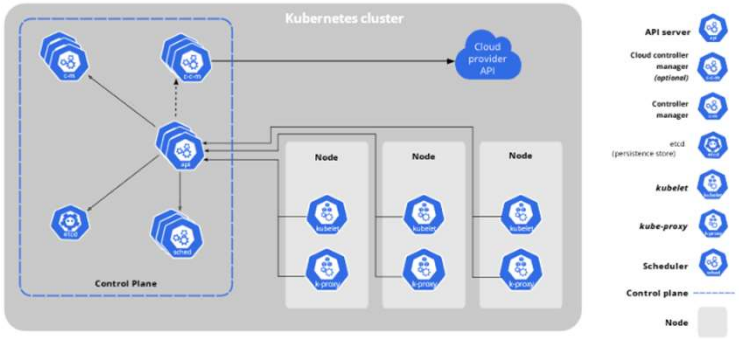


Figure: © Kubernetes

Kubernetes

- Så er der flere Kubernetes Komponenter:
  - **API-Server:** Det er ligesom en ”hovedstad”, hvor alt er forbundet med hinanden. Det er ligesom en gateway for alle brugere og komponenter der kan interagerer med cluster. Den modtager kommandoer, processer for dem, og kommunikerer med andre dele af systemet.
  - **Worker Plane:** Det er der, hvor den aktuelle arbejde er udført. Dette inkludere maskiner som eksempelvis ”nodes” som kører containerne og eksekvere opgaverne.
  - **Kubelet:** Det er ligesom en vikar for hvert mode. Den sørger for, at container er kørt som de skal, når de er i deres noder.
  - **Kube-Proxy:** Det er ligesom en trafikkontrollør som hjælper med at holde styr på netværkskommunikationen mellem de forskellige dele af clusteren og sørger for at de kan finde den de leder efter og snakke med hinanden.

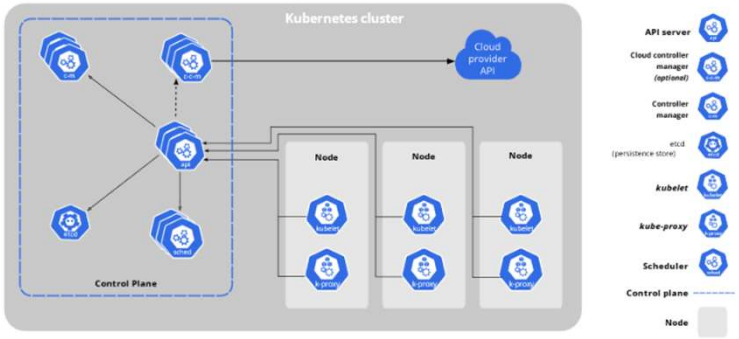


Figure: © Kubernetes

Kubernetes

- Nu kommer vi til at forklare om Kubernetes Objekter:
  - **Nodes:** Nodes er ligesom arbejdere i en gruppe. Nodes er individuelle maskiner som eksempelvis computere eller servere i Kubernetes Cluster, der kører applikation i form af containers. Hvert node har deres eget kapacitet og ressourcer til at eksekvere disse containers.
  - **Pods:** Det er en klumpe af underarbejdere. Man siger, at det er den mindste enhed i Kubernetes og repræsenterer en eller flere container der arbejder sammen. Containers inde i pod deler ressourcer og er stramt tæt. Pods er scheduleret ovenpå noderne, så de kan indeholde applikationer og andre relevante lagringsressourcer.
  - **Namespaces:** Namespaces er forskellige rumme og lokaler i en bygning. De hjælper med, at organisere og opdele ressourcer i en Kubernetes Cluster. De tillader dig, at kunne isolere steder hvor forskellige teams eller projekter kan arbejde uden indblanding af hinanden.
  - **Deployments:** Deployments er ligesom instruktioner eller planer for at holde styr på goder. De definerer, hvor mange kopier af en pod der burde køre og hvilke former af opdateringer der burde være lavet og håndtere fejl. Deployments sørger for, at den ønskede tilstand af applikationen er opnået.

Kubernetes

- Nu kommer vi til at forklare om Kubernetes Objekter:
  - **Networking:** Netværk i Kubernetes referer til, hvor mange pods og services kommunikerer med hinanden. Den sørger for, at forskellige dele af ens applikation kører i pods og kan snakke med hinanden, uanset om hvilken node de befinder sig i og hvilken netværksregler og protokoller der befindes.
  - **Services:** Services er ligesom en ”receptionist”. Den er i stand til at kommunikere mellem forskellige pods. De tilbyder en konsistent måde, at få adgang til en sæt af pods på, selvom podsene er konstant i forandring. Der er forskellige typer af services:

    CLUSTER-IP: Den tilbyder adgang til service med indenfor en cluster.

    LOAD BALANCER: Den udbyder service til internettet, gennem brug af Load Balanceren fra Cloud Provideren.

    NODEPORT: Den åbner et specifik port på alle noder og videreføre trafik til service.
  - **Secrets:** Secrets er ligesom lukkede låseskabe, der indeholder privat information. De gemmer meget vigtig data som eksempelvis adgangskoder, og nøgler som er brugt af applikationerne indenfor pods. Kubernetes hjælper med at holde styr på, samt sikre private information ved at gemme det i Secrets og dermed have adgang til det.

SLUT 5

Lavet af: Vivek Misra