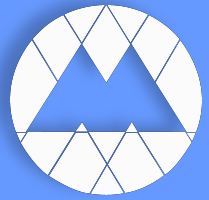
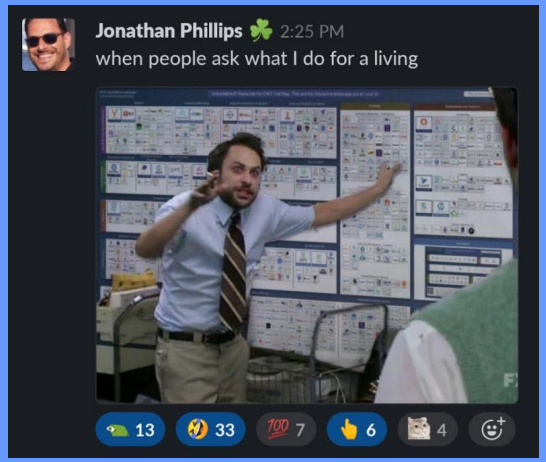
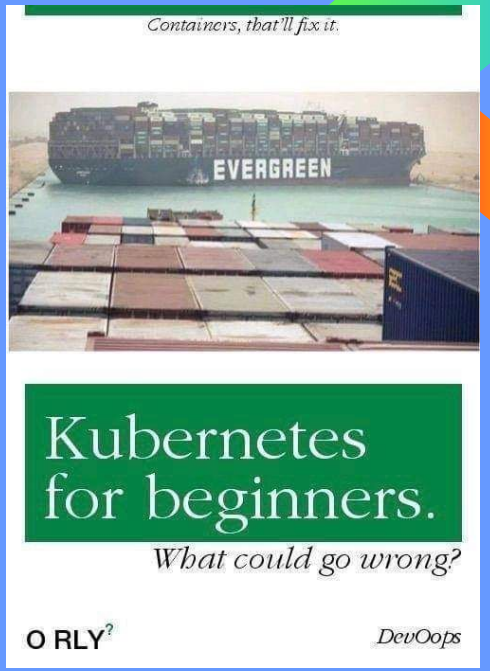


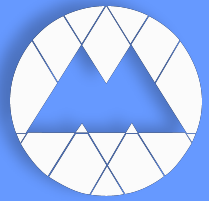
Velkommen til Kubernetes dag

Dagens formål

- Give jer en idé om hvorfor kubernetes bliver brugt så meget i moderne softwareudvikling
- Vise jer hvordan man laver simple deployments og services i kubernetes
- Forhåbentlig give jer værktøjerne til hvordan i lærer mere om kubernetes
- Lade jer forstå memes på r/kubernetes og r/programmerhumor





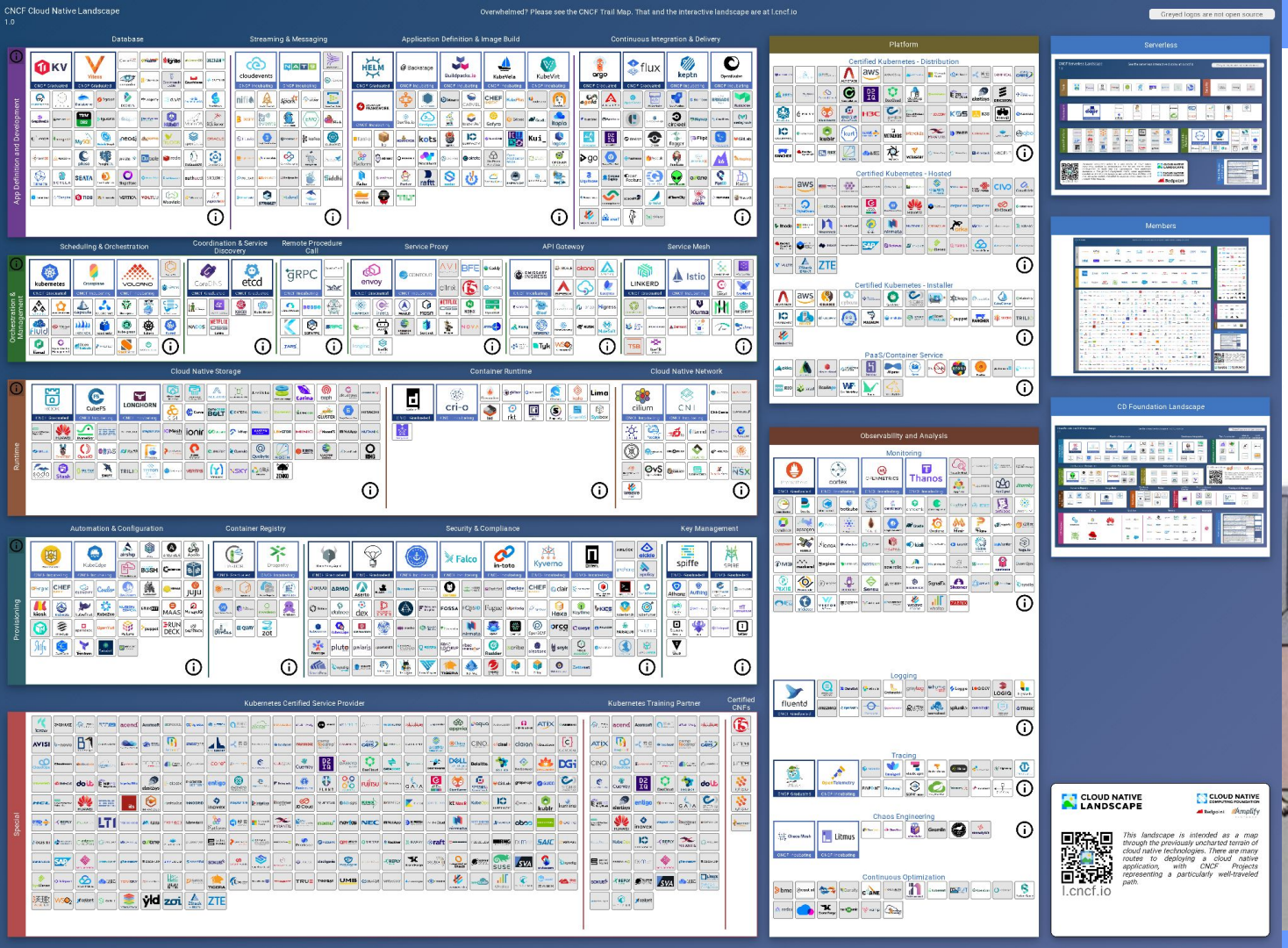


First things first!

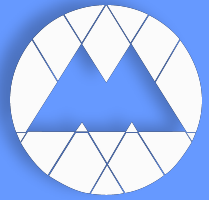
Git clone https://github.com/itm-fhm/k8s-dag.git for slides+exercises

Installér Docker + kind, minikube eller Docker indbygget kubernetes

- Kind
  - https://kind.sigs.k8s.io/
  - Krav:
    - Docker
    - Go (1.17+)
- Minikube
  - https://minikube.sigs.k8s.io/docs/start/
  - Krav:
    - Docker eller lign.
    - 20 GB disk



Hvorfor er Kub



Hvad er en kubernetes?

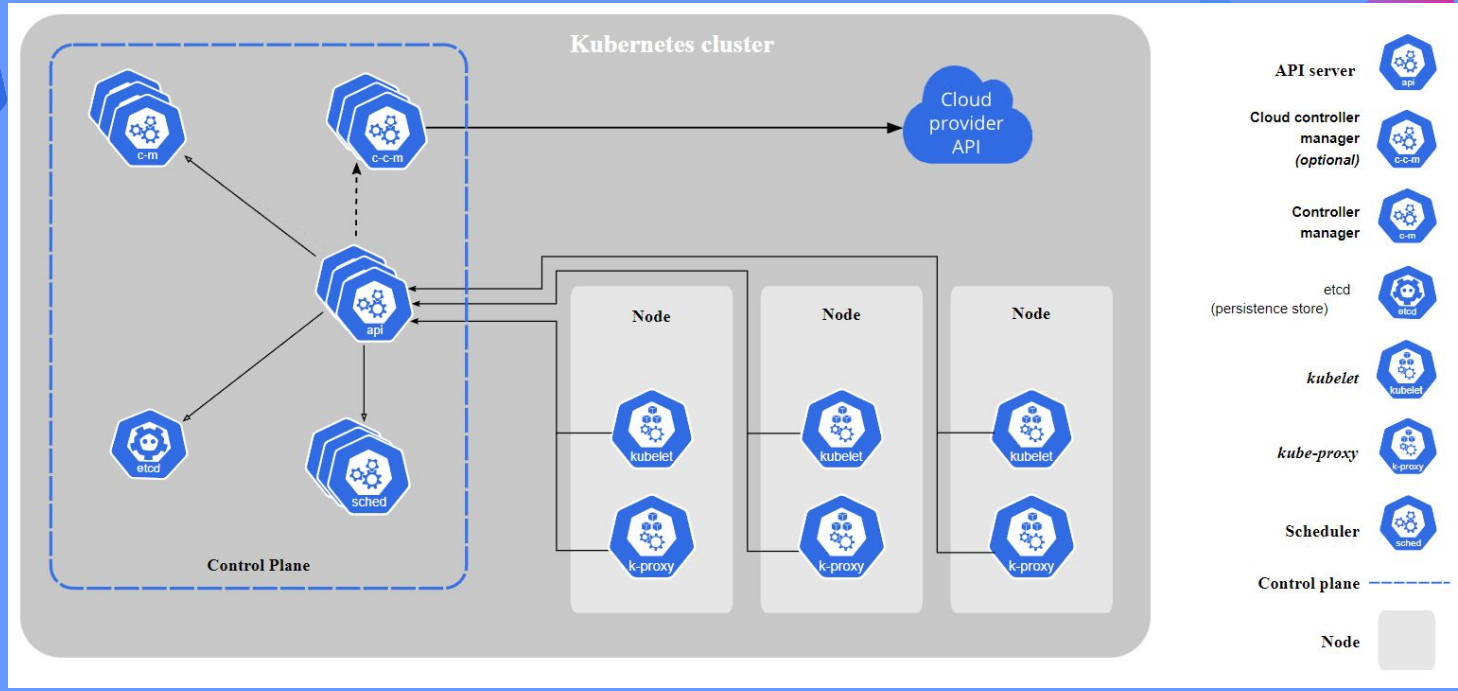
Fra Kubernetes homepage: “Kubernetes is an open source container orchestration engine for automating deployment, scaling, and management of containerized applications”

Kubernetes håndterer bl.a.:

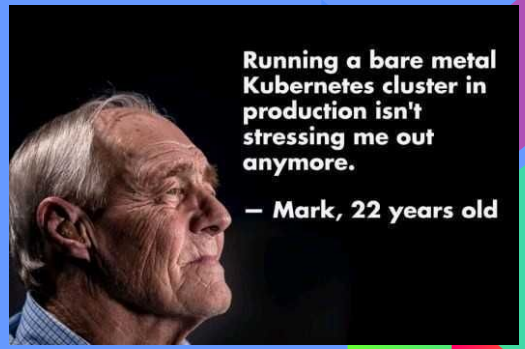
- Automated rollouts/rollbacks
- Service Discovery og load balancing*
- Storage Orchestration
- Secrets og andet configuration management
- Automatic bin packing/node allocations

Kubernetes kan bruges deklarativt via manifester (YAML) og/eller imperativt via kubectl (kube-cuddle? Kube-control? kube-c-t-l?) CLI’en

Hvad er en kubernetes?



Ikke vist: DNS Server (ofte CoreDNS)



Hvad er en kubernetes? - Flavors

Kubernetes har mange flavors:

- Cloud
  - AKS i Microsoft Azure
  - EKS i AWS
  - GKE i Google Cloud
- Baremetal
  - K8s
  - K3s
- Lokal development
  - Kind (kubernetes-in-docker)
  - Minikube
  - K3s

<- You will most likely be here

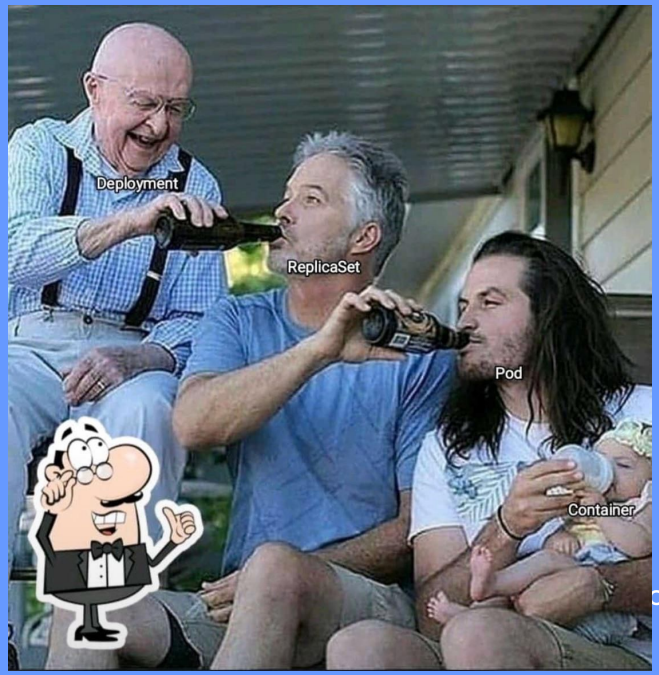
- Kommer med Loadbalancer
- ‘Nem’ patching
- Storage + Volumes er håndteret
- Masser ekstra hjælp

<- DON’T

- Ingen loadbalancer ud af boksen (brug MetalLB)
- Storage skal man selv komme med
- Meget customizable = du skal stå for alt selv

<- Os i dag

Hvad er en kubernetes? - Workload Typer



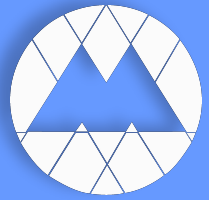
Hvad er et Deployment?

- Foretrukket måde håndtere workloads (for mig)
- Ligger ovenpå replicasets. Kan ses som: Deployments beskriver ReplicaSets beskriver Pods.
- Deklarative updates af ReplicaSets og Pods
- Kontrollér rollout+rollbacks

Hvad er DaemonSets?

- Sørger for at alle (eller nogen) nodes har en kopi af en Pod
- Bruges f.eks. Til monitoring

obs og StatefulSets.. I guess



Deployments sammenlignet

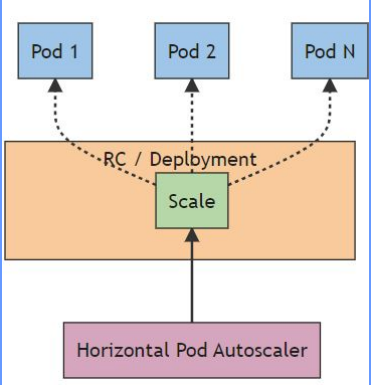
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2
    ports:
    - containerPort: 80
```

```yaml
apiVersion: apps/v1
kind: ReplicasSet
metadata:
  name: nginx-replicaset
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

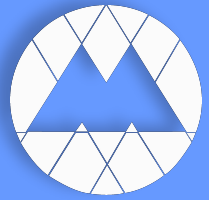
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

Scaling og Resource Quotas

- Kubernetes skalerer vha. en Horizontal Pod Autoscaler (HPA)
- HPA'en skalerer på baggrund af resource quotas
- Der findes to typer quotas:
  - Limits (max som pod'en kan assignes)
  - Requests (min som pod'en kan assignes
- Der er to typer resourcer CPU og Memory




```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
      - name: my-app-container
        image: nginx:latest
        resources:
            requests:
                memory: "64Mi"
                cpu: "250m"
            limits:
                memory: "128Mi"
                cpu: "500m"
        ports:
        - containerPort: 80
```



Hvad er en kubernetes? - Networking

Services

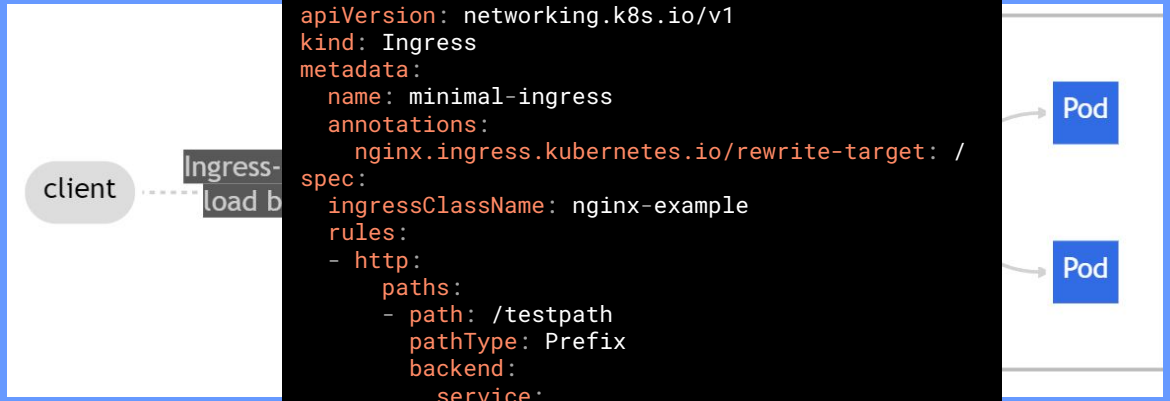
- ClusterIP - Default, exposer på Cluster Intern DNS, ikke reachable udefra
- NodePort - Allokerer en port på alle Nodes
- LoadBalancer - Cloud Provider provisioneret LB
- ExternalName - Mapper til et eksternt DNS name

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app.kubernetes.io/name: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```

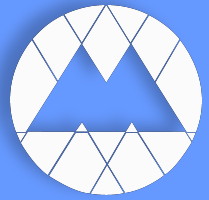
Hvad er en kubernetes? - Ingress

Vi bruger Ingress til at route traffik til Services.

Vi bruger IngressControllers til at styre Ingress (f.eks. NGINX, Traefik, HAProxy)




```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx-example
  rules:
  - http:
      paths:
      - path: /testpath
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              number: 80
```



Hvad er en kubernetes? - Storage

Volumes:

- Kubernetes Primitive
- Kan mountes til Pods
- Logisk abstraktion
- Mange varianter, emptyDir er mest lige til

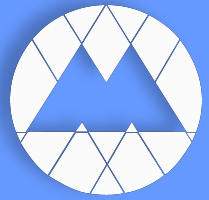
Nej. Brug en ekstern database i stedet

PersistentVolume

- Lag ovenpå Volume, er persisted selv når den mountede Pod dør

PersistentVolumeClaim (PVC)

- Type af volume der fortæller StorageController at en Pod/Deployment gerne vil have en delmængde af en PersistentVolume



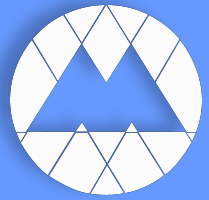
Hvad er en kubernetes? - Secrets og ConfigMaps

ConfigMaps

- Key-Value pairs
- Configuration separat fra applikationskode
- Pods kan consume ConfigMaps som environment variables, command-line argumenter eller konfigurationsfiler i en volume

Secrets

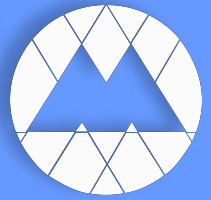
- Bruges til passwords, tokens, keys osv.
- Meget ligesom ConfigMaps, kan mountes på samme måde
- Secrets er pr. Default unencrypted i etcd, men kan med lidt konfiguration bruges meget sikkert (se https://kubernetes.io/docs/concepts/security/secrets-good-practices/)



Hvad er en kubernetes? - Secrets og ConfigMaps

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: myconfig
data:
  # property-like keys; each key maps to a
simple value
  some_key: "some-value"
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app-with-config
spec:
  selector:
    matchLabels:
      app: my-app-with-config
  template:
    metadata:
      labels:
        app: my-app-with-config
    spec:
      containers:
      - name: my-app-with-config
        image: nginx:latest
        ports:
        - containerPort: 80
        env:
        # Define the environment variable
            name: SOME_VALUE
            valueFrom:
                name: myconfig
# The ConfigMap this value comes from.
                key: some_key

# The key to fetch.
```



How do I kubernetes?

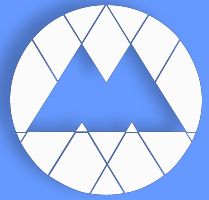
Tid til kubectl!

Syntax: https://kubernetes.io/docs/reference/kubectl/

Cheat Sheet: https://kubernetes.io/docs/reference/kubectl/cheatsheet/

Bash brugere:

```
apt-get install bash-completion

source /usr/share/bash-completion/bash_completion

kubectl completion bash | sudo tee /etc/bash_completion.d/kubectl > /dev/null

echo 'alias k=kubectl' >>~/.bashrc

echo 'complete -o default -F __start_kubectl k' >>~/.bashrc
source ~/.bashrc
```



Exercise 1

- Opret et namespace vha. `kubectl create namespace exercise1`
- Check via `kubectl get namespace exercise1`

```
NAME      STATUS   AGE
exercise1 Active   1h01m
```

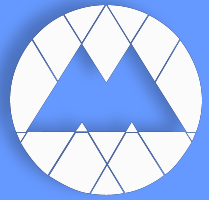
- Naviger til k8s-dag/exercises/make-a-deployment
- Opret deployment `kubectl apply -f deployment.yml -n exercise1`
- Check via `kubectl get deployments -n exercise1`

```
NAME     READY  UP-TO-DATE  AVAILABLE  AGE
my-app   1/1    1           1          1h02m
```

- Scale `kubectl scale deployment my-app -n exercise1 --replicas=2`

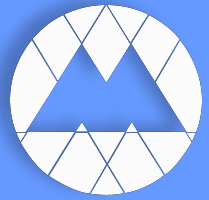
```
NAME     READY  UP-TO-DATE  AVAILABLE  AGE
my-app   2/2    2           2          1h05m
```

- Check via `kubectl describe deployments -n exercise1`
- Image skulle gerne være nginx:latest
- Edit deployment.yml så image bliver nginx:1.14.2
- Redeploy med `kubectl apply -f deployment.yml -n exercise1`
- Check via `kubectl describe deployments -n exercise1`
- Image skulle gerne være nginx:1.14.2



Exercise 2

- Opret et namespace vha. `kubectl create namespace exercise2`
- Naviger til k8s-dag/exercises/make-a-service
- Opret deployment `kubectl apply -f deployment.yml -n exercise2`
- Opret service `kubectl apply -f service.yml -n exercise2`
- Valider med `kubectl get svc -n exercise2`
- Port-forward `kubectl port-forward -n exercise2 deployments/my-app 8080:80`
- Gå til localhost:8080 i en browser og I skulle gerne se NGINX landing page

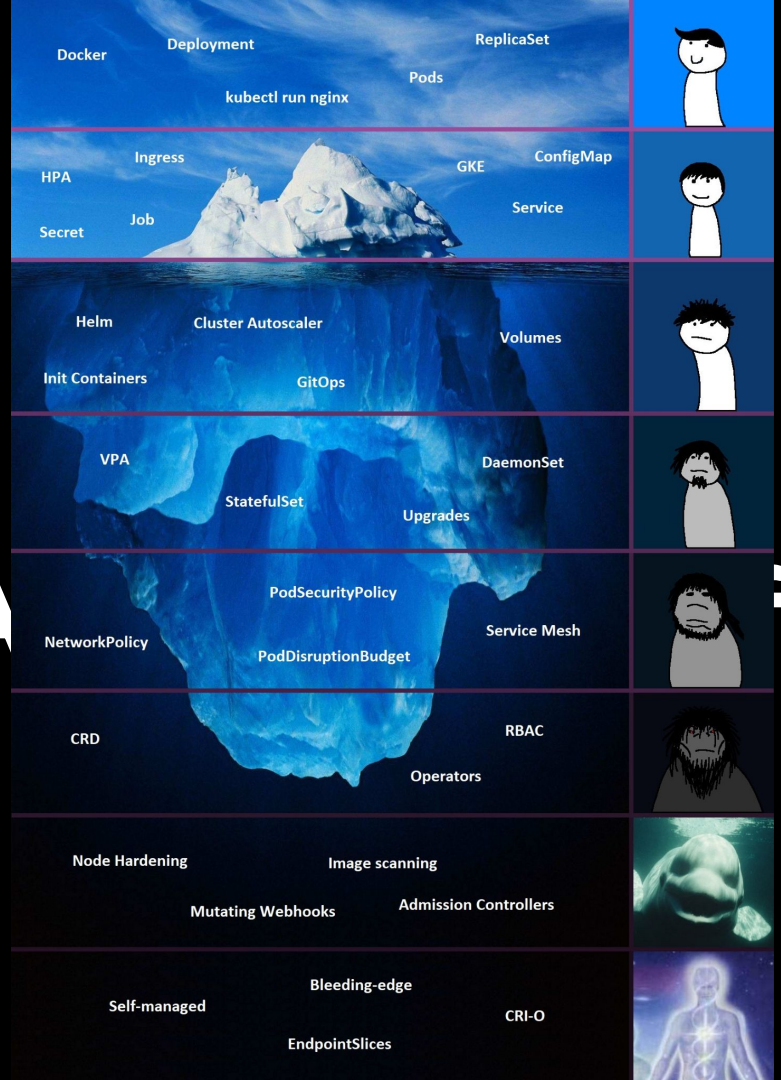


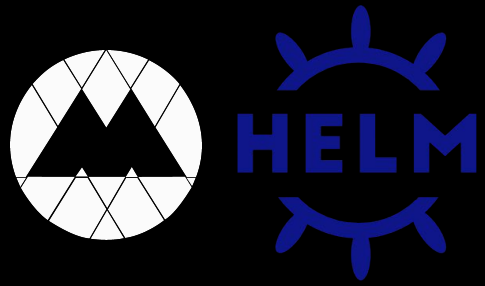
Exercise 3

- Opret et namespace vha. `kubectl create namespace exercise3`
- Naviger til k8s-dag/exercises/make-a-configmap
- Opret configmap `kubectl create configmap myconfig --from-literal=some_key=some-value -n exercise3`
- Opret deployment `kubectl apply -f deployment.yml -n exercise3`
- Find navnet på pod'en med `kubectl get pods -n exercise3`

```
NAME                                READY   STATUS    RESTARTS   AGE

my-app-with-config-88fdd8444-mct6c   1/1     Running   0          7m36s
```

- Print environment med `kubectl exec -i -n exercise3 my-app-with-config-[some uuid] -- env`
- I skulle gerne se SOME_VALUE=some-value


Docker
Deployment
ReplicaSet
Pods
kubectl run nginx
HPA
Ingress
GKE
ConfigMap
Service
Secret
Job
Helm
Cluster Autoscaler
Volumes
Init Containers
GitOps
VPA
DaemonSet
StatefulSet
Upgrades
PodSecurityPolicy
NetworkPolicy
Service Mesh
PodDisruptionBudget
CRD
RBAC
Operators
Node Hardening
Image scanning
Mutating Webhooks
Admission Controllers
Bleeding-edge
Self-managed
CRI-O
EndpointSlices

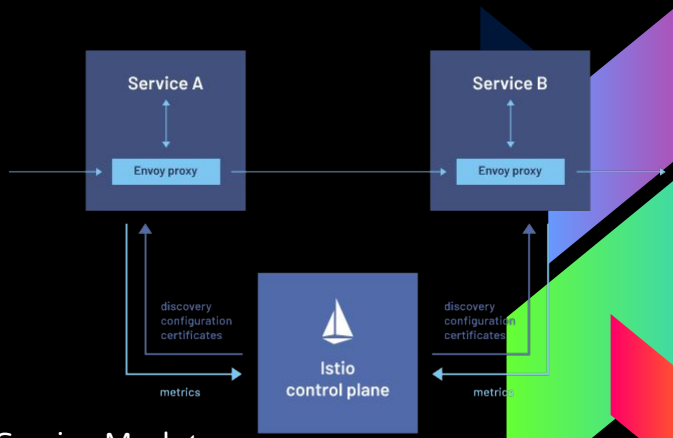


HELM

Hva' the fuck er Helm? https://helm.sh/docs/intro/quickstart/

- Der findes en masse software bygget til at køre i Kubernetes (NGINX, Prometheus, Istio, cert-manager osv. osv. osv.). Helm er bygget til at gøre installation af disse nemt.
- Helm Charts er en wrapper omkring Kubernetes manifester, der lader dig injecte specifik konfiguration i values filer
- Eksempel:

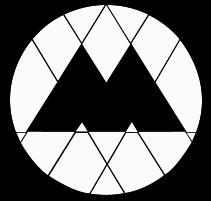
```
helm repo add bitnami https://charts.bitnami.com/bitnami

helm repo update
helm install bitnami/mysql --generate-name
```



Hvorfor skulle jeg nogensinde ville have mere end en container i en pod?

Mange gode grunde, men lad os snakke om Service Mesh

- En Service Mesh er et arkitekturlag inde i Kubernetes
- Normalt skal alt traffik mellem Pods specifikt sættes op. En Service Mesh tager sig af det for os. Pods snakker i en Service Mesh kun med hinanden gennem Sidecar proxier. På den måde får vi en masse godter “gratis”: metric collection, mTLS, network tracing og advancerede rollout taktikker
- Der findes mange varianter:
  - Istio
  - LinkerD
  - Traefik Mesh
  - OSM
  - Og mange flere



Kubernetes Tips and Tricks

Dry-run lader dig se hvordan en resource vil se ud uden at deploye den

```
kubectl run nginx --image=nginx --dry-run=client -o yaml > pod.yaml
```

Giver os et gyldigt kubernetes manifest, som vi kan redigere i.

Kubernetes kan give os shell adgang til en pod

```
kubectl exec -i -n exercise3 my-app-with-config-88fdd8444-mct6c -- /bin/bash
```

Super brugbart til debugging

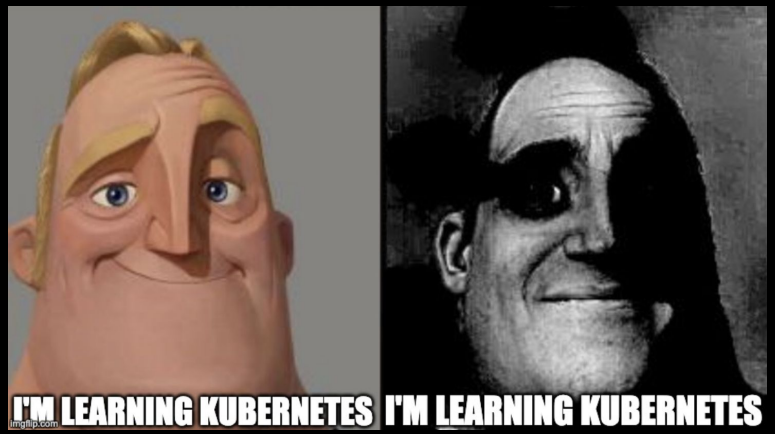
Gode kilder osv.

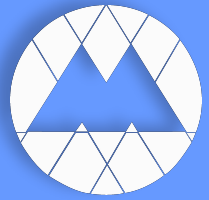
Kubernetes Docs: https://kubernetes.io/docs/home/

CNCF Landscape: https://landscape.cncf.io/

Kæmpe Repo med Helm Charts: https://artifacthub.io/

Diverse Kubernetes Playgrounds og eksamens simulatore: https://killercoda.com/





Tak for i dag! Husk at tilføje Kubernetes til jeres skills på Sursen