

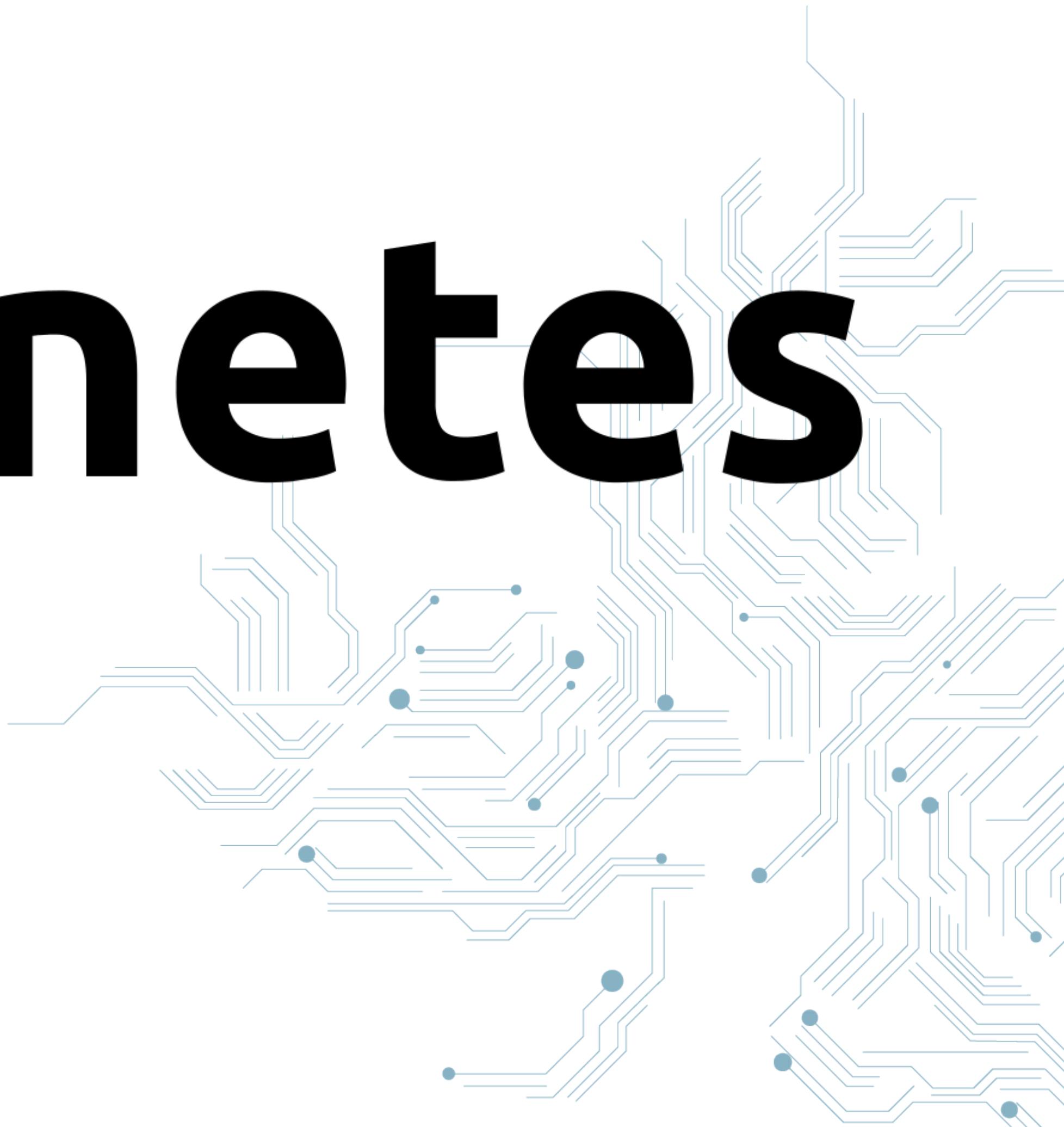
Kubernetes Workshop

Max Rosin & Maurice „Morre“ Meyer

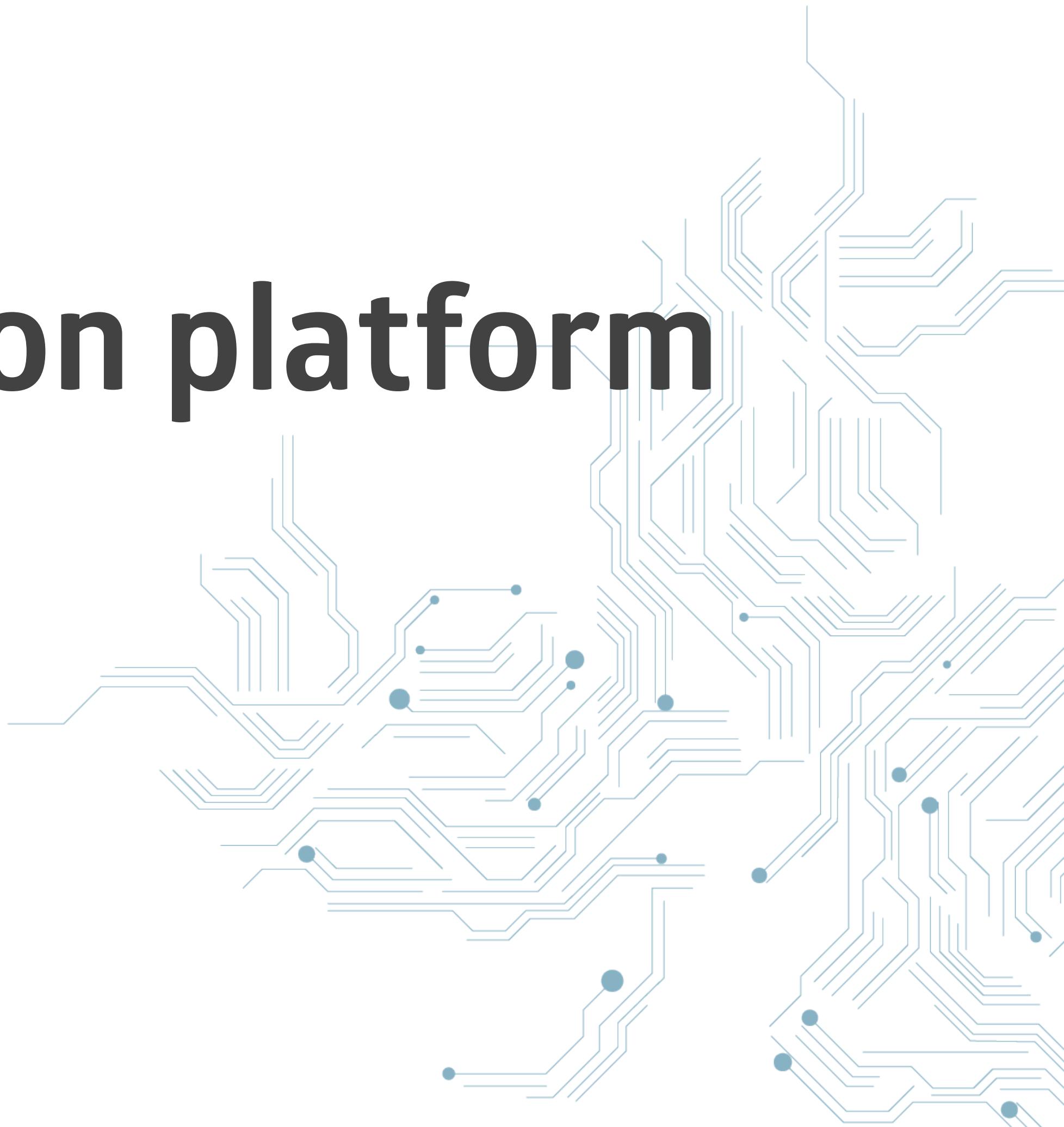




kubernetes



Container orchestration platform



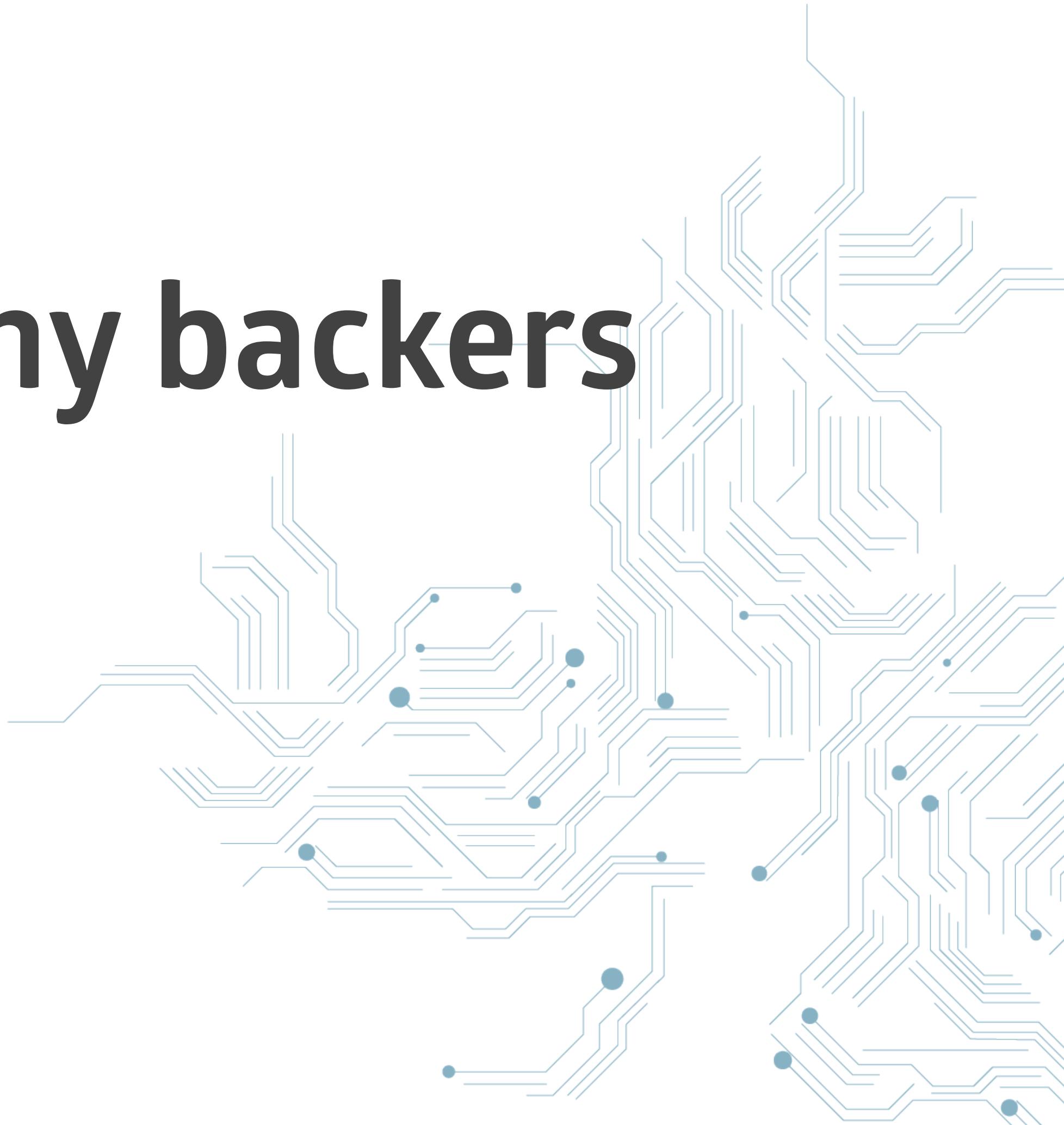
Deploy, run and scale your services in isolated containers



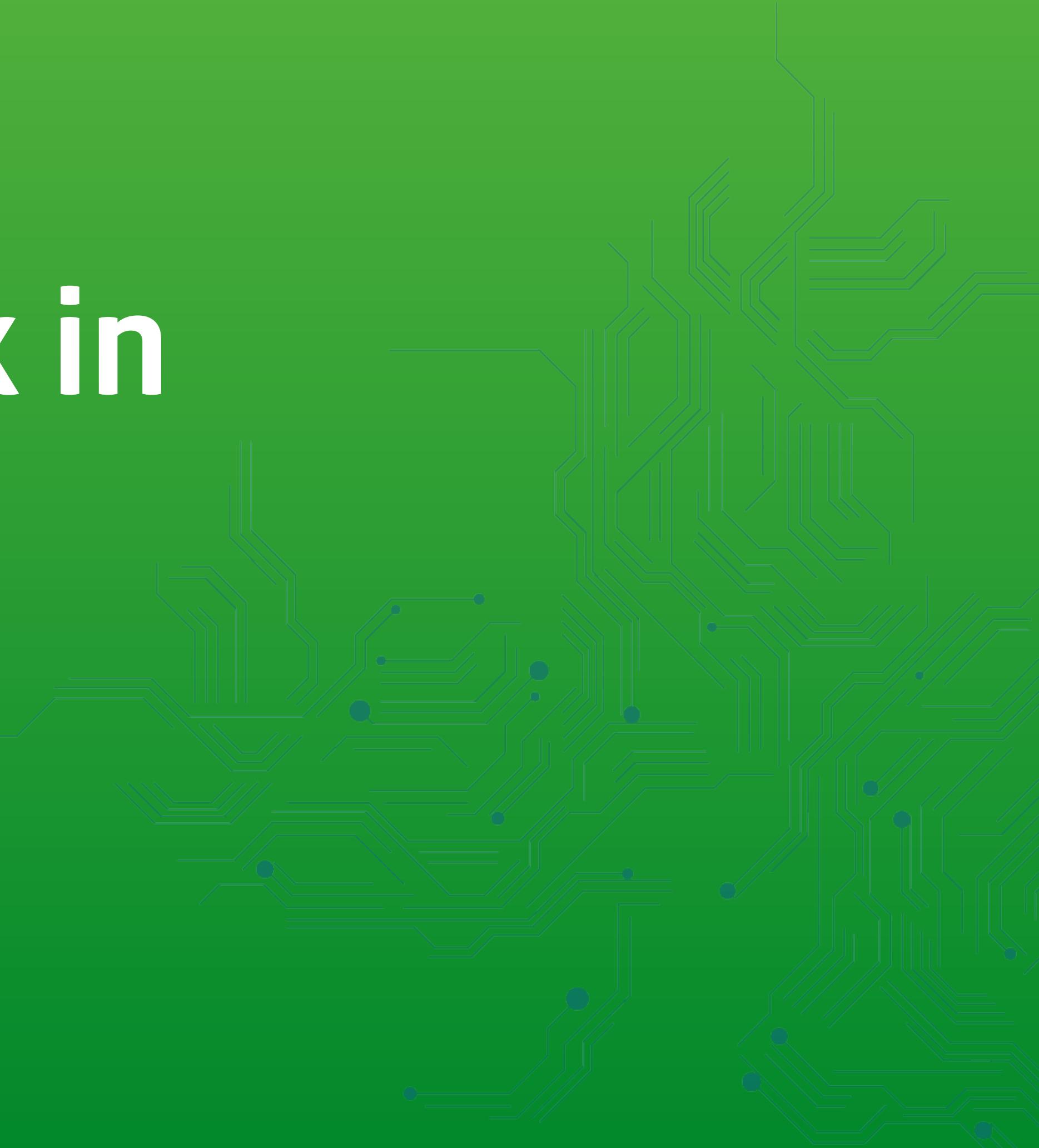
Very Powerful



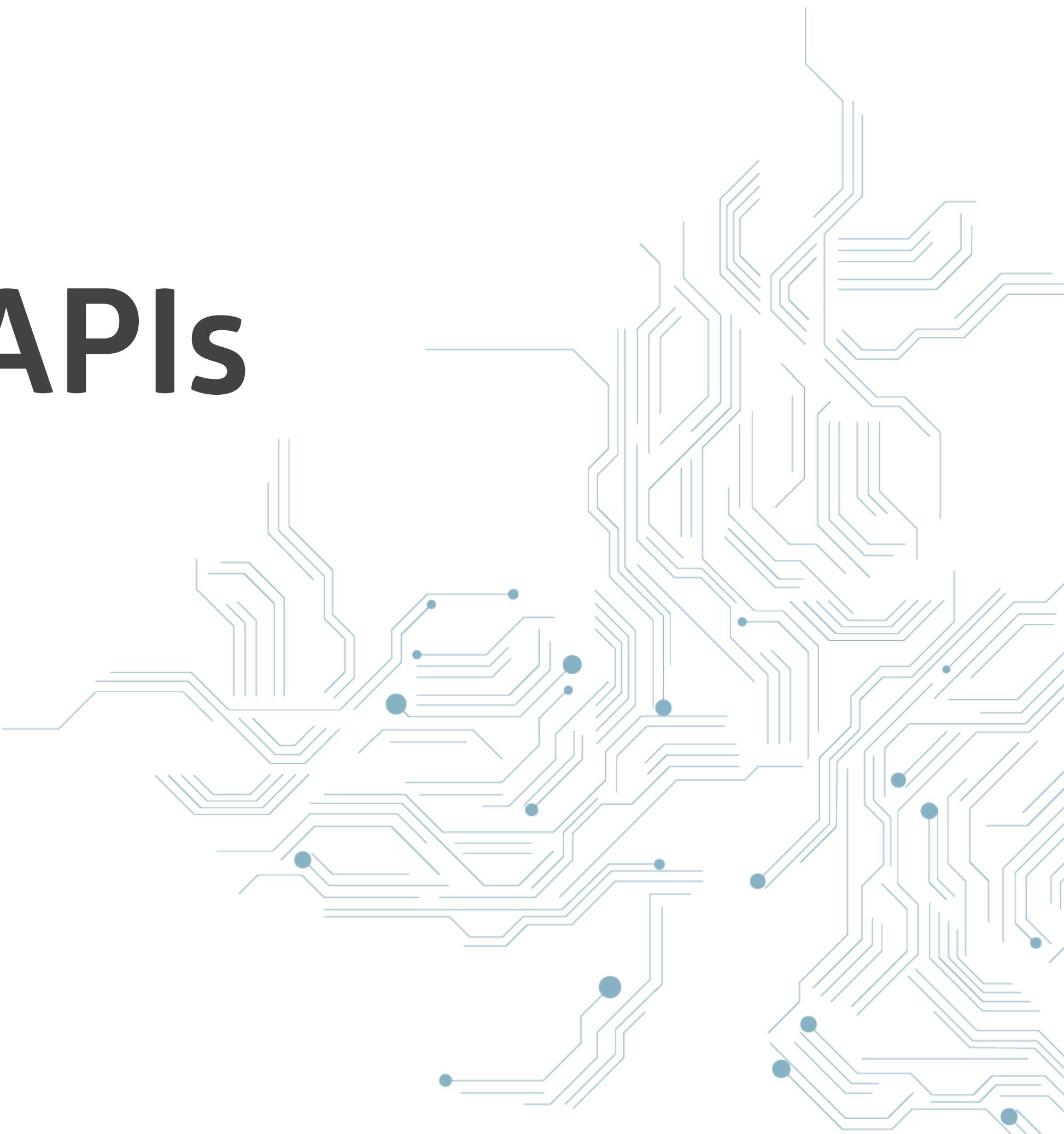
Lot's of large company backers



No vendor lock in



Standardized APIs



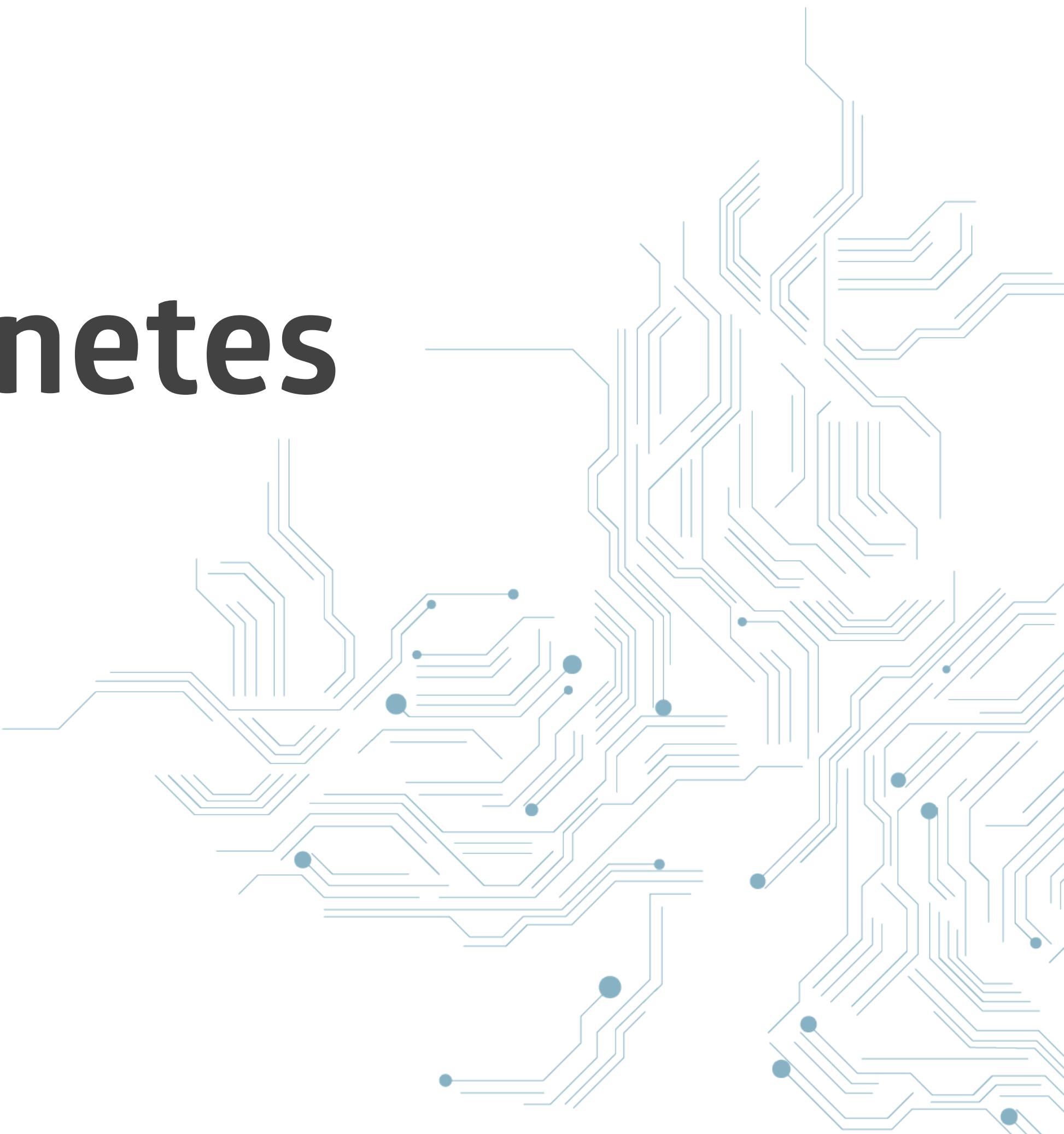
Runs on
Your laptop
Bare metal
Cloud Providers



**And if you don't want to install and
maintain Kubernetes yourself**



Managed Kubernetes



CNCF Cloud Native Interactive Landscape

The Cloud Native Trail Map ([png](#), [pdf](#)) is CNCF's recommended path through the cloud native landscape. The cloud native landscape ([png](#), [pdf](#)), serverless landscape ([png](#), [pdf](#)), and member landscape ([png](#), [pdf](#)) are dynamically generated below. Please [open](#) a pull request to correct any issues. Greyed logos are not open source. Last Updated: 2020-07-22 23:44:41Z

You are viewing 45 cards with a total of 746 stars, market cap of \$6.97T and funding of \$1.55B.

Landscape

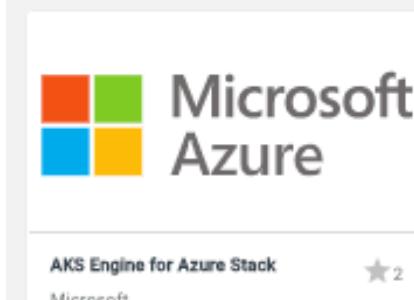
Card Mode

Serverless

Members

[Tweet](#) 1163

Platform - Certified Kubernetes - Hosted (45)



AKS Engine for Azure Stack
Microsoft



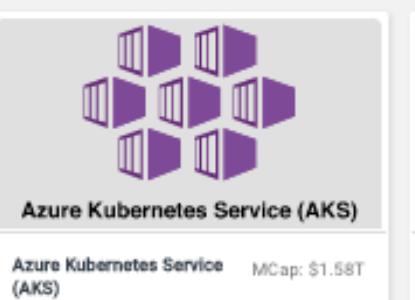
Alibaba Cloud Container Service for Kubernetes
Alibaba Cloud



Amazon Elastic Container Service for Kubernetes (EKS)
Amazon Web Services



Azure (AKS) Engine
Microsoft



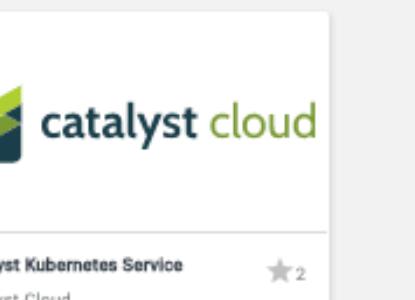
Azure Kubernetes Service (AKS)
Microsoft



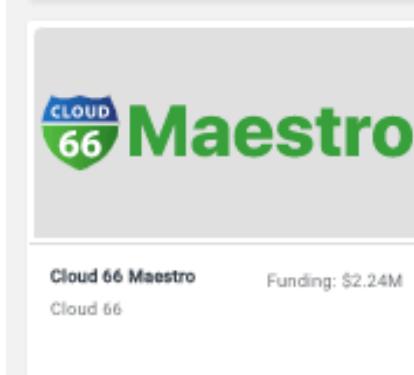
Baidu Cloud Container Engine
Baidu



BoCloud BeyondContainer
BoCloud



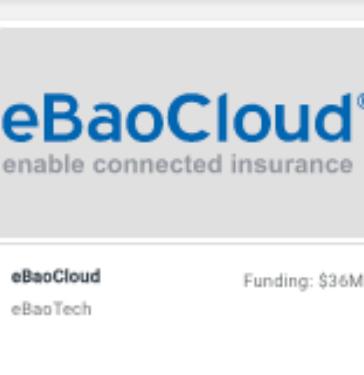
Catalyst Kubernetes Service
Catalyst Cloud



Cloud 66 Maestro
Cloud 66



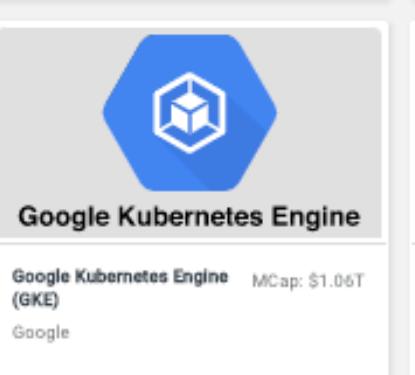
DigitalOcean Kubernetes
DigitalOcean



eBaoCloud enable connected insurance
eBaoTech



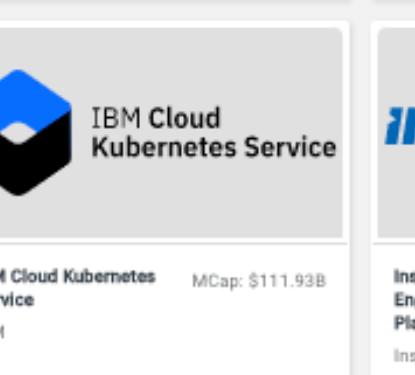
ELASTX Private Kubernetes
ELASTX



Google Kubernetes Engine (GKE)
Google



Huawei Cloud Container Engine (CCE)
Huawei Technologies



IBM Cloud Kubernetes Service
IBM



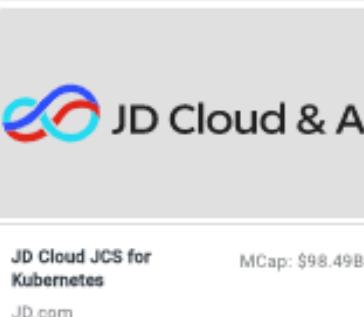
Inspur Cloud Container Engine and Inspur Open Platform
Inspur Group



Inspur ICKS
Inspur Group



Inspur Open Platform for ARM
Inspur Group



JD Cloud JCS for Kubernetes
JD.com



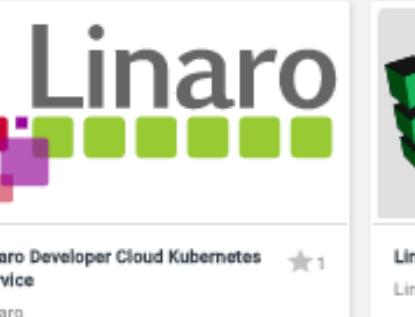
JD.com JDOS Hosted
JD.com



Kingsoft Container Engine
Kingsoft



Launcher Tech LStack Container Service for Kubernetes
Hangzhou Launcher Technology



Linaro Developer Cloud Kubernetes Service
Linaro



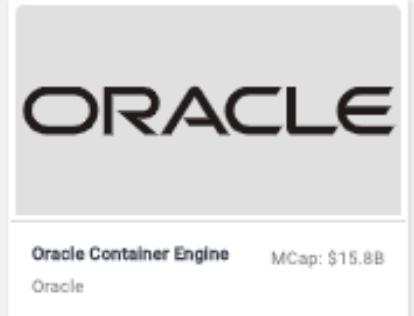
Mail.Ru Cloud Containers
Mail.Ru Group



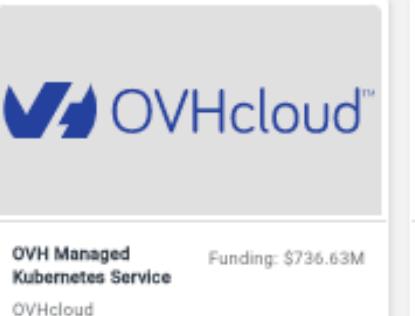
Nirmata Managed Kubernetes
Nirmata



Nutanix Karbon
Nutanix



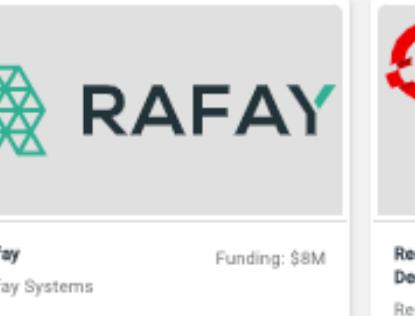
Oracle Container Engine
Oracle



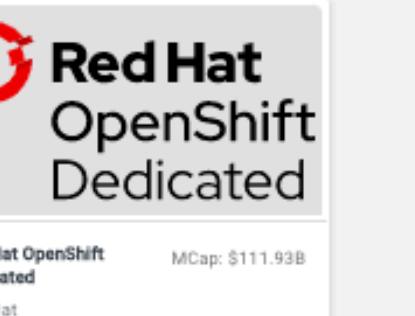
OVH Managed Kubernetes Service
OVHcloud



QingCloud KubeSphere Engine (QKE)
QingCloud



Rafay
Rafay Systems



Red Hat OpenShift Dedicated
Red Hat



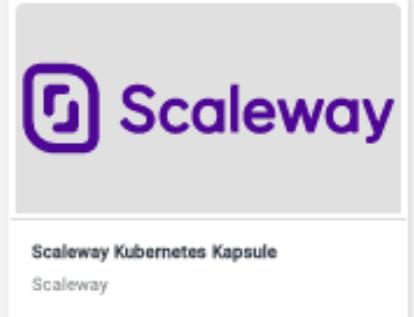
Red Hat OpenShift on IBM Cloud
IBM



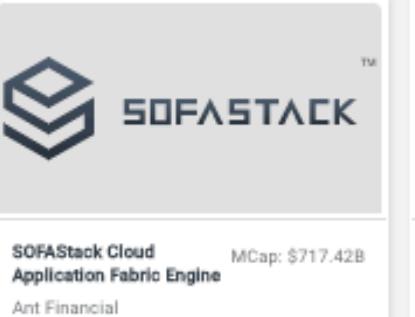
Samsung SDS Kubernetes Service
Samsung SDS



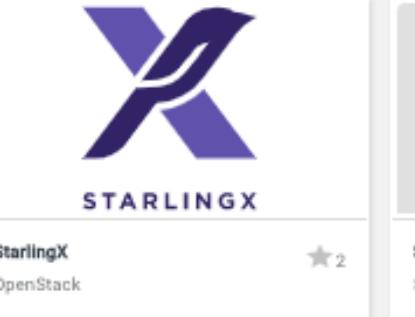
SAP Certified Gardener
SAP



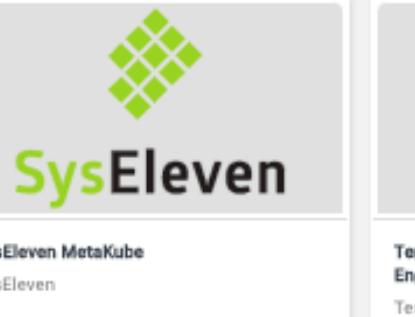
Scaleway Kubernetes Capsule
Scaleway



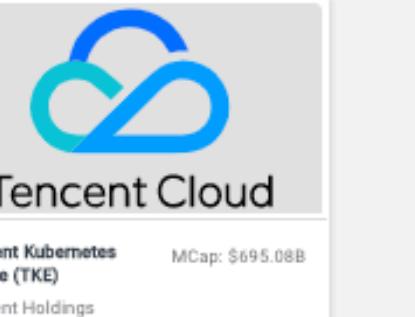
SOFASTACK Cloud Application Fabric Engine
Ant Financial



StarlingX
OpenStack



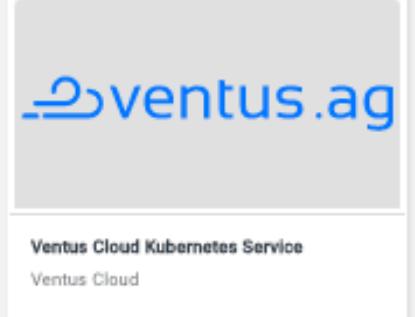
SysEleven MetaKube
SysEleven



Tencent Kubernetes Engine (TKE)
Tencent Holdings



UCloud Kubernetes Service (UKBS)
Ucloud Information Technology



Ventus Cloud Kubernetes Service
Ventus Cloud



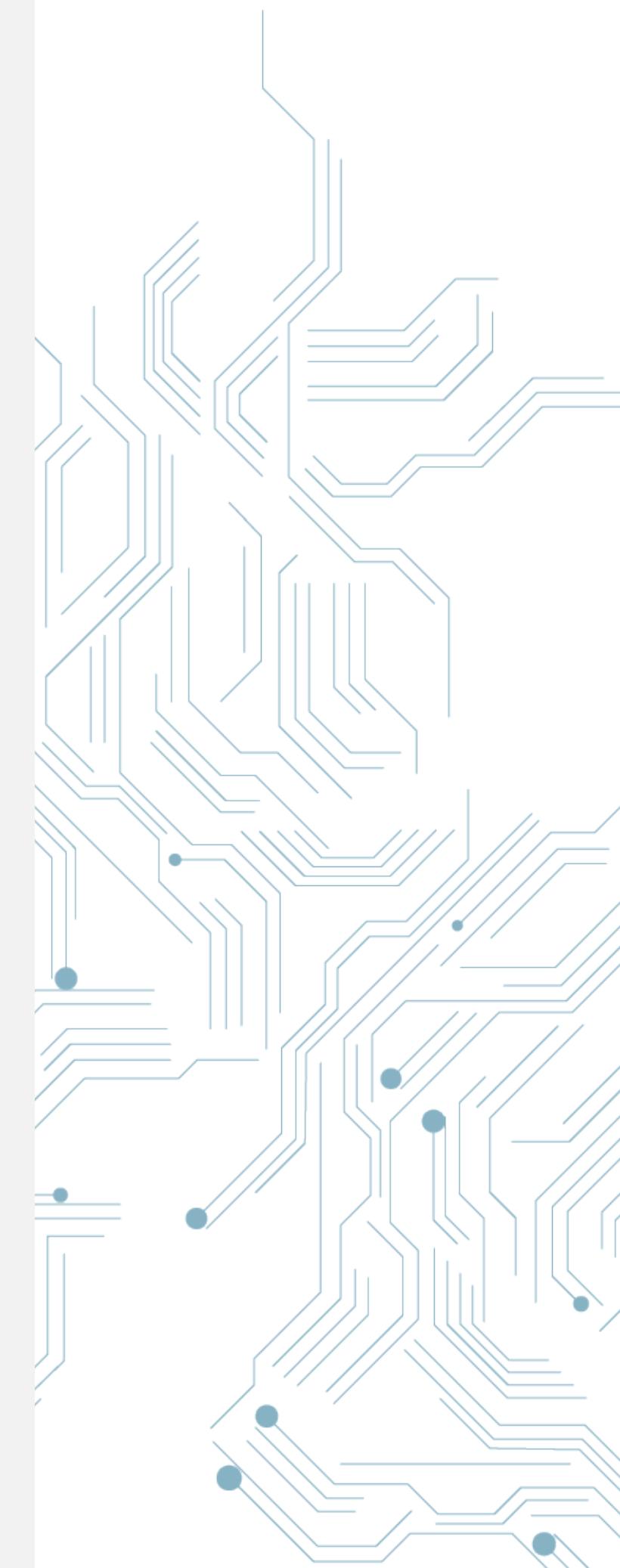
VEXXHOST Kubernetes Container Service
VEXXHOST



WANGSU CLOUD Container Service
Wangsu Science & Technology



ZTE TECS OpenPalette
ZTE



Easy setup



Easy upgrades



Easy scaling



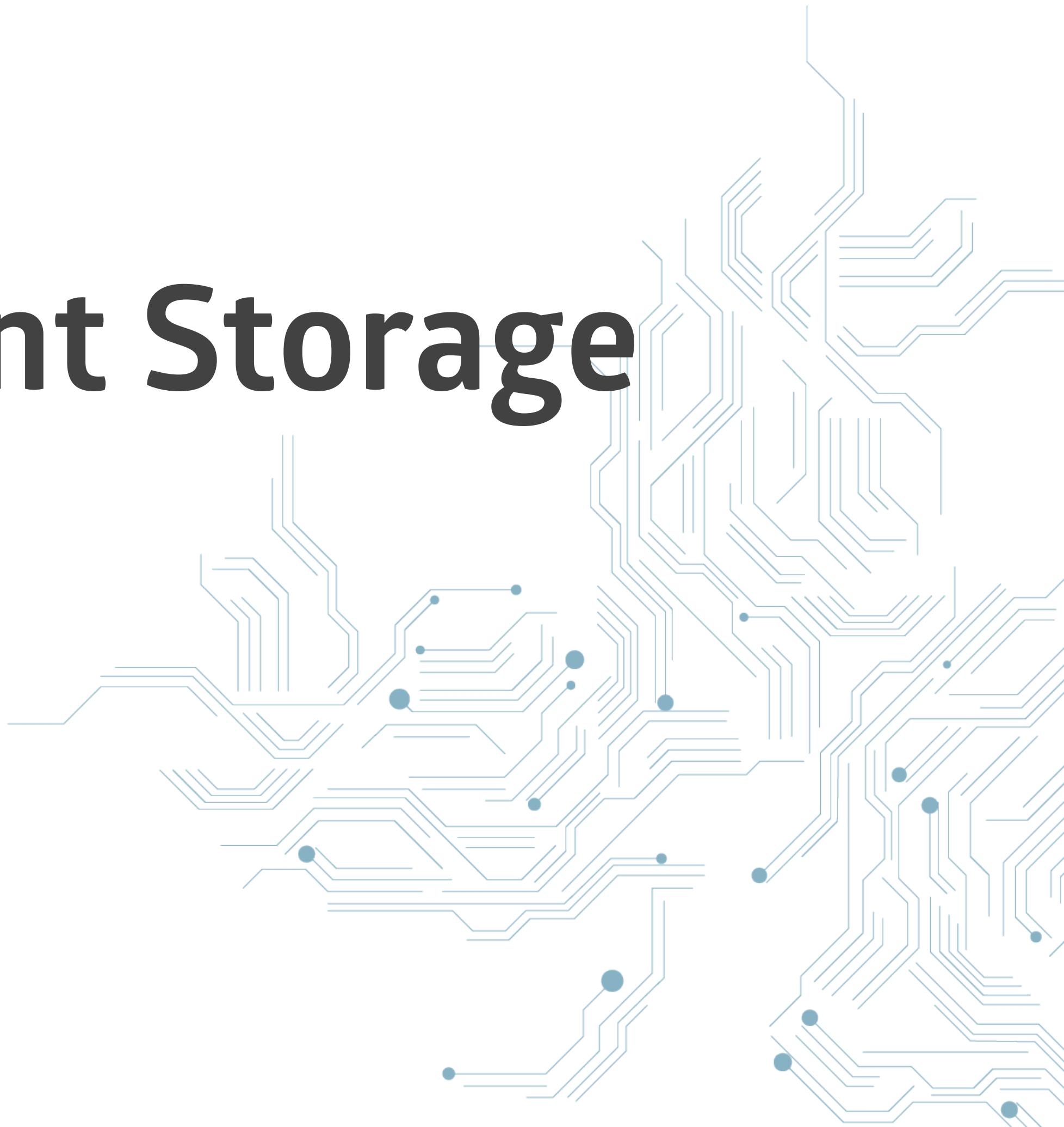
Features



Load Balancing



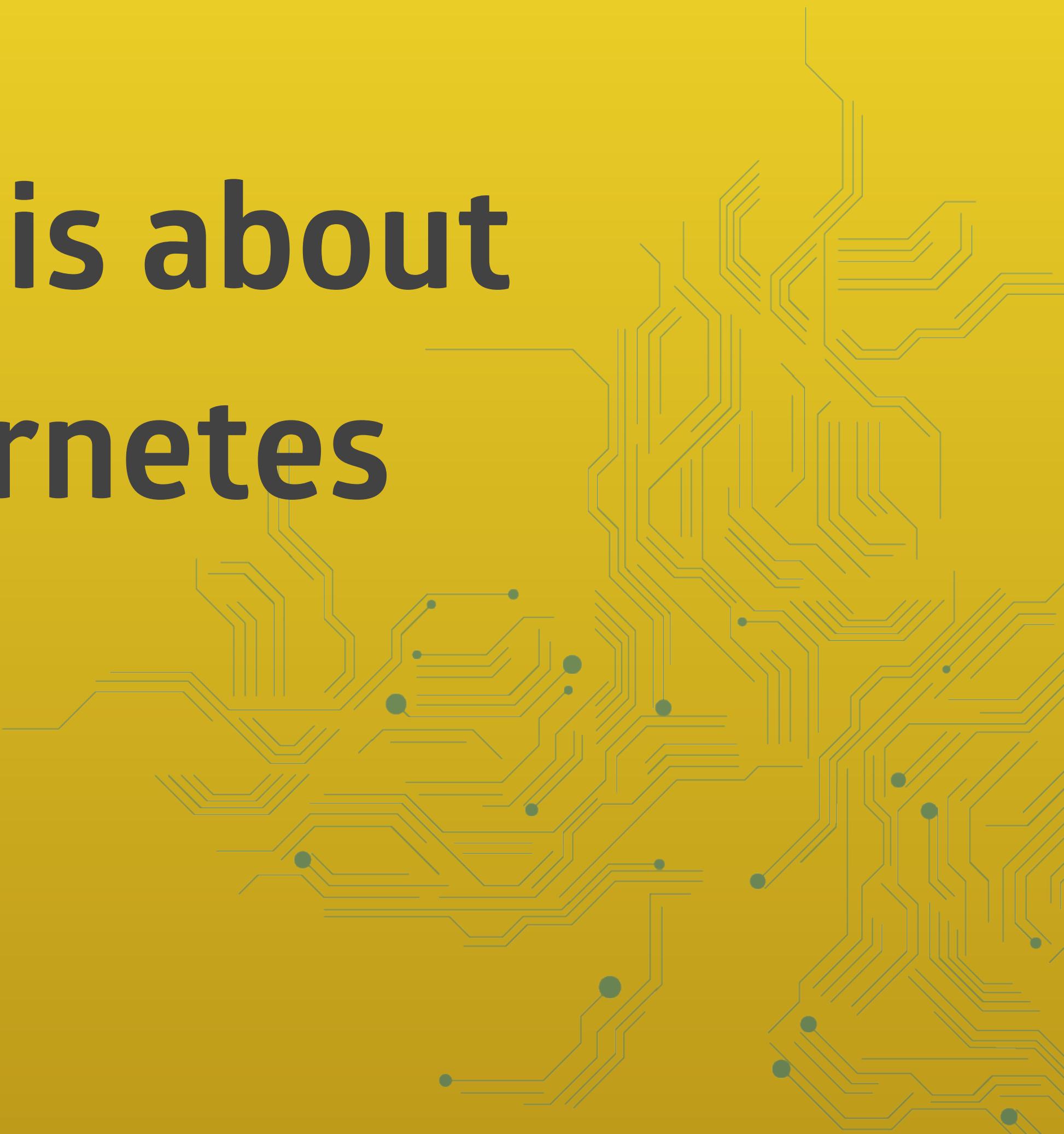
Distributed Persistent Storage



Backups



**But this workshop is about
how to use Kubernetes**



Learning curve

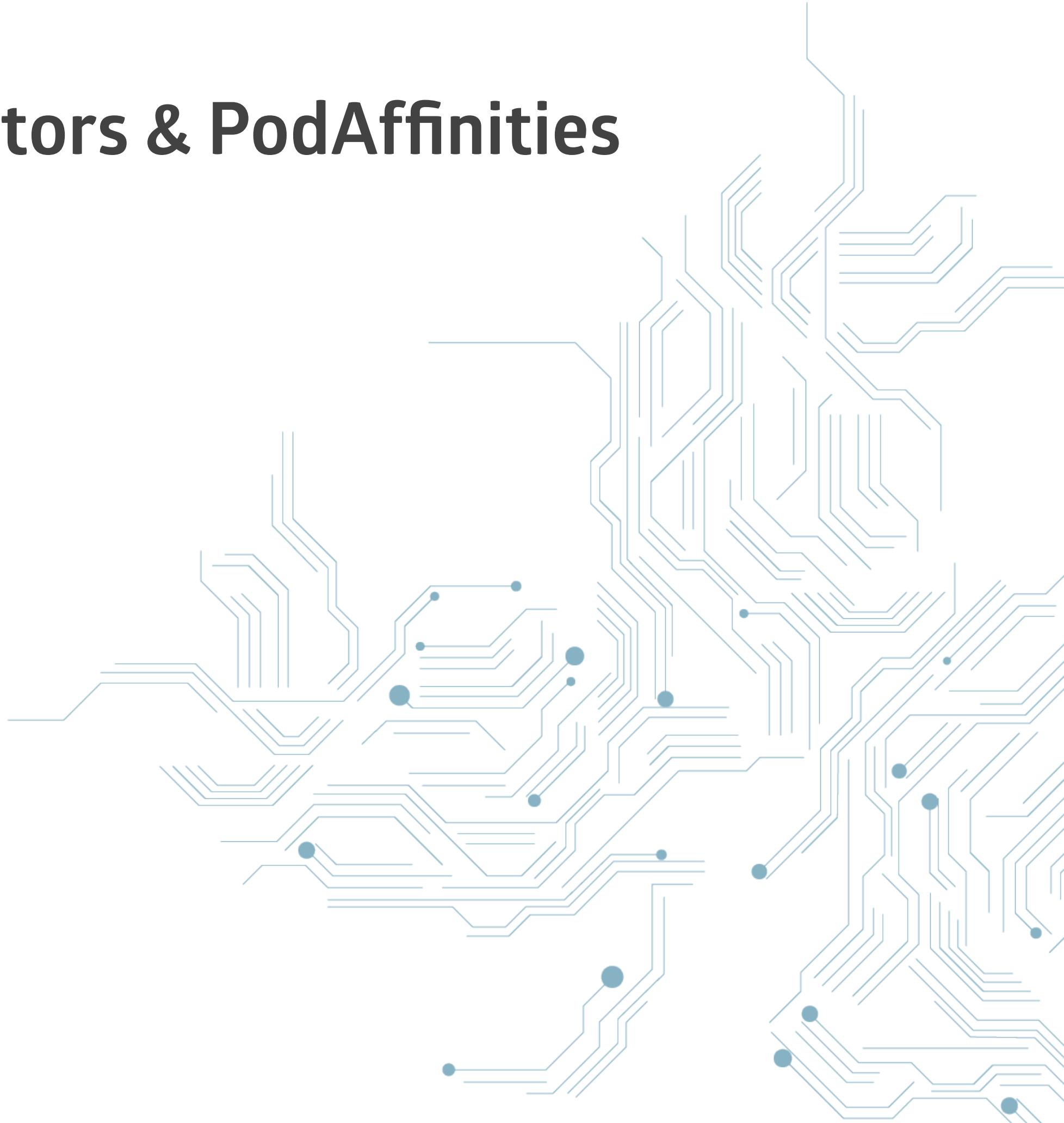


Agenda

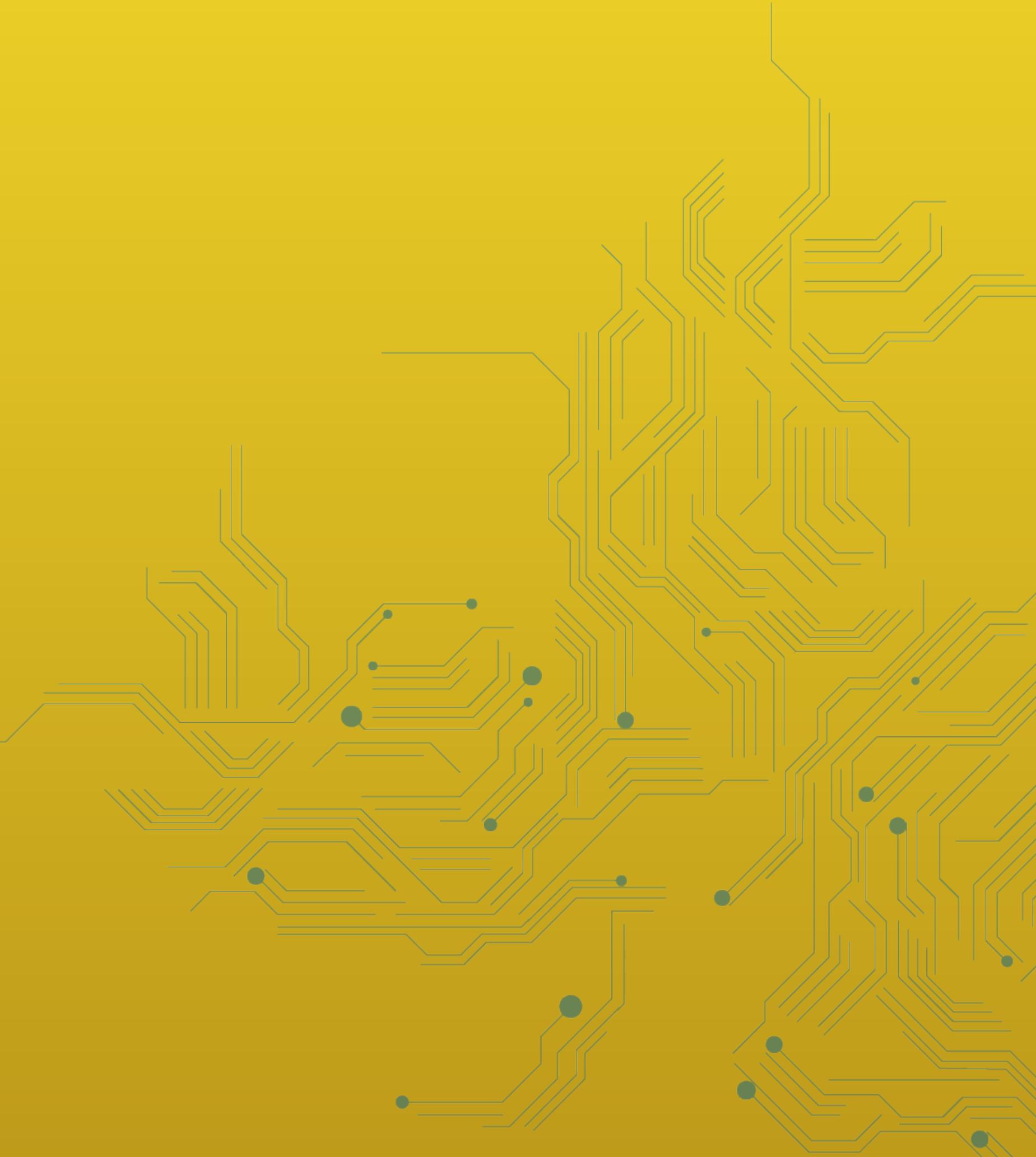




- **Deployments**
- **Readiness and Liveness-Probes, NodeSelectors & PodAffinities**
- **ConfigMaps & Secrets**
- **External DNS**
- **Let'sEncrypt with cert-manager**
- **nginx-ingress-controller**
- **Helm**



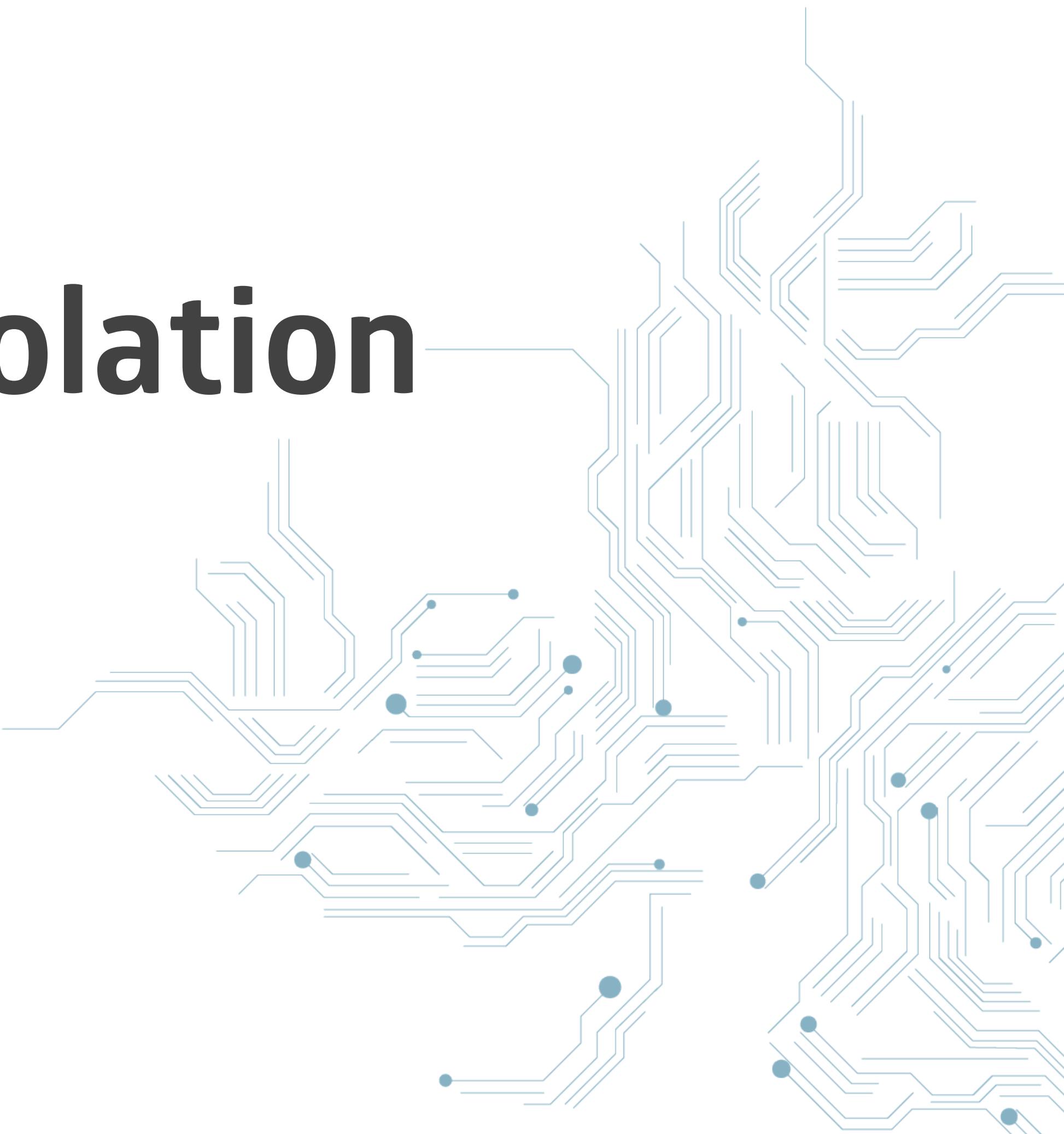
But first



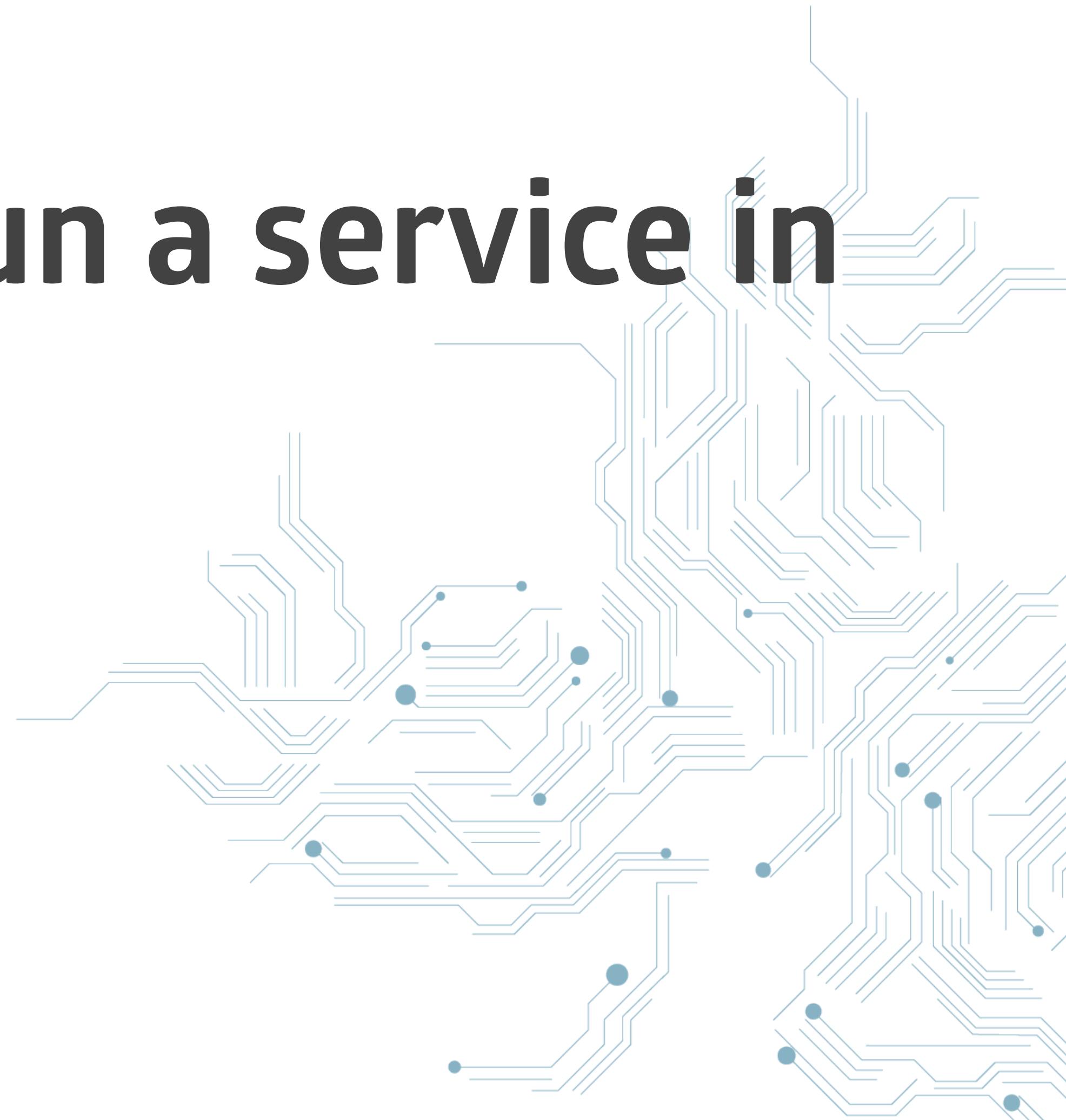
Why containers?



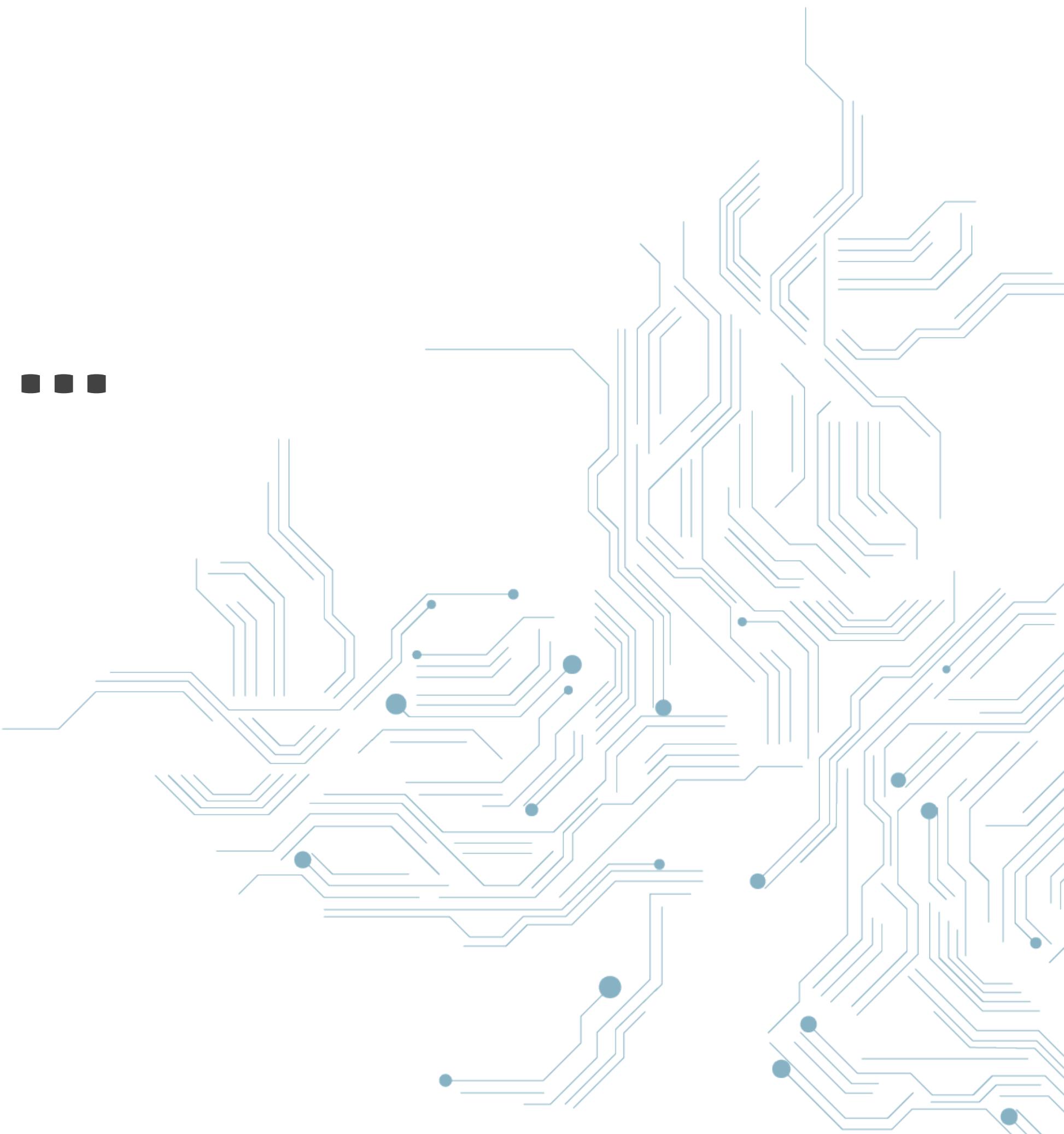
Services run in isolation



**Everything needed to run a service in
one image**



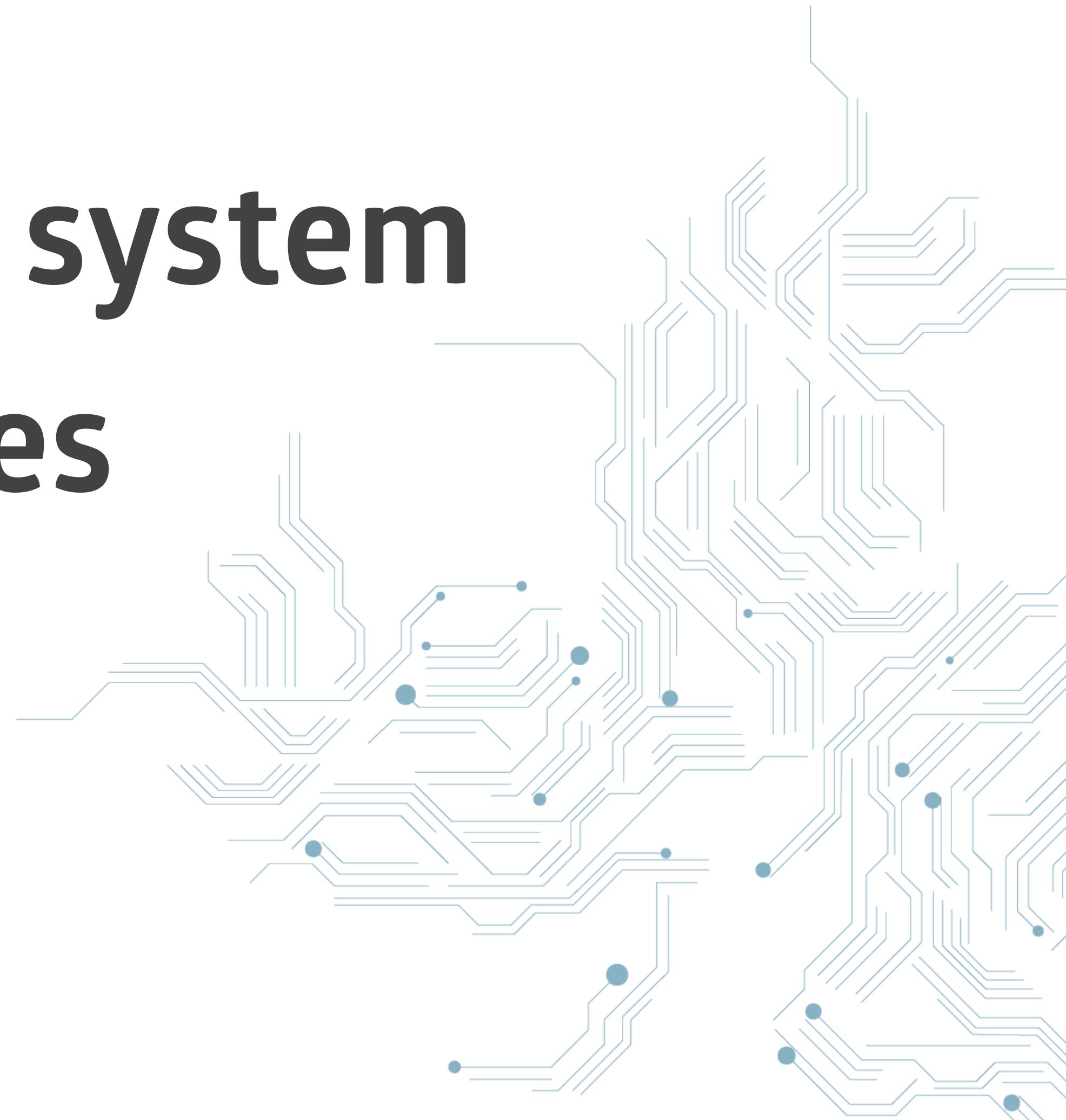
Make things ...



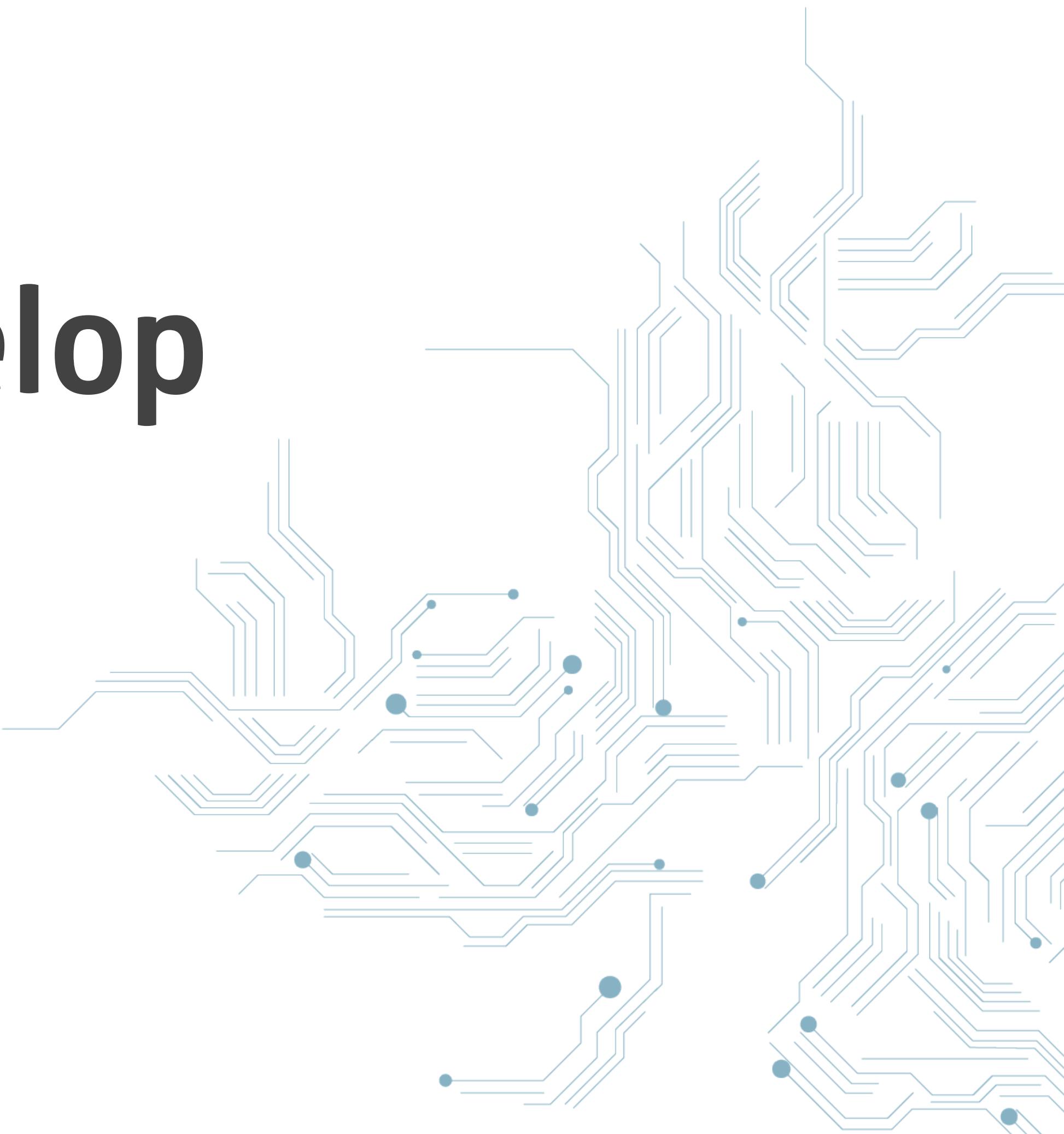
Easier to deploy



Easier to upgrade system dependencies



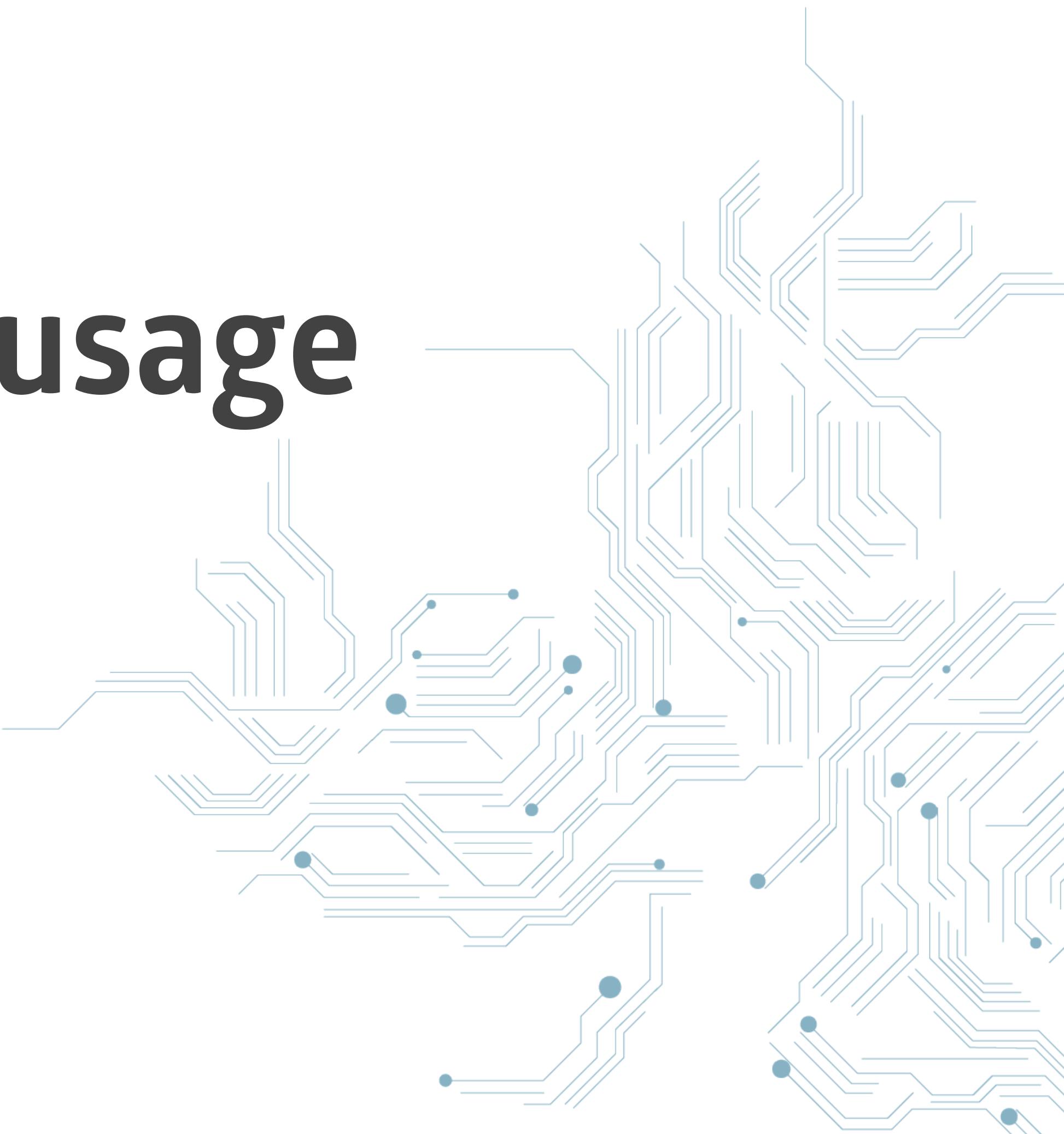
Easier to develop

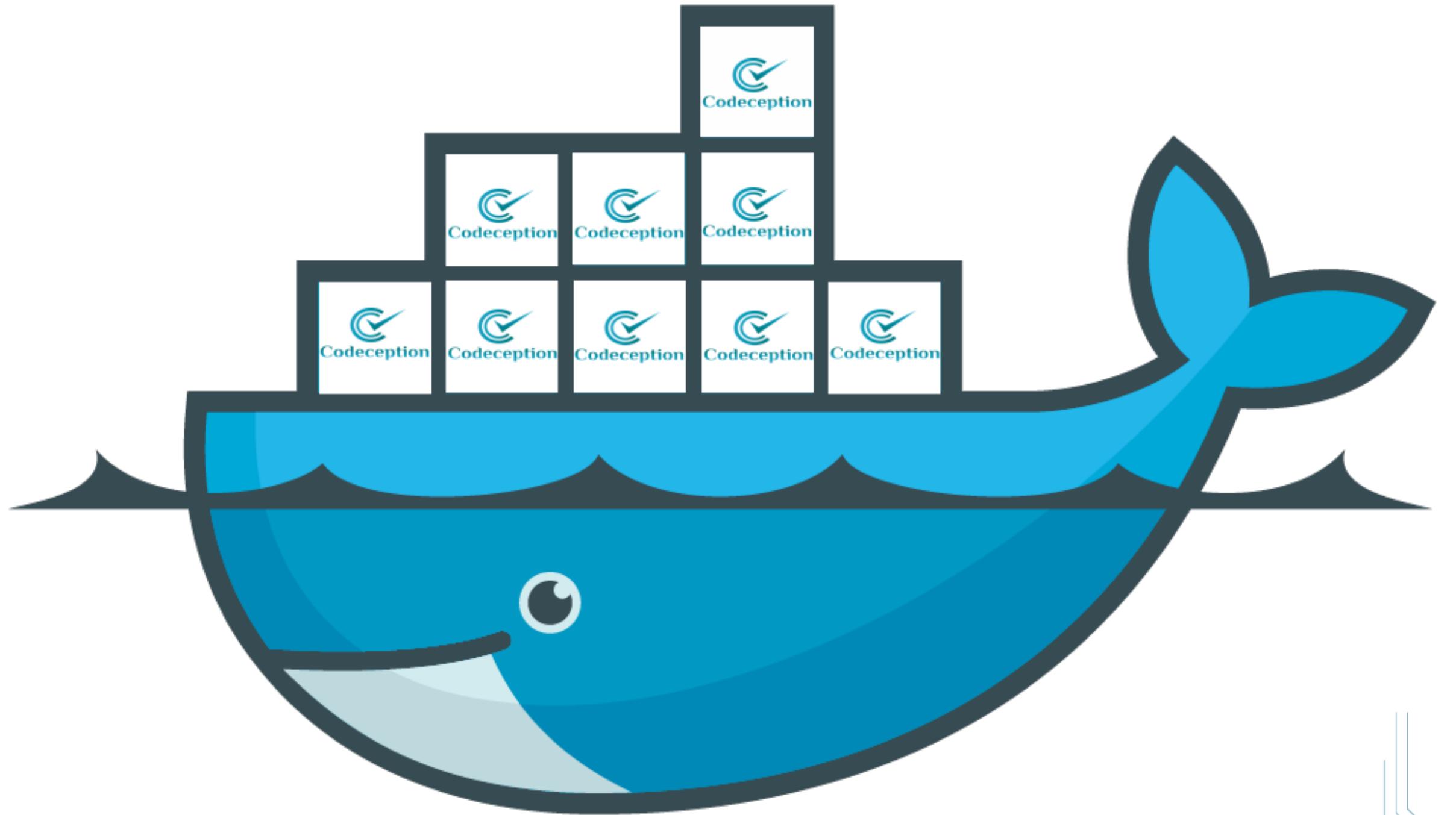


Easier to scale

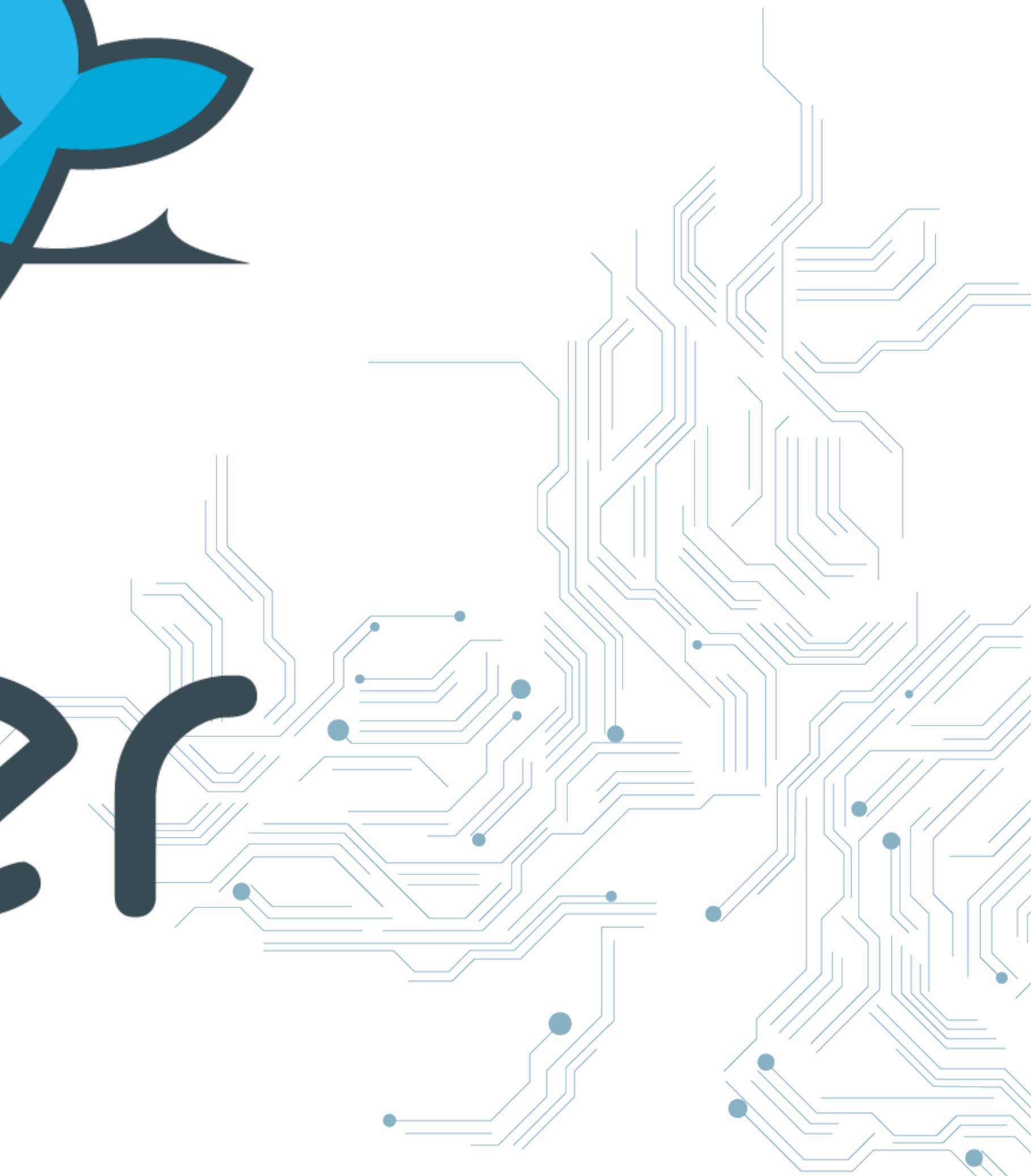


Better resource usage





docker



```
FROM php:7.2-apache  
WORKDIR /var/www/html
```

```
RUN apt-get update -y && \  
    apt-get install -y --no-install-recommends curl \  
    rm -rf /var/lib/apt/lists/*
```

```
ENV TMP_DIR /tmp
```

```
COPY . /var/www/html/
```

```
EXPOSE 80
```

```
docker build -t gitlab.syseleven.de/syseleven/symfony-demo:2.0.0 .
```



```
docker run -p 8080:80 syseleven/symfony-demo:2.0.0  
docker push syseleven/symfony-demo:2.0.0
```



Kubernetes helps you to run and deploy containers



Let's define some core concepts and
terminology first

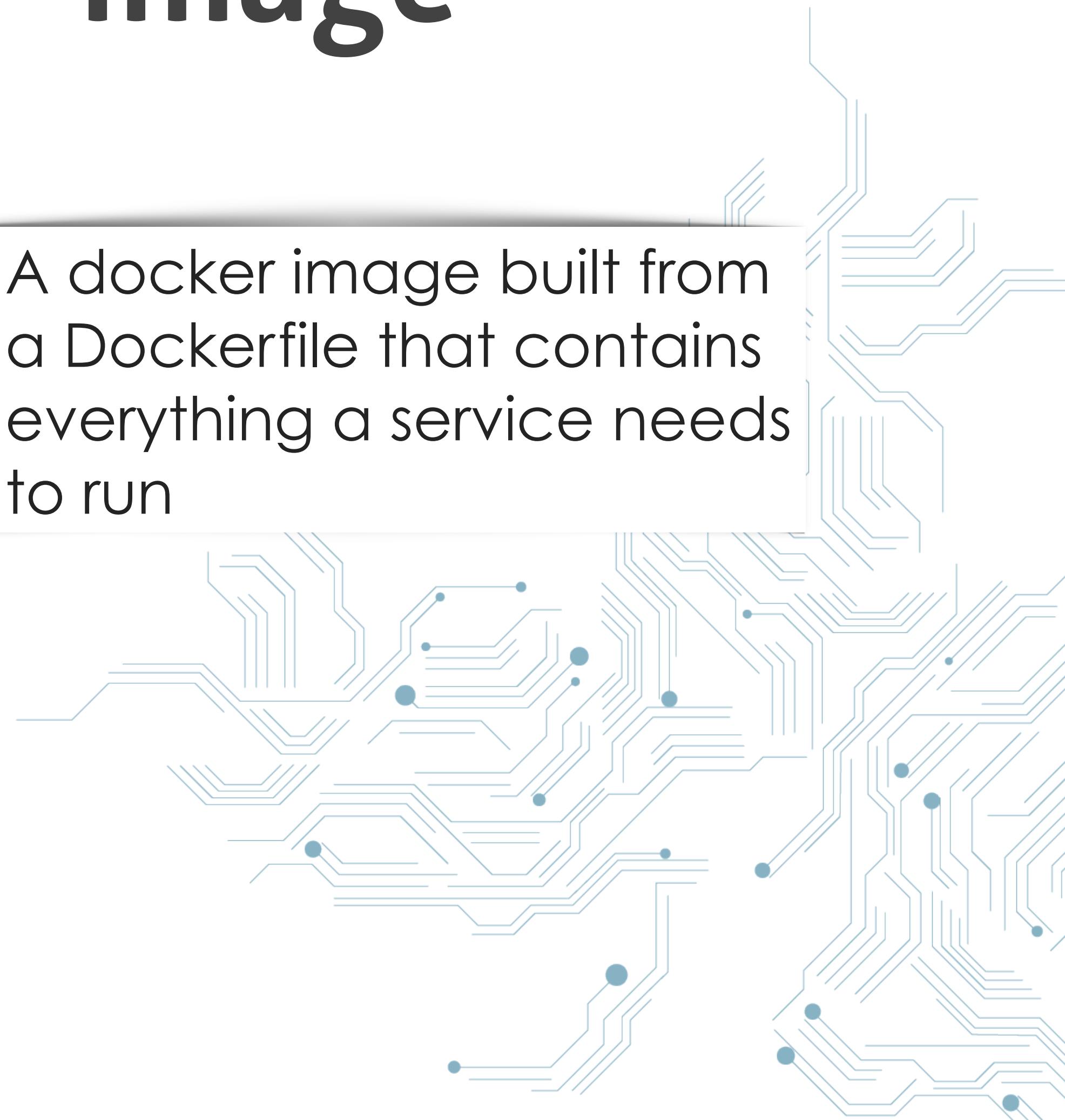


Kubernetes Cluster



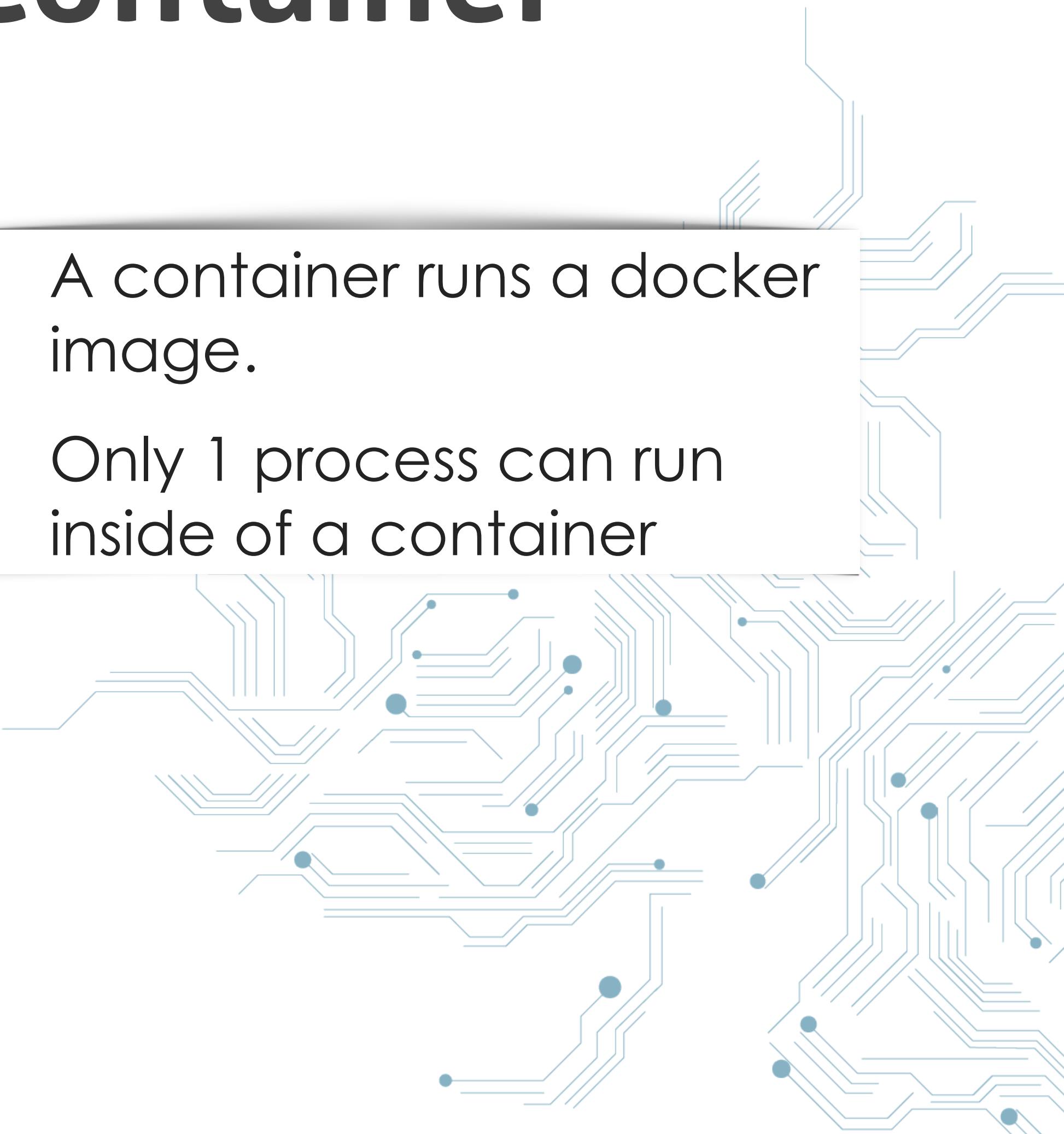
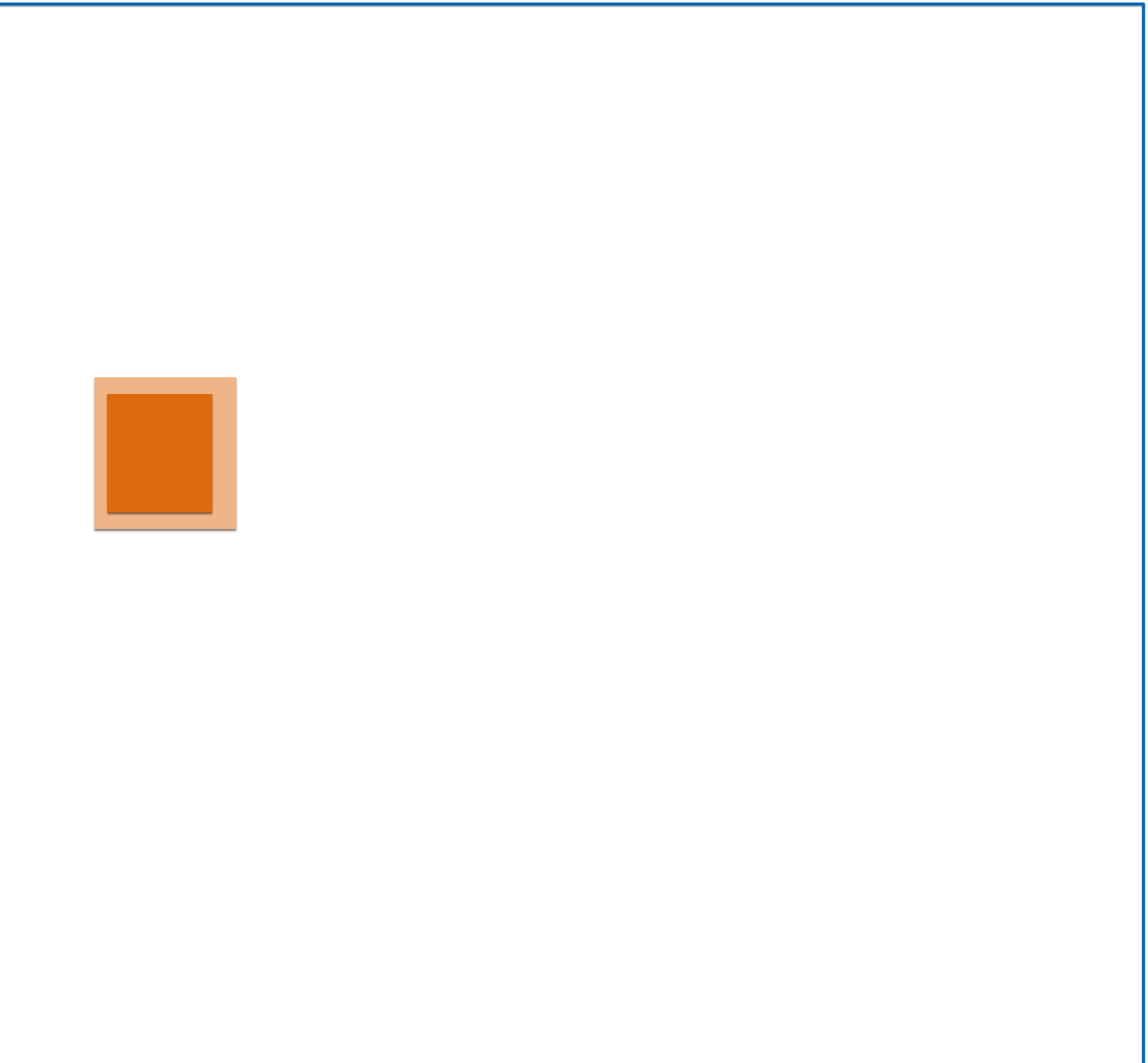
Image

- A docker image built from a Dockerfile that contains everything a service needs to run

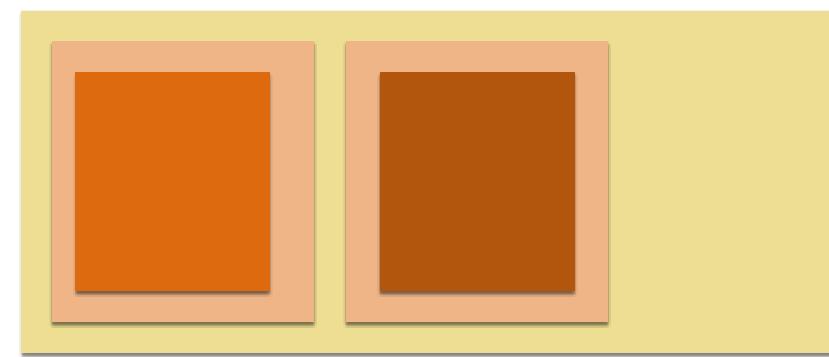


Container

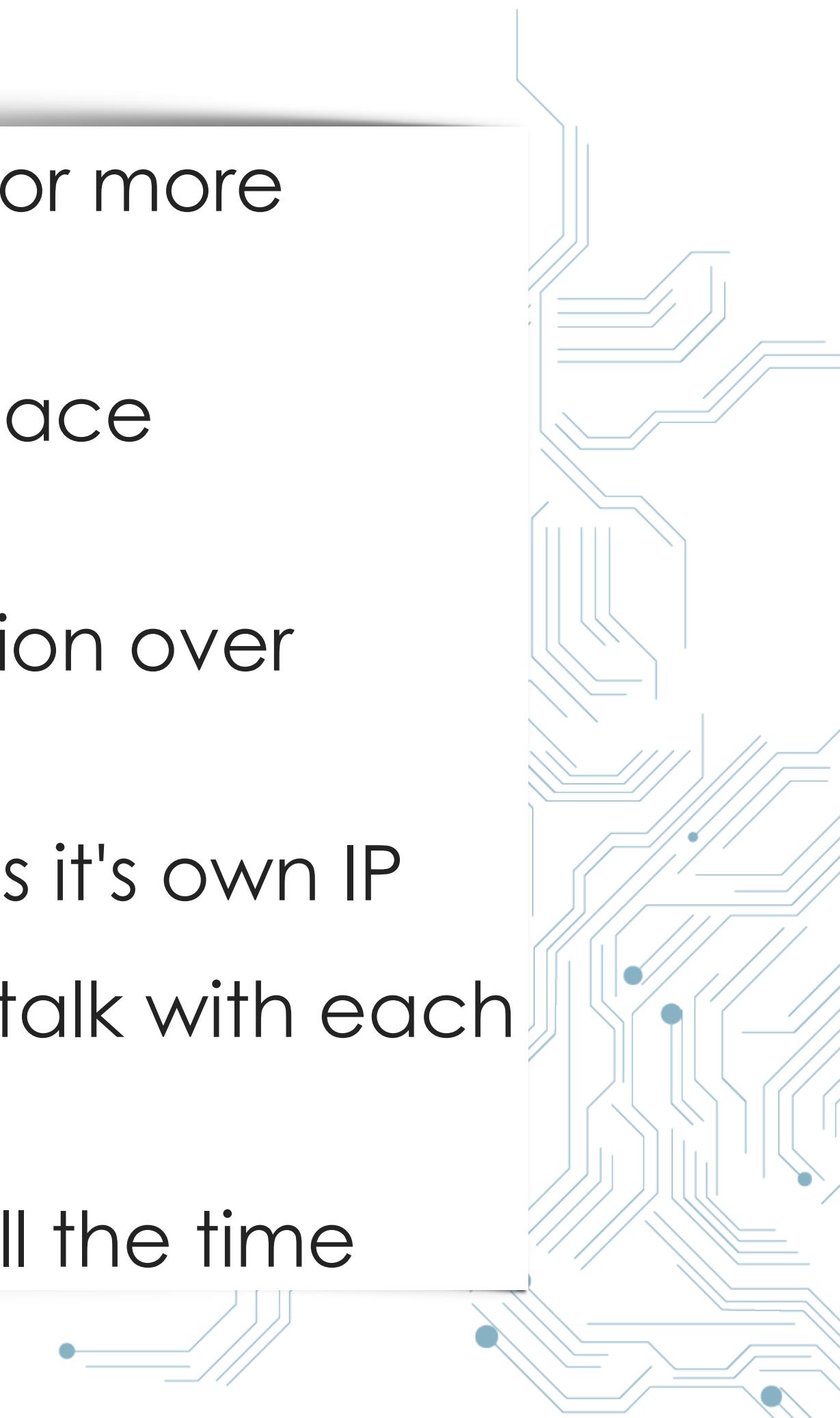
- A container runs a docker image.
- Only 1 process can run inside of a container



Pod

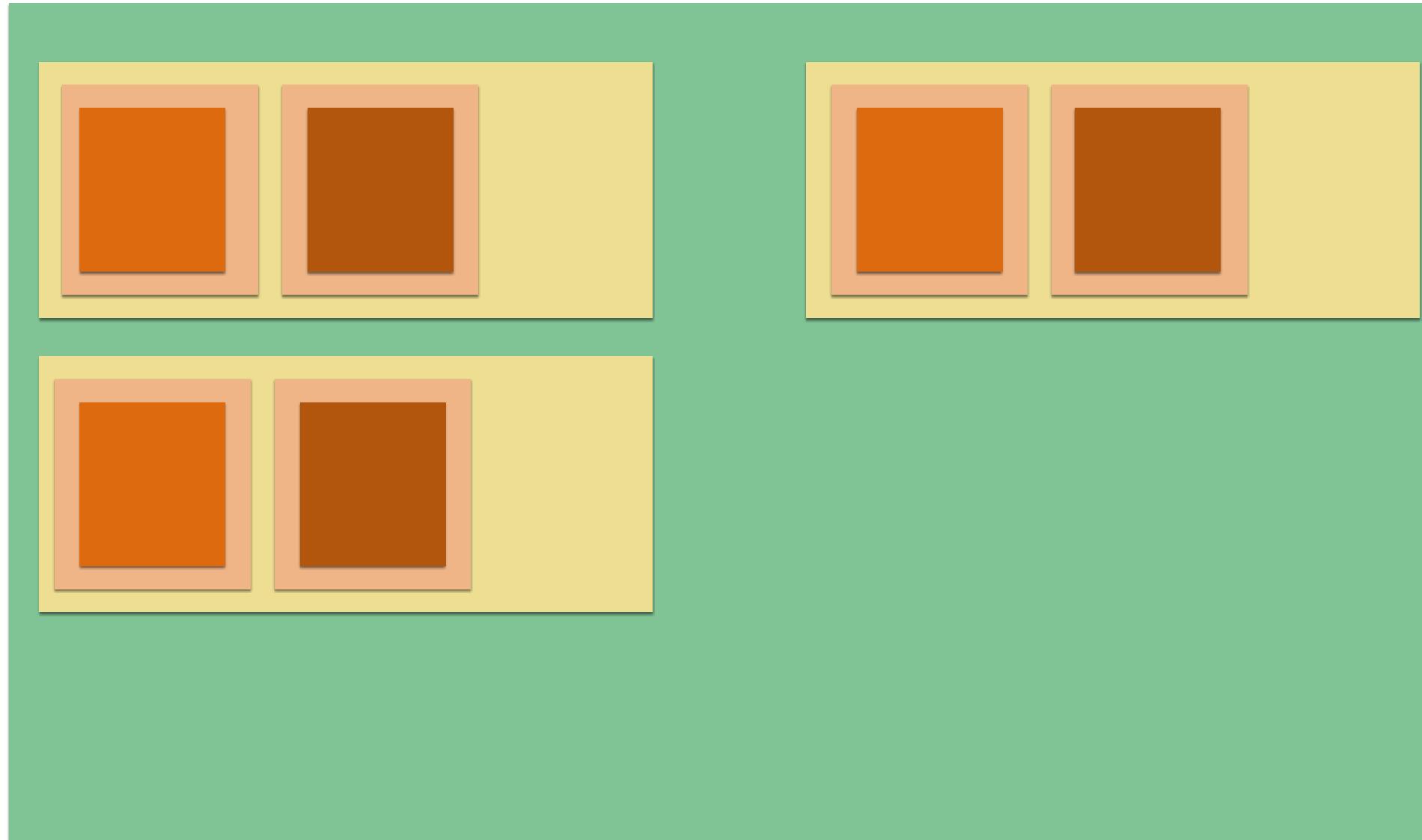


- A group of 1 or more containers
- Same port space
- Within a Pod:
communication over localhost
- Every Pod has it's own IP
- All Pods can talk with each other
- IPs change all the time



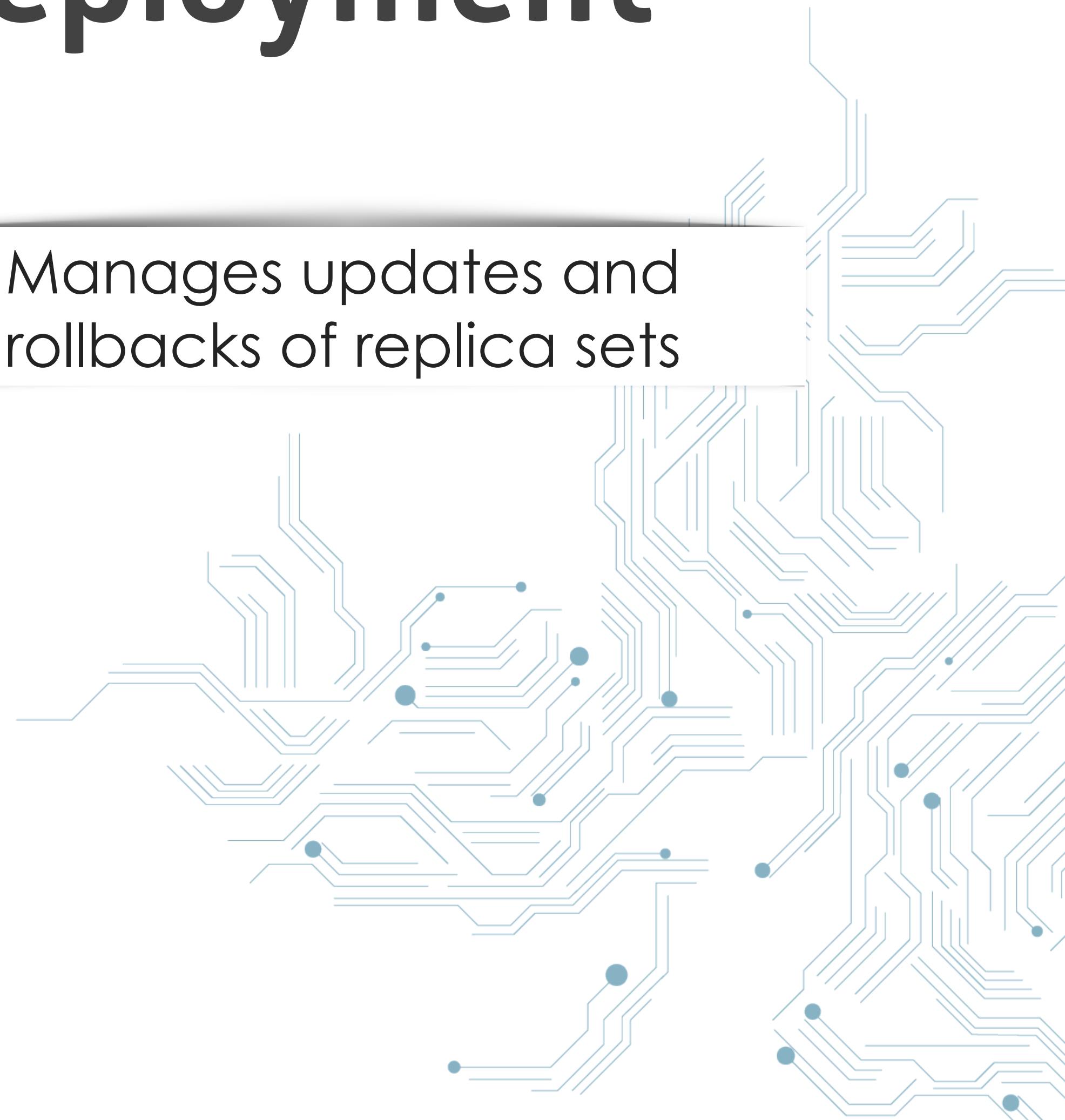
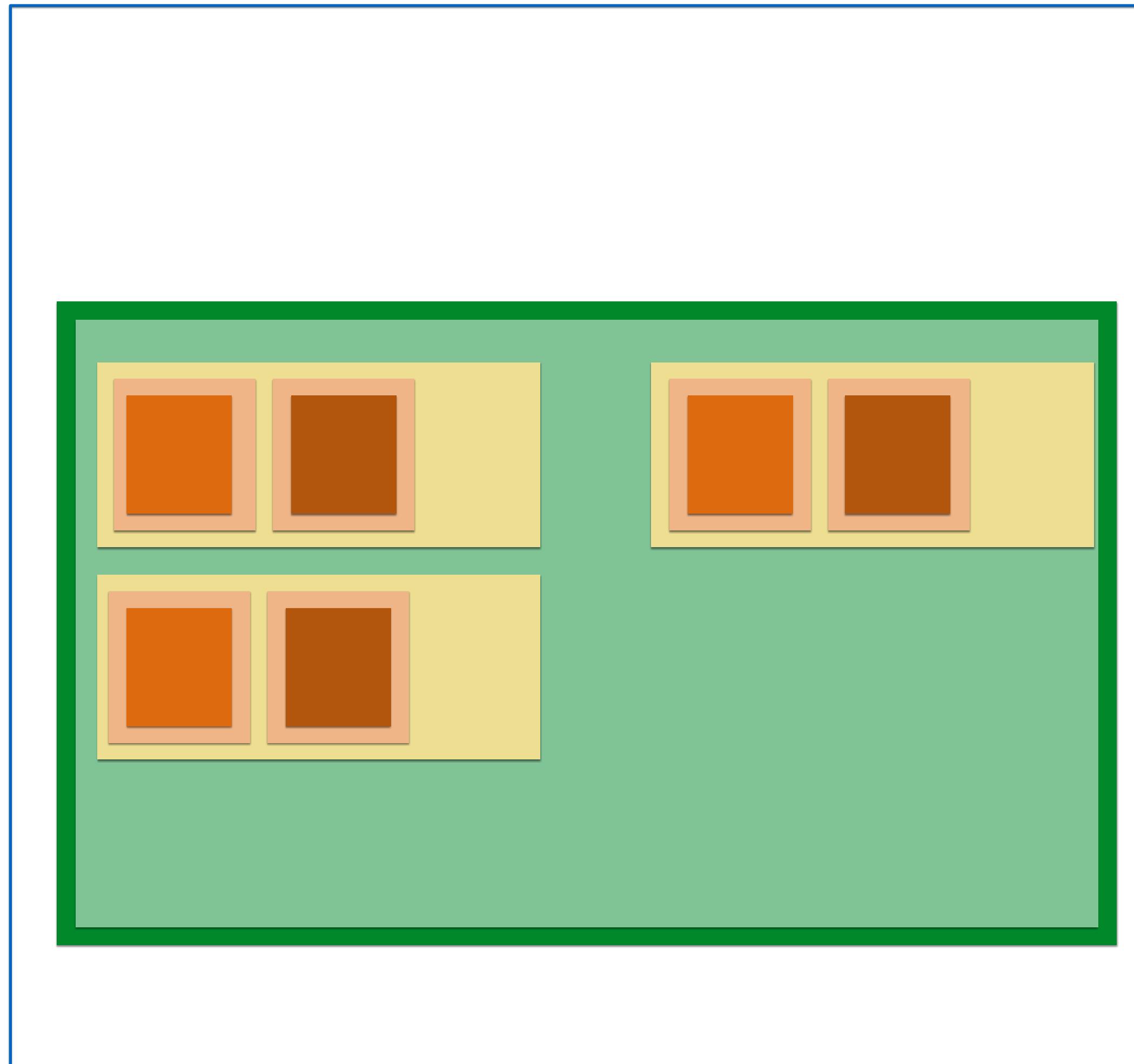
Replica Set

- Defines and manages how many instances of a pod should run
- ReplicaSet is tied to a specific definition of a Pod which is tied to specific image versions of the container
- Image versions in ReplicaSets can't be updated

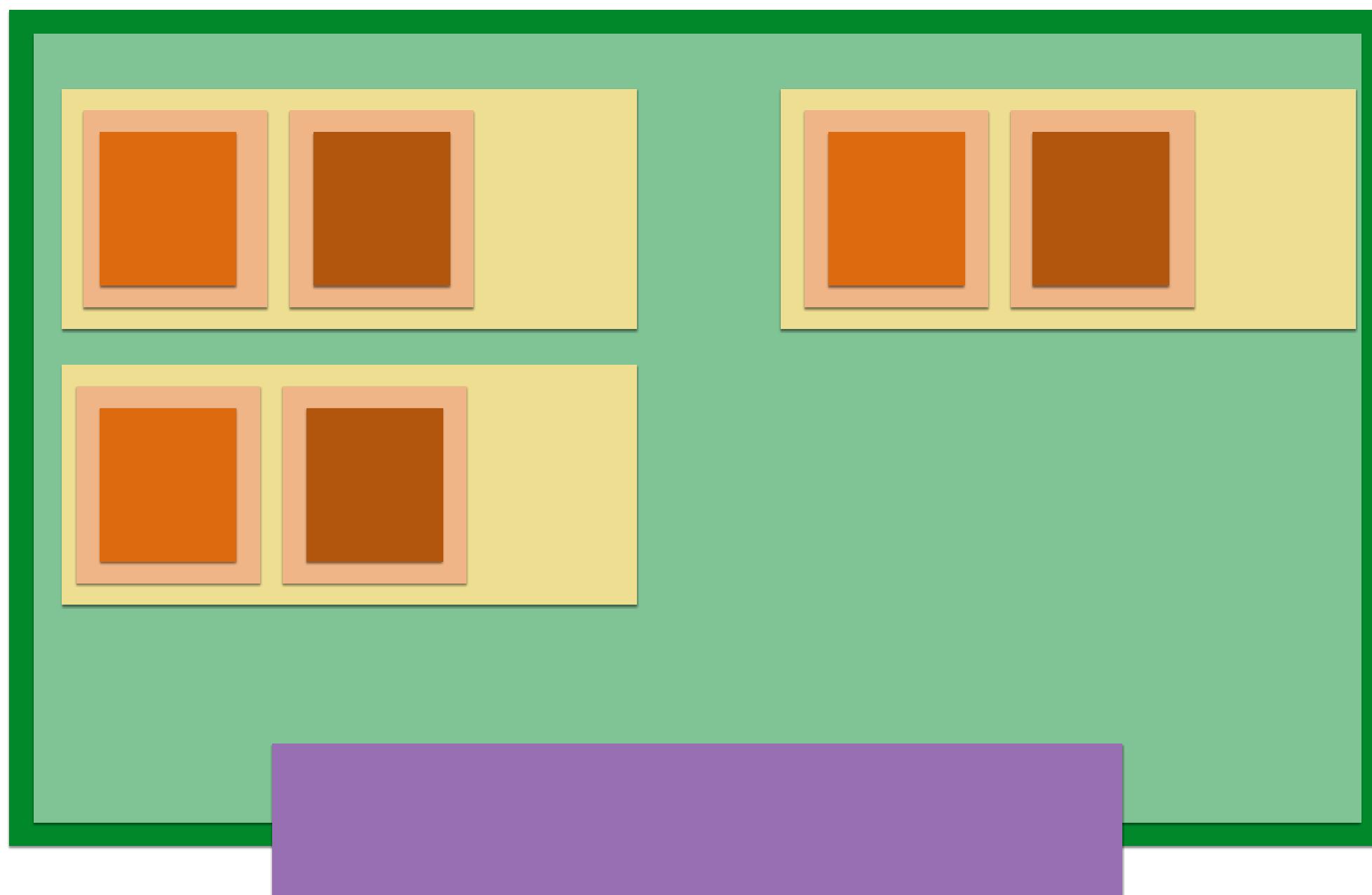


Deployment

- Manages updates and rollbacks of replica sets

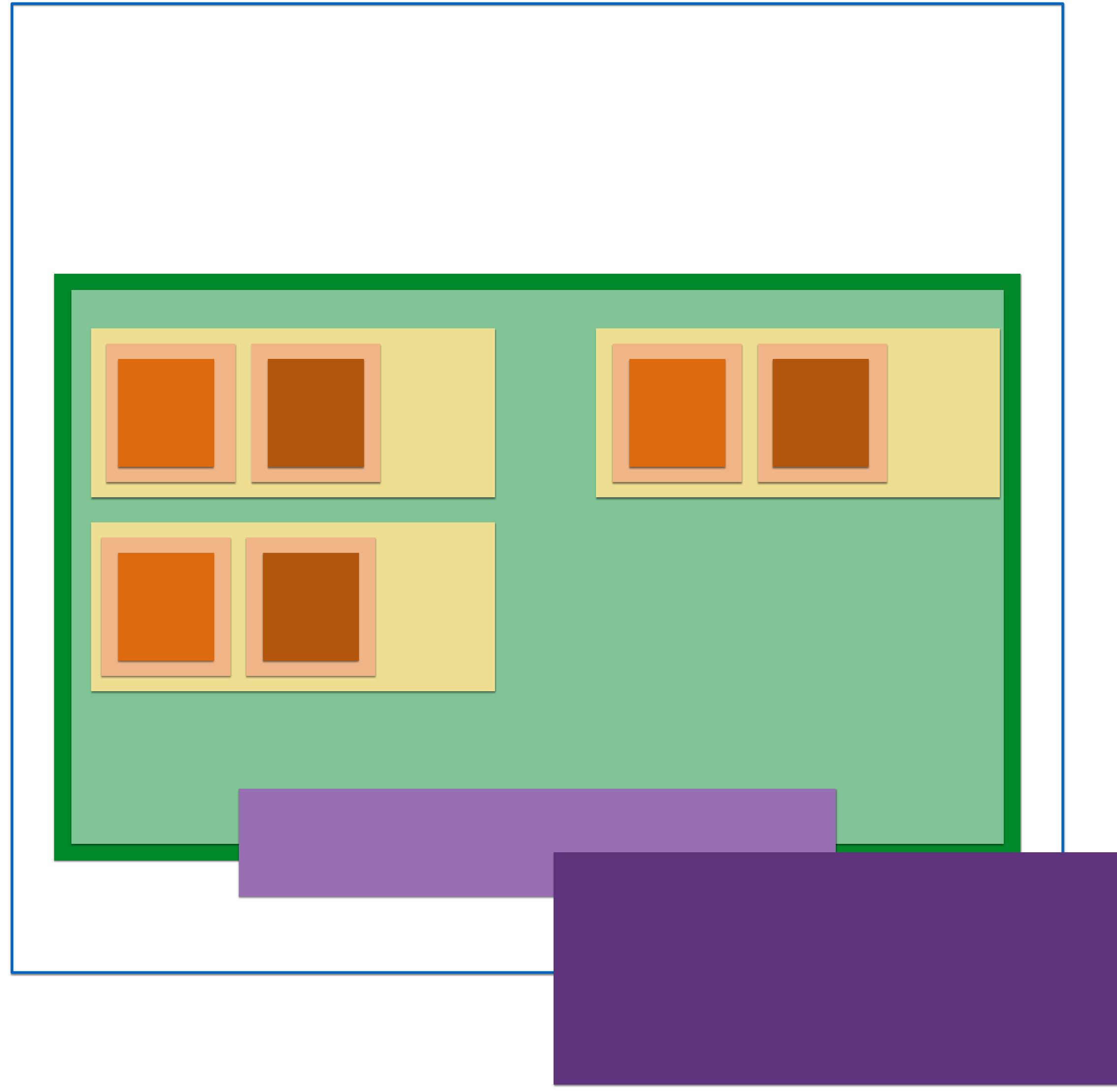


Service

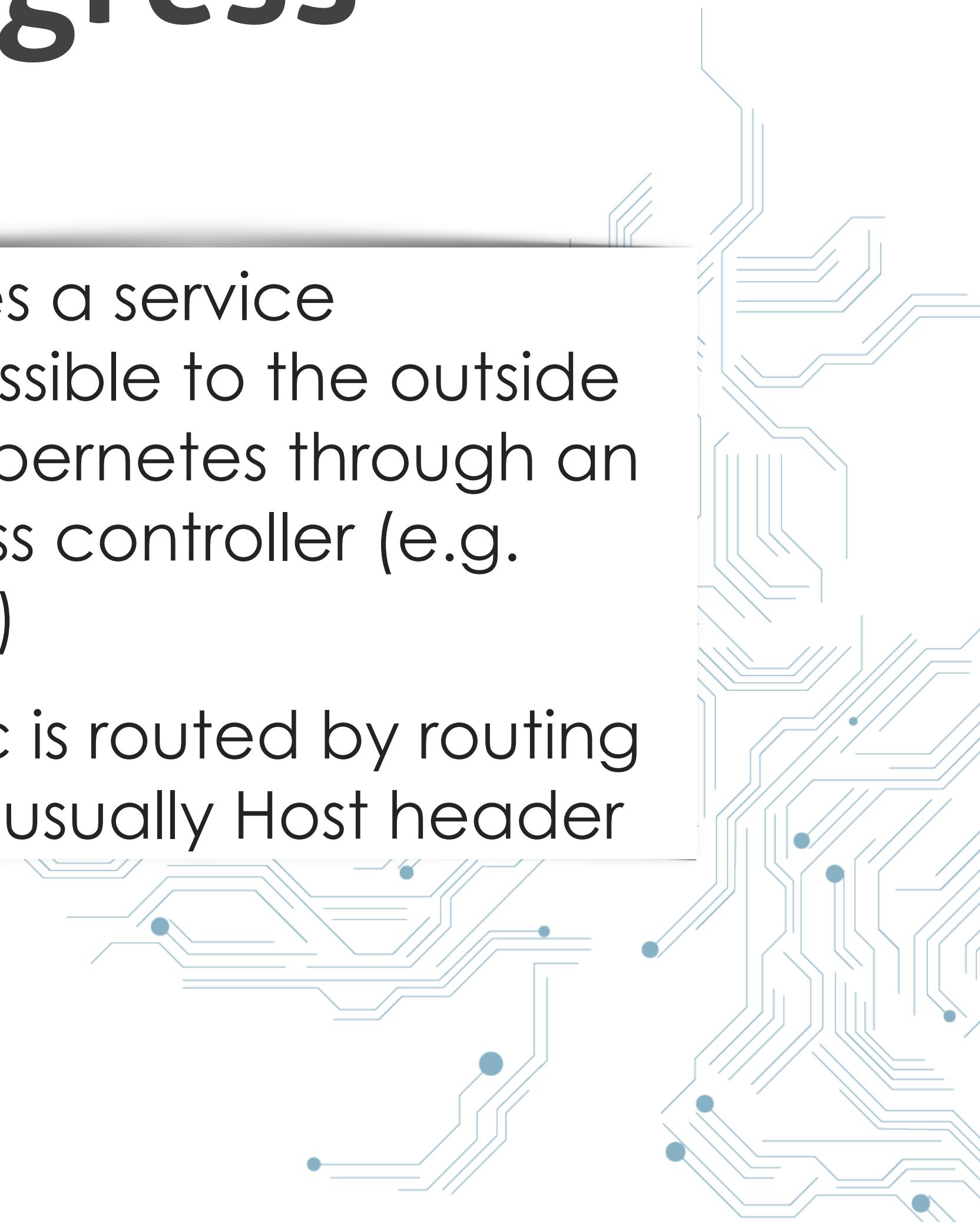


- Internal LoadBalancer
- Makes all pods matching a set of labels accessible through a stable, internal IP address
- You can attach external IP address through an cloud LoadBalancer

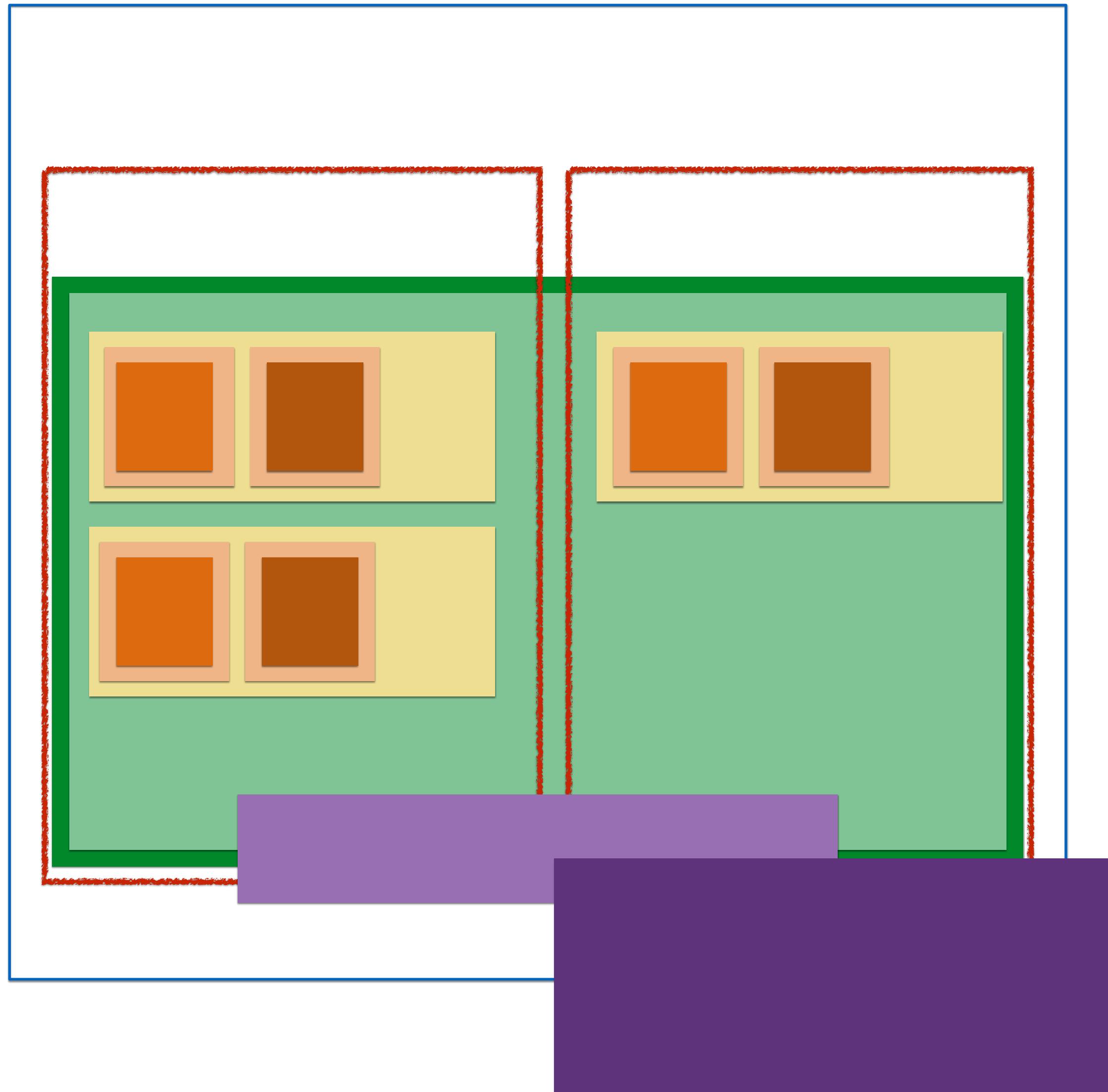
Ingress



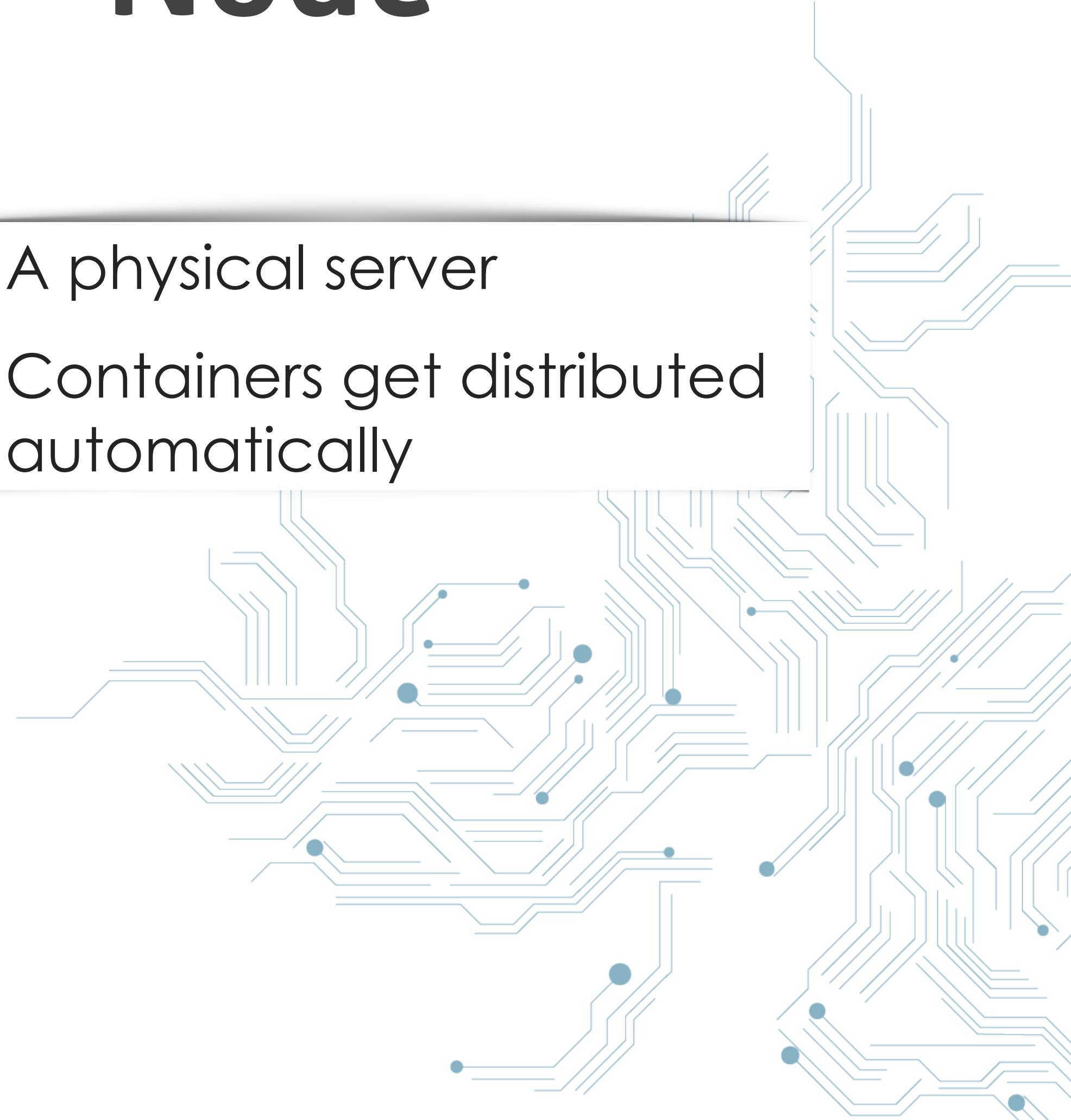
- Makes a service accessible to the outside of Kubernetes through an ingress controller (e.g. nginx)
- Traffic is routed by routing rules, usually Host header



Node

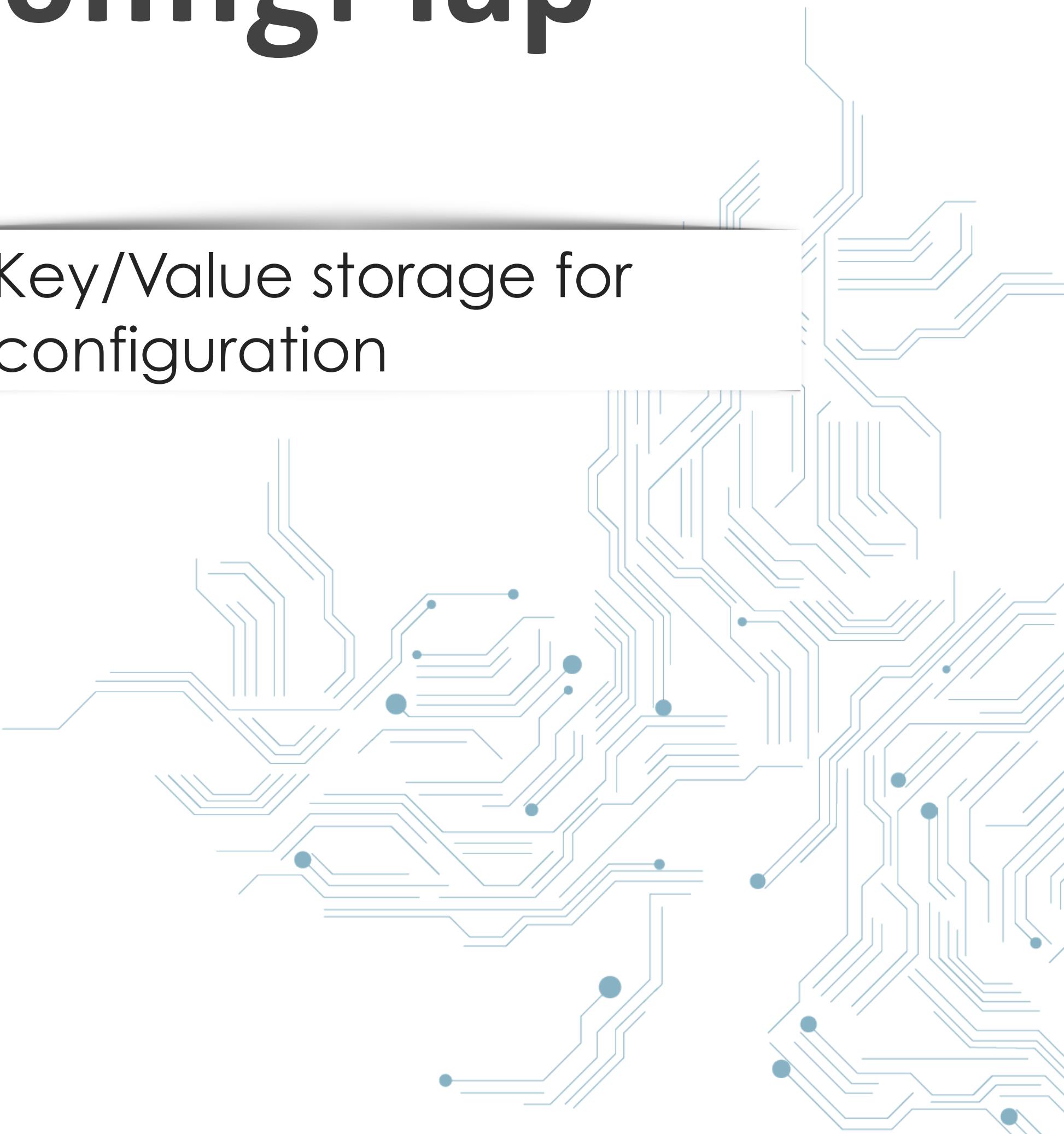
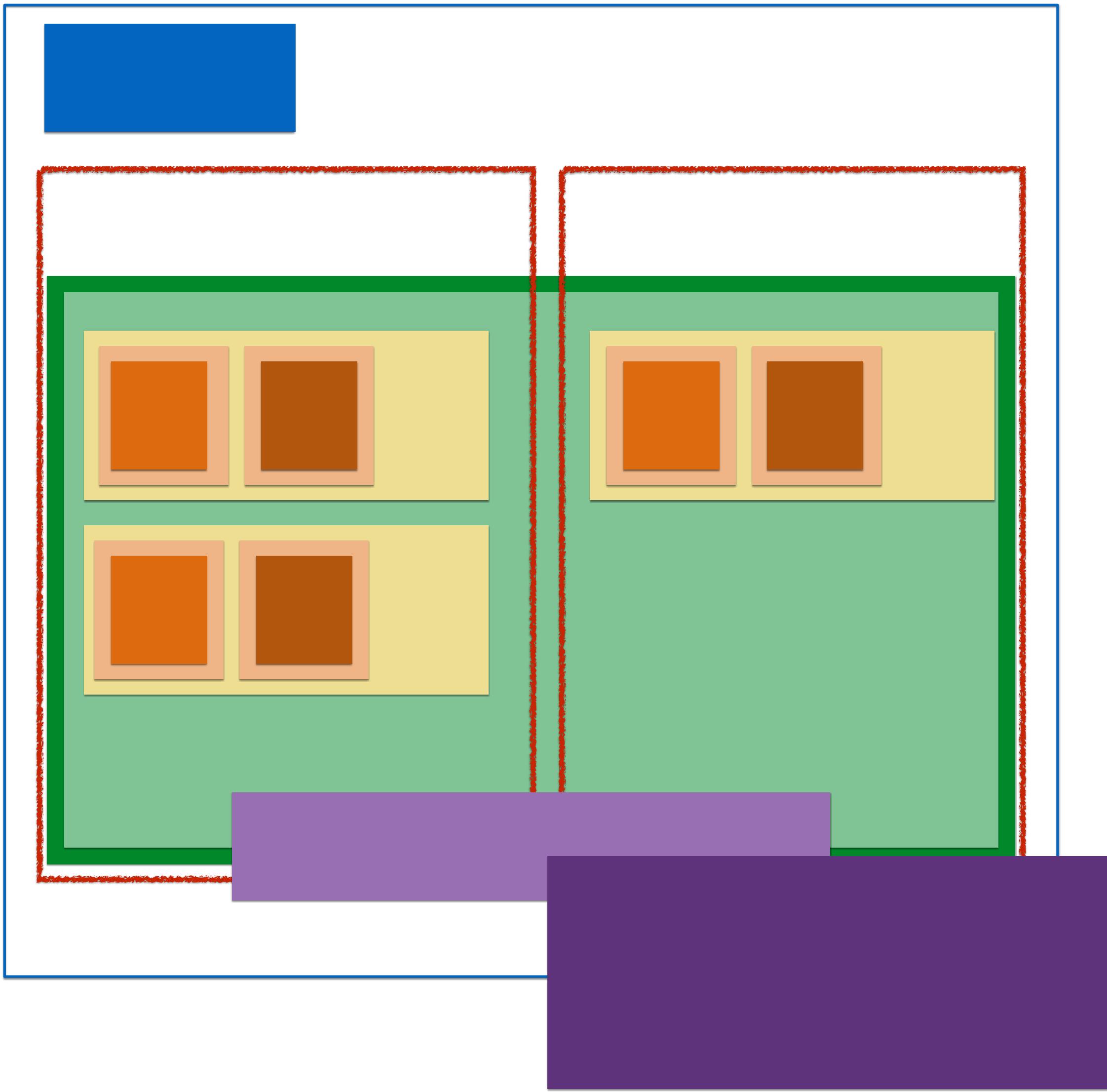


- A physical server
- Containers get distributed automatically



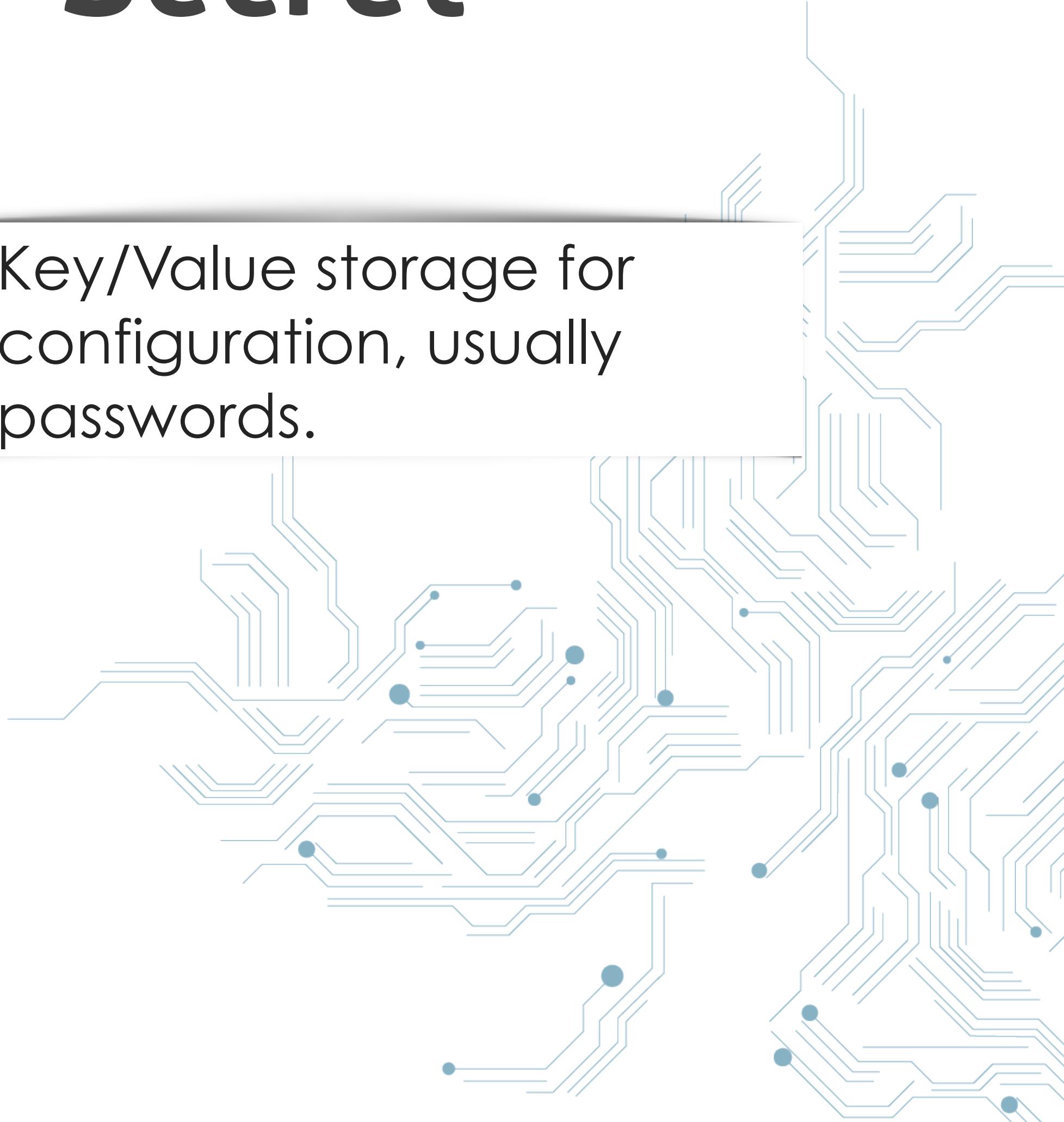
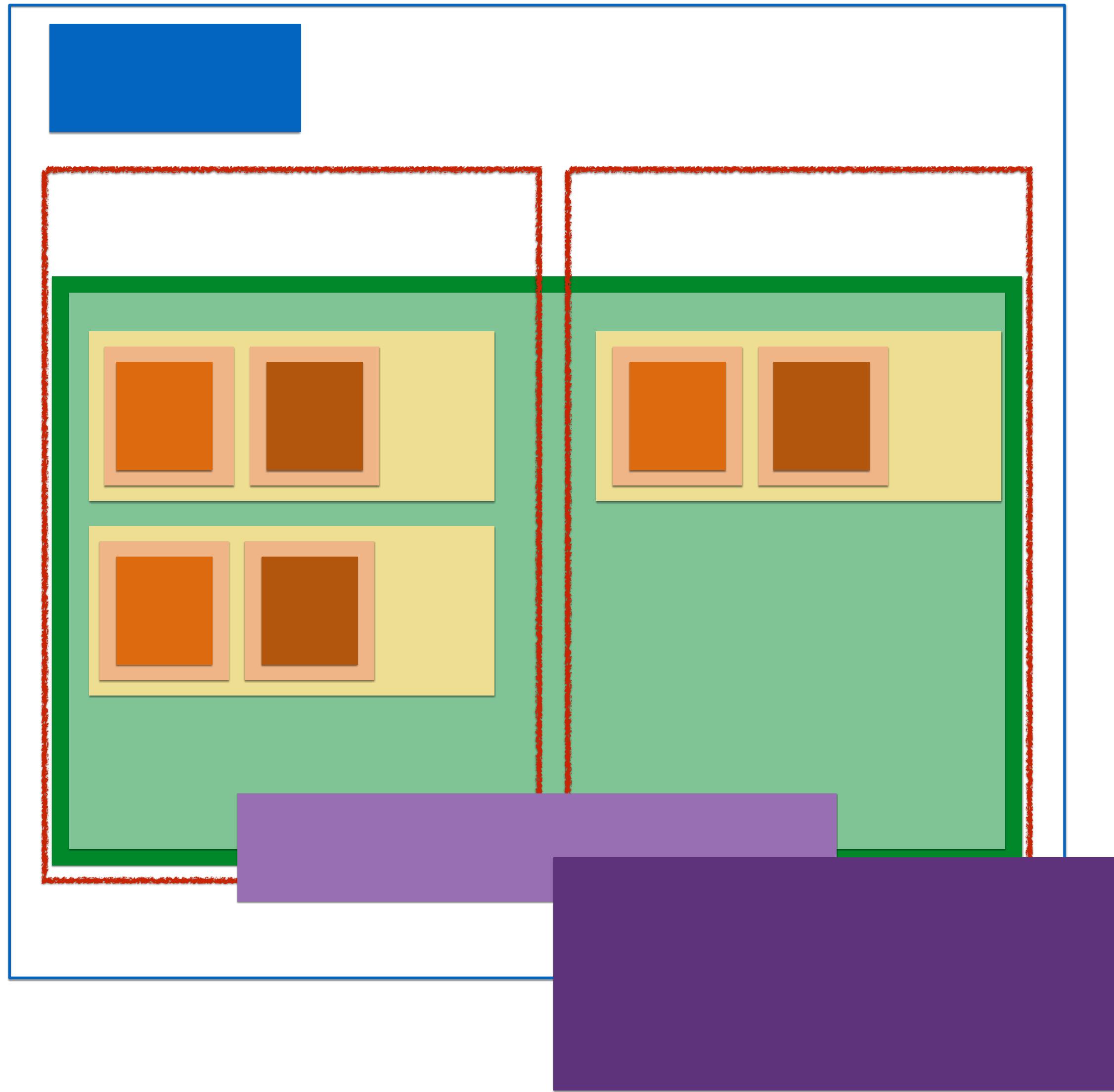
ConfigMap

- Key/Value storage for configuration

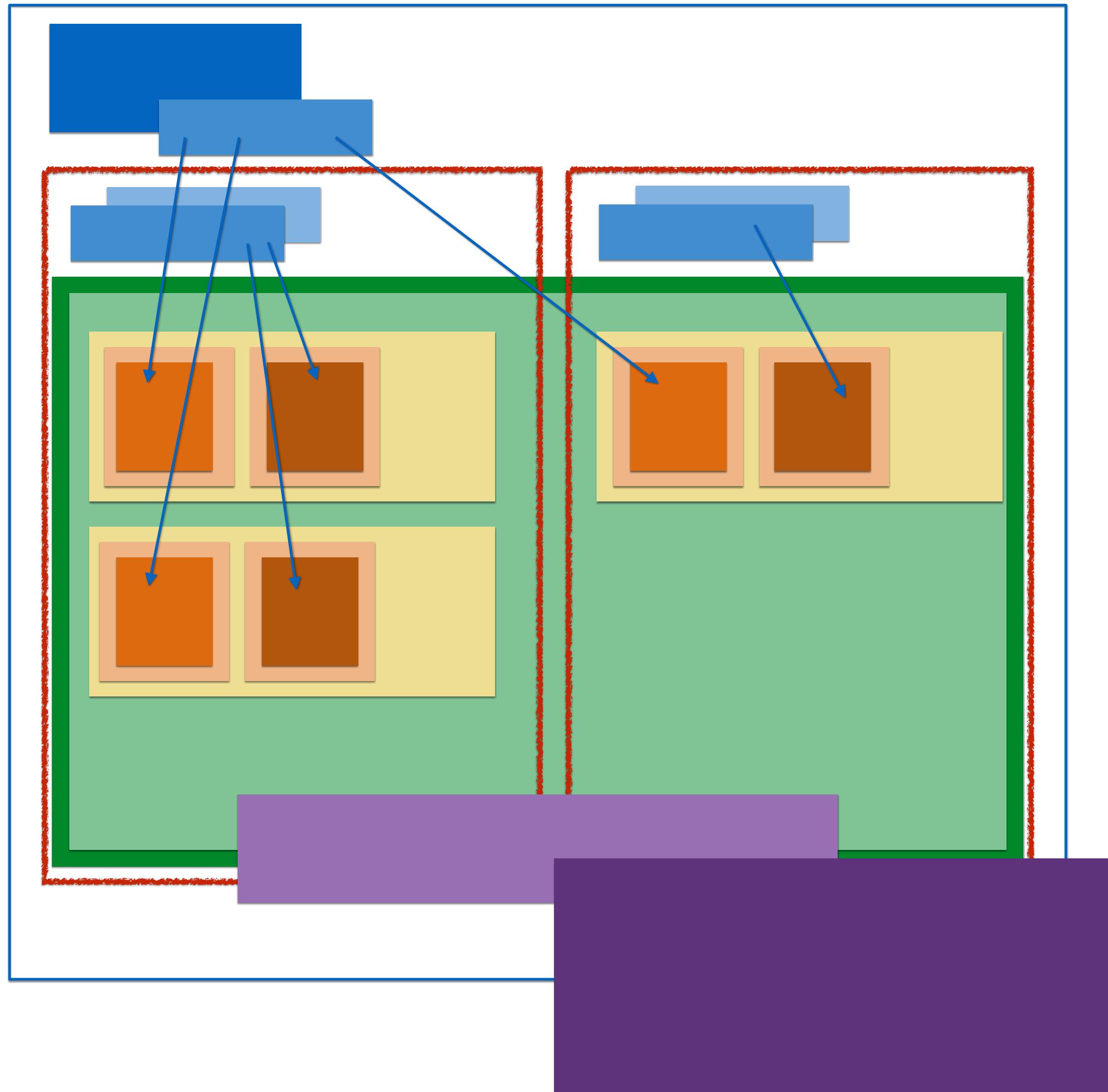


Secret

- Key/Value storage for configuration, usually passwords.



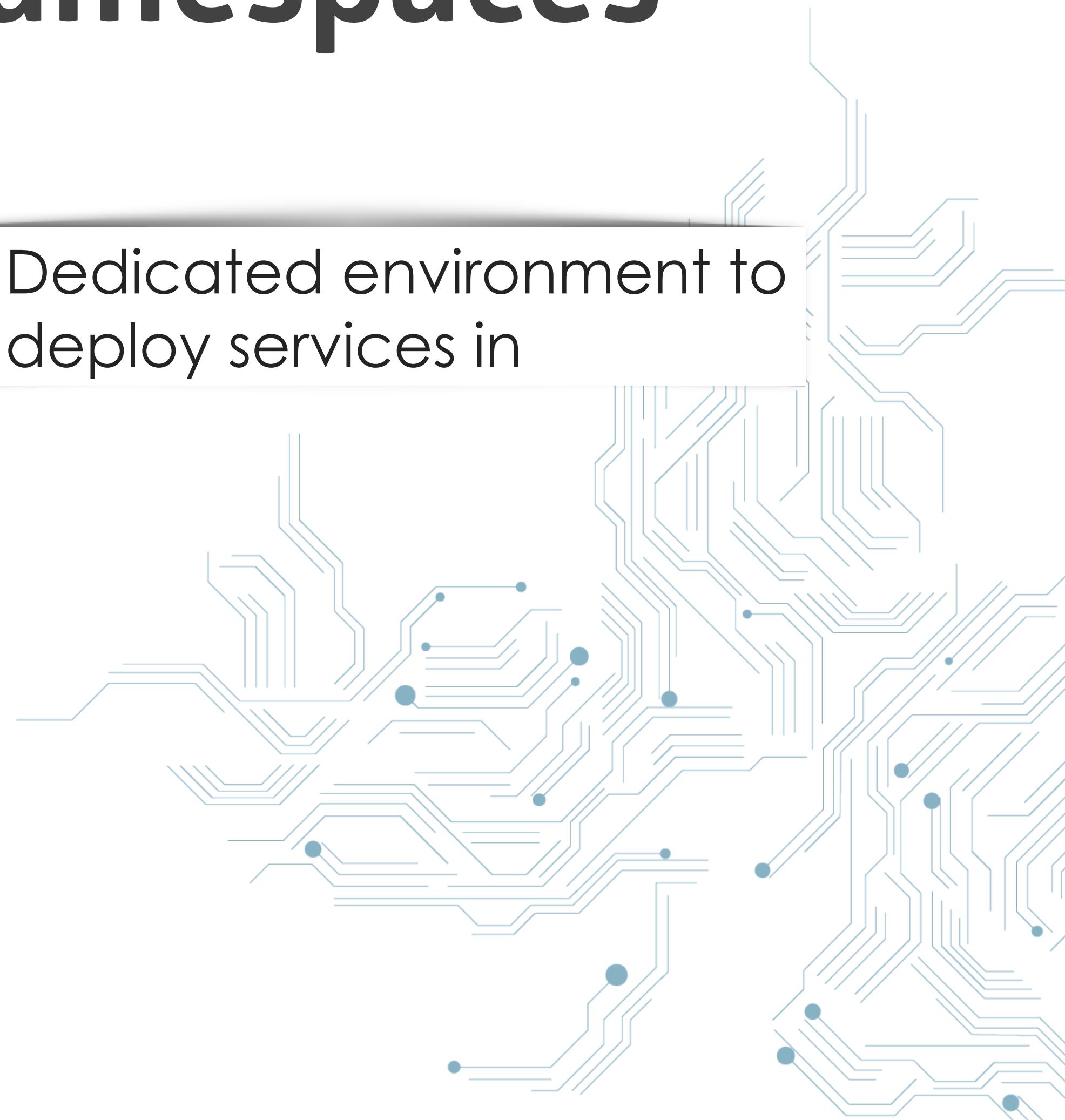
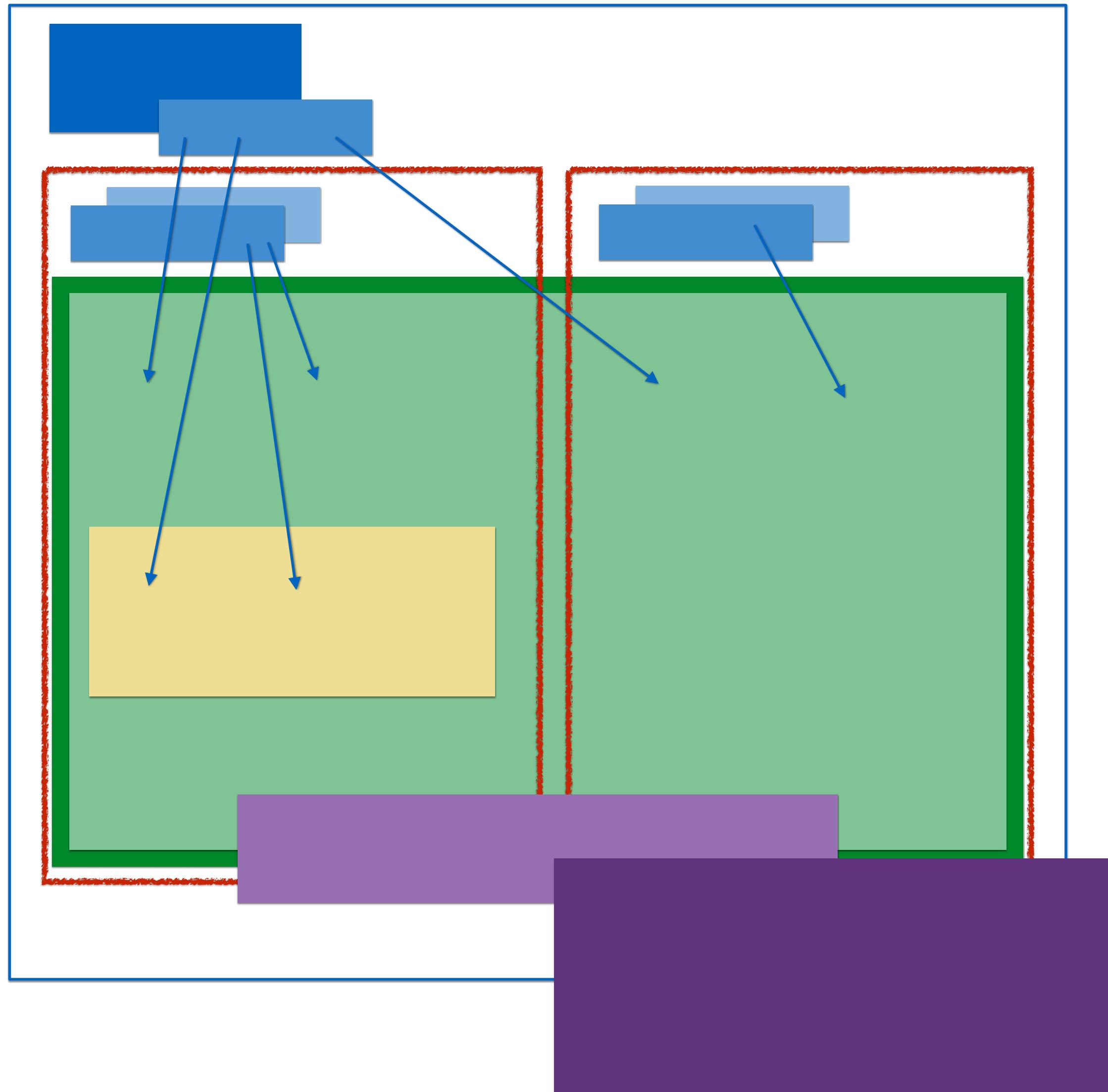
Volumes



- Volumes can be mounted into a container to access a ConfigMap, Secret, persistent volumes with network storage or a folder on the node

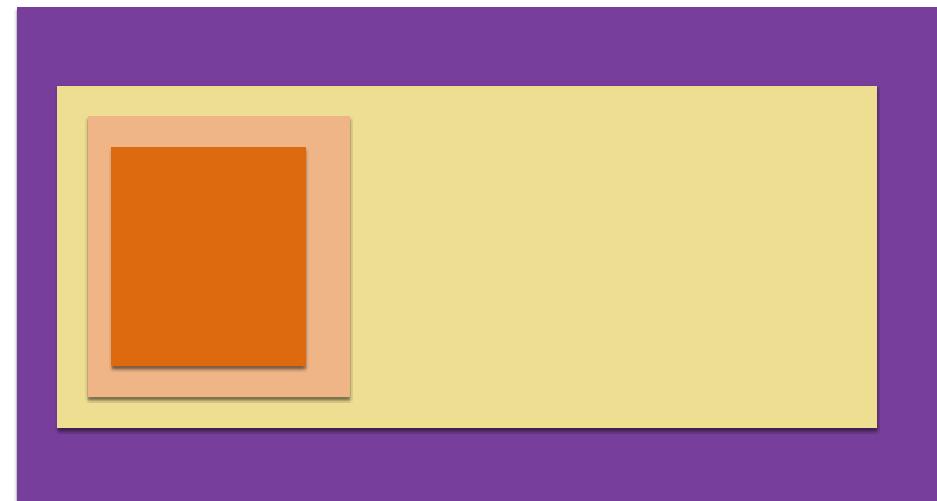
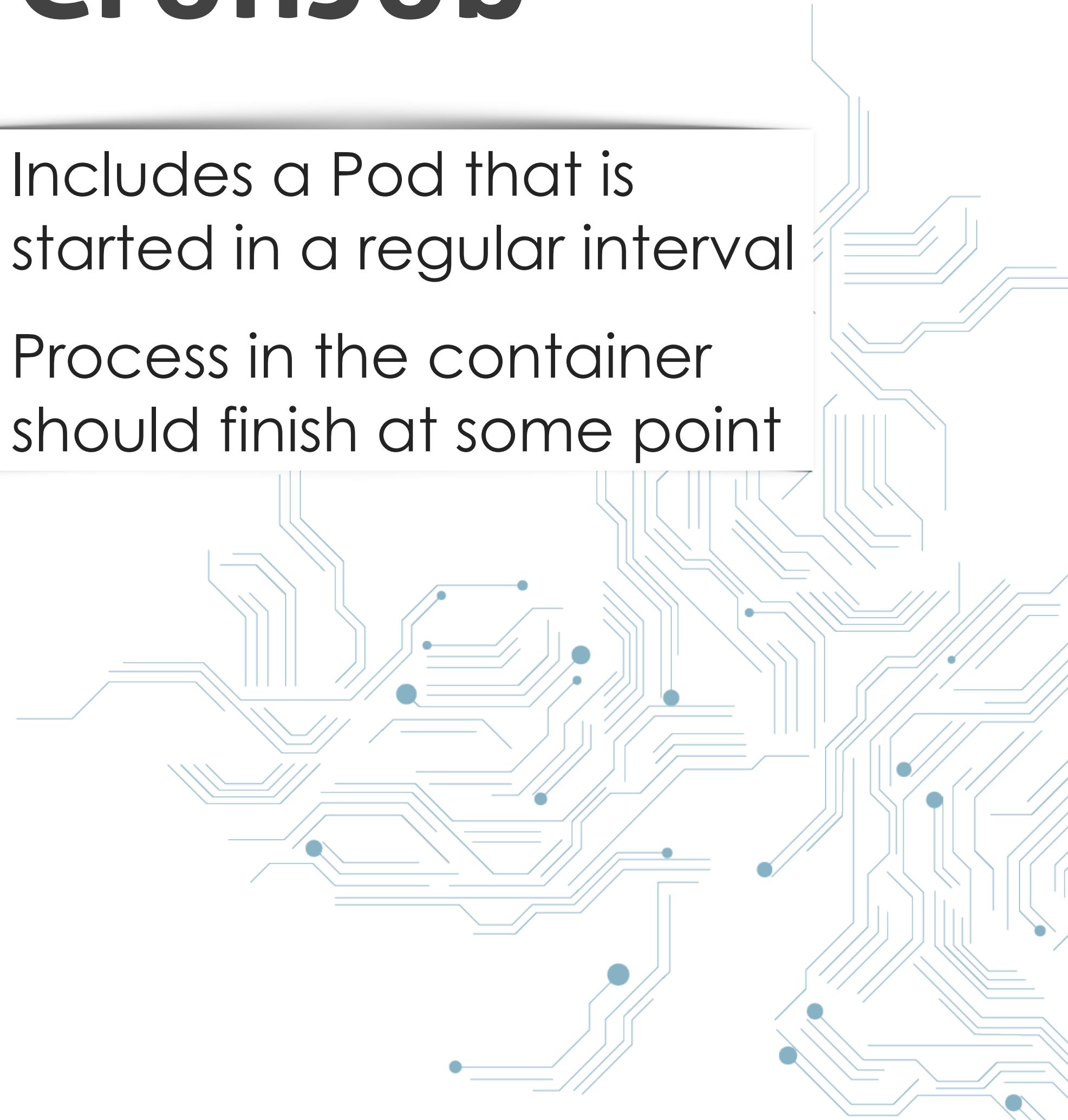
Namespaces

- Dedicated environment to deploy services in



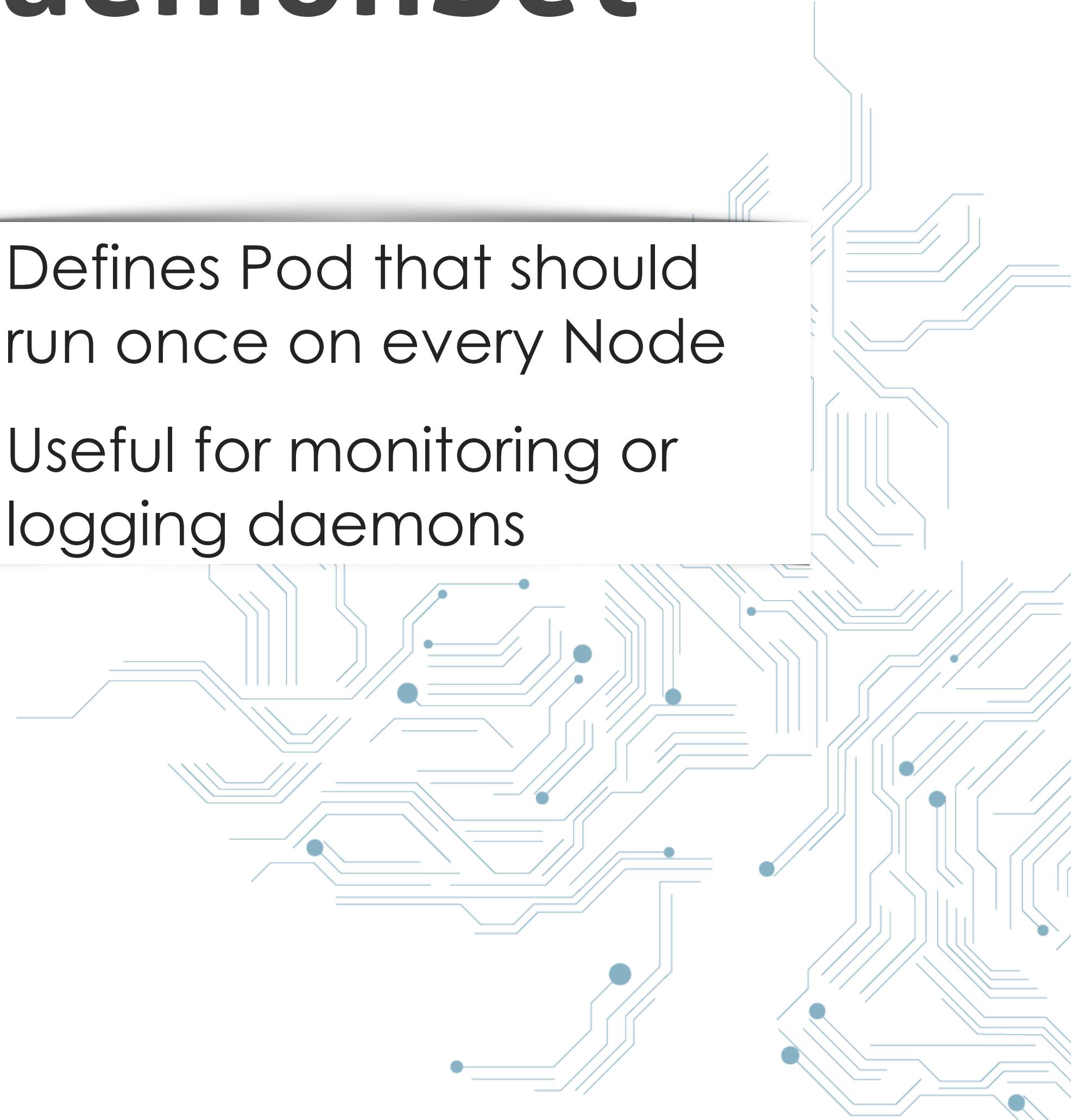
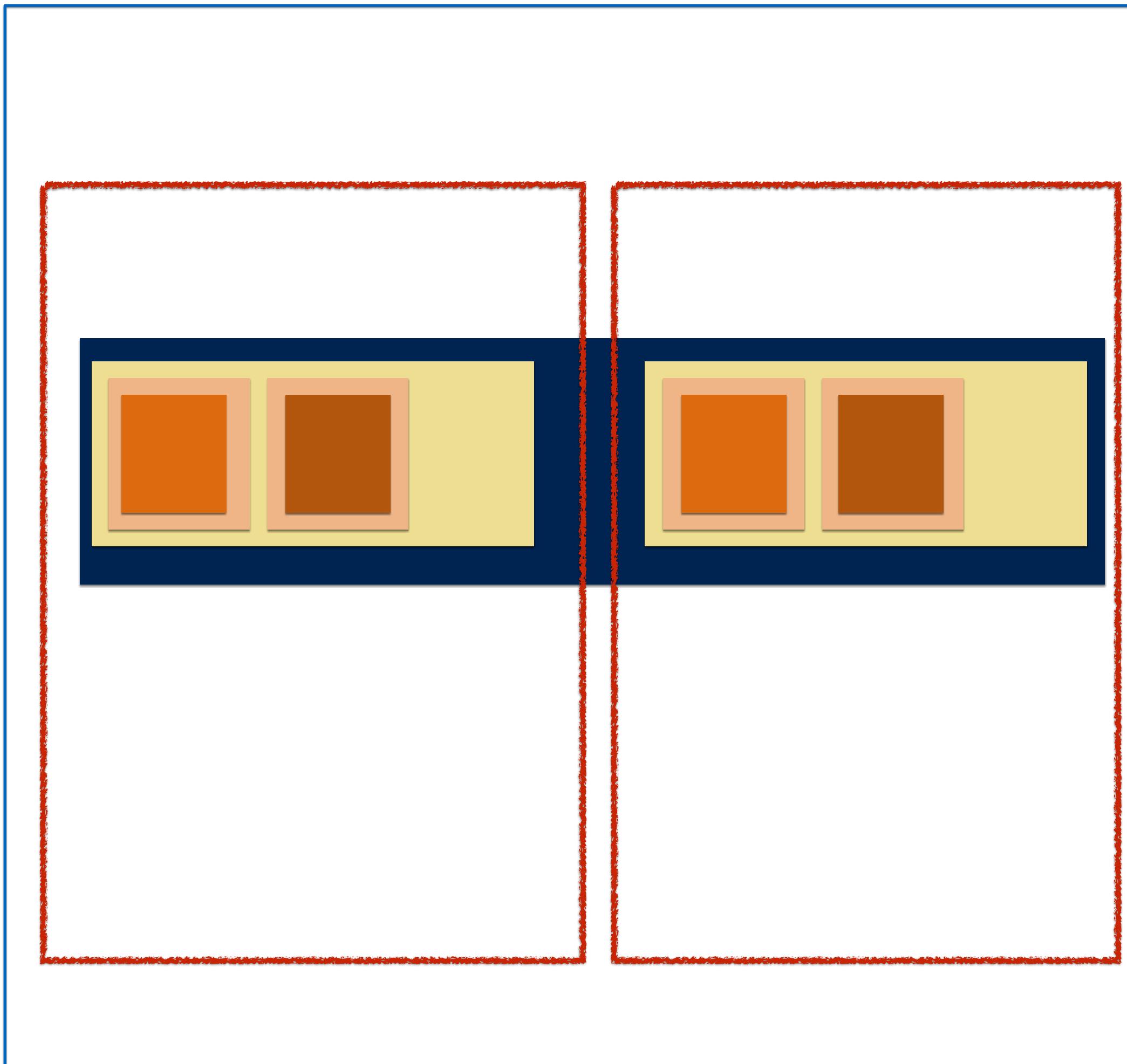
CronJob

- Includes a Pod that is started in a regular interval
- Process in the container should finish at some point

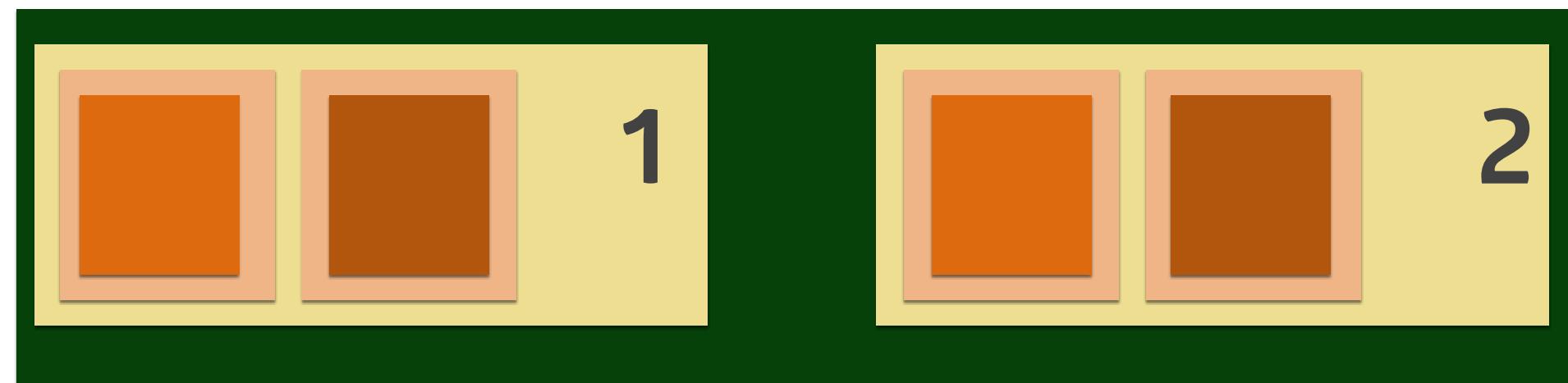


DaemonSet

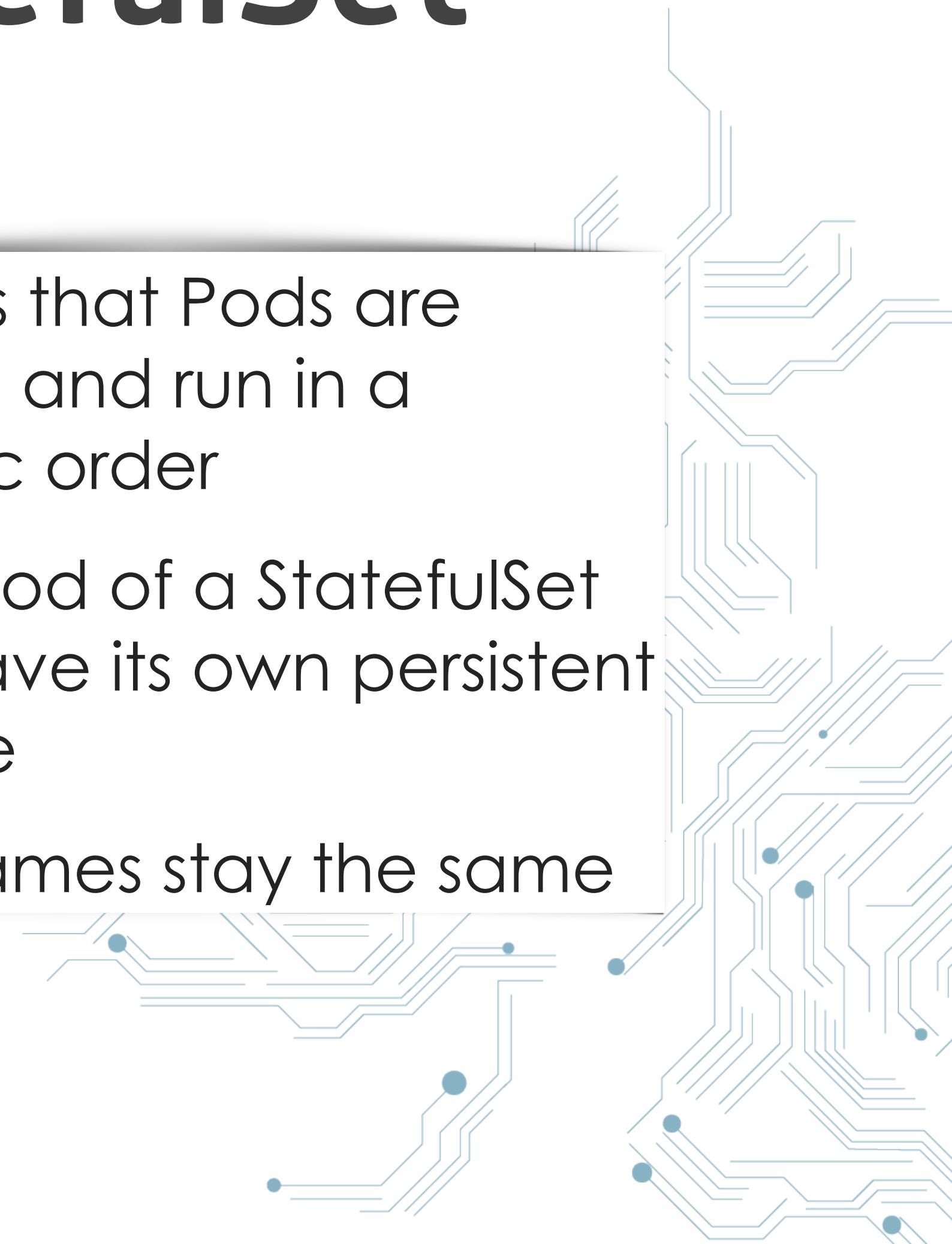
- Defines Pod that should run once on every Node
- Useful for monitoring or logging daemons



StatefulSet



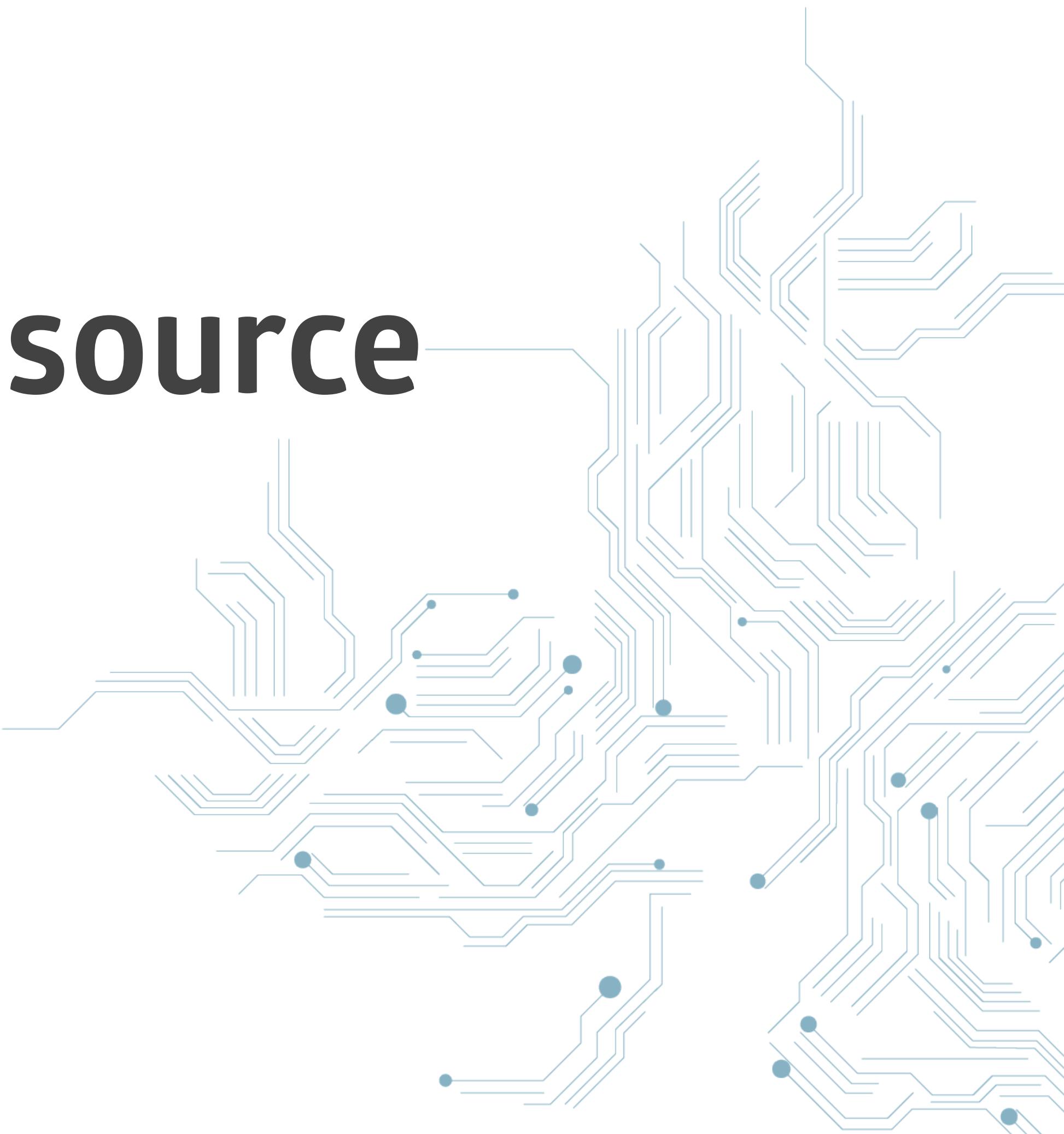
- Ensures that Pods are started and run in a specific order
- Each Pod of a StatefulSet can have its own persistent volume
- Pod names stay the same



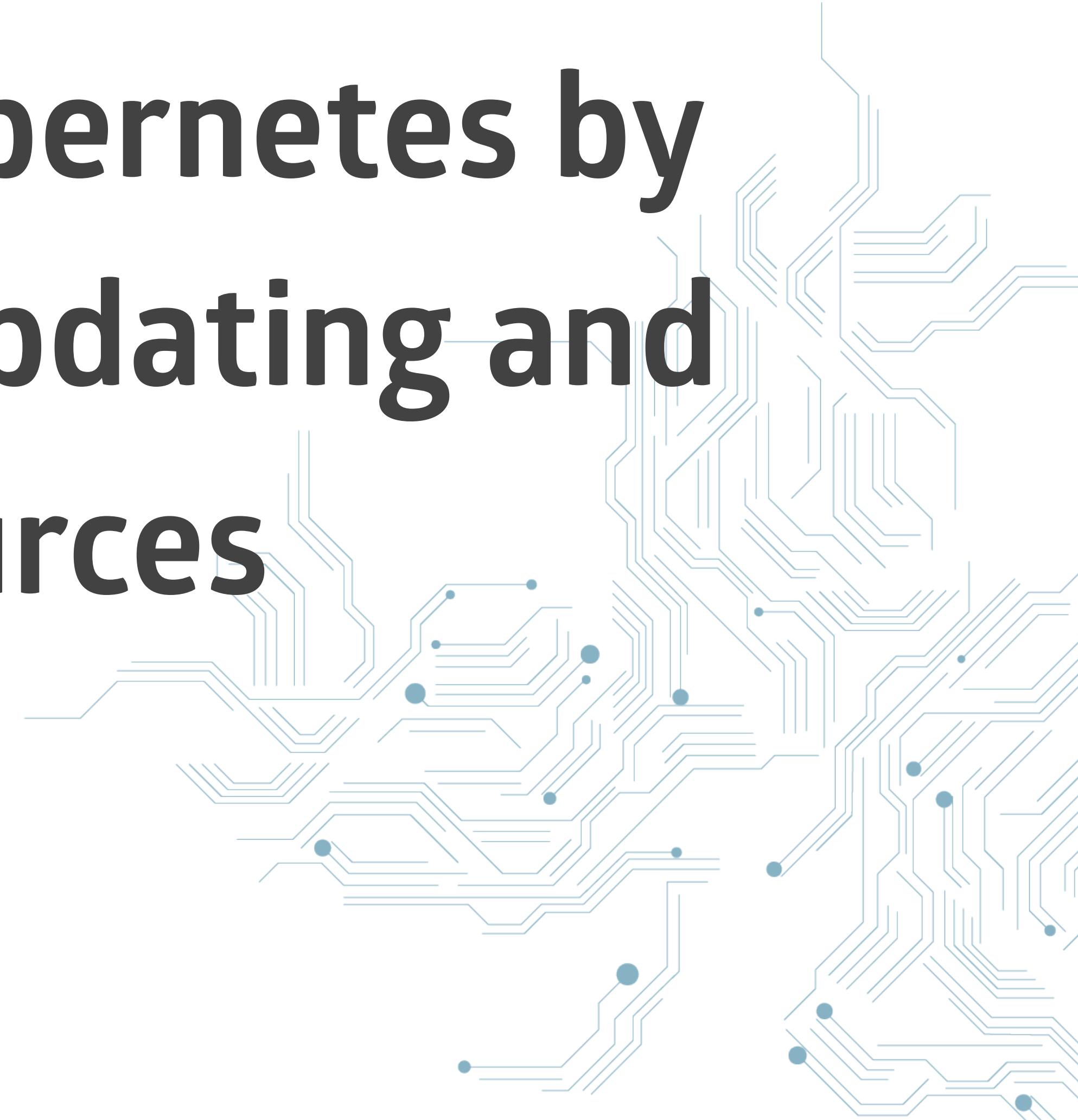
...



Everything is a resource



You interact with Kubernetes by
creating, receiving, updating and
deleting resources



**Kubernetes has controllers to listen
on these interactions and get the
cluster in the desired state.**



**The Kubernetes API can be extended
with additional Resources and
Controllers**



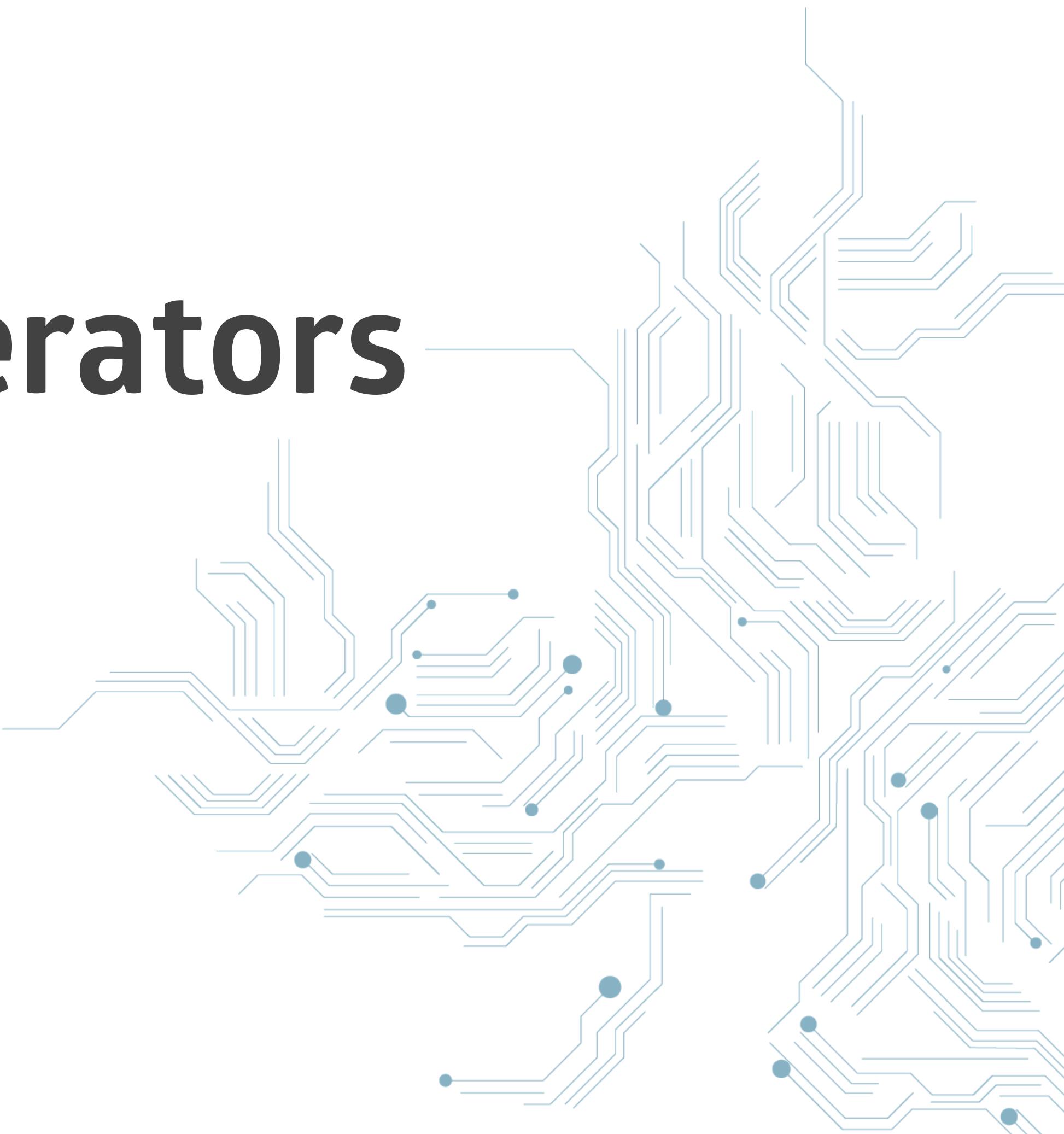
CustomResourceDefinitions



Certificate, Backup, Restore, MySQLCluster, Function, ...



Controllers / Operators



```
kind: Deployment
apiVersion: extensions/v1beta1
metadata:
  name: symfony-demo
spec:
  template:
    spec:
      containers:
        - name: symfony-demo
          image: symfony-demo:1.1.0
          ports:
            - containerPort: 80
```



```
$ kubectl apply -f deployment.yaml
```



```
$ kubectl get deployments
```

NAME	DESIRED	CURRENT	UP-TO-DATE	AVAILABLE	AGE
symfony-demo	1	1	1	21h	

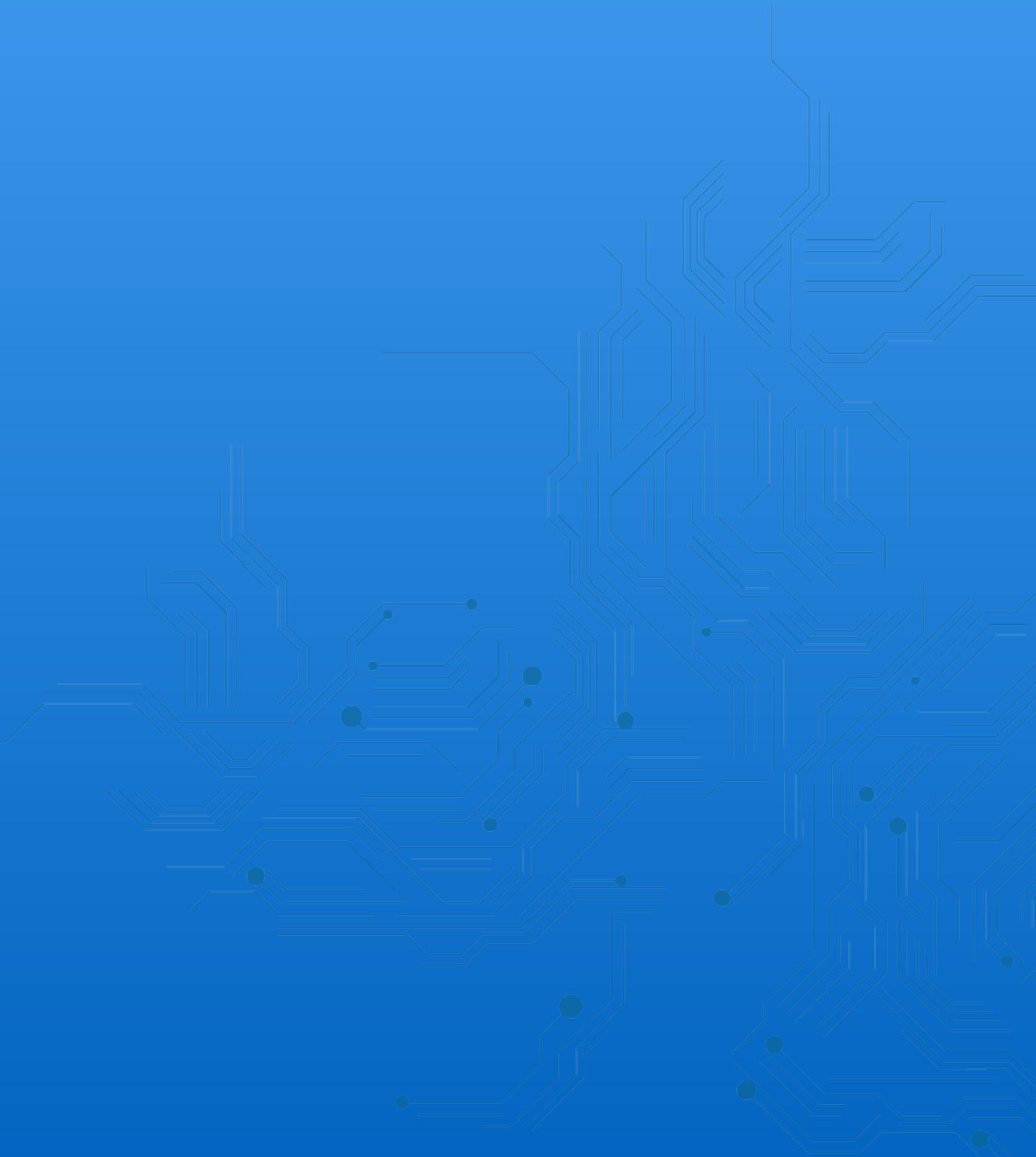
```
$ kubectl get deployment symfony-demo -o yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    ...
spec:
  ...
template:
  ...
    spec:
      containers:
        - name: symfony-demo
```



```
$ kubectl delete deployment symfony-demo
```



Tooling



kubectl

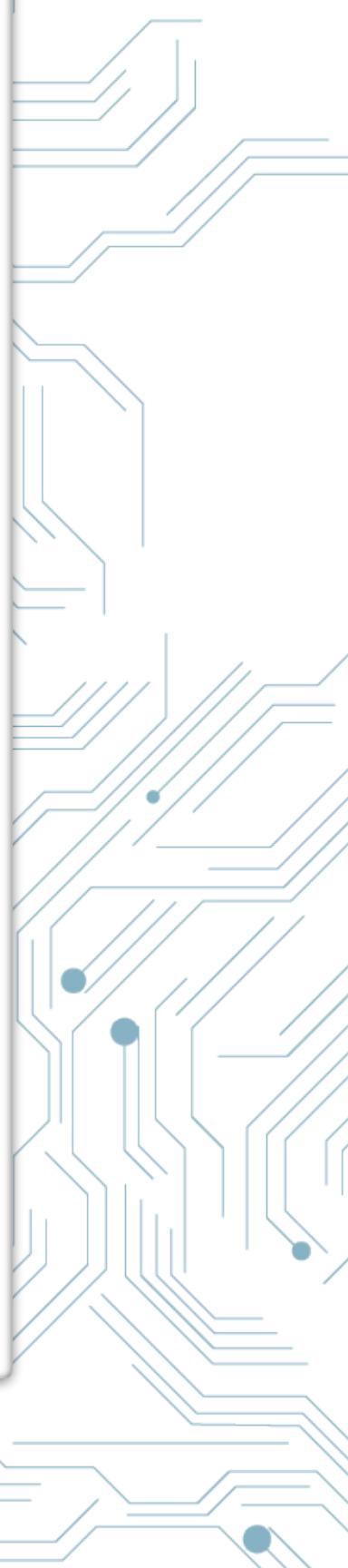


REST API

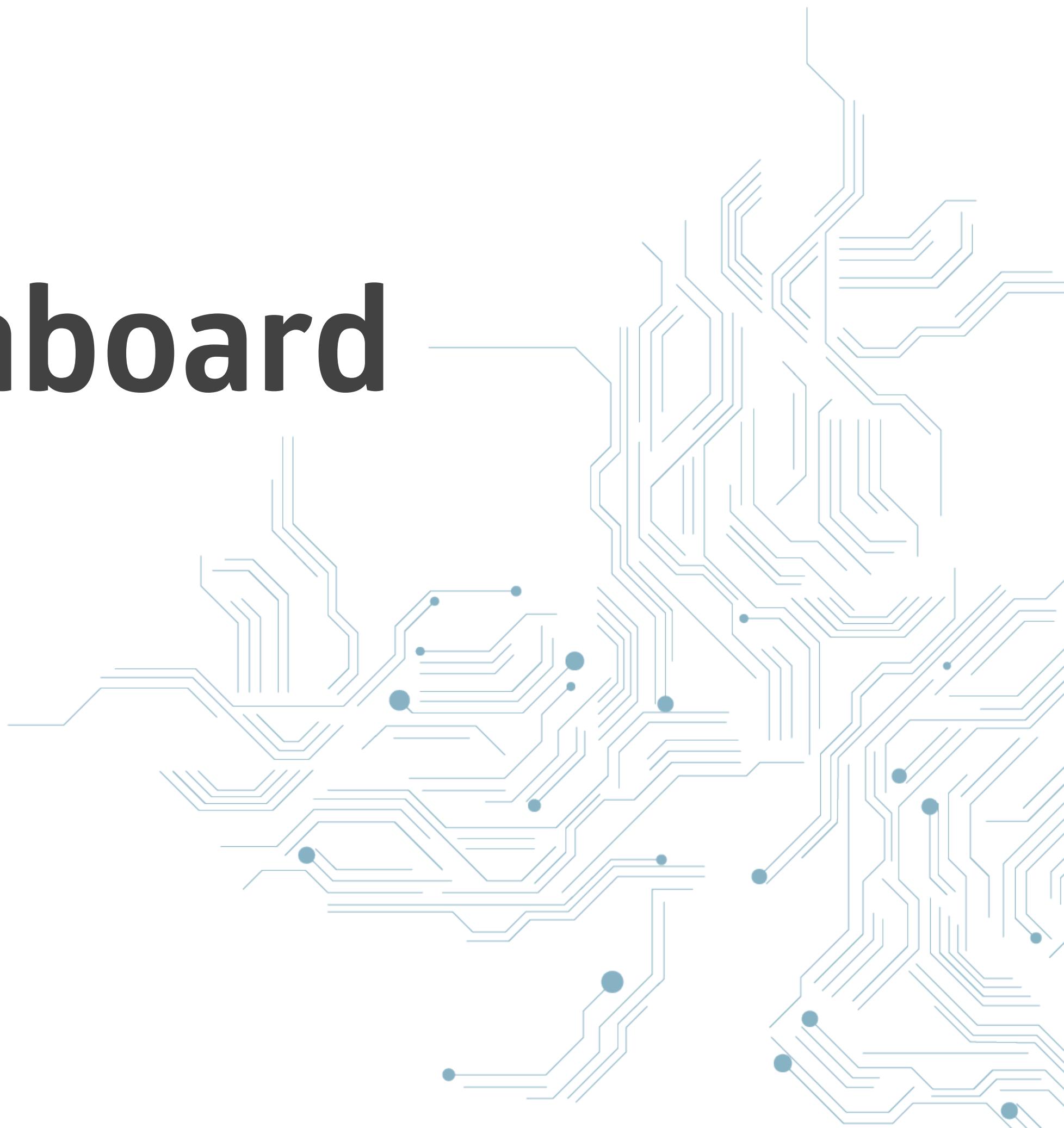


```
$ kubectl proxy --port=8080
$ curl http://localhost:8080/api/v1/namespaces/default/pods
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/namespaces/default/pods",
    "resourceVersion": "336834"
  },
  "items": [
    {
      "metadata": {
        "name": "kubernetes-dashboard-5b5bf59977-t9xb9",

```



kubernetes-dashboard



☰ Overview

Cluster

- Namespaces
- Nodes
- Persistent Volumes
- Roles
- Storage Classes

Namespace

- default

Overview

Workloads

- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

Discovery and Load Balancing

- Ingresses
- Services

Config and Storage

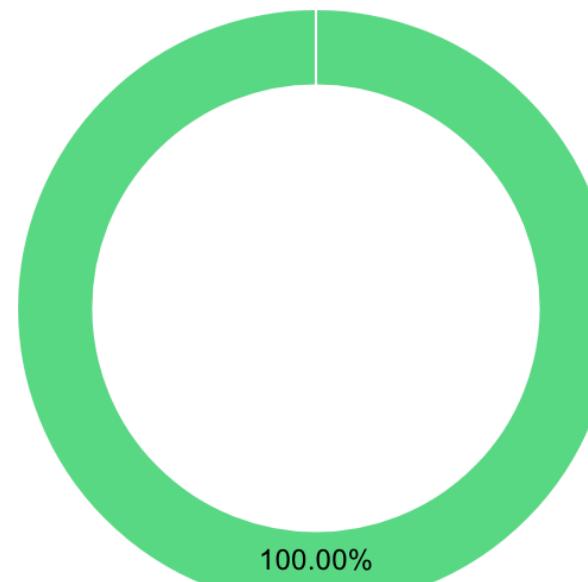
- Config Maps
- Persistent Volume Claims

Resource Status

Pods

- Running
- Pending
- Failed

- 6
- 0
- 0

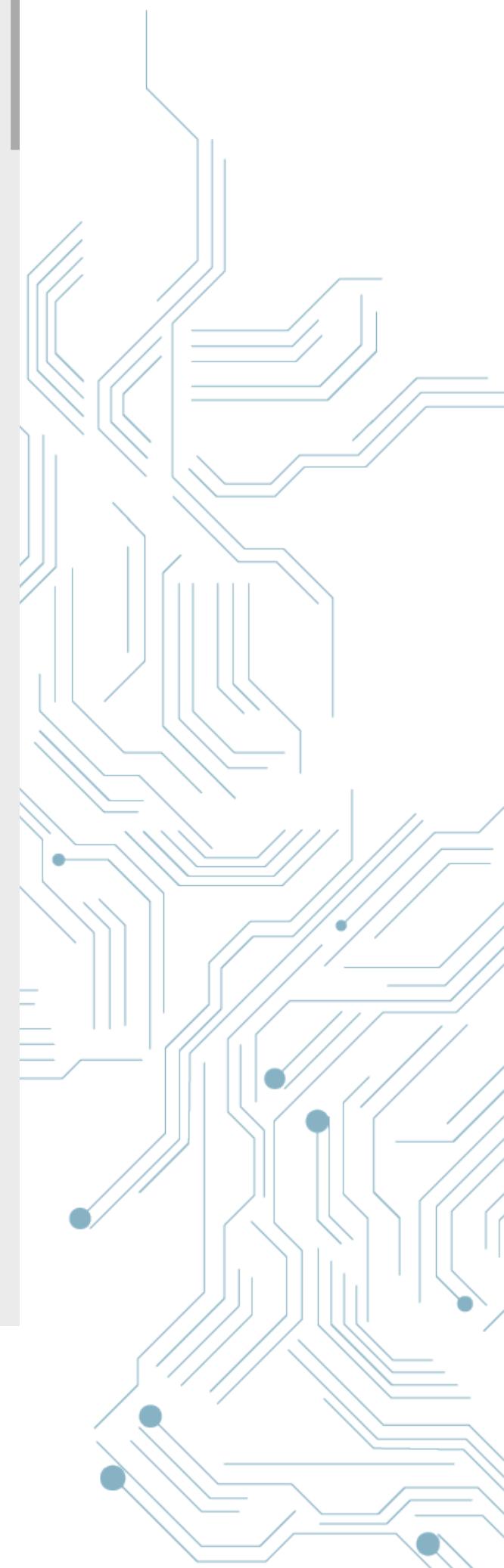


Pods

Name	Node	Status	Restarts	Age	⋮
✓ symfony-demo-5b75f5fc6-jg8n4	docker-for-desktop	Running	23	8 days	⋮
✓ symfony-demo-5b75f5fc6-c7wr9	docker-for-desktop	Running	0	8 days	⋮
✓ silly-grizzly-mysql-556c9b5bcb-5jdrt	docker-for-desktop	Running	1	8 days	⋮
✓ kindly-cardinal-nginx-ingress-controller-5549f5597c-97kcw	docker-for-desktop	Running	3	9 days	⋮
✓ kindly-cardinal-nginx-ingress-default-backend-564d9d9477-tmnrr	docker-for-desktop	Running	4	9 days	⋮
✓ idle-ferrit-kubernetes-dashboard-5b5bf59977-t9xb9	docker-for-desktop	Running	3	9 days	⋮

Deployments

Name	Labels	Pods	Age	Images
------	--------	------	-----	--------



Helm

The package manager for

Kubernetes



```
$ helm install stable/wordpress
```

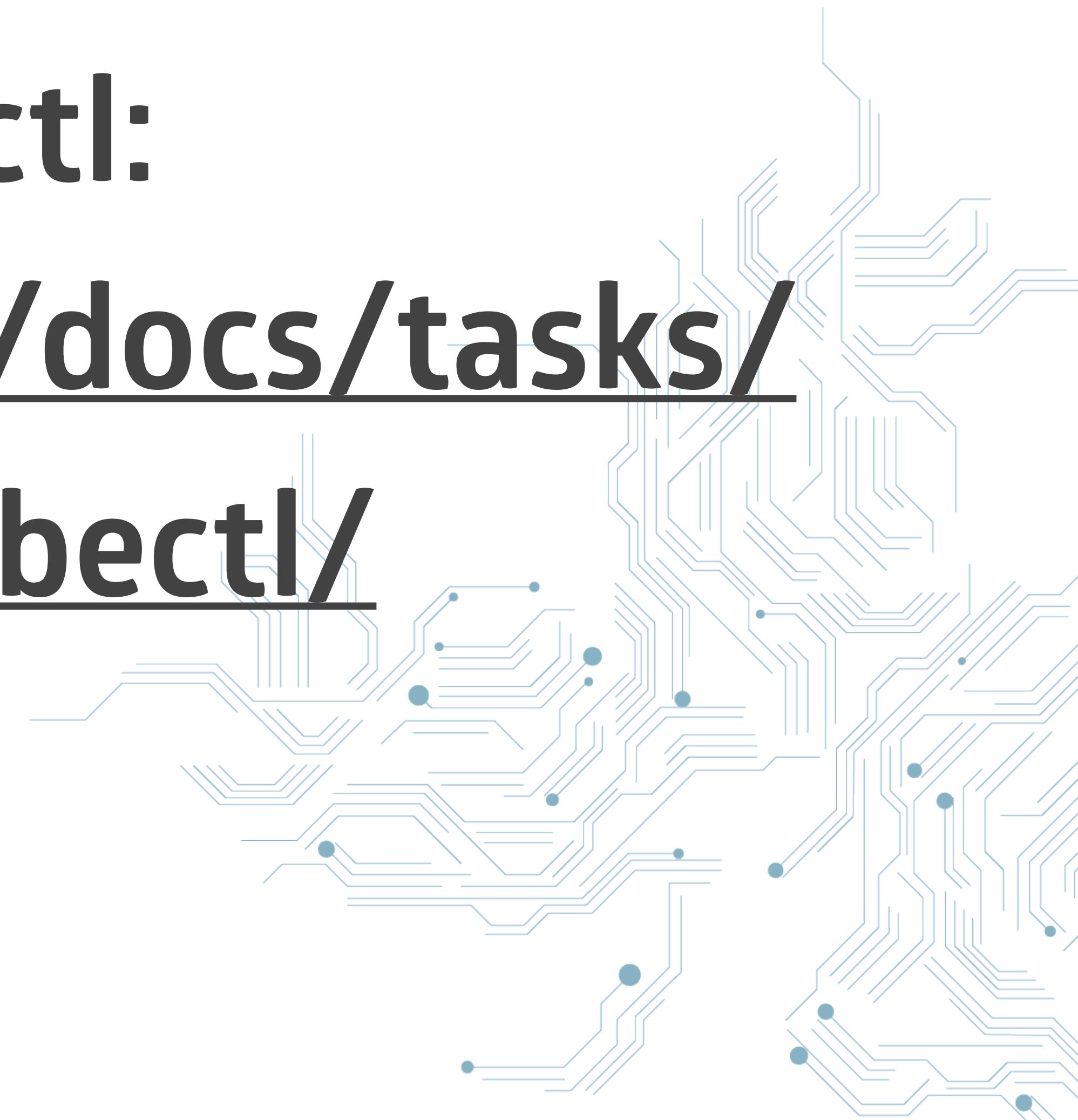


hands-on



Install kubectl:

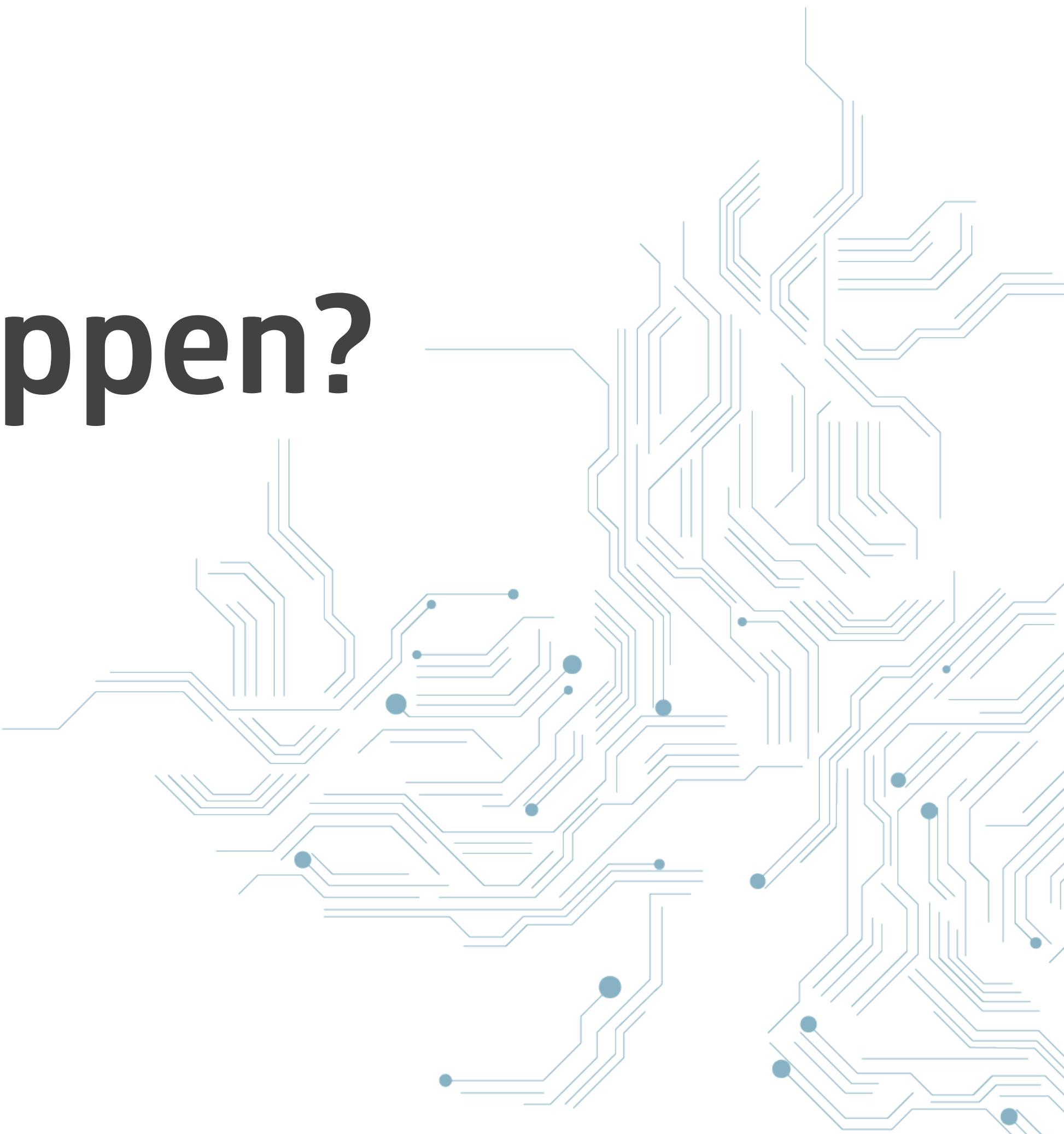
<https://kubernetes.io/docs/tasks/tools/install-kubectl/>

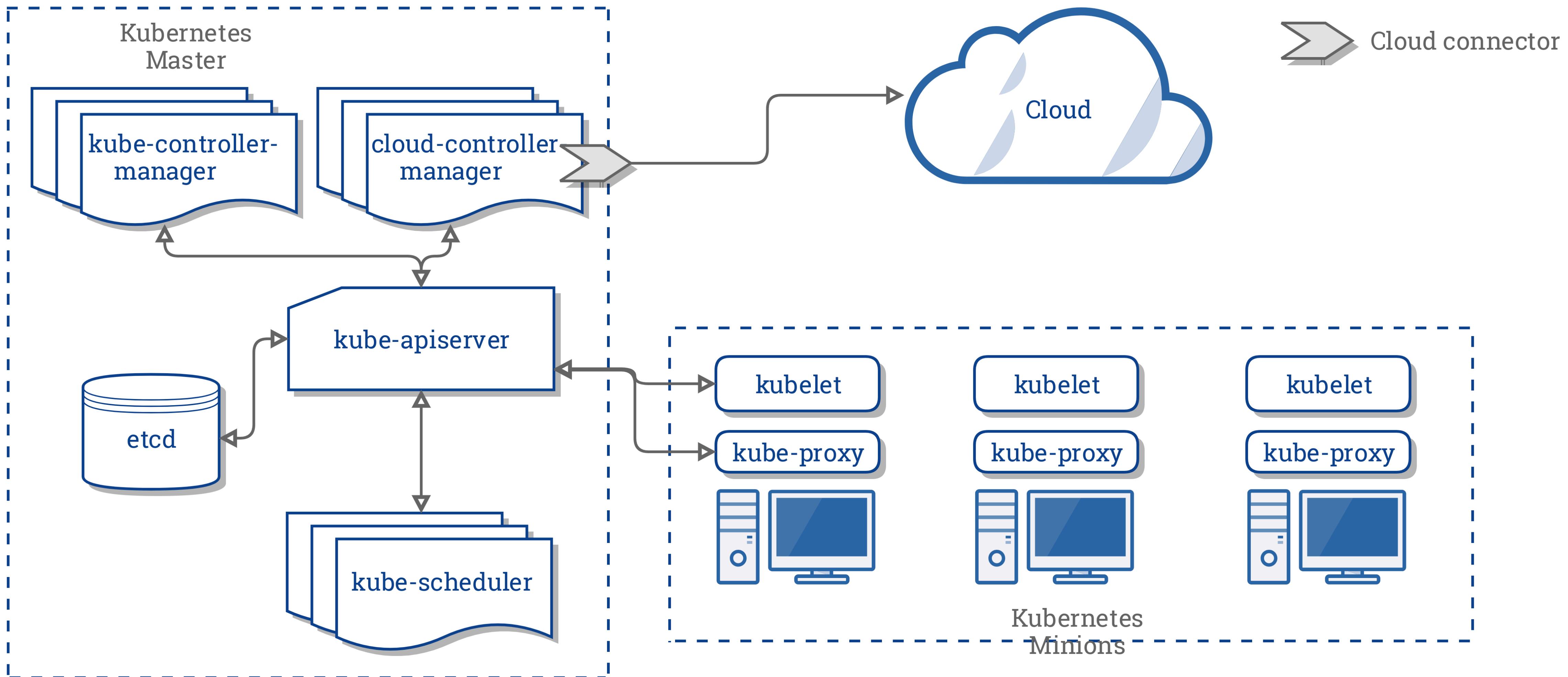


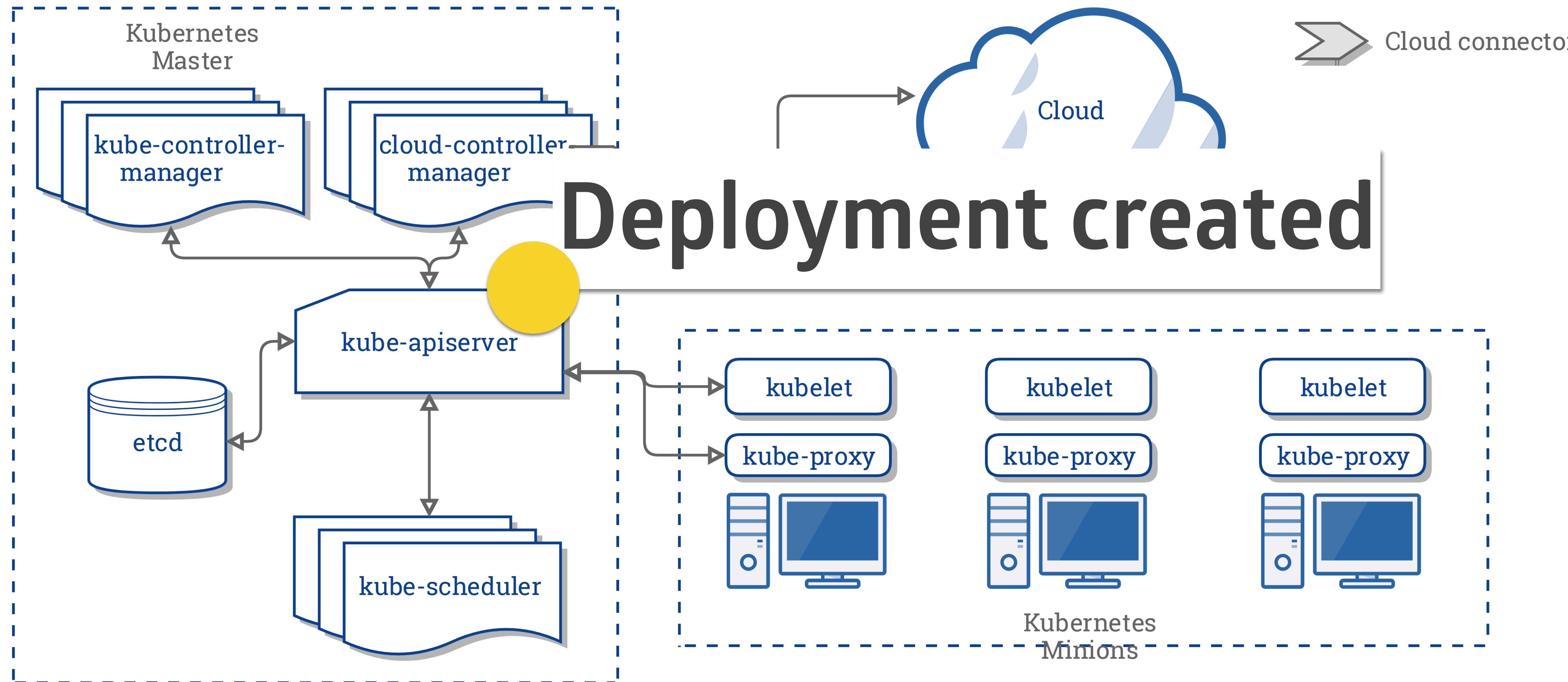
1 - Deploying a simple Web Application



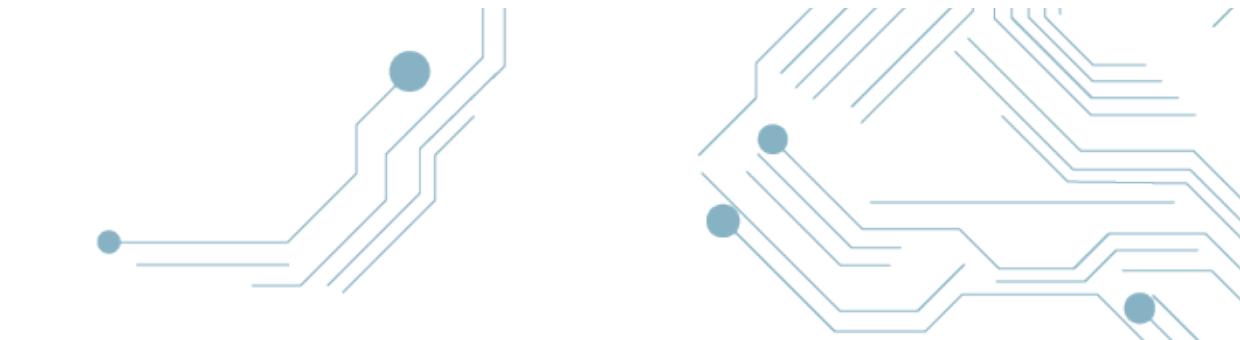
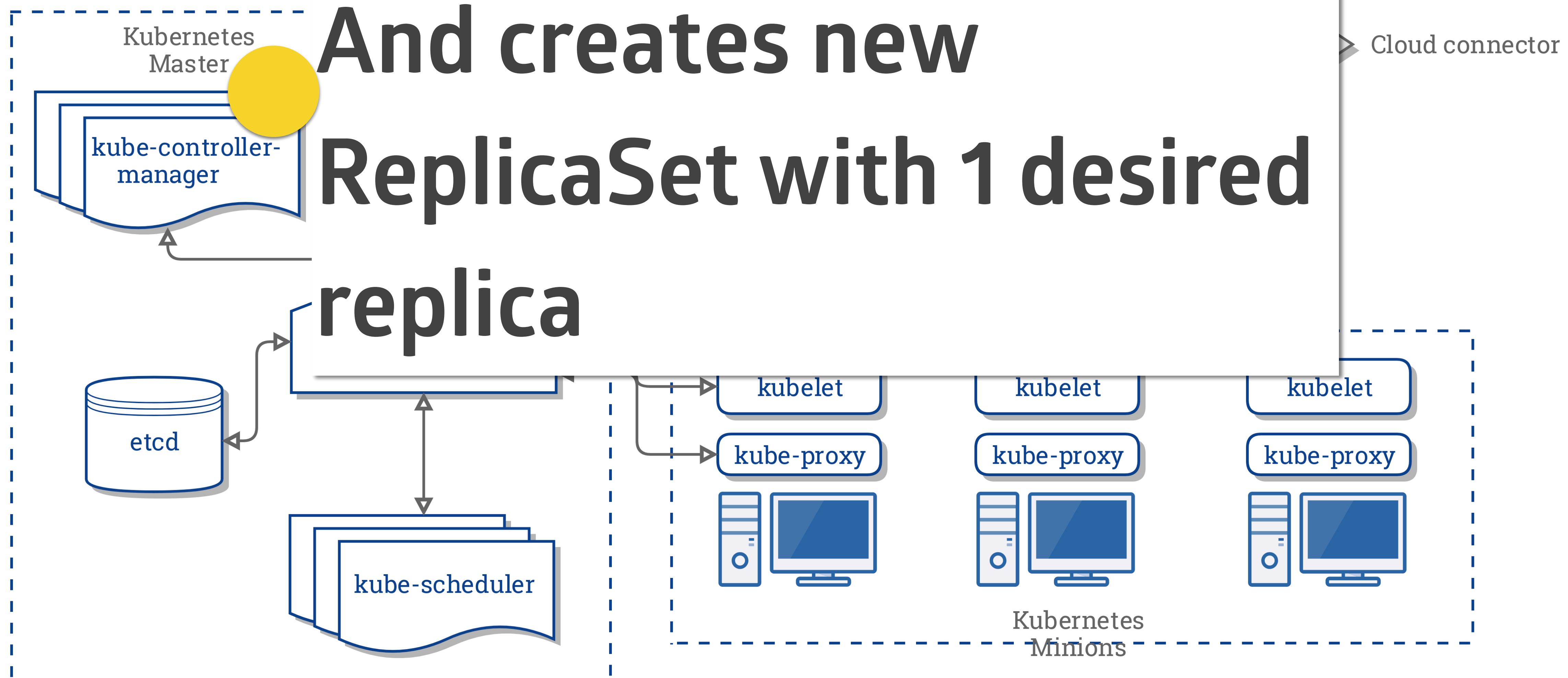
What did just happen?



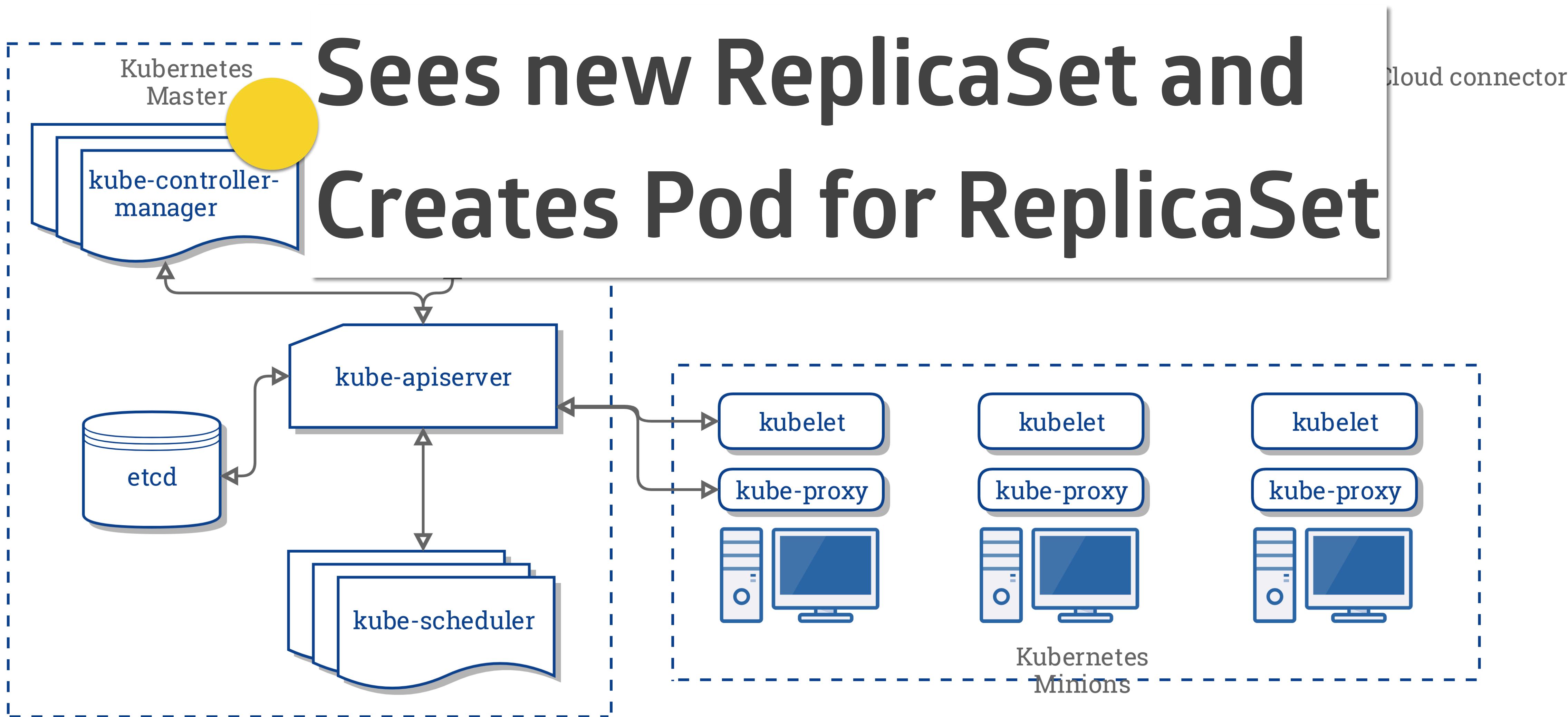


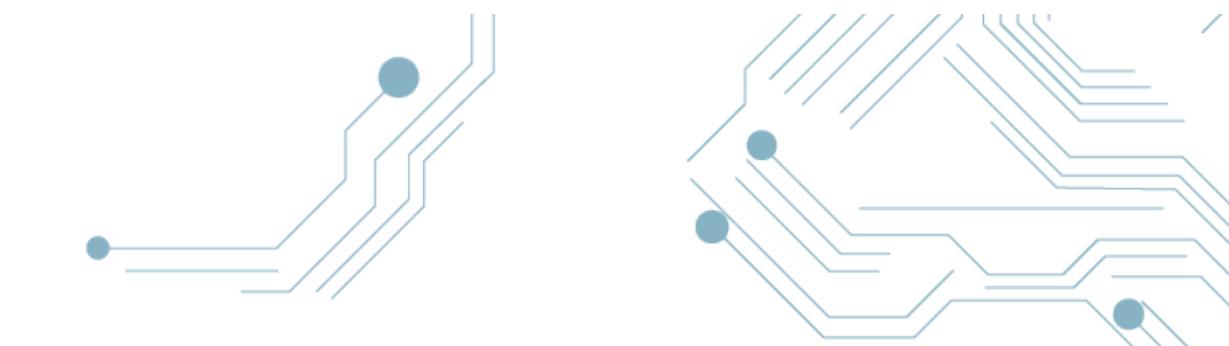
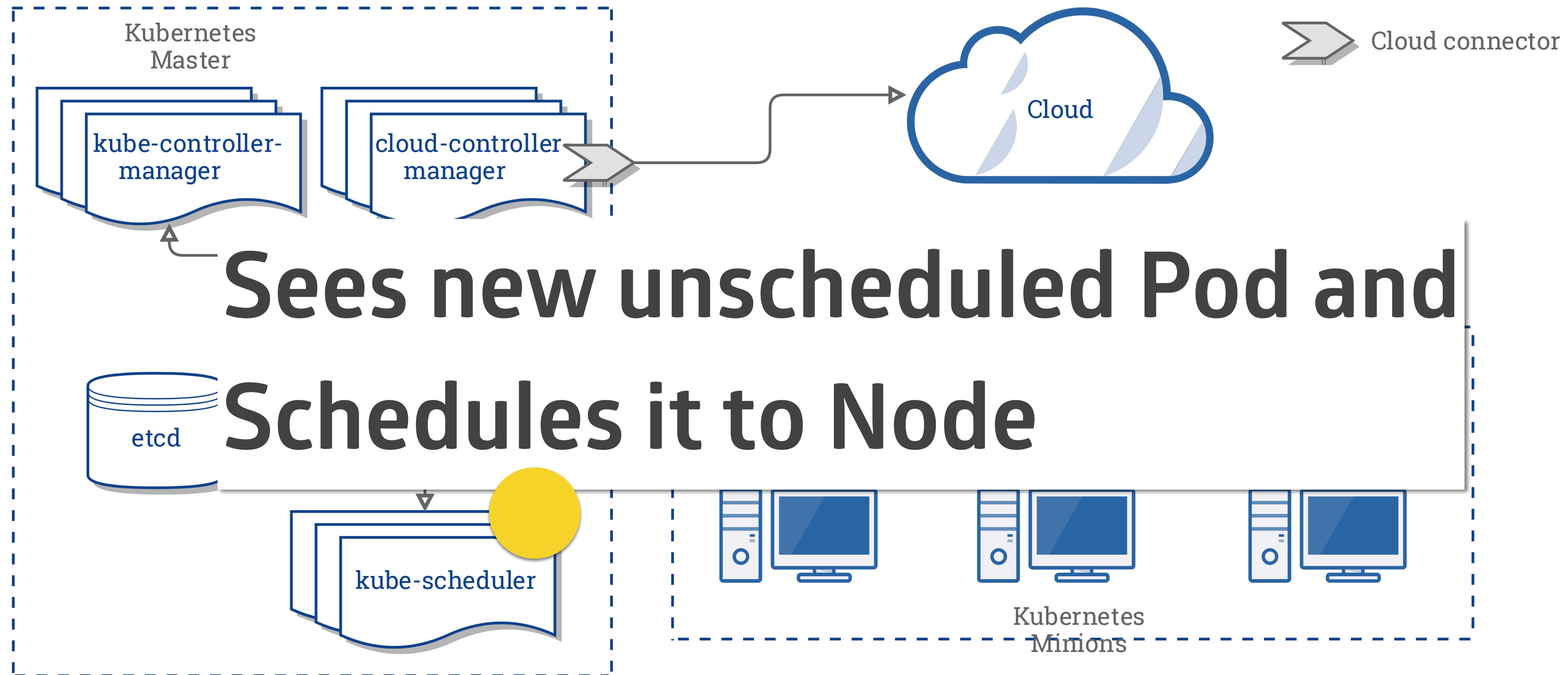


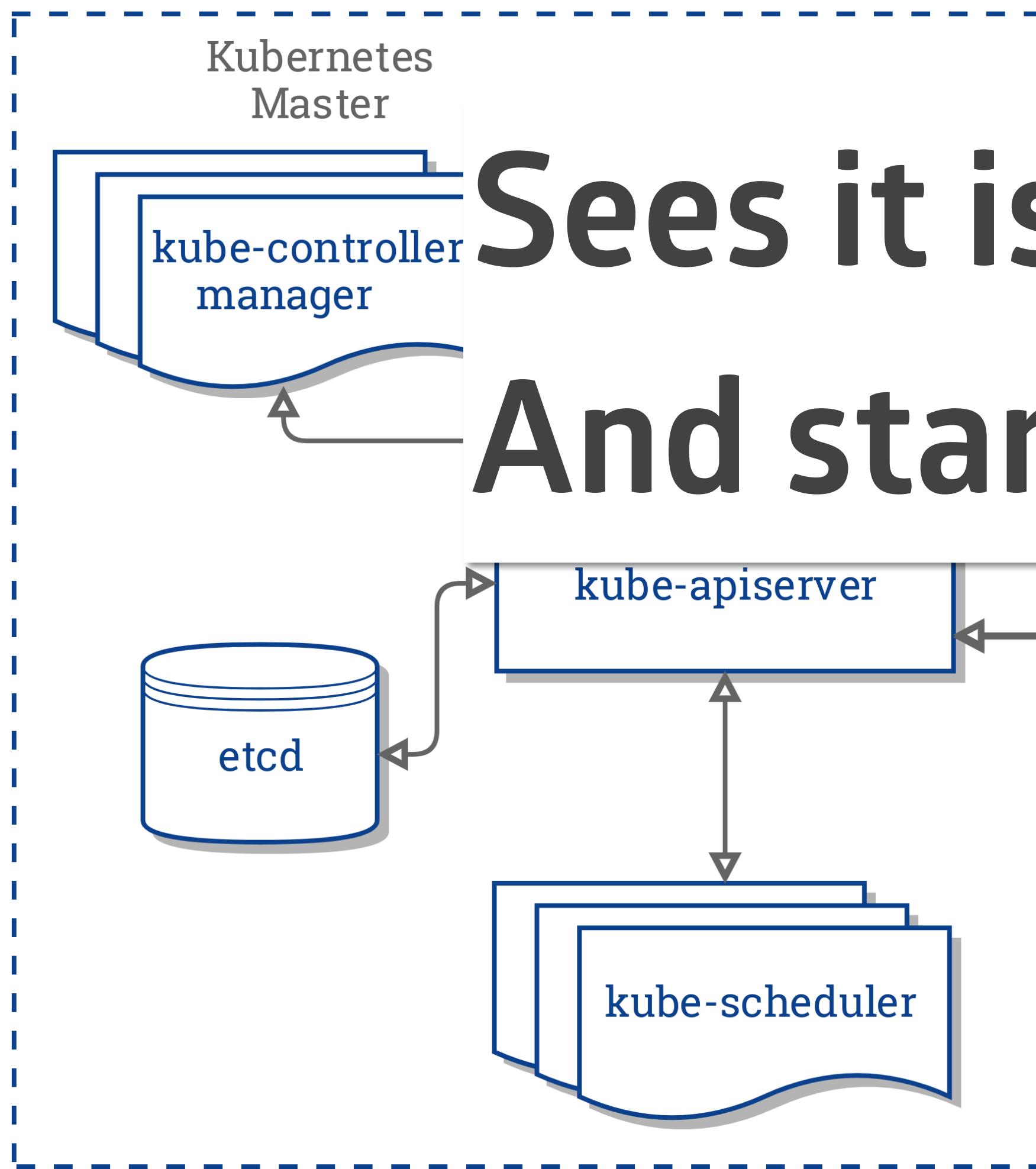
Sees new Deployment



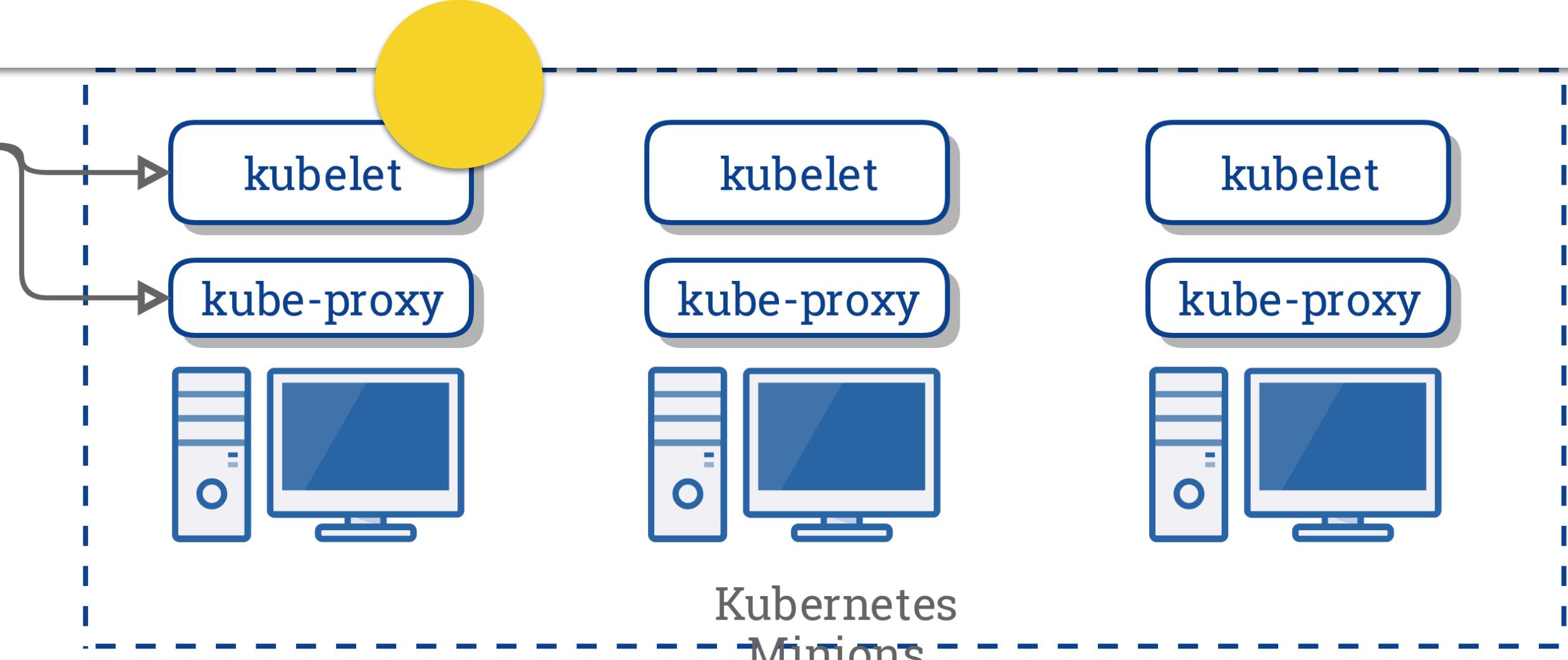
Sees new ReplicaSet and Creates Pod for ReplicaSet



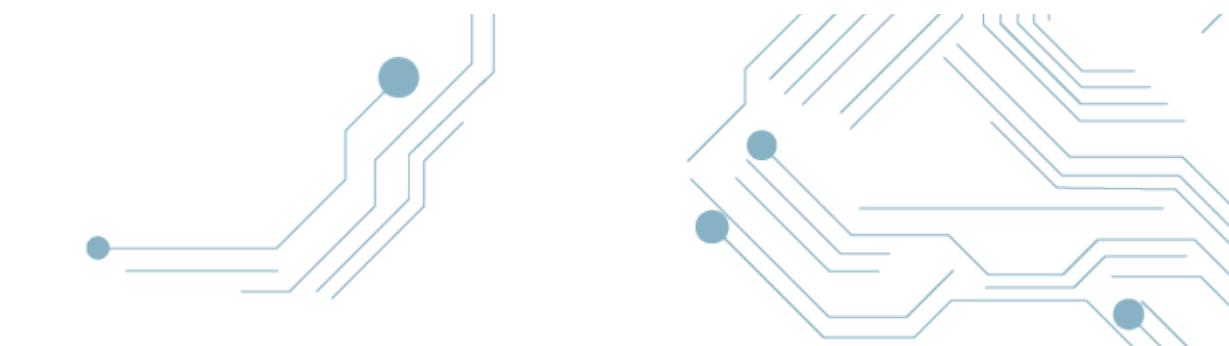
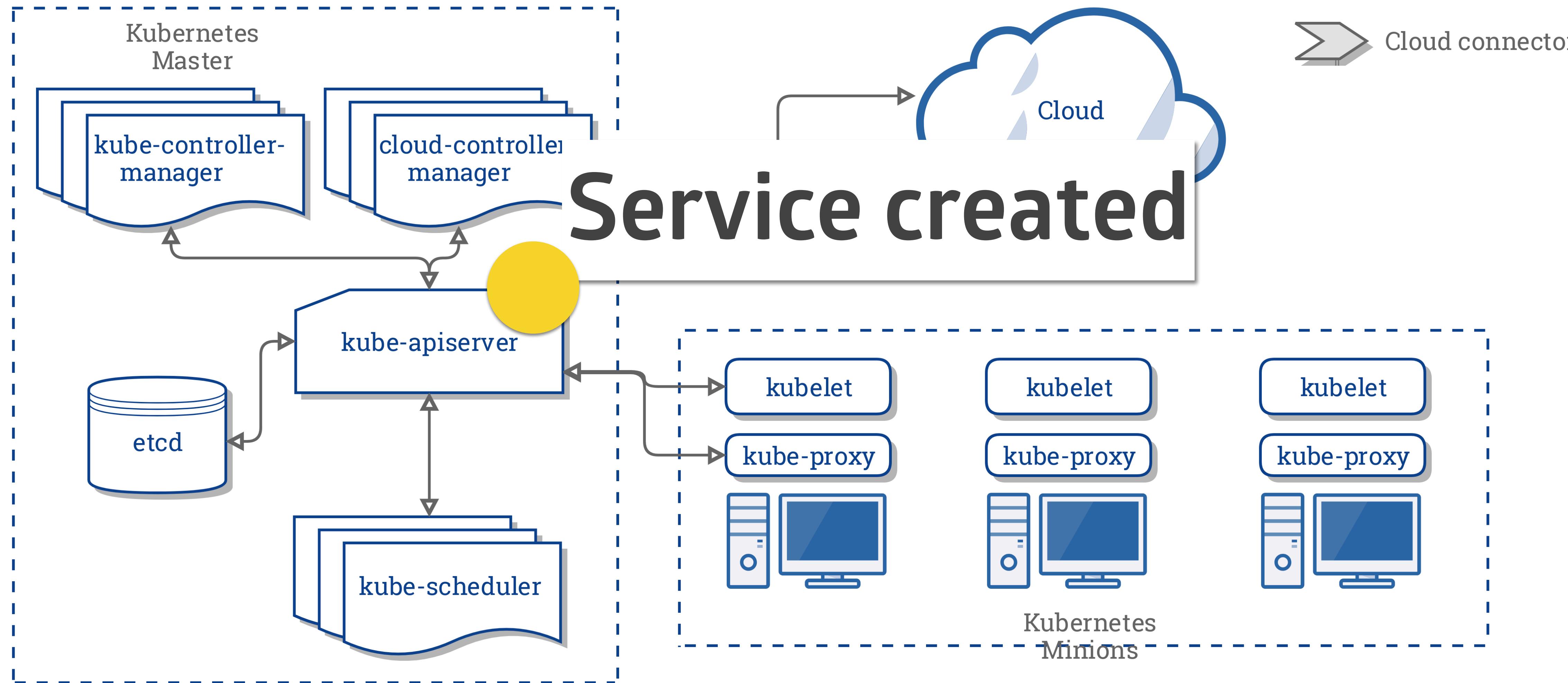




Sees it is supposed to start a Pod And starts its Containers



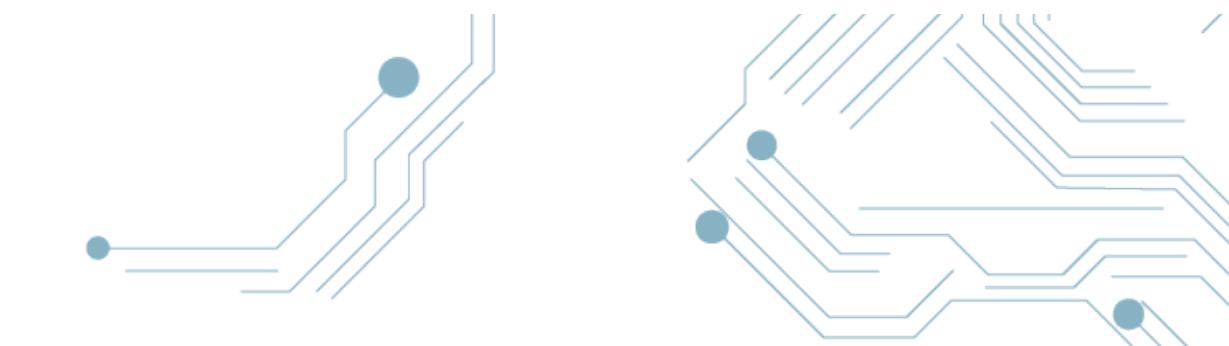
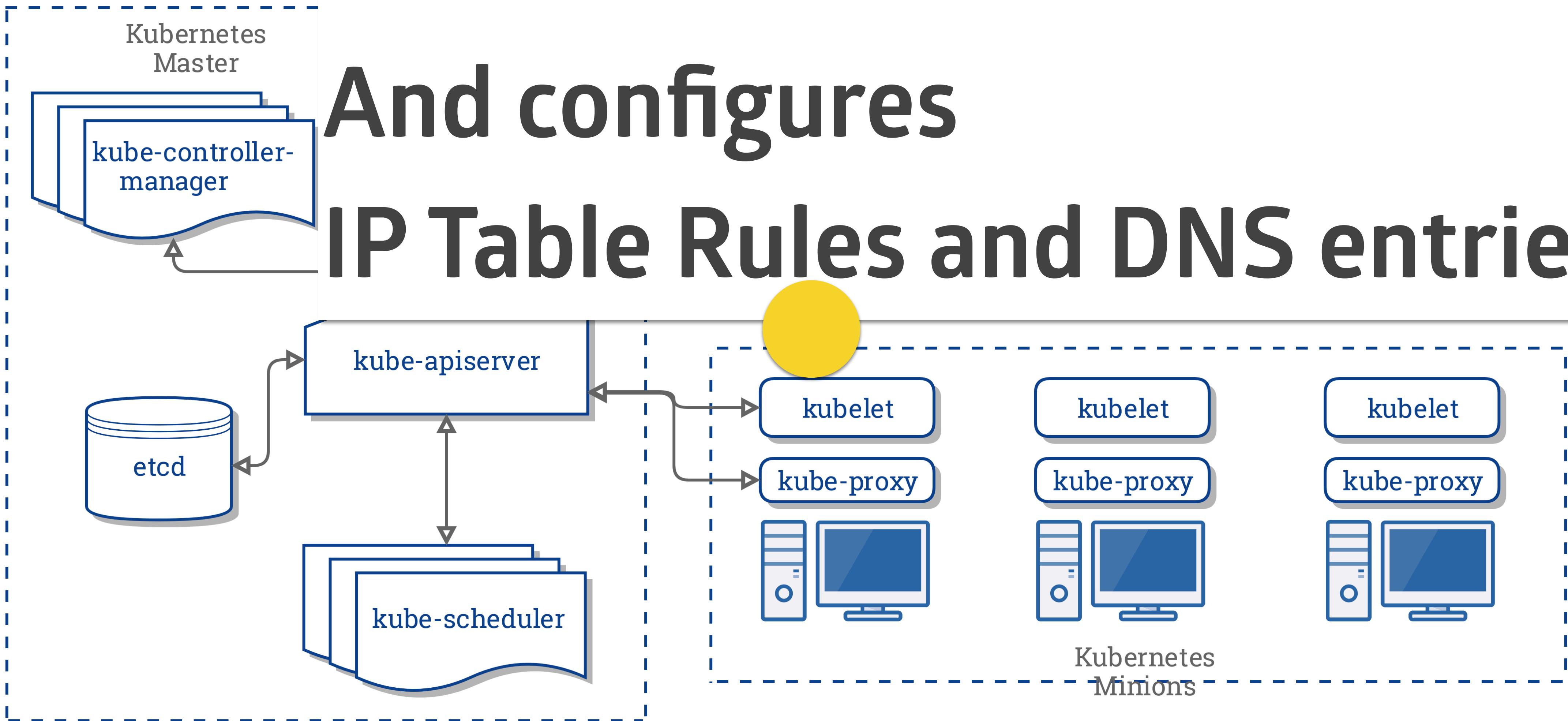
Cloud connector

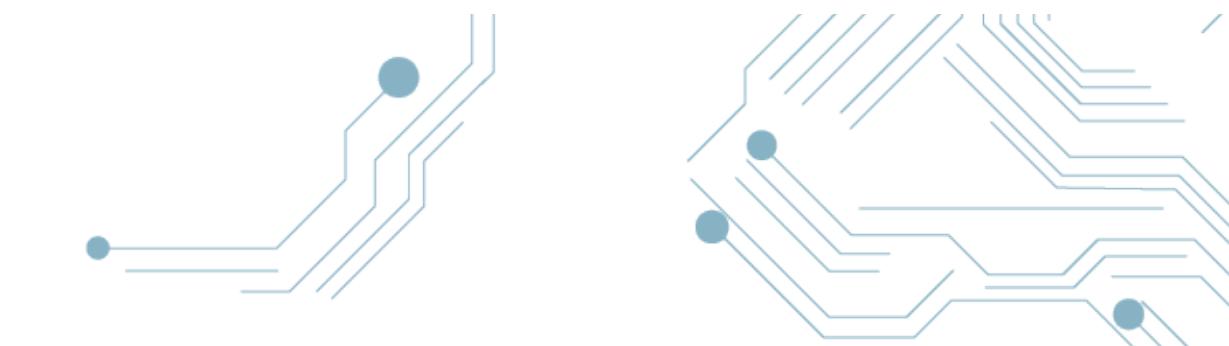
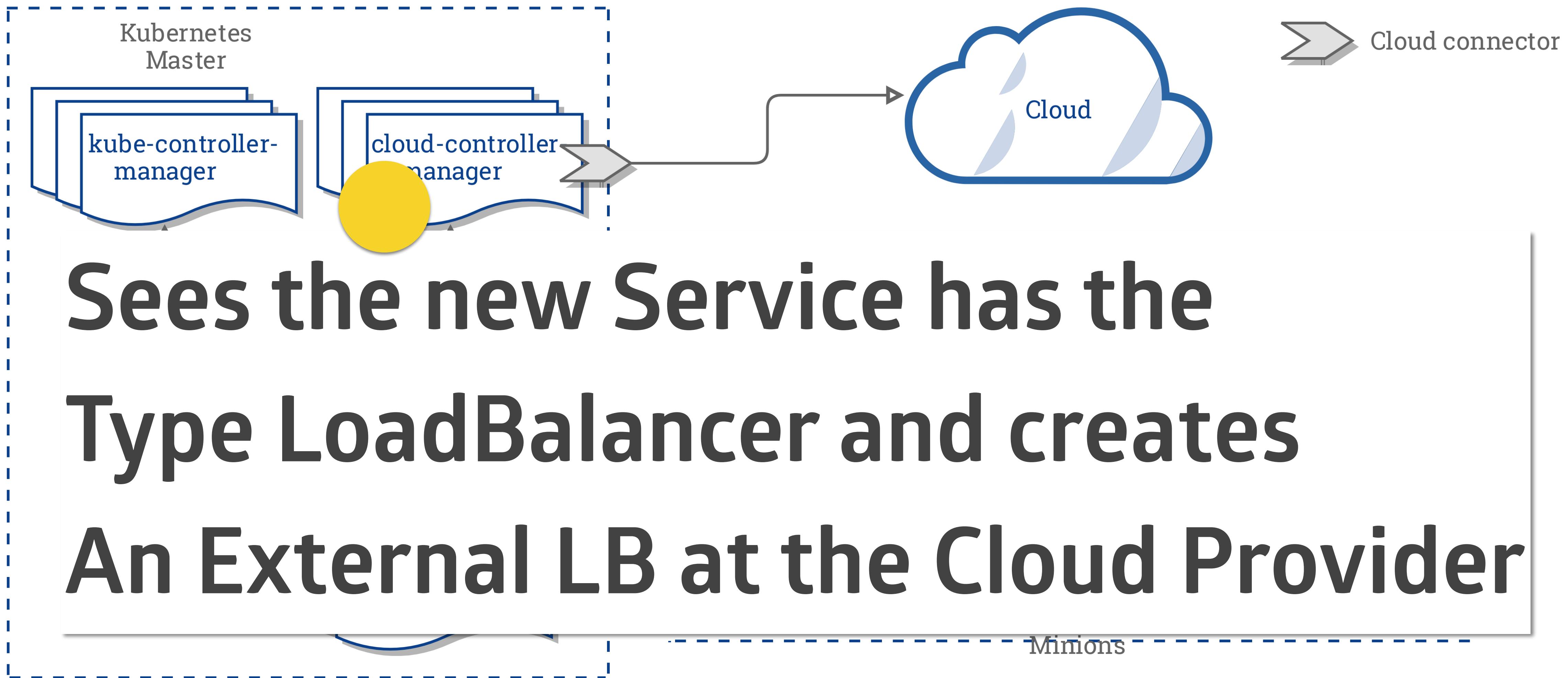


Sees the new Service

And configures

IP Table Rules and DNS entries

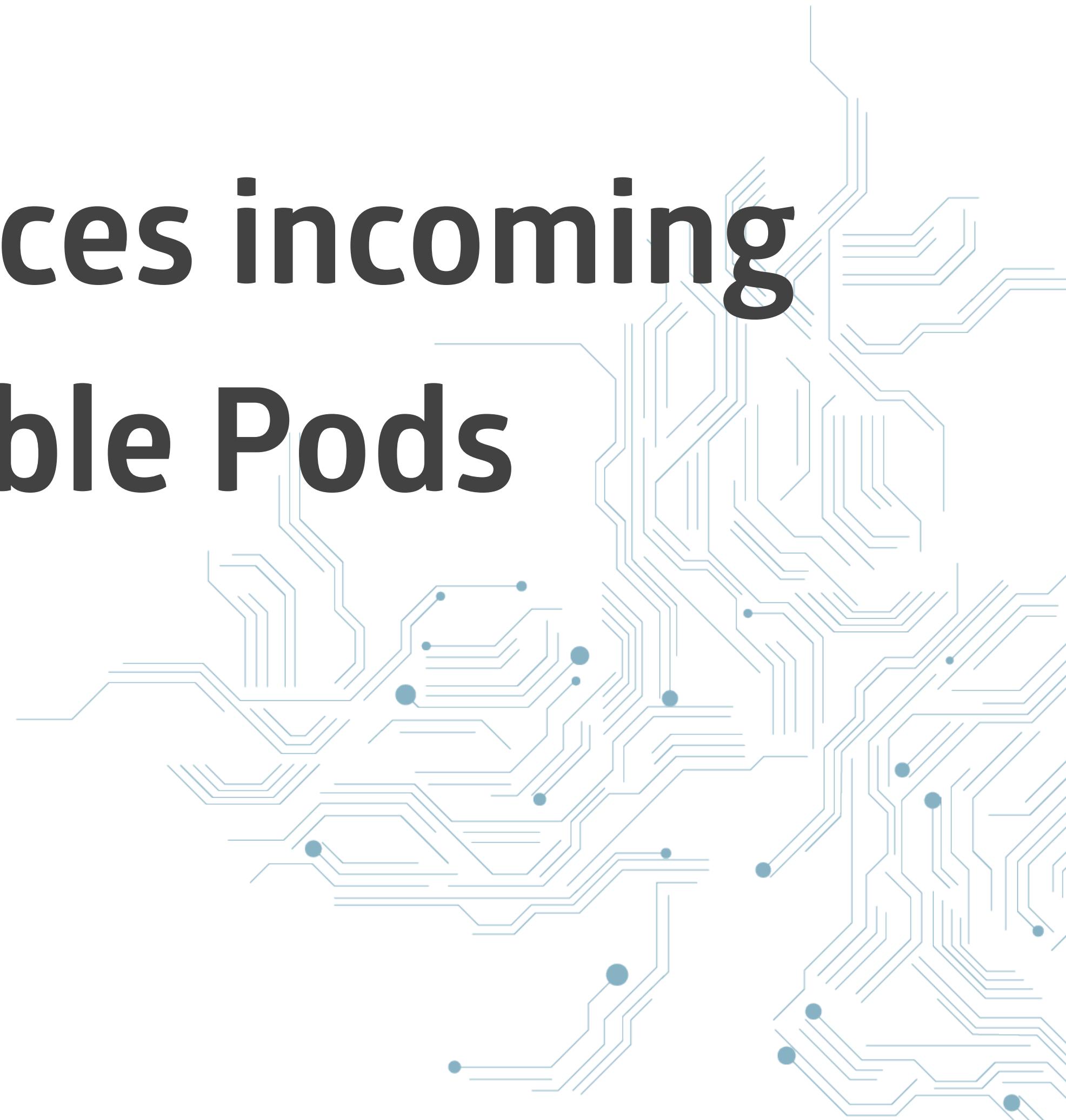




How is traffic routed to the Pod



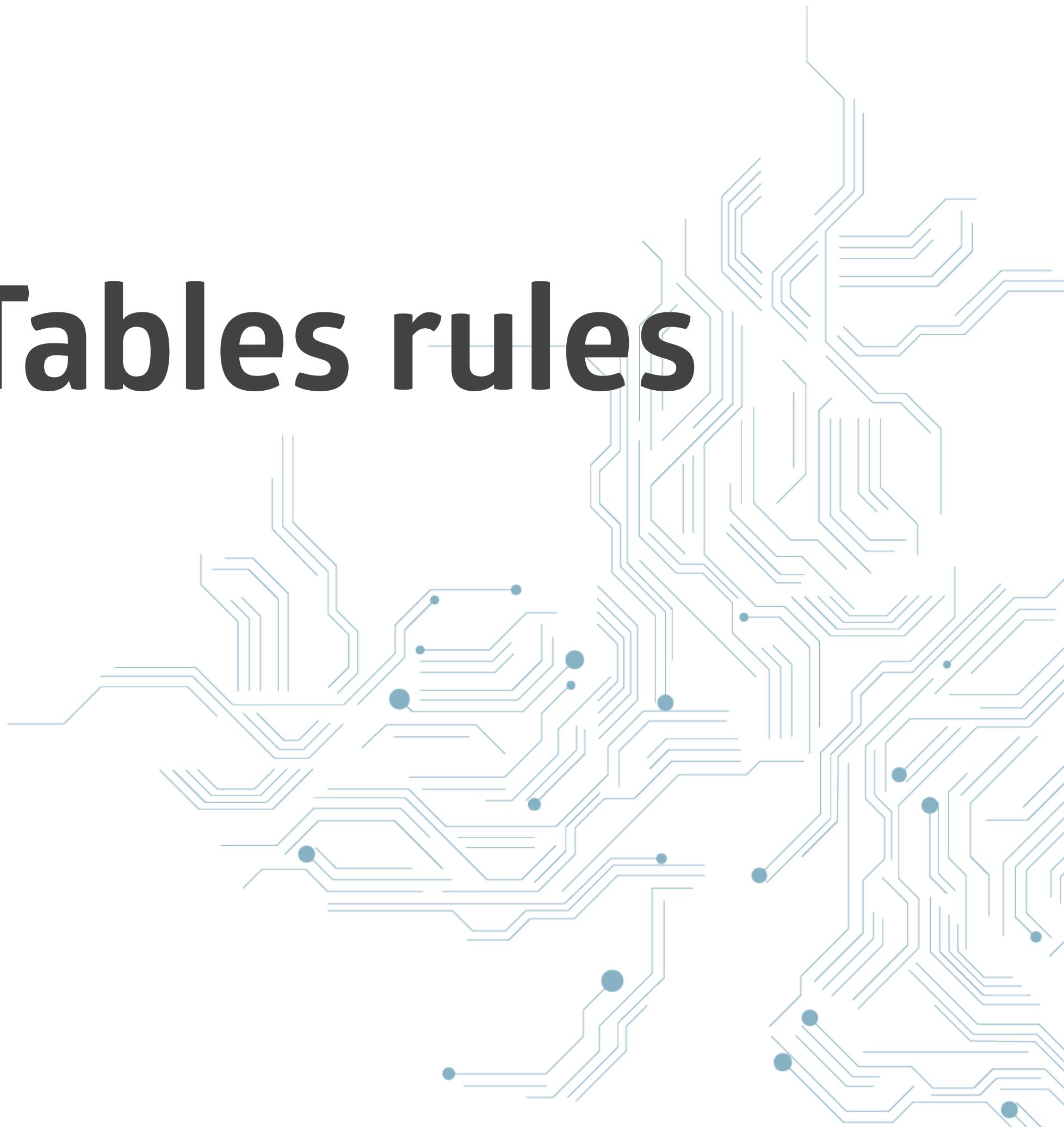
**The Service load-balances incoming
traffic to all available Pods**



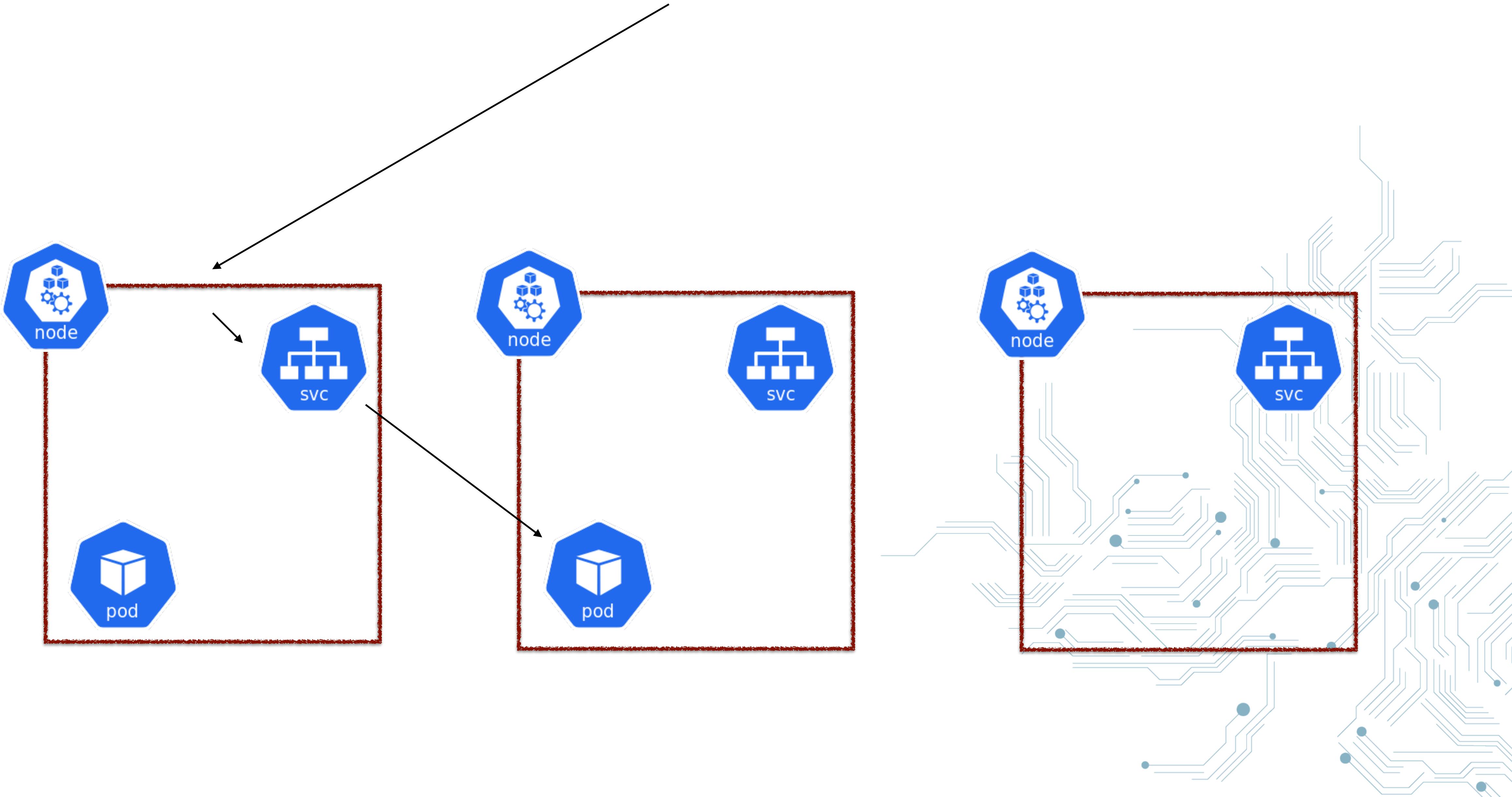
Every Service has a virtual IP



Round Robin with IP Tables rules



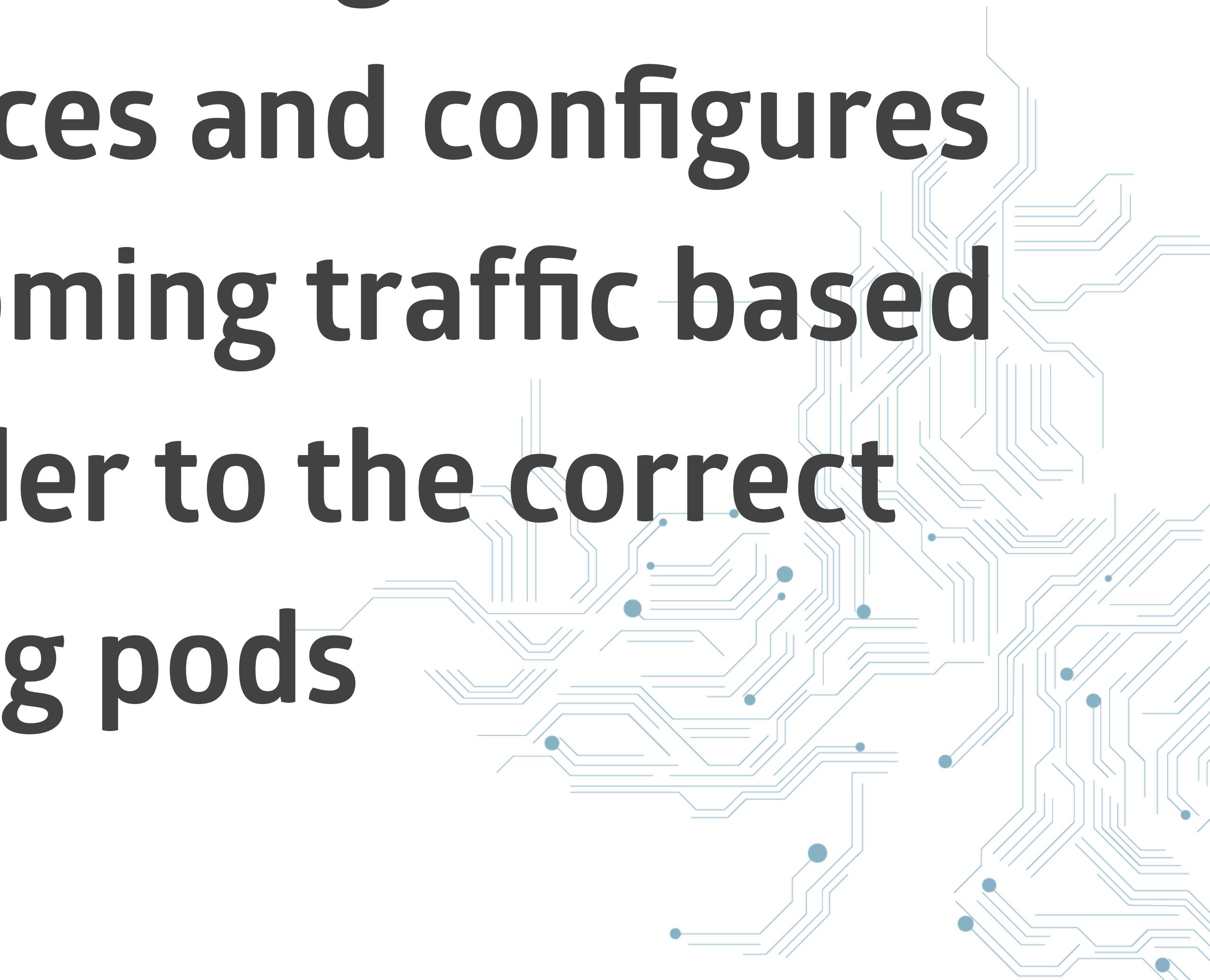
OpenStack LoadBalancer



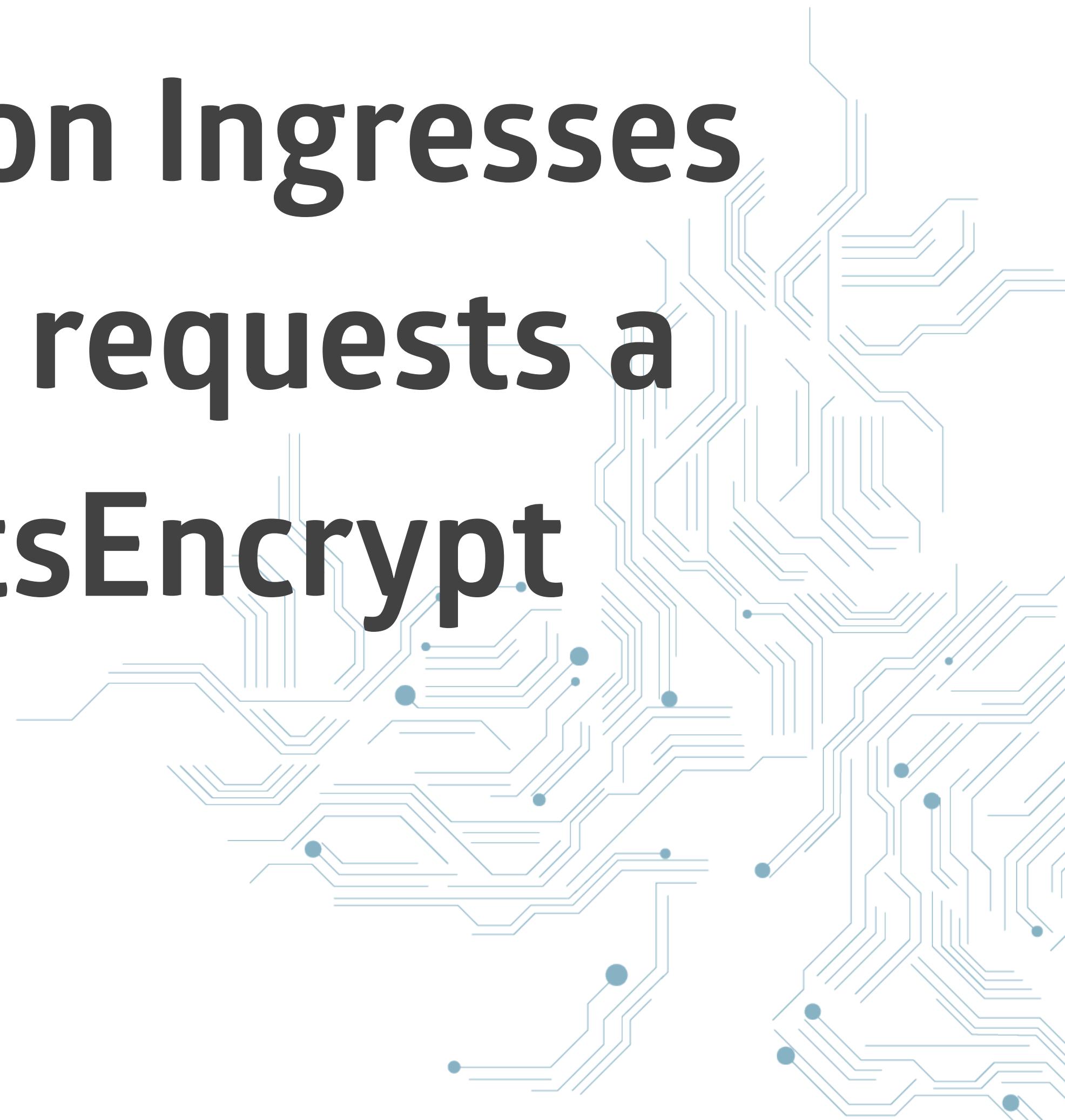
9 - Using an Ingress with TLS



The ingress controller (nginx) listens
on Ingress Resources and configures
itself to route incoming traffic based
on the host header to the correct
running pods



**Cert-manager listens on Ingresses
and if they want TLS, requests a
certificate from LetsEncrypt.**



**External-DNS listens on Ingresses
and creates DNS entries at AWS
Route 53**



How is traffic routed to the Pod



OpenStack LoadBalancer

