

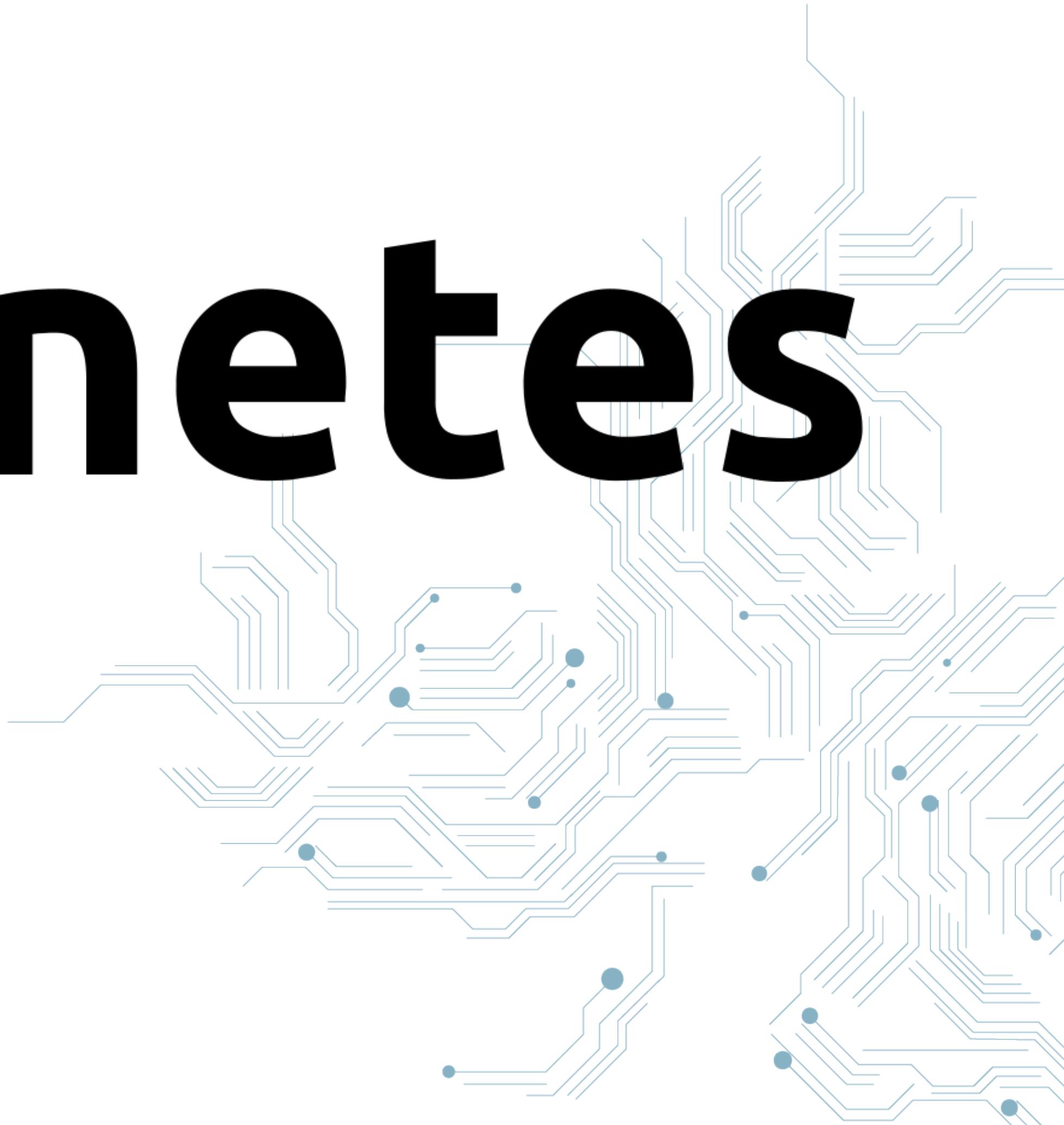
Kubernetes Workshop

Max Rosin & Maurice „Morre“ Meyer

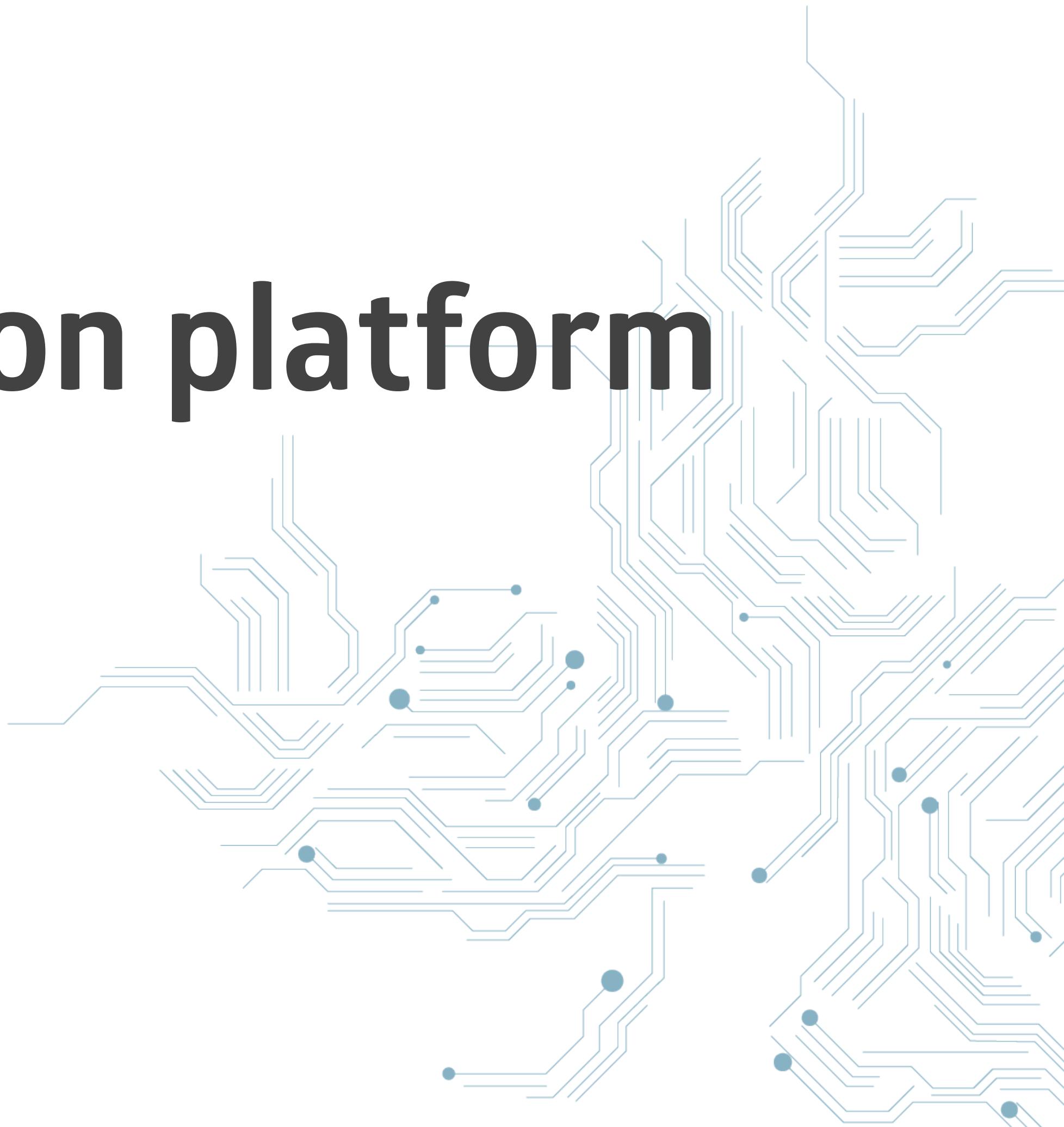




kubernetes



Container orchestration platform



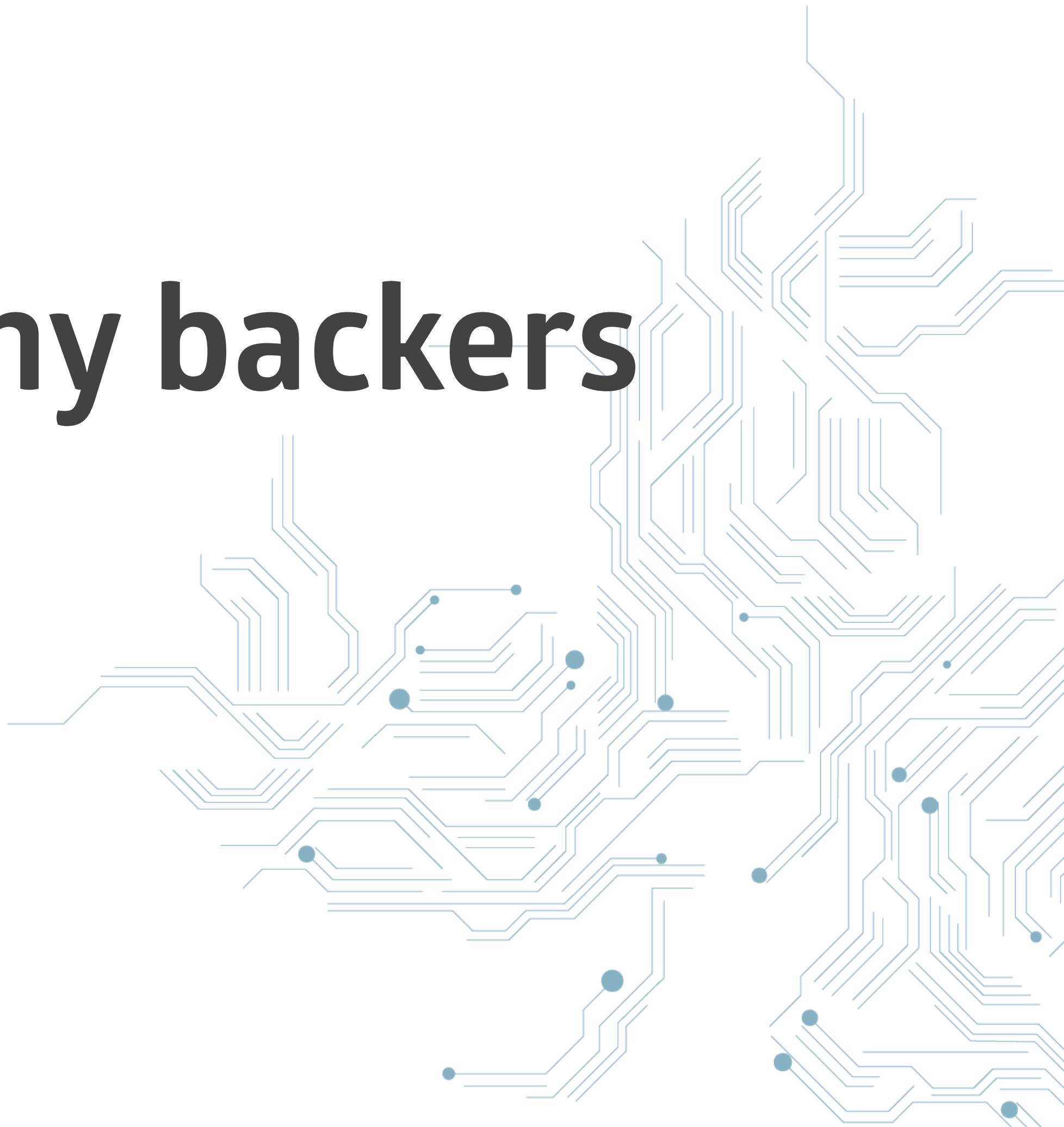
Deploy, run and scale your services in isolated containers



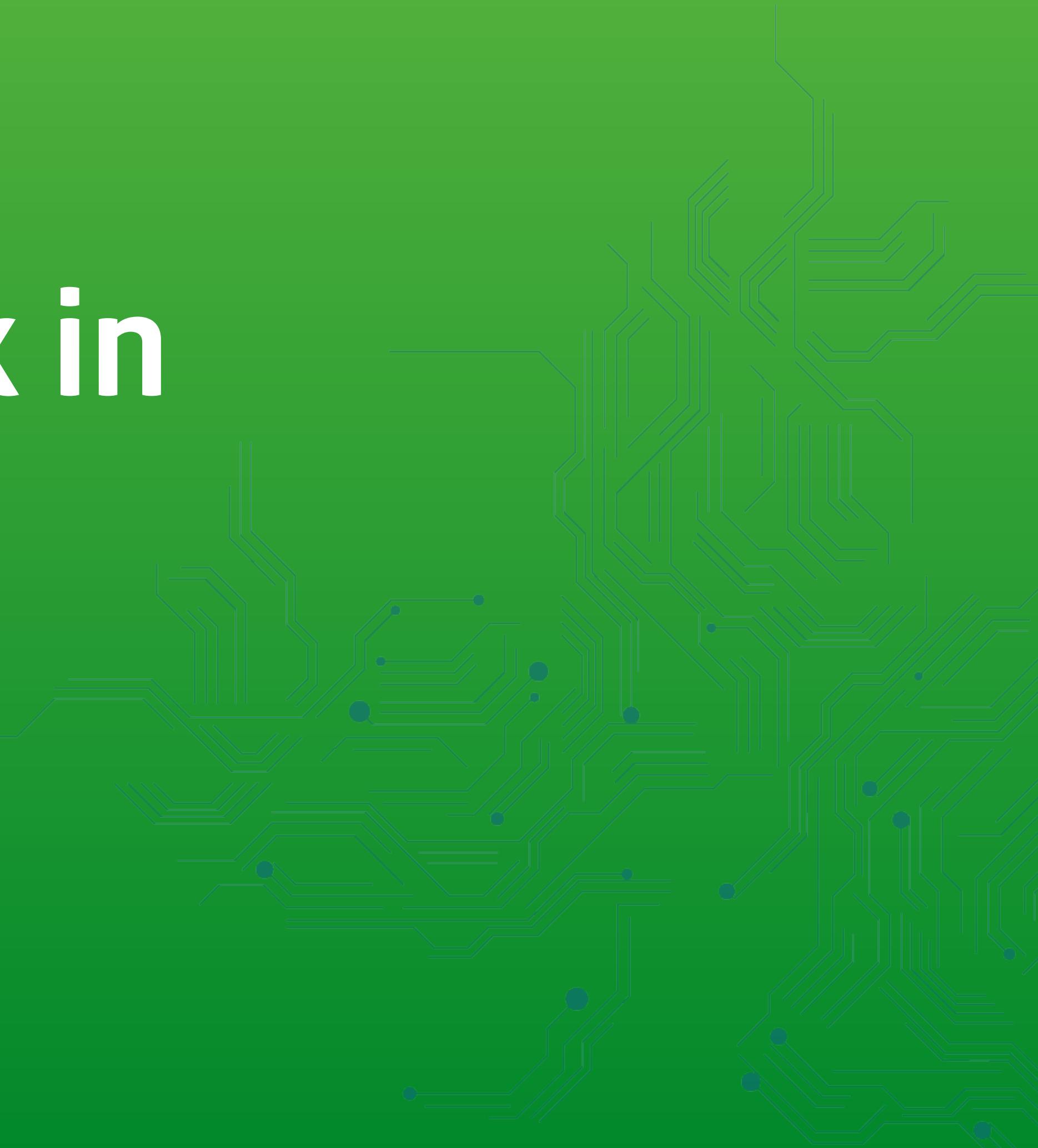
Very Powerful



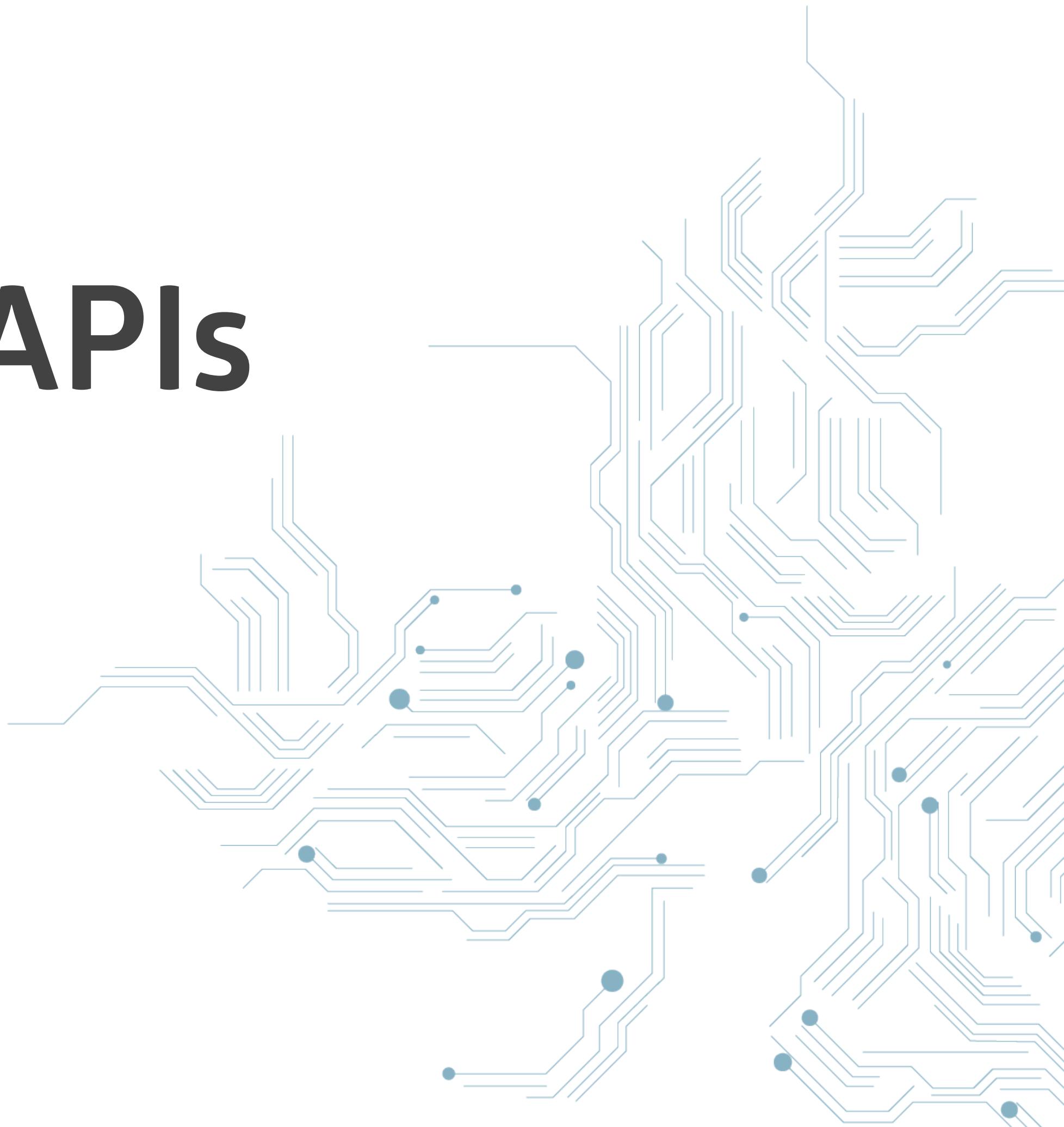
Lot's of large company backers



No vendor lock in



Standardized APIs



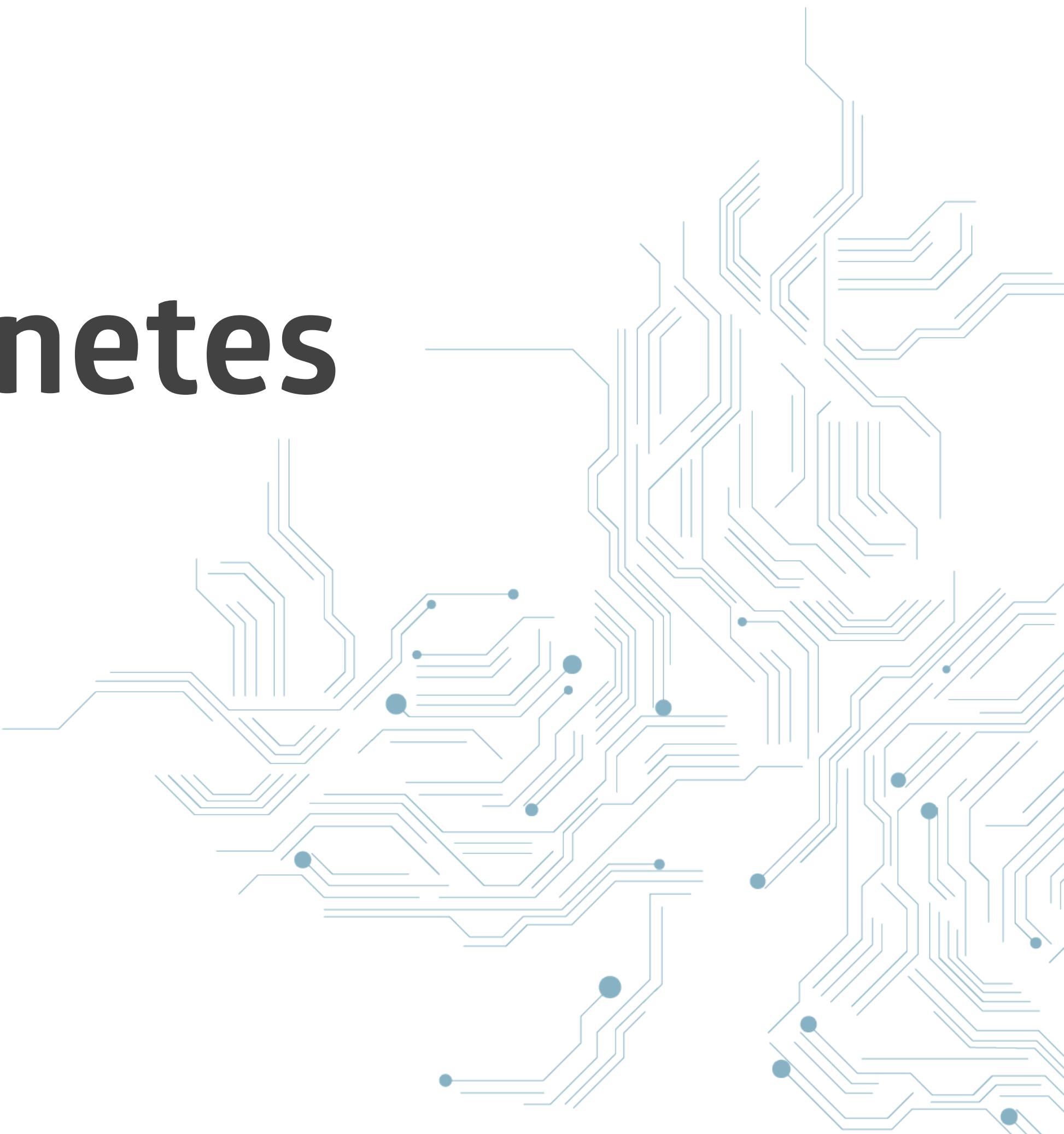
Runs on
Your laptop
Bare metal
Cloud Providers



**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

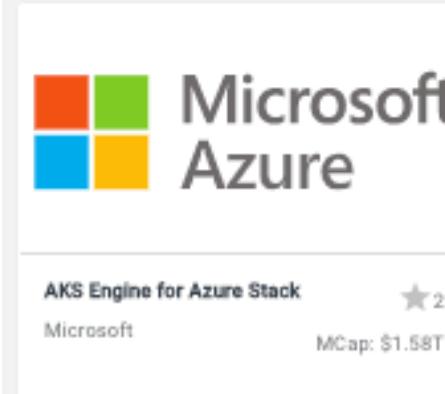
Serverless

Members



1163

Platform - Certified Kubernetes - Hosted (45)



Microsoft Azure

AKS Engine for Azure Stack
Microsoft
MCap: \$1.58T



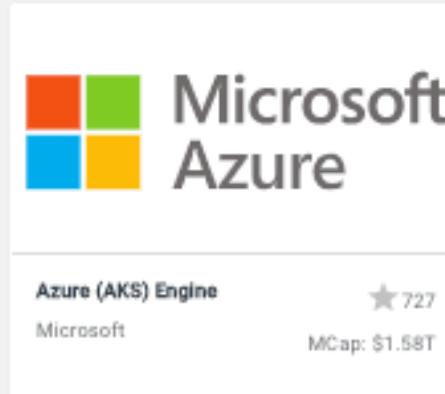
Alibaba Cloud

Alibaba Cloud Container Service for Kubernetes
Alibaba Cloud
MCap: \$717.42B



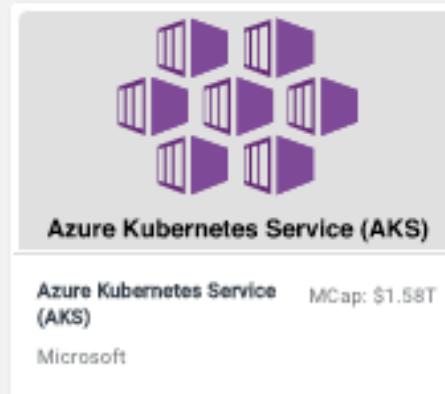
aws

Amazon Elastic Container Service for Kubernetes (EKS)
Amazon Web Services
MCap: \$1.57T



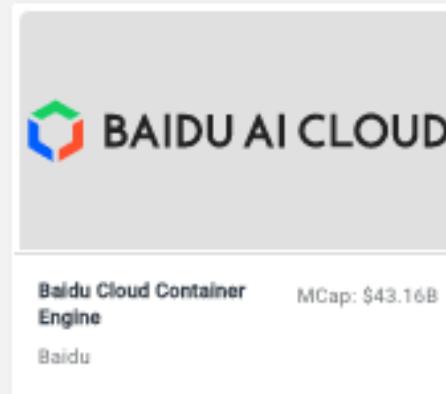
Microsoft Azure

Azure (AKS) Engine
Microsoft
MCap: \$1.58T



Azure Kubernetes Service (AKS)
Microsoft

Azure (AKS) Engine
Microsoft
MCap: \$1.58T



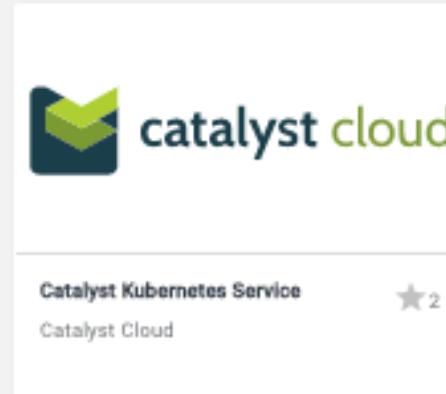
BAIDU AI CLOUD

Baidu Cloud Container Engine
Baidu
MCap: \$43.16B



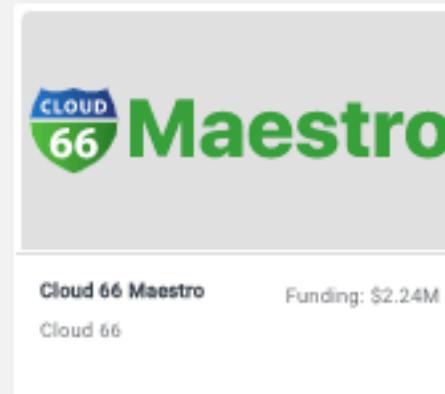
博云 BoCloud

BoCloud BeyondContainer
BoCloud
Funding: \$29.77M



catalyst cloud

Catalyst Kubernetes Service
Catalyst Cloud
MCap: \$1.2M



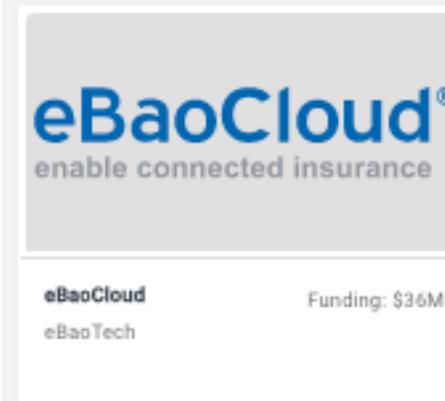
Cloud 66 Maestro

Cloud 66
Funding: \$2.24M



Maestro

DigitalOcean Kubernetes
DigitalOcean
Funding: \$455.41M



eBaoCloud®
enable connected insurance

eBaoCloud
eBaoTech
Funding: \$36M



ELASTX

ELASTX Private Kubernetes
ELASTX



Google Kubernetes Engine

Google Kubernetes Engine (GKE)
Google
MCap: \$1.06T



HUAWEI

Huawei Cloud Container Engine (CCE)
Huawei Technologies



IBM Cloud Kubernetes Service

IBM Cloud Kubernetes Service
IBM
MCap: \$111.93B



inspur 浪潮

Inspur Cloud Container Engine and Inspur Open Platform
Inspur Group
MCap: \$8.19B



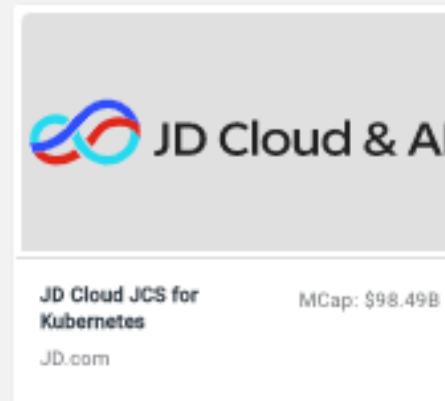
inspur ICKS

Inspur ICKS
Inspur Group
MCap: \$8.19B



inspur Open Platform for ARM

Inspur Open Platform for ARM
Inspur Group
MCap: \$8.19B



JD Cloud JCS for Kubernetes

JD.com
JD Cloud & AI
JD.com
MCap: \$98.49B



JD.COM

JD.com JDOS Hosted
JD.com
MCap: \$98.49B



Kingsoft Cloud

Kingsoft Container Engine
Kingsoft
MCap: \$7.24B



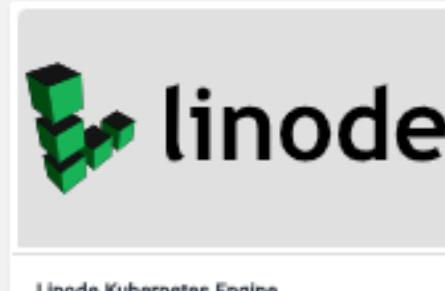
朗澈科技

Launcher Tech LStack Container Service for Kubernetes
Hangzhou Launcher Technology



Linaro

Linaro Developer Cloud Kubernetes Service
Linaro
MCap: \$1.1B



linode

Linode Kubernetes Engine
Linode
MCap: \$5.64B



MAIL.RU CLOUD SOLUTIONS

Mail.Ru Cloud Containers
Mail.Ru Group
MCap: \$5.64B



nirmata

Nirmata Managed Kubernetes
Nirmata
MCap: \$4.48B



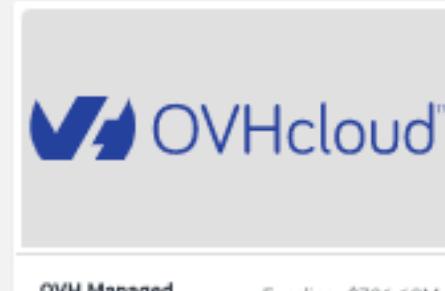
NUTANIX™

Nutanix Karbon
Nutanix
MCap: \$4.48B



ORACLE

Oracle Container Engine
Oracle
MCap: \$15.8B



OVHcloud™

OVH Managed Kubernetes Service
OVHcloud
Funding: \$736.63M



QINGCLOUD

QingCloud KubeSphere Engine (QKE)
QingCloud
Funding: \$280.83M



RAFAY

Rafay
Rafay Systems
Funding: \$8M



Red Hat OpenShift Dedicated

Red Hat OpenShift Dedicated
Red Hat
MCap: \$111.93B



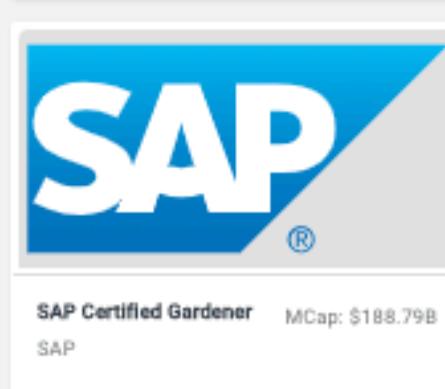
Red Hat OpenShift on IBM Cloud

Red Hat OpenShift on IBM Cloud
IBM
MCap: \$111.93B



SAMSUNG SDS

Samsung SDS Kubernetes Service
Samsung SDS



SAP

SAP Certified Gardener
SAP
MCap: \$188.79B



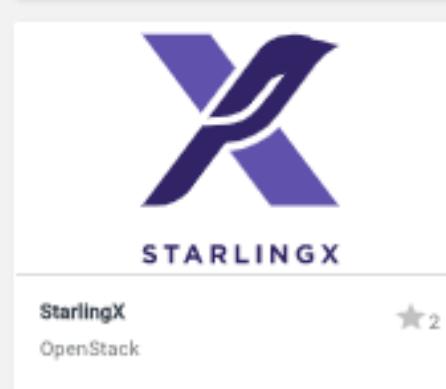
Scaleway

Scaleway Kubernetes Kapsule
Scaleway
MCap: \$717.42B



SOFASTACK™

SOFASTACK Cloud Application Fabric Engine
Ant Financial
MCap: \$717.42B



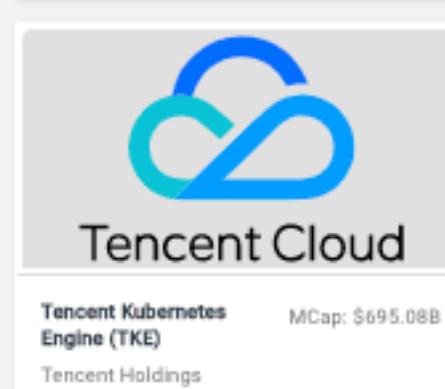
STARLINGX

StarlingX
OpenStack
MCap: \$1.2M



SysEleven

SysEleven MetaKube
SysEleven
MCap: \$1.2M



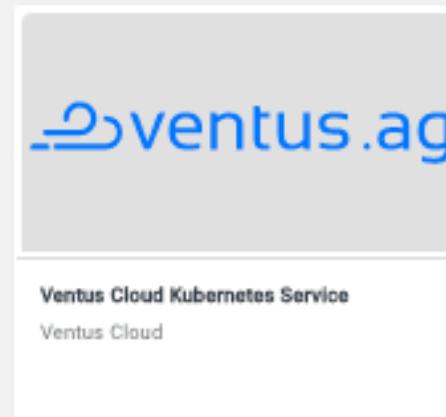
Tencent Cloud

Tencent Kubernetes Engine (TKE)
Tencent Holdings
MCap: \$695.08B



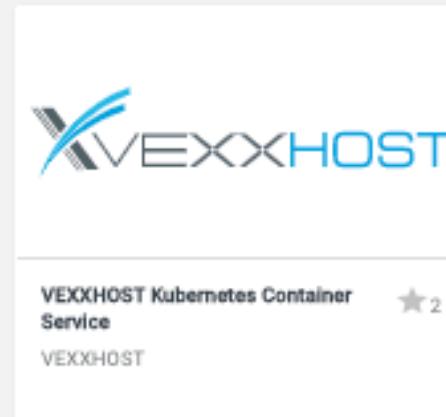
UCLOUD优刻得

UCLOUD Kubernetes Service (UKBS)
UCLOUD Information Technology
MCap: \$4.04B



ventus.ag

Ventus Cloud Kubernetes Service
Ventus Cloud
MCap: \$1.04B



VEXXHOST

VEXXHOST Kubernetes Container Service
VEXXHOST
MCap: \$1.04B



网宿云

WANGSU CLOUD Container Service
Wangsu Science & Technology
MCap: \$3.16B



ZTE

ZTE TECS OpenPalette
ZTE
MCap: \$24.92B

Easy setup



Easy upgrades



Easy scaling



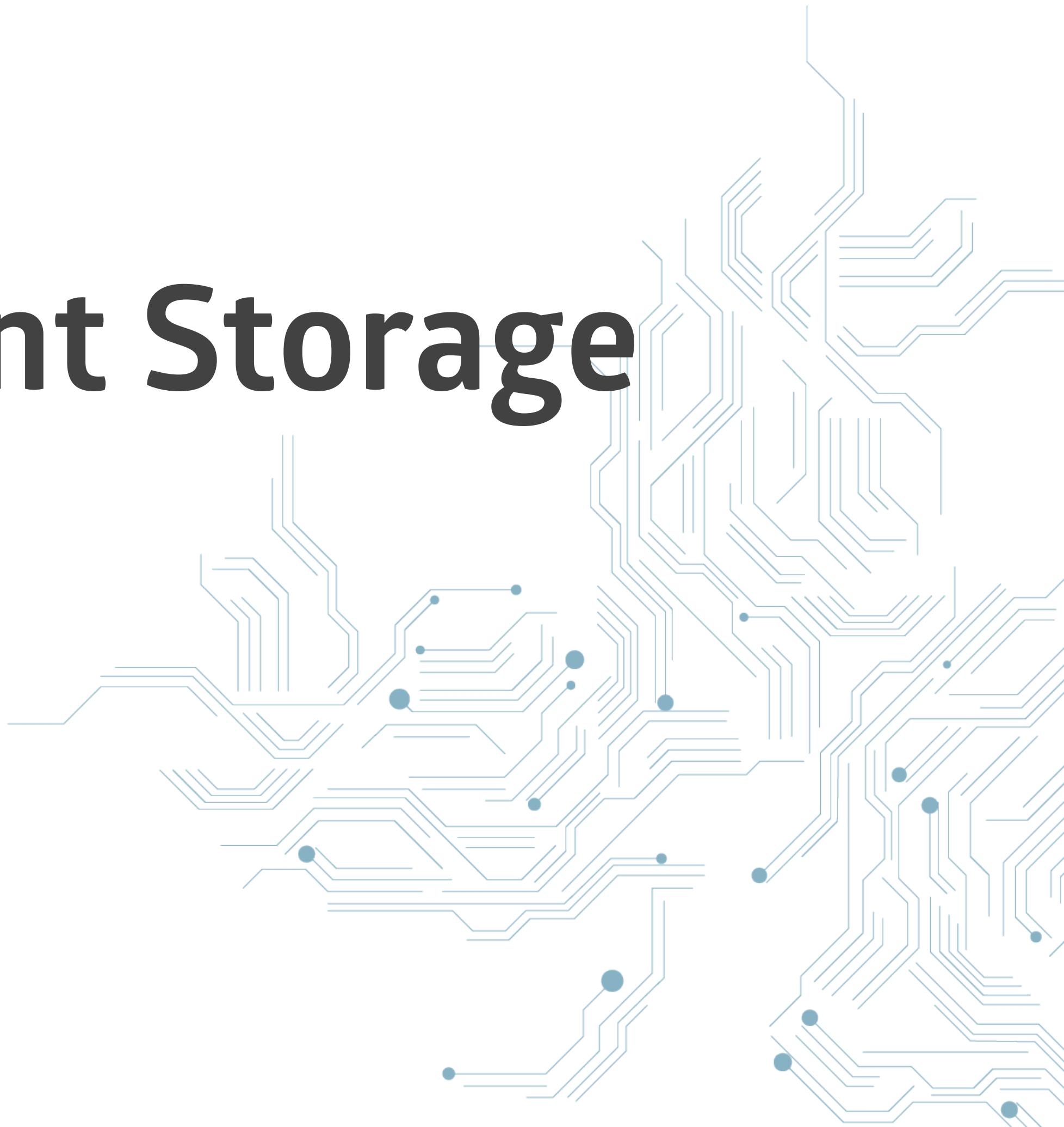
Features



Load Balancing



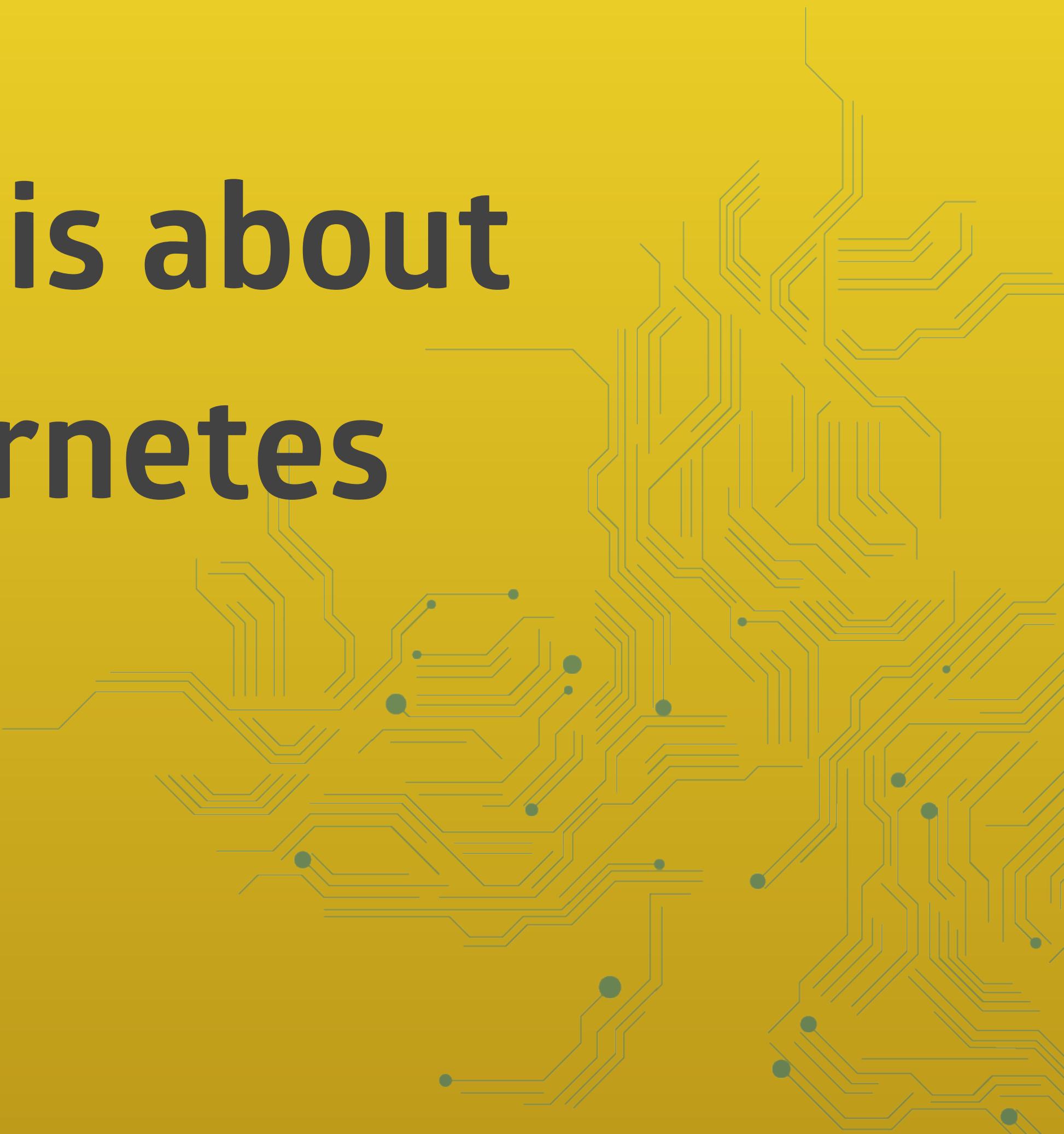
Distributed Persistent Storage



Backups



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



Learning curve

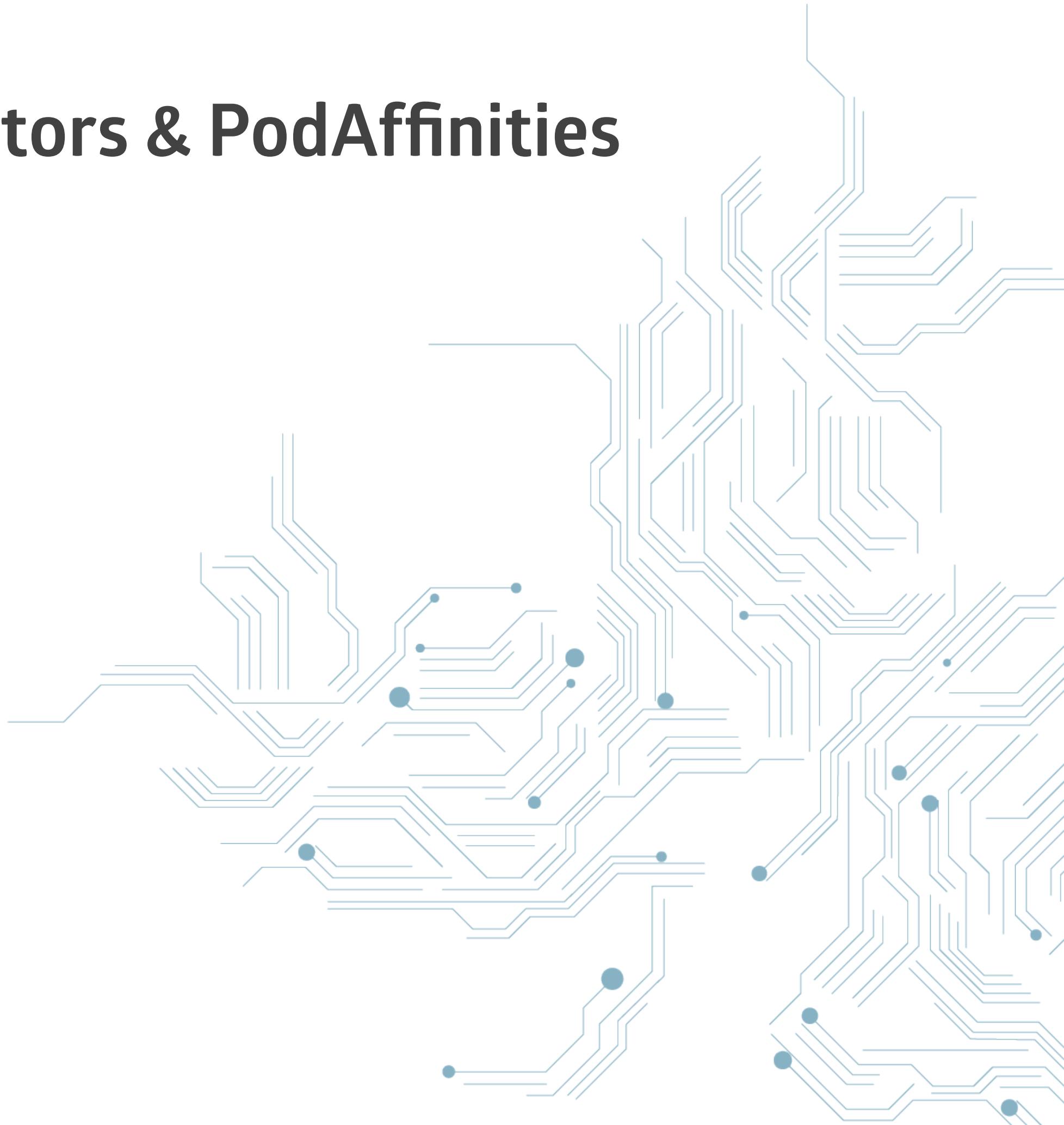


Agenda

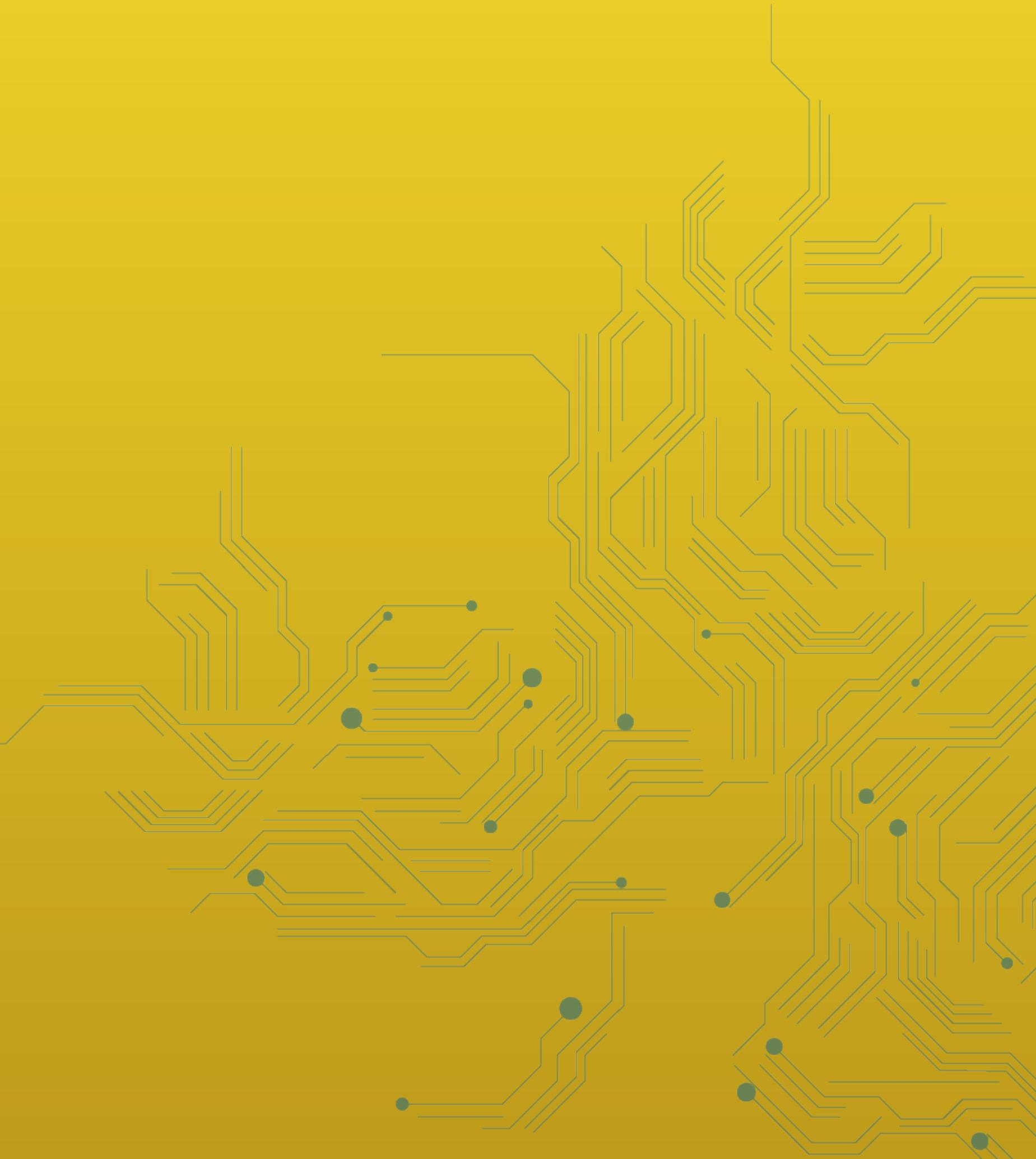




- **Deployments**
- **Readiness and Liveness-Probes, NodeSelectors & PodAffinities**
- **ConfigMaps & Secrets**
- **External DNS**
- **Let'sEncrypt with cert-manager**
- **nginx-ingress-controller**
- **Helm**



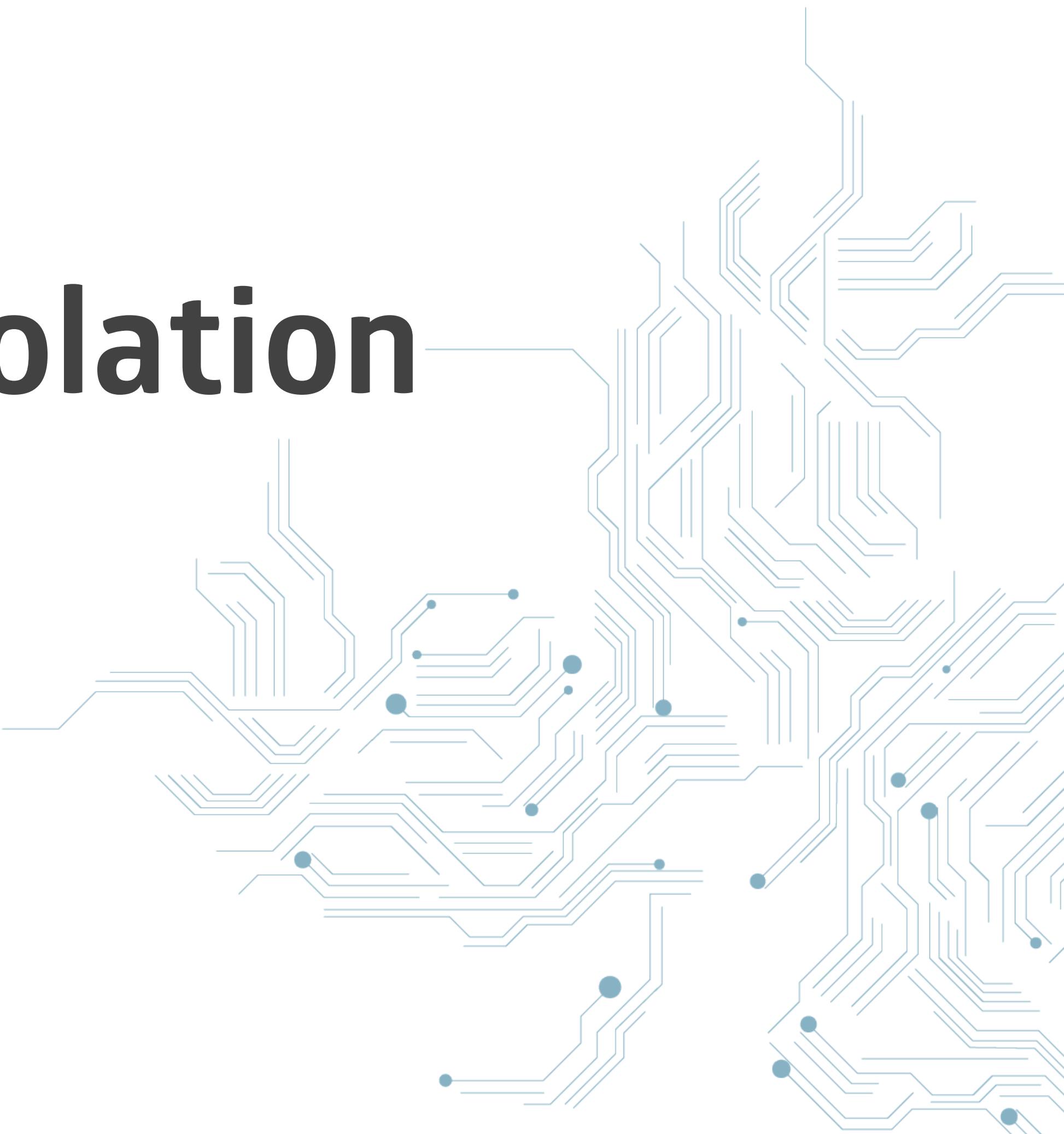
But first



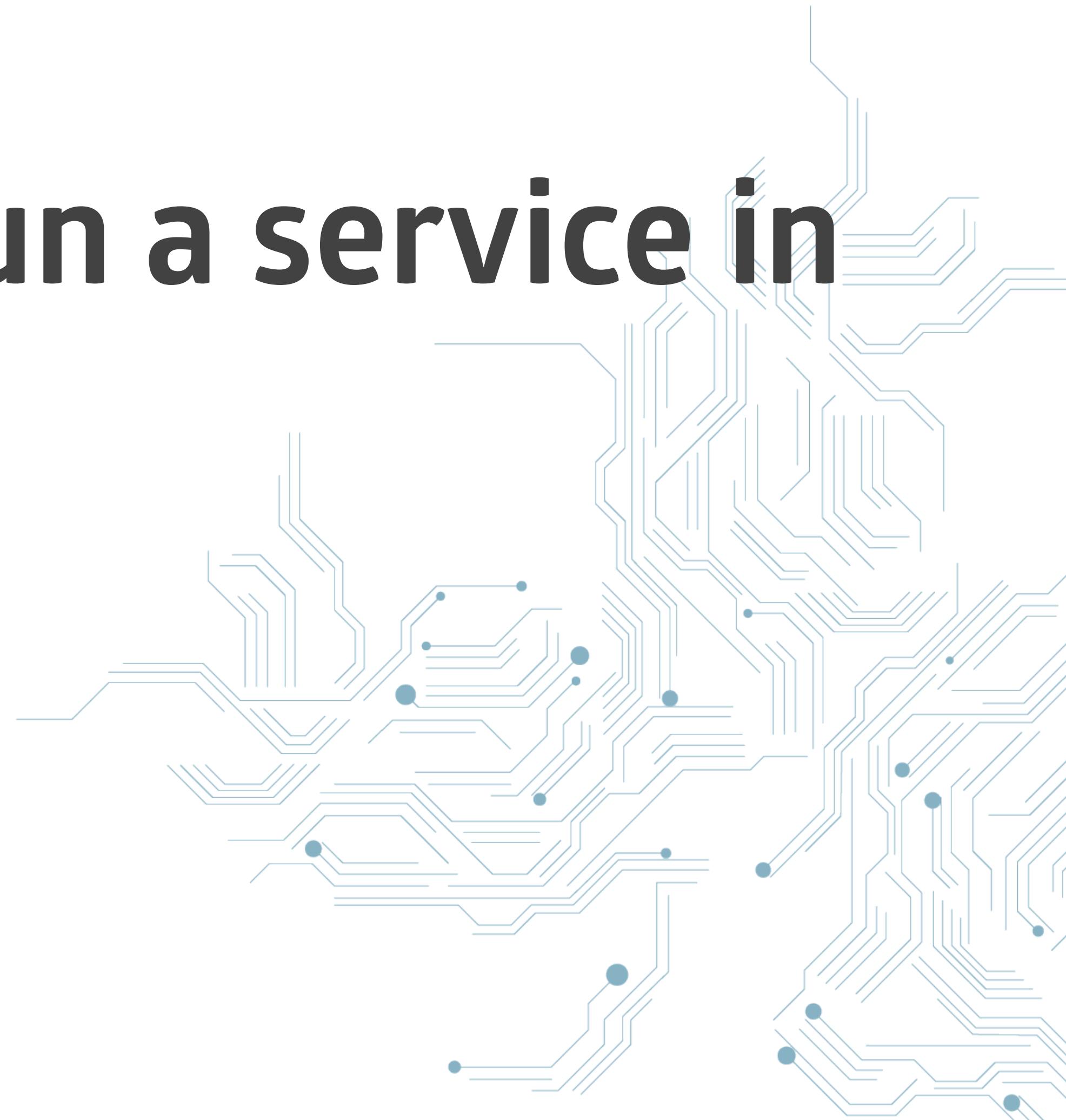
Why containers?



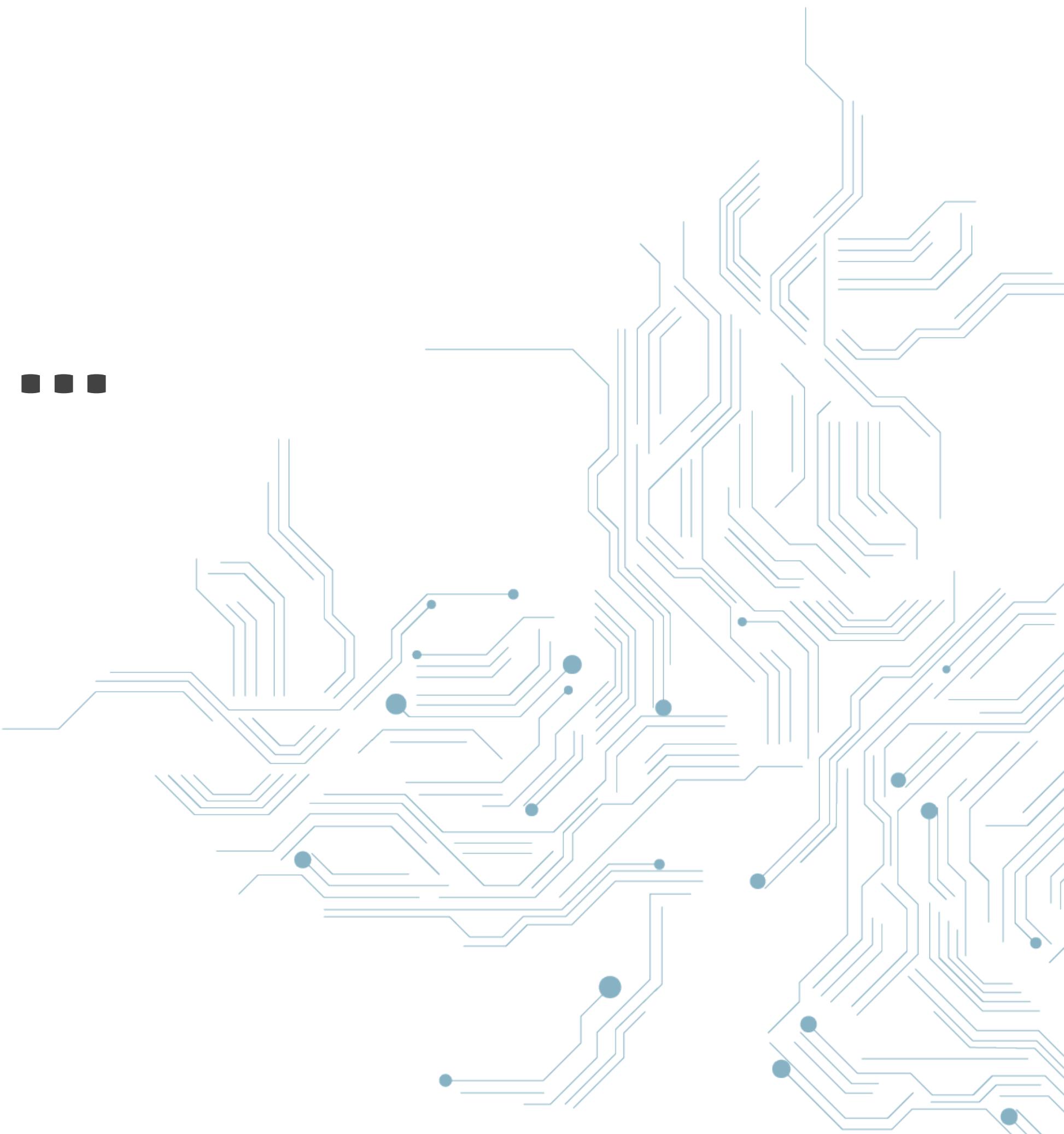
Services run in isolation



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



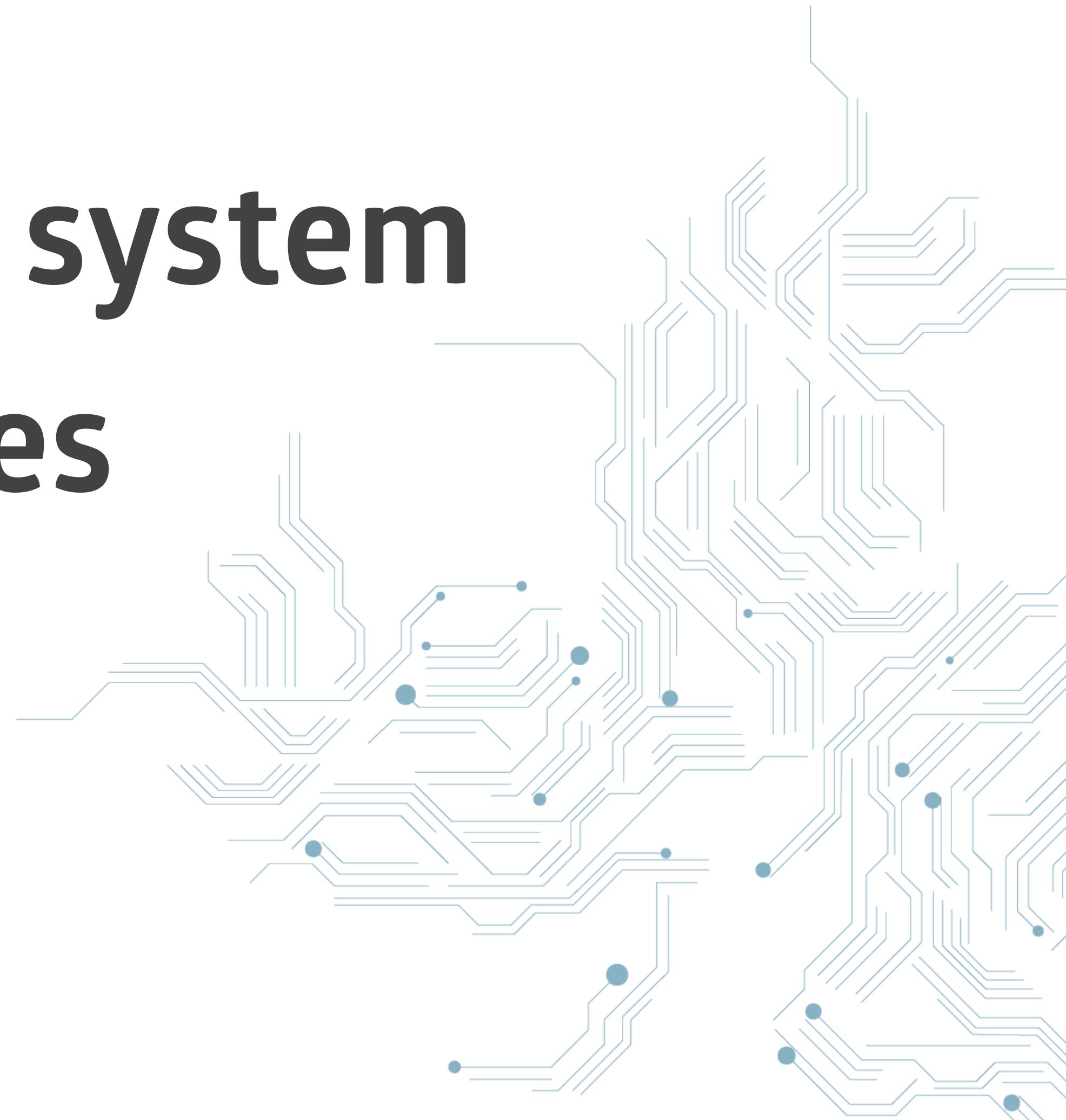
Make things ...



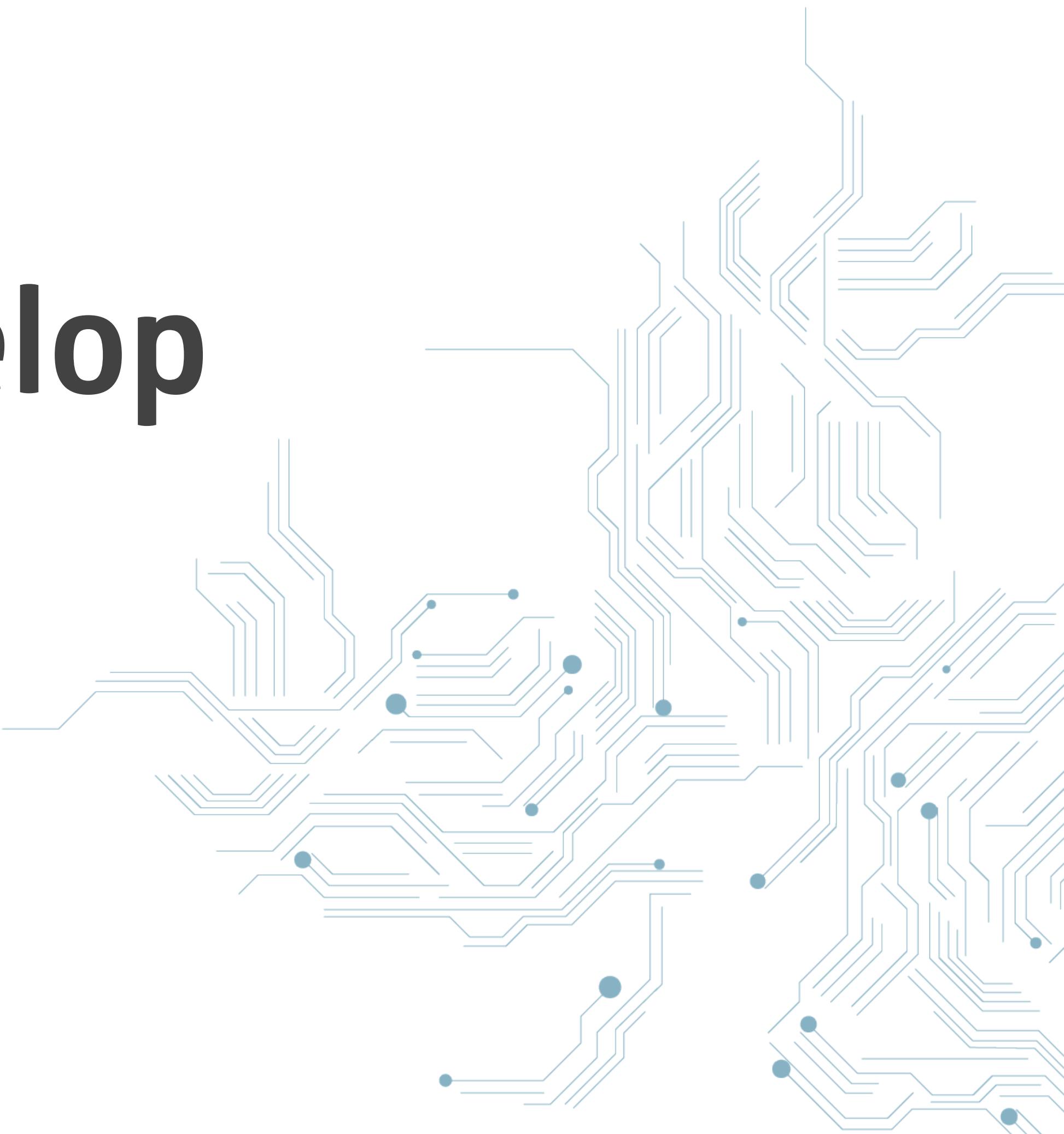
Easier to deploy



Easier to upgrade system dependencies



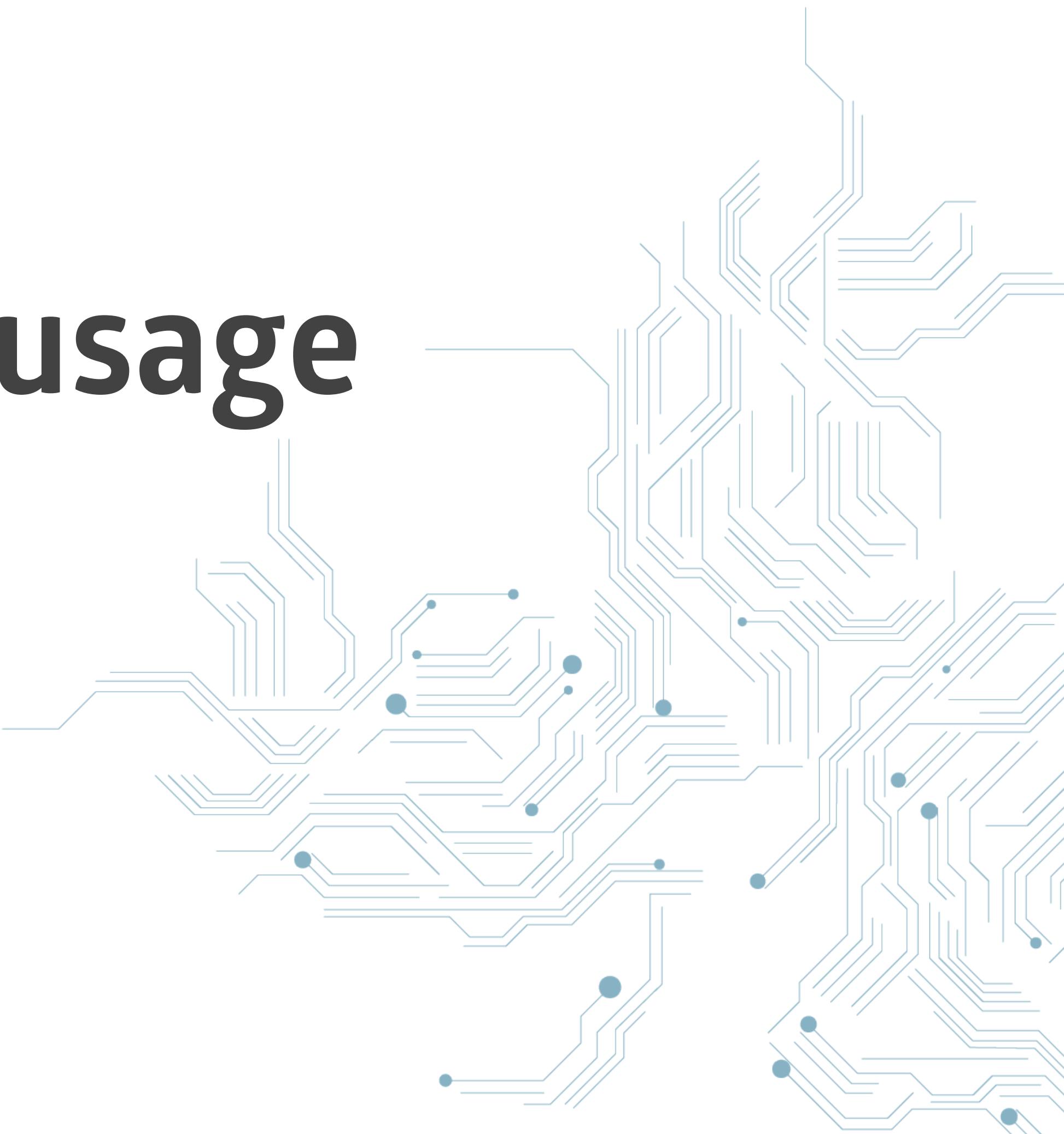
Easier to develop

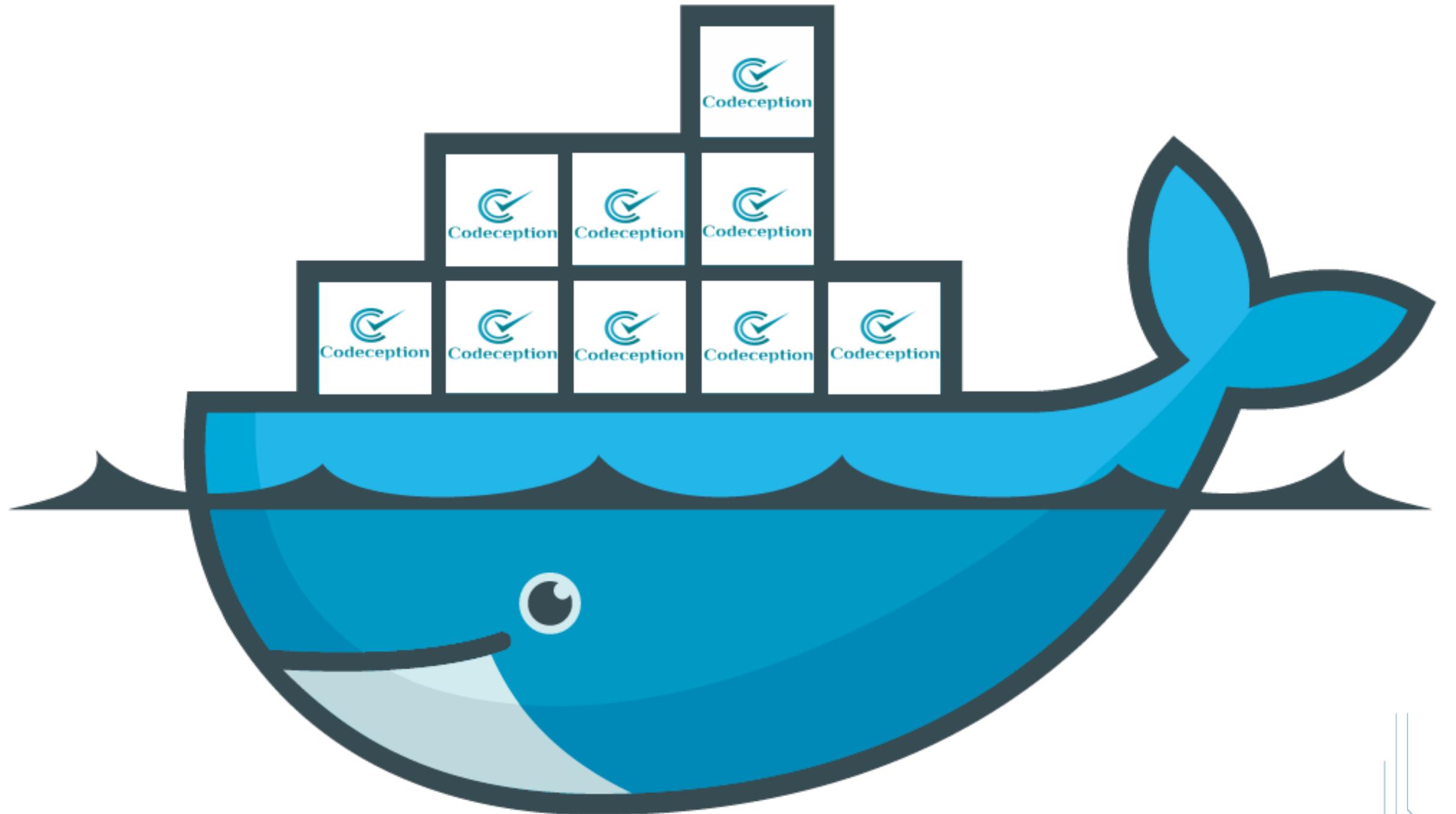


Easier to scale

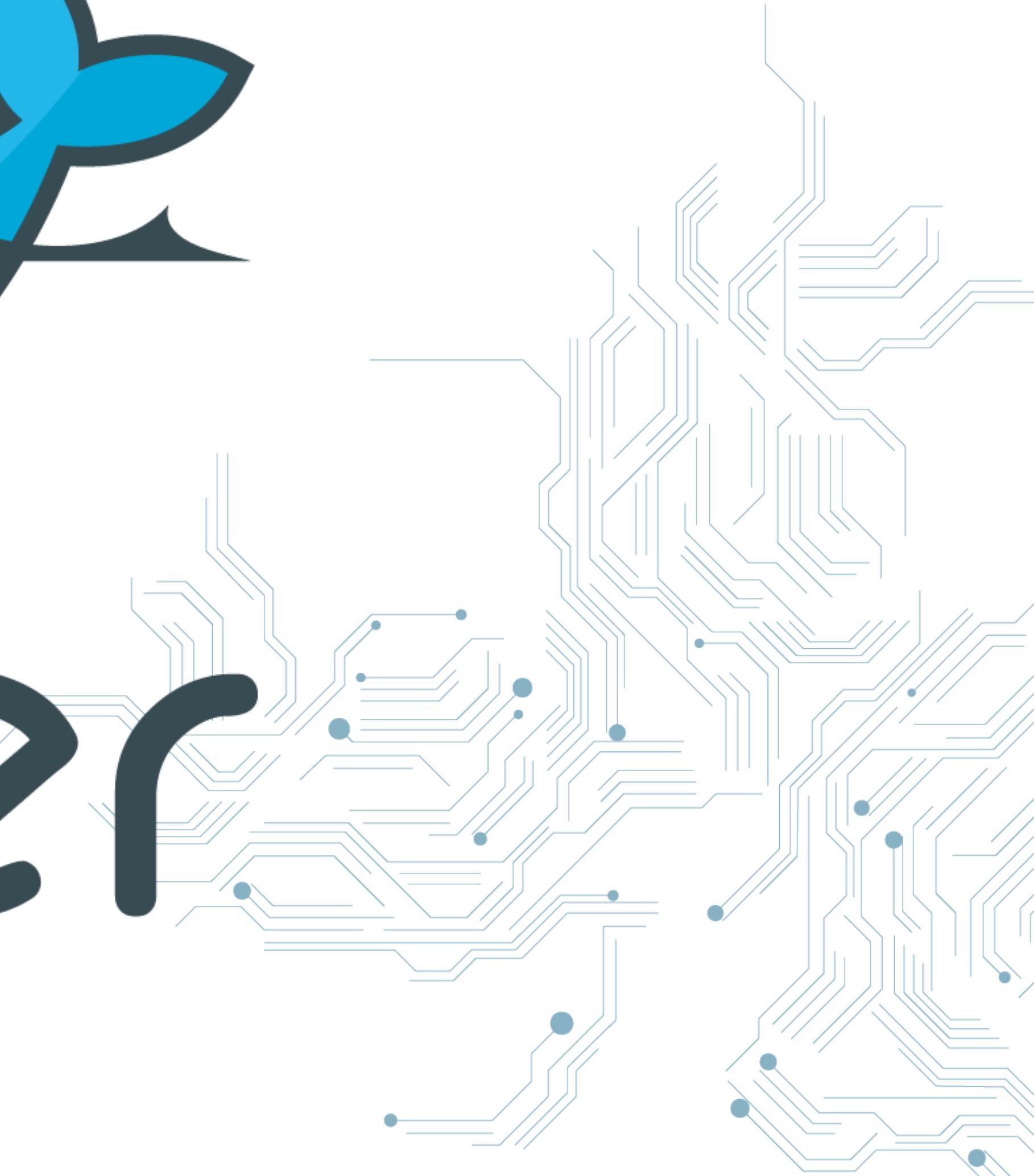


Better resource usage





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

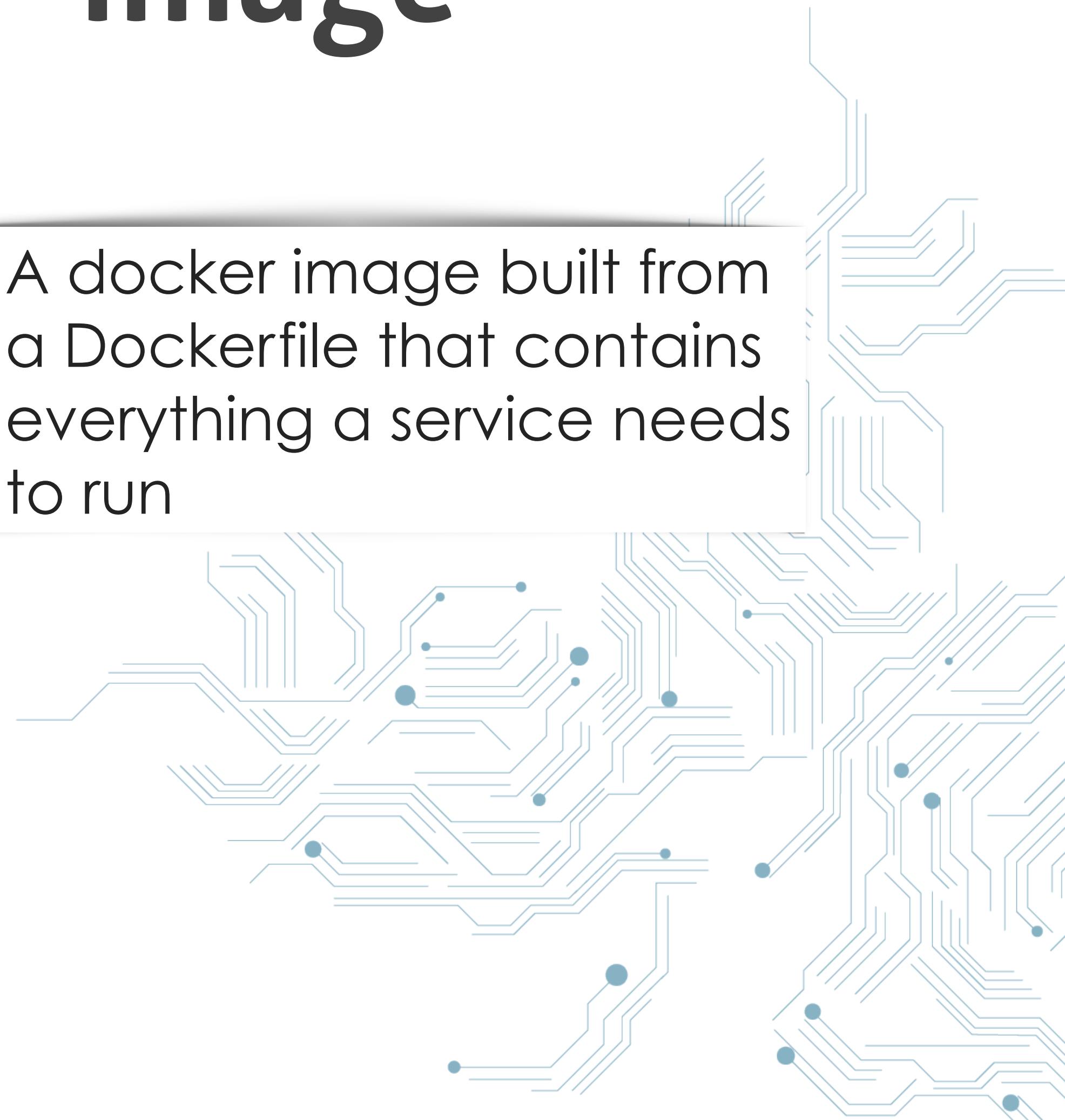


Kubernetes Cluster



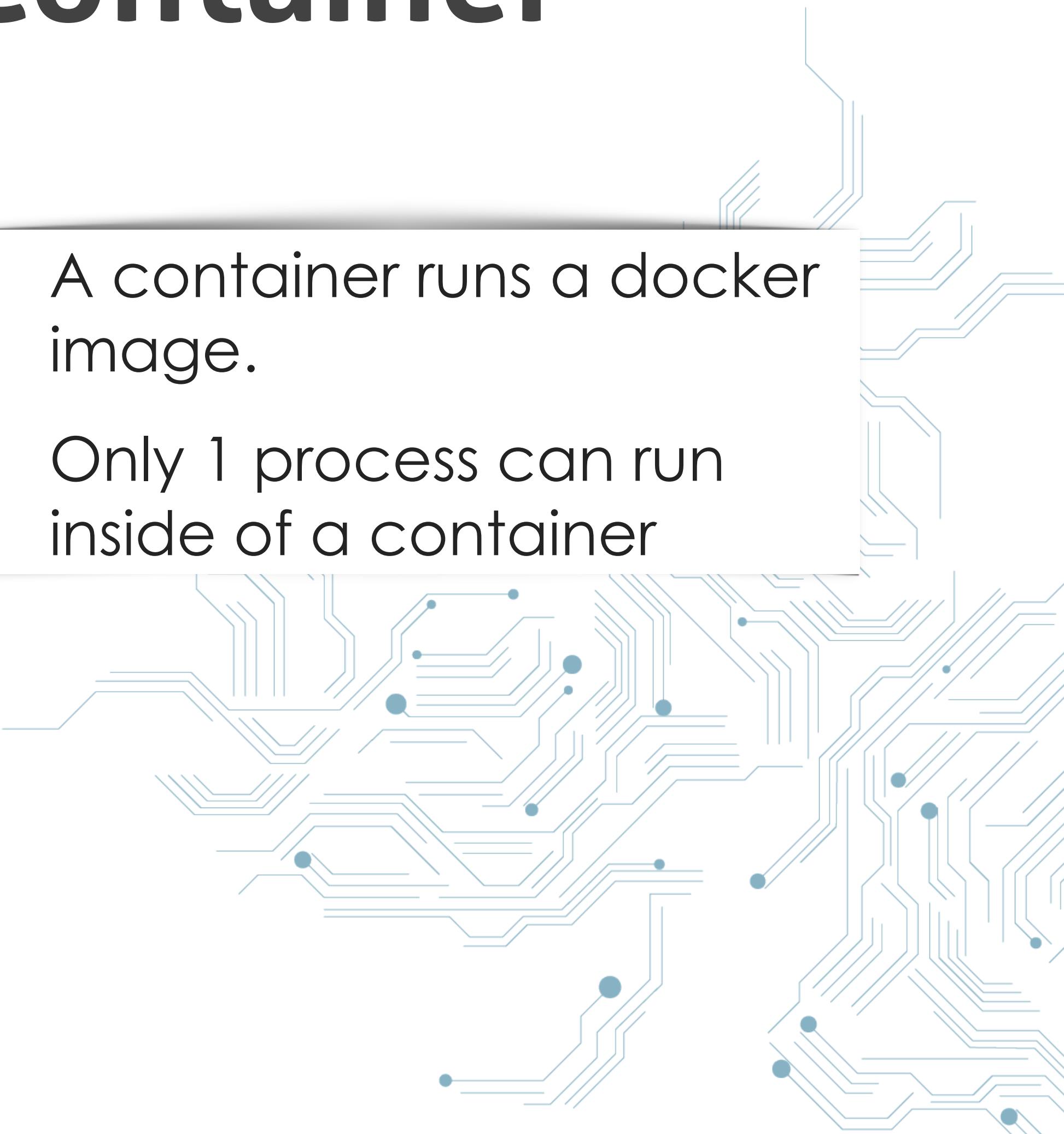
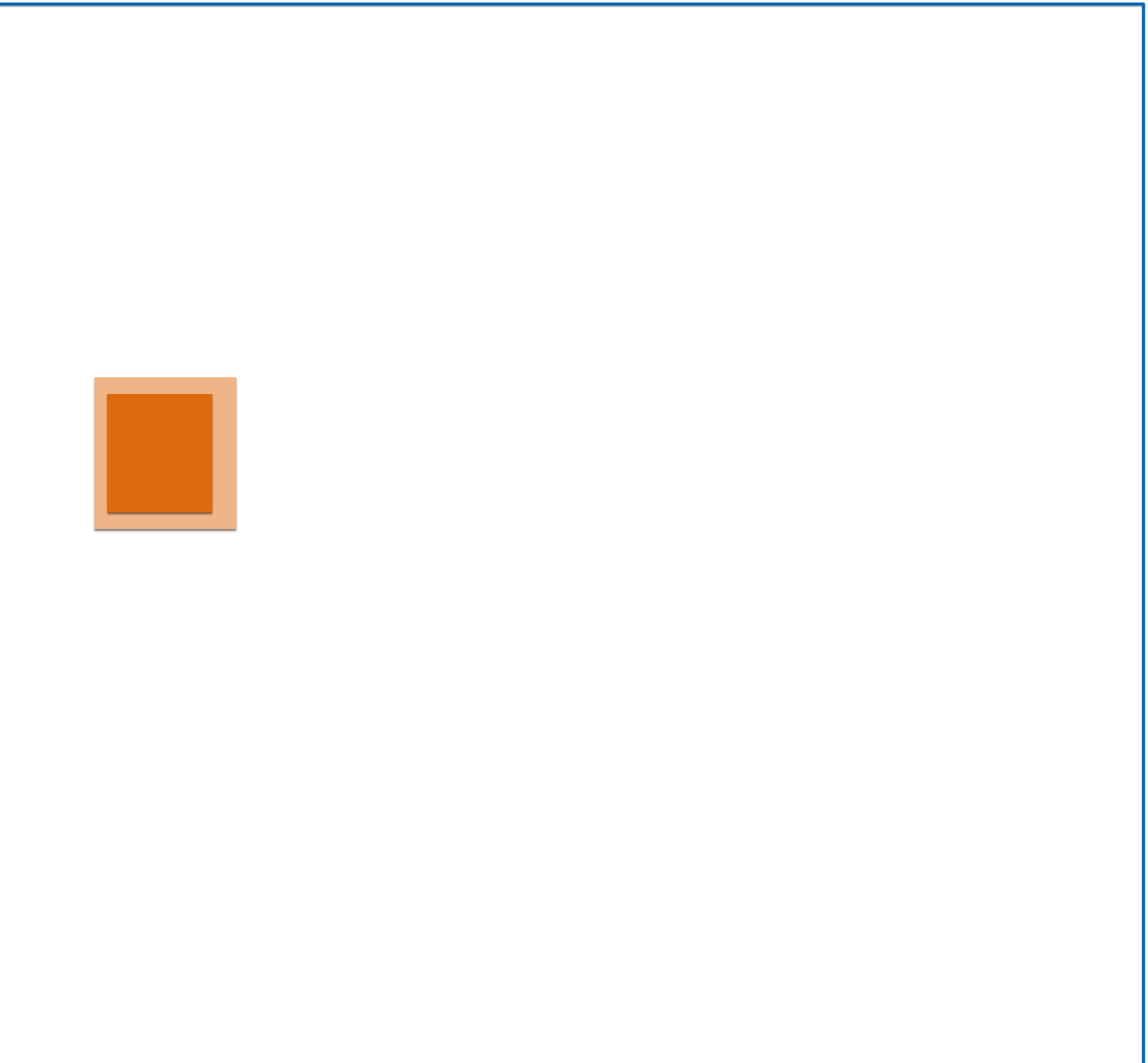
Image

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

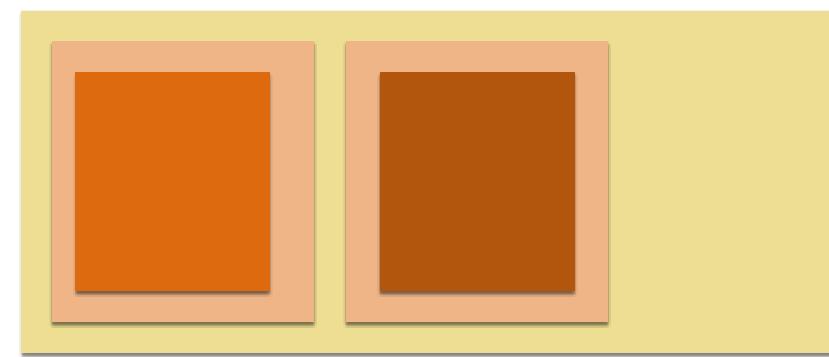


Container

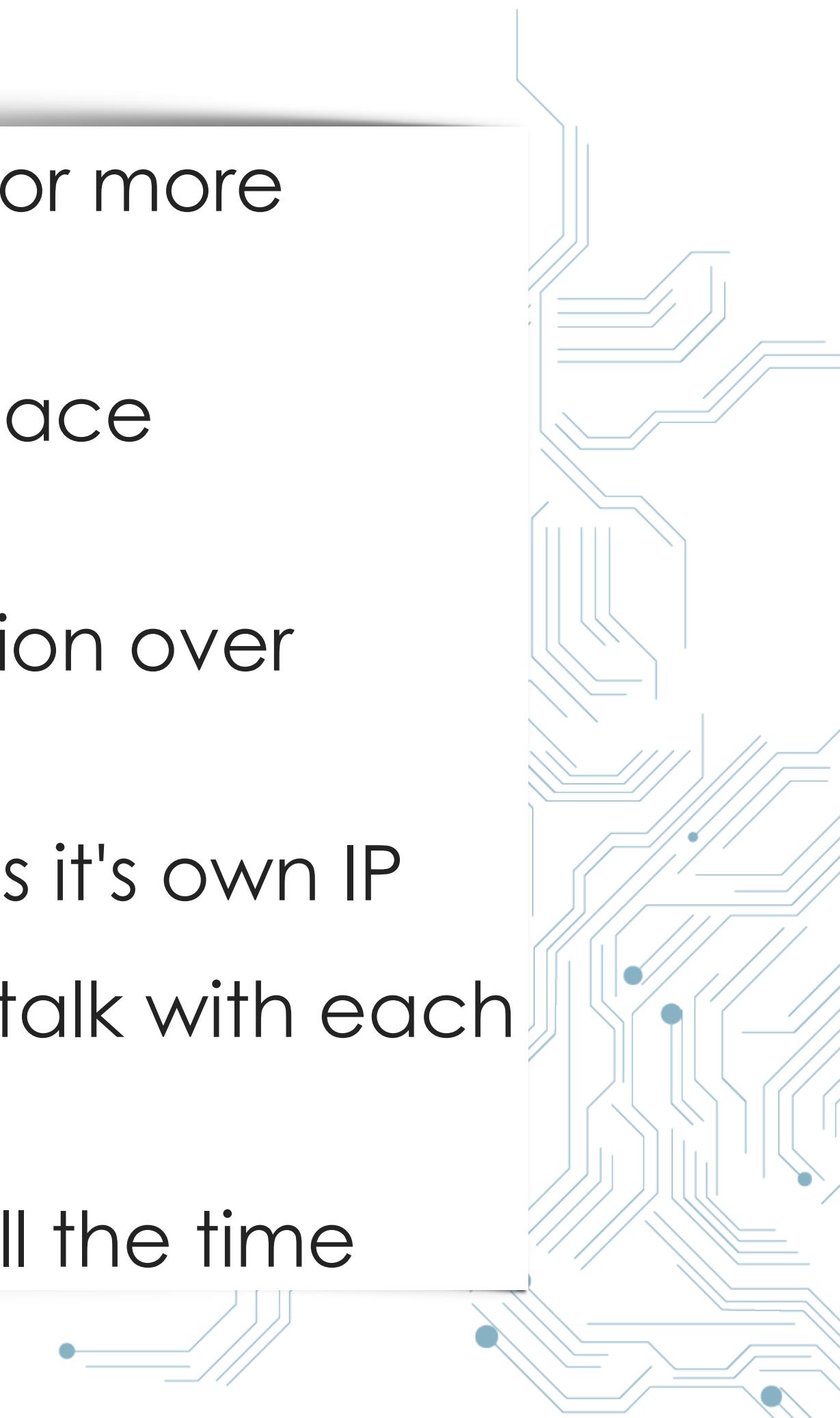
- A container runs a docker image.
- Only 1 process can run inside of a container



Pod

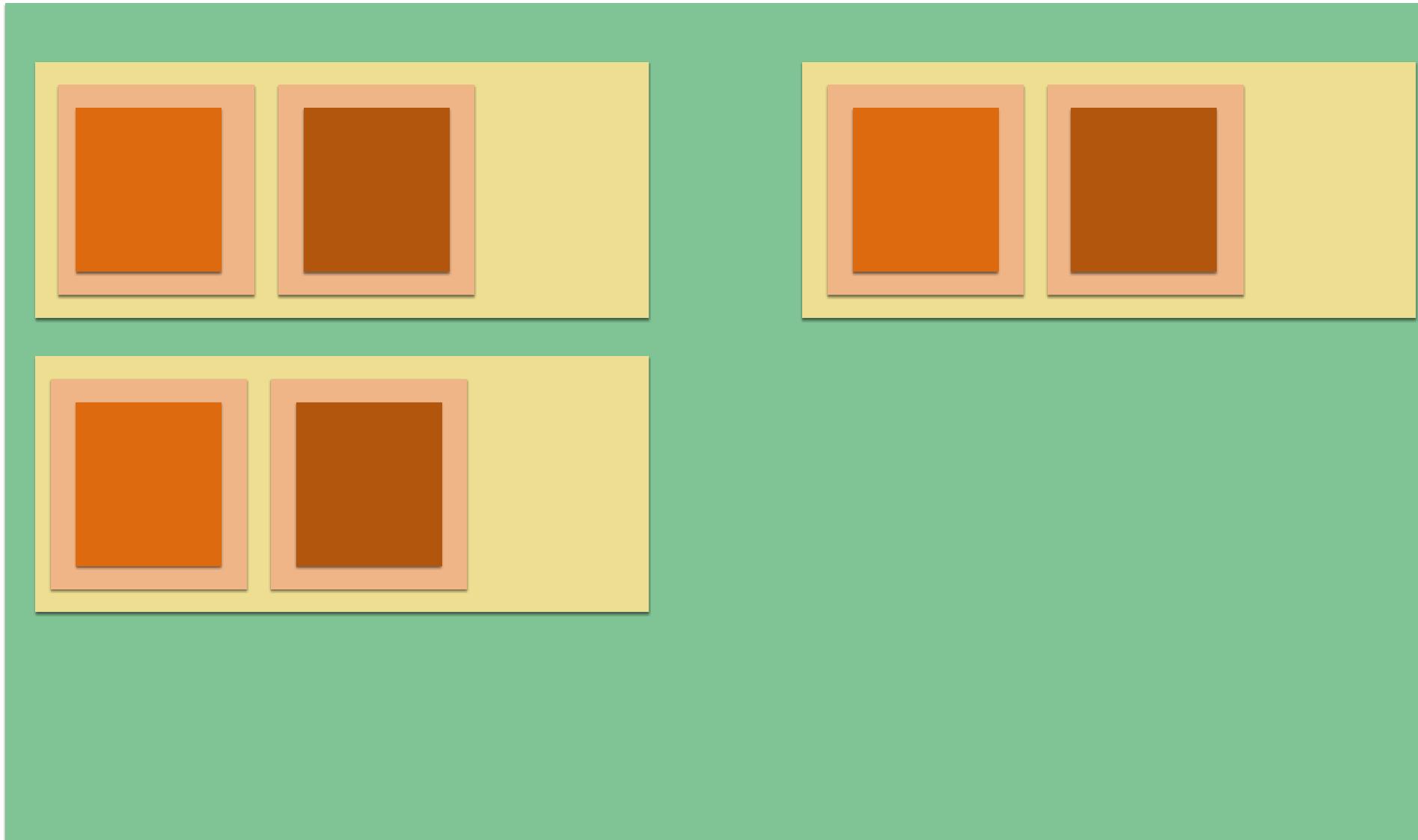


- A group of 1 or more containers
- Same port space
- Within a Pod:
communication over localhost
- Every Pod has it's own IP
- All Pods can talk with each other
- IPs change all the time



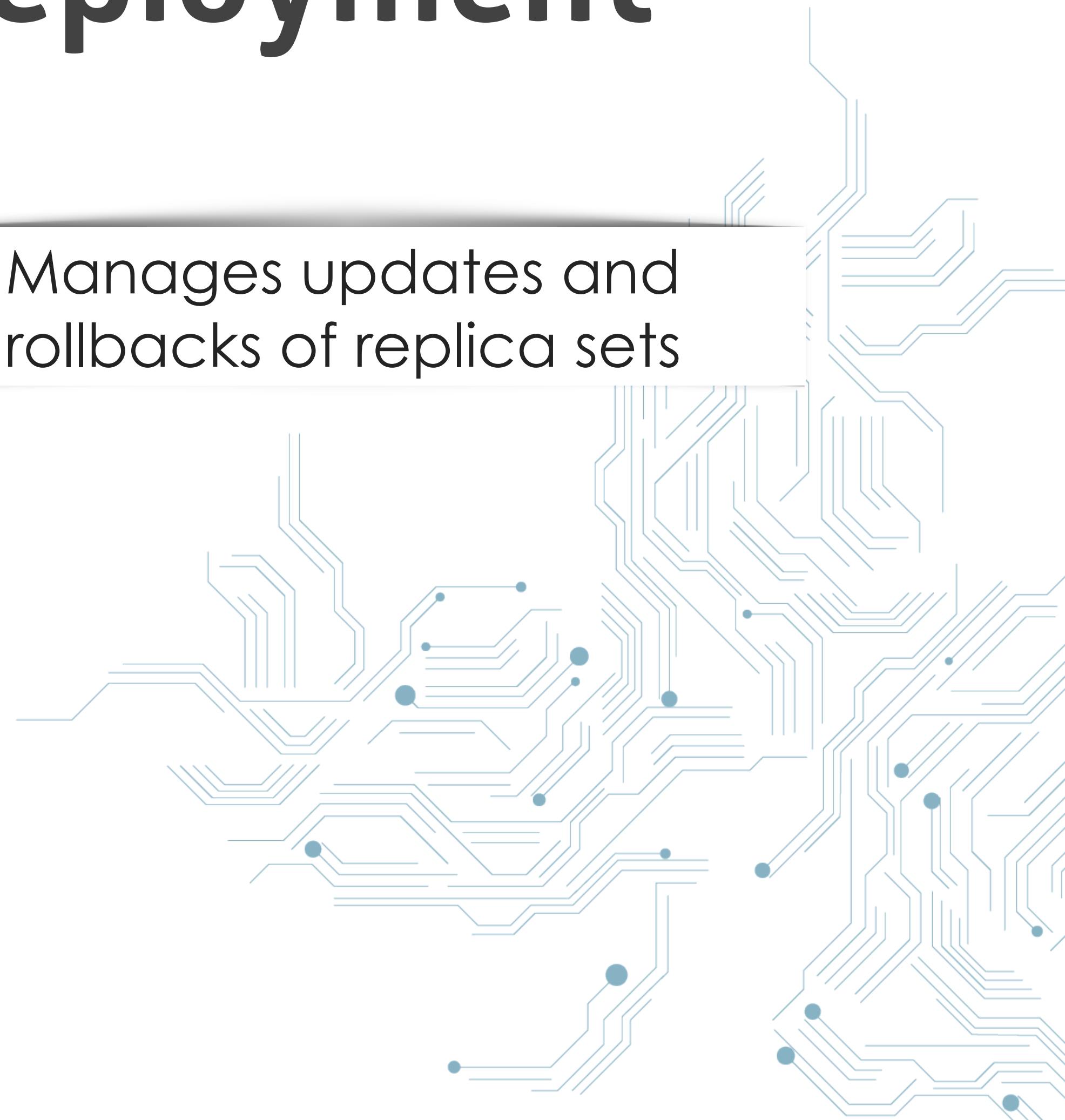
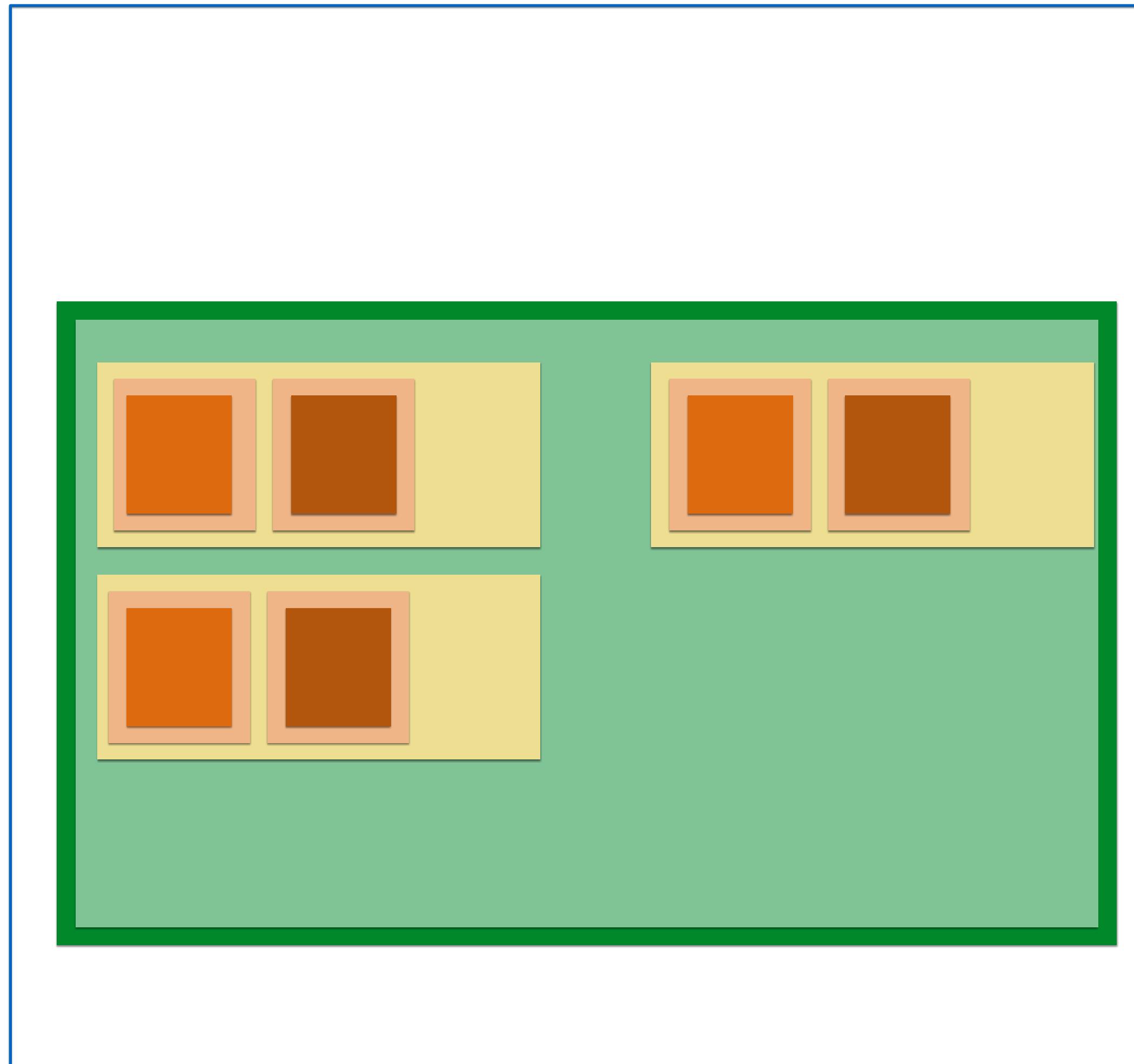
Replica Set

- 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

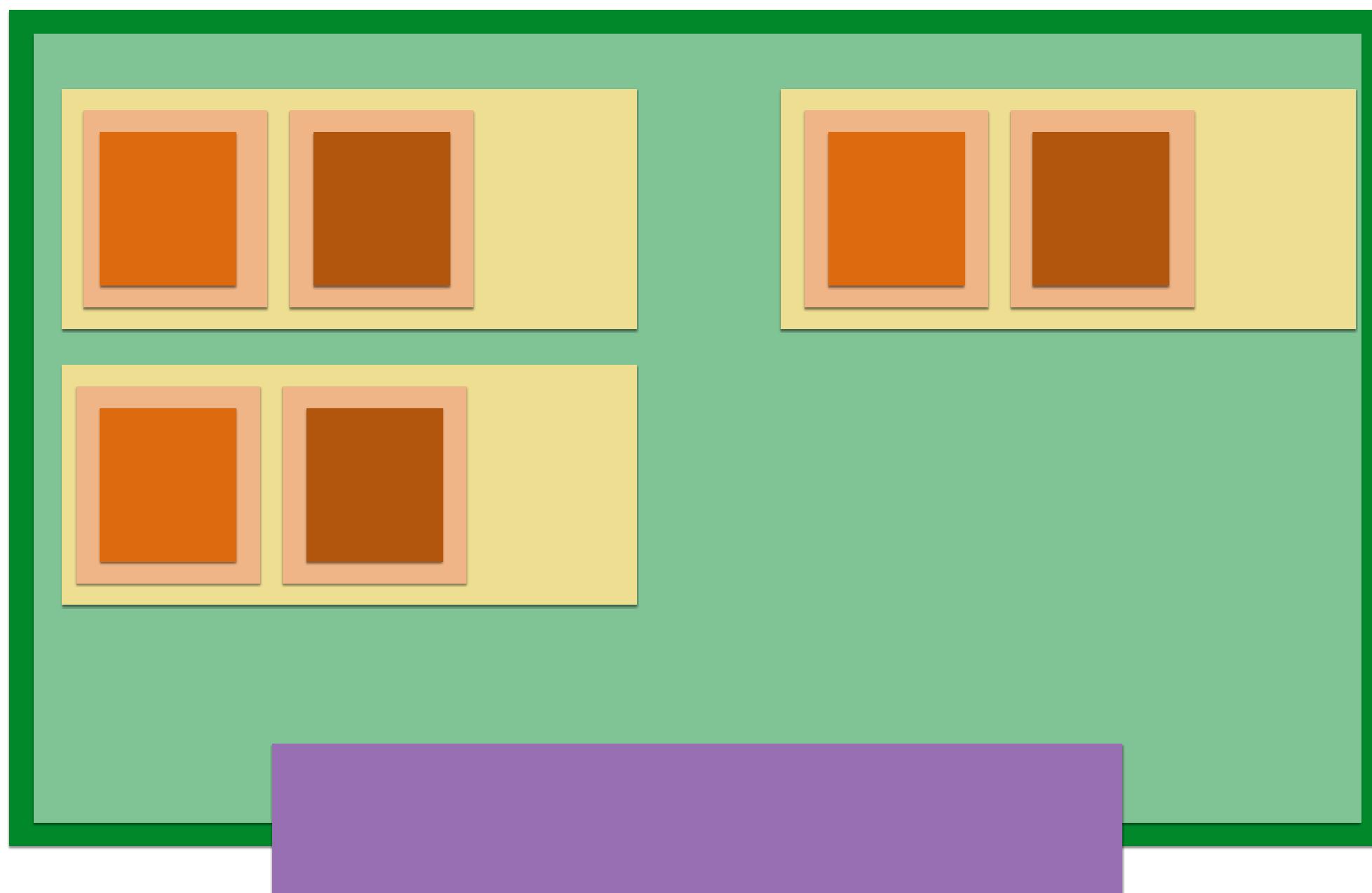


Deployment

- Manages updates and rollbacks of replica sets

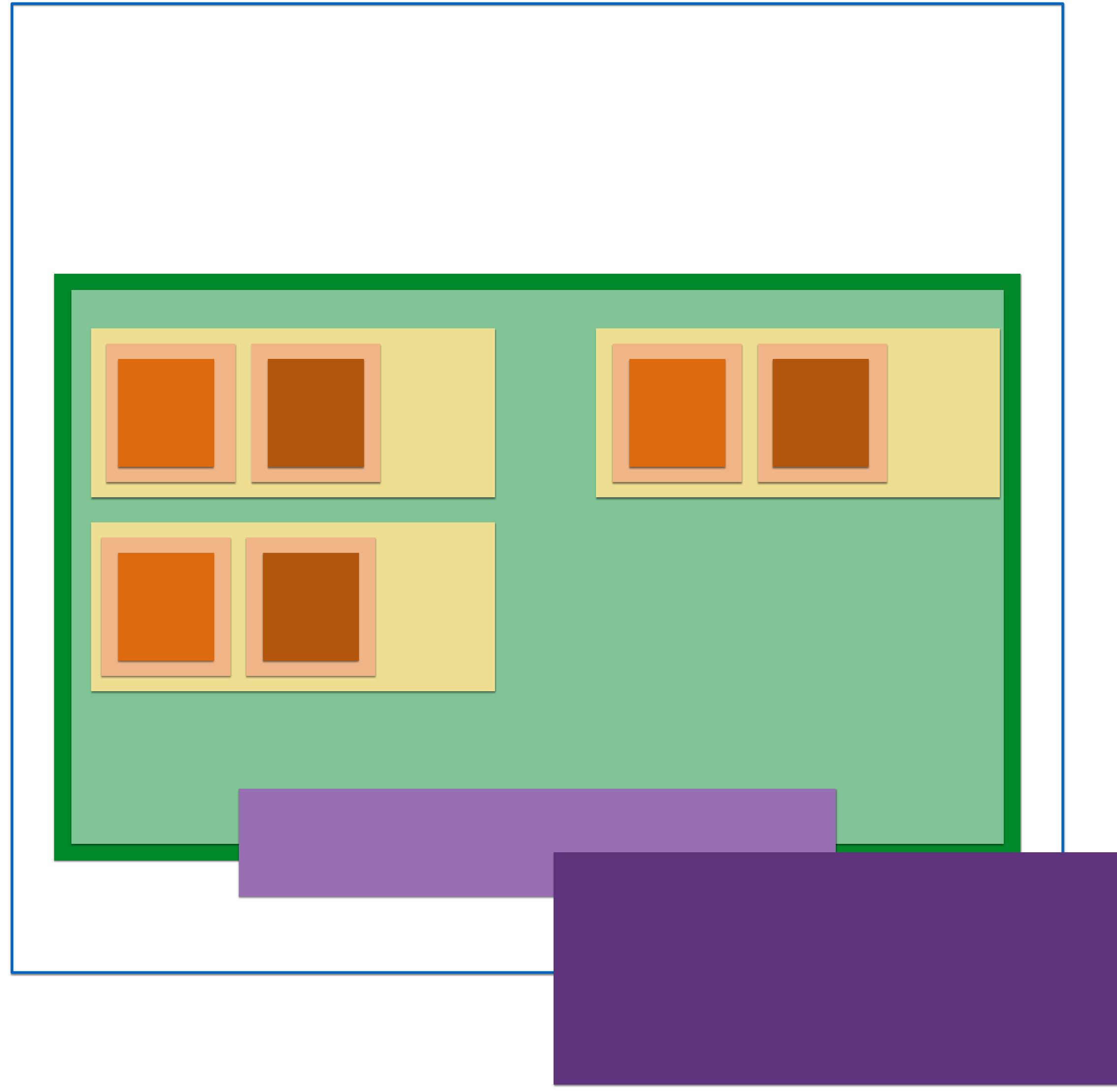


Service

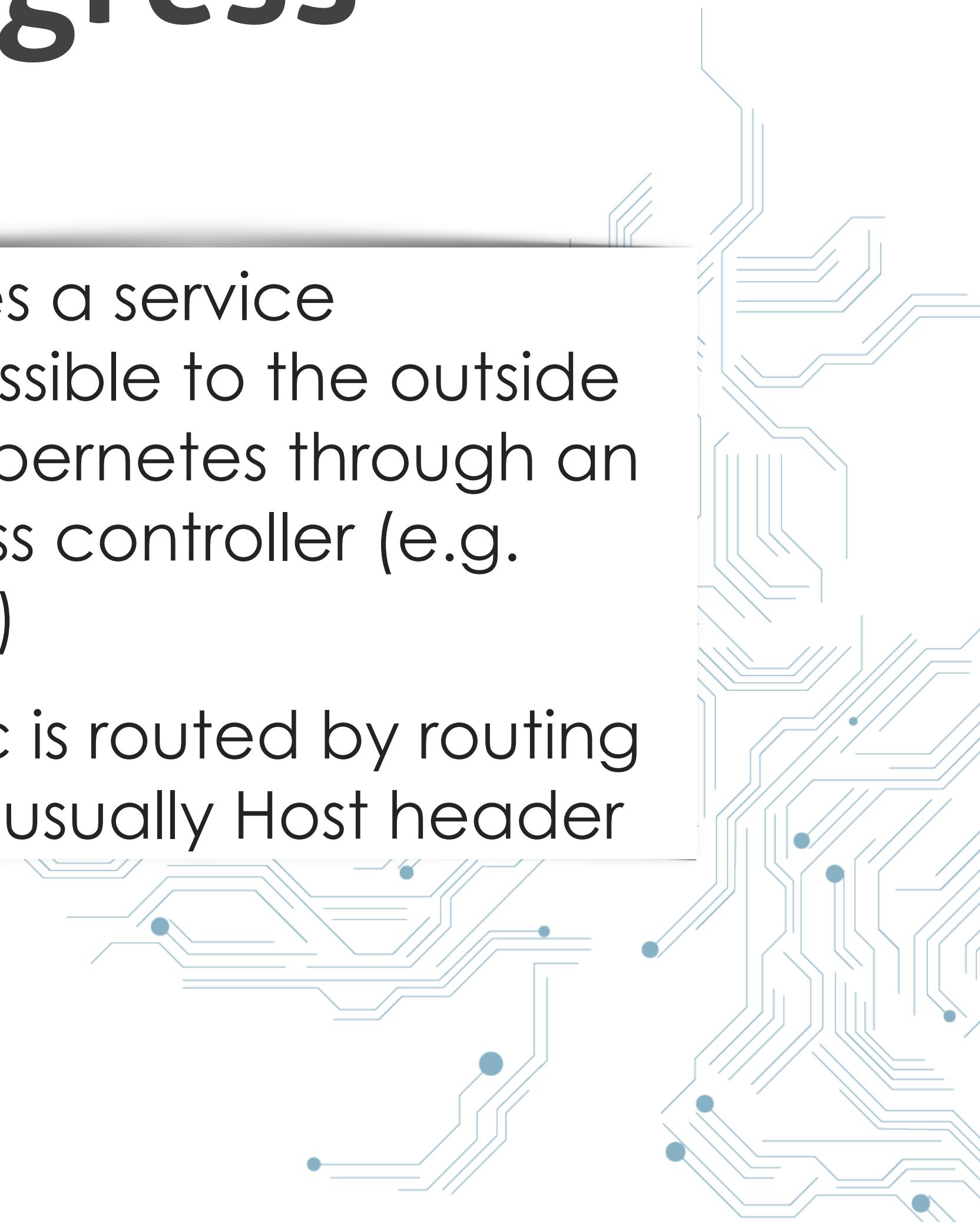


- Internal LoadBalancer
- Makes all pods matching a set of labels accessible through a stable, internal IP address
- You can attach external IP address through an cloud LoadBalancer

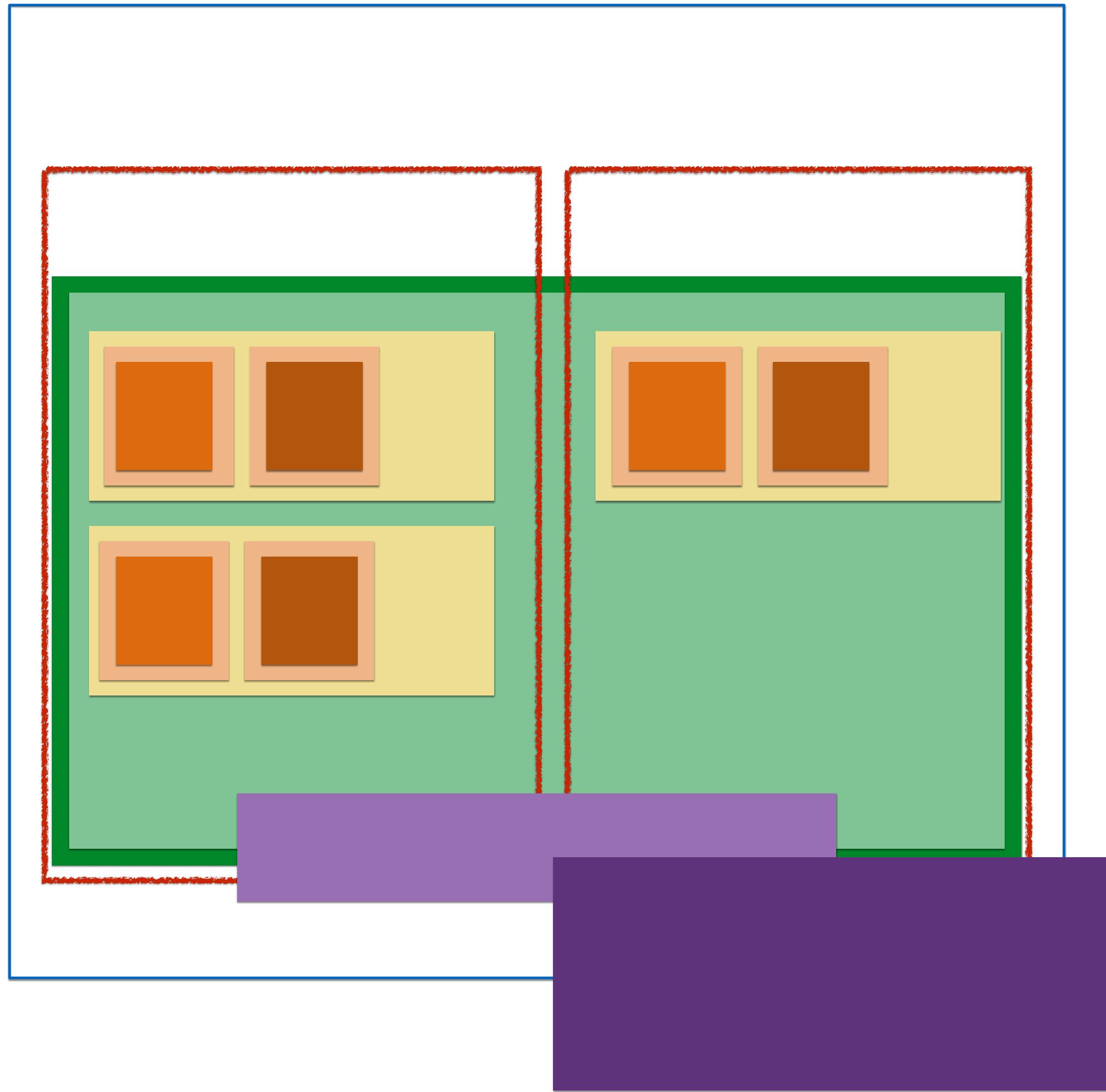
Ingress



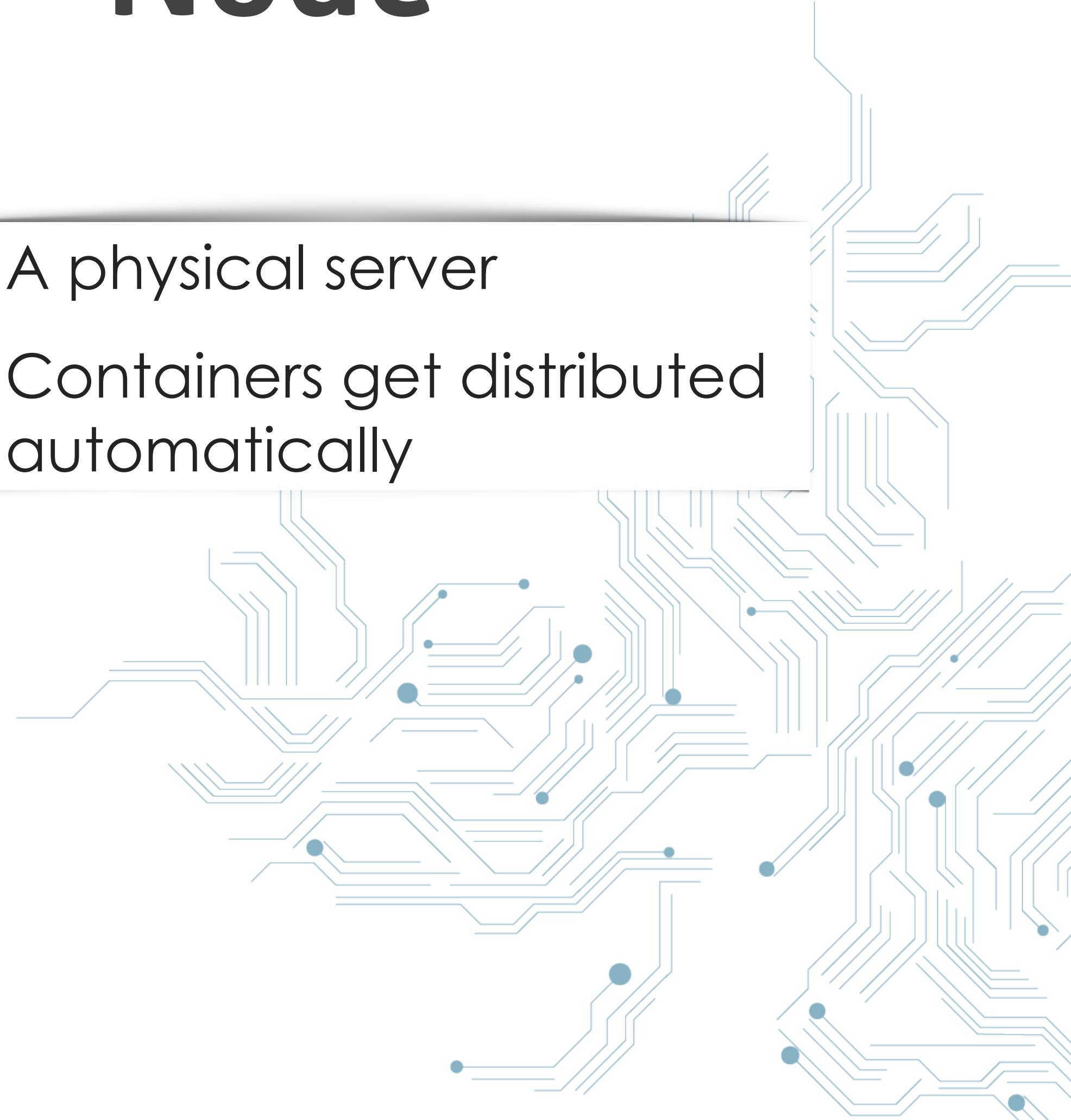
- Makes a service accessible to the outside of Kubernetes through an ingress controller (e.g. nginx)
- Traffic is routed by routing rules, usually Host header



Node

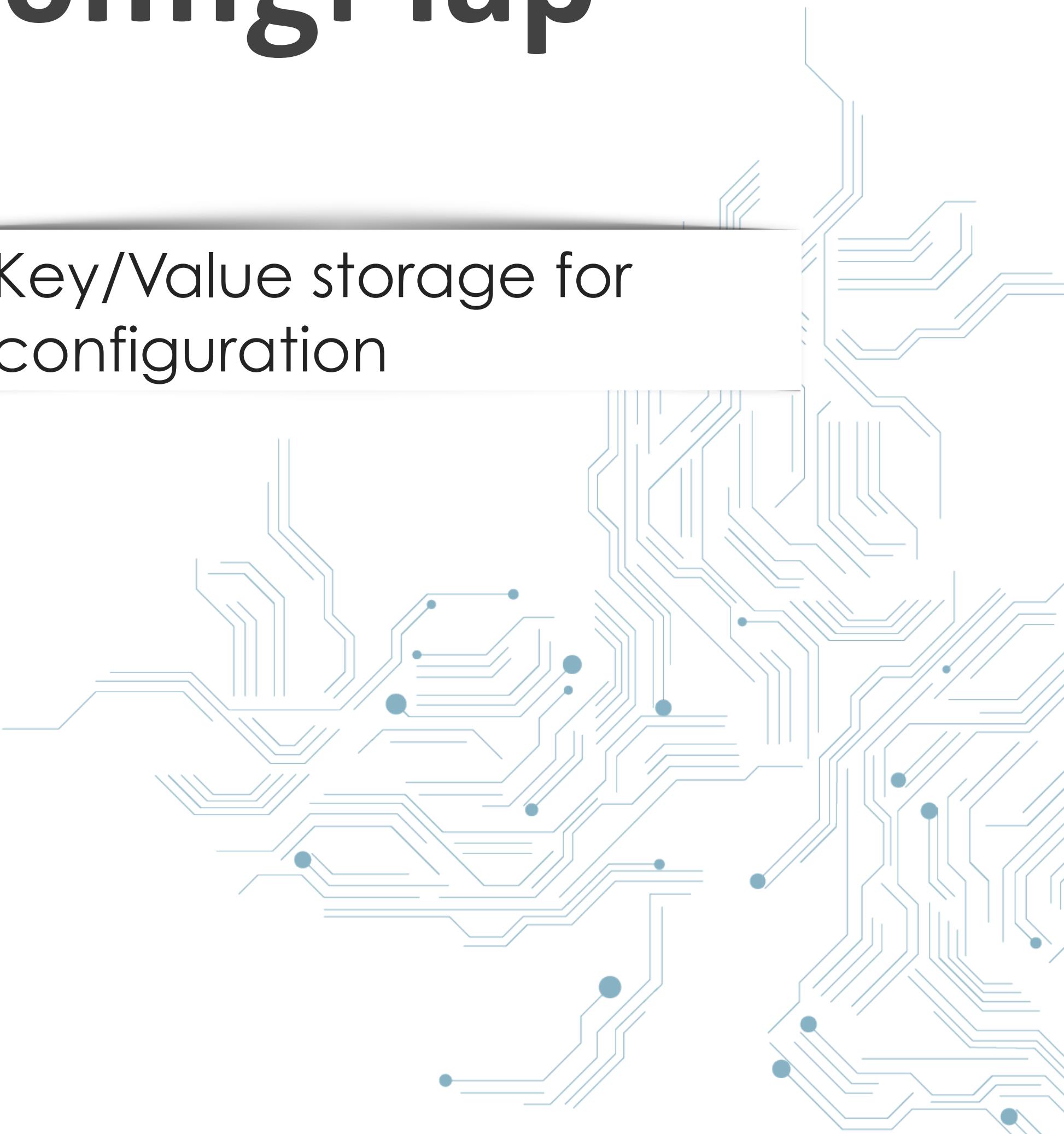
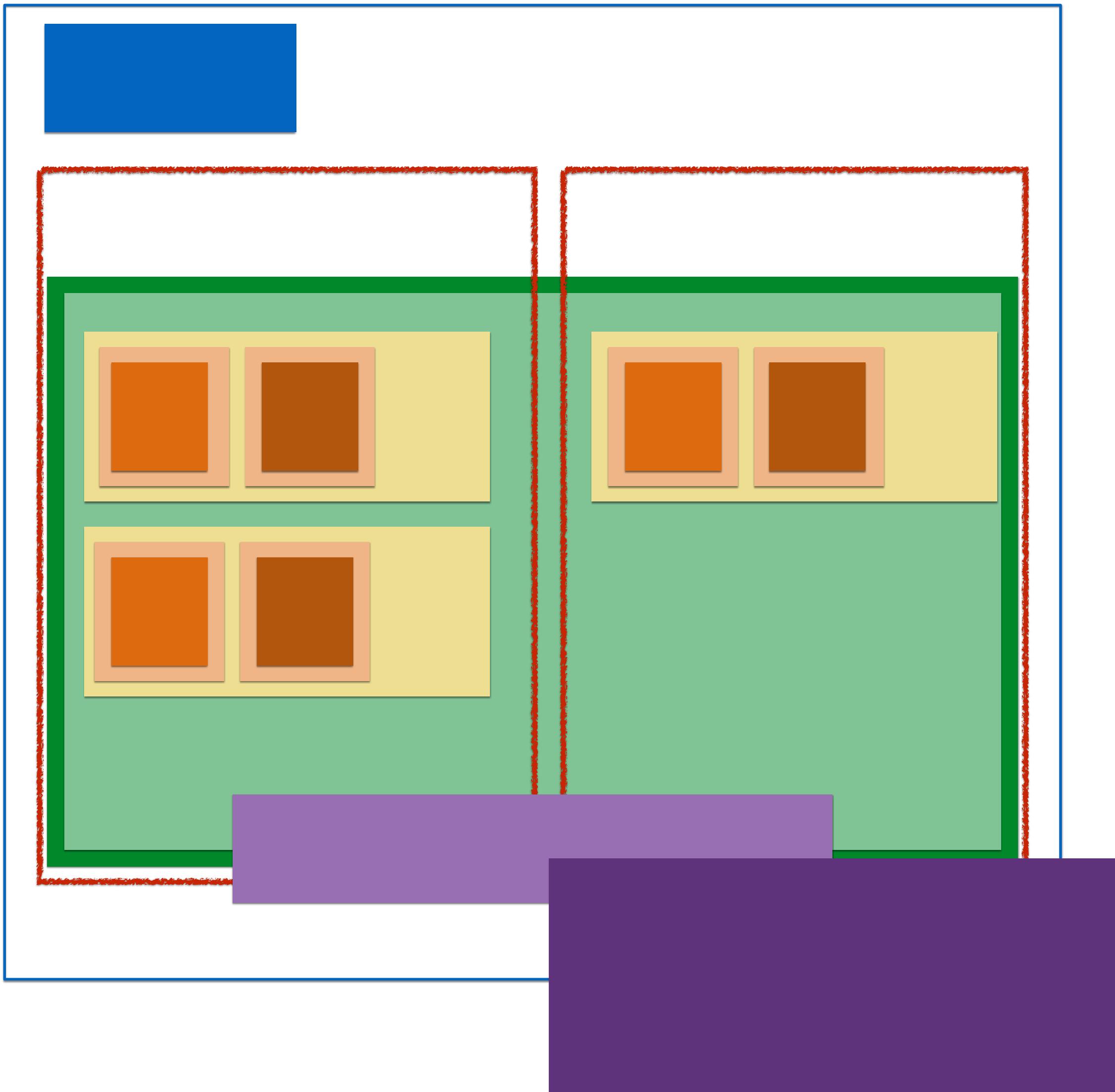


- A physical server
- Containers get distributed automatically



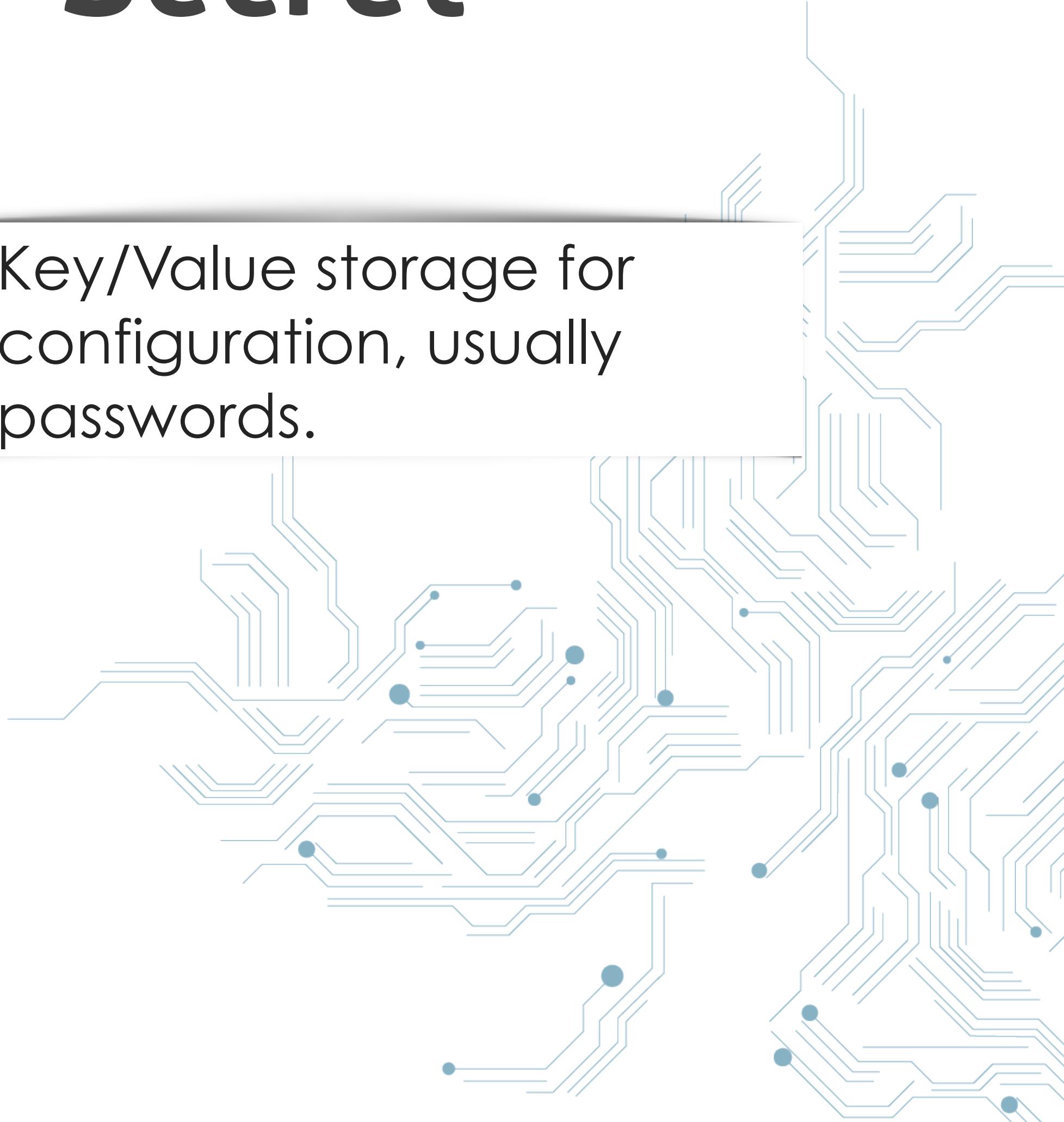
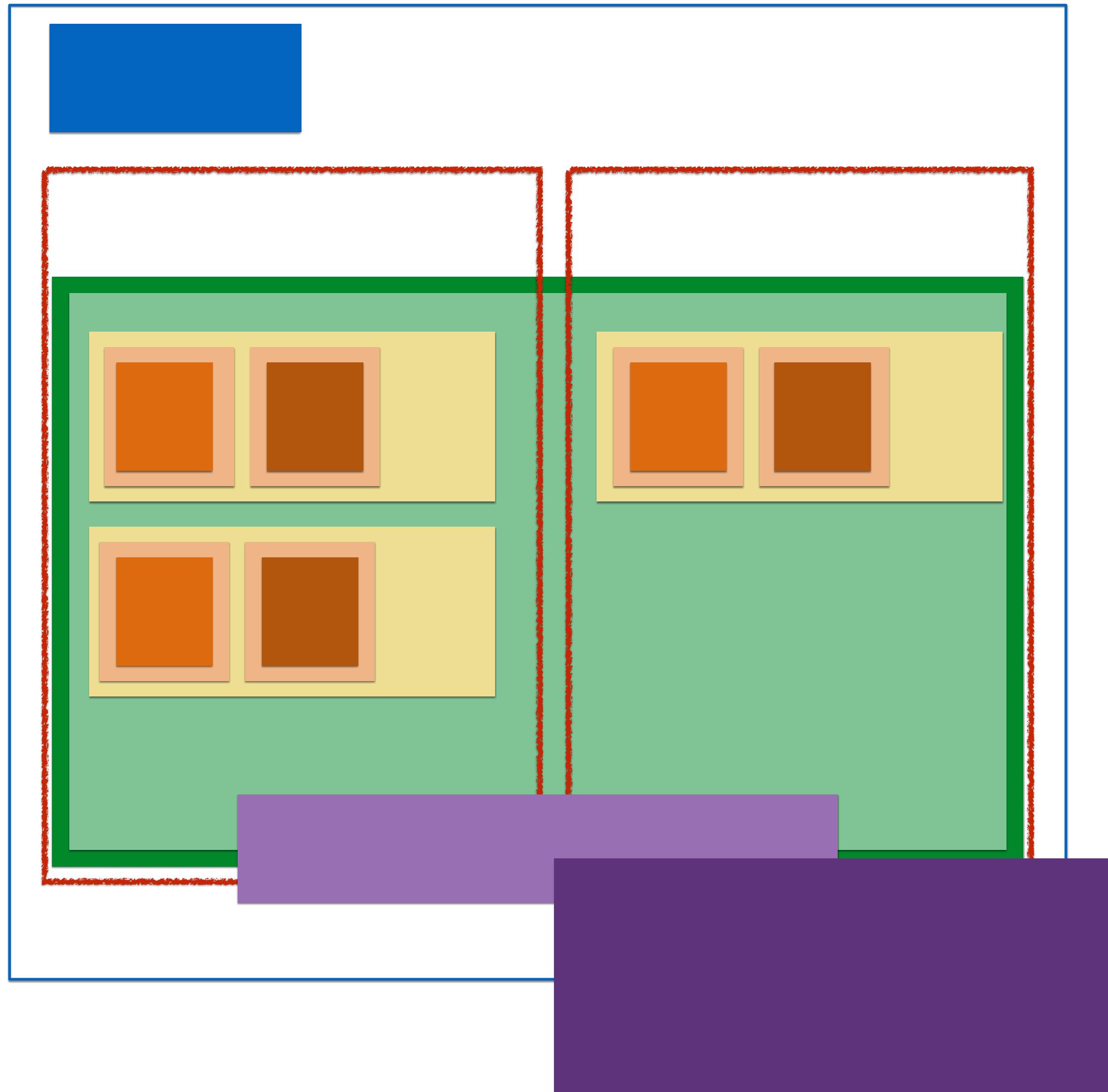
ConfigMap

- Key/Value storage for configuration

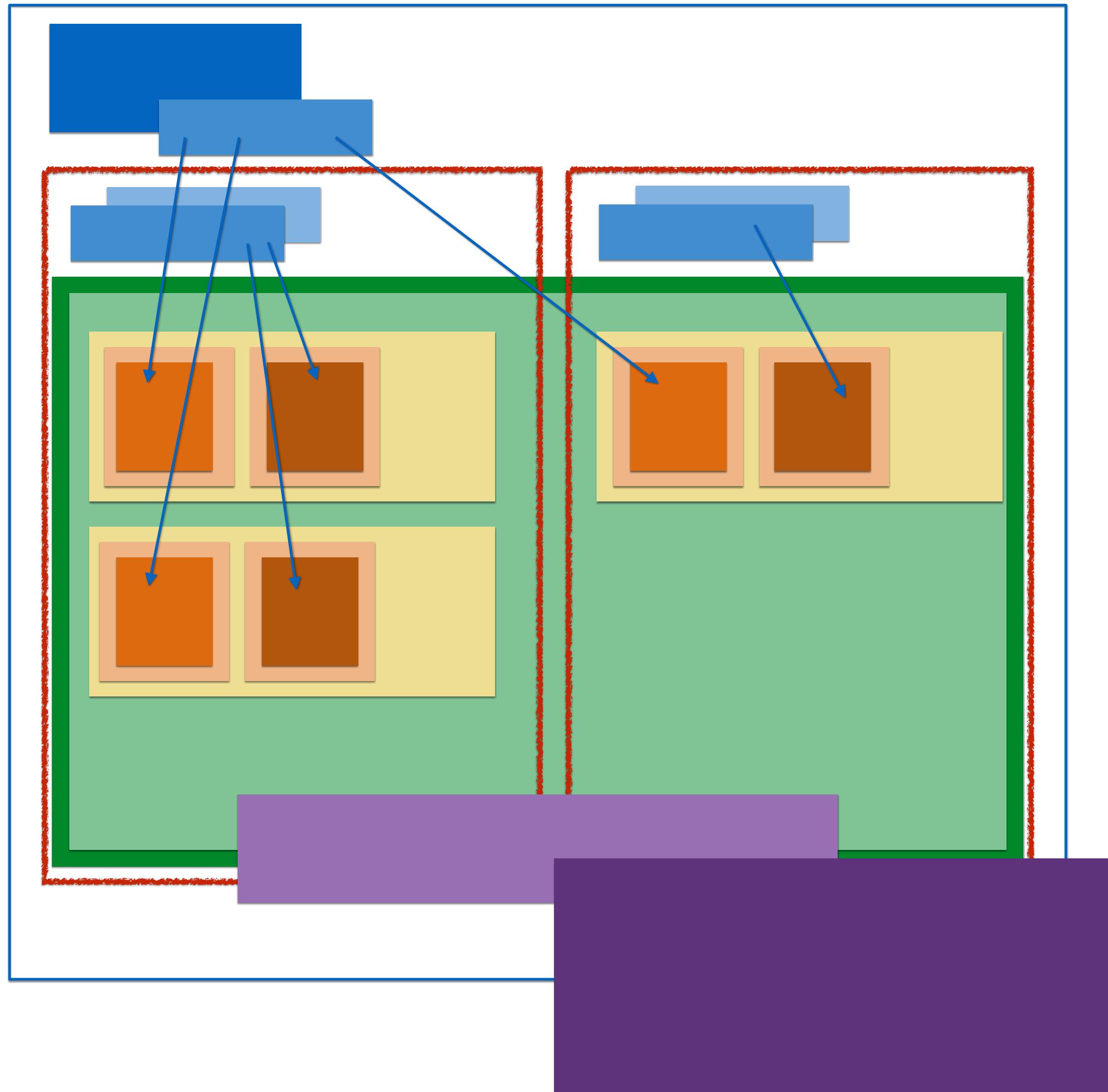


Secret

- Key/Value storage for configuration, usually passwords.



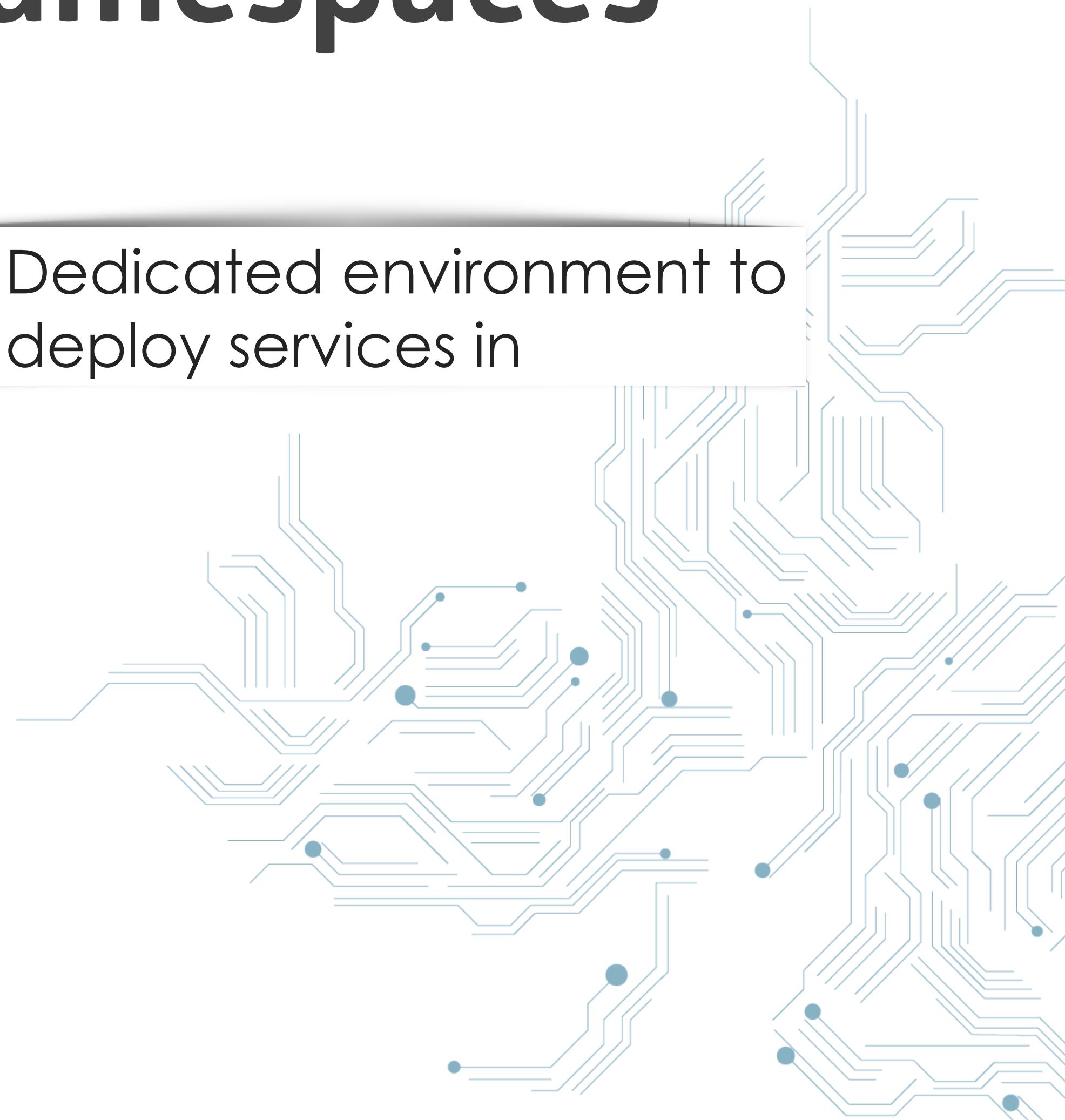
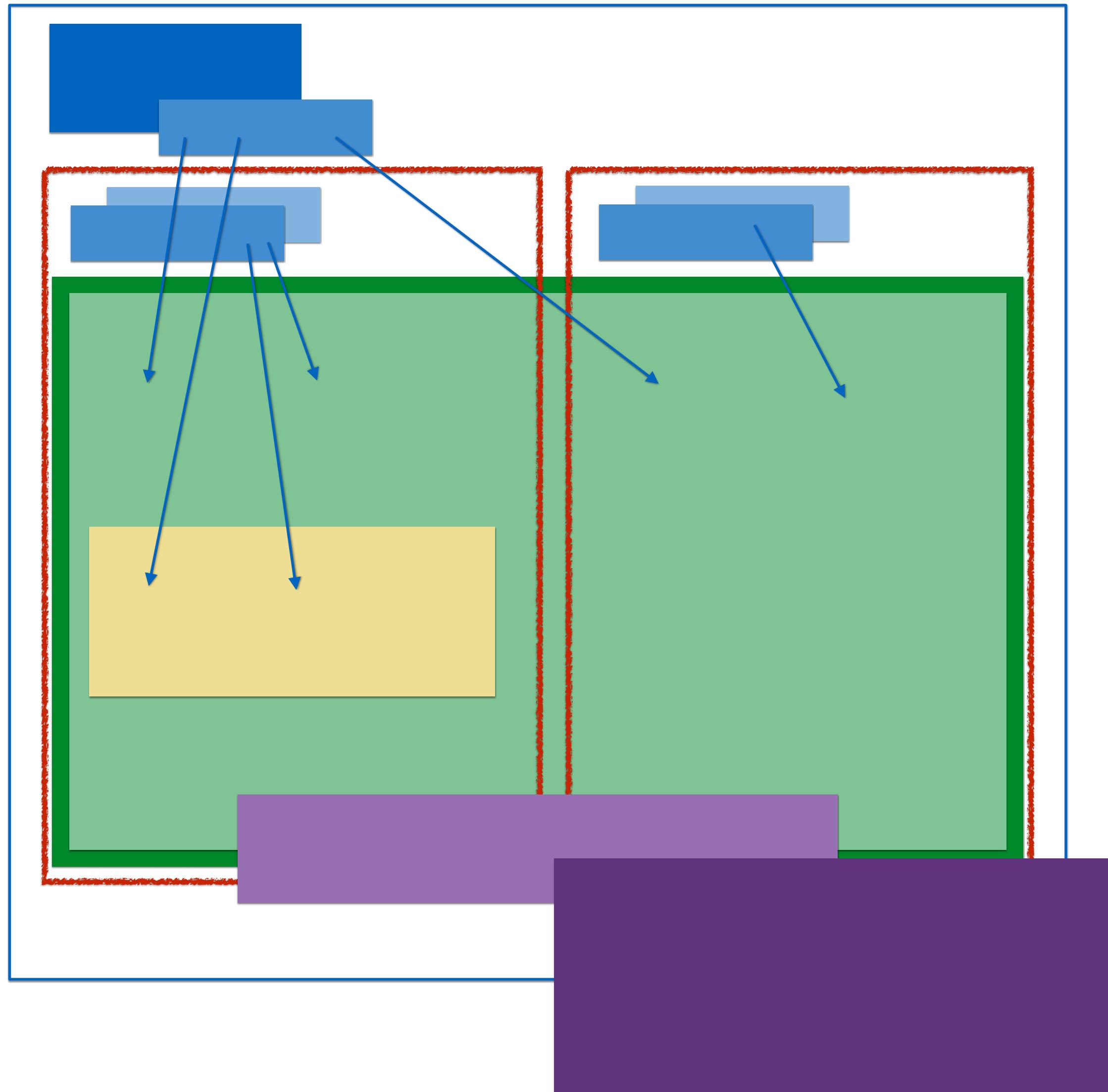
Volumes



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

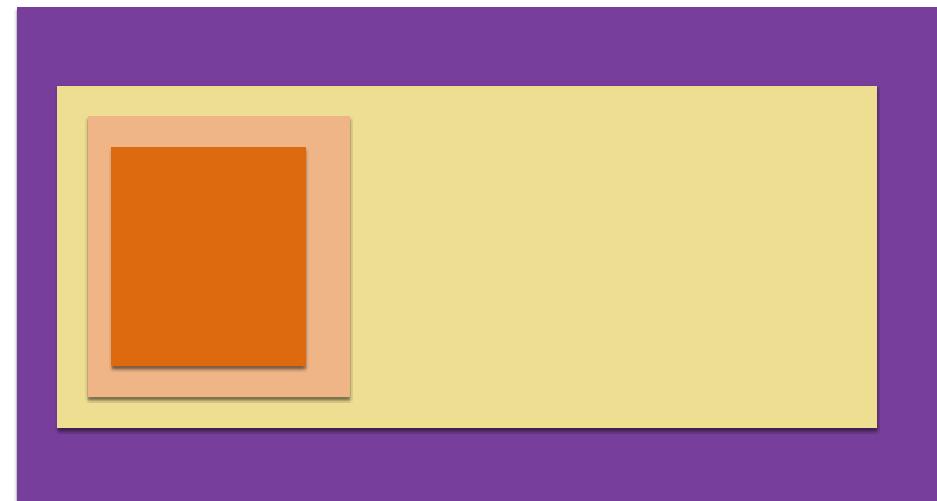
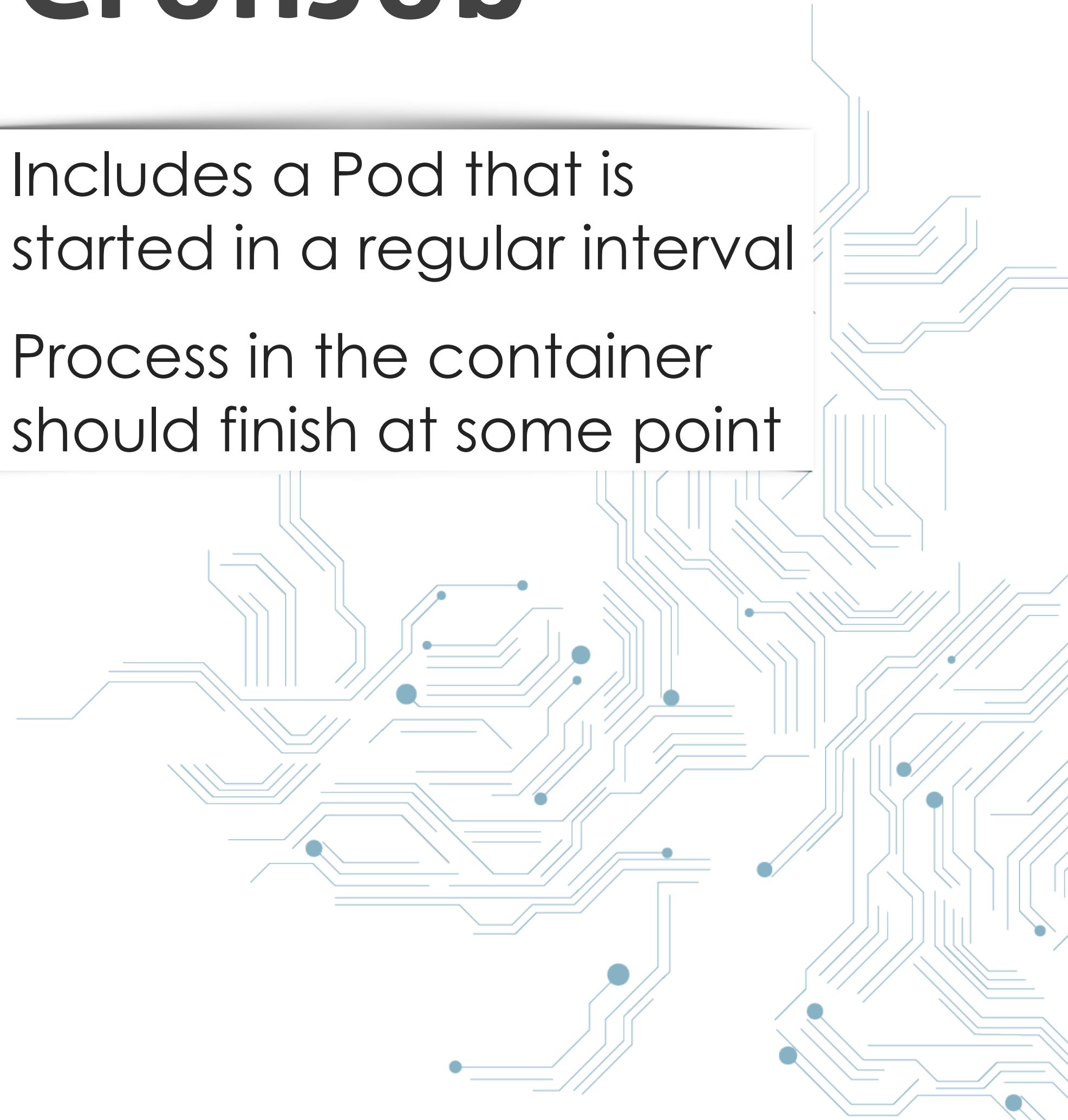
Namespaces

- Dedicated environment to deploy services in



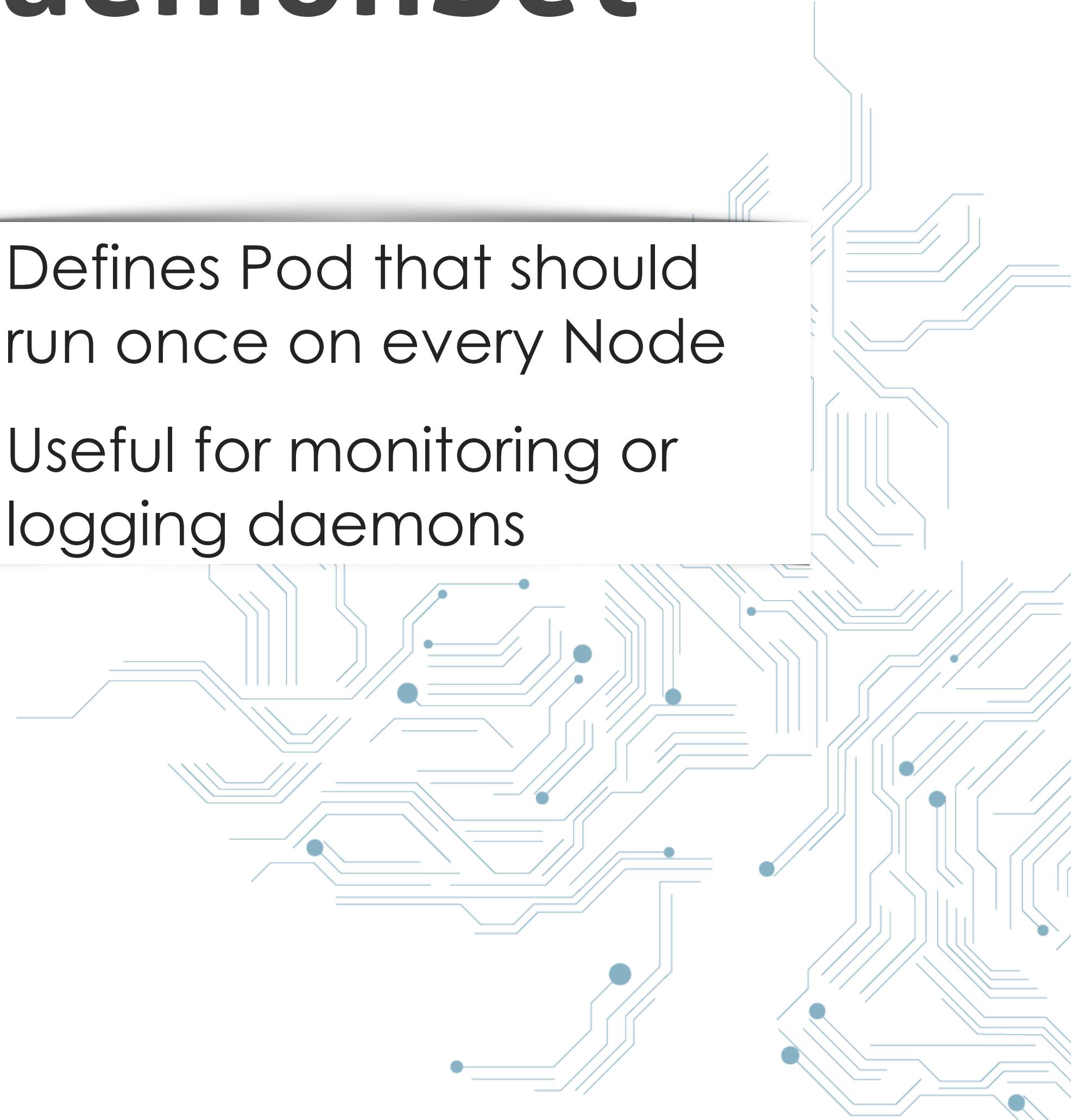
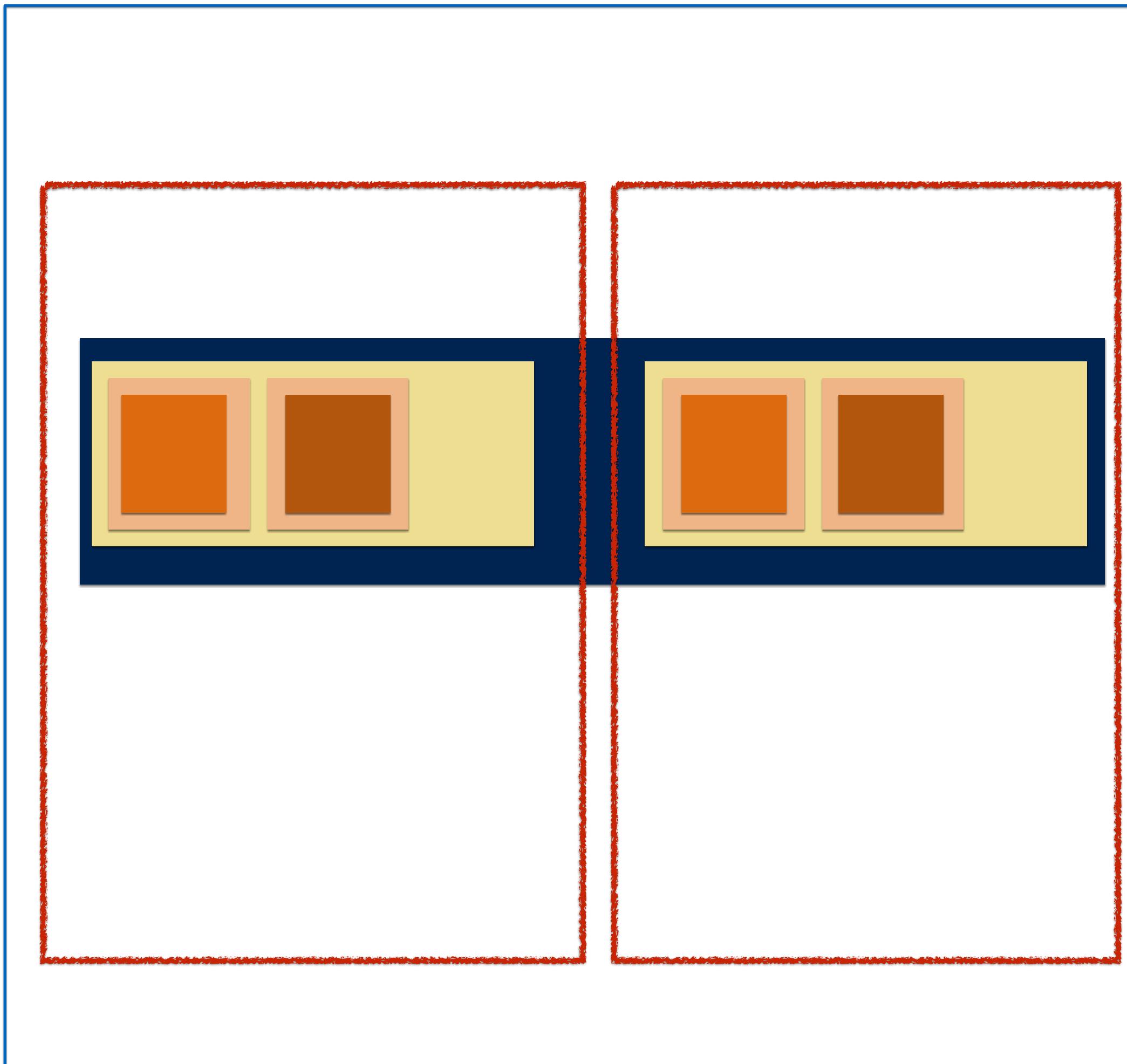
CronJob

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

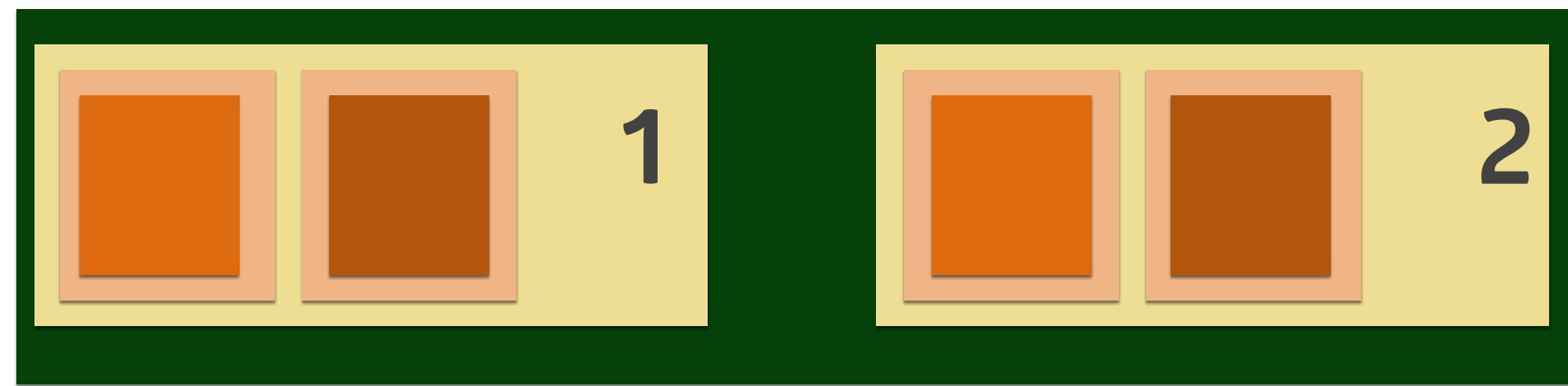


DaemonSet

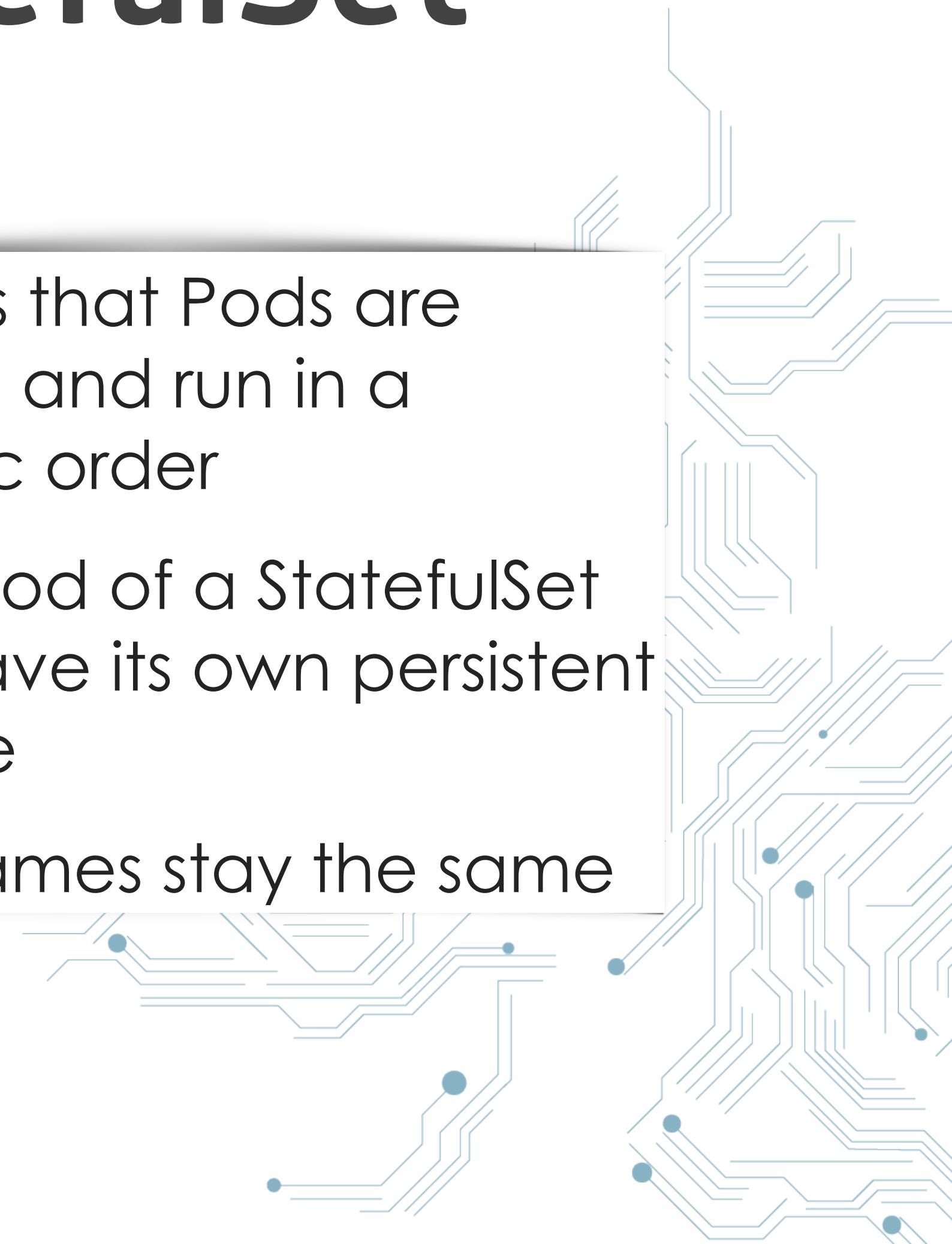
- Defines Pod that should run once on every Node
- Useful for monitoring or logging daemons



StatefulSet



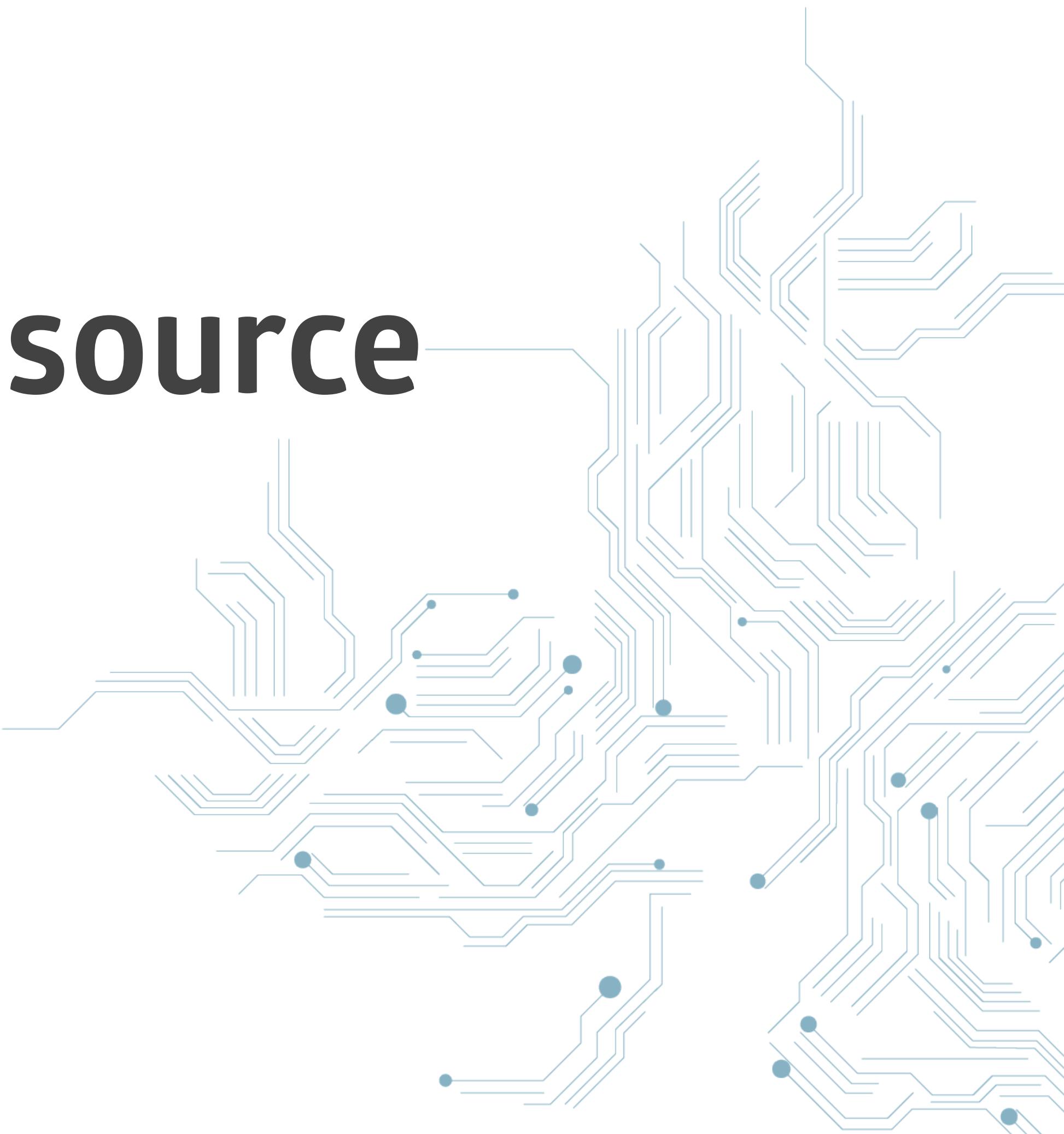
- Ensures that Pods are started and run in a specific order
- Each Pod of a StatefulSet can have its own persistent volume
- Pod names stay the same



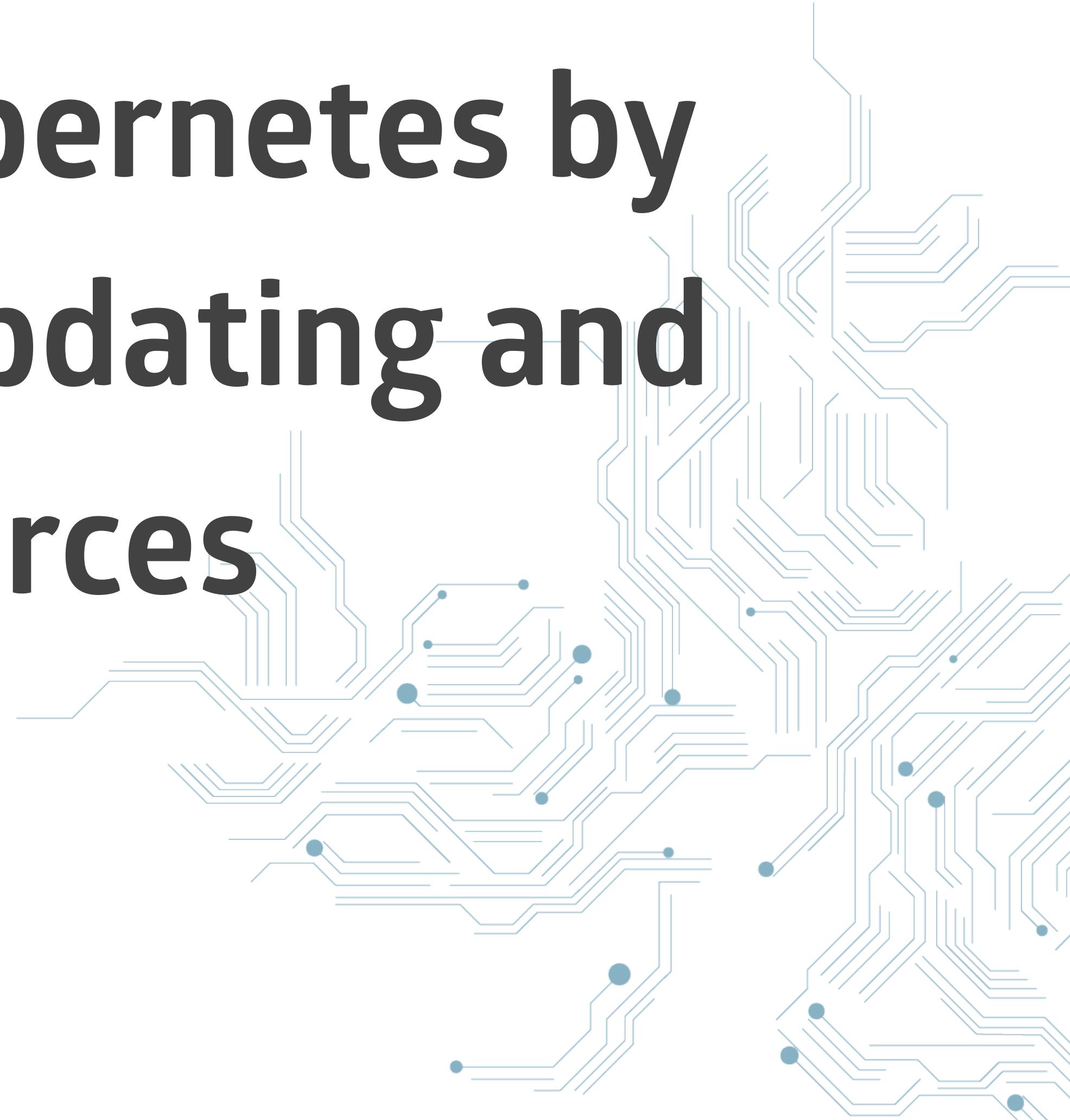
...



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**



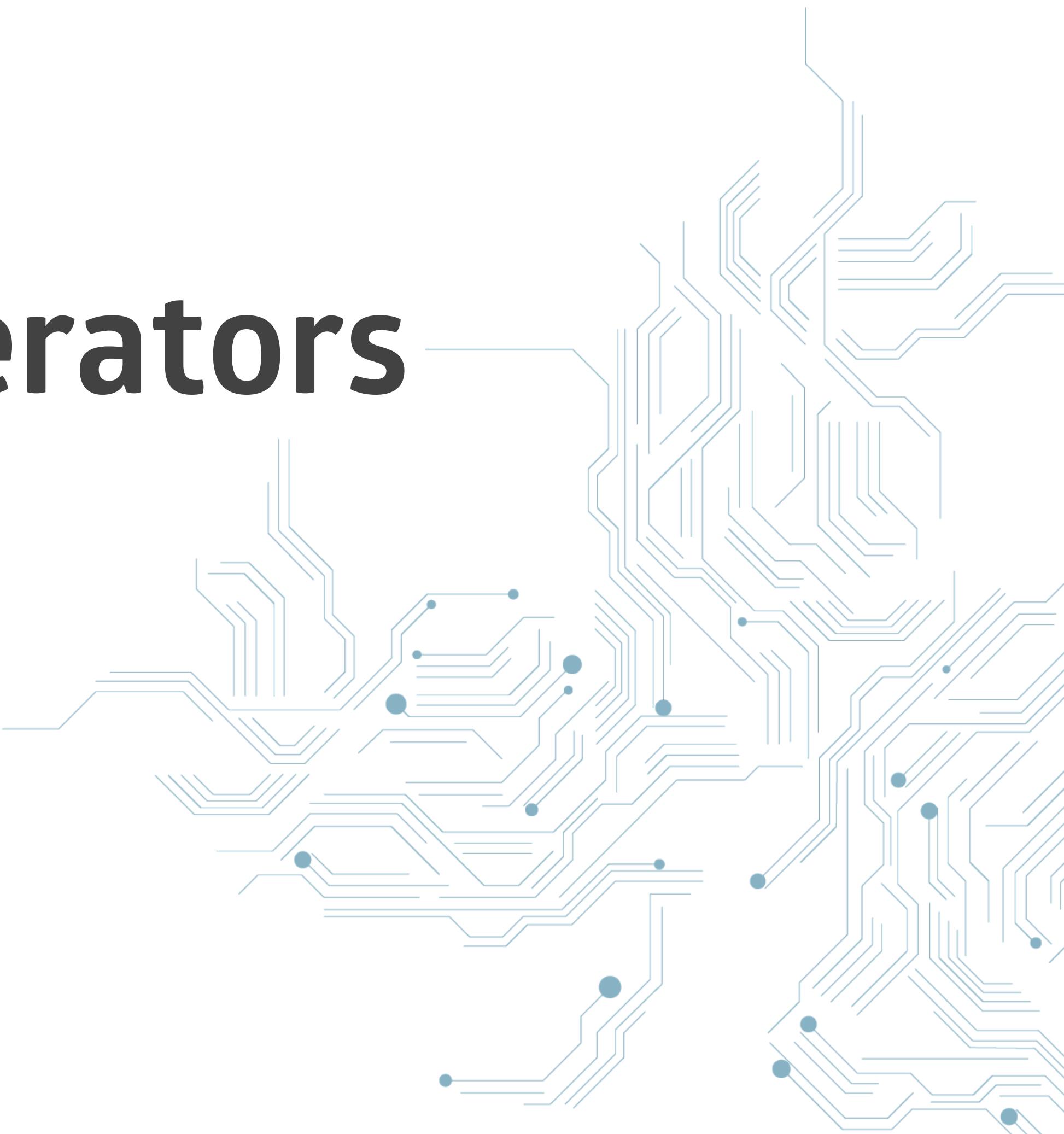
CustomResourceDefinitions



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	

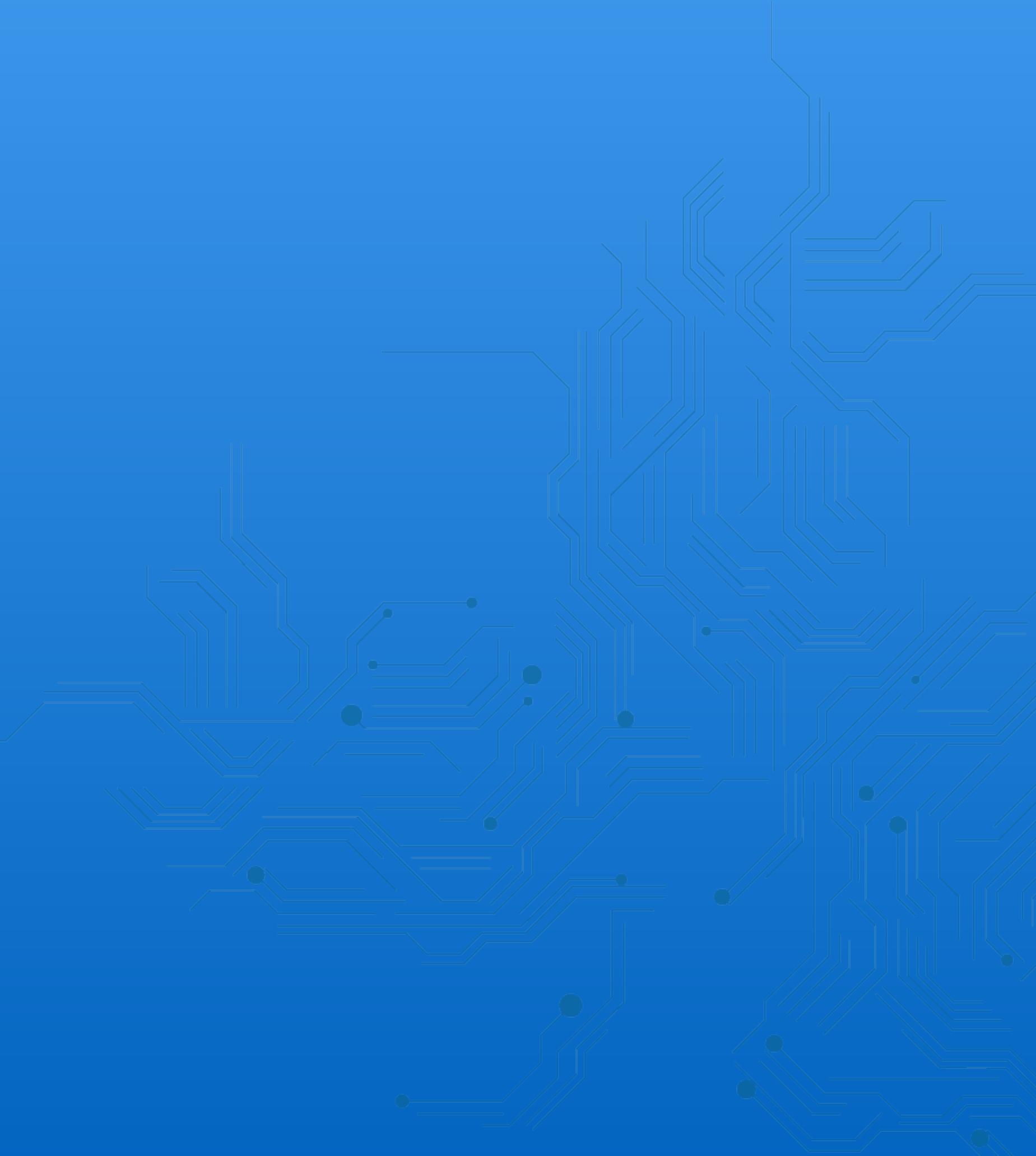
```
$ kubectl get deployment symfony-demo -o yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    ...
spec:
  ...
template:
  ...
    spec:
      containers:
        - name: symfony-demo
```



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



Tooling



kubectl

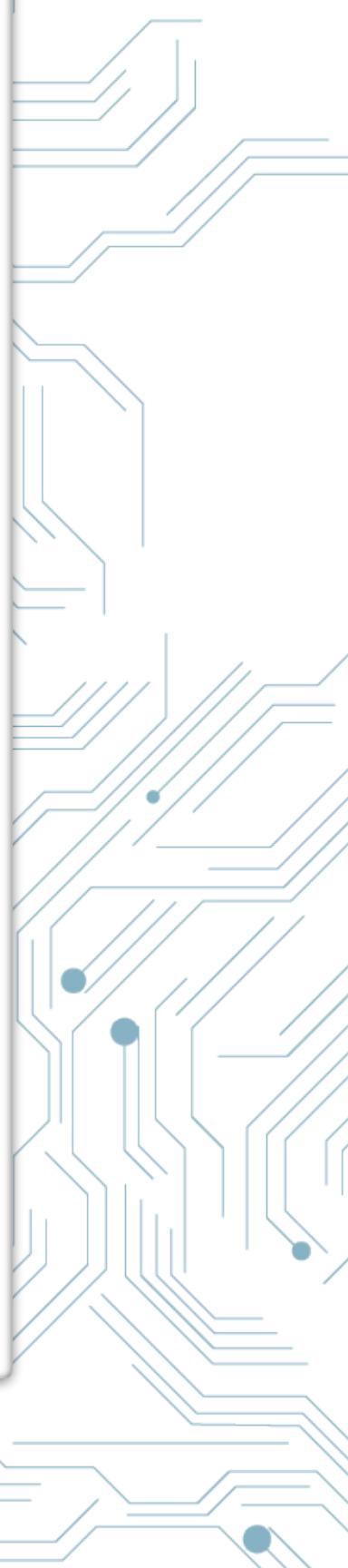


REST API

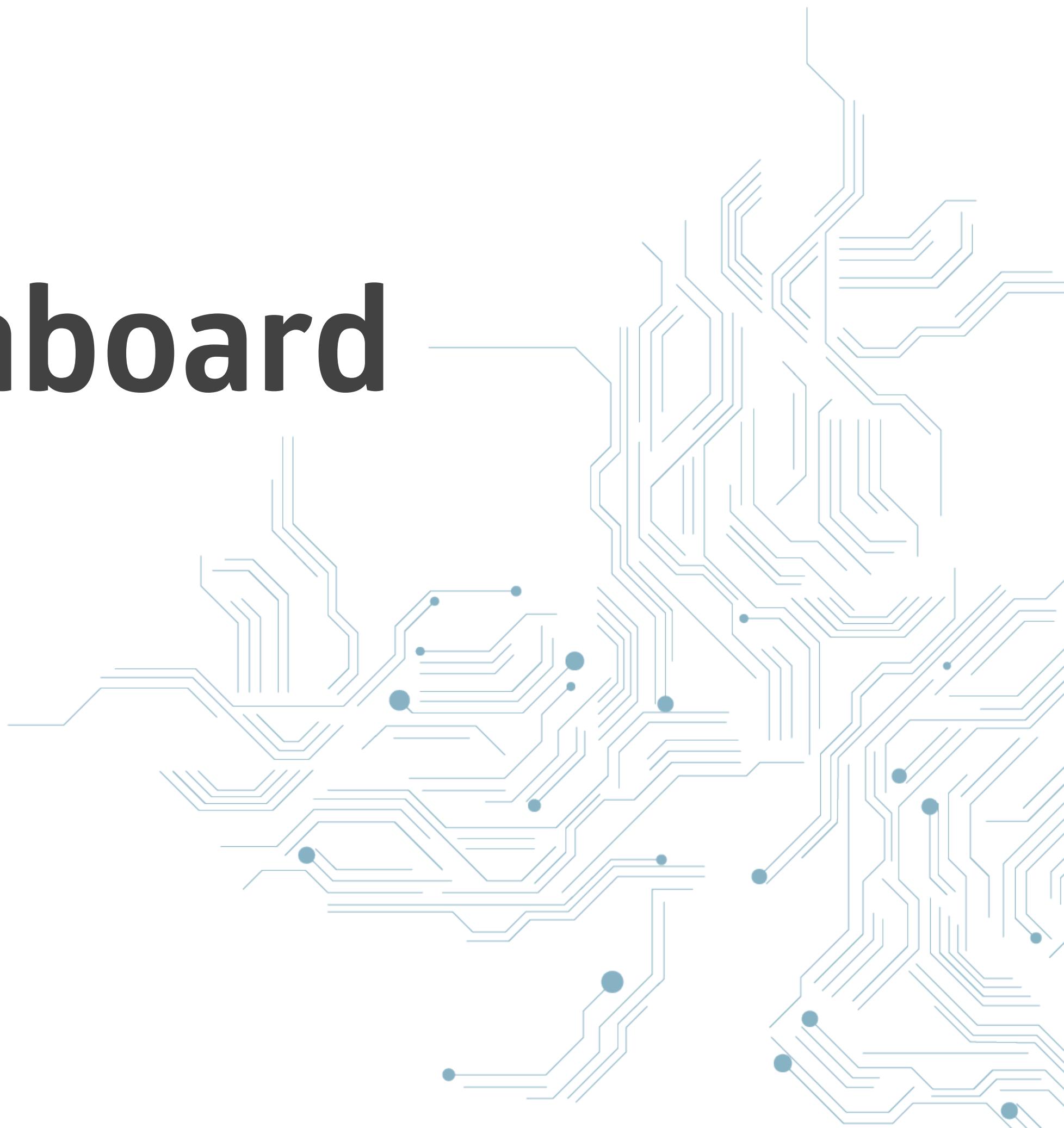


```
$ kubectl proxy --port=8080
$ curl http://localhost:8080/api/v1/namespaces/default/pods
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/namespaces/default/pods",
    "resourceVersion": "336834"
  },
  "items": [
    {
      "metadata": {
        "name": "kubernetes-dashboard-5b5bf59977-t9xb9",

```



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

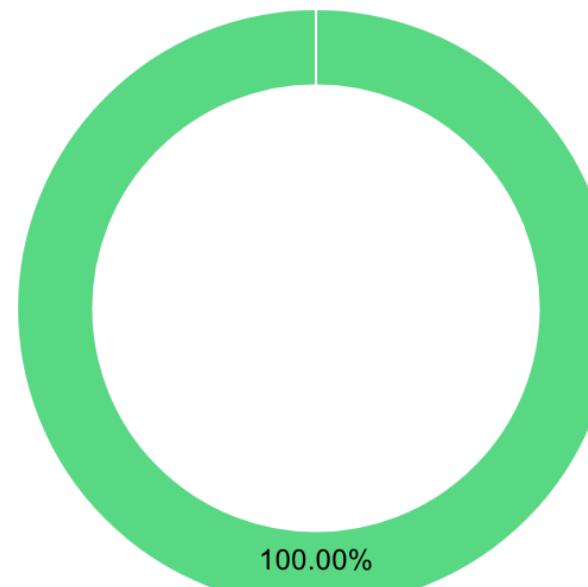
- Config Maps
- Persistent Volume Claims

Resource Status

Pods

- Running
- Pending
- Failed

- 6
- 0
- 0

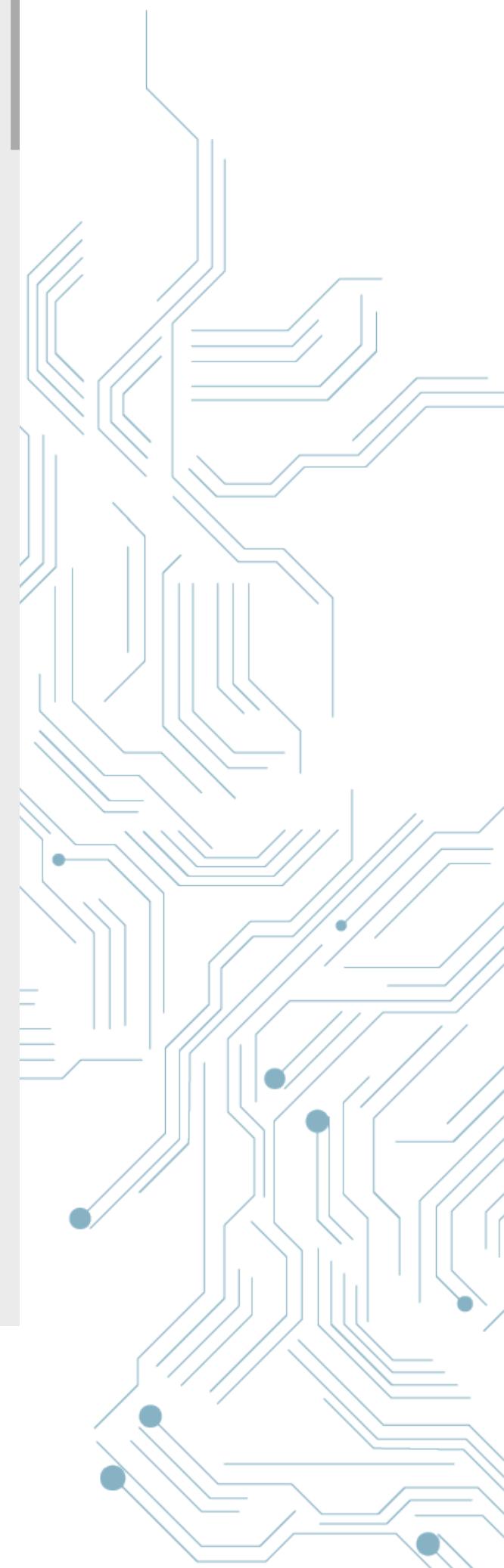


Pods

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
```

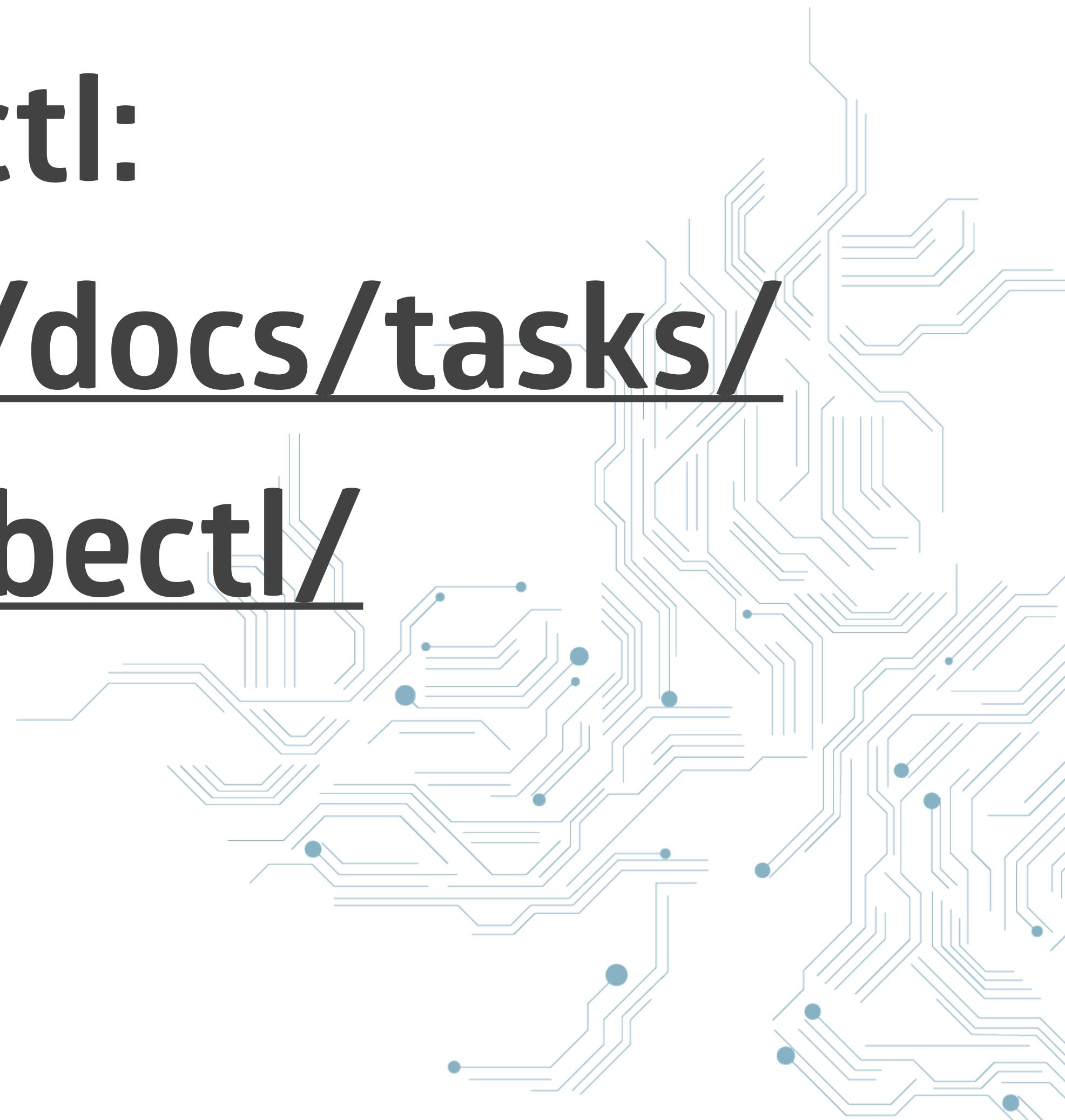


hands-on



Install kubectl:

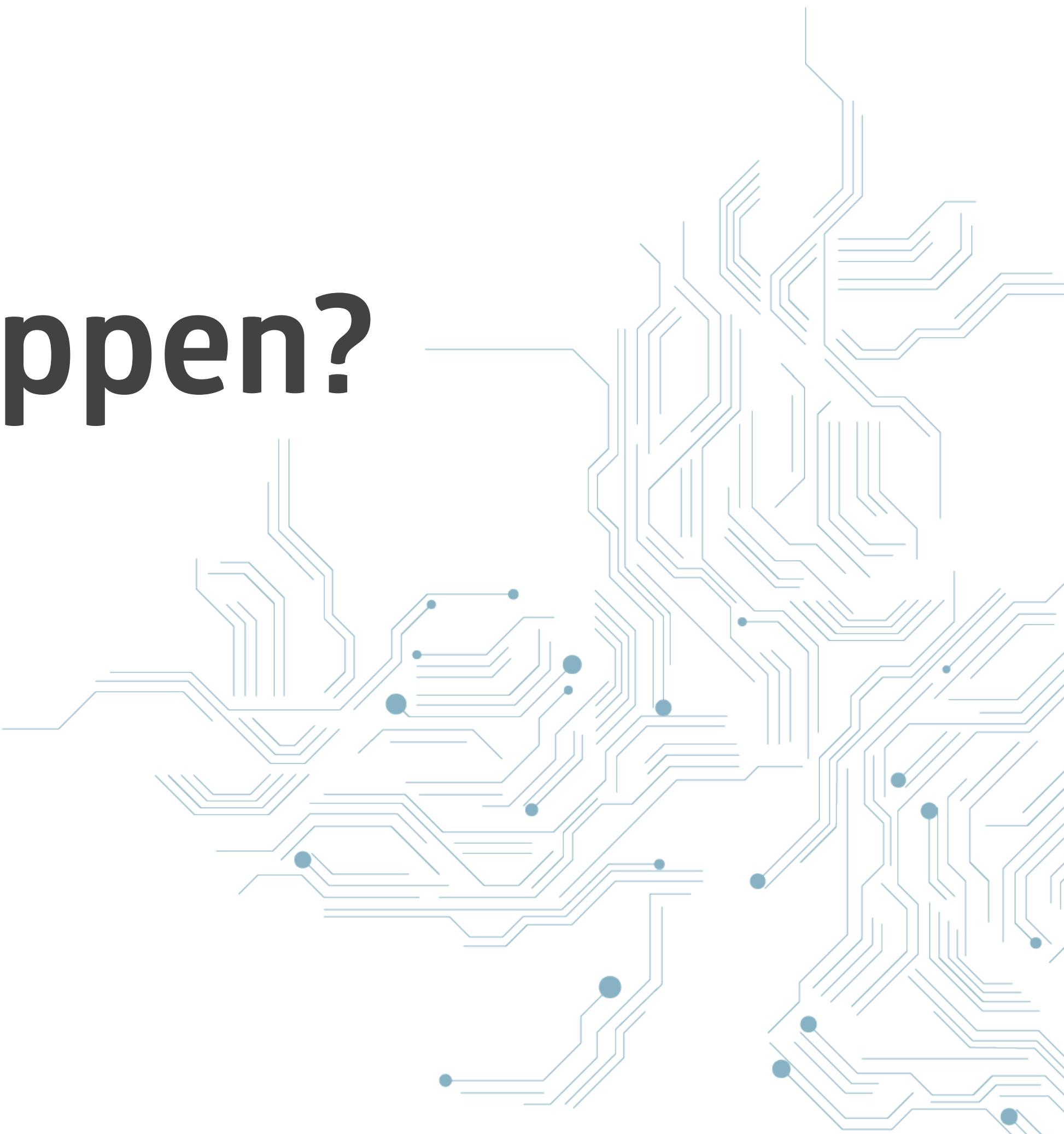
<https://kubernetes.io/docs/tasks/tools/install-kubectl/>

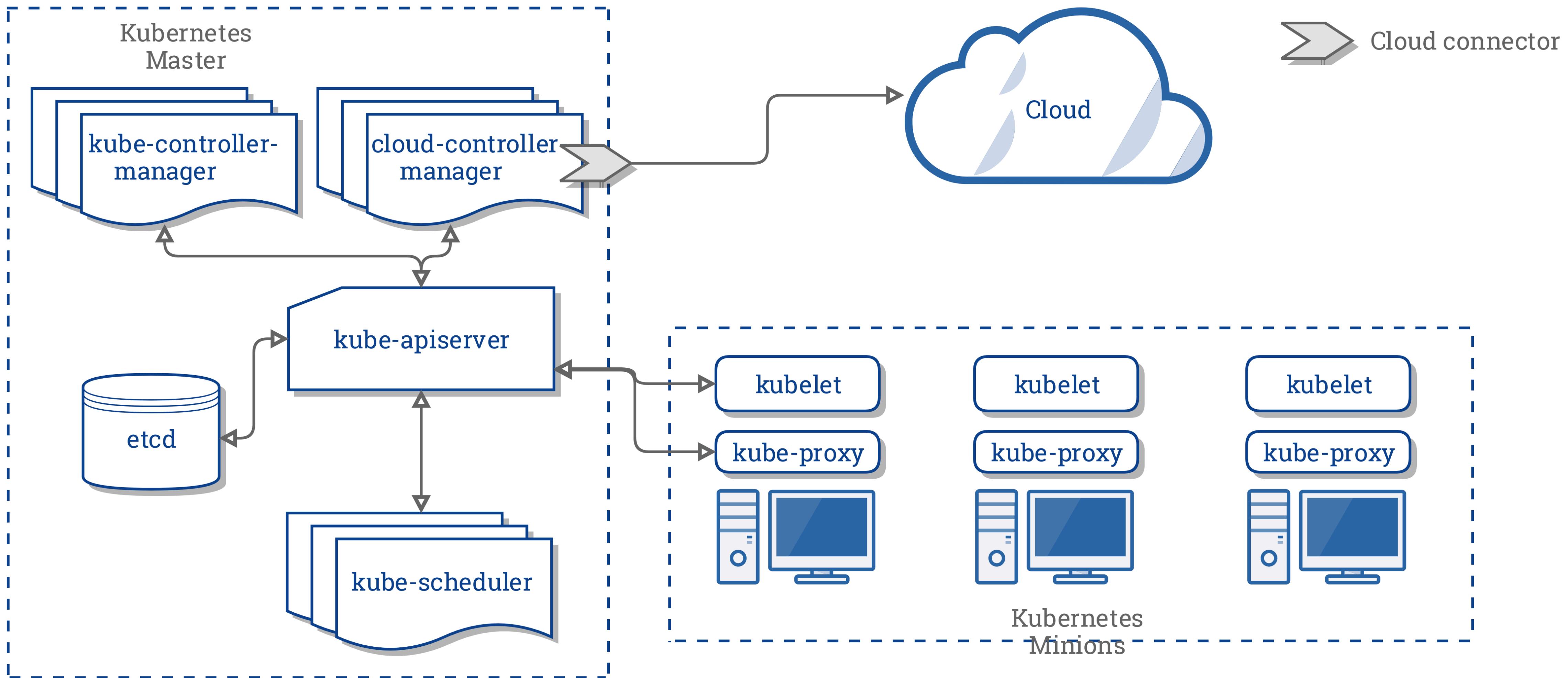


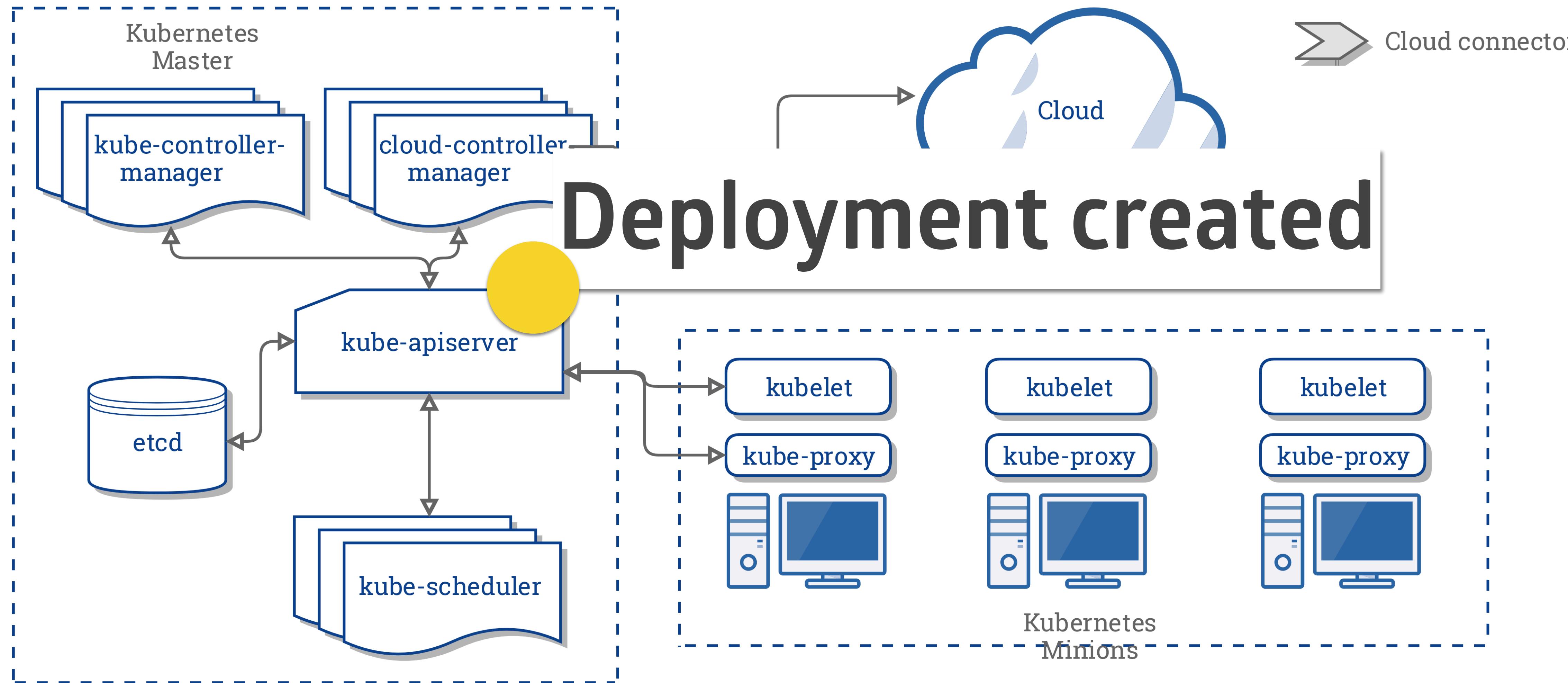
1 - Deploying a simple Web Application



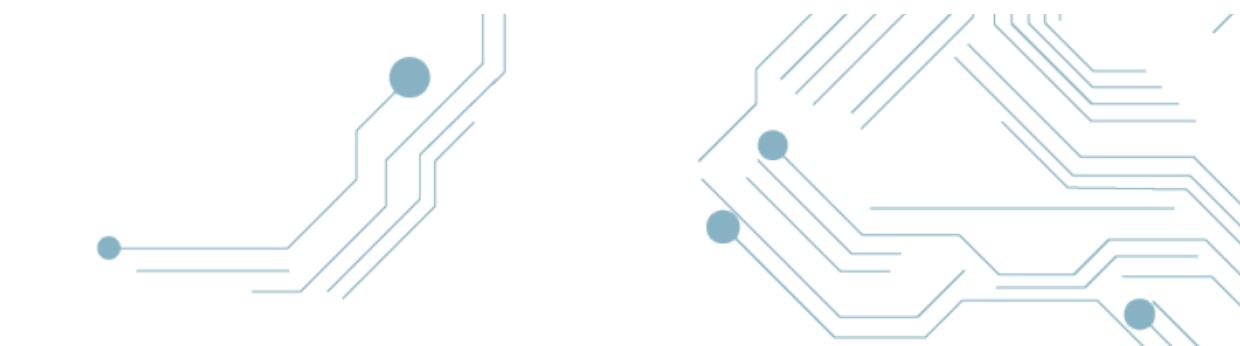
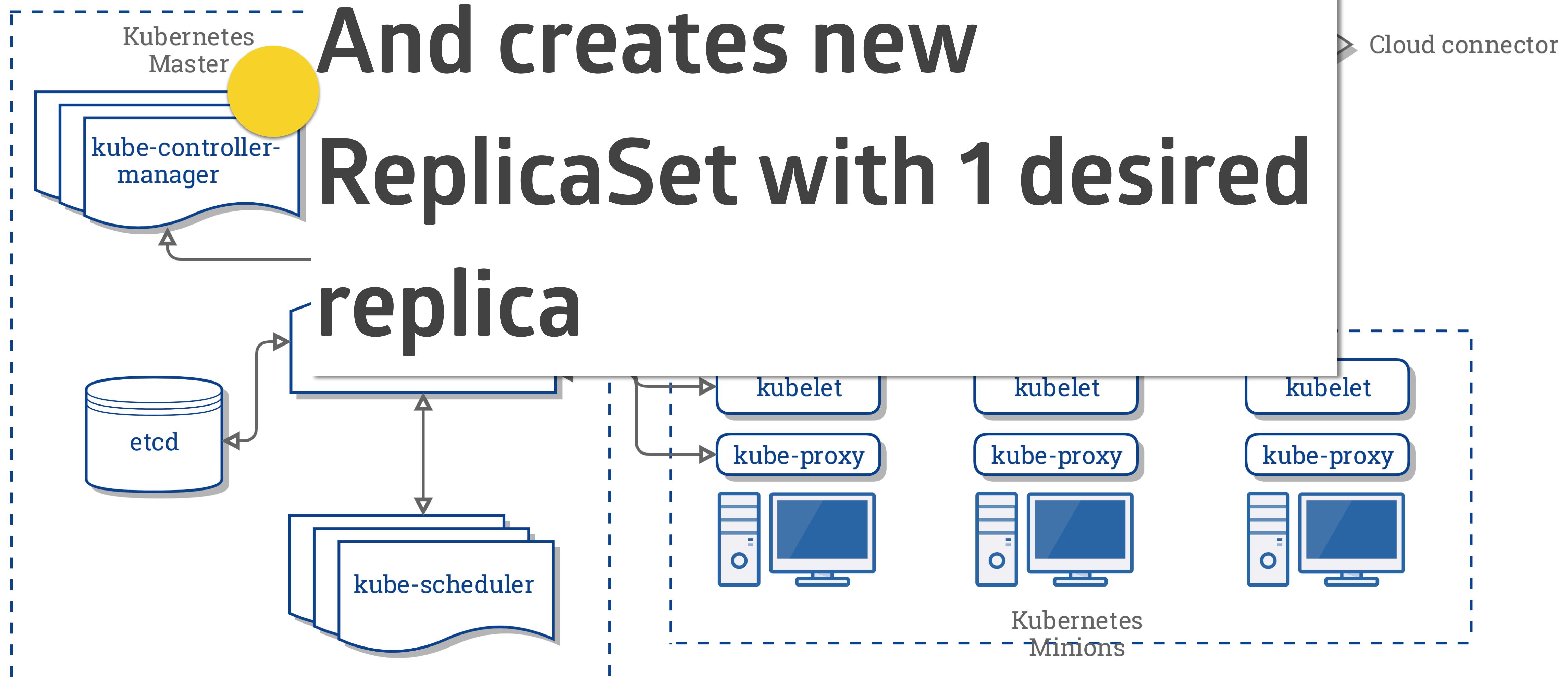
What did just happen?



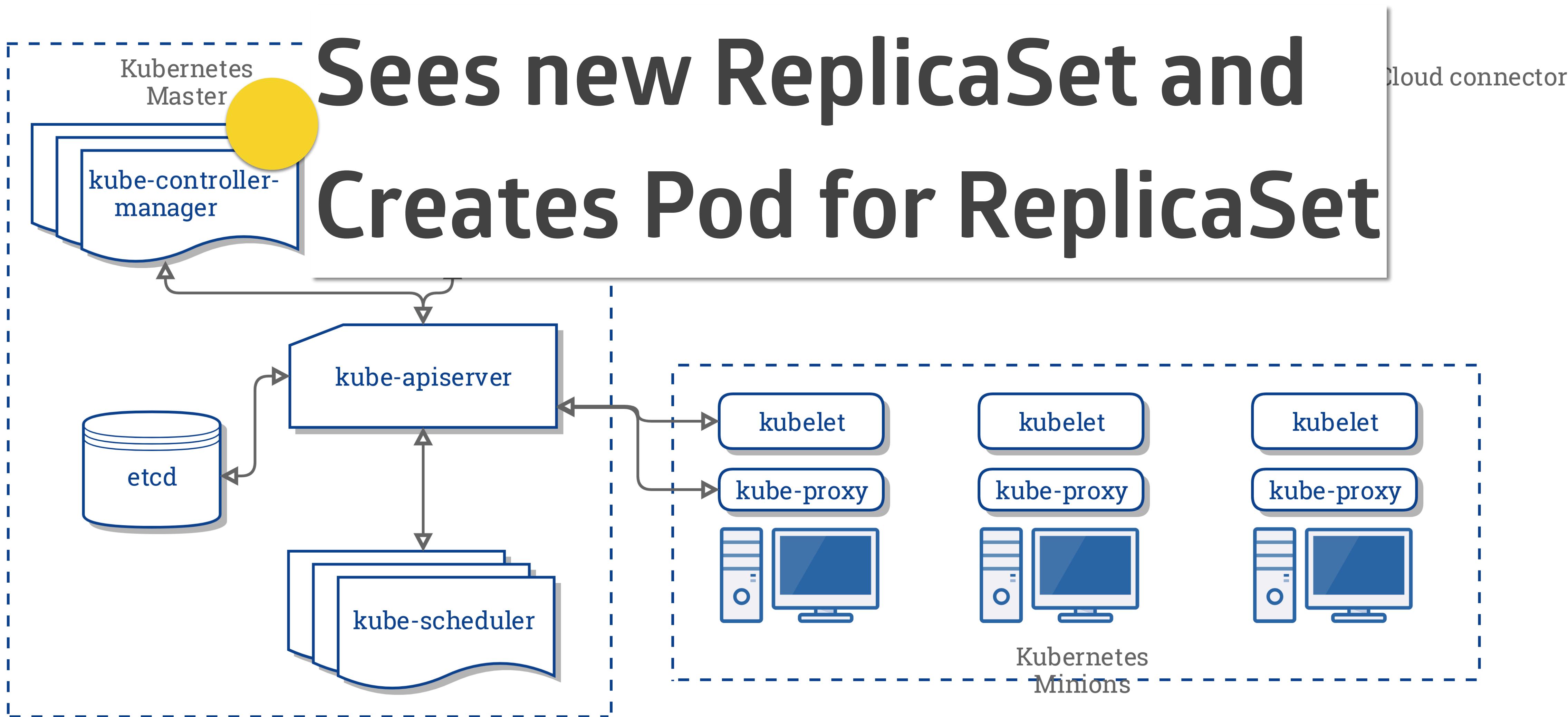


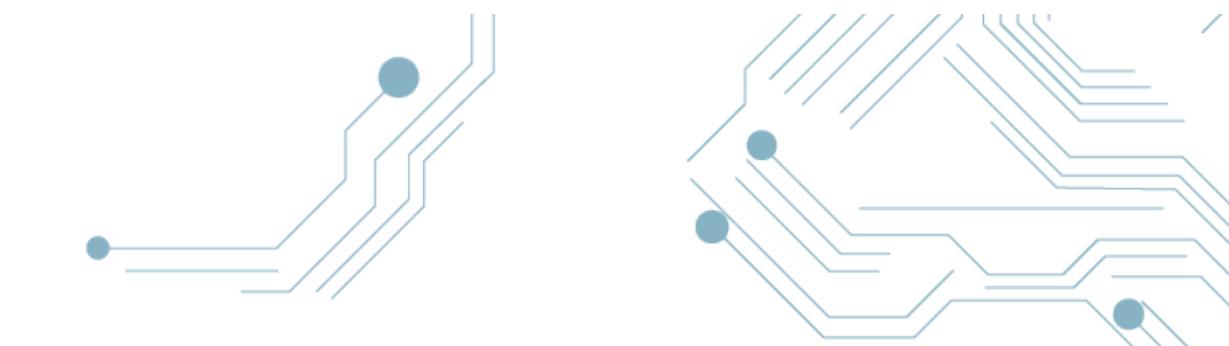
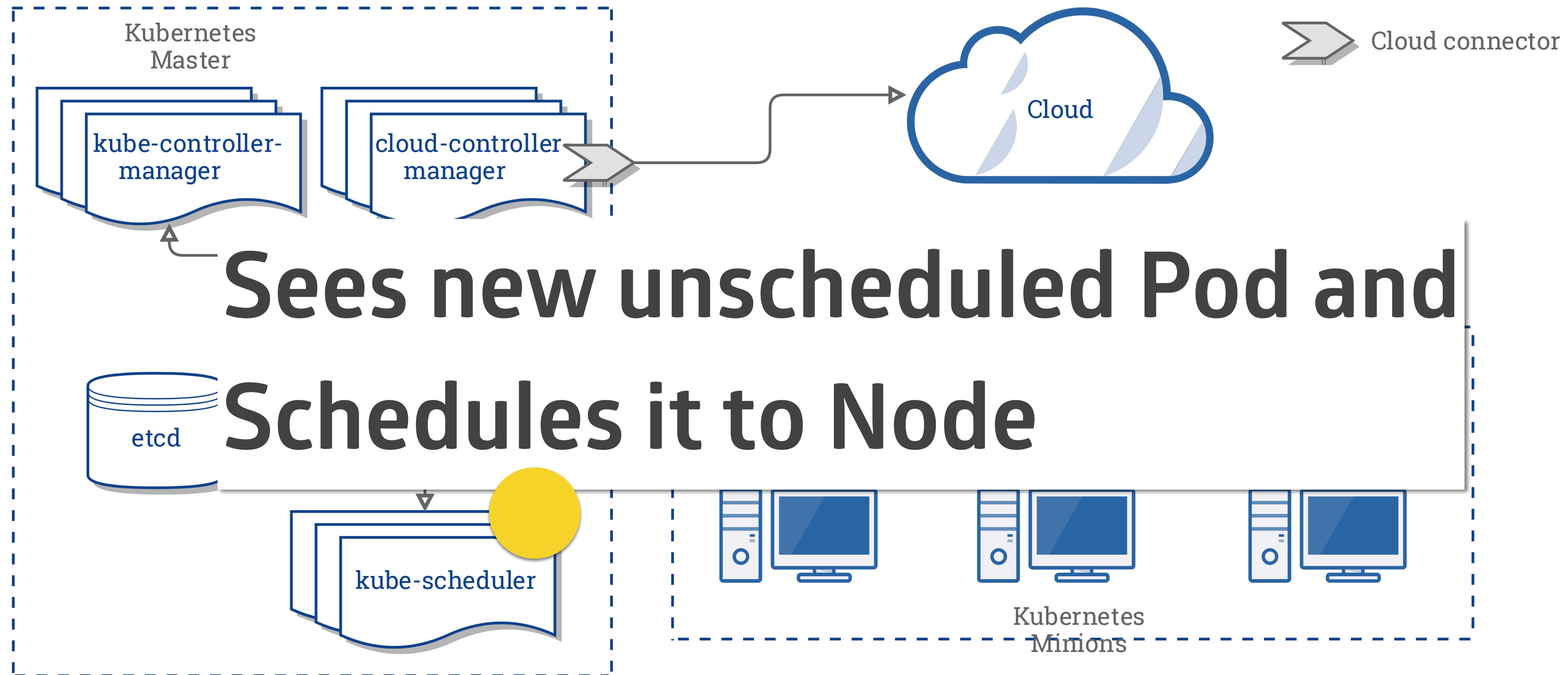


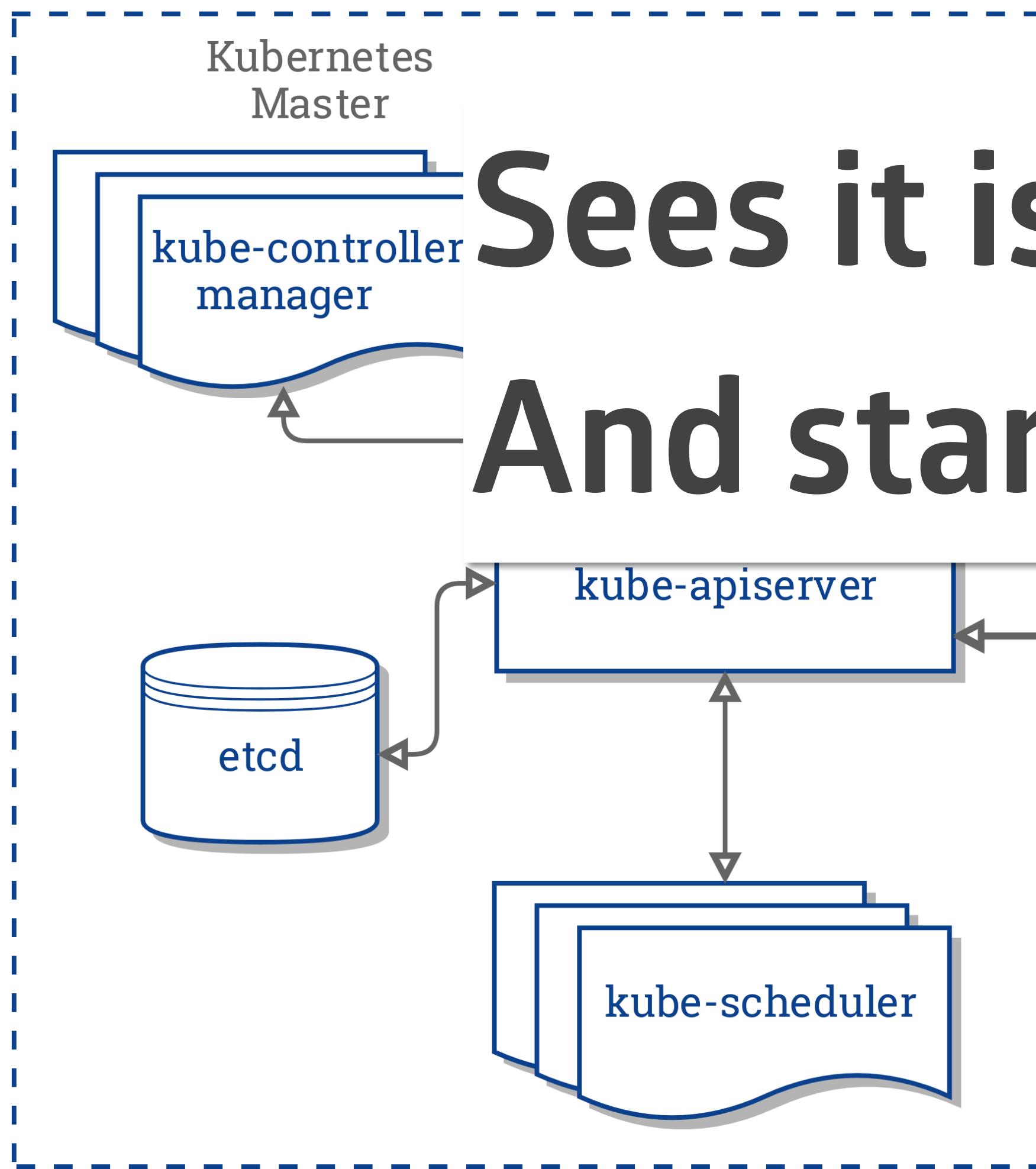
Sees new Deployment



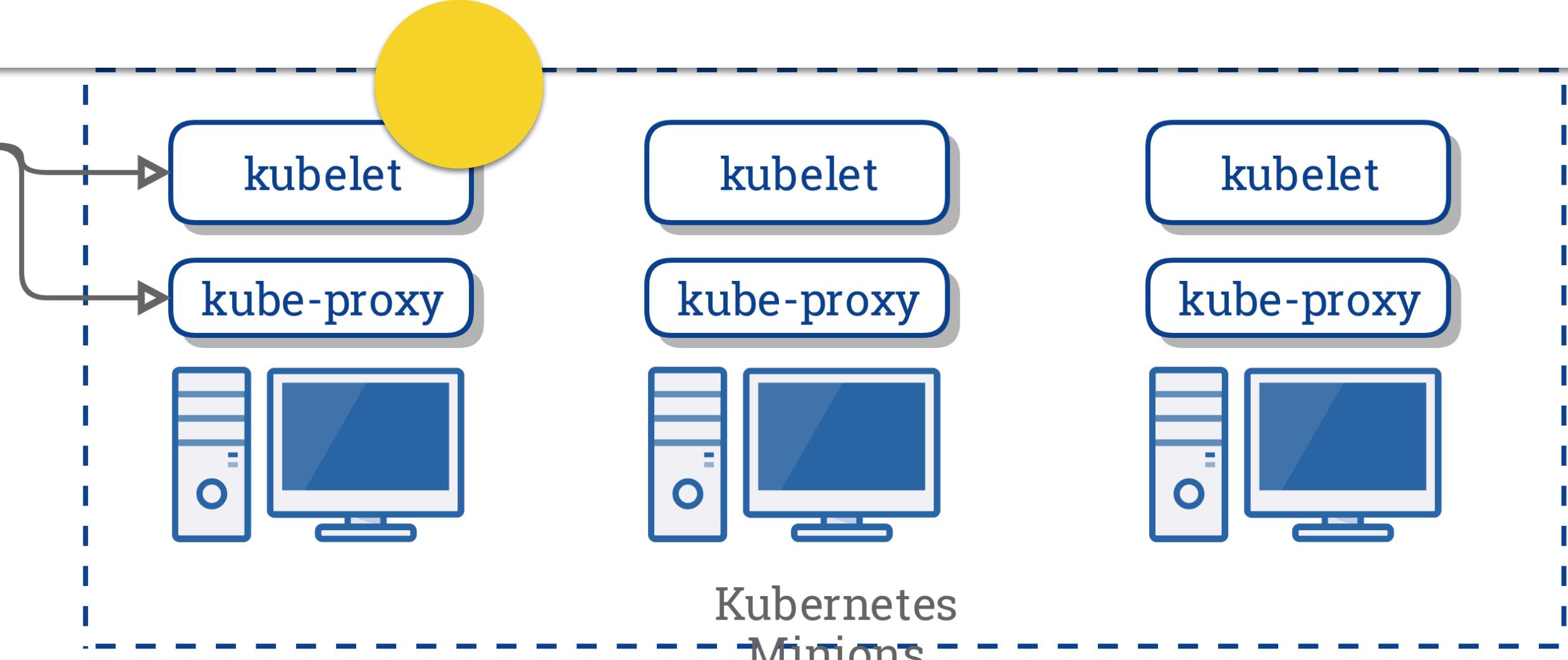
Sees new ReplicaSet and Creates Pod for ReplicaSet



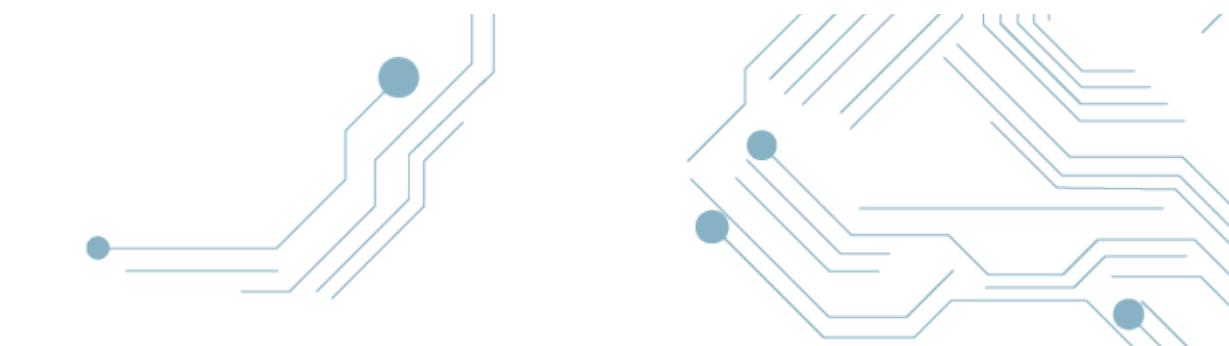
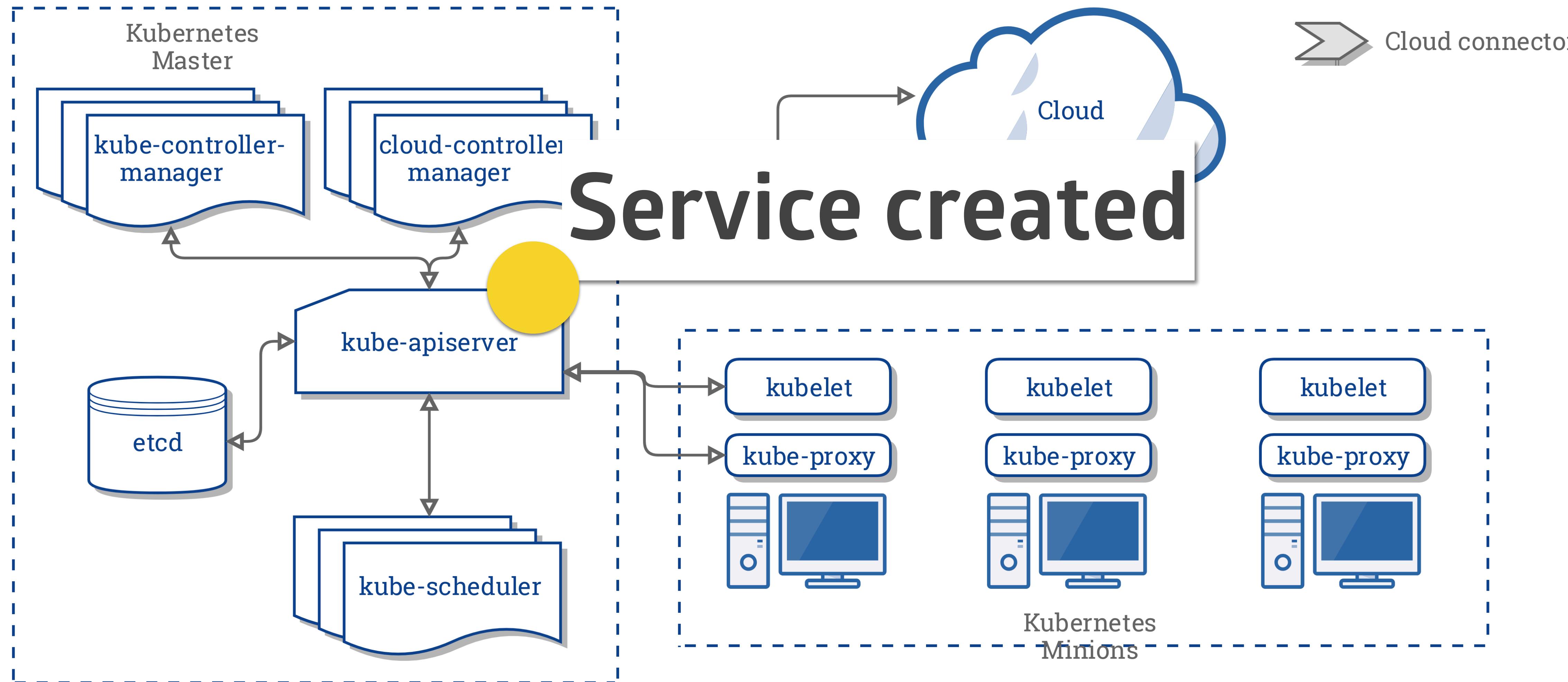




Sees it is supposed to start a Pod And starts its Containers



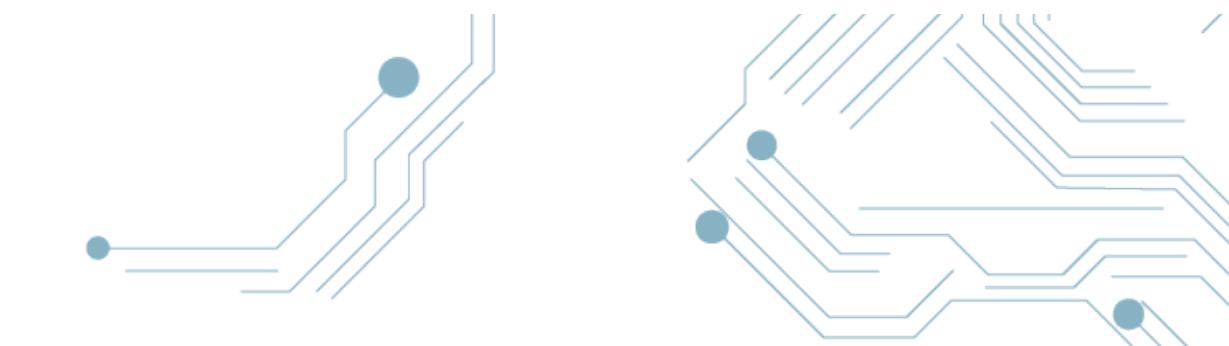
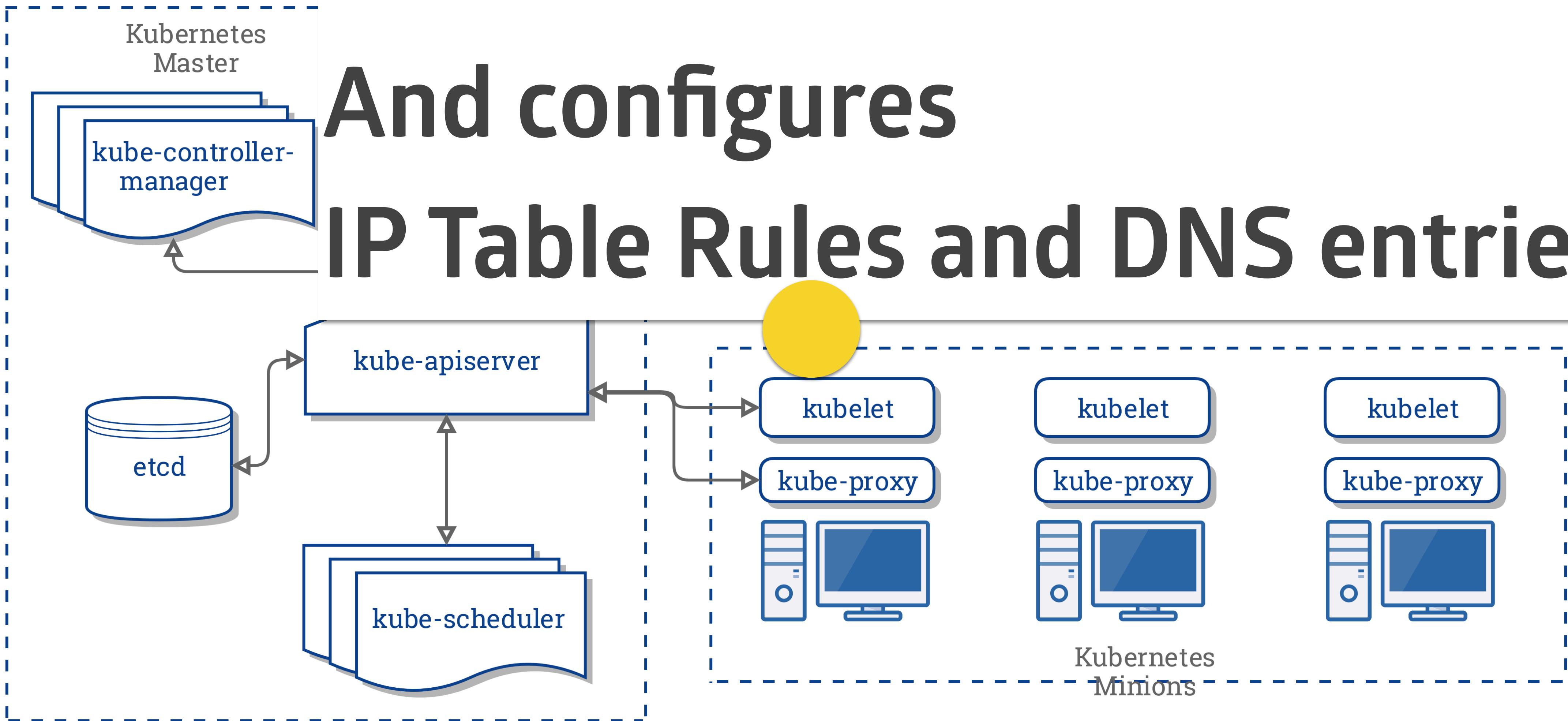
Cloud connector

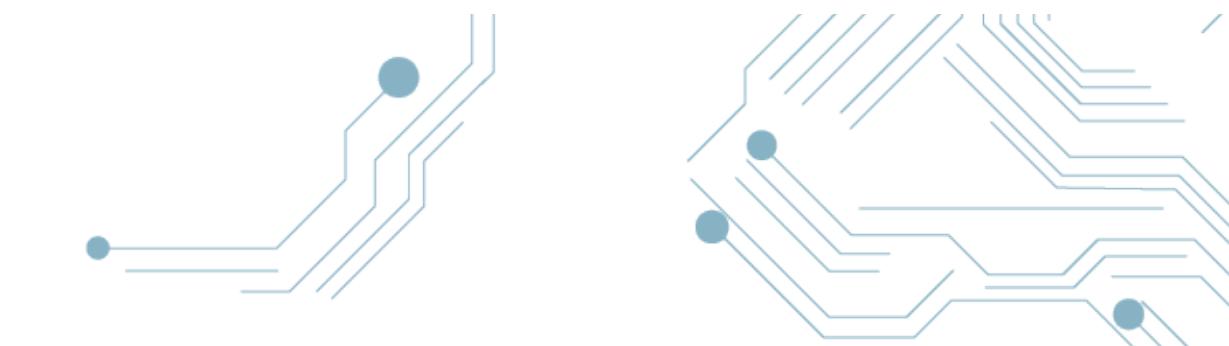
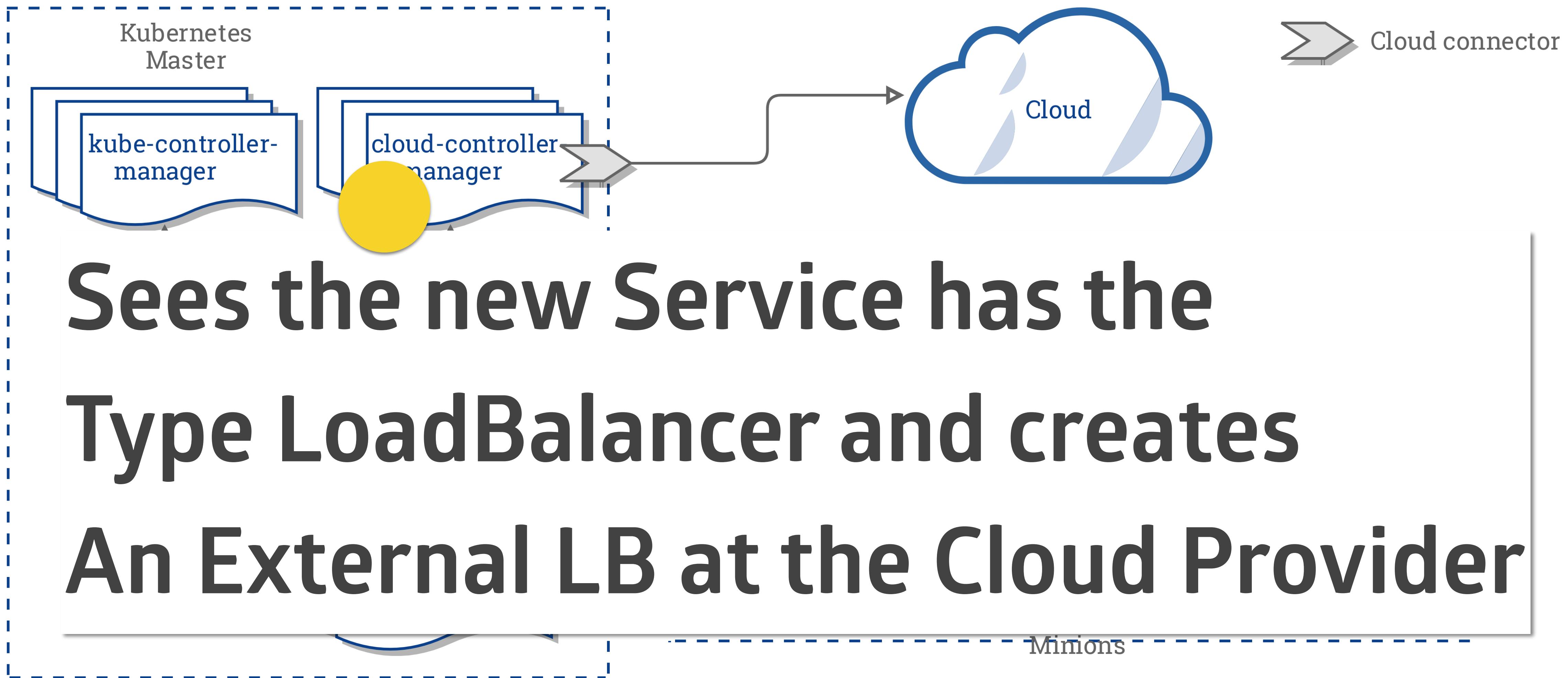


Sees the new Service

And configures

IP Table Rules and DNS entries

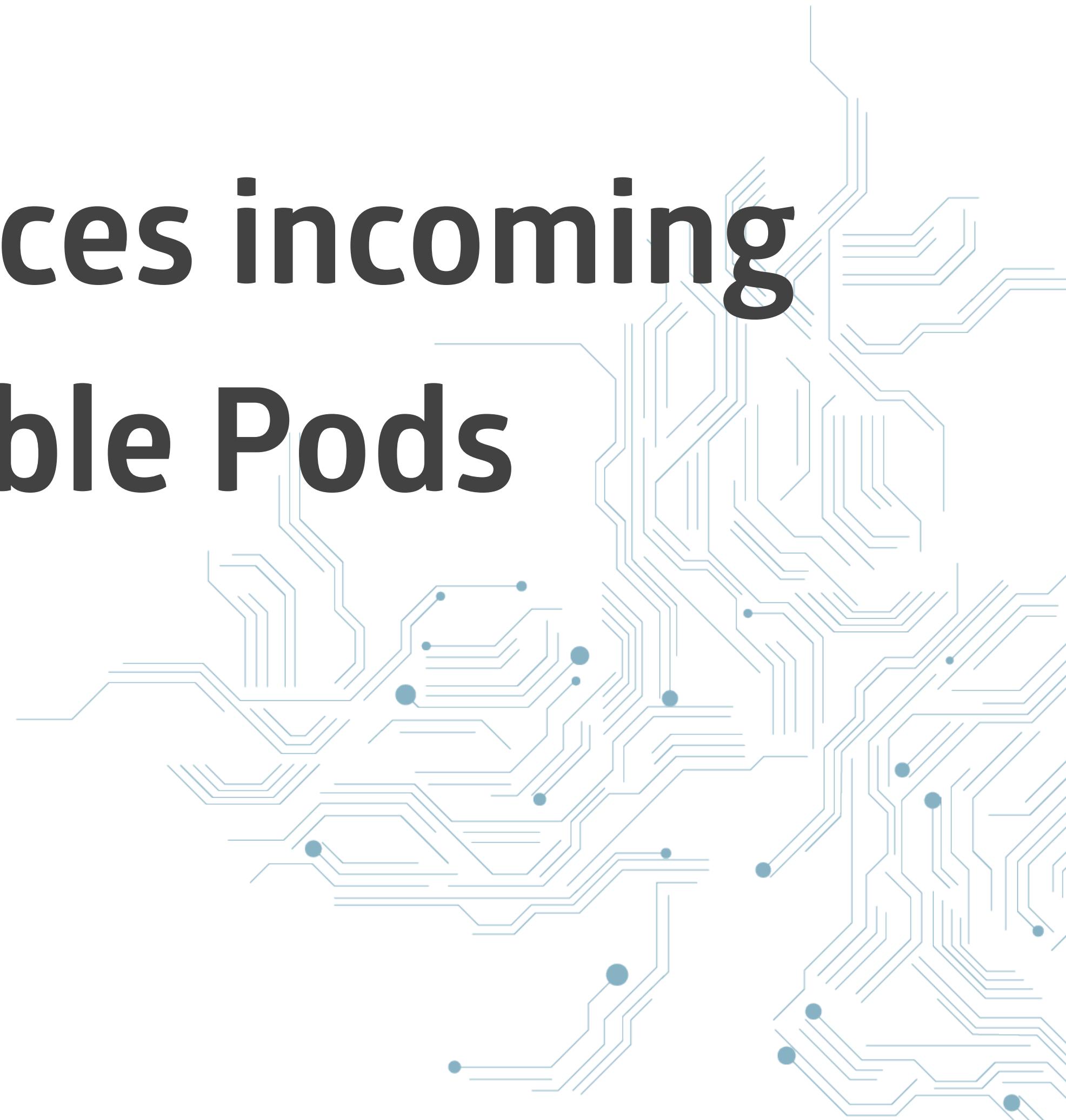




How is traffic routed to the Pod



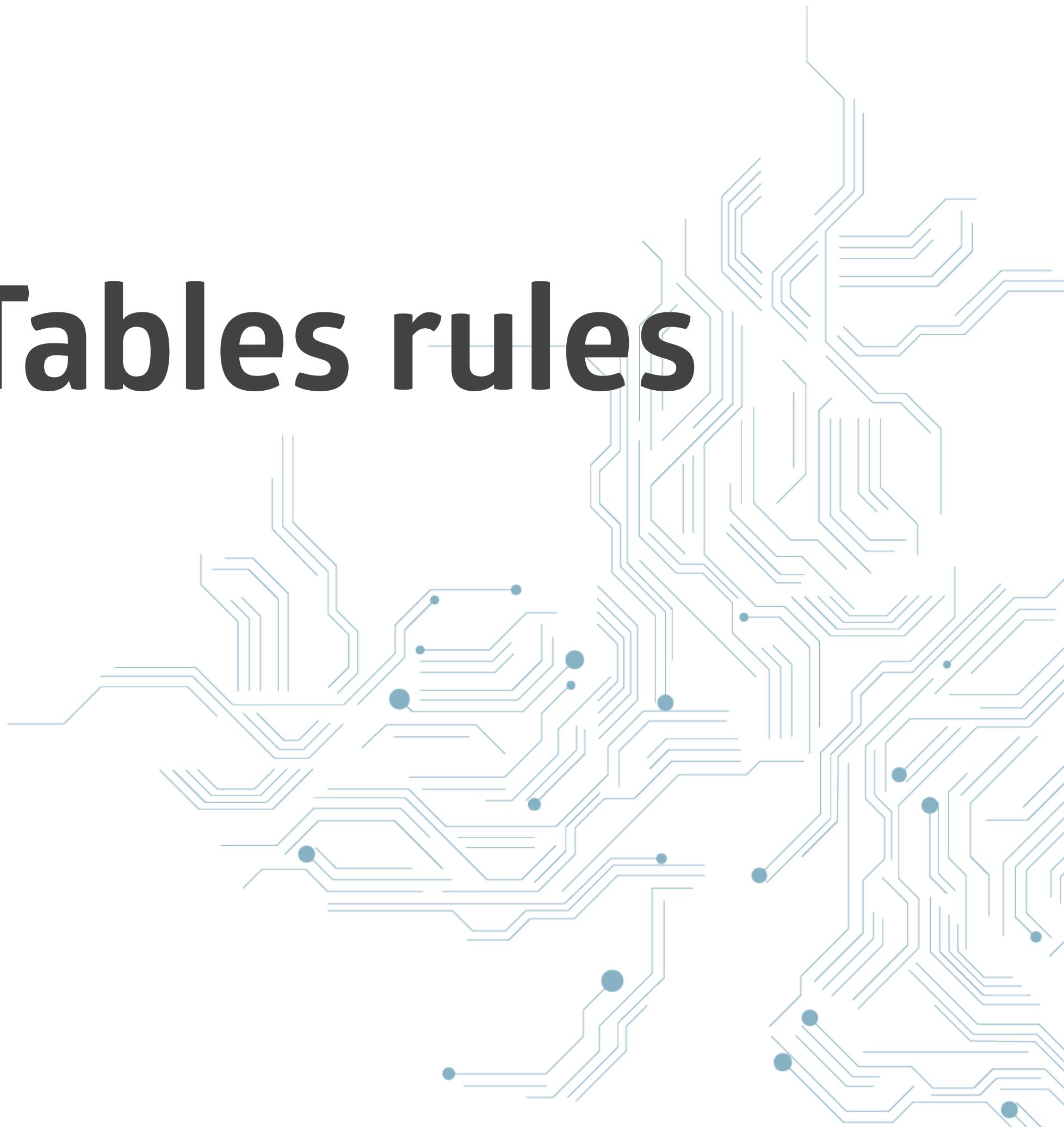
**The Service load-balances incoming
traffic to all available Pods**



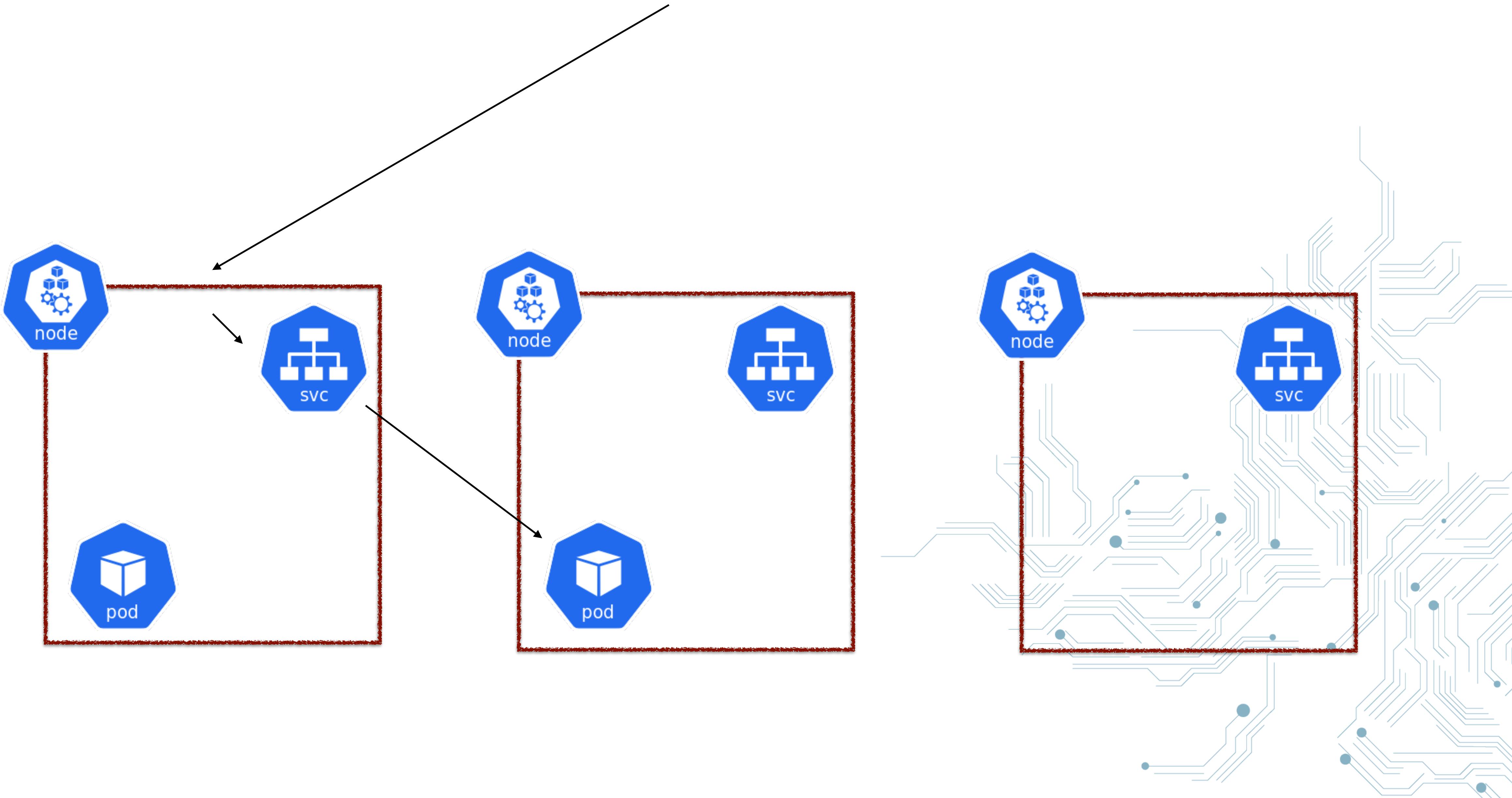
Every Service has a virtual IP



Round Robin with IP Tables rules



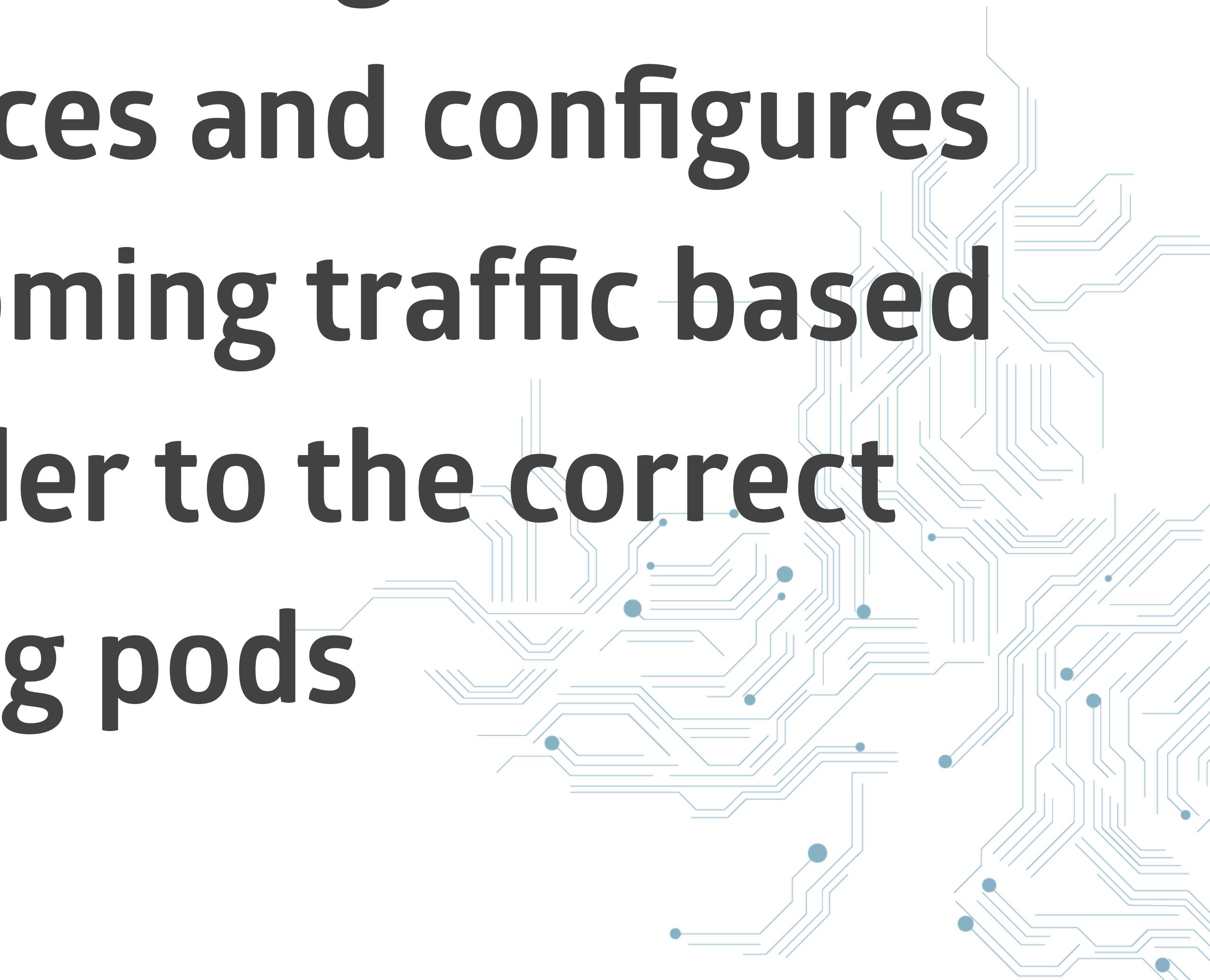
OpenStack LoadBalancer



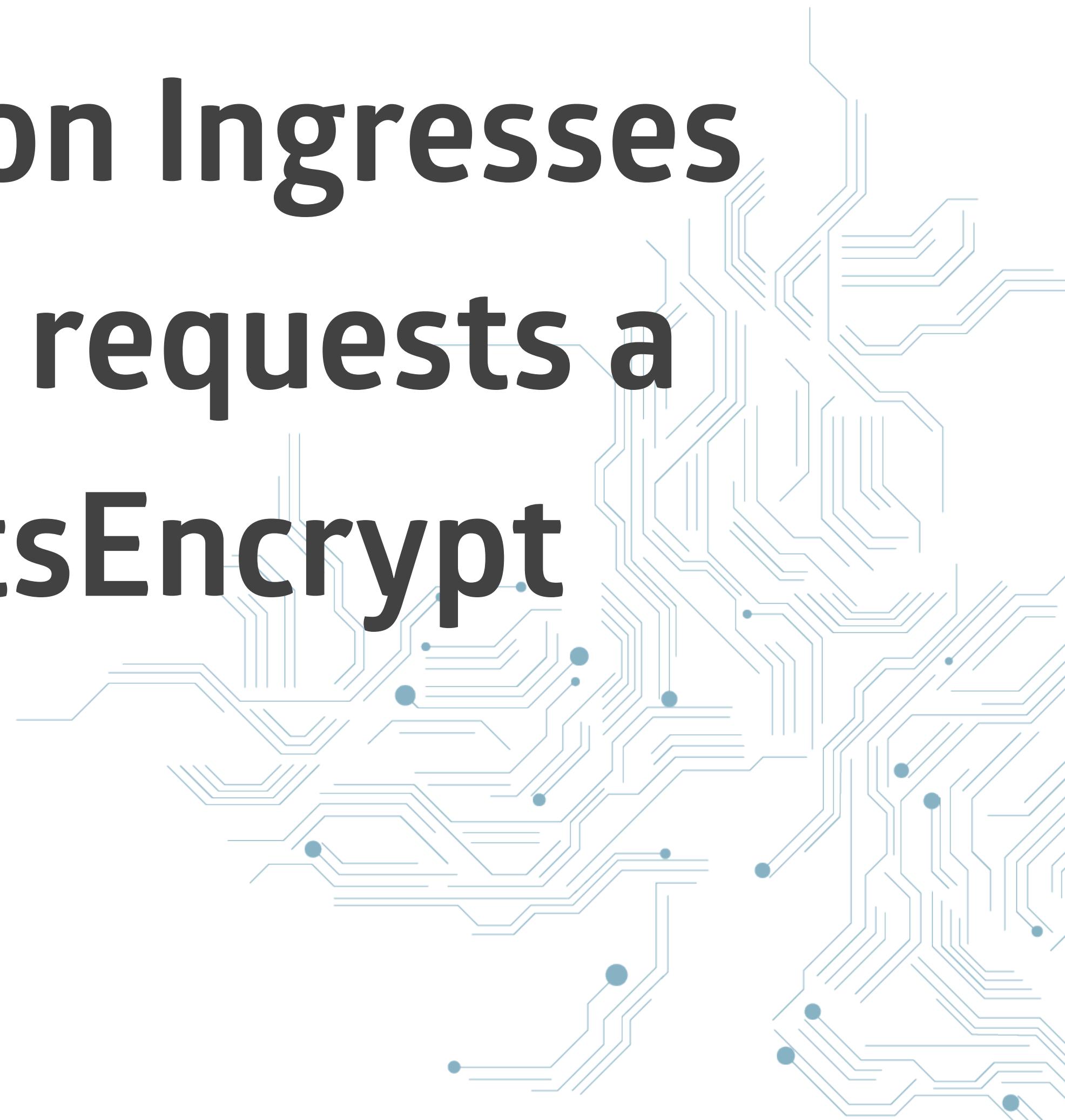
9 - Using an Ingress with TLS



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

