



An Overview

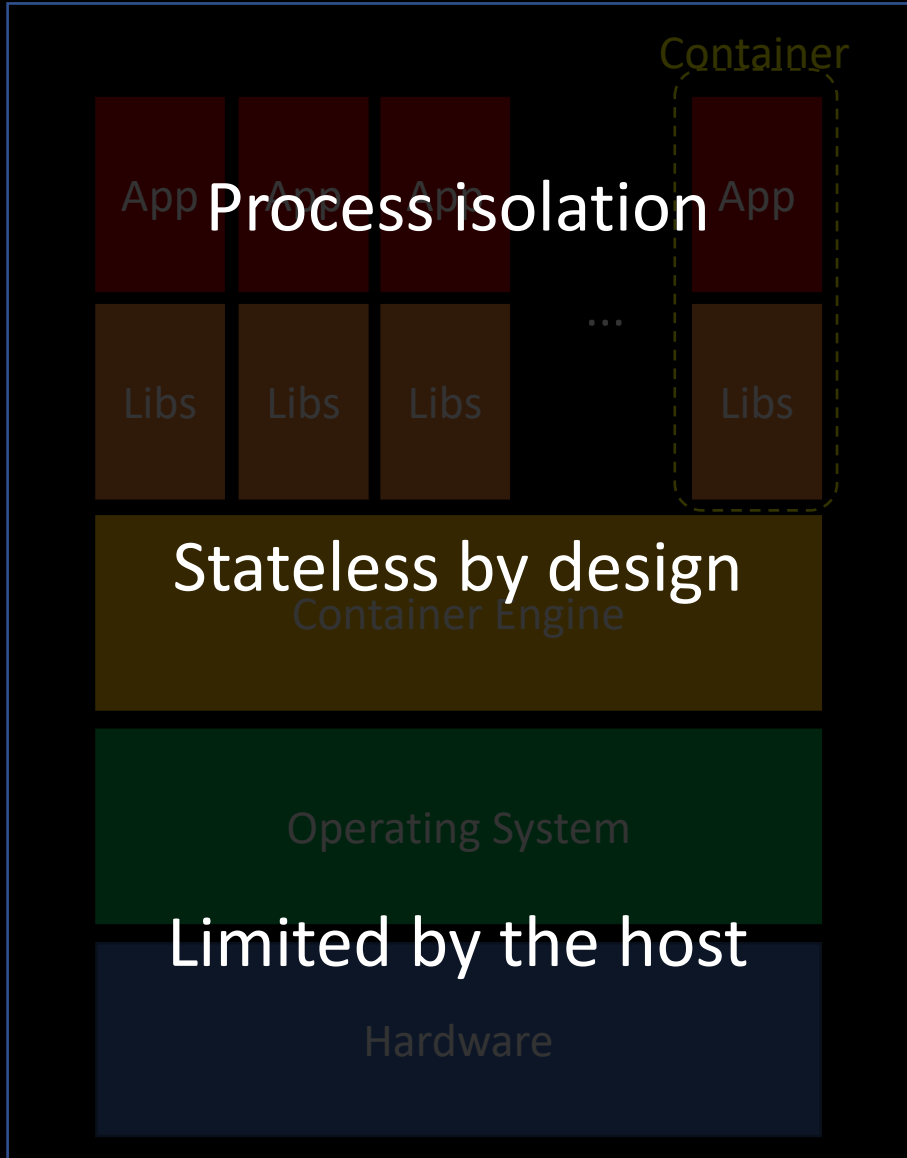
Containers and Kubernetes

Mattias Berg

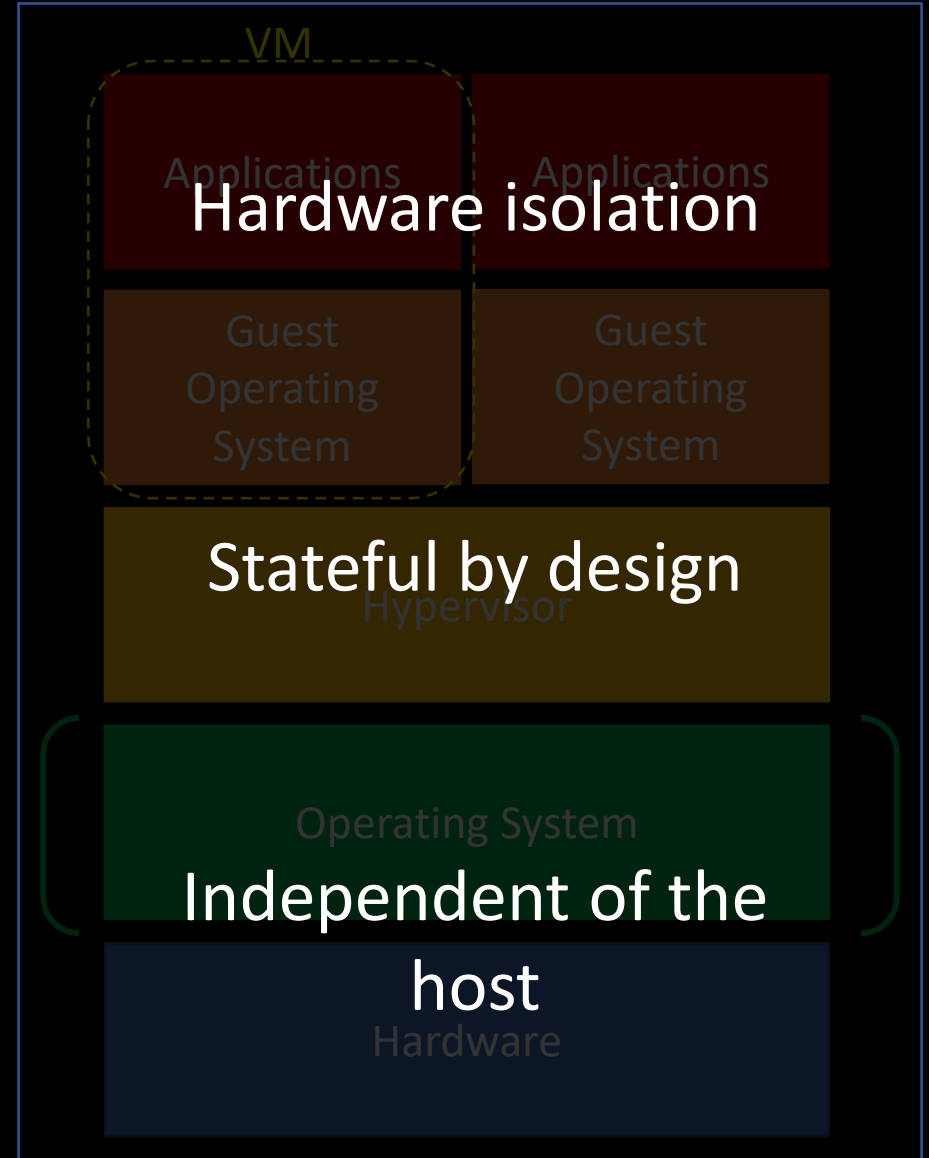
Coder, Scrum Master, DevOps, Architect



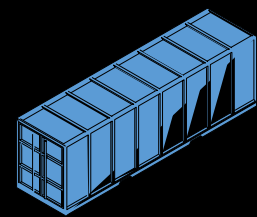
Containers



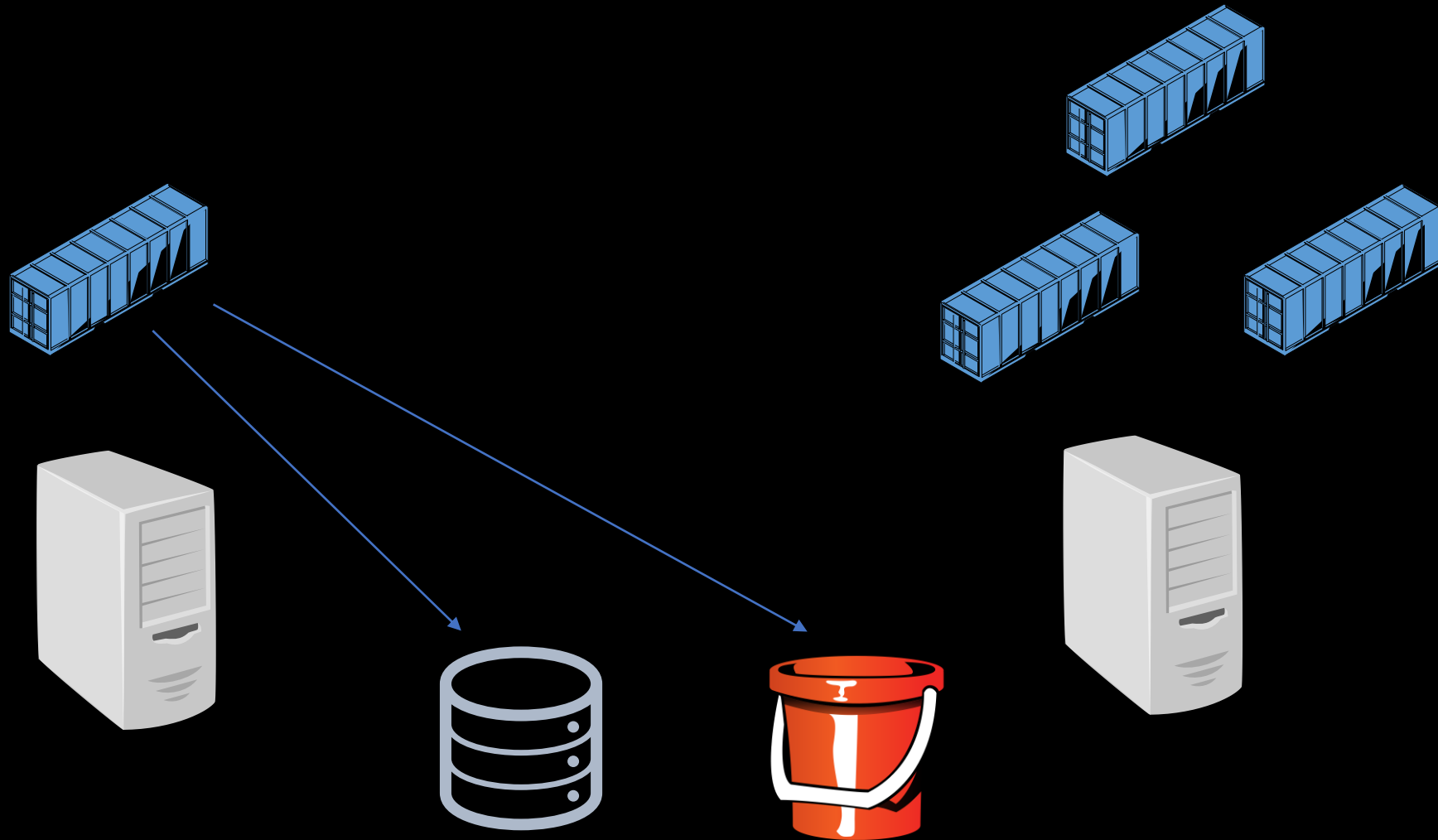
Virtual Machines



Statelessness



Statelessness



A yellow shipping container is shown against a blue sky with a white cloud. The container is the right half of the image, and the sky with the cloud is the left half. The text "How do I get a container?" is overlaid in white on the blue sky area.

How do I get a container?

Dockerfile

my-node-app/Dockerfile

```
FROM node:12-alpine # Base image
RUN apk add --no-cache python2 g++ make # Dependencies
WORKDIR /app # Somewhere sensible
COPY . . # Puts your code to /app
RUN yarn install --production # Each keyword creates a layer
CMD ["node", "src/index.js"] # When the container starts
EXPOSE 3000 # Help the user
```

\$ docker build -t my-first-app .

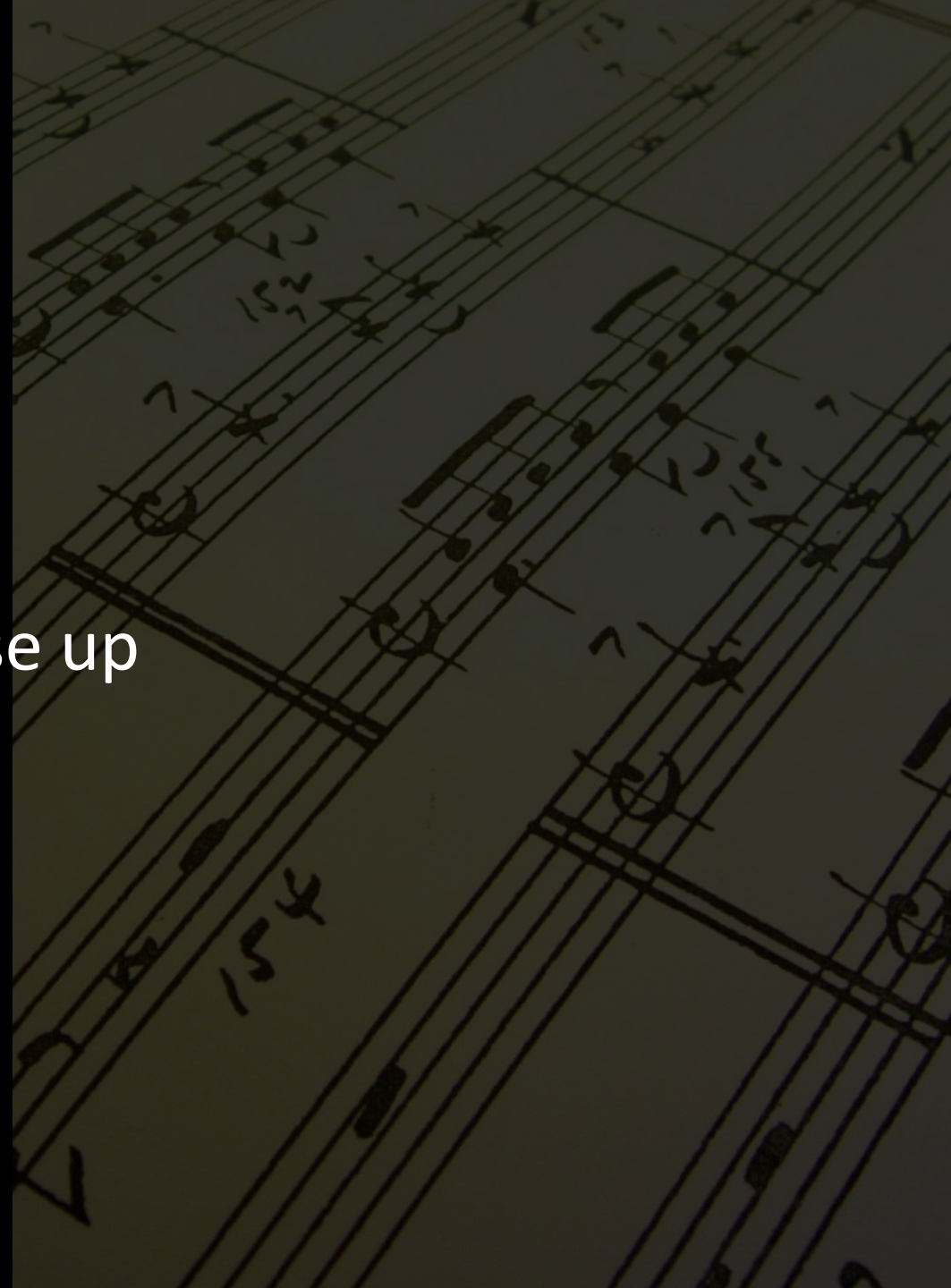
```
$ docker run -d my-first-app
```


Composing containers

my-django-app/docker-compose.yml

```
services:  
  db:  
    image: postgres  
    volumes:  
      - ./data/db:/var/lib/postgresql/data  
    environment:  
      - POSTGRES_DB=postgres  
      - POSTGRES_USER=postgres  
      - POSTGRES_PASSWORD=Postgre5  
  web:  
    image: my-django-app  
    ports:  
      - "8000:8000"  
    environment:  
      - POSTGRES_NAME=postgres  
      - POSTGRES_USER=postgres  
      - POSTGRES_PASSWORD=Postgre5
```

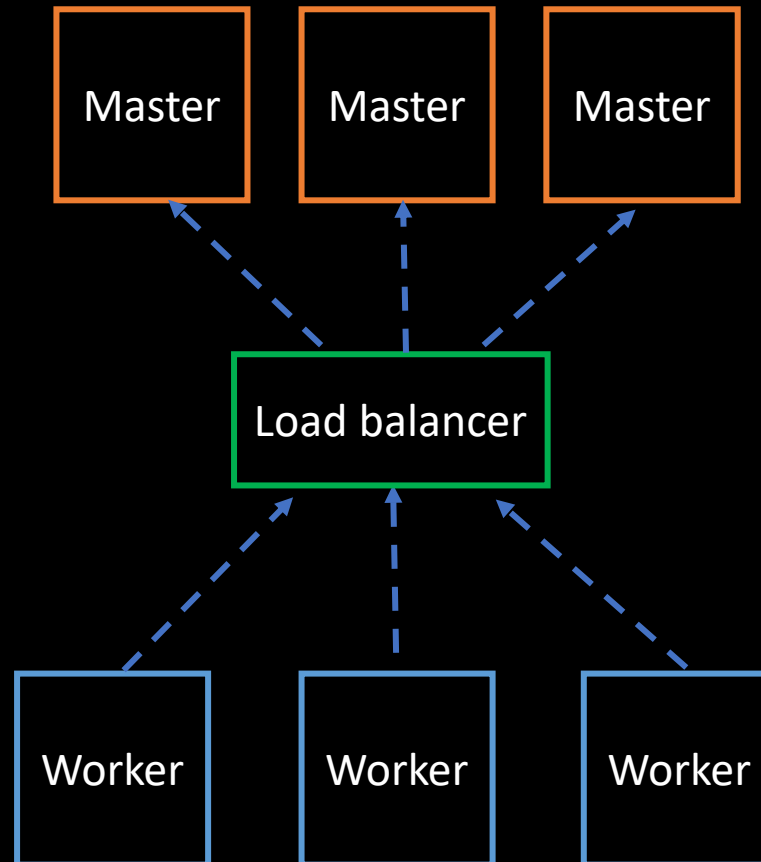
\$ docker-compose up



A close-up, low-angle shot of several violins and their bows, arranged in a row. The violins are made of polished wood and have dark, curved bodies. The bows are dark and have four strings. The background is dark and out of focus. The word "Orchestration" is overlaid in white text in the center of the image.

Orchestration

Kubernetes topology



Manifests

apps/nginx-app.yaml

kind: Service

kind: Deployment

```
apiVersion: v1
kind: Service
metadata:
  name: my-nginx-svc
  labels:
    app: nginx
spec:
  type: LoadBalancer
  ports:
  - port: 80
  selector:
    app: nginx
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-nginx
  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
```

Manifests

apps/nginx-app.yaml

```
spec:  
  type: LoadBalancer  
  ports:  
  - port: 80  
  selector:  
    app: nginx
```

```
apiVersion: v1  
kind: Service  
metadata:  
  name: my-nginx-svc  
  labels:  
    app: nginx  
spec:  
  type: LoadBalancer  
  ports:  
  - port: 80  
  selector:  
    app: nginx  
---  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: my-nginx  
  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
```

Manifests

apps/nginx-app.yaml

```
labels:  
  app: nginx
```

```
apiVersion: v1  
kind: Service  
metadata:  
  name: my-nginx-svc  
  labels:  
    app: nginx  
spec:  
  type: LoadBalancer  
  ports:  
  - port: 80  
  selector:  
    app: nginx  
---  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: my-nginx  
  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
```

Manifests

apps/nginx-app.yaml

```
spec:  
  replicas: 3
```

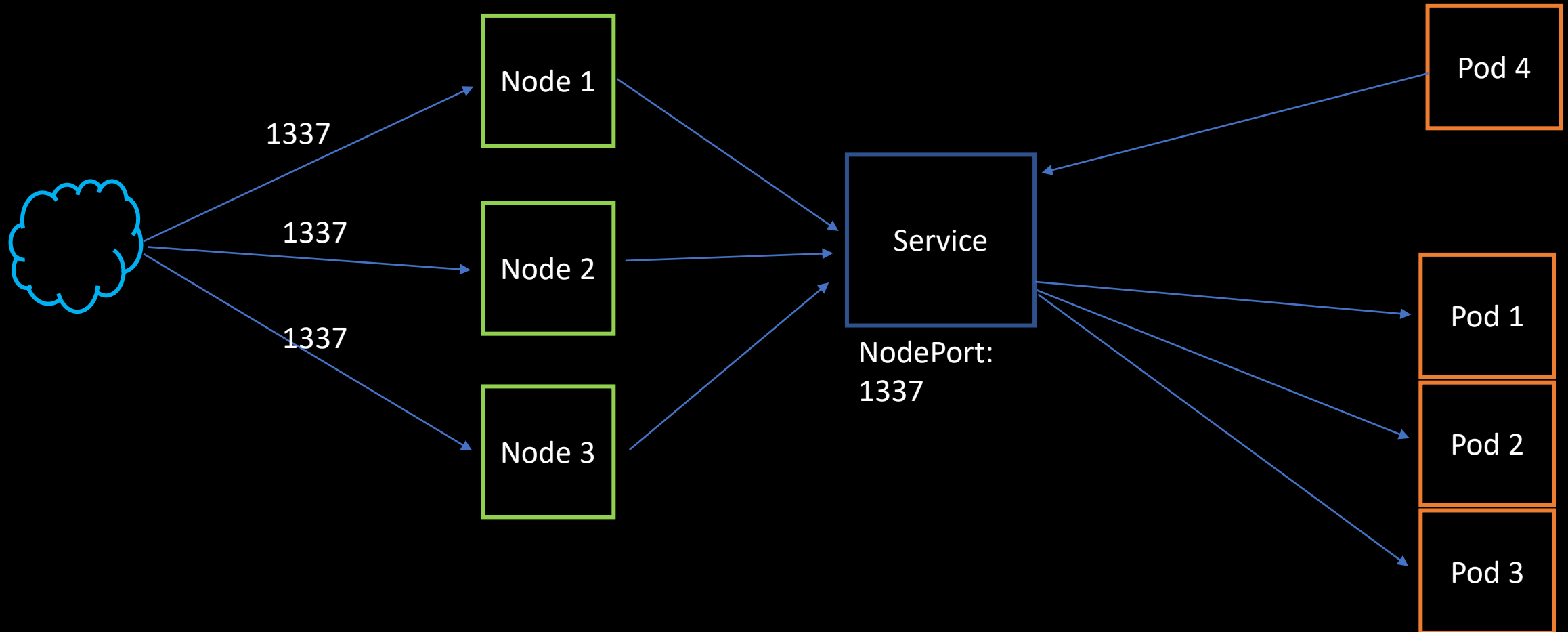
```
spec:  
  containers:  
    - name: nginx  
      image: nginx:1.14.2  
      ports:  
        - containerPort: 80
```

```
apiVersion: v1  
kind: Service  
metadata:  
  name: my-nginx-svc  
labels:  
  app: nginx  
spec:  
  type: LoadBalancer  
  ports:  
    - port: 80  
  selector:  
    app: nginx  
---  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: my-nginx  
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
```

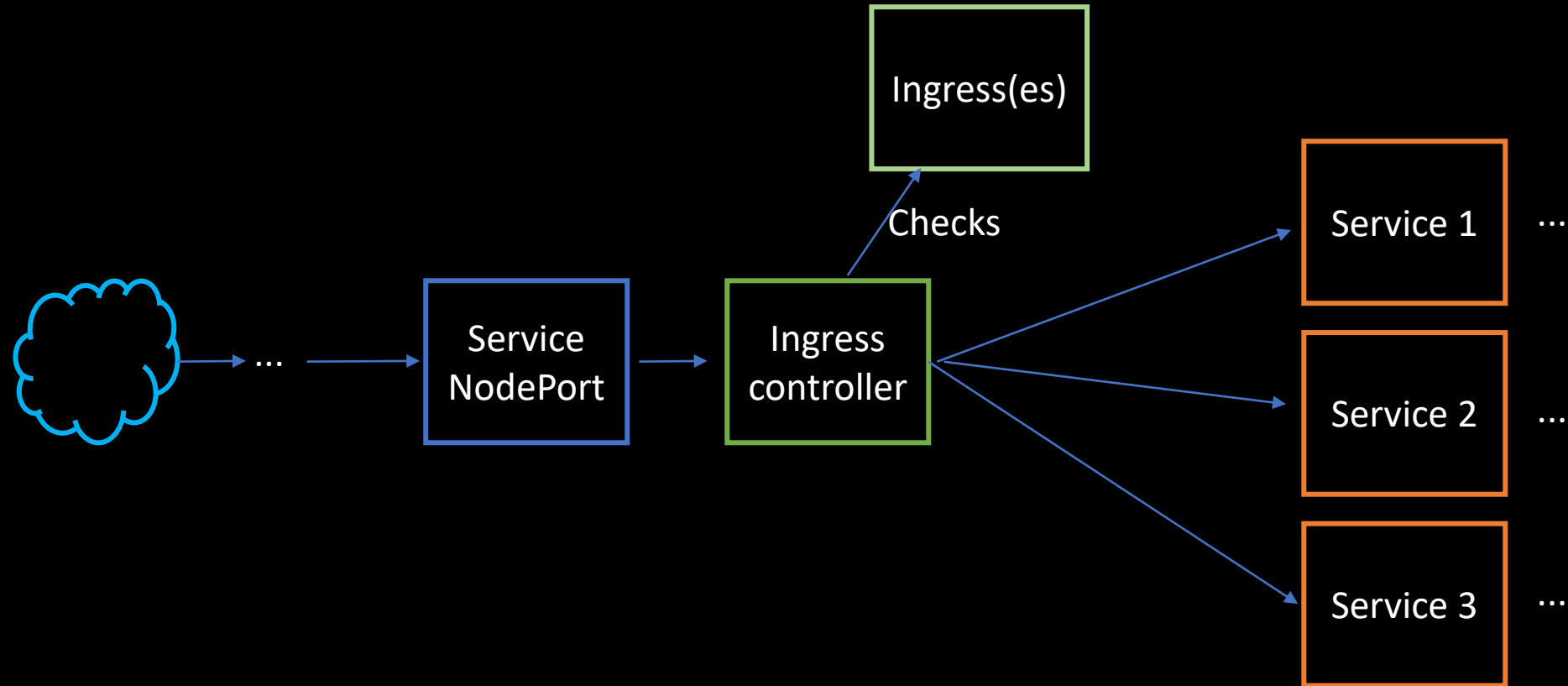


Self-healing

Communication is key



Can we do better?



Summary

- Containers != Virtual Machines
- Dockerfile -> Image -> Container
- Docker compose helps you spin up multiple containers
- Kubernetes is basically a bunch of servers and DNS
- Declare your desired state and let the computer handle the hard part
- Self healing and high availability
- Most problems are solved with good communication

Thanks!