



WTR

WORKSHOP
DE TECNOLOGIA DE REDES DO POP-RS

> 2021

Quero migrar
minha infra para a
nuvem:
Por onde começar?

21
SET

César Augusto Hass Loureiro



- Mover máquinas virtuais de sua infra para a nuvem é fácil e rápido. Mas o custo é proibitivo
- O que é preciso entender e por onde deve-se iniciar o aprendizado para realizar a migração para as nuvens (minha visão)

Assuntos abordados

- Conceitos referentes a nuvens e serviços
- O que são containers?
- Orquestração de containers
- Arquitetura dos kubernetes - K8S



WTR

WORKSHOP
DE TECNOLOGIA DE REDES DO POP-RS

> 2021

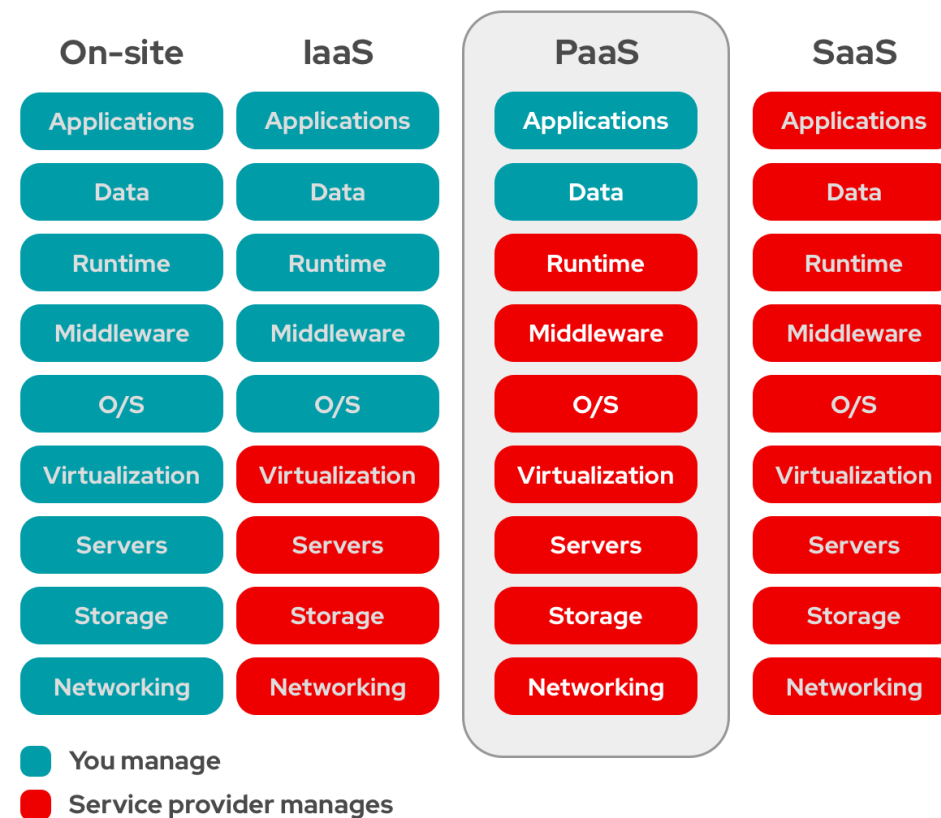
22
SET

Nuvens e serviços



- **Nuvem Privada (on-premises)**
 - OpenStack, XenServer, Vmware, Docker, Kubernetes
- **Nuvem Pública**
 - Amazon Web Services (AWS), Google Cloud, IBM Cloud, Microsoft Azure, etc.
- **Nuvens Híbridas**
 - Quando aplicações, serviços ou VMs podem se mover por vários ambientes diferentes, conectados entre si.

- **On-site**
 - Servidores em sua infra
- **IaaS (infraestrutura como serviço)**
 - Servidores virtuais na Nuvem, onde você é responsável pela gerência das máquinas virtuais (AWS EC2)
- **PaaS (Plataforma como serviço)**
 - **Containers** na Nuvem (Docker, Kubernetes, etc)
- **SaaS (Software como Serviço)**
 - Aplicações em nuvem (Mysql, MongoDB, Wordpress, etc)



fonte: <https://www.redhat.com/pt-br/topics/cloud-computing/iaas-vs-paas-vs-saas>

- **MS Azure**
 - Azure Container Service
- **Google**
 - Google Container Engine
- **Amazon**
 - Amazon Elastic Kubernetes Service (Amazon EKS)
- **RNP**
 - NasNuvens



WORKSHOP
DE TECNOLOGIA DE REDES DO POP-RS

> 2021

22
SET

Containers

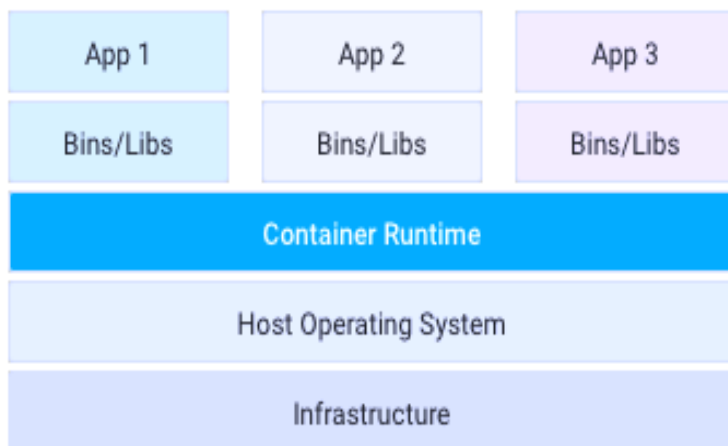


- Um container é um unidade padrão de software que empacota o código e todas as suas dependências para que um aplicativo seja executado de forma isolada.
- Inclui tudo o que é necessário para executar o aplicativo: código, bibliotecas e configurações.
- Vários containers podem ser executados por um único Sistema Operacional, isso é, são mais leves do que as máquinas virtuais.

Containers x máquinas virtuais

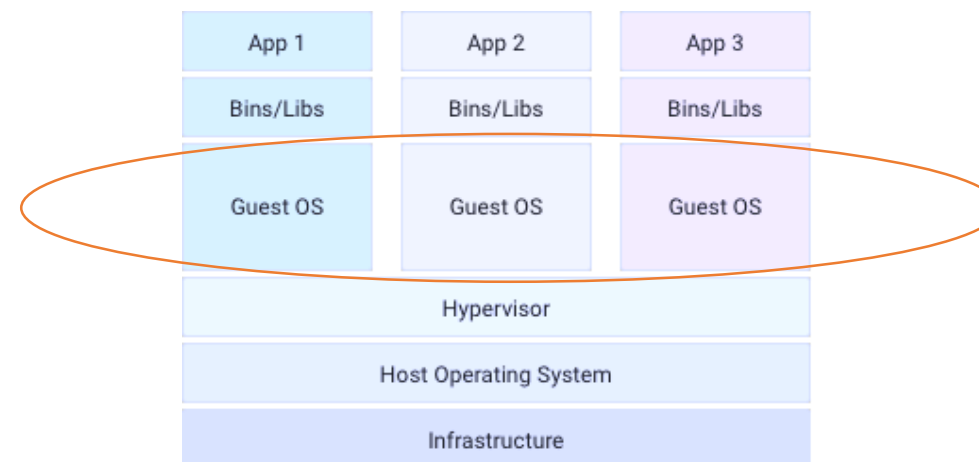
■ Containers

- Kernel do host compartilhado
- Portabilidade
- Escalabilidade mais rápida



■ Virtual Machines

- Sistema Operacional separado por instâncias



Explicando a evolução containers

- **Jail (chroot)**

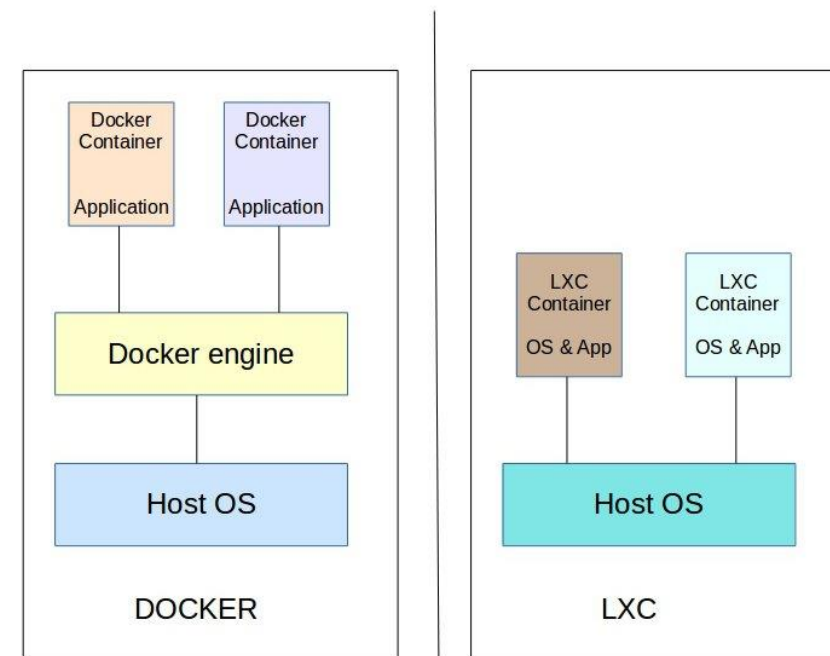
Desde os primórdios *unix possui a capacidade de isolar um processo e uma área do disco do restante do Sistema Operacional para que, em caso de falha ou invasão ao serviço, o acesso indevido ficasse restrito a área do disco e da memória definida no Jail.

- **LXC (Linux Containers)**

Possibilidade de isolar processos e área do disco no Linux, através de containers, desenvolvido em 2008 e posteriormente implantado no kernel do Linux (namespaces, cgroups, chroot, SELinux e apparmor).

- **Docker**

Criado em 2013, é uma evolução (ou não) do LXC, transformando o container em uma imagem, contendo tudo que é necessário para execução do serviço e possibilitando a execução do container em qualquer sistema hospedeiro (Host OS)



<https://bobcares.com/blog/lxc-vs-docker/>

- Plataforma de código aberto, desenvolvido na linguagem Go para criar, testar e implementar aplicações em um ambiente controlado.
- implementa a ideia de **microserviço** onde cada container realiza uma parte do todo.
- Exemplo: uma aplicação em PHP que tem 3 módulos (fornecedores, produtos e clientes), poderia ser dividida em 5 containers, cada um responsável por um microserviço.
 - Apache
 - PHP - módulo clientes
 - PHP - modulo produtos
 - PHP - modulo fornecedores
 - My-SQL

- Escalabilidade
 - De acordo com a quantidades de acessos, podem ser instanciados mais containers.
 - Exemplo: triplicar a quantidade de containers com Apache
- Portabilidade
 - É possível mover o container para outras máquinas sem a necessidade de novas instalações
- Isolamento
 - Como cada container realiza apenas um serviço, facilitando a gestão de **segurança** e a **manutenção** do serviço

- Devido a quantidade de aplicações/serviços e a granularidade dos mesmos em microserviços, é necessário uma ferramenta que gerencie os containers. Quais tem acesso externo? Quais portas estão abertas? Quantas instancias de cada container está em execução? Como realizo um rollback de uma aplicação que está com problema? Como atualizo n instâncias de uma aplicação sem que ela saia de execução?

**** Resolver esses problemas é que a orquestração de containers promete ****

- **Kubernetes**
 - Open Source, desenvolvido pela Google e doado para o Cloud Native Computing Foundation
- **Docker Swarm**
 - Orquestrador de containers da Docker inc.
- **OpenShift**
 - Mantido pela Red Hat e também possui sua versão open source
- **AWS ECS - Elastic Container Service**
 - Baseado nos kubernetes
 - Integra com os demais serviços da Amazon DNS, balanceamento, segurança, monitoramento ...



WTR

WORKSHOP
DE TECNOLOGIA DE REDES DO POP-RS

> 2021

22 SET

kubernetes



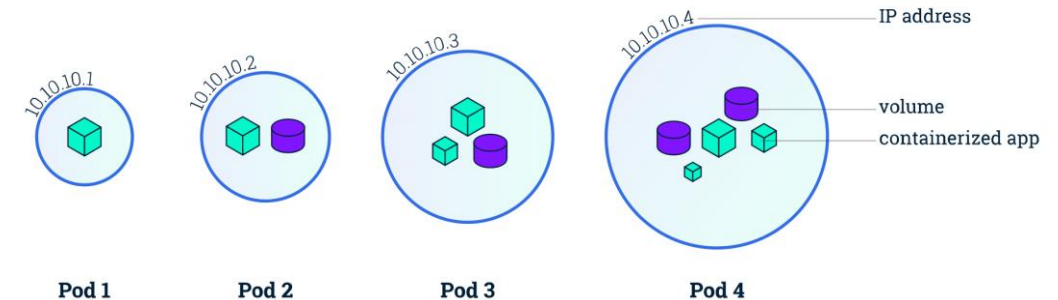
- Provê a orquestração dos containers, de rede, dos volumes (discos) e dos demais recursos para que as aplicações executem com disponibilidade e escalabilidade
- Não possui nenhum run-time nativo: Precisa que seja instalado no mínimo um gerenciador de rede e o run-time dos containers (Docker?, ContainerD?, CRI-O?)

Kubernetes

Principais objetos

- **Pod**

- Menor unidade de computação
- Consiste em um objeto que executa uma instância de uma aplicação. Possui um endereço IP interno único e pode possuir acesso a volumes e outros objetos.



- **Deployment**

- O Deployment define a criação das instâncias do seu aplicativo (Pods). Depois de criar um Deployment, o Master do Kubernetes agenda a criação dos Pods nos nós do Cluster.



Kubernetes

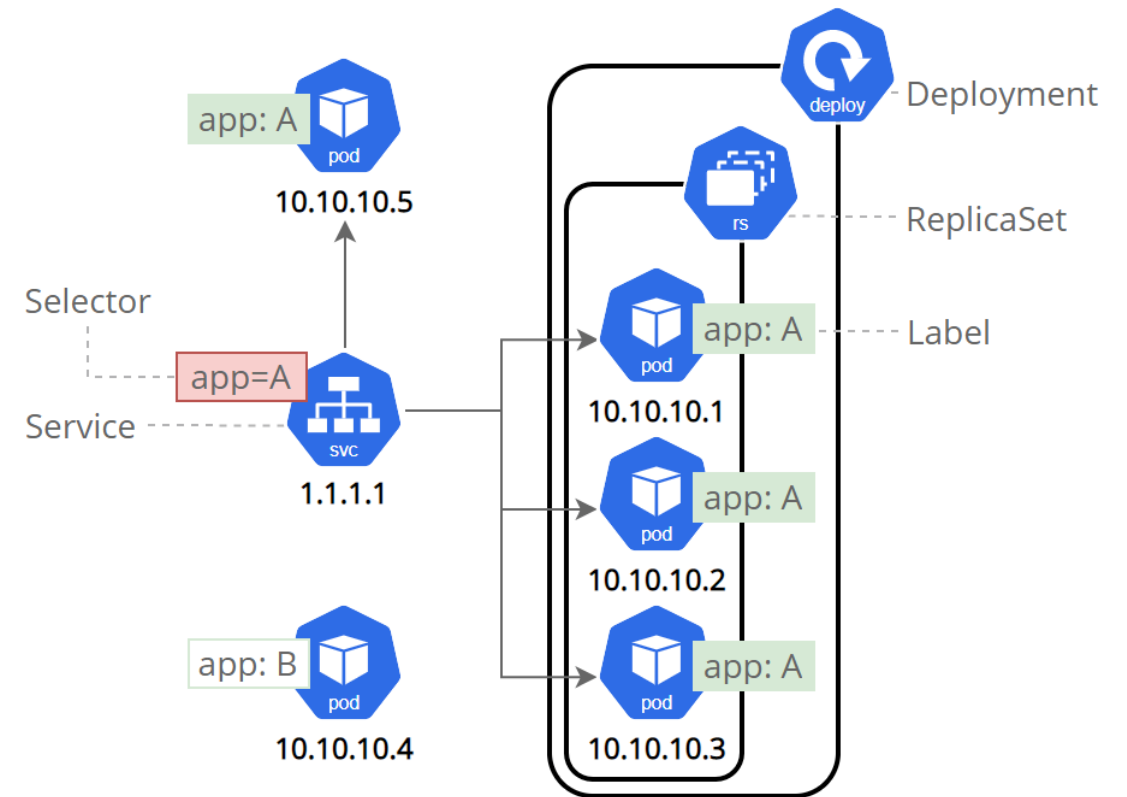
Principais objetos

- **Services**

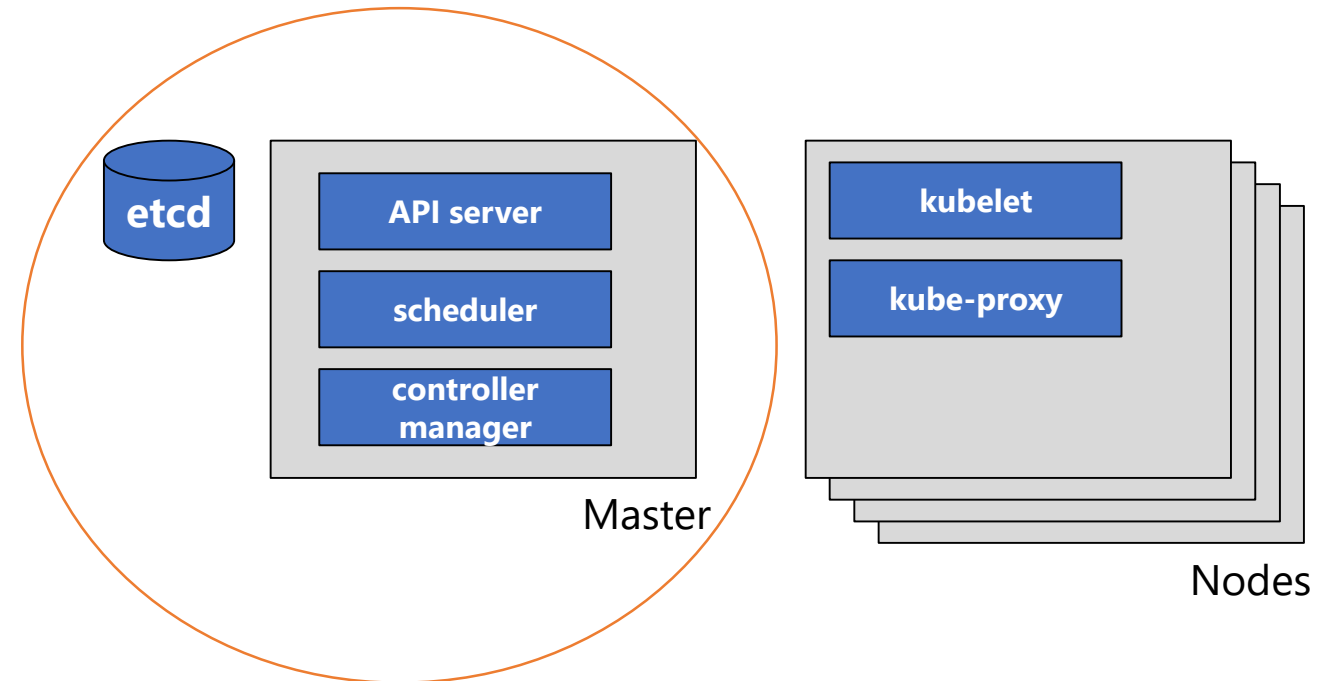
- Um serviço no Kubernetes é uma abstração que define uma política para acessar determinados Pods. Realiza a atribuição de IP/porta para a aplicação ser acessada.

- **Exemplos de outros objetos:**

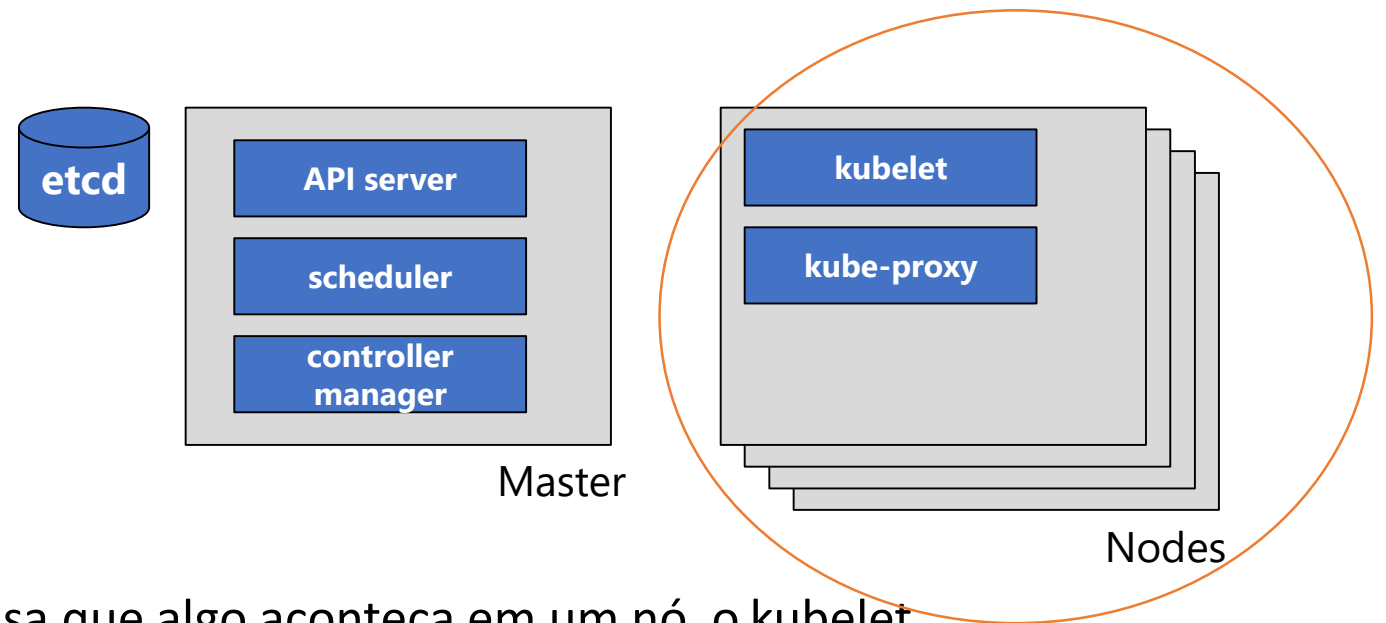
- Volumes, ConfigMap



- **etcd**: é um BD de “chave:valor” com todas as informações do cluster, como o estado dos serviços e as configurações



- **API server**: front-end do plano de controle. Ela processa solicitações internas e externas
- **scheduler**: Ele analisa quais os recursos de que um pod necessita, como CPU e memória e aloca o POD a algum node
- **controller manager**: executa o controle do cluster, verifica se os pods estão em execução, se os serviços estão disponíveis, etc.



- **Kubelet:** Quando o plano de controle precisa que algo aconteça em um nó, o kubelet realiza a ação
- **kube-proxy:** encaminhamento de pacotes TCP/UDP interno e externo

Execução de aplicações em kubernetes

```
D:\Users\Cesar Loureiro>kubectl get services -o wide
NAME                TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE   SELECTOR
kubernetes          ClusterIP      10.96.0.1       <none>           443/TCP          18d   <none>
meuapache-service3  LoadBalancer  10.110.246.4    200.132.0.75    80:31543/TCP    17d   app=meuapache
nginx               ClusterIP      10.109.220.40   200.132.0.77    80/TCP,443/TCP  9d    app=nginx,tier=backend
php                 ClusterIP      10.105.199.85   <none>           9000/TCP         9d    app=php,tier=backend

D:\Users\Cesar Loureiro>kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP           NODE           NOMINATED NODE   READINESS GATES
meu-nginx-5d464b9b67-m75pk  1/1     Running   1          9d    10.38.0.6    k8s-node02    <none>           <none>
meuapache-deployment-6dc575d4cd-fmm8m  1/1     Running   0          3d19h  10.40.0.5    k8s-node01    <none>           <none>
meuapache-deployment-6dc575d4cd-mc26t  1/1     Running   0          3d19h  10.38.0.7    k8s-node02    <none>           <none>
meuapache-deployment-6dc575d4cd-tj99d  1/1     Running   0          3d19h  10.40.0.6    k8s-node01    <none>           <none>
php-7bcf88555c-tngxk        1/1     Running   1          9d    10.40.0.2    k8s-node01    <none>           <none>

D:\Users\Cesar Loureiro>kubectl get deploy -o wide
NAME                READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS          IMAGES                SELECTOR
meu-nginx           1/1     1             1           9d    nginx               nginx:1.7.9           app=nginx,tier=backend
meuapache-deployment  3/3     3             3           17d   meuapache-container cesarloureiro/aplicacao:v8  app=meuapache
php                 1/1     1             1           9d    php                 php:7-fpm              app=php,tier=backend
```

Exemplo de YAML para deploy de uma aplicação e um serviço

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: meuapache-deployment
  labels:
    app: meuapache
spec:
  replicas: 6
  selector:
    matchLabels:
      app: meuapache
  template:
    metadata:
      labels:
        app: meuapache
    spec:
      containers:
      - name: meuapache-container
        image: cesarloureiro/aplicacao:v8
        ports:
        - containerPort: 80
      imagePullSecrets:
      - name: regcred
```

deploy-meuapache.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: meuapache-service3
spec:
  selector:
    app: meuapache
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: LoadBalancer
  externalIPs:
  - 200.132.0.75
```

service-meuapache.yaml

Quero ir para a nuvem: Por onde começar?

- **Entenda os conceitos**
 - Crie sua própria “nuvem”
 - Se familiarize com as ferramentas
 - Migre **uma** aplicação -- Seu “Hello World” ---
 - Realize um levantamento de seus serviços
 - Avalie os recursos e os custos das soluções disponíveis no mercado
 - Treine sua equipe na plataforma escolhida
 - Migre uma aplicação para a plataforma escolhida
 - Contrate um serviço
- } Assuntos da oficina do WTR

Obrigado!

César Loureiro

cesar.loureiro@pop-rs.rnp.br



APOIO



REALIZAÇÃO



MINISTÉRIO DO
TURISMO

MINISTÉRIO DA
DEFESA

MINISTÉRIO DA
SAÚDE

MINISTÉRIO DAS
COMUNICAÇÕES

MINISTÉRIO DA
EDUCAÇÃO

MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÕES

