EdgeX Geneva 在Kubernetes的部署实践

张丽斌
Developer
zlibin@vmware.com

https://github.com/DaveZLB/edgex-helm

vmware

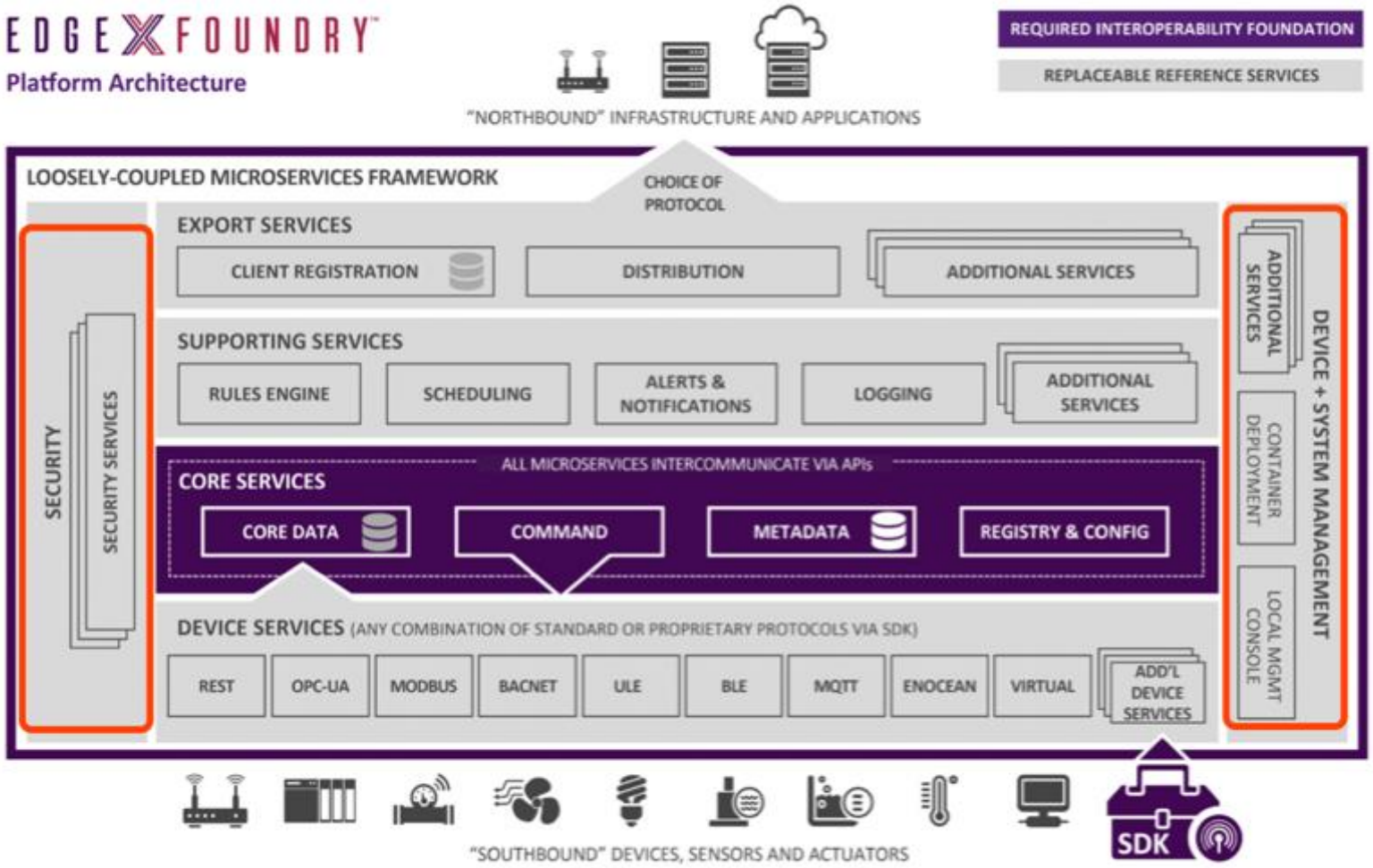
©2019 VMware, Inc.

Agenda

- 定制化EdgeX Kubernetes应用
- EdgeX Kubernetes 部署实践

定制EdgeX kubernetes 应用（一）

为什么要定制？



原因：

与Docker耦合度比较高

定制EdgeX kubernetes 应用（一）

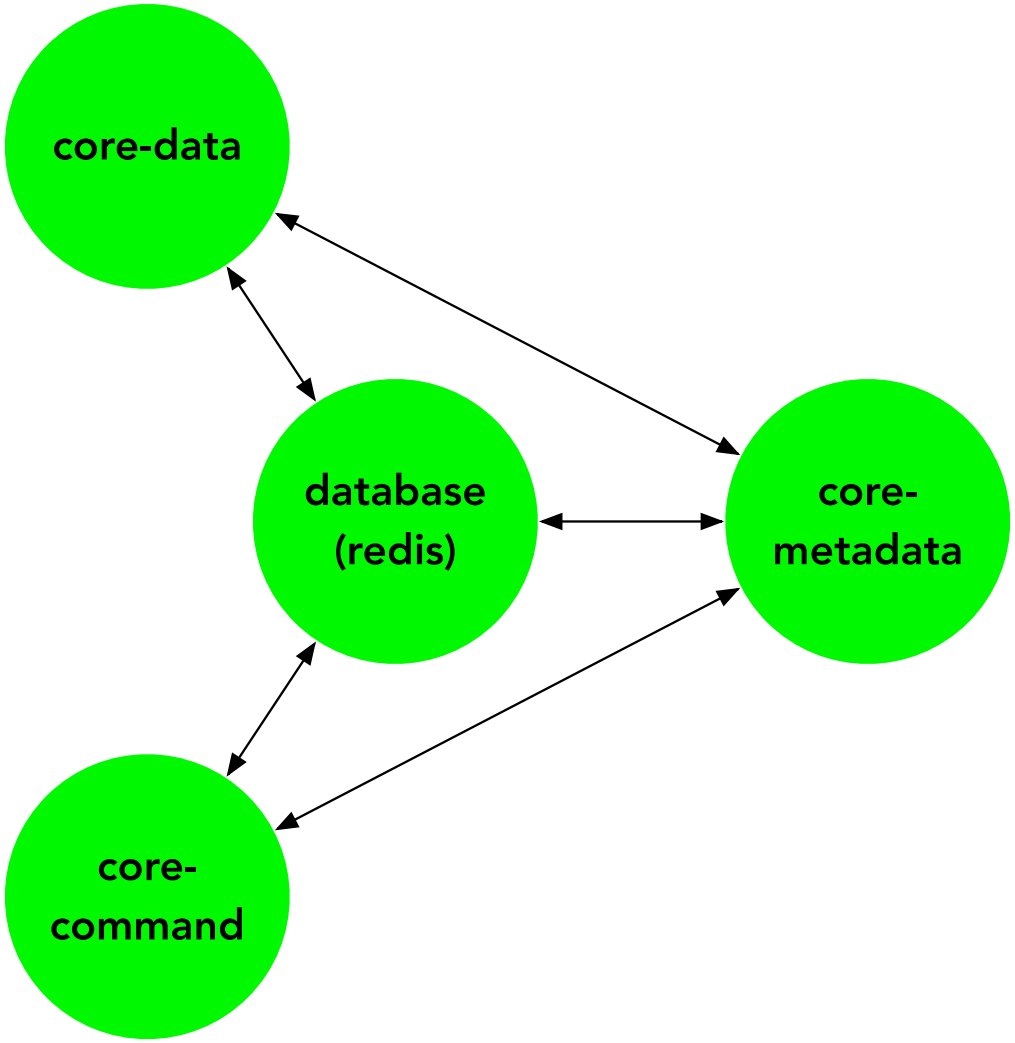
如何定制

基于Kubernetes环境

- 问题一：EdgeX服务哪些是必须的？哪些是可替换？高可用？
- 问题二：不熟悉EdgeX的参数配置？

定制EdgeX kubernetes 应用（一）

Mini 型 EdgeX kubernetes



参数：

- -cp : 注册中心consul地址
- --registry : 将自己注册到consul
- --confdir : 配置文件所在路径

Docker file:

- 从consul 加载配置(默认)

```
ENTRYPOINT ["/core-command","-
cp=consul.http://edgex-core-
consul:8500","--registry","--
confdir=/res"]
```

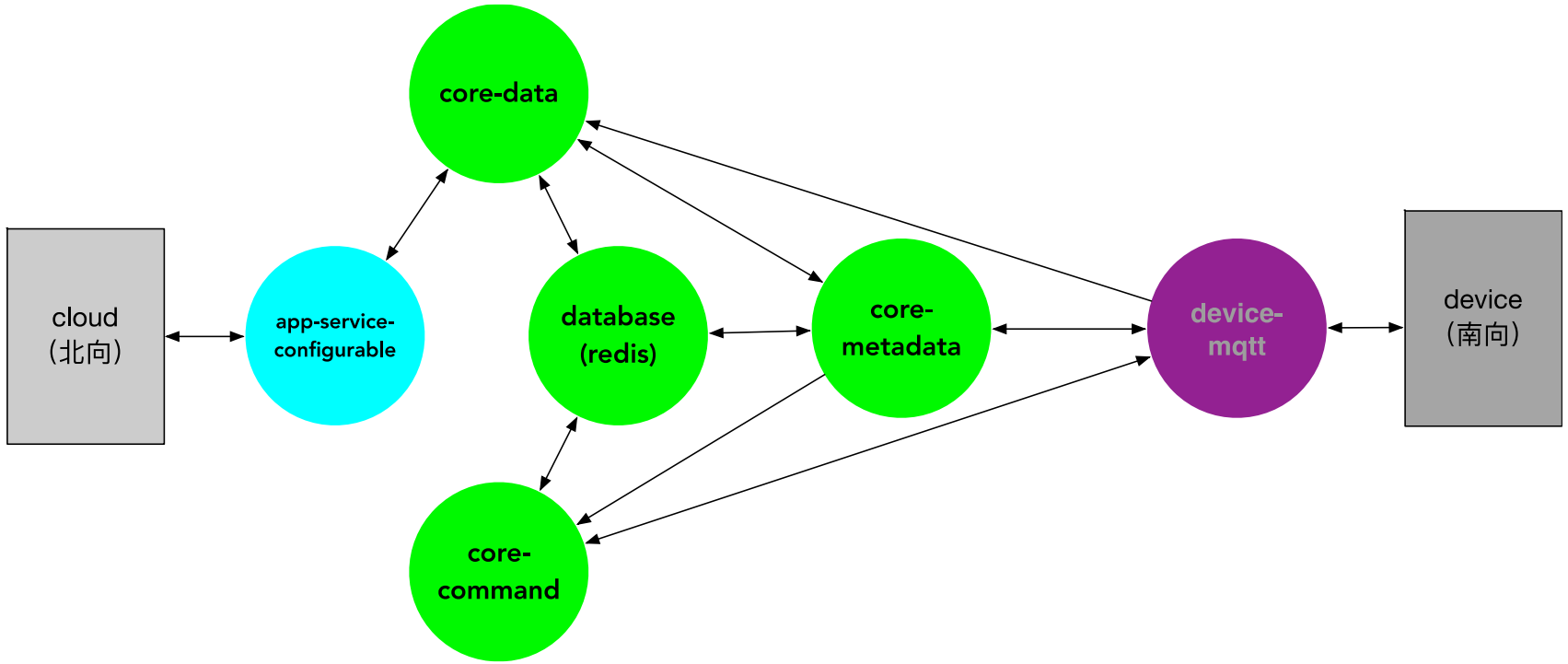
- 从configuration文件加载配置

```
ENTRYPOINT ["/core-command","-
-confdir=/res"]
```

定制EdgeX kubernetes 应用（一）

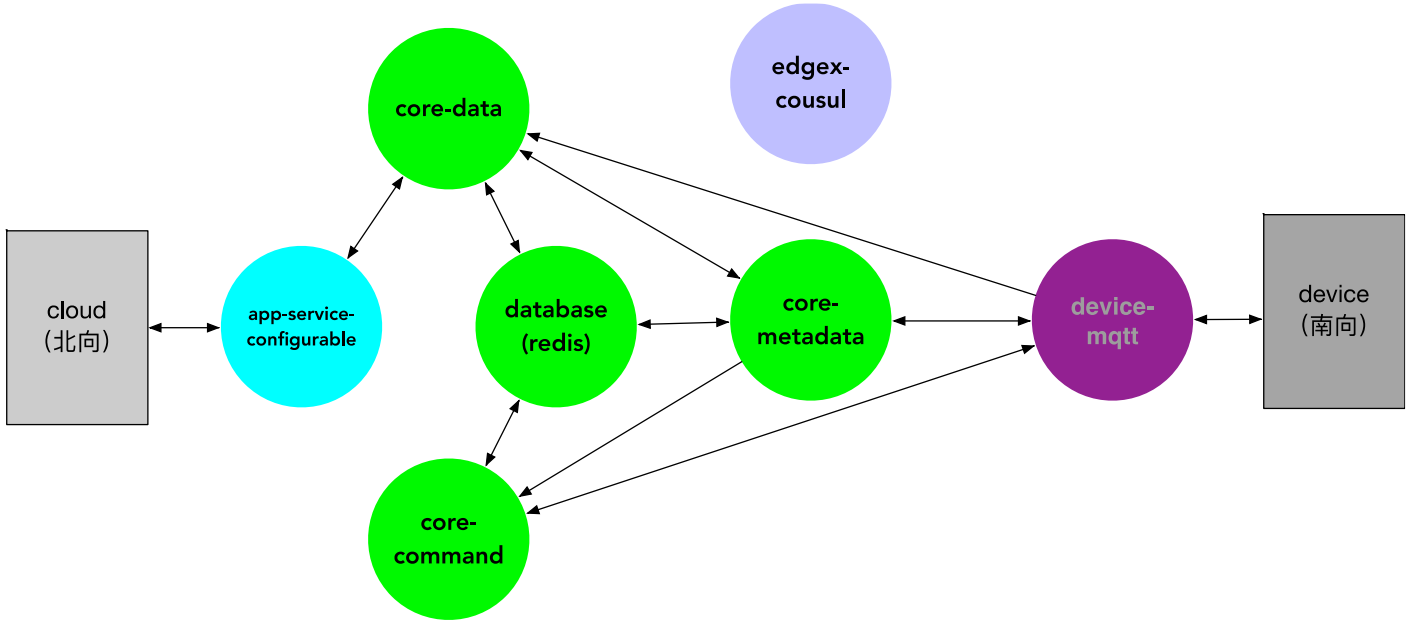
具有实际意义的EdgeX Kubernetes

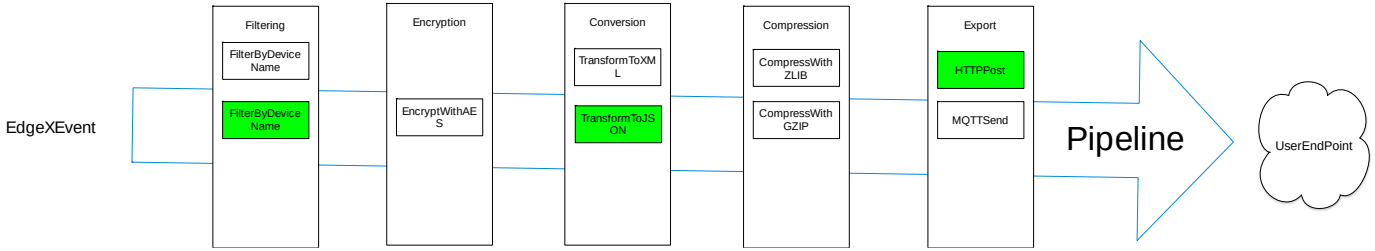
- 具备从设备端（南向）接收数据的能力
- 具备向云端（北向）推送数据的能力



定制EdgeX kubernetes 应用（一）

是否需要添加consul





Consul（Registry & Config）作用：

① 服务注册

② 配置中心

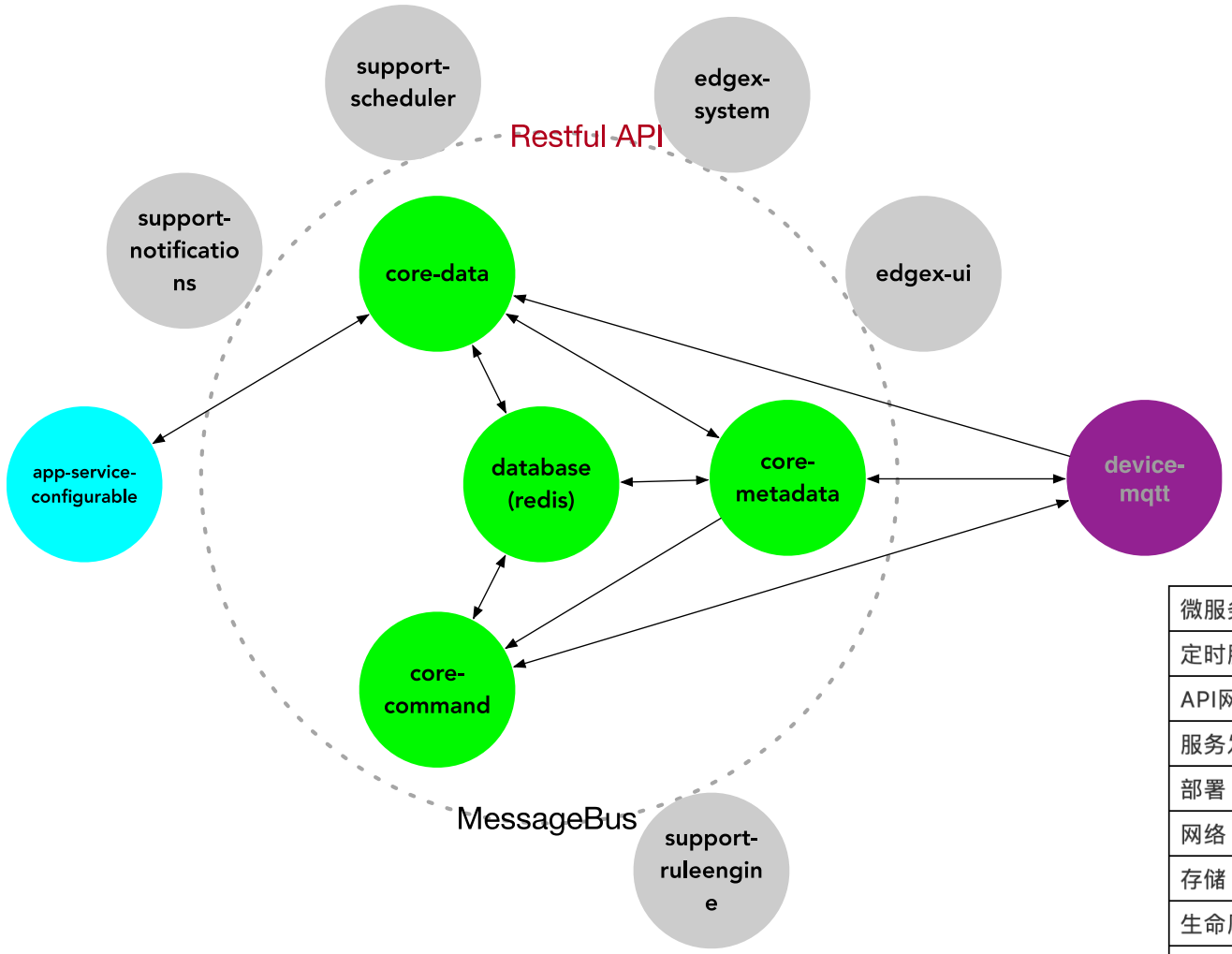
③ 可以使用pipeline动态配置app-service导出规则。

在kubernetes的替代方案：

- 使用Kubernetes Service + livenessProbe + readinessProbe 代替
- 使用 Kubernetes ConfigMap 代替
- 可以传递系统变量的方式来配置到处规则。

定制EdgeX kubernetes 应用（一）

看需求增加其他EdgeX服务

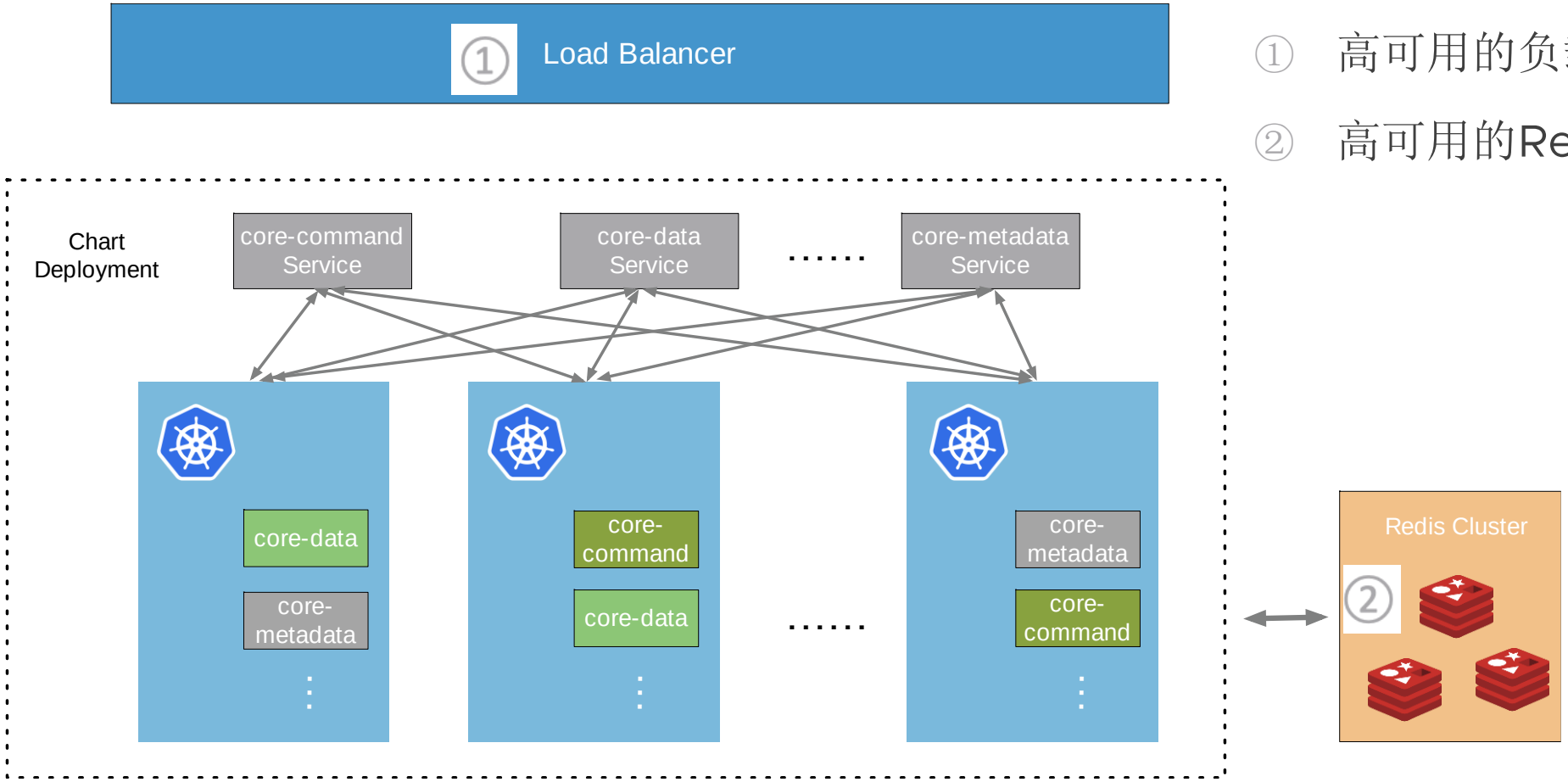


- support-scheduler： 定时任务
- support-notifications : 事件通知
- support-ruleengine：规则引擎(kuiper)，过滤导出数据
- edgex-system：EdgeX 系统管理，相当于Cli
- edgex-ui：基于EdgeX Restful API 的界面实现(dev/test)

微服务/模块	EdgeX Foundry on Docker	Kubernetes生态
定时服务	自制scheduler	Job/CronJob
API网关	Kong	Envoy
服务发现	Consul	CoreDNS
部署	Docker Compose	Helm Chart
网络		Flannel/Calico
存储	Volume	PV/CSI
生命周期管理		Operator
Service Mesh		Istio/Network Service Mesh

定制EdgeX kubernetes 应用（一）

高可用方案



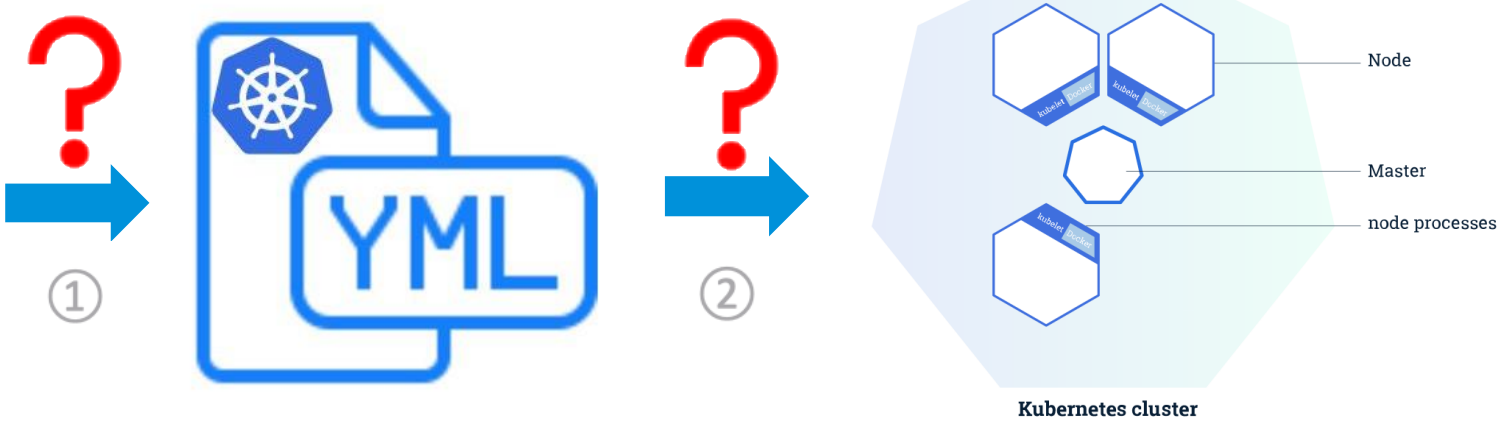
① 高可用的负载均衡器(Ingress)

② 高可用的Redis集群

EdgeX Kubernetes 部署实践（二）

二步走

```
  data:
    image: edgexfoundry/docker-core-data-go:1.2.1
    ports:
      - "127.0.0.1:48080:48080"
      - "127.0.0.1:5563:5563"
    container_name: edgex-core-data
    hostname: edgex-core-data
    networks:
      - edgex-network
    environment:
      <<: *common-variables
      Service_Host: edgex-core-data
    depends_on:
      - consul
#       - logging  # uncomment if re-enabled remote logging
      - redis
      - metadata
```



① 将Docker Compose 文件 转换成 Kubernetes 对象部署清单

② 使用Helm部署

EdgeX Kubernetes 部署实践（二）

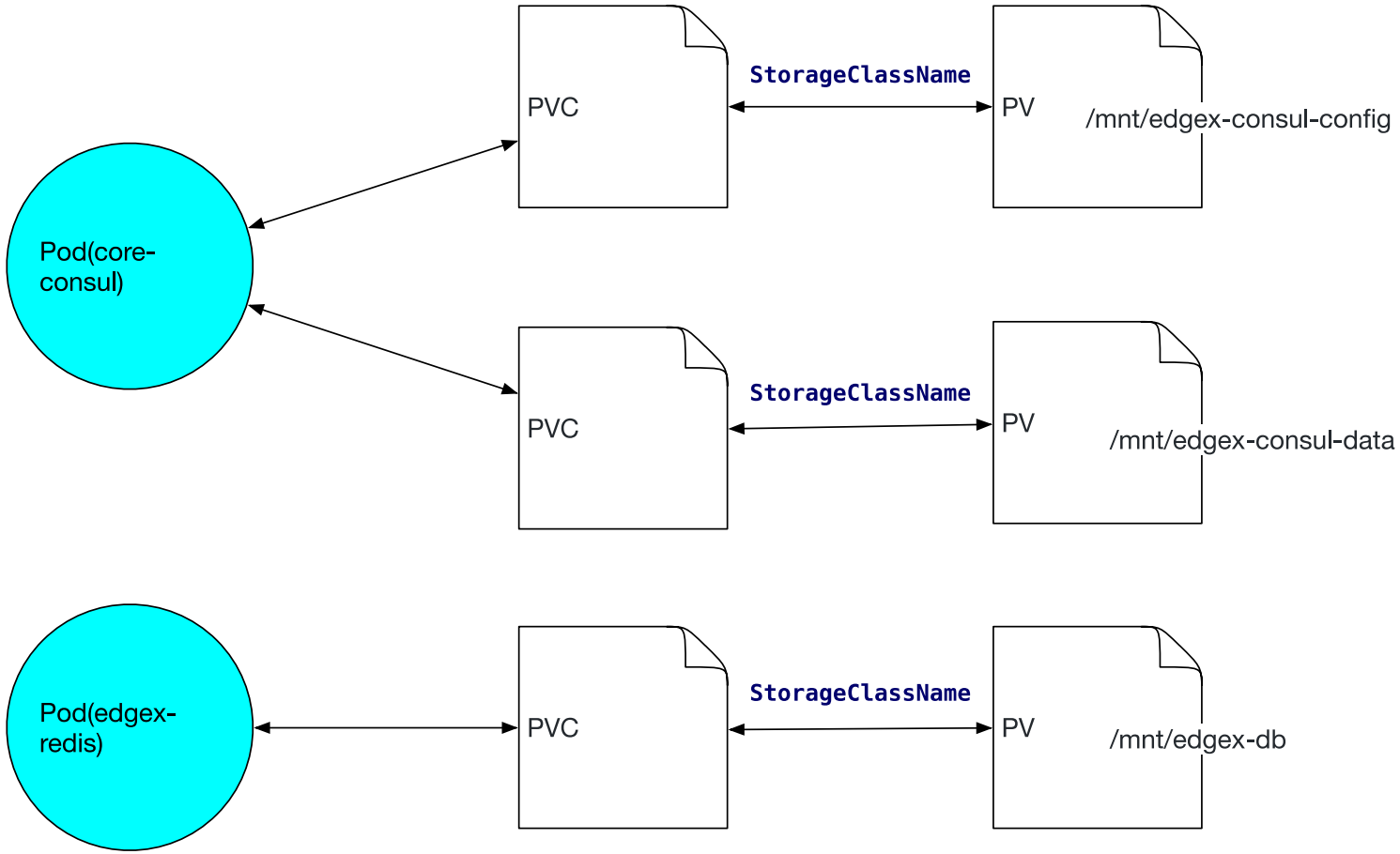
Kubernetes对象介绍

- Pod：Kubernetes创建或部署的最小对象
- Deployment： Pod的控制器
- Service ： 为Pod对外暴露服务
- PersistentVolume(PV)：类似Docker volume
- PersistentVolumeClaim(PVC)：用户对PV资源的请求
- ConfigMap：key-values,存储非机密数据

Docker	Kubernetes
Container	**Pod**
ContainerName(ContainerToContainer 通信**)**	**ServiceName**(PodToPod 通信)
Volume	**PersistentVolume/PersistentVolumeClaim**
Environment	ConfigMap

EdgeX Kubernetes 部署实践（二）

创建 EdgeX Volume部署清单



PV的类型

- GCEPersistentDisk
- AWSElasticBlockStore
- AzureFile
- AzureDisk
- CSI
- FC (Fibre Channel)
- FlexVolume
- Flocker
- NFS
- iSCSI
- RBD (Ceph Block Device)
- CephFS
- Cinder (OpenStack block storage)
- Glusterfs
- VsphereVolume
- Quobyte Volumes
- HostPath (Single node testing only - NOT WORK in a multi-node cluster)
- Portworx Volumes
- ScaleIO Volumes
- StorageOS

EdgeX Kubernetes 部署实践（二）

创建 Deployment 部署清单

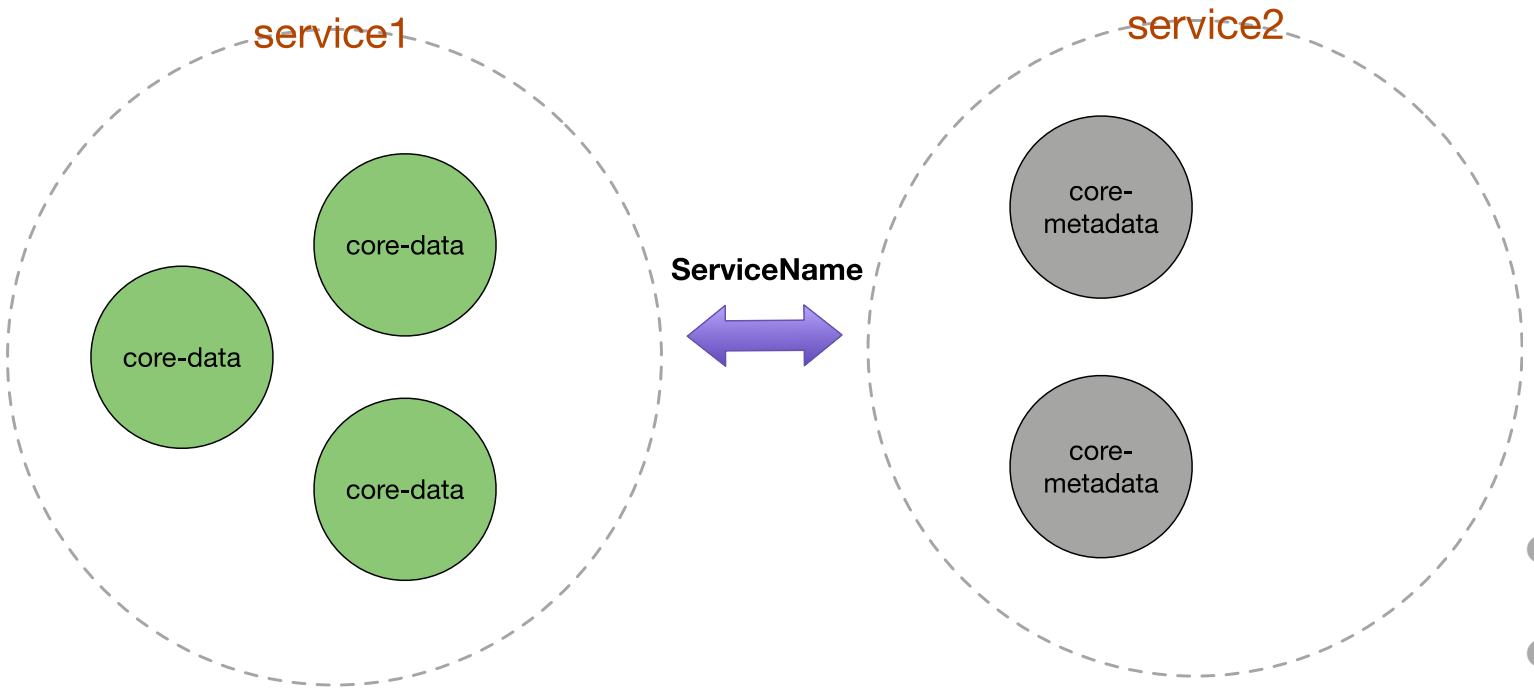
```
  data:
    image: edgexfoundry/docker-core-data-go:1.2.1
    ports:
      - "127.0.0.1:48080:48080"
      - "127.0.0.1:5563:5563"
    container_name: edgex-core-data
    hostname: edgex-core-data
    networks:
      - edgex-network
    environment:
      <<: *common-variables
      Service_Host: edgex-core-data
    depends_on:
      - consul
#       - logging  # uncomment if re-enabled remote logging
      - redis
      - metadata
```

Kubernetes Deployment .yaml

- pod 副本数 ： spec.replicas
- 创建Service标签 ： spec.matchLabels
- Container配置： spec.containers
- 健康检测 / 微服务可用:
  spec.livenessProbe
  /spec.readinessProbe
- 硬件资源配置 ： spec.resources

EdgeX Kubernetes 部署实践（二）

创建 Kubenetes Service部署清单



- Service通过标签选择器选定一组Pod，为其提供服务暴露能力.

Service类型

- ClusterIP： 集群内部访问
- NodePort：<NodeIP>:<NodePort>
- LoadBlancer：<ExternalIP>:<NodePort>

EdgeX Kubernetes 部署实践（二）

Kubectl部署方式

① 创建PersistentVolume（PV）

② 创建PersistentVolumeClaim（PVC）

③ 创建Services

④ 创建Deployments

① kubectl create –f edgex-pv.yml

② kubectl create –f edgex-pvc.yml

③ kubectl create –f edgex-core-data-deployment.yml

④ kubectl create -f edgex-core-data-service.yml

- 需要考虑Kubernetes 对象创建的顺序
- 需要执行多次kubectl create 命令

EdgeX Kubernetes 部署实践（二）

Helm Chart 部署

Helm是Kubernetes的一个包管理工具，简化Kubernetes应用的部署和管理

- edgex-core-command-deployment.yaml
- edgex-core-command-service.yaml
- edgex-core-config-seed-pod.yaml
- edgex-core-consul-deployment.yaml
- edgex-core-consul-service.yaml
- edgex-core-data-deployment.yaml
- edgex-core-data-service.yaml
- edgex-core-metadata-deployment.yaml
- edgex-core-metadata-service.yaml
- edgex-device-virtual-deployment.yaml
- edgex-device-virtual-service.yaml

```
edgex-helm/
  Chart.yaml        # Chart描述信息
  README.md         # README
  values.yaml       # 默认配置值
  charts/           # 依赖包
  templates/        # Chart模版
```

```
$ helm install edgex-geneva edgex-helm
```

① 创建EdgeX Helm，执行 helm create edgex-helm

② 将Edgex Kubernetes清单文件拷贝到templates目录

③ 部署EdgeX Helm，执行helm install edgex-geneva edgex-helm

EdgeX Kubernetes 部署实践（二）

Helm 创建Kubernetes 对象的顺序

Helm

```
$ helm install edgex-geneva edgex-helm
```

执行创建kubernetes的顺序

① ConfigMap

② PersistentVolume

③ PersistentVolumeClaim

④ Service

⑤ Deployment

详细见helm kind_sorter.go

EdgeX Kubernetes 部署实践（二）

edgex-helm 项目

① 基于Kubernetes 包管理工具Helm Chart

② 基于 docker-compose-geneva-redis-no-secty.yml

③ Github: https://github.com/DaveZLB/edgex-helm

目前状态：可以在Kubernetes 部署测试

下一步计划：高可用方案（redis集群化，volume高可用）

Thank You

vmware