Flink on Yarn/Kubernetes原理剖析及实践

周凯波 · 阿里巴巴 / 技术专家

Apache Flink Community China

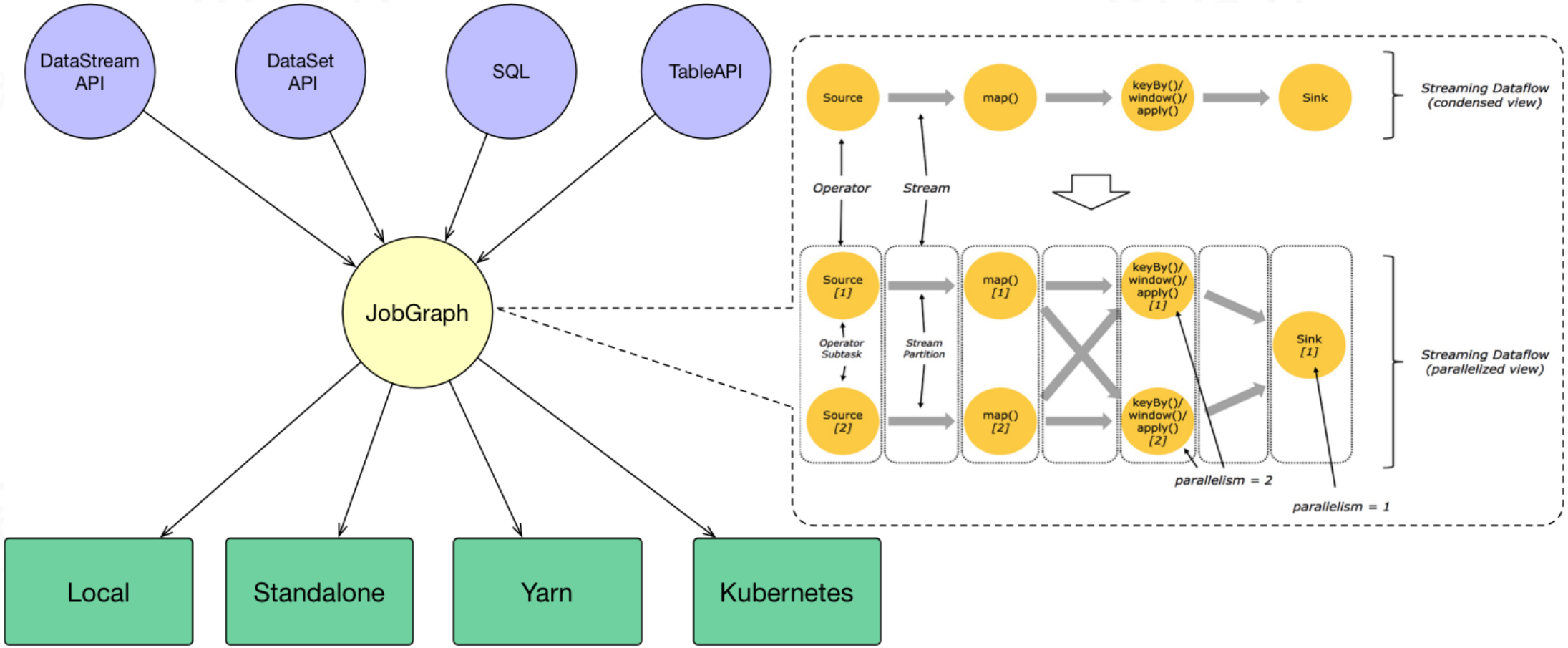
CONTENT

目录 >>

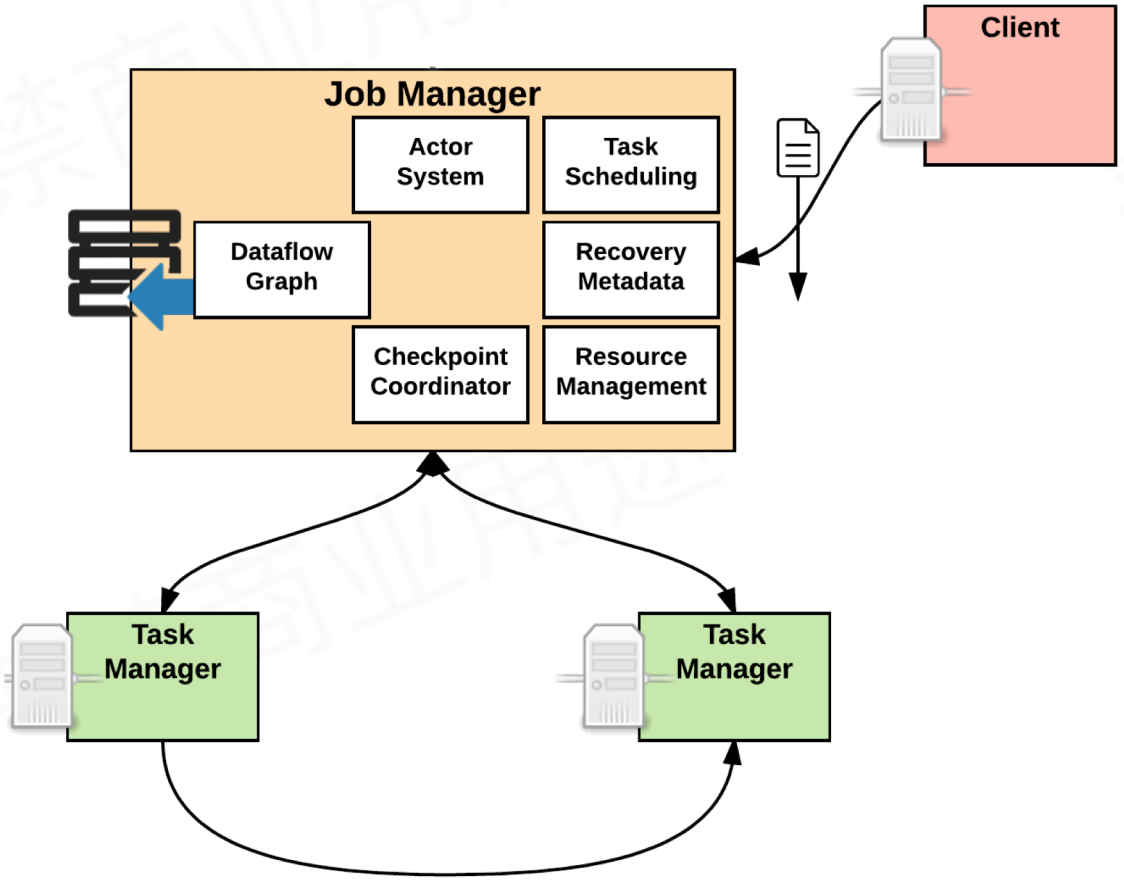
01

Flink 架构概览

Flink 架构概览 — Job

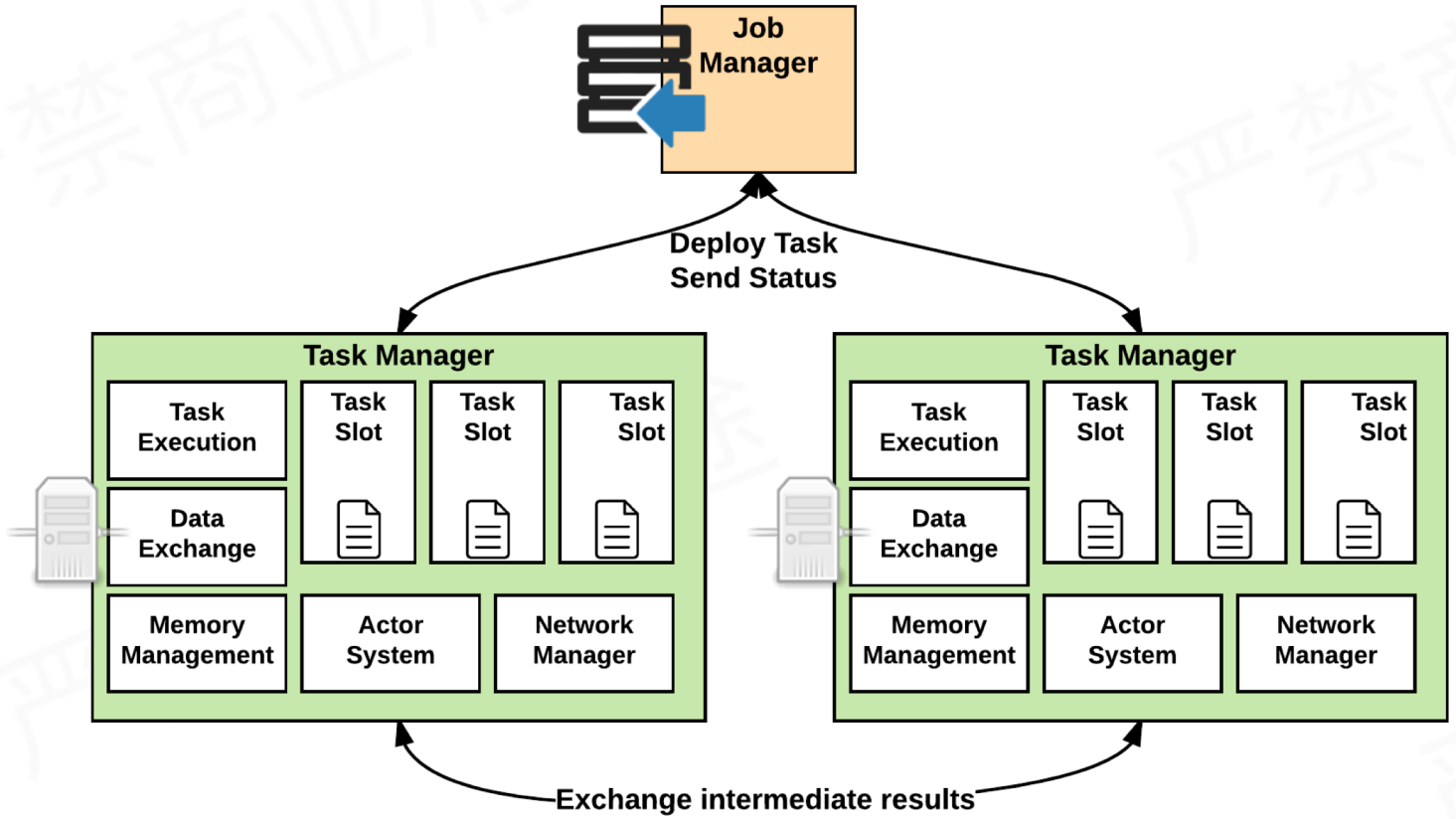


Flink 架构概览 — JobManager



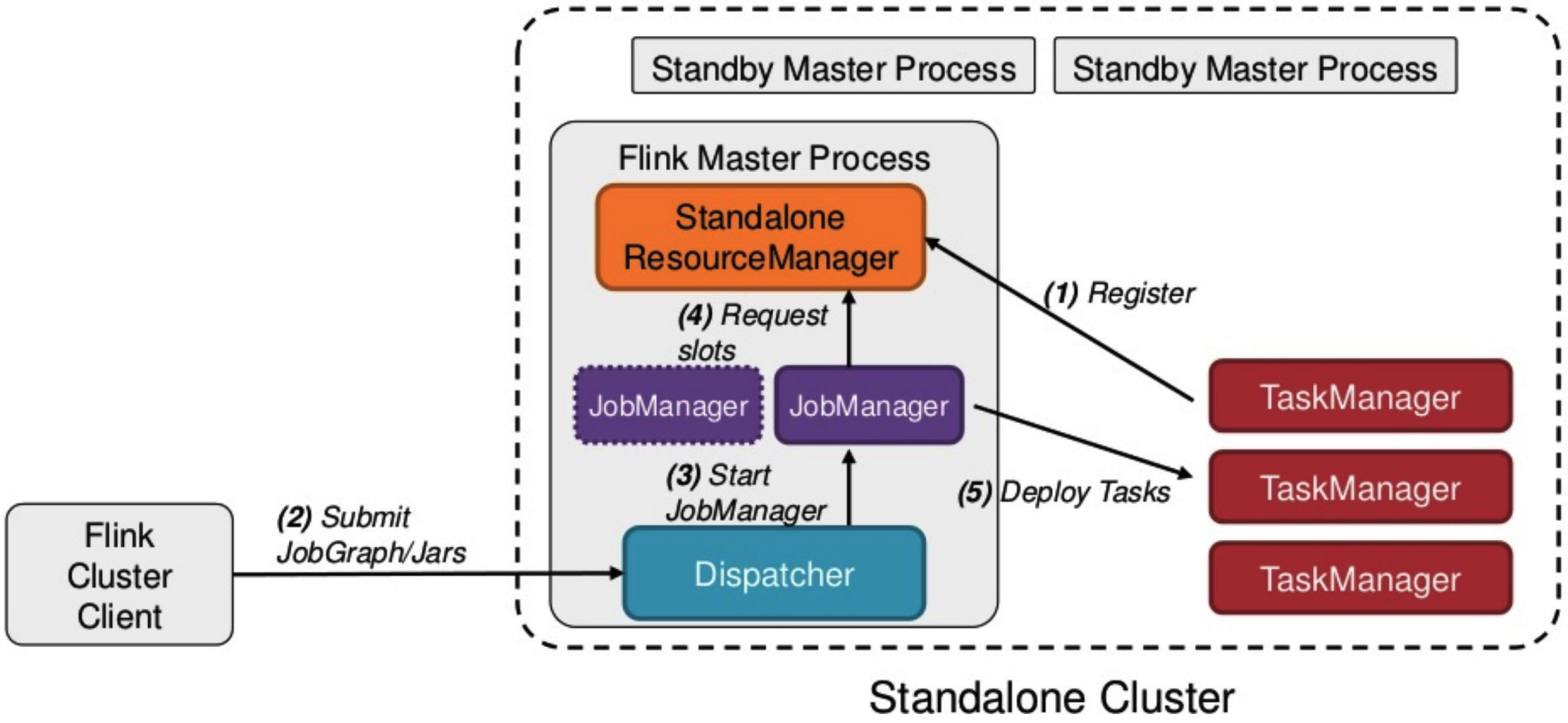
- Execution Graph
- Scheduler
- Checkpoint Coordinator
- Actor System

Flink 架构概览 — TaskManager



- Memory & I/O Manager
- Network Manager
- Actor system

Flink Standalone



Flink 运行时相关组件

Client

- 任务提交，生成 JobGraph

JobManager

- 调度Job，协调Task，通信，资源申请等

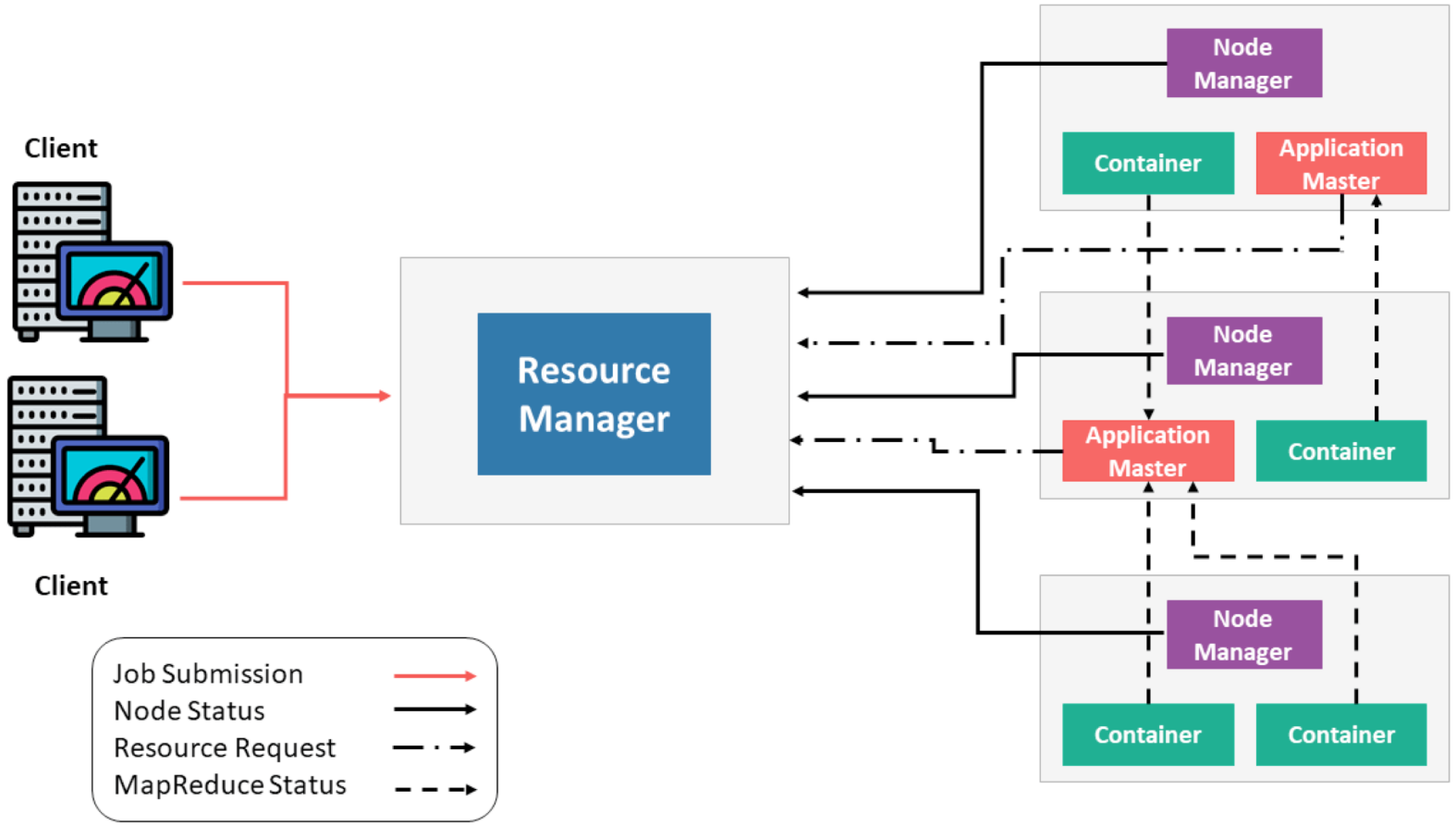
TaskManager

- 具体任务的执行，请求资源

02

Flink on Yarn 原理剖析

Yarn 架构原理 – 总览



图片来源：https://www.edureka.co/blog/hadoop-yarn-tutorial/

Yarn 架构原理 – 组件

ResourceManager (RM)

- 处理客户端请求、启动/监控App Master、监控NodeManager、资源的分配与调度
- 包含 Scheduler 和 Applications Manager

ApplicationMaster (AM)

- 运行在Slave上，负责数据切分，申请资源和分配，任务监控和容错

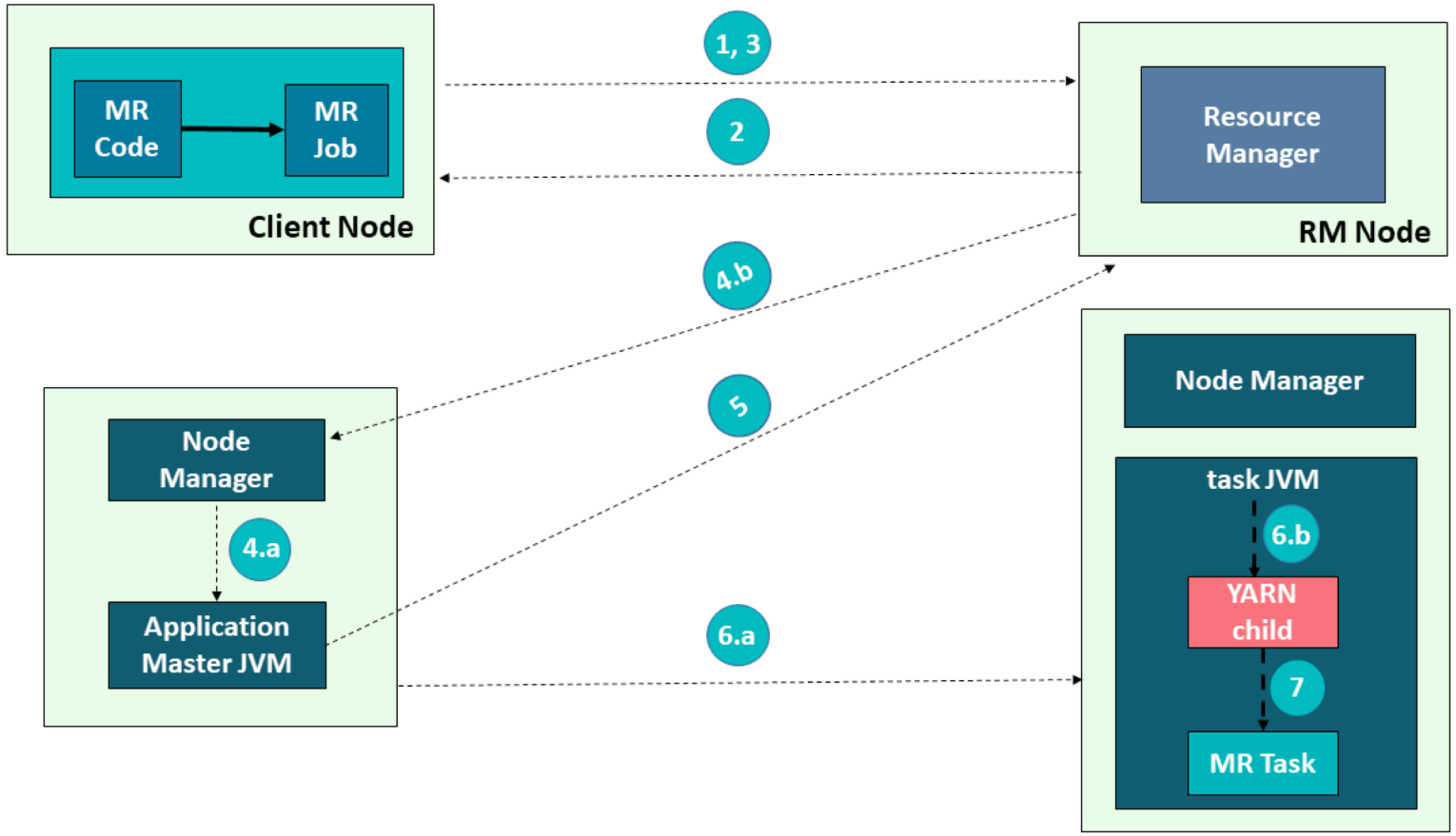
NodeManager (NM)

- 运行在Slave上，单节点资源管理，与AM/RM通信，汇报状态

Container

- 资源抽象，包括内存、CPU、磁盘，网络等资源

Yarn 架构原理 – 交互

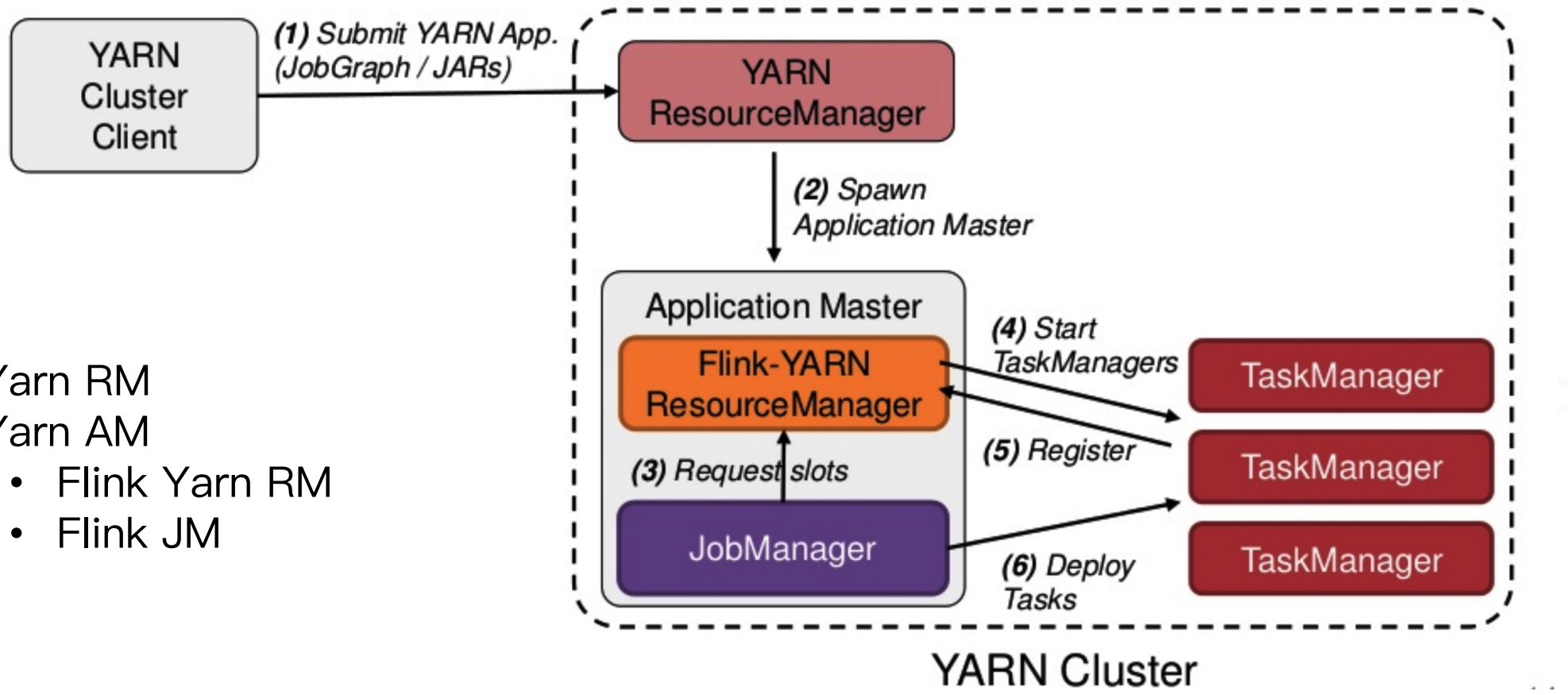


1. 提交任务
2. 获取 Application ID
3. 提交任务上下文
4.a. 启动 Container Launch
4.b. 启动 Application Master
5. 分配资源
6.a. 启动 Container
6.b. 启动子进程
7. 运行Task

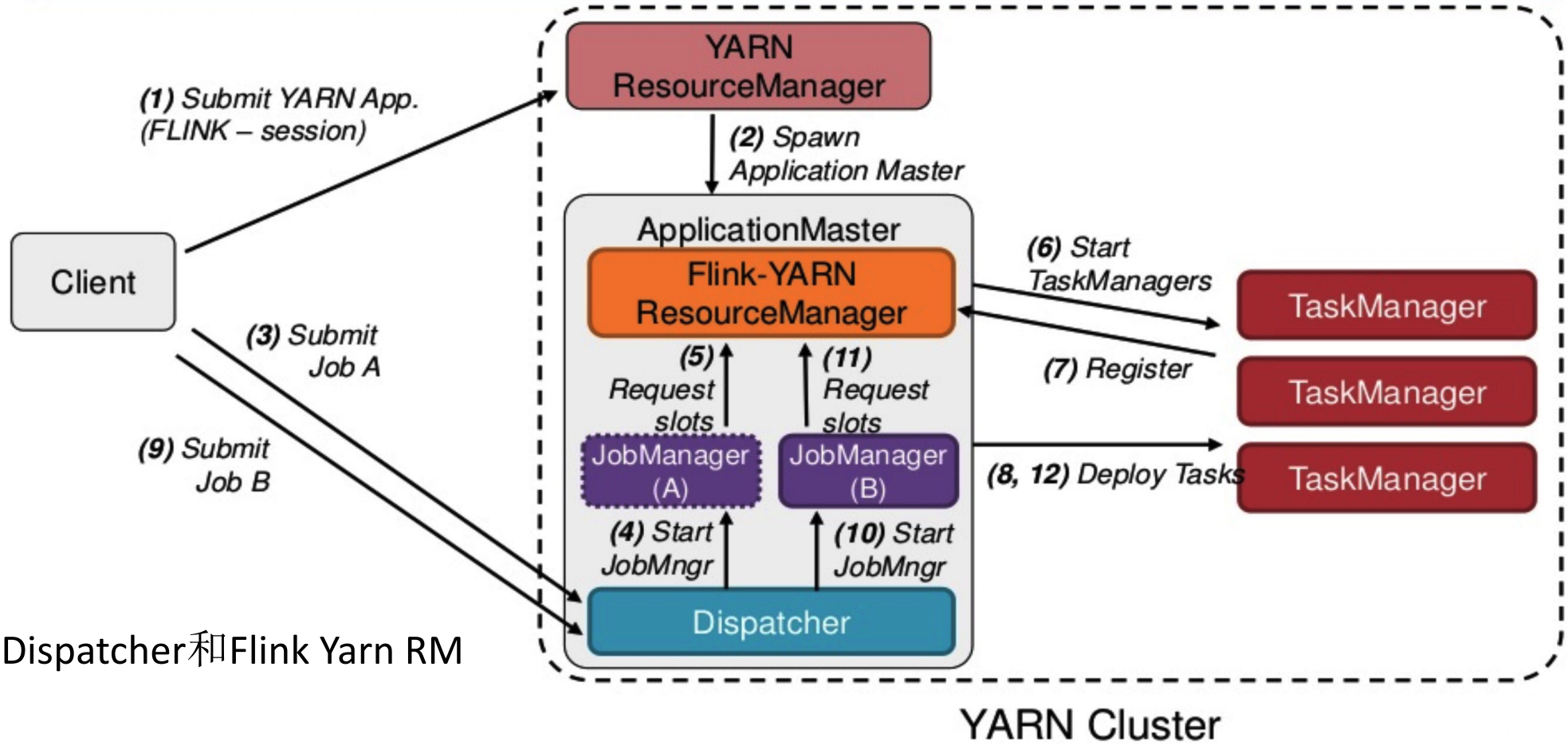
图片来源：https://www.edureka.co/blog/hadoop-yarn-tutorial/

Flink on Yarn — Per Job

- Yarn RM
- Yarn AM
  - Flink Yarn RM
  - Flink JM



Flink on Yarn — Session



多JM共享Dispatcher和Flink Yarn RM

Yarn模式优点

- 资源按需使用，提高集群的资源利用率
  - Yarn ResourceManager 资源管理
- 多种任务调度策略
  - FIFO Scheduler
  - Capacity Scheduler
  - Fair Scheduler
  - 任务优先级
- 通过 Yarn 进行自动 failover 处理
  - Yarn NodeManager 监控
  - Yarn ApplicationManager 异常恢复

Flink on Yarn 实践

- 参考
  - 《Flink 安装部署、环境配置及运行应用程序》
  - 《Flink 客户端操作》

- 链接：https://zh.ververica.com/developers/flink-training-course1/

03

Flink on Kubernetes 原理剖析

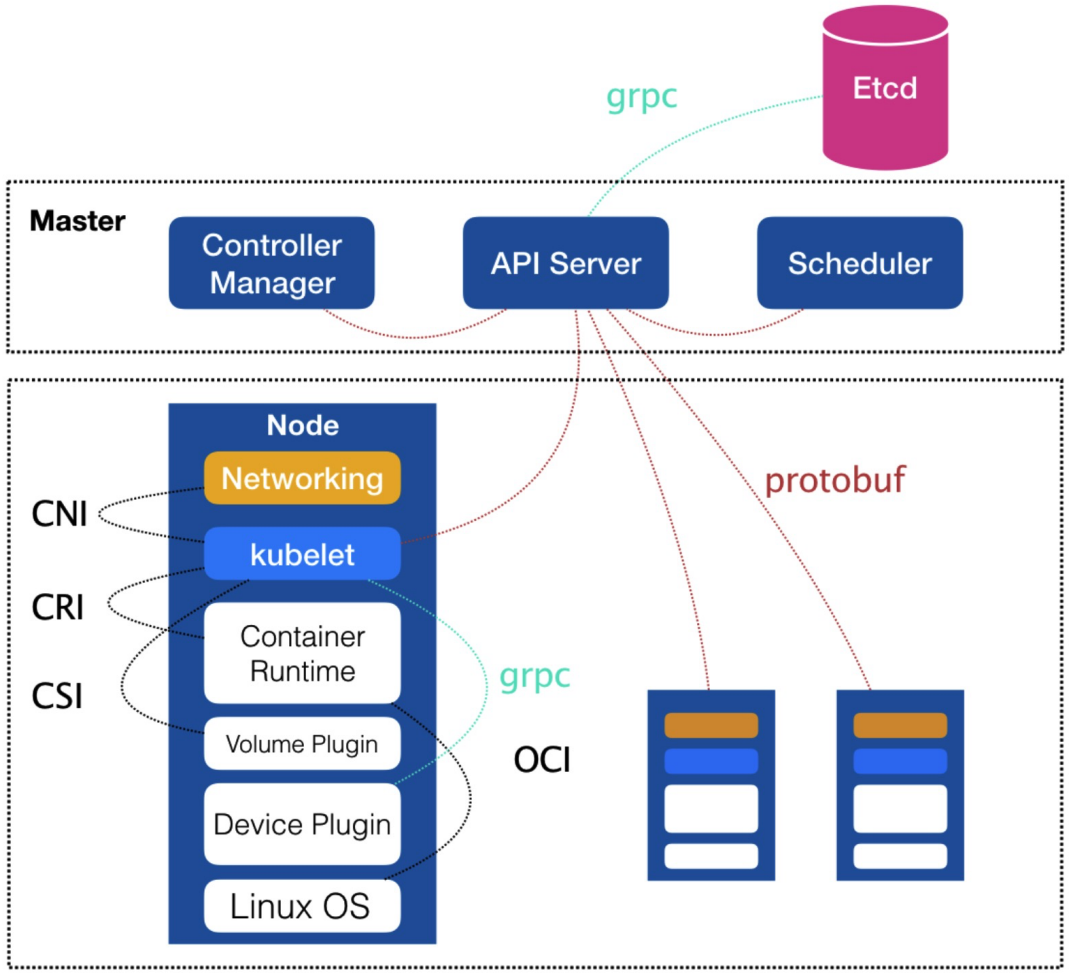
Kubernetes — 基本概念

- Master
  - 负责管理集群，集群的资源数据访问入口
  - 运行 API Server，Controller Manager 及 Scheduler 服务
  - Etcd 高可用键值存储服务

- Node
  - 集群操作的单元，是 Pod 运行的宿主机
  - kubelet：agent进程，维护和管理该Node上的所有容器的创建、启停等
  - kube-proxy：服务发现，反向代理和负载均衡
  - docker engine：docker引擎，负责本机容器的创建和管理工作

- Pod
  - 运行于Node节点上，若干相关容器的组合
  - 创建、调度和管理的最小单位

Kubernetes – 架构图



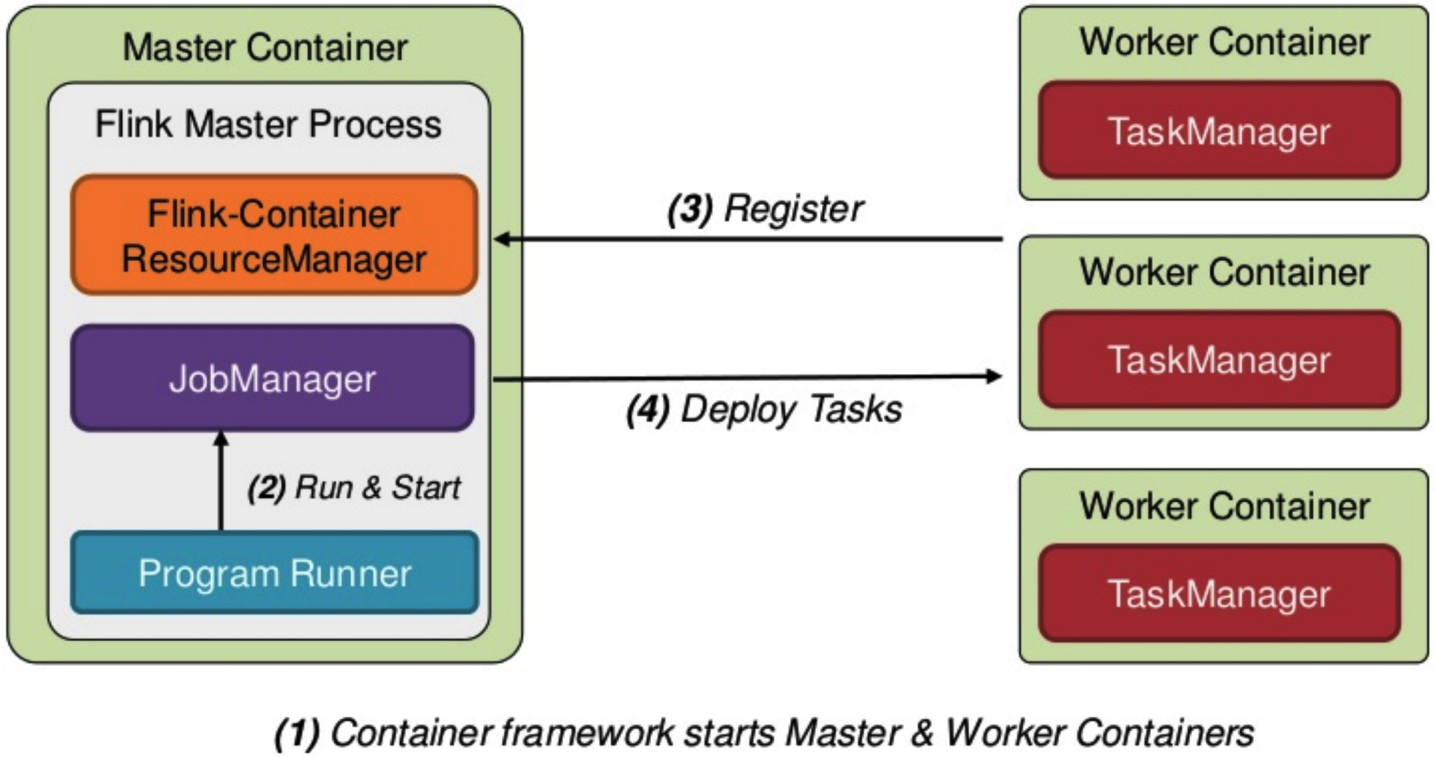
- 用户通过kubectl提交需要运行的docker container
- API Server 把请求存储在etcd
- Scheduler 扫描，分配机器
- Kubelet找到自己需要跑的container，在本机上运行

- 用户提交RC描述，replication controller监视集群中的容器并保持数量
- 用户提交service描述文件，由kube proxy负责具体的工作流量转发

Kubernetes – 核心概念

- Replication Controller (RC)
  - 管理Pod的副本，保证集群中存在指定数量的Pod副本
  - Deployment & Job
- Service
  - 提供了一个统一的服务访问入口以及服务代理和发现机制
- Persistent Volume(PV) 和 Persistent Volume Claim(PVC)
  - 数据卷
- ConfigMap
  - 键值对

Flink on Kubernetes — 架构



同一个镜像根据不同的启动命令运行 Master 或 Worker

Flink on Kubernetes — JobManager

- JobManager **Deployment**
  - 保证 1 个副本的container运行JobManager
  - 应用标签，例如 flink-jobmanager

- JobManager **Service**
  - 通过 service name 和 port 暴露 JobManager 服务
  - 通过标签选择对应的pods，例如 flink-jobmanager

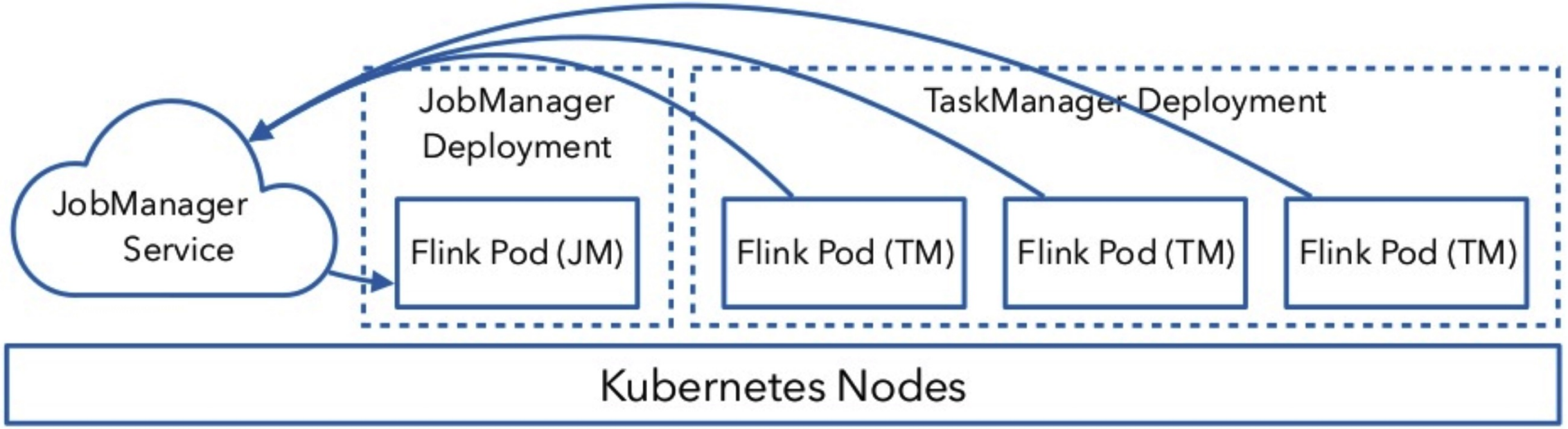
Flink on Kubernetes — TaskManager

- TaskManager **Deployment**
  - 保证 n 个副本的 container 运行 TaskManager
  - 应用标签，例如 flink-taskmanager

- Flink conf **ConfigMap**
  - flink-conf.yaml, core-site.xml …

Flink on Kubernetes — 交互

- Service: 通过标签(label selector)找到job manager 的 pod 暴露服务
- Deployment：保证 n 个副本的container运行JM/TM，应用升级策略
- ConfigMap：在每个pod上通过挂载 /etc/flink 目录包含 flink-conf.yaml 内容



Flink on Kubernetes — 实践

- Session Cluster
  - 启动
    - kubectl create -f jobmanager-service.yaml
    - kubectl create -f jobmanager-deployment.yaml
    - kubectl create -f taskmanager-deployment.yaml
  - Submit job
    - kubectl port–forward service/flink–jobmanager 8081:8081
    - bin/flink run –d –m localhost:8081 ./examples/streaming/TopSpeedWindowing.jar
  - 停止
    - kubectl delete -f jobmanager-deployment.yaml
    - kubectl delete -f taskmanager-deployment.yaml
    - kubectl delete -f jobmanager-service.yaml

jobmanager-deployment.yaml

```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: flink-jobmanager
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: flink
        component: jobmanager
    spec:
      containers:
      - name: jobmanager
        image: flink:latest
        args:
        - jobmanager
        ports:
        - containerPort: 6123
          name: rpc
        - containerPort: 6124
          name: blob
        - containerPort: 6125
          name: query
        - containerPort: 8081
          name: ui
        env:
        - name: JOB_MANAGER_RPC_ADDRESS
          value: flink-jobmanager
```

API版本
资源类型为Deployment

副本数为1

标签，用于pod的选取

镜像名和版本
默认从dockerhub仓库下载

启动参数，这里决定了启动的是jobmanager

服务端口

环境变量，会传给启动脚本

taskmanager-deployment.yaml

```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: flink-taskmanager
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: flink
        component: taskmanager
    spec:
      containers:
      - name: taskmanager
        image: flink:latest
        args:
        - taskmanager
        ports:
        - containerPort: 6121
          name: data
        - containerPort: 6122
          name: rpc
        - containerPort: 6125
          name: query
        env:
        - name: JOB_MANAGER_RPC_ADDRESS
          value: flink-jobmanager
```

标签是 taskmanager

启动参数，这里决定了启动的是taskmanager

jobmanager-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: flink-jobmanager
spec:
  ports:
  - name: rpc
    port: 6123
  - name: blob
    port: 6124
  - name: query
    port: 6125
  - name: ui
    port: 8081
  selector:
    app: flink
    component: jobmanager
```

资源类型为 Service

要暴露的服务端口

通过标签选择 jobmanager 的pod

Flink on Kubernetes — 实践

- Job Cluster
  - build 镜像
    - 在 flink/flink-container/docker目录下执行

      sh build.sh --from-release --flink-version 1.7.0 --hadoop-version 2.8 --scala-version 2.11 --job-jar ~/flink/flink-1.7.1/examples/streaming/TopSpeedWindowing.jar --image-name topspeed

      成功后会输出一行：Successfully tagged topspeed:latest

  - 上传镜像
    - 在 https://hub.docker.com/ 注册帐号和创建仓库，然后上传镜像

      (以我的仓库 zkb555 为例)
      - docker tag topspeed zkb555/topspeedwindowing
      - docker push zkb555/topspeedwindowing

Flink on Kubernetes — 实践

- Job Cluster
  - 启动任务
    - kubectl create -f job-cluster-service.yaml

    - FLINK_IMAGE_NAME=zkb555/topspeedwindowing:latest FLINK_JOB=org.apache.flink.streaming.examples.windowing.TopSpeedWindowing FLINK_JOB_PARALLELISM=3 envsubst < job-cluster-job.yaml.template | kubectl create -f –

    - FLINK_IMAGE_NAME=zkb555/topspeedwindowing:latest FLINK_JOB_PARALLELISM=4 envsubst < task-manager-deployment.yaml.template | kubectl create -f -

04

Q/A

THANKS

Apache Flink 中文学习网站：zh.ververica.com