阿里巴巴云原生专场

Serverless 场景下 Pod 创建效率优化

张翼飞 阿里云容器服务技术专家

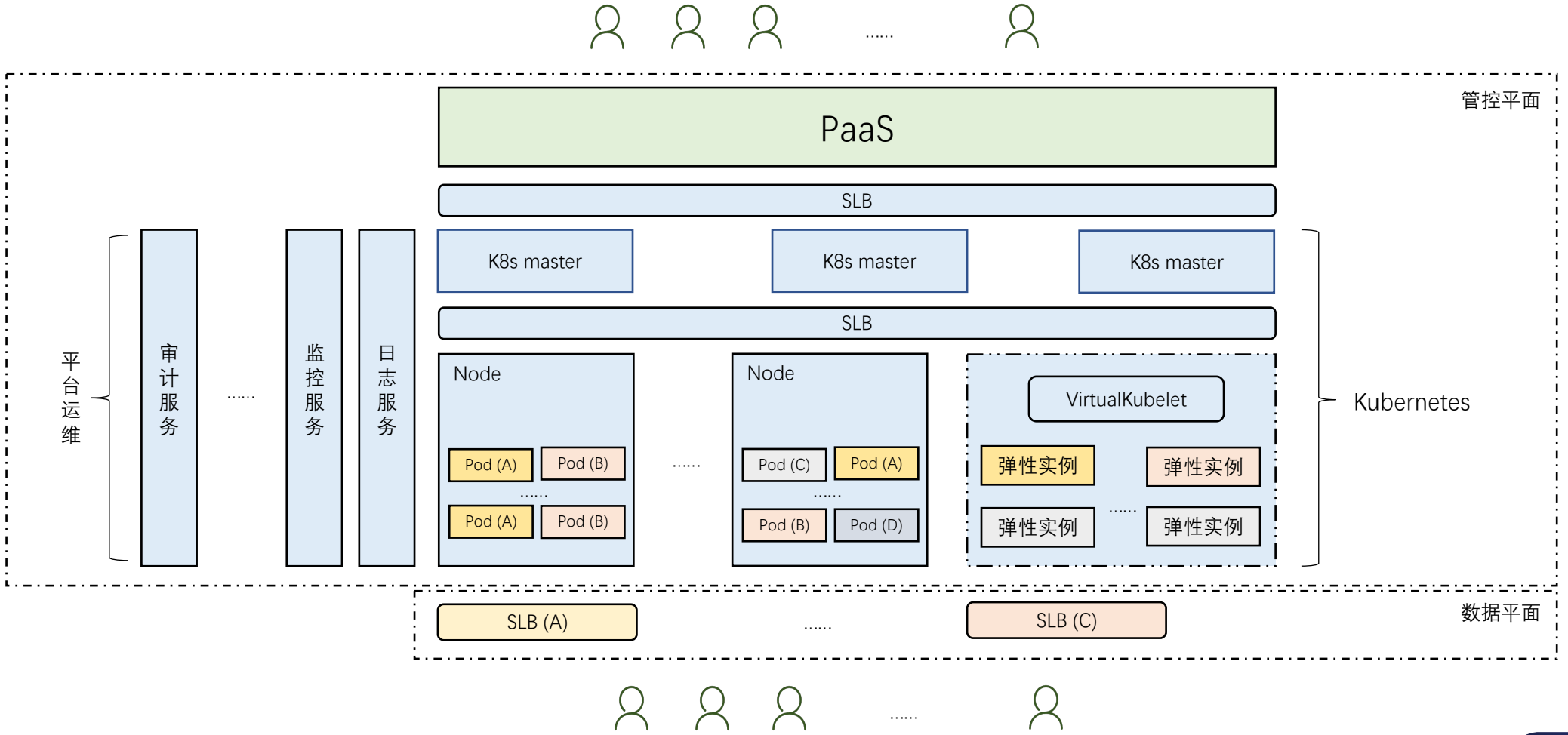
观看视频回放

Serverless 计算简介

Serverless computing is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources. Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity.

From: https://en.wikipedia.org/wiki/Serverless_computing

Serverless 计算平台架构



弹性能力

Pod 规模

创建 Pod 效率

Pod 创建相关场景

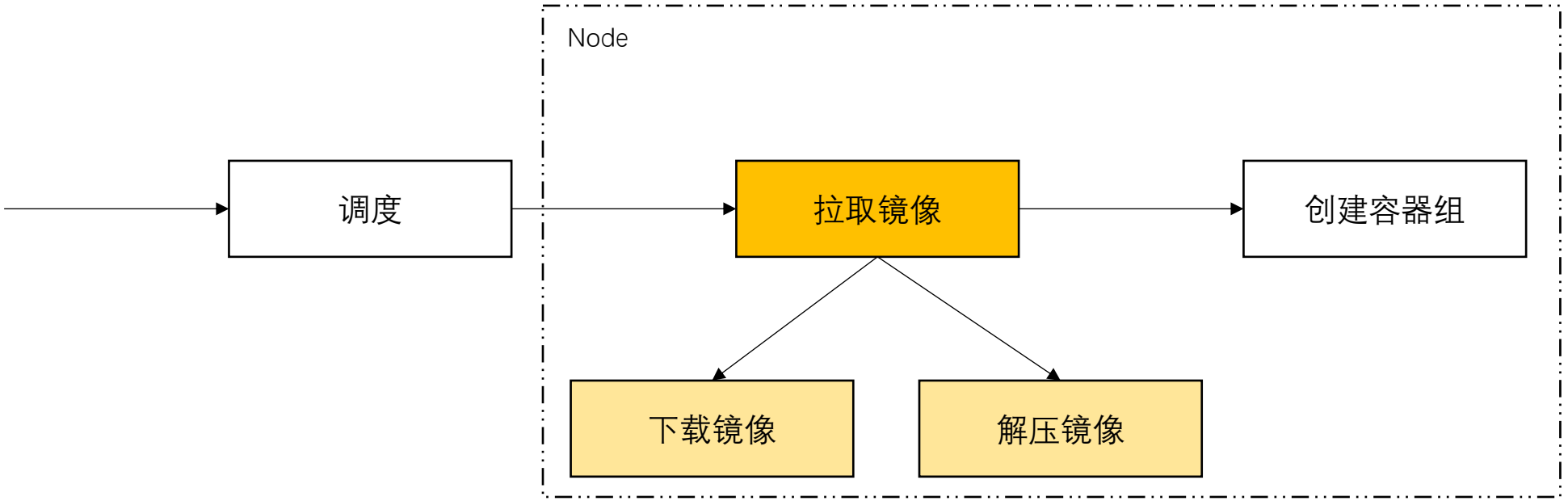
创建应用

创建应用 -> 调度 **-> 创建 Pod**

升级应用

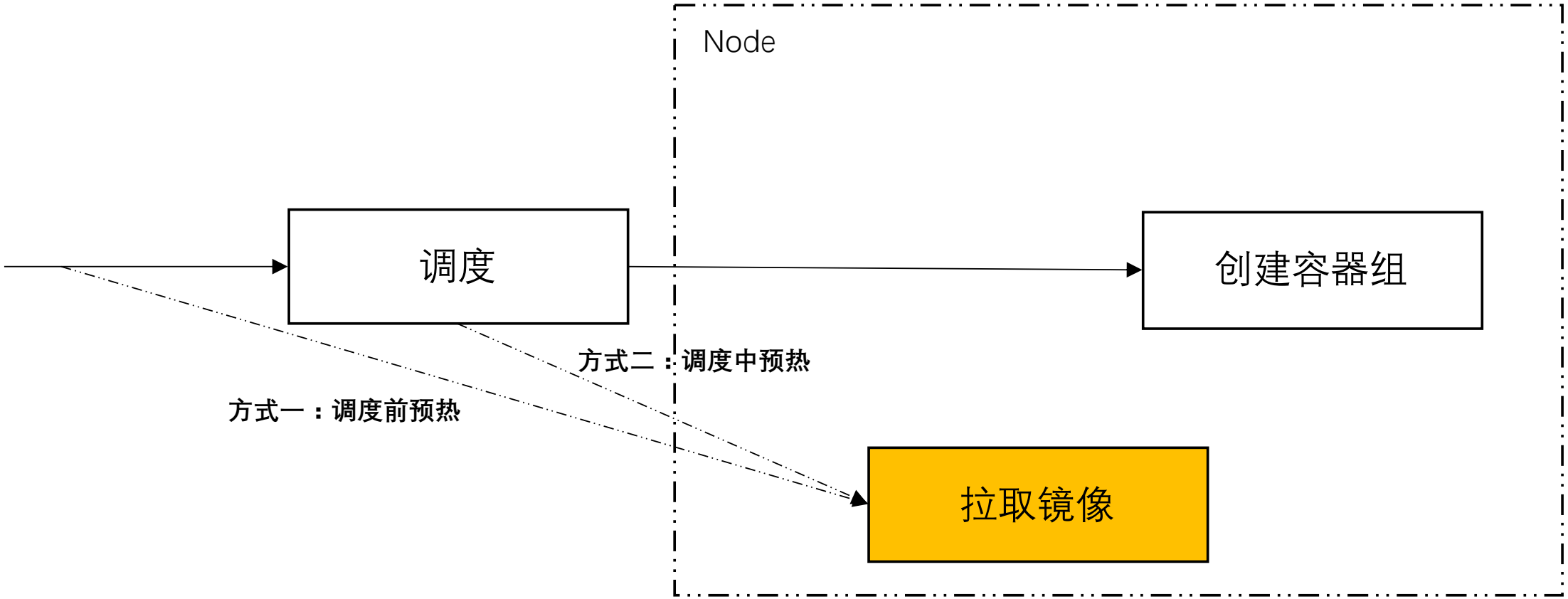
升级应用 -> 调度 **-> 创建新 Pod，销毁旧 Pod**

创建 Pod 流程



镜像	解压前大小	解压后大小	(内网) 拉取镜像总耗时	下载镜像耗时	解压镜像耗时
golang:1.10	248.76 MB	729.90 MB	16.97s	3.90s (占比 22.98%)	13.07s (占比 77.02%)
bde2020/hadoop-namenode:2.0.0-hadoop3.1.1-java8	506.87 MB	1.29 GB	38.87s	23.26s (占比 59.84%)	15.61s (占比 40.16%)

拉取镜像效率提升 - 镜像预热

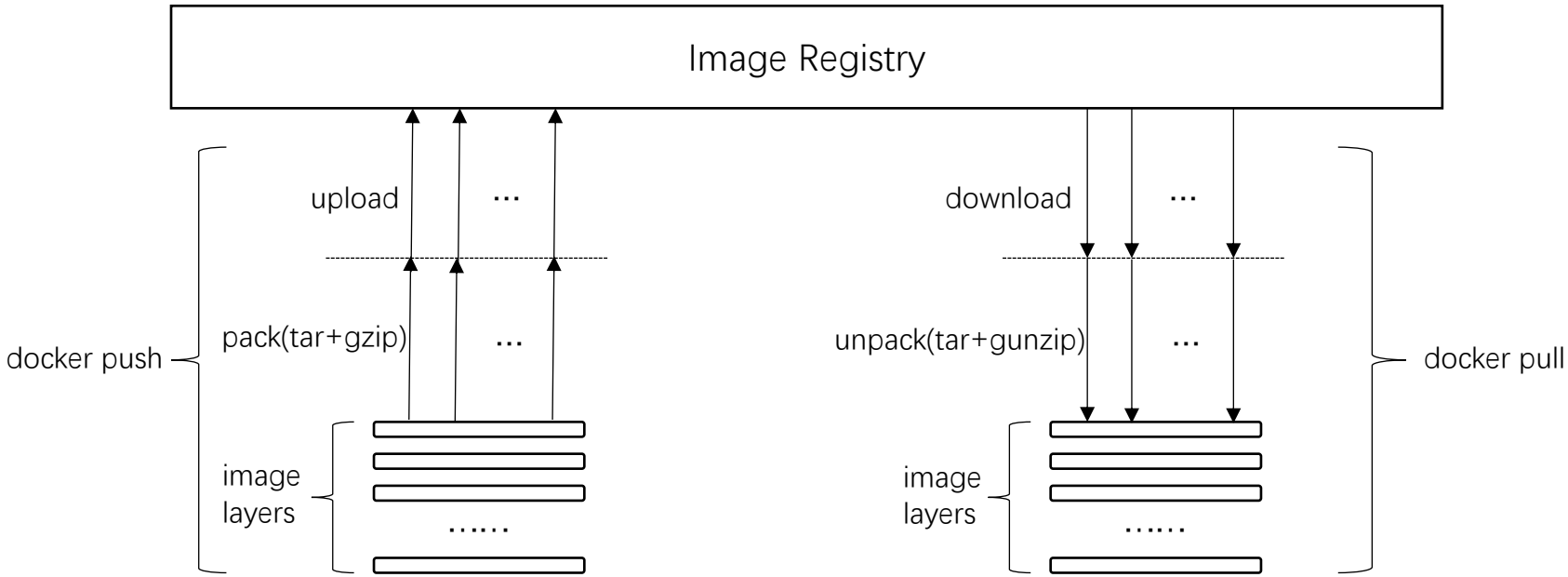


拉取镜像效率提升 - 镜像预热

```yaml
apiVersion: apps.kruise.io/v1alpha1
kind: ImagePullJob
metadata:
  name: golang
spec:
  completionPolicy:
    activeDeadlineSeconds: 300
    ttlSecondsAfterFinished: 600
  image: registry-vpc.cn-beijing.aliyuncs.com/test/golang:1.10
  parallelism: 10
  pullSecrets:
  - test
  selector:
    names:
    - i-2zed6ptcmrk9que37wq0
    - i-2zed6ptcmrk9que37wq1
    - i-2zed6ptcmrk9que37wq2
```

OpenKruise 项目下半年将会推出该服务

拉取镜像效率提升 - 解压效率



拉取镜像效率提升 - 解压效率

压缩/解压工具: gzip (单线程) -> pigz (多线程)

containerd 从 1.2 版本开始支持，节点上需要安装 unpigz 工具。

Upload / Download 并发度

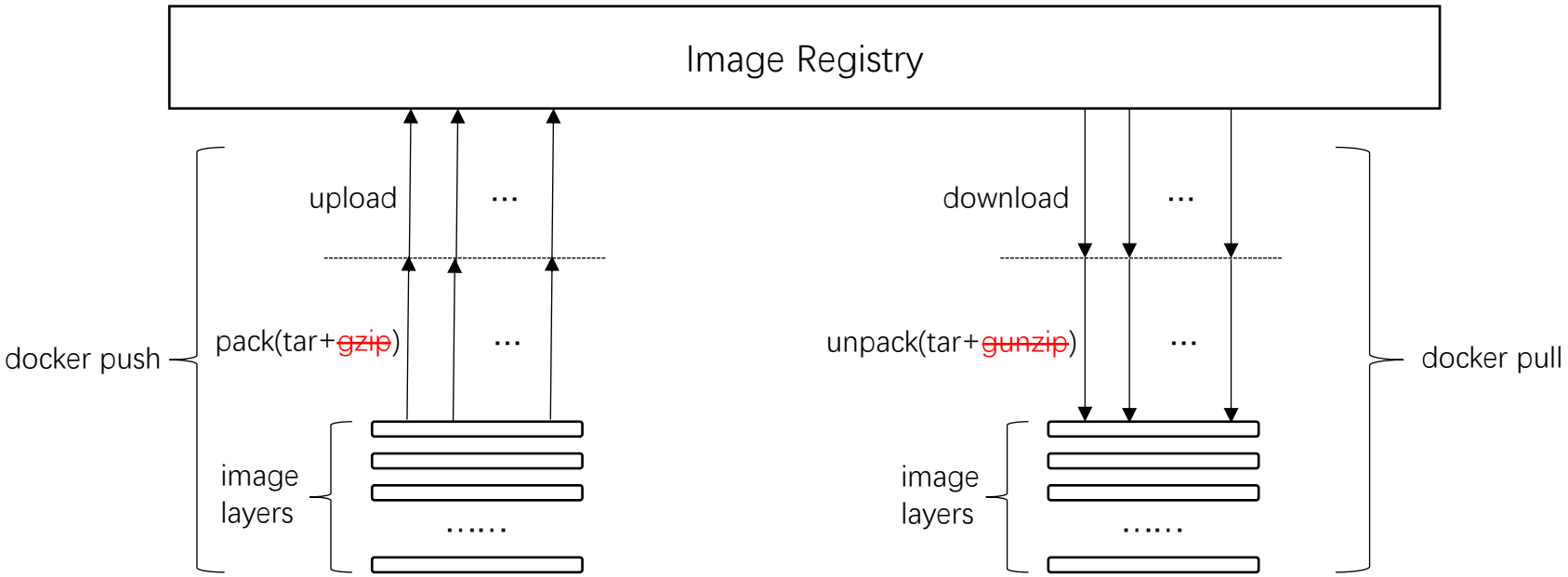
docker daemon

```
--max-concurrent-downloads int          Set the max concurrent downloads for each pull (default 3)
--max-concurrent-uploads int            Set the max concurrent uploads for each push (default 5)
```

拉取镜像效率提升 - 解压效率

镜像	解压前大小	解压后大小	(内网) 拉取镜像总耗时	下载镜像耗时	解压镜像耗时	解压效率提升
golang:1.10	248.76 MB	729.90 MB	16.97s	3.90s	**Gunzip: 13.07s**	**35.88%**
			11.98s	3.60s	**Unpigz: 8.38s**	
bde2020/hadoop-namenode:2.0.0-hadoop3.1.1-java8	506.87 MB	1.29 GB	38.87s	15.60s	**Gunzip:23.27s**	**16.41%**
			29.35s	9.90s	**Unpigz 19.45s**	

拉取镜像效率提升 - 非压缩镜像



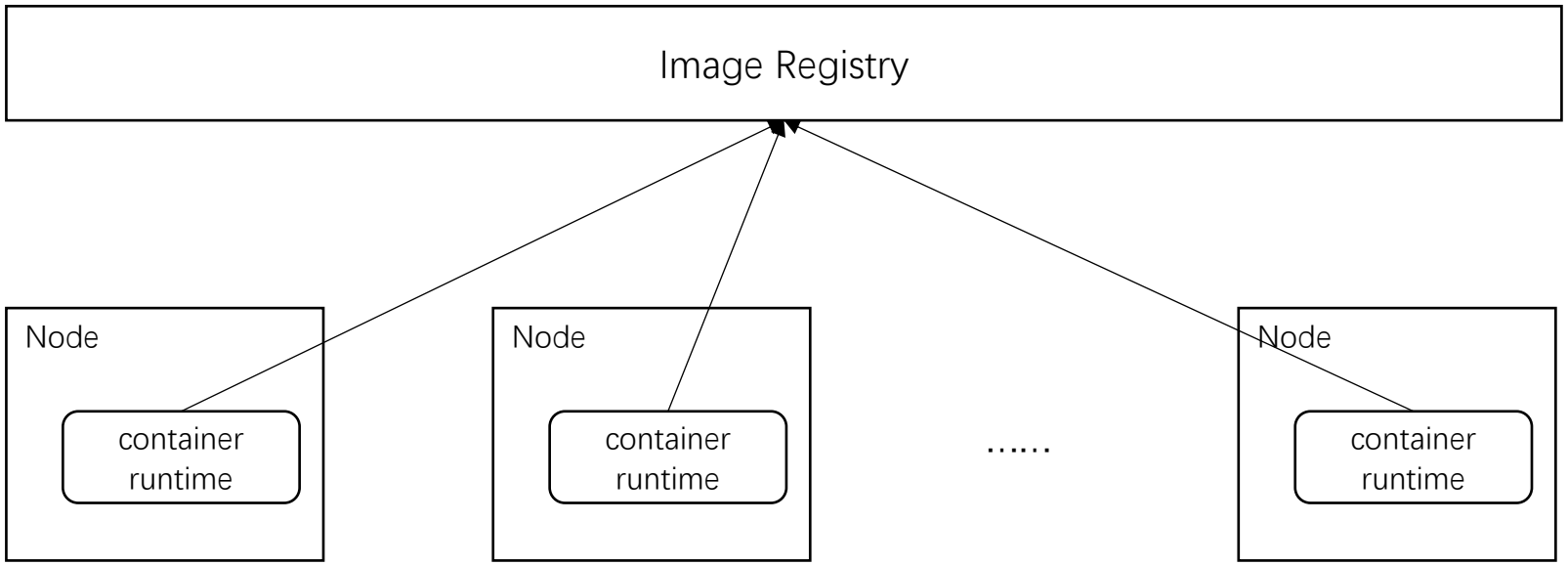
Push 时不压缩 tar 包，Pull 时会根据 tar 包是否压缩选择是否解压

拉取镜像效率提升 - 非压缩镜像

镜像	解压前大小	解压后大小	(内网) 拉取镜像总耗时	下载镜像耗时	解压镜像耗时	解压效率提升
golang:1.10	248.76 MB	729.90 MB	16.97s	3.90s	**13.07s**	**49.81%**
			13.16s	6.60s	**6.56s**	
bde2020/hadoop-namenode:2.0.0-hadoop3.1.1-java8	506.87 MB	1.29 GB	38.87s	15.60s	**23.27s**	**27.89%**
			33.28s	16.50s	**16.78s**	

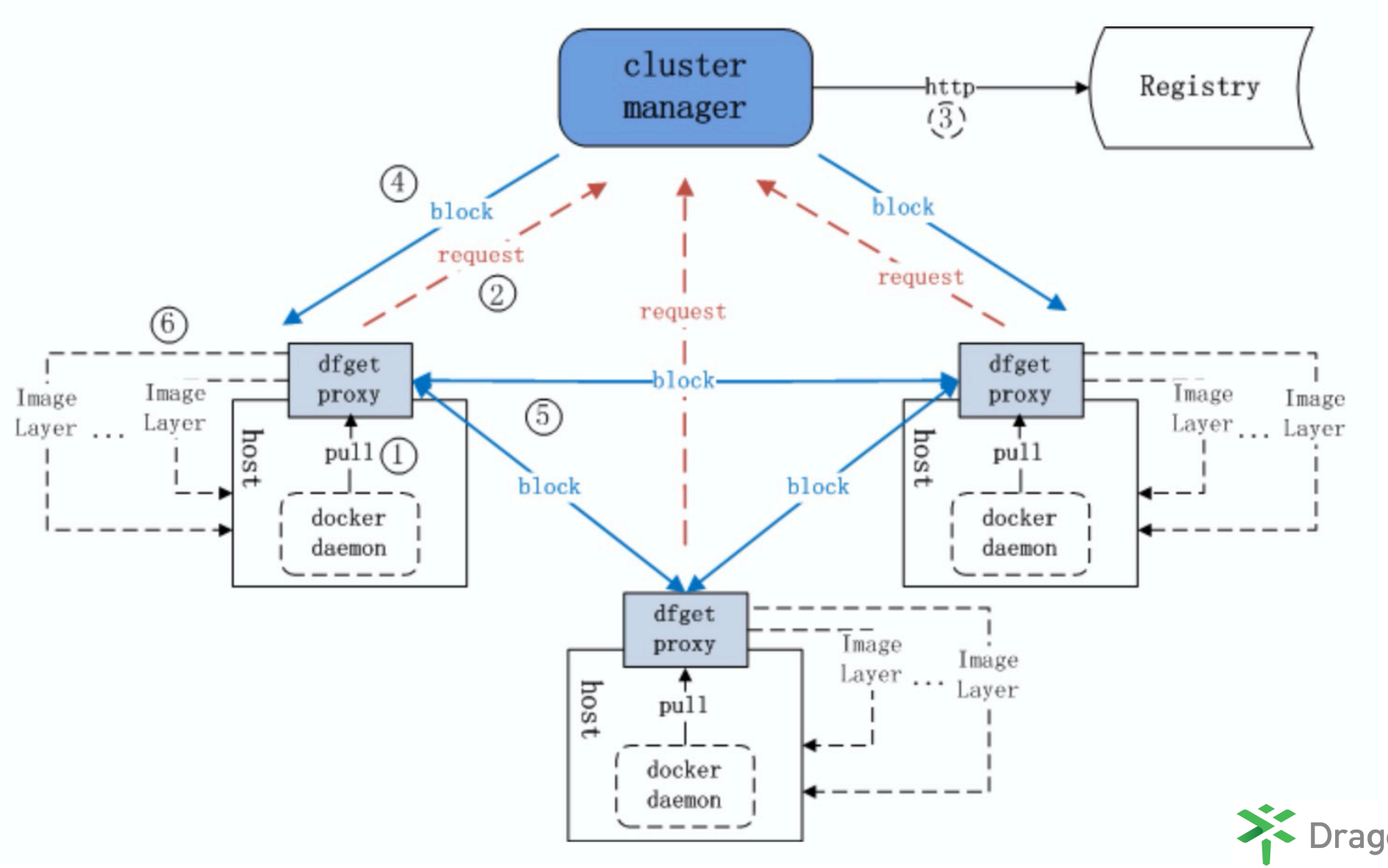
Note: docker daemon 暂不支持

拉取镜像效率提升 - 镜像分发现状



下载镜像的压力集中在中心式的 Image Registry

拉取镜像效率提升 - 基于 P2P 镜像分发



分散 Image Registry 压力
充分利用内网带宽

Dragonfly

From: https://d7y.io/en-us/docs/overview/what_is_dragonfly.html

拉取镜像效率提升 - 按需加载镜像

Our analysis shows that pulling packages accounts for 76% of container start time, but only 6.4% of that data is read.

From: Slacker: Fast Distribution with Lazy Docker Containers

拉取镜像效率提升 - 按需加载镜像

改变使用镜像的方式

Image 所有 layers 下载完后才能启动镜像
-> **启动容器时按需加载镜像**

tar 文件没有索引，gzip 文件不能从任意位置读取
-> **使用可索引的文件格式**

拉取镜像效率提升 - 按需加载镜像

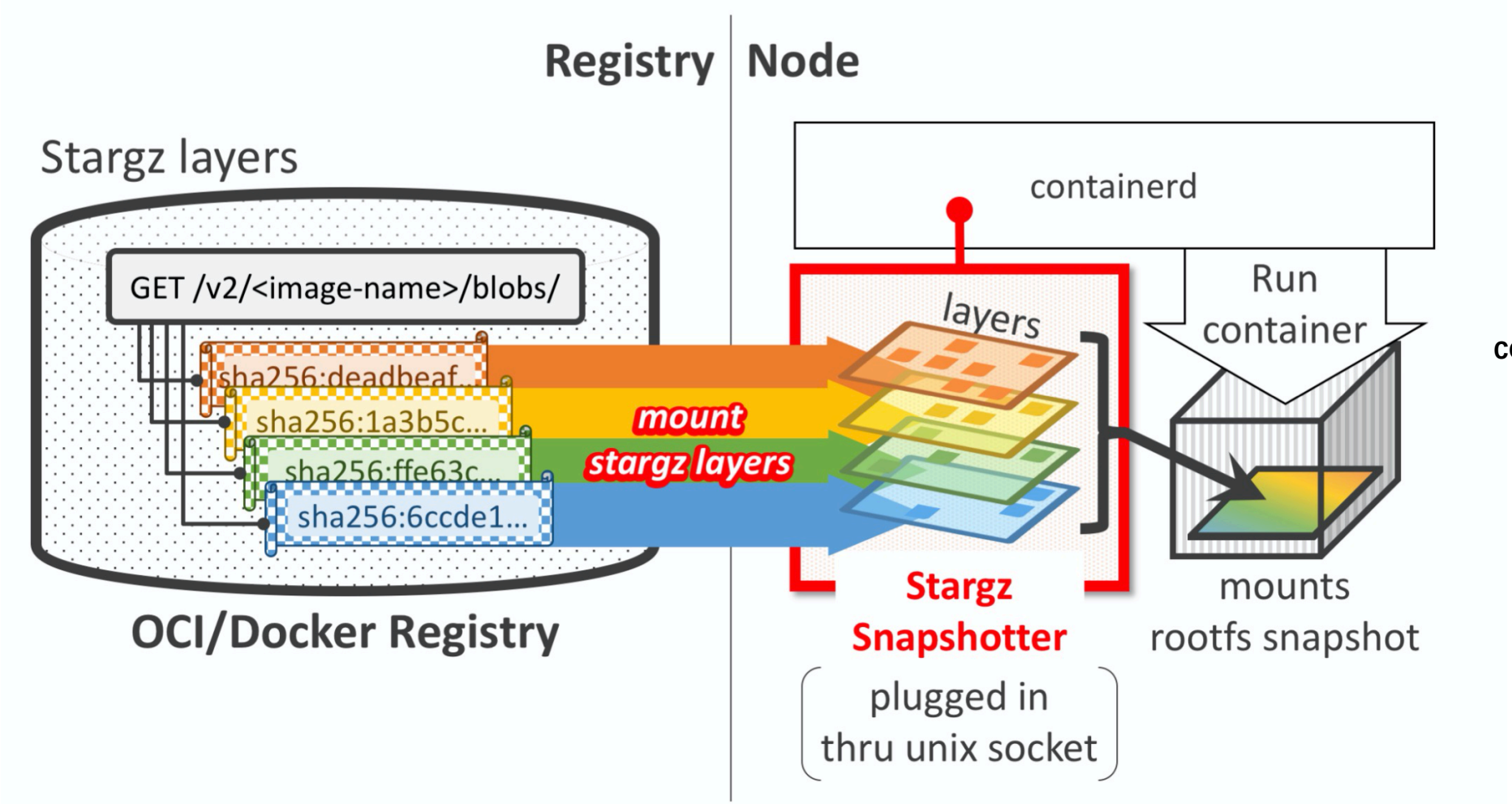
We introduce a format, **Stargz**, a **S**eekable **tar.gz** format that's still a valid tar.gz file for everything else that's unaware of these details.

In summary:

- That traditional `*.tar.gz` format is: `Gzip(TarF(file1) + TarF(file2) + TarF(file3) + TarFooter))`
- Stargz's format is: `Gzip(TarF(file1)) + Gzip(TarF(file2)) + Gzip(TarF(file3_chunk1)) + Gzip(F(file3_chunk2)) + Gzip(F(index of earlier files in magic file), TarFooter)`, where the trailing ZIP-like index contains offsets for each file/chunk's GZIP header in the overall **stargz** file.

From: https://github.com/google/crfs

拉取镜像效率提升 - 按需加载镜像



containerd/stargz-snapshotter

From: https://github.com/containerd/stargz-snapshotter/blob/master/docs/overview.md

拉取镜像效率提升 - 按需加载镜像

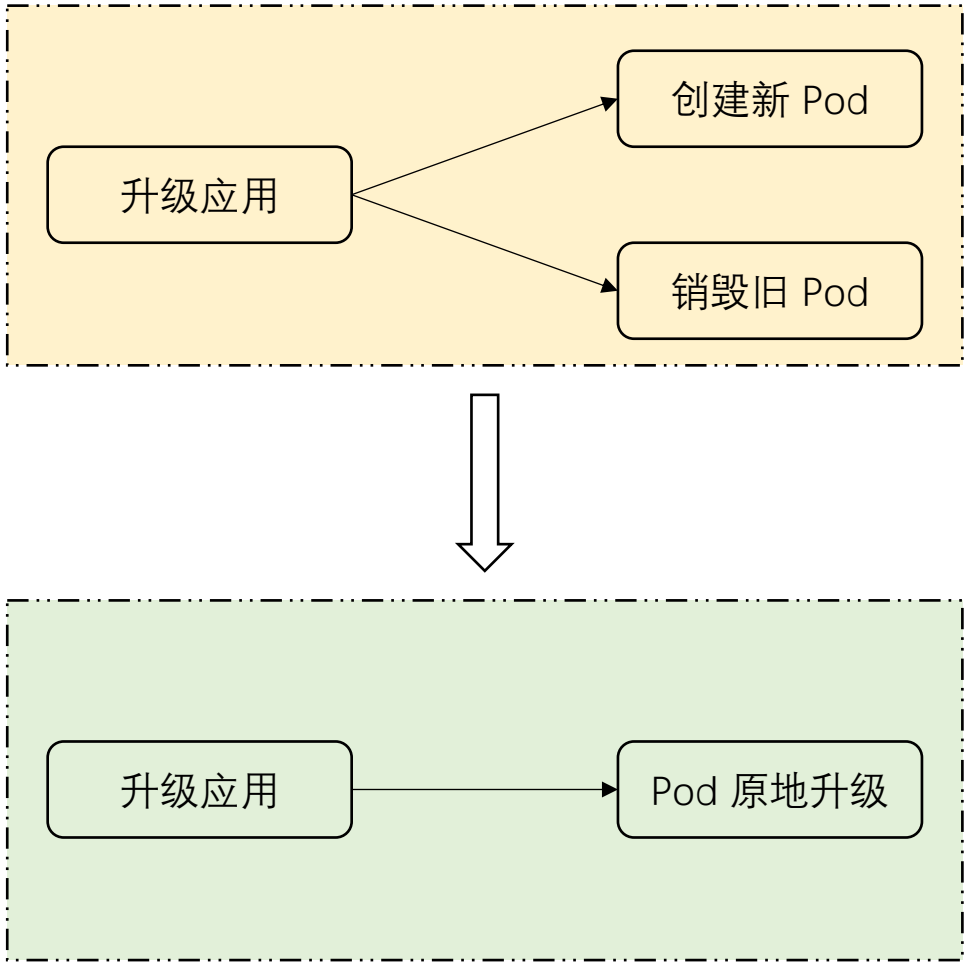
商业化使用：阿里云容器服务 + DADI 加速器

简介： 阿里云容器与存储团队展开合作，利用DADI加速器支持镜像按需读取和P2P分发，实现3.01秒启动10000个容器，完美杜绝容器冷启动的数分钟漫长等待，以及镜像仓库大规模并行分发场景下的网络拥堵。

阿里云容器与存储团队展开合作，利用DADI加速器支持镜像按需读取和P2P分发，实现3.01秒启动10000个容器，完美杜绝容器冷启动的数分钟漫长等待，以及镜像仓库大规模并行分发场景下的网络拥堵。

From: https://developer.aliyun.com/article/742103

升级应用 - 原地升级



升级应用 - 原地升级

```yaml
apiVersion: apps.kruise.io/v1alpha1
kind: CloneSet
metadata:
  labels:
    app: sample
  name: sample
spec:
  replicas: 5
  selector:
    matchLabels:
      app: sample
  template:
    metadata:
      labels:
        app: sample
    spec:
      containers:
      - name: nginx
        image: nginx:alpine
```

原理

通过 K8s patch 能力，实现原地升级 container
通过 K8s readinessGates 能力，实现升级过程中流量无损

THANKS