SACC

2020 中国系统架构师大会

SYSTEM ARCHITECT CONFERENCE CHINA 2020

架构融合 云化共建

live 2020年10月22日 - 24日网络直播

云上会议

架构融合
云化共建

基于Kubernetes的在/离线业务混部实践

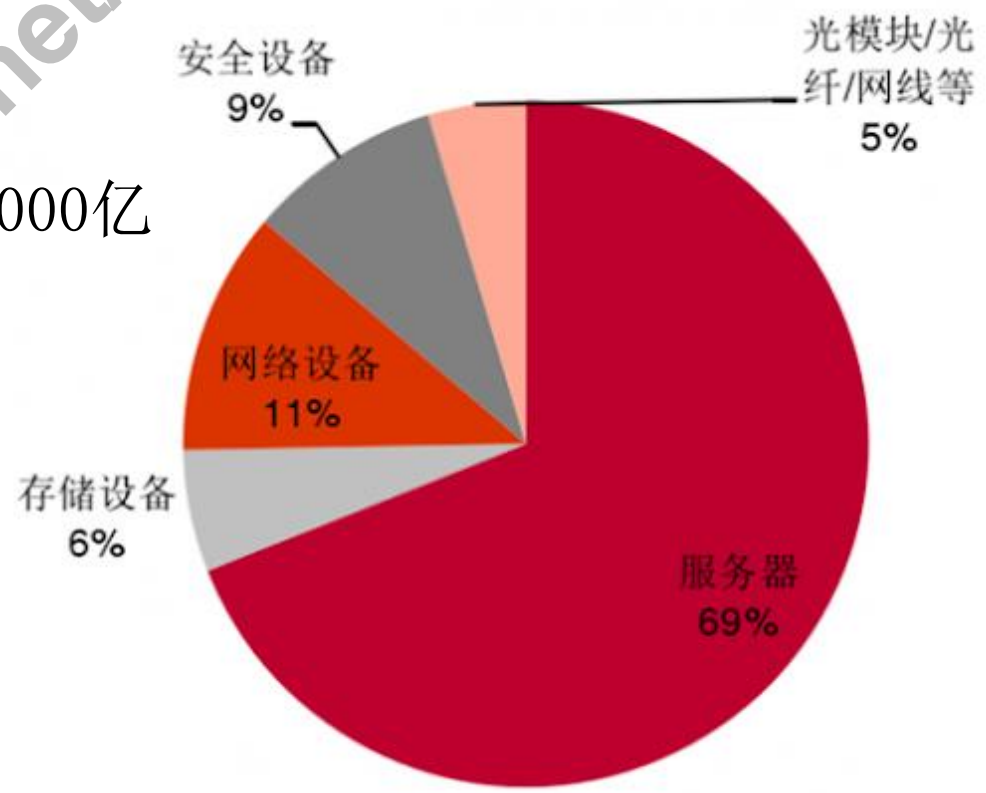
网易数帆轻舟 李岚清

目录

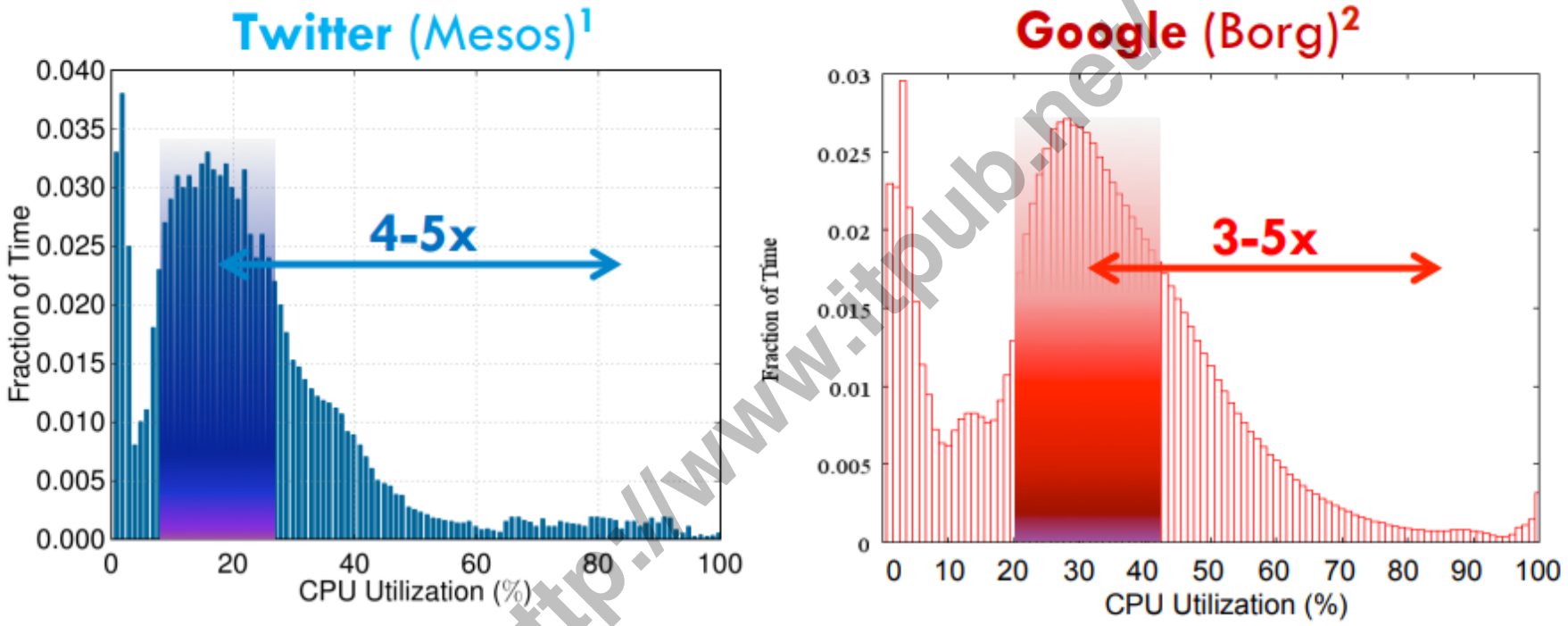
- 资源利用率现状和原因
- Kubernetes Native Feature
- 如何基于Kubernetes进行业务混部
- 落地成果

IDC规模和成本构成

- 数字经济、互联网+、大数据
- IDC行业市场规模快速增长，预计2020年将突破2000亿
- IDC成本中，服务器成本占比达到69%
- 服务器3~5年折旧



资源利用率现状



[1] C. Delimitrou and C. Kozyrakis. Quasar: Resource-Efficient and QoS-Aware Cluster Management, ASPLOS 2014.

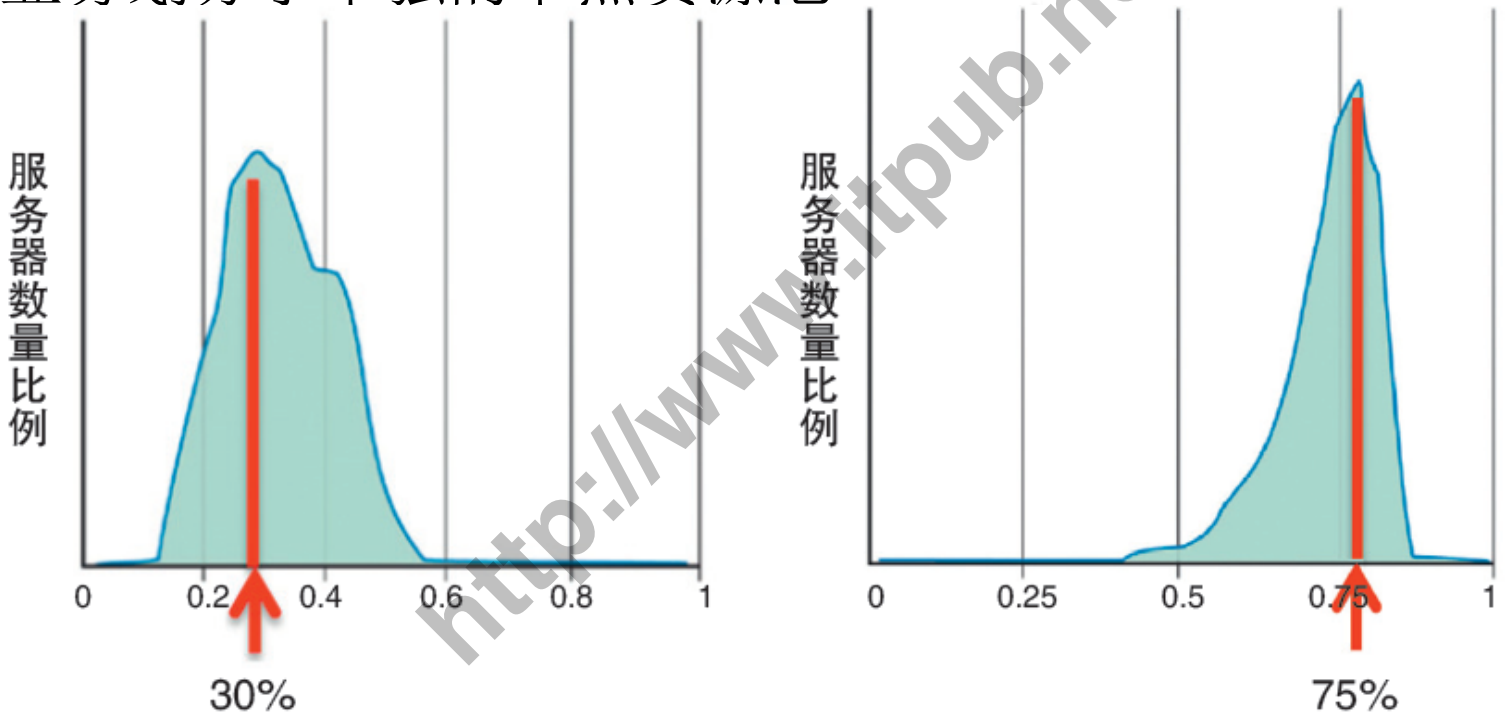
[2] L. A. Barroso, U. Holzle. The Datacenter as a Computer, 2013.

在/离线业务定义

	在线业务	离线业务
举例	搜索，推荐，支付，即时通讯，游戏等	大数据批处理，AI训练，视频转码等
时延	敏感	不敏感
SLO	高	低
容错率	错误容忍度低	错误容忍度高
负载模型	白天负载高，夜间负载低	只要运行，负载就会一直很高

原因分析

- 在/离线业务划分了单独的节点资源池



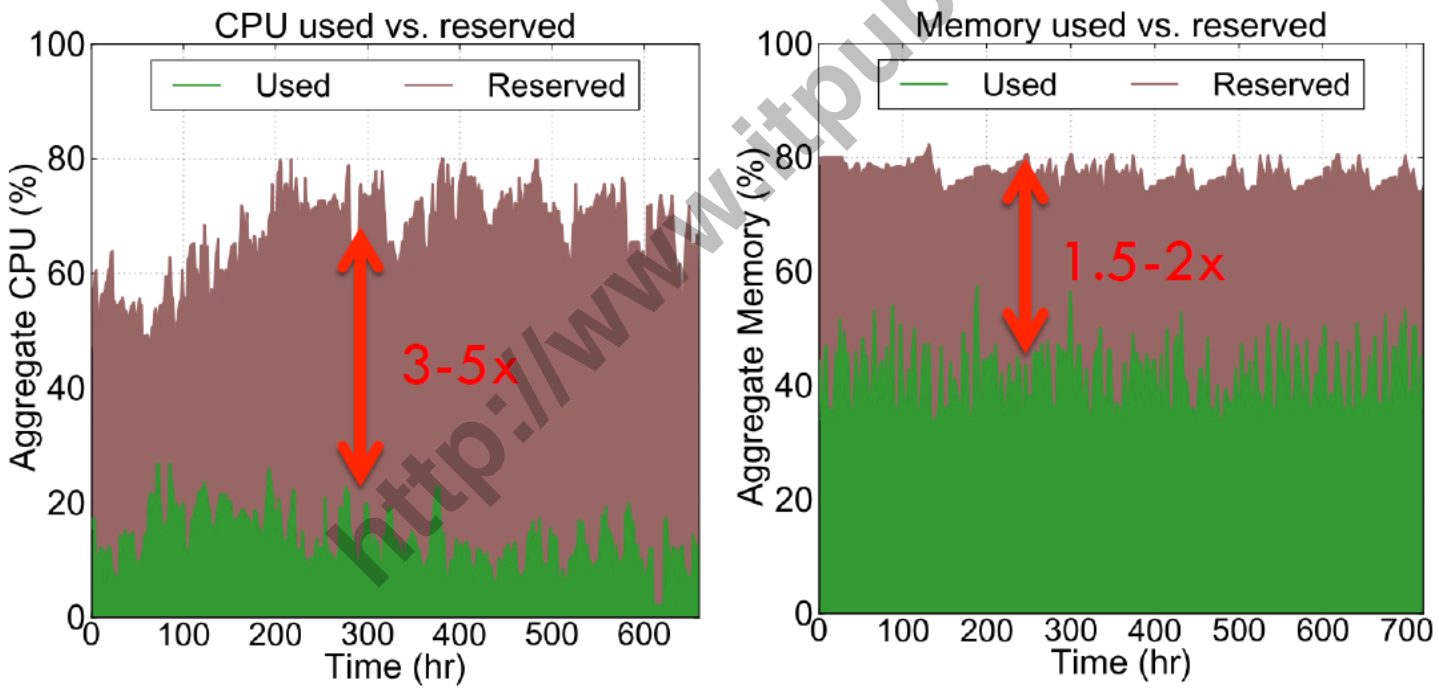
(a) 在线业务服务器　　(b) 离线业务服务器

为什么划分资源池

- 混部会带来底层共享资源（CPU、内存、网络、磁盘等）的竞争
- 在/离线服务分属不同的研发、产品团队，成本管理是分开的
- 在/离线服务使用不同的资源调度管理系统，无法统一调度

原因分析

- 在线业务申请了过多的资源



为什么申请过多资源

- 在线业务为了保证服务的稳定性，都会按照波峰申请资源
- 对于服务的资源使用情况不是特别了解，盲目申请过多的资源
- 三机房容灾部署，单副本需要能够承载所有用户请求
- 因为资源的隔离性不够，业务方希望通过申请过多资源来降低部署密度

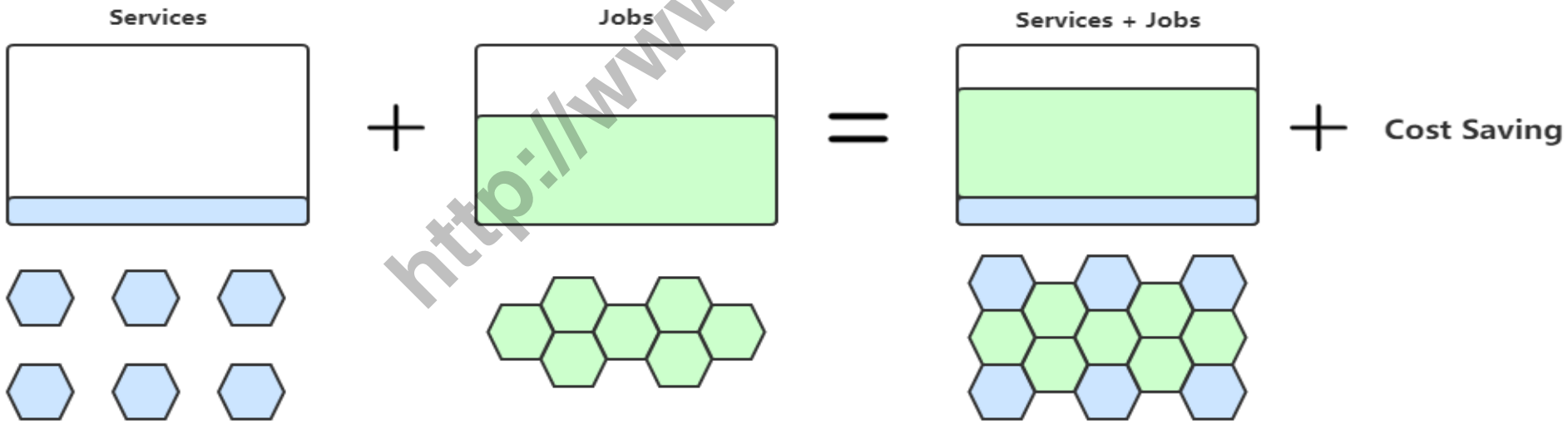
原因分析

- 业务存在波峰波谷，

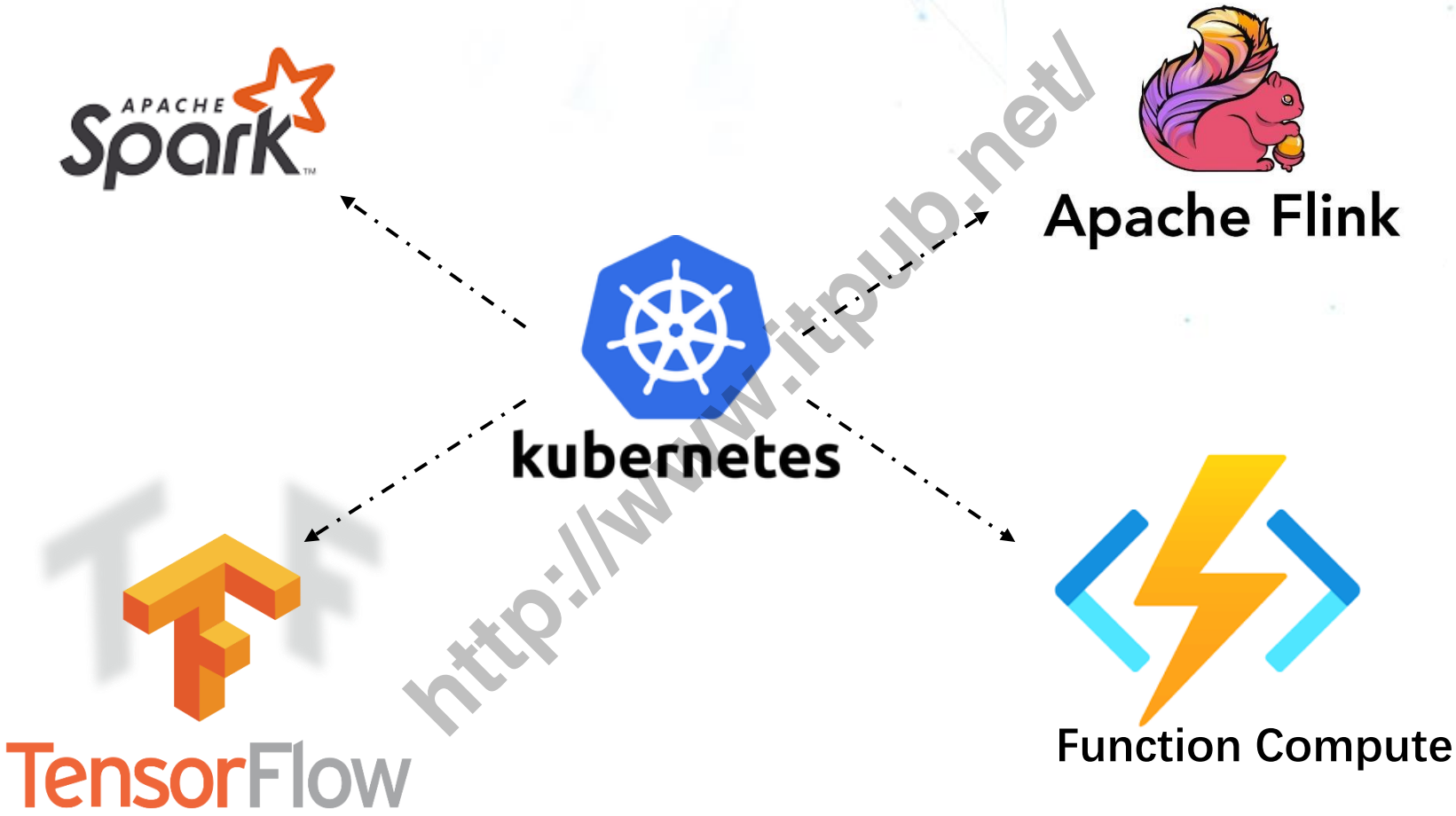
  波谷时大量资源空闲

混部提高资源利用率

- 离线业务不能影响在线业务的SLO
- 在线业务分配的资源是High Quality的，有资源保证
- 离线业务分配的资源是Low Quality的，可随时被在线业务抢占
- 避免离线业务饥饿



Why Kubernetes?



Kubernetes存在的问题

- 静态调度
- 资源隔离性较弱

Kubernetes native feature

Pod Priority

- 表示pod的重要程度，值越大优先级越高
- 调度器调度的时候会优先调度高优先级的pod
- Kubelet在驱逐过载节点的pod时，会优先驱逐低优先级的pod

Kubernetes native feature

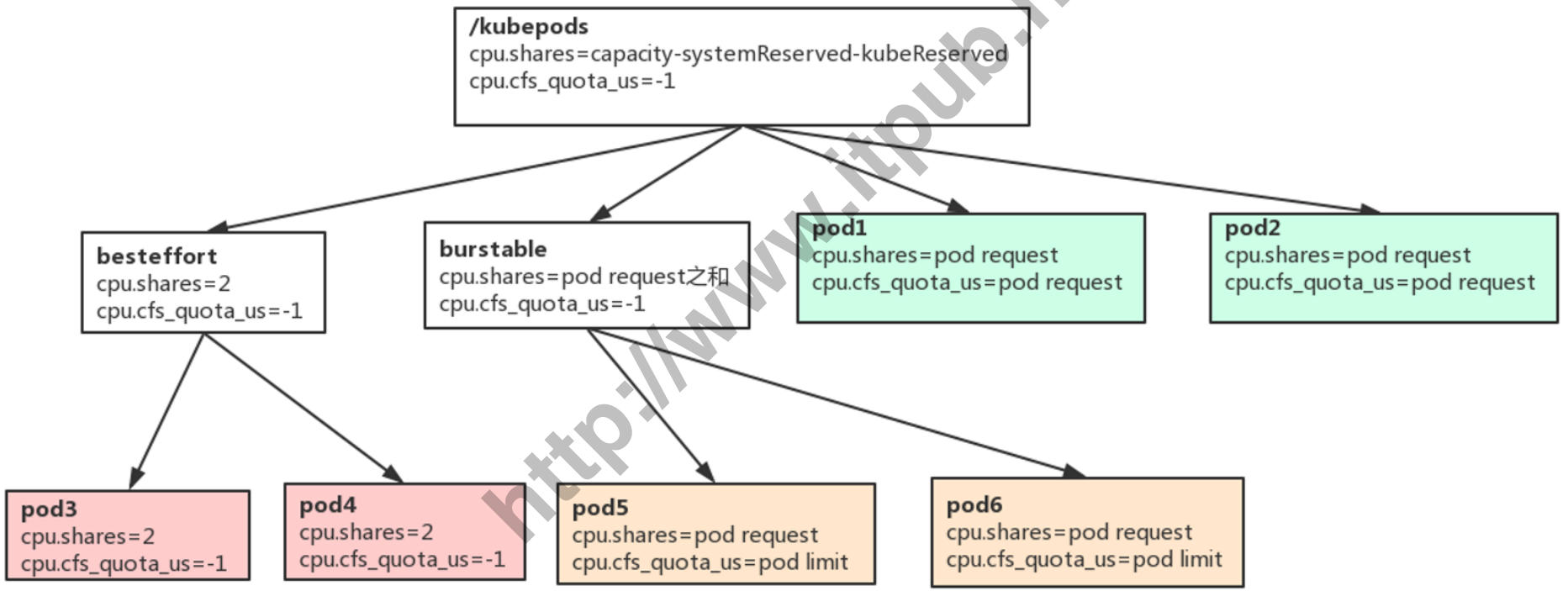
Pod QoS

- Best Effort： cpu/memory request 和 limit都没设置
- Guaranteed：cpu/memory request=limit
- Burstable： 其他

QoS class	oom_score_adj
Guaranteed	-998
BestEffort	1000
Burstable	min(max(2, 1000 - (1000 * memoryRequestBytes) / machineMemoryCapacityBytes), 999)

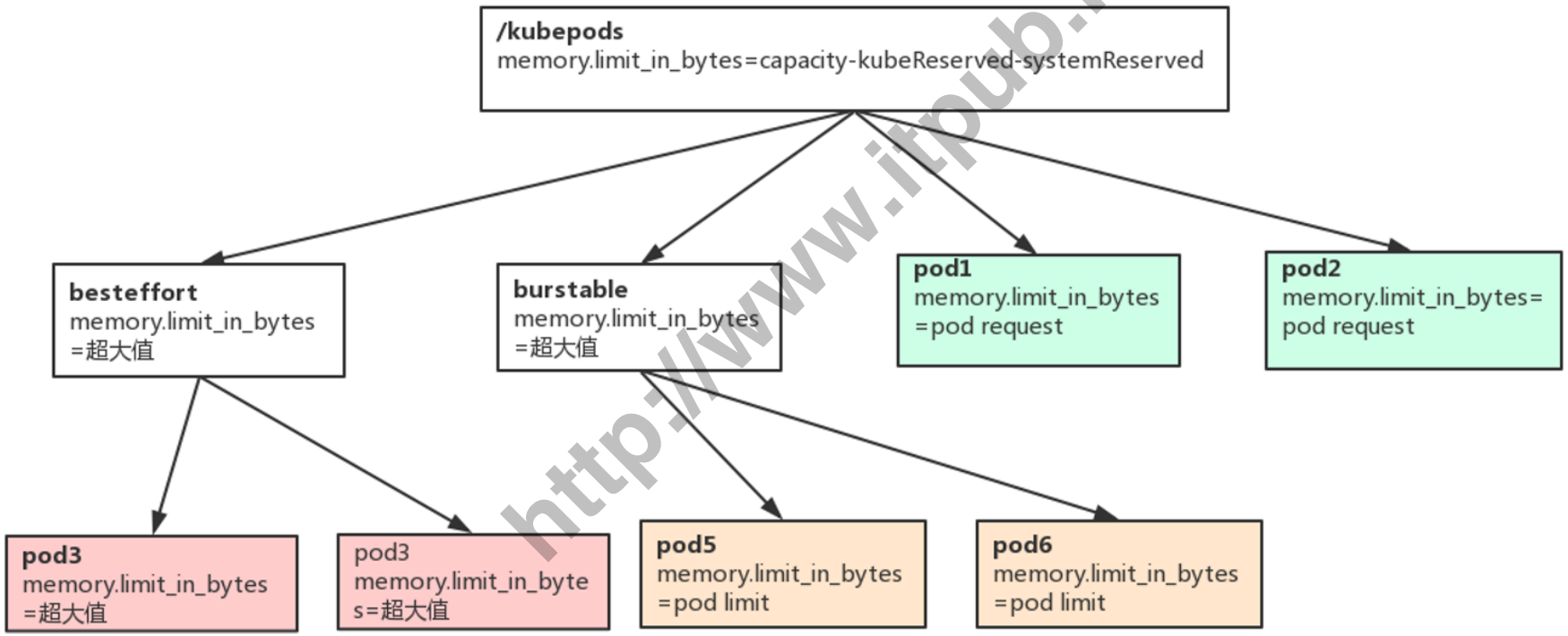
Kubernetes native feature

CPU CGroup



Kubernetes native feature

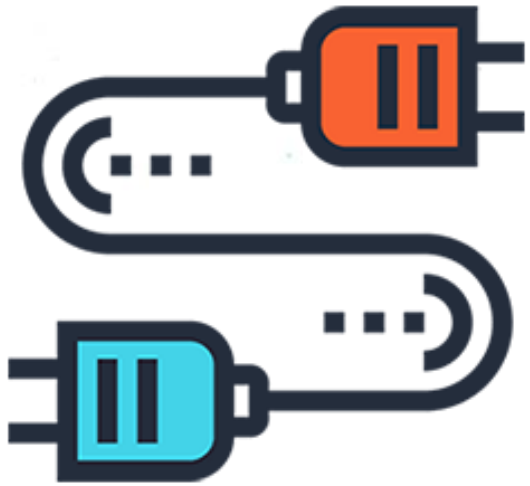
Mem CGroup



Kubernetes native feature

Extension Mechanism

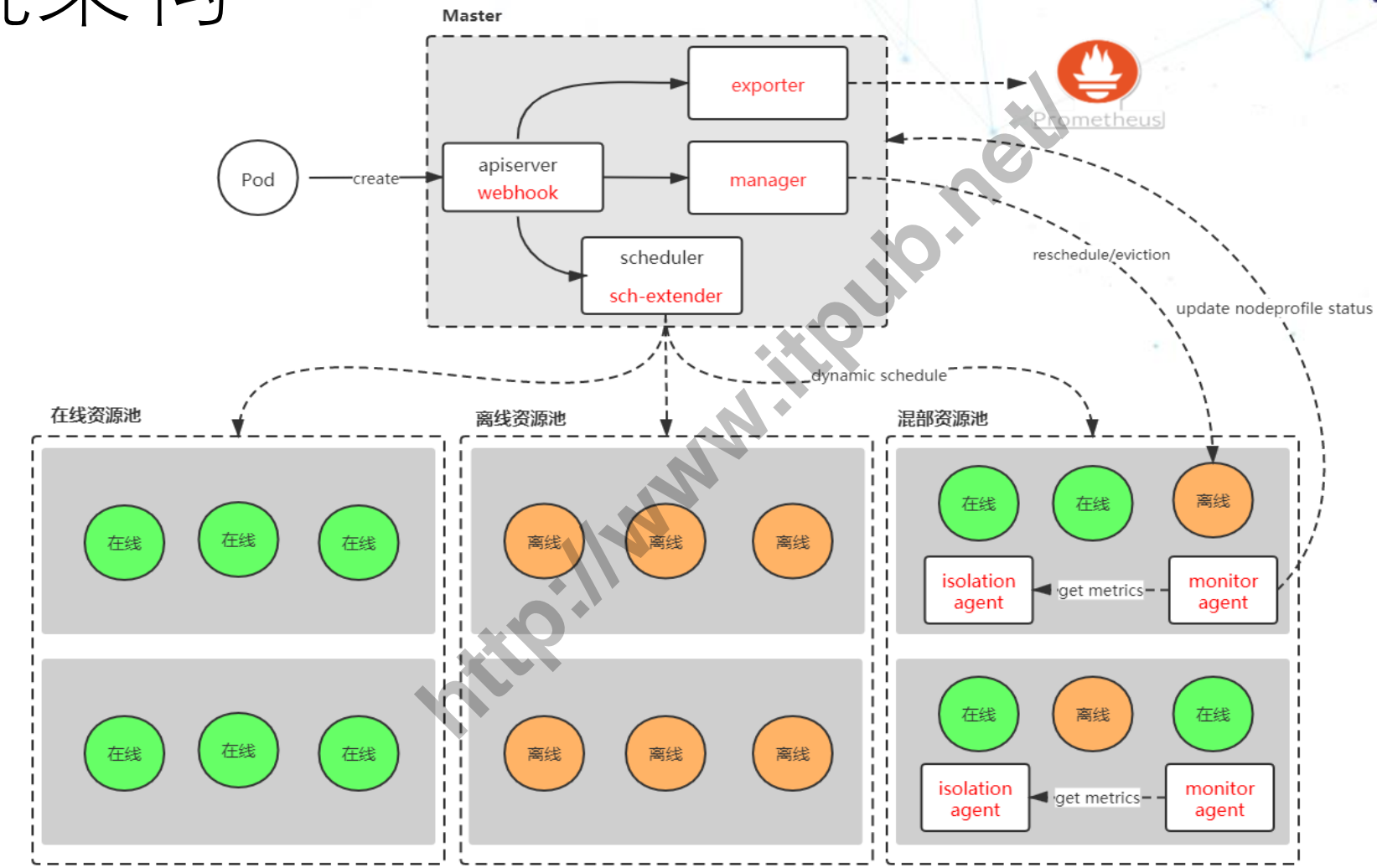
- Dynamic Admission control
  - Validating Webhook
  - Mutating Webhook
- Custom Resource Definition + Operator
- Device Plugin
- Scheduler Extender



遵循的设计原则

- 动态调度
- 动态资源分配和隔离
- 插件化
- 及时响应
- 可运维、可观测

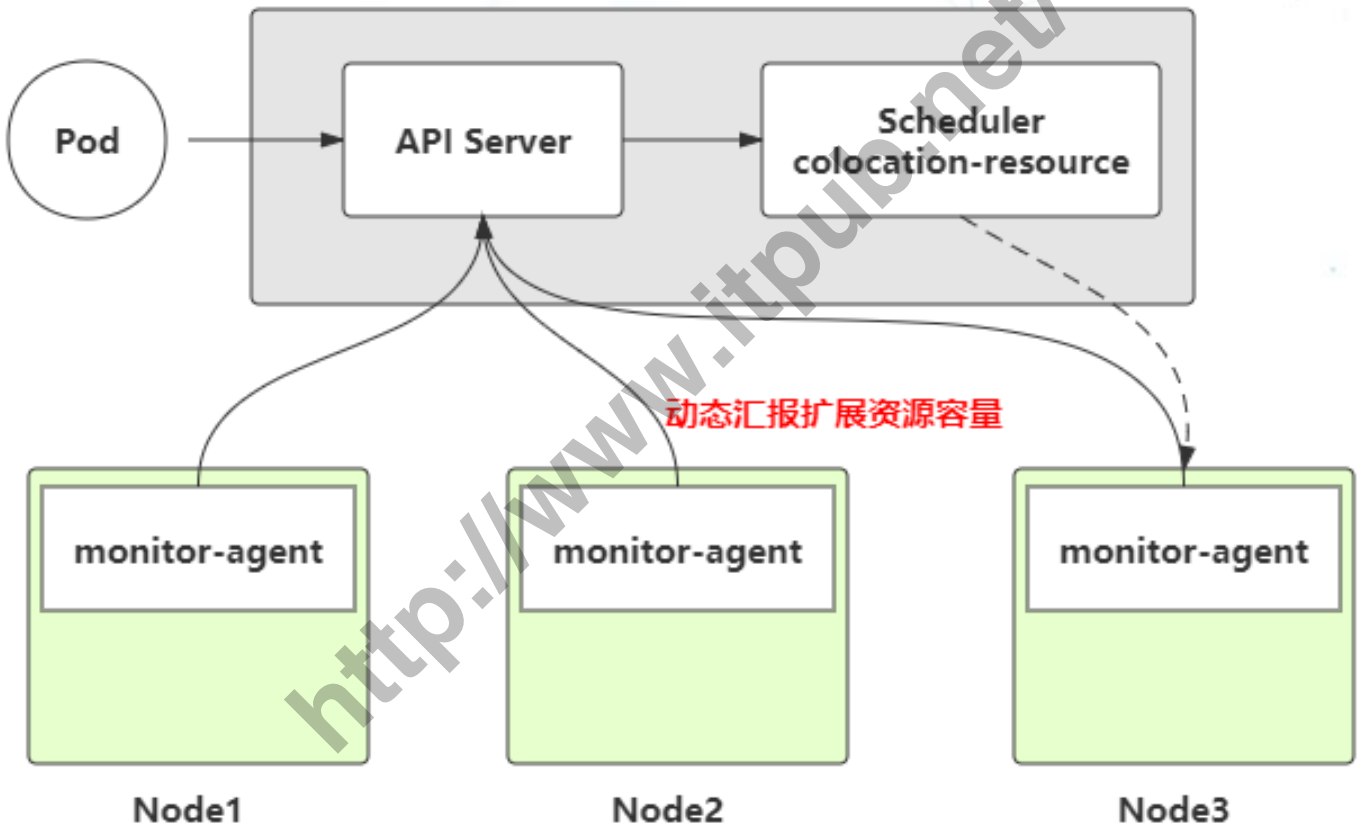
系统架构



Extended Resource

- colocation/cpu和colocation/memory
- 离线任务基于扩展资源进行调度
- 扩展资源是在线业务申请去但是目前空闲的那部分资源
- 这两个资源是Low Quality的，可能会被在线业务收回

Resource Reclaim



Webhook

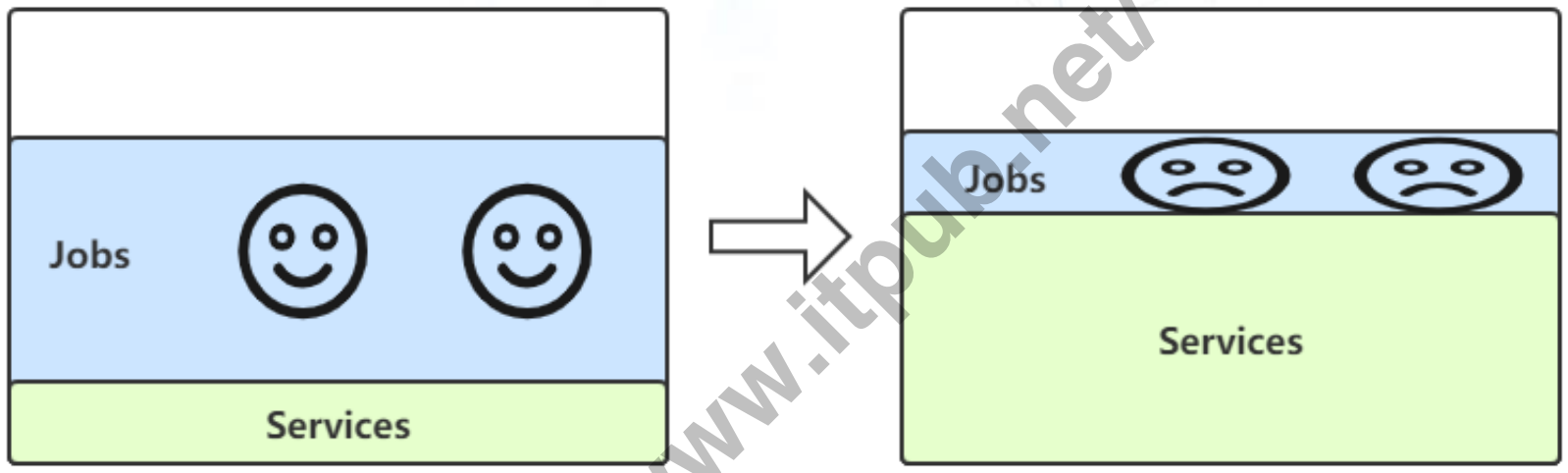
```
apiVersion: v1
kind: Pod
metadata:
    name: example
    labels:
        colocation.netease.com/workload-type: colocation-job
spec:
    resources:
        request.cpu: 2
        request.memory: 4G
```


```
apiVersion: v1
kind: Pod
metadata:
    name: example
    labels:
        colocation.netease.com/workload-type: colocation-job
    annotations:
        colocation/cpu: 2
        colocation/memory: 4G
spec:
    resources: {}
    priority: -100
    nodeSelector:
        colocation.netease.com/node-pool: colocation
```

Dynamic Schedule

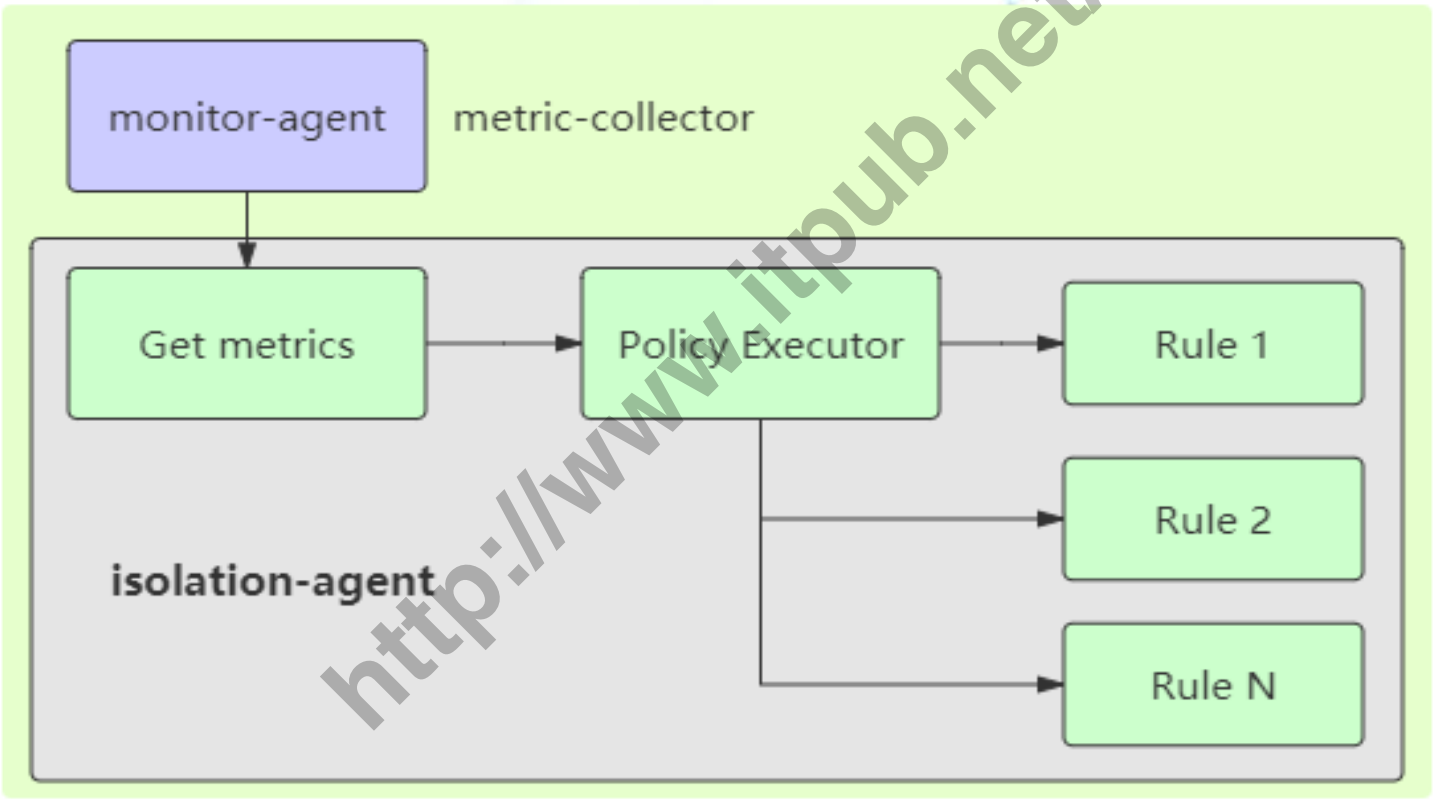
- 不能将离线任务调度到过载节点
- 优先将离线任务调度到较空闲节点

Reschedule



- 避免离线任务饥饿, 提高执行效率
- 减少在/离线业务的资源竞争

Resource Isolation



Resource Isolation

- CPU: cpu share, cfs quota, cpuset
- Memory: memory limit, MBA(Memory Bandwidth Allocation)
- Cache: CAT (Cache Allocation Technology)
- Disk: block & buffer io
- Network: separate NIC and switch

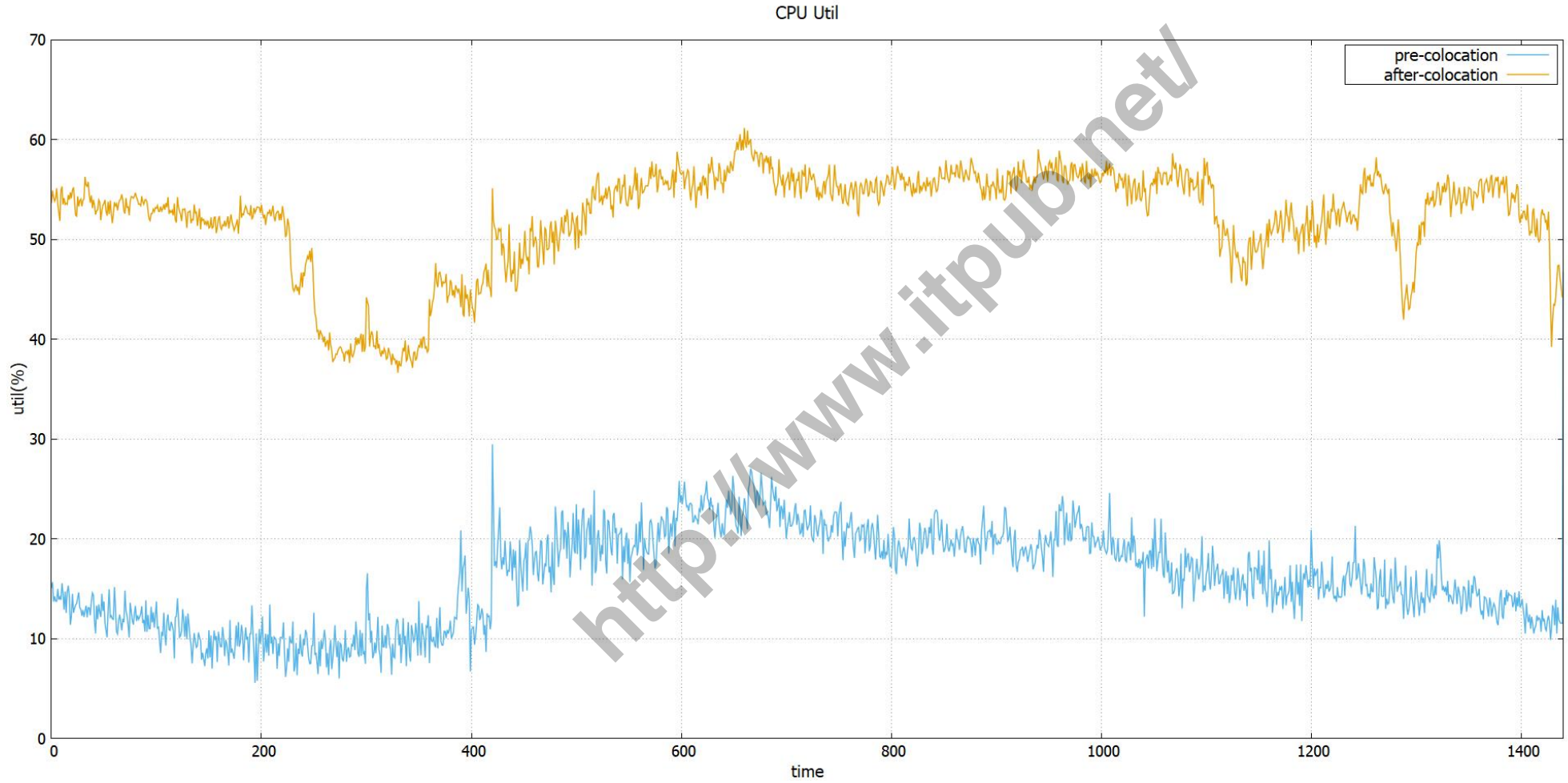
其他相关工作

- HPA
- CronHPA
- Resource Recommender

优势

- 基于Kubernetes扩展机制开发，没有修改Kubernetes一行代码，可以方便地部署到任何一个标准的Kubernetes集群中
- 对业务不做任何假设，不需要业务进行改造，易实施落地
- 配置灵活丰富，满足多种场景的混部需求

落地效果 – CPU利用率



落地效果 – 在线业务RT

-	调用次数	失败次数	平均RT	Ms0_10	Ms10_20	Ms20_n
混部前	10.3E	0	6.59	81.99%	5.62%	12.39%
混部后	10.2E	0	6.65	81.82	5.61%	12.57%

业界研究方向

- 发现、量化、预测以及管理 **性能干扰**
- 基于业务的SLO和实时Metric信息，动态调整资源分配
- 基于历史数据，预测业务的资源需求情况，调整超售策略
- 研究分配给离线业务的Slack Resource的可用性

Thanks