kuberentes源码中调度器设计:

kuberentes调度器设计概述

我们先整体了解一下Scheduler的设计原理，然后再看这些过程是如何用代码实现的。关于调度器的设计在官网有介绍，结合官网给的说明，简化掉不影响理解的复杂部分，和大家介绍一下Scheduler的工作过程。

我们可以看下官网有一段描述如下：

The Kubernetes scheduler runs as a process alongside the other master components such as the API server. Its interface to the API server is to watch for Pods with an empty PodSpec.NodeName, and for each Pod, it posts a binding indicating where the Pod should be scheduled.

Scheduler是一个跑在其他组件边上的独立程序，对接Apiserver寻找PodSpec.NodeName为空的Pod，然后用post的方式发送一个api调用，指定这些pod应该跑在哪个node上。

通俗地说，就是scheduler是相对独立的一个组件，主动访问api server，寻找等待调度的pod，然后通过一系列调度算法寻找哪个node适合跑这个pod，然后将这个pod和node的绑定关系发给api server，从而完成了调度的过程。

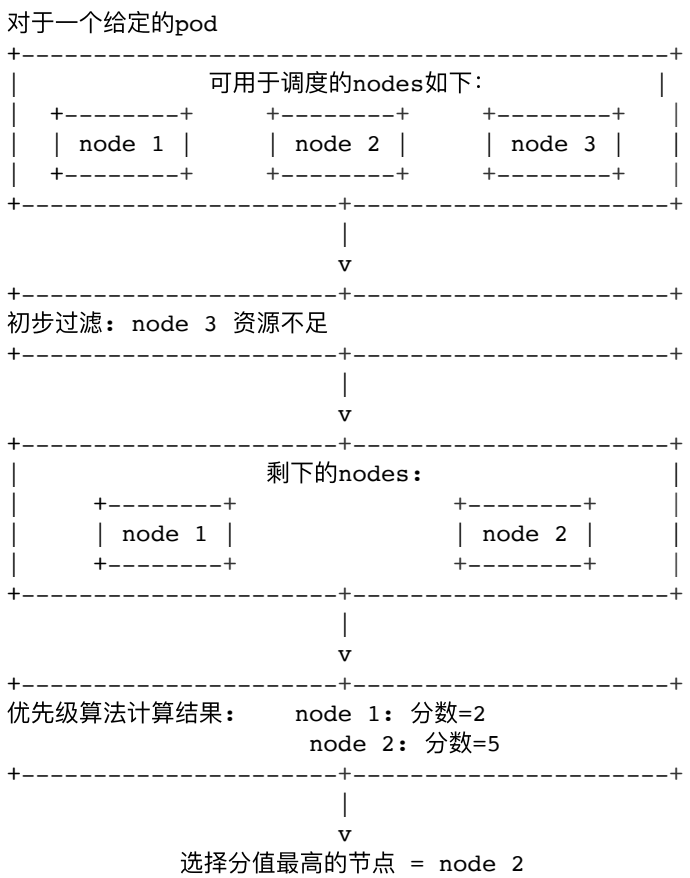
源码层

从源码层看,scheduler的源码可以分为3层：

- cmd/kube-scheduler/scheduler.go: main() 函数入口位置，在scheduler过程开始被调用前的一系列初始化工作。
- pkg/scheduler/scheduler.go: 调度框架的整体逻辑，在具体的调度算法之上的框架性的代码。
- pkg/scheduler/core/generic_scheduler.go: 具体的计算哪些node适合跑哪些pod的算法。

调度算法

调度过程整体：



Scheduler为每个pod寻找一个适合其运行的node，分成三步：

1.通过一系列的“predicates”过滤掉不能运行pod的node，比如一个pod需要500M的内存，有些节点剩余内存只有100M了，就会被剔除；

1. 通过一系列的“priority functions”给剩下的node排一个等级，分出三六九等，寻找能够运行pod的若干node中最合适的一个node；
2. 得分最高的一个node，也就是被“priority functions”选中的node胜出了，获得了跑对应pod的资格。

Predicates 和 priorities 策略

Predicates是一些用于过滤不合适node的策略 . Priorities是一些用于区分node排名（分数）的策略（作用在通过predicates过滤的node上）. Kubernetes默认内建了一些predicates 和 priorities 策略，官方文档介绍地址： scheduler_algorithm.md. Predicates 和 priorities 的代码分别在：

pkg/scheduler/algorithm/predicates/predicates.go pkg/scheduler/algorithm/priorities.

调度策略的修改

kubernetes中默认调度策略是通过defaultPredicates() 和 defaultPriorities()函数定义的，源码在 pkg/scheduler/algorithmprovider/defaults/defaults.go，我们可以通过命令行flag --policy-config-file来覆盖默认行为。

所以我们可以通过配置文件的方式或者修改pkg/scheduler/algorithm/predicates/predicates.go 和/pkg/scheduler/algorithm/priorities，然后注册到defaultPredicates()/defaultPriorities()来实现。配置文件类似下面这个样子：

```
{
"kind" : "Policy",
"apiVersion" : "v1",
"predicates" : [
    {"name" : "PodFitsHostPorts"},
    {"name" : "PodFitsResources"},
    {"name" : "NoDiskConflict"},
    {"name" : "NoVolumeZoneConflict"},
    {"name" : "MatchNodeSelector"},
    {"name" : "HostName"}
    ],
"priorities" : [
    {"name" : "LeastRequestedPriority", "weight" : 1},
    {"name" : "BalancedResourceAllocation", "weight" : 1},
    {"name" : "ServiceSpreadingPriority", "weight" : 1},
    {"name" : "EqualPriority", "weight" : 1}
    ],
"hardPodAffinitySymmetricWeight" : 10,
"alwaysCheckAllPredicates" : false
}
```

这里是对Kubernetes中的Scheduler的介绍,下面我们会详细分析.