Taint 和 Toleration

节点亲和性，是 *pod* 的一种属性（偏好或硬性要求），它使 *pod* 被吸引到一类特定的节点。Taint 则相反，它使 *节点* 能够 *排斥* 一类特定的 pod

Taint 和 toleration 相互配合，可以用来避免 pod 被分配到不合适的节点上。每个节点上都可以应用一个或多个 taint ，这表示对于那些不能容忍这些 taint 的 pod，是不会被该节点接受的。如果将 toleration 应用于 pod 上，则表示这些 pod 可以（但不要求）被调度到具有匹配 taint 的节点上

污点(Taint)

Ⅰ、 污点 (Taint) 的组成

使用 `kubectl taint` 命令可以给某个 Node 节点设置污点，Node 被设置上污点之后就和 Pod 之间存在了一种相斥的关系，可以让 Node 拒绝 Pod 的调度执行，甚至将 Node 已经存在的 Pod 驱逐出去

每个污点的组成如下：

```
key=value:effect
```

每个污点有一个 key 和 value 作为污点的标签，其中 value 可以为空，effect 描述污点的作用。当前 taint effect 支持如下三个选项：

- **`NoSchedule`：表示 k8s 将不会将 Pod 调度到具有该污点的 Node 上**
- **`PreferNoSchedule`：表示 k8s 将尽量避免将 Pod 调度到具有该污点的 Node 上**
- **`NoExecute`：表示 k8s 将不会将 Pod 调度到具有该污点的 Node 上，同时会将 Node 上已经存在的 Pod 驱逐出去**

Ⅱ、污点的设置、查看和去除

```
# 设置污点
kubectl taint nodes node1 key1=value1:NoSchedule

# 节点说明中，查找 Taints 字段
kubectl describe pod  pod-name

# 去除污点
kubectl taint nodes node1 key1:NoSchedule-
```

容忍(Tolerations)

设置了污点的 Node 将根据 taint 的 effect：NoSchedule、PreferNoSchedule、NoExecute 和 Pod 之间产生互斥的关系，Pod 将在一定程度上不会被调度到 Node 上。 但我们可以在 Pod 上设置容忍 (Toleration)，意思是设置了容忍的 Pod 将可以容忍污点的存在，可以被调度到存在污点的 Node 上

pod.spec.tolerations

```
tolerations:
- key: "key1"
  operator: "Equal"
  value: "value1"
  effect: "NoSchedule"
  tolerationSeconds: 3600
- key: "key1"
  operator: "Equal"
  value: "value1"
  effect: "NoExecute"
- key: "key2"
  operator: "Exists"
  effect: "NoSchedule"
```

- **其中 key, vaule, effect 要与 Node 上设置的 taint 保持一致**
- **operator 的值为 Exists 将会忽略 value 值**
- **tolerationSeconds 用于描述当 Pod 需要被驱逐时可以在 Pod 上继续保留运行的时间**

Ⅰ、当不指定 key 值时，表示容忍所有的污点 key：

```
tolerations:
- operator: "Exists"
```

Ⅱ、当不指定 effect 值时，表示容忍所有的污点作用

```
tolerations:
- key: "key"
  operator: "Exists"
```

Ⅲ、有多个 Master 存在时，防止资源浪费，可以如下设置

```
kubectl taint nodes Node-Name node-role.kubernetes.io/master=:PreferNoSchedule
```