Node Maintenance

- It is the **Node Controller** that is responsible for managing the Node objects
  - It assigns **IP space** to the node when a new node is launched
  - It keeps the **node list** up to date with the available machines
  - The node controller is also monitoring the **health of the node**
    - If a node is **unhealthy it gets deleted**
    - Pods running on the unhealthy node will then get **rescheduled**

Node Maintenance

- When adding a new node, the **kubelet** will attempt to register itself
- This is called **self-registration** and is the default behavior
- It allows you to **easily add more nodes** to the cluster without making API changes yourself
- A new node object is **automatically** created with:
  - The metadata (with a name: IP or hostname)
  - Labels (e.g. cloud region / availability zone / instance size)

Node Maintenance

- When you want to **decommission** a node, you want to do it gracefully
  - You drain a node before you shut it down or take it out of the cluster
- To drain a node, you can use the following command:

```
$ kubectl drain nodename --grace-period=600
```

- If the node runs pods not managed by a controller, but is just a single pod:

```
$ kubectl drain nodename --force
```

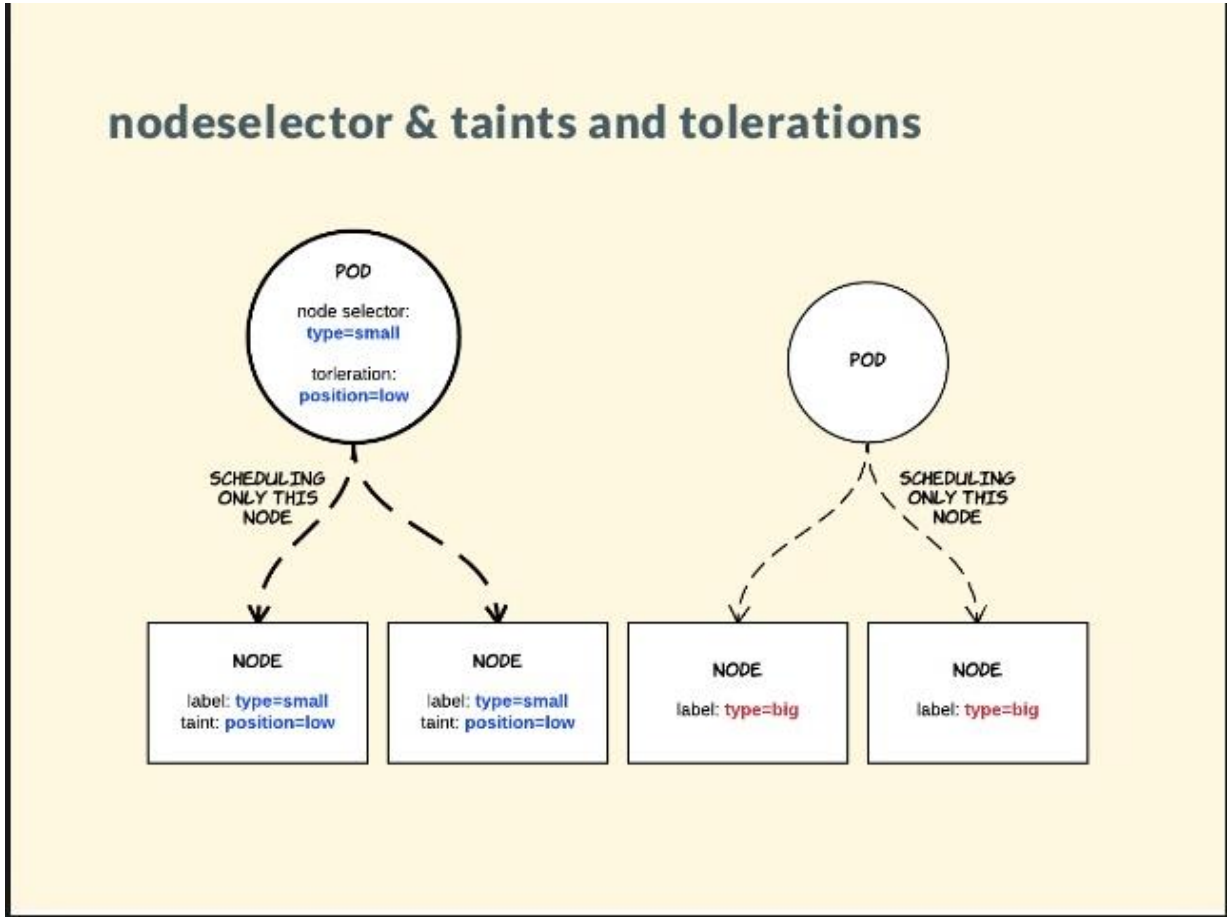
Demo Placeholder

- Drain the node

NodeSelector

- nodeSelector is the simplest form of constraint. nodeSelector is a field of PodSpec.
- It specifies a map of key-value pairs.
- For the pod to be eligible to run on a node, the node must have each of the indicated key-value pairs as labels (it can have additional labels as well).
- **Give some customize label for classify host and define some selector criteria for specific node run Pods**

NodeSelector



Taint and Tolerations

Node side consideration • Force/Repel Pods from Node

Taint will apply to Node for protect “node” to run any pods without “tolerate” match taint

Tolerations apply to Pods for suggest (Not required) Pods to schedule

Use Case: o Dedicated Node / Maintenance Node o Special Hardware Node

Scheduling Controlled | Node Constraints

- Host constraints
- Labels and node selectors
- Taints and tolerations


```
kind: Pod
spec:
  tolerations:
  - key: error
    value: disk
    operator: Equal
    effect: NoExecute
    tolerationSeconds: 60
```

```
kind: Node
spec:
  taints:
  - effect: NoSchedule
    key: error
    value: disk
    timeAdded: null
```

Scheduling Controlled | Taints

Taints communicate

node conditions

- Key – condition category
- Value – specific condition
- Operator – value wildcard
  - Equal
  - Exists
- Effect
  - NoSchedule – filter at scheduling time
  - PreferNoSchedule – prioritize at scheduling time
  - NoExecute – filter at scheduling time, evict if executing
- TolerationSeconds – time to tolerate “NoExecute” taint

```
kind: Pod
spec:
  tolerations:
  - key: <taint key>
    value: <taint value>
    operator: <match operator>
    effect: <taint effect>
    tolerationSeconds: 60
```