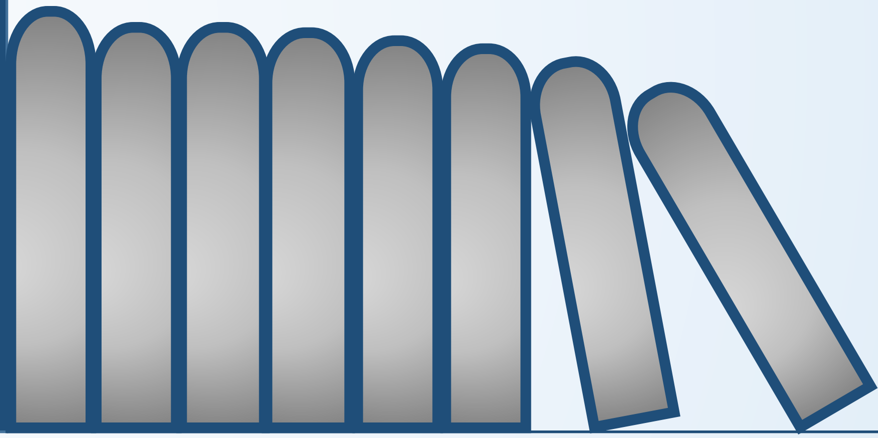
Kubernetes运维

主讲人：宋小金



目录

1 Kubernetes监控

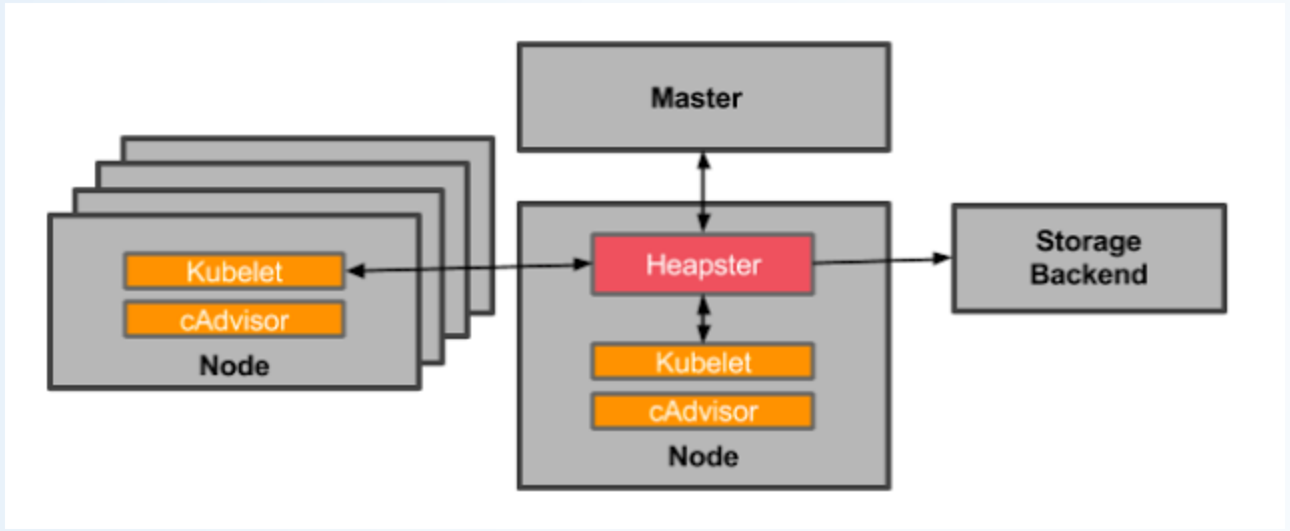
2 常见问题定位

3 Kubernetes常规运维

预期收获

- *了解Kubernetes网络模型*
- 了解Kubernetes网络通信

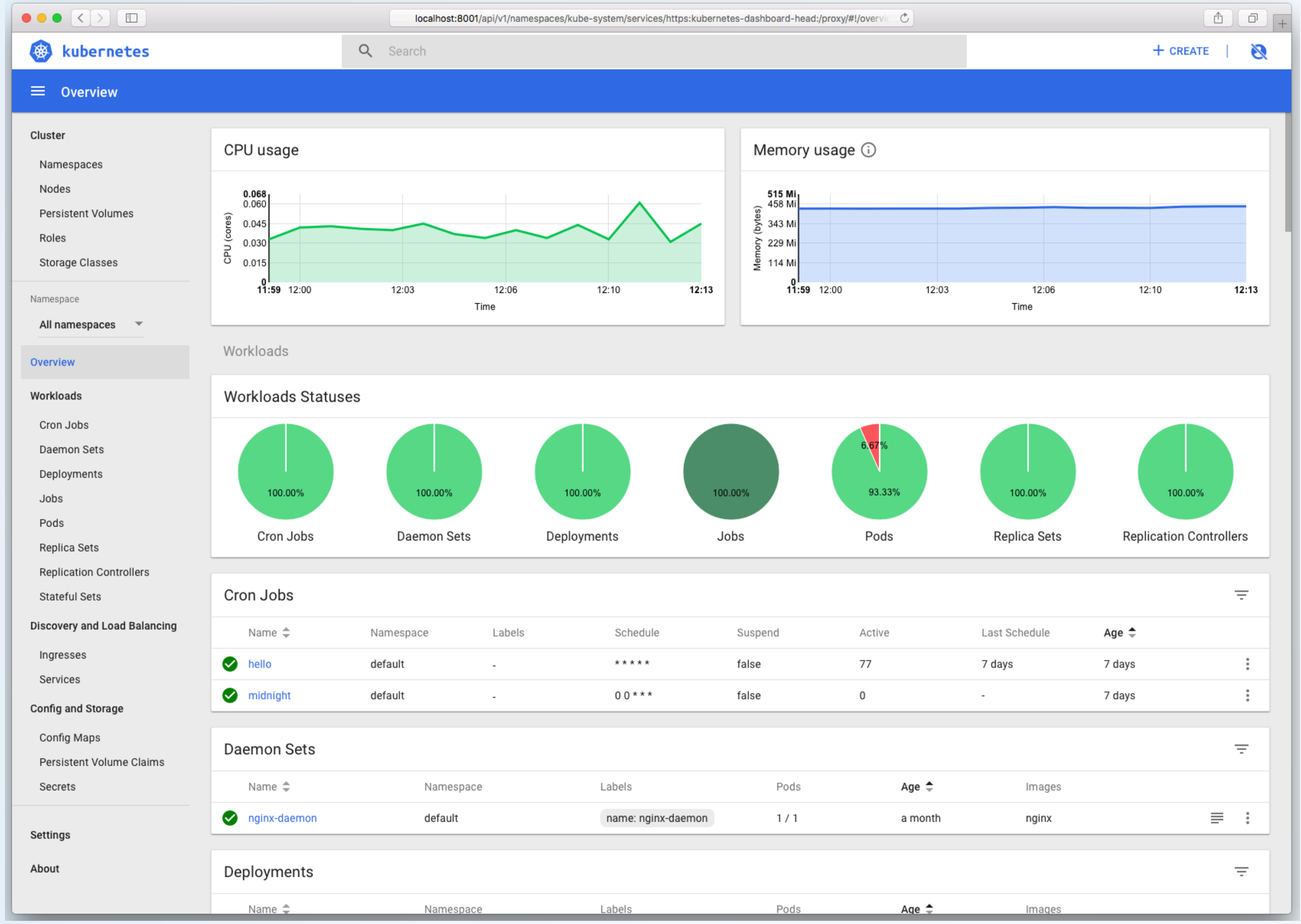
Heapster + cAdvisor监控集群组件



对接了heapster或metrics-server后展示Node CPU/内存/存储资源消耗：
$ kubectl top node {node name}

cAdvisor既能收集容器CPU、内存、文件系统和网络使用统计信息，还能采集节点资源使用情况；
cAdvisor和Heapster都不能进行数据存储、趋势分析和报警。因此，还需要将数据推送到InfluxDB，Grafana等后端进行存储和图形化展示。
Heapster即将被metrics-server替代

Kuberneets Dashboard UI



Cron Jobs

	Name	Namespace	Labels	Schedule	Suspend	Active	Last Schedule	Age
✓	hello	default	-	* * * * *	false	77	7 days	7 days
✓	midnight	default	-	0 0 * * *	false	0	-	7 days

Daemon Sets

	Name	Namespace	Labels	Pods	Age	Images
✓	nginx-daemon	default	name: nginx-daemon	1 / 1	a month	nginx

Deployments

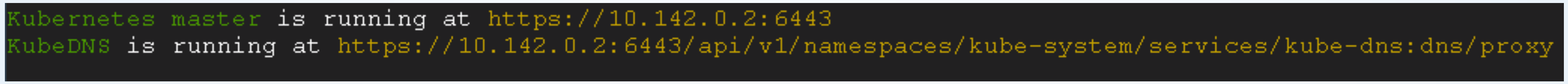
Name	Namespace	Labels	Pods	Age	Images

Kubernetes Dashboard用于监控/展示Kubernetes所有的资源对象：
Cluster（Node，PV等）
Workload（Pod，Deployment等）
Config（Configmap，Secrets等）
…

监控集群组件

集群整体状态：

```
$ kubectl cluster-info
```



```
Kubernetes master is running at https://10.142.0.2:6443
KubeDNS is running at https://10.142.0.2:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

更多集群信息：

```
$ kubectl cluster-info dump
```

通过插件部署：

```
$ kubectl get pod etcd -n kube-system
$ kubectl describe pod kube-apiserver -n kube-system
```

组件metrics：

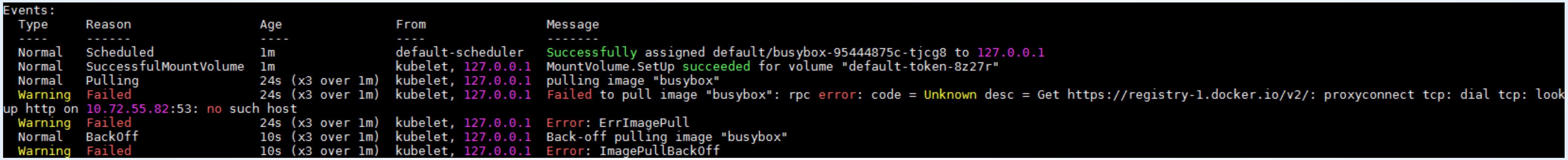
```
$ curl localhost:10250/stats/summary
```

组件健康状况：

```
$ curl localhost:10250/healthz
```

监控应用

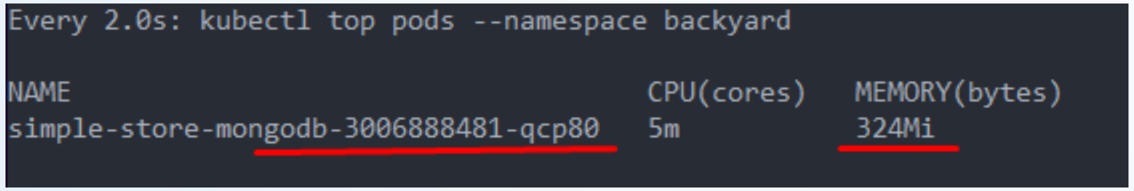
$ kubectl describe pod



```
Events:
  Type     Reason                 Age                From                  Message
  ----     ------                 ----               ----                  -------
  Normal   Scheduled              1m                 default-scheduler     Successfully assigned default/busybox-95444875c-tjcg8 to 127.0.0.1
  Normal   SuccessfulMountVolume  1m                 kubelet, 127.0.0.1    MountVolume.SetUp succeeded for volume "default-token-8z27r"
  Normal   Pulling                24s (x3 over 1m)   kubelet, 127.0.0.1    pulling image "busybox"
  Warning  Failed                 24s (x3 over 1m)   kubelet, 127.0.0.1    Failed to pull image "busybox": rpc error: code = Unknown desc = Get https://registry-1.docker.io/v2/: proxyconnect tcp: dial tcp: look
up http on 10.72.55.82:53: no such host
  Warning  Failed                 24s (x3 over 1m)   kubelet, 127.0.0.1    Error: ErrImagePull
  Normal   BackOff                10s (x3 over 1m)   kubelet, 127.0.0.1    Back-off pulling image "busybox"
  Warning  Failed                 10s (x3 over 1m)   kubelet, 127.0.0.1    Error: ImagePullBackOff
```

对接了heapster或metrics-server后，展示Pod CPU/内存/存储资源消耗：

$ kubectl top pod {pod name}



```
Every 2.0s: kubectl top pods --namespace backyard

NAME                                  CPU(cores)   MEMORY(bytes)
simple-store-mongodb-3006888481-qcp80   5m           324Mi
```

$ kubectl get pod {pod name} --watch

管理K8S组件日志

组件日志：

```
/var/log/kube-apiserver.log
/var/log/kube-proxy.log
/var/log/kube-controller-manager.log
/var/log/kubelet.log
```

使用systemd管理：

```
$ journalctl –u kubelet
```

使用K8S插件部署：

```
$ kubectl logs -f kube-proxy
```

管理K8S组件日志

# 从容器标准输出截获：
```
$ kubectl logs -f {pod name} –c {container name}
$ docker logs -f {docker name}
```

# 日志文件挂载到主机目录：
```
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: gcr.io/google_containers/test-webserver
    name: test-container
    volumeMounts:
    - mountPath:  /log
      name: log-volume
  volumes:
  - name: log-volume
    hostPath:
      # directory location on host
      path:  /var/k8s/log
```

直接进入容器内查看日志：
```
$ kubectl exec -it {pod} -c {container} /bin/sh
$ docker exec -it {container} /bin/sh
```

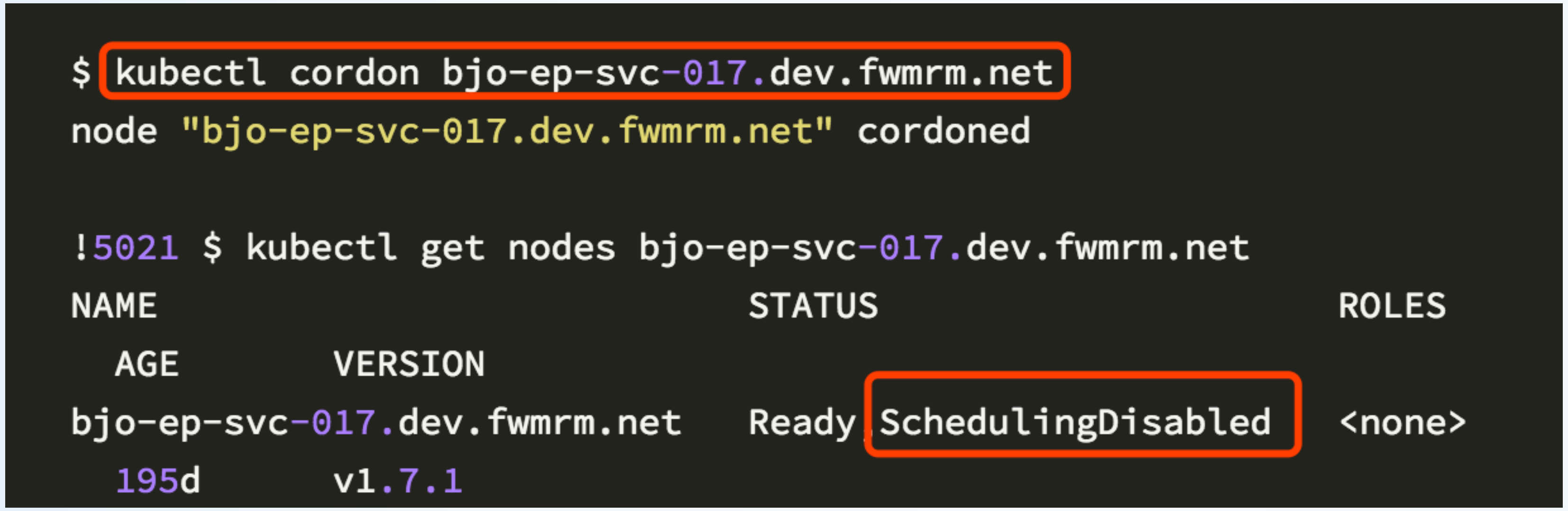
应用自恢复：restartPolicy + livenessProbe

Pod Restart Policy : Always, OnFailure, Never
livenessProbe : http/https Get, shell exec, tcpSocket
# tcp socket的liveness探针 + always restart例子

```
apiVersion: v1
kind: Pod
metadata:
  name: goproxy
spec:
  restartPolicy: Always
  containers:
  - name: goproxy
    image: k8s.gcr.io/goproxy:0.1
    ports:
    - containerPort: 8080
    livenessProbe:
      tcpSocket:
        port: 8080
      initialDelaySeconds: 15
      periodSeconds: 20
```

Node隔离与恢复

在某些场景下需要对Node进行隔离，比如硬件升级或维护，目前隔离node有量种方式：



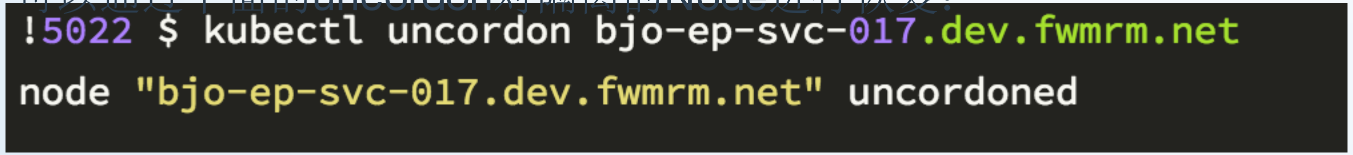
```
$ kubectl cordon bjo-ep-svc-017.dev.fwmrm.net
node "bjo-ep-svc-017.dev.fwmrm.net" cordoned

!5021 $ kubectl get nodes bjo-ep-svc-017.dev.fwmrm.net
NAME                           STATUS                     ROLES
  AGE       VERSION
bjo-ep-svc-017.dev.fwmrm.net   Ready SchedulingDisabled   <none>
  195d      v1.7.1
```

隔离操作并不会停止或删除正在运行的Pod，需要人工介入手动停止或删除，比如下面例子，尽管已经对对Node进行了隔离，

但其上面运行的Pod的状态并没有受到任何影响：

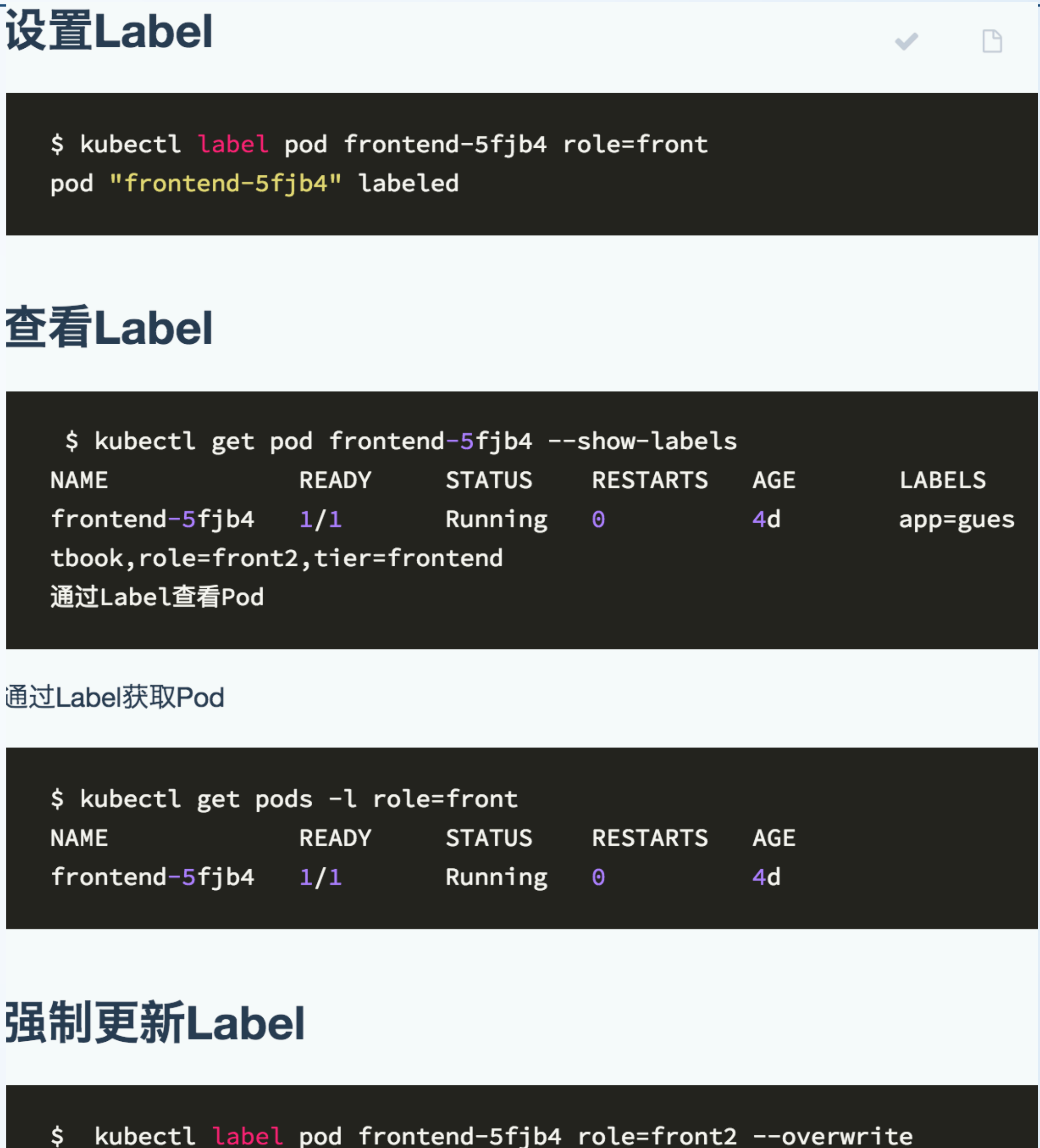
可以通过下面的uncordon对隔离的Node进行恢复：



```
!5022 $ kubectl uncordon bjo-ep-svc-017.dev.fwmrm.net
node "bjo-ep-svc-017.dev.fwmrm.net" uncordoned
```

更新资源对象Label

Kubernetes全部资源对象的Label（key=value形式）都可以随时随地的增加、修改或删除，一个资源对象可以有多个Label，其中key不可以重复，但value可以重复。



设置Label

```
$ kubectl label pod frontend-5fjb4 role=front
pod "frontend-5fjb4" labeled
```

查看Label

```
 $ kubectl get pod frontend-5fjb4 --show-labels
NAME             READY     STATUS    RESTARTS   AGE       LABELS
frontend-5fjb4   1/1       Running   0          4d        app=gues
tbook,role=front2,tier=frontend
通过Label查看Pod
```

通过Label获取Pod

```
$ kubectl get pods -l role=front
NAME             READY     STATUS    RESTARTS   AGE
frontend-5fjb4   1/1       Running   0          4d
```

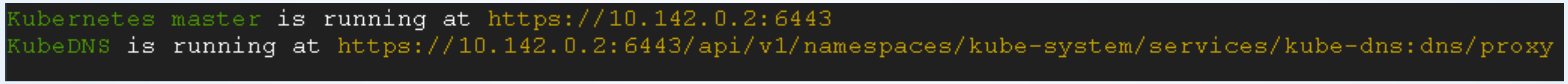
强制更新Label

```
$  kubectl label pod frontend-5fjb4 role=front2 --overwrite
```

监控集群组件

集群整体状态：

```
$ kubectl cluster-info
```



```
Kubernetes master is running at https://10.142.0.2:6443
KubeDNS is running at https://10.142.0.2:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

更多集群信息：

```
$ kubectl cluster-info dump
```

通过插件部署：

```
$ kubectl get pod etcd -n kube-system
$ kubectl describe pod kube-apiserver -n kube-system
```

组件metrics：

```
$ curl localhost:10250/stats/summary
```

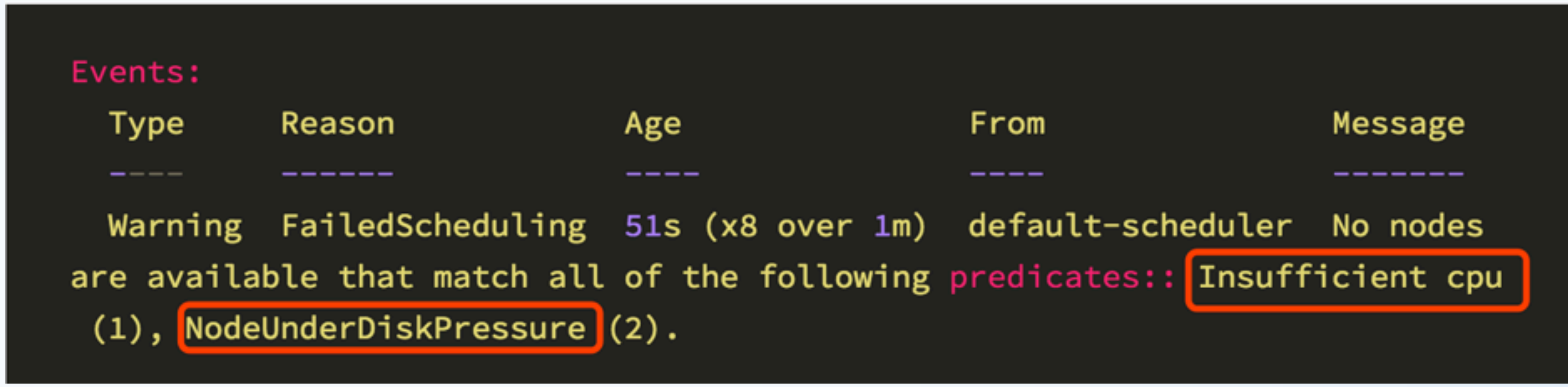
组件健康状况：

```
$ curl localhost:10250/healthz
```

常见问题定位

问题： Pod Pending

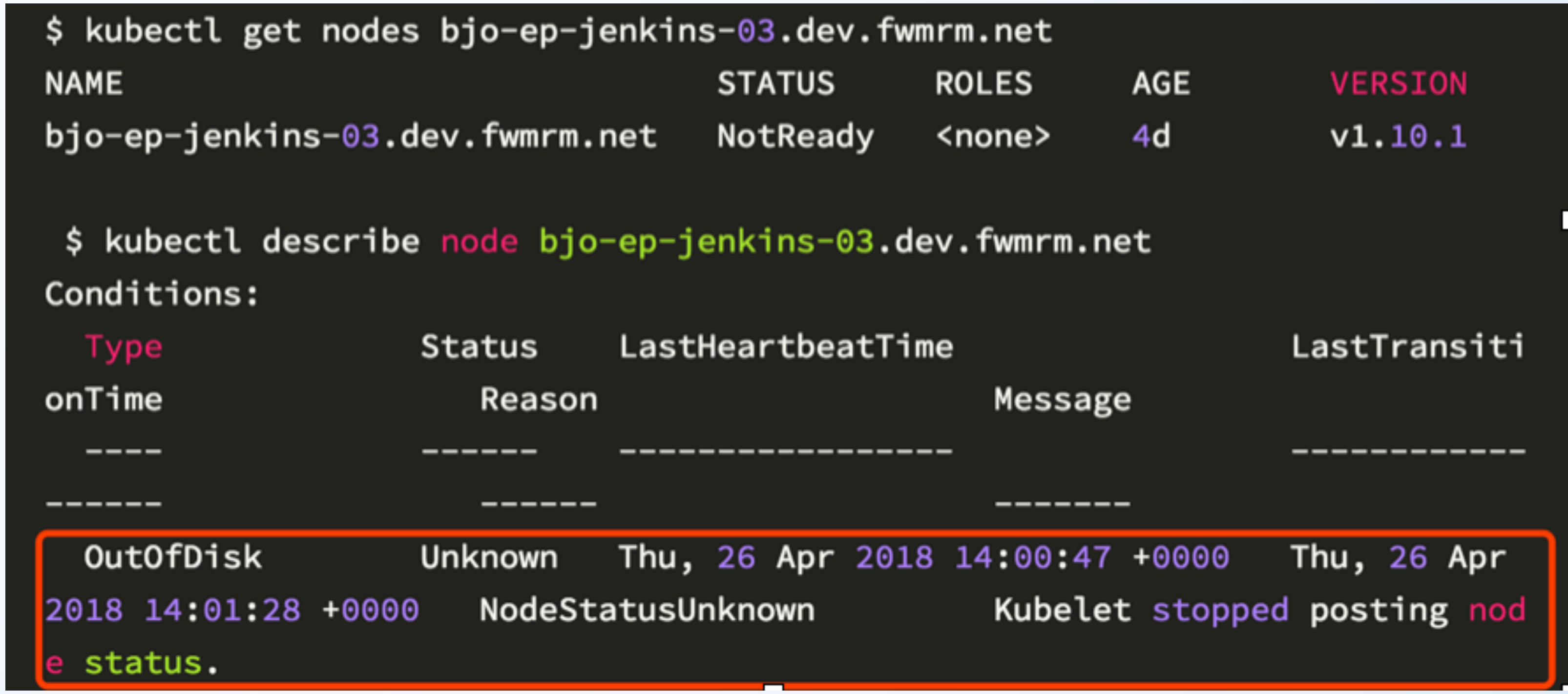
定位：通过 *kubectl describe pod pod-xxx* 发现**CPU资源不足**造成的



```
Events:
  Type     Reason            Age                From               Message
  ----     ------            ----               ----               -------
  Warning  FailedScheduling  51s (x8 over 1m)   default-scheduler  No nodes
are available that match all of the following predicates:: Insufficient cpu
 (1), NodeUnderDiskPressure (2).
```

问题：Node状态为NotReady

定位：kubectl describe node node-xxx，提示kubelet不发送node状态数据，原因是该节点**kubelet已停**掉。

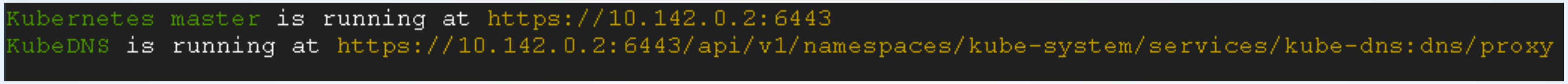


```
$ kubectl get nodes bjo-ep-jenkins-03.dev.fwmrm.net
NAME                              STATUS     ROLES     AGE       VERSION
bjo-ep-jenkins-03.dev.fwmrm.net   NotReady   <none>    4d        v1.10.1

 $ kubectl describe node bjo-ep-jenkins-03.dev.fwmrm.net
Conditions:
  Type             Status    LastHeartbeatTime                     LastTransiti
onTime                 Reason                         Message
  ----             ------    -----------------                     ------------
------                 ------                         -------
  OutOfDisk        Unknown   Thu, 26 Apr 2018 14:00:47 +0000       Thu, 26 Apr
2018 14:01:28 +0000    NodeStatusUnknown              Kubelet stopped posting nod
e status.
```

常见问题定位

集群整体状态：
$ kubectl cluster-info



```
Kubernetes master is running at https://10.142.0.2:6443
KubeDNS is running at https://10.142.0.2:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

更多集群信息：
$ kubectl cluster-info dump

通过插件部署：
$ kubectl get pod etcd -n kube-system
$ kubectl describe pod kube-apiserver -n kube-system

组件metrics：
$ curl localhost:10250/stats/summary

组件健康状况：
$ curl localhost:10250/healthz

课程回顾

已学知识要点

了解Kubernetes的常见问题定位