# K8s 故障排查 Cheat Sheet

AUTHOR: 彭玲 TIME: 2022/1/19



# 常用命令

## Nodes

```
# 查看节点
> kubectl get node [-o wide]

# 查看节点事件
> kubectl describe node ${k8s-node} --show-events
```

## Pods

```
# 查看 k8s pod
> kubectl get pod -n kube-system -o wide

# 查看 k8s pod 详情 (容器启动前的信息)
> kubectl describe pod ${pod_name} -n kube-system

# 查看 pod 日志 (容器启动后，应用内部的日志)
> kubectl logs ${pod_name} -n kube-system
```

## 其他

```
# 查看日志 (基于 systemd 的系统)
> journalctl -u kubelet -f

# 查看配置
> kubectl get pod ${pod_name} [-n ${namespace}] -o yaml
> kubectl get svc ${svc_name} [-n ${namespace}] -o yaml
> kubectl get deployments ${deployment} -o yaml
```

# 常见问题

## Node `NotReady`

### flannel 网络问题

如果是新加入的节点，查看下 flannel 网络是否可以初始化，如果不能，可能就是缺少配置文件。

1. 确认 flannel 网络是否可以初始化。

```
# flannel 网络已初始化
anxin@node38:~$ kubectl get pod -n kube-system | { head -1; grep flannel; }
NAME                    READY    STATUS    RESTARTS    AGE
kube-flannel-ds-7ctrb      1/1     Running   0          22d
kube-flannel-ds-grvbw      1/1     Running   0          22d
kube-flannel-ds-jxwxf      1/1     Running   0          22d
```

2. 若 flannel 网络未初始化，可能缺少了配置文件`/etc/cni/net.d/*`。

```
# 查看 节点 `/etc/cni/net.d `
$ ls /etc/cni/net.d
# 如果没有该路径,手动创建目录
$ mkdir -p /etc/cni/net.d
# 拷贝配置文件到节点主机
$ scp test-master:/etc/cni/net.d/* /etc/cni/net.d/
```

查看配置文件及内容：

```
# 查看配置文件
anxin@node38:~$ ls -l /etc/cni/net.d
total 4
-rw-r--r-- 1 root root 292 Dec 15 11:16 10-flannel.conflist # conflist (config list)

# 配置信息
anxin@node38:~$ vi /etc/cni/net.d/10-flannel.conflist

{
  "name": "cbr0",
  "cniVersion": "0.3.1",
  "plugins": [
    {
      "type": "flannel",
      "delegate": {
        "hairpinMode": true,
        "isDefaultGateway": true
      }
    },
    {
      "type": "portmap",
      "capabilities": {
        "portMappings": true
      }
    }
  ]
}
```

## kubelet 问题

```
# swap 没有关掉
$ journalctl -u kubelet -f
server.go:266] failed to run Kubelet: Running with swap on is not supported, please disable swap! or set --fail-swap-on flag to false. /proc/swaps contained: [Filename
```

PS：Ubuntu16 无法永久关闭 `swap`，经常开机后 `swap` 还是有可能自动挂载。

```
# 关闭 swap
$ swapoff -a
```

查看日志：

```
$ journalctl -u kubelet -f
Error: "MountVolume.SetUp failed for volume \"flannel-token-4g7bs\"
(UniqueName: \"kubernetes.io/secret/96cbc74a-58d5-4d10-b7df-52732ba63938-
flannel-token-4g7bs\") pod \"kube-flannel-ds-amd64-j7bl2\" (UID: \"96cbc74a-
58d5-4d10-b7df-52732ba63938\") : failed to sync secret cache: timed out
waiting for the condition"
```

解决：

```
# 编辑 `/var/lib/kubelet/config.yaml`，追加
featureGates:
  CSIMigration: false
# 然后重启 `kubelet`
$ service kubelet restart
```

## pod 被调度但工作异常

pod 被调度到节点后无法启动，或无法正常通信：**flannel 网络有问题。**

### CNI：`cni0`网桥地址设置失败

查看日志或 pod 详情（`kubectl describe`）：

```
NetworkPlugin cni failed to set up pod "demo-deployment-675b5f9477-
hdcwg_default" network: failed to set bridge addr: "cni0" already has an IP
address different from 10.0.2.1/24
```

重置节点：

```
$ kubeadm reset
$ systemctl stop kubelet && rm -rf /var/lib/cni/ && rm -rf /var/lib/kubelet/*
&& rm -rf /var/lib/etcd && rm -rf /etc/cni/
$ ifconfig cni0 down && ifconfig flannel.1 down && ifconfig docker0 down
$ ip link delete cni0 && ip link delete flannel.1
$ systemctl restart docker
$ systemctl start kubelet
```

重新生成`token`，并注册节点(`kubeadm join`)即可：

```
# 重新获取 token
$ kubeadm token create --print-join-command [--ttl=0]
```

### ENOENT：`subnet.env`

```
"open /run/flannel/subnet.env: no such file or directory"
```

解决方案：查看其他节点，并从其他节点的文件创建一个`/run/flannel/subnet.env`。

```
anxin@node38:~$ cat /run/flannel/subnet.env
FLANNEL_NETWORK=10.244.0.0/16
FLANNEL_SUBNET=10.244.0.1/24
FLANNEL_MTU=1450
FLANNEL_IPMASQ=true
```

## pod 内无法访问外网

检查 CoreDNS 组件的配置：coredns 的 ConfigMap（`kubectl edit cm coredns -n kube-system`）。

注意`coredns`镜像各版本的差异配置。

```
# 下面的 ConfigMap 对应的 image 为: coredns/coredns:1.7.0
apiVersion: v1
kind: ConfigMap
metadata:
  name: coredns
  namespace: kube-system
data:
  Corefile: |
    .:53 {
        errors
        health {
          lameduck 5s
        }
        hosts {
          10.8.30.157 test-master
          10.8.30.152 test-n1
          10.8.30.156 test-n2
          10.8.30.155 test-n3
          10.8.30.161 test-n4
          10.8.30.158 test-n5
          10.8.30.35 node35
          10.8.30.36 node36
          10.8.30.37 node37
          10.8.30.38 node38
          10.8.30.39 node39
          fallthrough
        }
        ready
        kubernetes CLUSTER_DOMAIN REVERSE_CIDRS {
          pods insecure
          fallthrough in-addr.arpa ip6.arpa
          ttl 30
        }
        prometheus :9153
        forward . /etc/resolv.conf
        cache 30
        loop
        reload
        loadbalance
    }
```

# pod 其他常见问题

## pod 一直处于`Pending`状态

表示该 pod 没有被调度到这个节点上，通常是由于**资源不足**造成的。解决方案，就是增加资源：

1. 增加工作节点。
2. 移除部分 pod 来释放资源。
3. 调低当前 pod 的资源限制。

## pod 一直处于`Waiting`状态

pod 卡在`waiting`说明 这个 pod 已经调度到这个节点，但是没有运行起来，可能是由于**镜像拉取失败**造成的。解决方案：

1. 检查网络。
2. 考虑镜像加速。
3. 使用 `docker pull <image>` 来验证镜像是否正常拉取。

## pod 一直处于`CrashLoopBackOff`状态

这个说明容器曾经启动过，它的重启次数是大于`0`的，无法通过健康检查。解决方案：

1. 一个是**镜像制作**的问题，镜像内 app 无法正常运行，检查 app 和镜像制作过程。
2. 重新设置合适的**健康检查阈值**。
3. 优化容器的性能，提高启动速度。
4. 关闭健康检查。

# connect: no route to host

查看日志得到：无法连接到服务器，原因为`no route to host`(没有到主机的路由，主机没有路由)。

```
github.com/coredns/coredns/plugin/kubernetes/controller.go:322: Failed to
list *v1.Namespace: Get https://10.96.0.1:443/api/v1/namespaces?
limit=500&resourceVersion=0: dial tcp 10.96.0.1:443: connect: no route to
host
```

该问题很有可能是防火墙（iptables）规则错乱或者缓存导致的，可以依次执行以下命令进行解决：

```
$ systemctl stop kubelet
$ systemctl stop docker
$ iptables --flush
$ iptables -tnat --flush
$ systemctl start kubelet
$ systemctl start docker
```

# K8s 健康检查

k8s 组件主要分为`Master组件`和`节点组件`：

- `Master组件`对集群做出全局性决策（比如调度），以及检测和响应集群事件。

如果`Master组件`出现问题，可能会导致集群不可访问，Kubernetes API 访问出错，各种`控制器`无法工作等等。

- `节点组件`在每个节点上运行，维护运行的 Pod 并提供 Kubernetes 运行时环境。

  如果`节点组件`出现问题，可能会导致该节点异常并且该节点 Pod 无法正常运行和结束。

```
anxin@node38:~$ kubectl get componentstatus # 或 kubectl get cs
NAME                 STATUS    MESSAGE             ERROR
scheduler            Healthy   ok
controller-manager   Healthy   ok
etcd-0               Healthy   {"health":"true"}

# 检查 kubelet、docker 等服务是否在运行 (active)
$ systemctl status kubelet docker

# k8s 组件
$ kubectl get pods -o wide -n kube-system
```

## kube-apiserver

对外暴露了 Kubernetes API，如果 kube-apiserver 出现异常可能会导致：

- 集群无法访问，无法注册新的节点。
- 资源（Deployment、Service 等）无法创建、更新和删除。
- 现有的不依赖 Kubernetes API 的 pods 和 services 可以继续正常工作（比如，Web 网站可以正常访问）。

## etcd

etcd 用于 Kubernetes 的后端存储，所有的集群数据都存在这里。保持稳定的 etcd 集群对于 Kubernetes 集群的稳定性至关重要。

因此，我们需要在`专用计算机`或`隔离环境`上运行 **etcd 集群** 以确保资源需求。

```
# etcd 单节点模式
anxin@node38:~$ kubectl get po -n kube-system | { head -1; grep etcd; }
NAME              READY    STATUS    RESTARTS   AGE
etcd-node38         1/1     Running   0         22d

# etcd 集群模式
$ kubectl get po -n kube-system | { head -1; grep etcd; }
NAME              READY    STATUS    RESTARTS   AGE
etcd-anxinyun-m1     1/1     Running   14966      493d
etcd-anxinyun-m2     1/1     Running   71         493d
etcd-anxinyun-m3     1/1     Running   73         493d
```

当 etcd 出现异常时可能会导致：

- kube-apiserver 无法读写集群状态，apiserver 无法启动。
- Kubernetes API 访问出错。
- kubectl 操作异常。
- kubelet 无法访问 apiserver，仅能继续运行已有的 Pod。

## kube-controller-manager 和 kube-scheduler

kube-controller-manager 和 kube-scheduler 分别用于控制器管理和 Pod 的调度，如果他们出现问题，则可能导致：

- 相关控制器无法工作。
- 资源（Deployment、Service 等）无法正常工作。
- 无法注册新的节点。
- Pod 无法调度，一直处于`Pending`状态。

## kubelet

kubelet 是主要的节点代理，如果节点宕机（VM关机）或者 kubelet 出现异常（比如无法启动），那么可能会导致：

- 该节点上的 Pod 无法正常运行，如果节点关机，则当前节点上所有 Pod 都将停止运行。
- 已运行的 Pod 无法**伸缩**，也无法正常终止。
- 无法启动新的 Pod。
- 节点会标识为`不健康`状态。
- 副本控制器会在其它的节点上启动新的 Pod。
- Kubelet 有可能会删掉当前运行的 Pod。

## CoreDNS

CoreDNS（在 1.11 以及以上版本的 Kubernetes 中，CoreDNS 是默认的 DNS 服务器）是 k8s 集群默认的 DNS 服务器，如果其出现问题则可能导致：

- 无法注册新的节点。
- 集群网络出现问题。
- Pod 无法解析域名。

## kube-proxy

kube-proxy 是 Kubernetes 在每个节点上运行的`网络代理`。如果它出现了异常，则可能导致：

- 该节点 Pod 通信异常。

# Init 容器排错

## Pod 应用

创建一个包含`应用容器`和`Init容器`的 Pod。Init 容器在`应用容器`启动前运行完成。下面是 Pod 的配置文件：注意 Init 容器在 nginx 服务器的根目录写入`index.html`。

```
anxin@node38:~/pengling/k8s$ vi init-containers.yaml

apiVersion: v1
kind: Pod
metadata:
  name: init-demo
spec:
  containers:
  - name: nginx
    image: nginx
```

```
    ports:
    - containerPort: 80
    volumeMounts:
    - name: workdir
      mountPath: /usr/share/nginx/html
  # These containers are run during pod initialization
  initContainers:
  - name: install
    image: busybox
    command:
    - wget
    - "-O"
    - "/work-dir/index.html"
    - http://info.cern.ch
    volumeMounts:
    - name: workdir
      mountPath: "/work-dir"
  dnsPolicy: Default
  volumes:
  - name: workdir
    emptyDir: {}
```

配置文件中，应用容器和 Init 容器共享了一个卷。Init 容器将共享卷挂载到了 `/work-dir` 目录，应用容器将共享卷挂载到了 `/usr/share/nginx/html` 目录。

Init 容器执行完`wget -O /work-dir/index.html http://info.cern.ch`命令就终止。

## 创建 Pod

```
anxin@node38:~/pengling/k8s$ kubectl create -f init-containers.yaml
pod/init-demo created
```

## 检查 Pod 状态

检查 nginx 容器运行状态：

```
anxin@node38:~$ kubectl get pod init-demo
NAME        READY   STATUS    RESTARTS   AGE
init-demo   0/1     Init:0/1   0          7s
# nginx 容器运行正常
anxin@node38:~$ kubectl get po init-demo
NAME        READY   STATUS    RESTARTS   AGE
init-demo   1/1     Running   0          2m55s
```

## 进入 nginx 容器

通过 shell 进入 `init-demo` Pod 中的 nginx 容器。

在 shell 中，发送 GET 请求到 nginx 服务器：结果表明 nginx 正在为 Init 容器编写的 web 页面服务。

```
anxin@node38:~$ kubectl exec -it init-demo -- /bin/bash # 进入 nginx 容器
root@init-demo:/#
root@init-demo:/# apt-get update
Get:1 http://deb.debian.org/debian bullseye InRelease [116 kB]
Get:2 http://deb.debian.org/debian bullseye-updates InRelease [39.4 kB]
```

```
Get:3 http://security.debian.org/debian-security bullseye-security InRelease [44.1 kB]
Get:4 http://security.debian.org/debian-security bullseye-security/main amd64 Packages [102 kB]
Get:5 http://deb.debian.org/debian bullseye/main amd64 Packages [8183 kB]

Get:6 http://deb.debian.org/debian bullseye-updates/main amd64 Packages [2592 B]
Fetched 8487 kB in 5min 41s (24.9 kB/s)

Reading package lists... Done
root@init-demo:/#
root@init-demo:/# apt-get install curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
curl is already the newest version (7.74.0-1.3+deb11u1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@init-demo:/#
root@init-demo:/# curl localhost # nginx 正在为 Init 容器编写的 web 页面 (http://info.cern.ch) 服务
<html><head></head><body><header>
<title>http://info.cern.ch</title>
</header>

<h1>http://info.cern.ch - home of the first website</h1>
<p>From here you can:</p>
<ul>
<li><a href="http://info.cern.ch/hypertext/WWW/TheProject.html">Browse the first website</a></li>
<li><a href="http://line-mode.cern.ch/www/hypertext/WWW/TheProject.html">Browse the first website using the line-mode browser simulator</a></li>
<li><a href="http://home.web.cern.ch/topics/birth-web">Learn about the birth of the web</a></li>
<li><a href="http://home.web.cern.ch/about">Learn about CERN, the physics laboratory where the web was born</a></li>
</ul>
</body></html>
```

## 附：MicroK8s 环境问题分析

MicroK8s 上部署，Init 容器状态异常。

```
# microk8s 上部署
dragon@test-pc:~/julin/init-container$ kubectl create -f init-demo.yaml
pod/init-demo created
```

### 1. 检查 Init 容器的状态

```
dragon@test-pc:~$ kubectl get po init-demo
NAME        READY   STATUS                  RESTARTS       AGE
init-demo   0/1     Init:CrashLoopBackOff   211 (98s ago)   17h

dragon@test-pc:~$ kubectl get po init-demo
NAME        READY   STATUS       RESTARTS        AGE
init-demo   0/1     Init:Error   211 (5m16s ago)   17h
```

## 2. 获取 Init 容器详情

使用 `kubectl describe pod <pod-name>` 查看 Init 容器运行的更多详情：

```
dragon@test-pc:~$ kubectl describe po init-demo
Name:       init-demo
Namespace:    default
...
Init Containers: # Init 容器
  install: # init-container name
    Container ID:
containerd://e2394af616ea2754deadf08d62cb8469bbac721c5d8e7704782fd92def044115
    Image:      busybox
    ...
    Command:
      wget
      -O
      /work-dir/index.html
      http://info.cern.ch
    State:        Waiting # 当前状态
      Reason:     CrashLoopBackOff
    Last State:     Terminated # 上一次状态
      Reason:     Error
      Exit Code:    1 # 退出码 { 0: 正常退出, 1: 异常退出 }
      Started:      Tue, 11 Jan 2022 06:55:18 +0000
      Finished:     Tue, 11 Jan 2022 06:55:18 +0000
    Ready:        False # 准备状态
    Restart Count:  276 # 重启次数
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-b9gl6 (ro)
      /work-dir from workdir (rw)
Containers: # 应用容器
  nginx: # app-container name
    Container ID:
    Image:        nginx
    Image ID:
    Port:         80/TCP
    Host Port:    0/TCP
    State:        Waiting
      Reason:     PodInitializing # 等待 Pod 初始化完成
    Ready:        False
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /usr/share/nginx/html from workdir (rw)
```

```
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-b9gl6 (ro)
...
Events:
  Type     Reason   Age                From     Message
  ----     ------   ----               ----     -------
  Warning  BackOff  81s (x6368 over 23h)  kubelet  Back-off restarting failed container
```

### 3. 通过 Init 容器访问日志

```
dragon@test-pc:~/julin/init-container$ kubectl logs init-demo
Error from server (BadRequest): container "nginx" in pod "init-demo" is waiting to start: PodInitializing

# 查看 Init 容器日志
dragon@test-pc:~/julin/init-container$ kubectl logs init-demo -c install
wget: bad address 'info.cern.ch'
```