

cpu.cfs_period_us ⊙ 设置时间周期长度)(每500ms能使用1000ms的CPU时间,即使用两个核 # echo # echo 时间周期长度内能 (每50ms能使用10ms的CPU时间,即使用cpu.cfs_quota_us ⊙ 使用的CPU总量 memory.limit_in_bytes ⊙ 设置进程可用内存空间 <u>0: 直接kill</u>
<u>0: 直接kill</u>
<u>0: 直接kill</u>
<u>1: 暂停执行,直到有足够内存</u> memory.oom_control under_oom 💿 显示当前进程是否达到最大内存限制 rootfs仅包含操作系统的"壳",并不包含内核,就如同Shell与Linux内核的关系一样 这就好比千层饼,你看到的是一张饼,但是咬开之后会发现有很多层面饼 device mapper等方法将不同的文件组合成一个 用户实际对容器进行创建、修改和删除的地方,每次操作均会增量的在该层产生 当用户删除了一个文件时,实际的文件并不会从镜像中移除,而是将其加入`whiteout`,使其逻辑不可 读写层 👓 见,但物理存在,所以删除文件并不会使得镜像大小减少 当用户修改只读层内容,即ubuntu的系统目录文件时,Docker不会在只读层进行修改,而是在读写层拷 贝一份内容,修改后增量的计入只读层,由此覆盖原有只读层的内容 Init层 ◎ 用来存放 /etc/hosts、/etc/resolv.conf 等信息,本应属于ubuntu只读层,但是可能会被频繁的修改,故Docker单独抽取出来 只读层 ◎ 所有的只读层以增量的方式构成操作系统的文件,配置以及相关目录 在 /proc/container_pid/ns 目录下,可以看到进程id为container_pid的所有 Namespace链接文件,所有文件均链接至真实的Namespace文件 将容器所创建的所有Namespace文件均使用setns()调用并加入,即 在有了Namespace文件之后,就可以使用setns()系统调用,将当前 可获得与容器完全相同的命名空间进程,此时执行的命令就相当于 在容器中执行 目前Docker使用cgroupfs作为Cgroup driver,但是k8s官网却建议使用systemd作为Cgroup driver #docker 💿 本身k8s使用CNI将容器运行时抽离出来,作为热拔插组件,但是目前来看,Docker仍是较为优秀 的容器运行时实现,而且k8s的Cgroup driver改起来也不复杂,故均使用cgroupfs作为driver 系统设置 o 若使用树莓派等内存本身较小的机器安装k8s集群时,可先临时关闭交换分区,而后对kubeadm o vim /etc/system/kubelet.service.d/10-kubeadm.conf Environment="KUBELET_EXTRA_ARGS=--fail-swap-on=false" 镜像如果没有梯子的话,可以先从阿里镜像源中拉取,而后tag成所需要的镜像名称 docker pull registry.alivuncs.com/google_containers/ImageName 执行kubeadm init初始化master节点,期间会检查当前节点的相关配置并拉取必须的镜像 💿 docker tag registry.aliyuncs.com/google_containers/ImageName k8s.gcr.io/ImageName vim /etc/systemd/system/kubelet.service.d/10-kubeadm 修改k8sCgroup driver为cgroupfd Environment="KUBELET_CGROUP_ARGS=--cgroup-driver=cgroupfs" stry.aliyuncs.com/google_containers/\$imageName try aliyuncs.com/google_containers/\$imageName_k8s.gcr.io/\$im CNI(Container Netwaork Interface) 插件的安装 🧧 weave 💿 kubectI apply -f "https://cloud.weave.works/k8s/net?k8s-version=\$(kubectI version | base64 | tr -d '\nr)" kubectl apply -f https://raw.githubusercontent.com/rook/rook/master/cluster/examples/kubernetes/ceph/common.yaml CSI(Container Storage Interface)插件的安装 💿 rook o kubectl apply -f https://raw.githubusercontent.com/rook/rook/master/cluster/examples/kubernetes/ceph/operator.yaml need VPN kubectl apply -f https://raw.githubusercontent.com/rook/rook/master/cluster/examples/kubernetes/ceph/cluster.yaml kubernetes-dashboard 💿 kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-rc3/aio/deploy/recommended.yaml yaml: line 3: mapping values are not allowed in this context O \$HOME/.kube/config未正确设置 APIServer组件运行不正常,通常因为kul Jnable to connect to the server: dial tcp 192.168.1.109:6443: connect: no route to host 包括v1alpha1,v1alpha2,v1beta1等诸多版本,官网对其并无详细说明,所以在编写init YAML文件之前,首先要确定安装的kubeadm所支持的apiVersion 关于kubeadm init --config my_config.yaml中自定义YAML文件格式 。 apiVersion 。 使用 `kubeadm config print init-defaults`获取默认配置文件进行确定 - YAML文件具体配置 ◎ 配置名称项会跟随apiVersion而改变,所以此时需借助GoDoc的协助 应第一时间使用 journalctl -xe 查看启动日志 kubeadm init会使用一个可用的网卡IP地址作为集群地址,Linux与 Windows的系统切换会导致Linux IP发生变更 etplan/01-network-manager-all.yam 修改Ubuntu IP地址 dhcp4: false gateway4: 192.168.1. 8.8.8.8 114.114.114.114 sudo netplan apply 使配置生效 通常是Docker安装时的问题 etcd容器启动失败,失败原因: "error while creating mount source path '/var/lib/etcd': mkdir /var/lib/etcd: read-only file system" 或者使用sudo权限重启Docker以及kubelet