Grep –Ei ‘vmx|svm’ /proc/cpuinfo 输出vmx表示intel的处理器，svm表示amd

# 第一天

Namespace cgroup 资源隔离，保证某个服务不会占用太多系统资源

## 容器跟虚拟机的区别

截图

虚拟机底下是物理机，有操作系统跟内核，中间经过hyperviser层，去跑一些虚拟机。虚拟机的操作系统可以跟底层的操作系统不一样。 缺点。： 有底层host os跟guest os，应用程序需要先进入guest os 的内核，guest 通过hypervisor 访问host os的内核运行，有一层虚拟化的损失。虚拟机可以安装不同的操作系统，更灵活

容器比较轻便，只有一个host os，容器跟host公用一个内核，以一个进程的方式运行在host os上。 通过docker engine，公用同一个内核。所以docker必须是同一种操作系统。性能强，启动快。轻量级高性能。

Docker通过namespace cgroup 实现了资源隔离

Docker 统一了应用的封发格式，打包了程序的运行环境，解决了程序的依赖问题。

Docker是容器中最流行的一种实现，其他还有rkt,warden,pouch(阿里的，胖容器，一个容器里面运行多个应用程序，其他的容器一般运行一个程序)

## 容器很重要，现在不再重要

Opencontainer 组织 -cloud native 组织 （云原生应用）

单机-》多机

小规模，简单应用-》大规模，复杂应用

## Borg

Google的程序运行管理框架系统

略

## Kubernetes入门

1. 用户通过kubectl提交需要运行的docker container（pod）
2. Apiserver 把请求存储在etcd里面
3. Scheduler扫描，分配机器
4. Kubelet找到自己需要跑的container，在本机上运行
5. 用户提交replicaset描述，replication controller 监视集群中的容器数量并保持数量
6. 用户提交service描述文件，由kube proxy负责具体的工作流转发

### Kubectl 特性

Docker 容器构建方式

使用容器的标准生命周期

支持挂载外部的各种持久层来维持状态

从容器中读取监控信息，从多个层面检查应用性能

支持在pod中包含日志处理容器

开源

使用docker容器，或者任何实现cri标准的容器实现

Go语言编写

目前还没有做很多性能优化

目前但集群支持上千节点这个数量级

可以对接各种I层，如aws azure openstack需要自己找文档

## 单体vs分布式

|  |  |  |
| --- | --- | --- |
|  | **传统单体架构** | **分布式服务化架构** |
| 新功能开发 | 需要时间 | 容易开发和实现 |
| 部署 | 不经常且容易部署 | 经常发布，部署复杂 |
| 隔离性 | 故障影响范围大 | 故障影响范围小 |
| 架构设计 | 难度小 | 难度技术增加 |
| 系统性能 | 响应时间快，吞吐量小 | 响应时间慢，吞吐量达 |
| 系统运维 | 运维简单 | 运维复杂 |
| 新人上手 | 学习曲线大（应用逻辑） | 学习曲线大（架构逻辑 |
| 技术 | 技术单一且封闭 | 技术多样且开放 |
| 测试和查错 | 简单 | 复杂 |
| 系统扩展性 | 扩展性很差 | 扩展性很好 |
| 系统管理 | 重点在于开发成本 | 重点在于服务治理和调度 |

### 云原生应用的里面

云原生应用，cloud native 应用在设计期间就是分布式的

1. Containerized . each part (applications,processes,etc) is packaged in its own container. This facilitates reproducibility transparency and resource isolation.
2. Dynamically orchestrated. Containers are actively scheduled and managed to optimize resource utilization.
3. Microservice oriented.application are segmented into microservices. This significantly increases the overall agilty and maintainability of application

## 微服务

把整个应用拆解成多个服务，各个服务解耦

可用性，扩展性

对用户更友好，对开发、运维复杂

## Paas

截图

Devops 用开发的方式去解决运维的问题

# 第二天

## 环境准备，安装minikube

新建虚拟机

**安装docker-ce**

<https://docs.docker.com/install/linux/docker-ce/centos/>

1. $ sudo yum install -y yum-utils device-mapper-persistent-data lvm2
2. $sudo yum-config-manager --add-repo <https://download.docker.com/linux/centos/docker-ce.repo>
3. $ sudo yum install docker-ce docker-ce-cli containerd.io

安装的时候碰到的问题

Error: Package: docker-ce-18.03.1.ce-1.el7.centos.x86\_64 (docker-ce-edge)

           Requires: container-selinux >= 2.9

 You could try using --skip-broken to work around the problem

 You could try running: rpm -Va --nofiles –nodigest

解决： wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

 yum install epel-release   #阿里云上的epel源

然后yum install container-selinux

**安装minicube**

<https://github.com/kubernetes/minikube/releases> 下载对应的版本

安装： curl -Lo minikube https://storage.googleapis.com/minikube/releases/v0.34.1/minikube-linux-amd64 && chmod +x minikube && sudo cp minikube /usr/local/bin/

**安装kubectl**

Ubuntu：<https://mirrors.ustc.edu.cn/kubernetes/apt/pool/> 可以从国内镜像直接下载

其他：<https://kubernetes.io/docs/tasks/tools/install-kubectl/> 有很多下载方法，我用的是下载二进制的方法

curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl

以上方法比较慢，可以先获取稳定版本的版本号，在替换到网址中在迅雷中下载：

curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.13.4/bin/linux/amd64/kubectl

chmod +x ./kubectl && sudo mv ./kubectl /usr/local/bin/kubectl

**启动minikube**

<https://github.com/kubernetes/minikube>

需要先安装kvm驱动(也可以不用装，指定--vm-driver=none)

<https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#kvm2-driver>

$sudo yum install libvirt-daemon-kvm qemu-kvm

$sudo systemctl enable libvirtd.service

$sudo systemctl start libvirtd.service

$sudo systemctl status libvirtd.service

$sudo usermod -a -G libvirt $(whoami) //把自己加入到libvirt group

$newgrp libvirt

安装驱动

$curl -LO https://storage.googleapis.com/minikube/releases/latest/docker-machine-driver-kvm2 \

&& sudo install docker-machine-driver-kvm2 /usr/local/bin/

NOTE: Ubuntu users on a release older than 18.04, or anyone experiencing [#3206: Error creating new host: dial tcp: missing address.](https://github.com/kubernetes/minikube/issues/3206) you will need to build your own driver until [#3689](https://github.com/kubernetes/minikube/issues/3689) is resolved. Building this binary will require [Go v1.11](https://golang.org/dl/) or newer to be installed.以下是安装方法：

$sudo apt install libvirt-dev

$test -d $HOME/go/src/k8s.io/minikube || \

git clone https://github.com/kubernetes/minikube.git $HOME/go/src/k8s.io/minikube

cd $HOME/go/src/k8s.io/minikube

git pull

make out/docker-machine-driver-kvm2

sudo install out/docker-machine-driver-kvm2 /usr/local/bin

至此结束

$minikube start --vm-driver --vm-driver kvm2 //系统自己执行初始化工作 默认使用virtualblx，使用--vm-driver 来指定hypervisor

to use kvm2 as a default driver:

$minikube config set vm-driver kvm2 （或者none）

$minikube start

日志： /var/lib/localkube/localkube.err 有一些镜像pull不下来。 解决方法： 因为pause跑不起来，我们可以从其他网站把kubernites/pause 下载下来。

#Docker pull kubernites/pause

#Docker tags {image ID} grc.io/google\_container/pause-amd:3.0

然后就可以正常的启动应用

## K8s 抽象