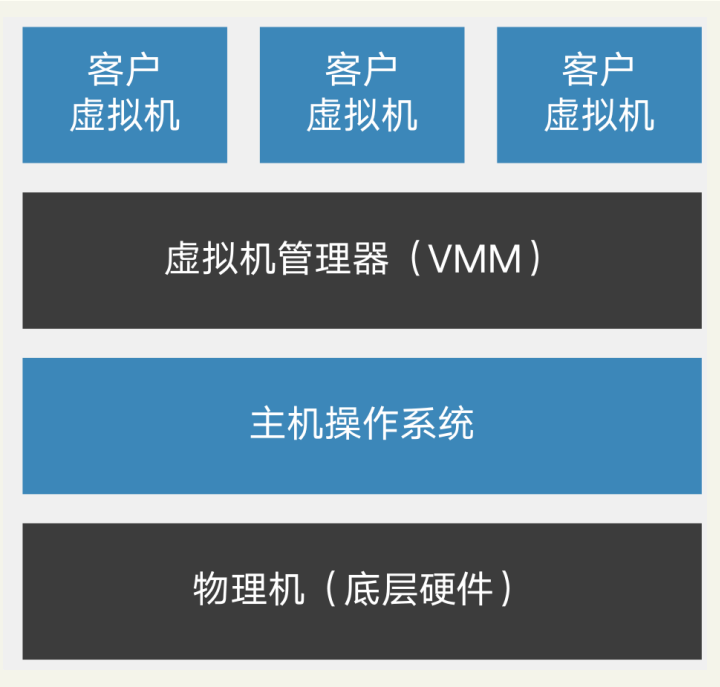
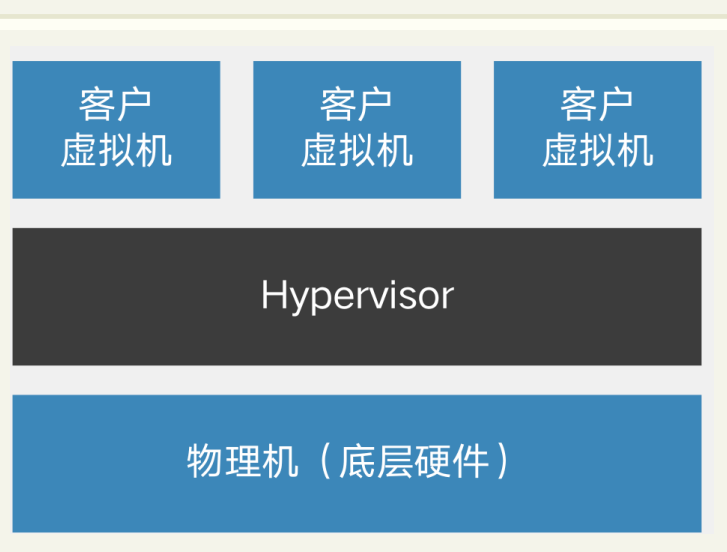
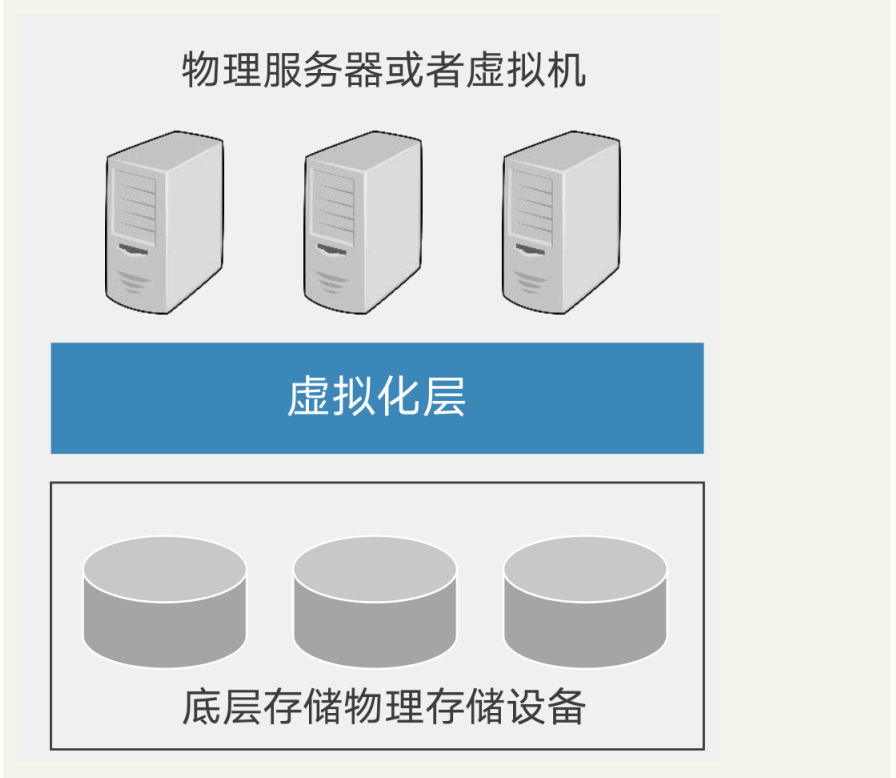
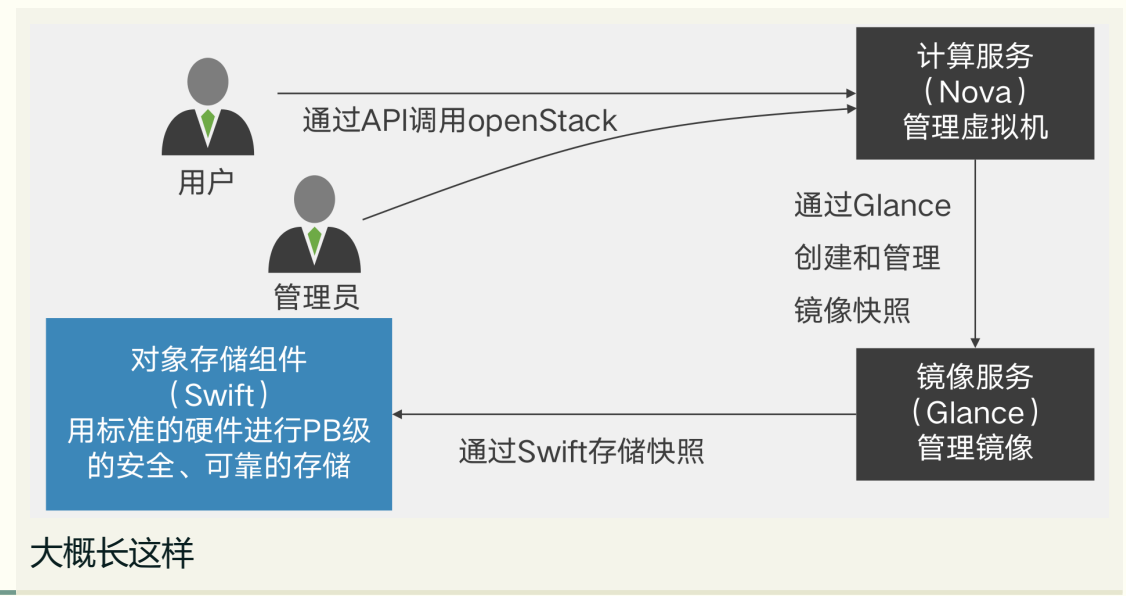
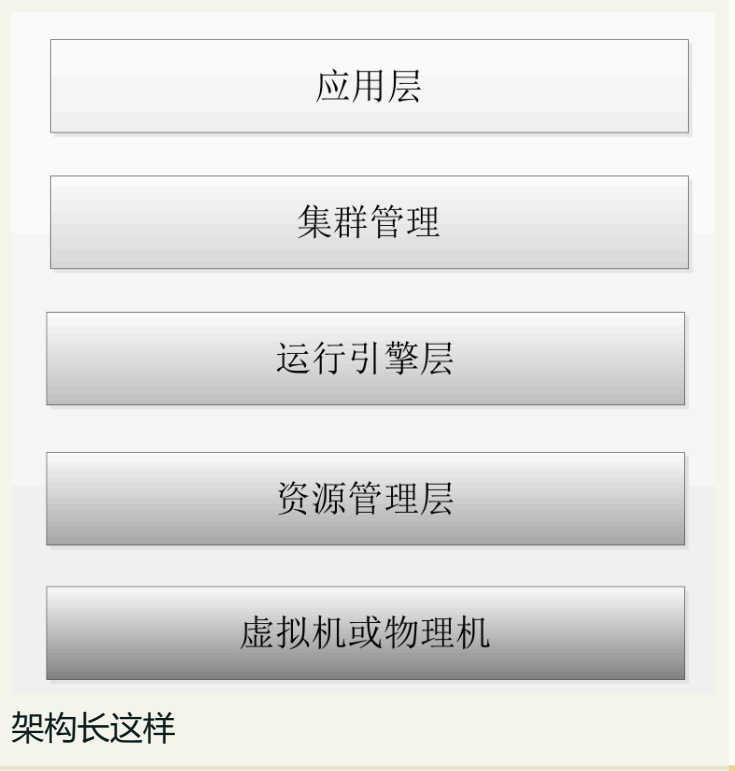
虚拟化技术

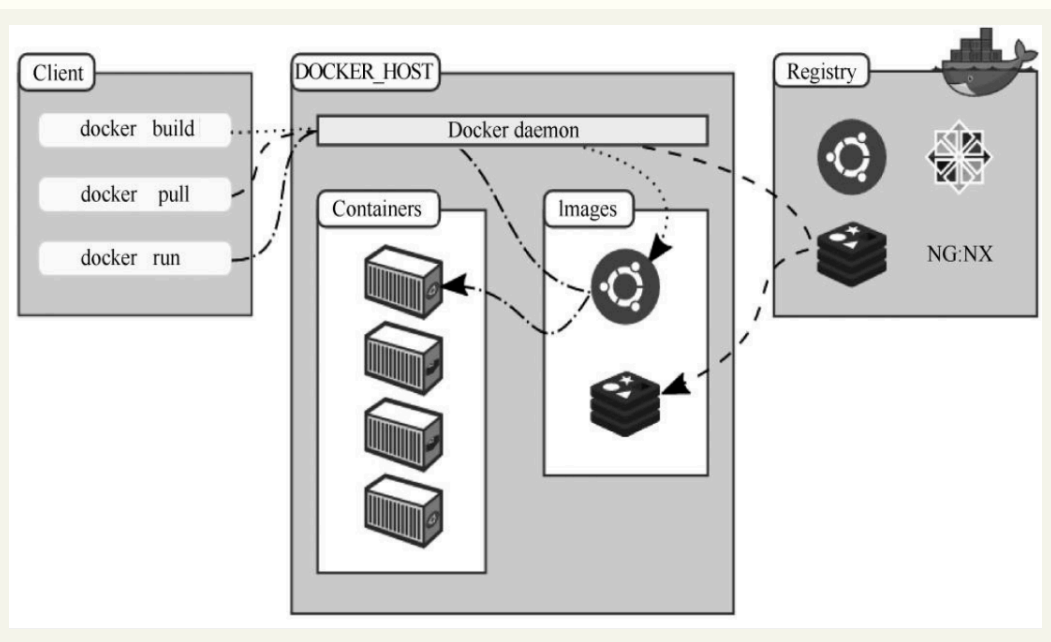
- 虚拟化
  - 常见虚拟化技术 理解 练习题
    - 服务器虚拟化
      - 将物理服务器虚拟成多个逻辑上的服务器
      - 层次/实现方式 练习题
        - 

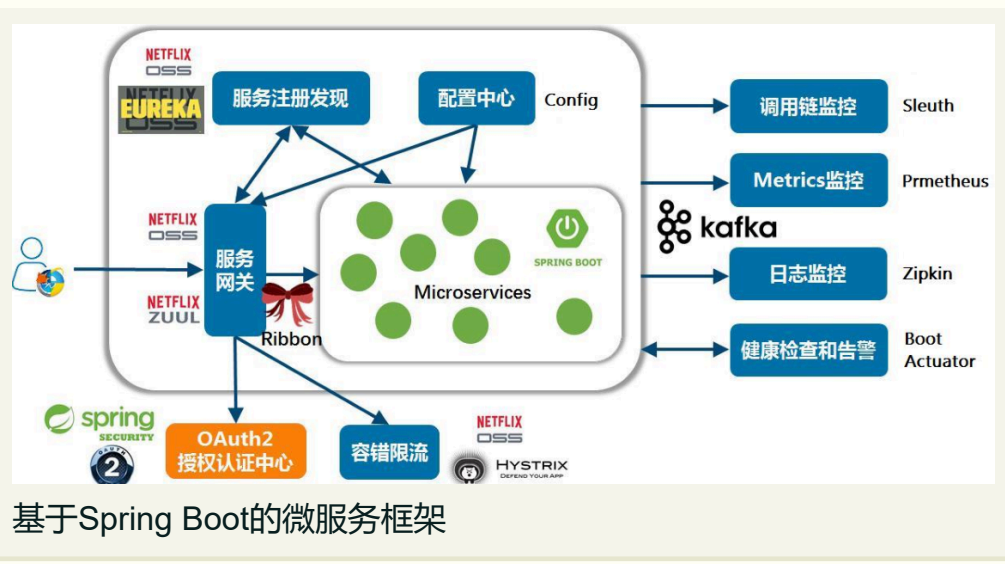
          寄居虚拟化
          - 虚拟机管理器与服务器操作系统位于两个层面，意味着VMM运行在主机操作系统上，虚拟机只能运行同一个操作系统
        - 

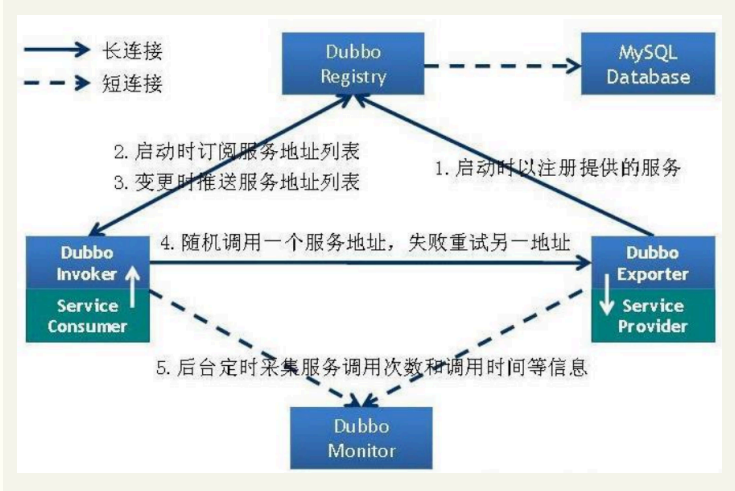
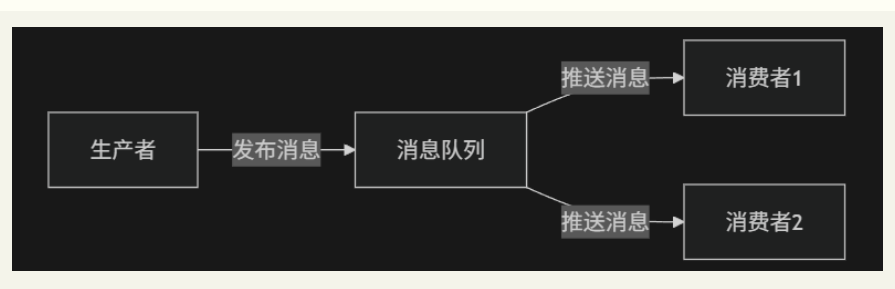
          裸机虚拟化
          - 将VMM与操作系统统一为了Hyperviisor，这种用的比较多，好处是有效隔离客户虚拟系统
    - 存储虚拟化
      - 把分布式的异构存储设备统一为一/多个大的存储池
      - 目的
        - 提高存储资源利用率
        - 降低提供、管理单位存储空间的成本
      - 模型
        - 

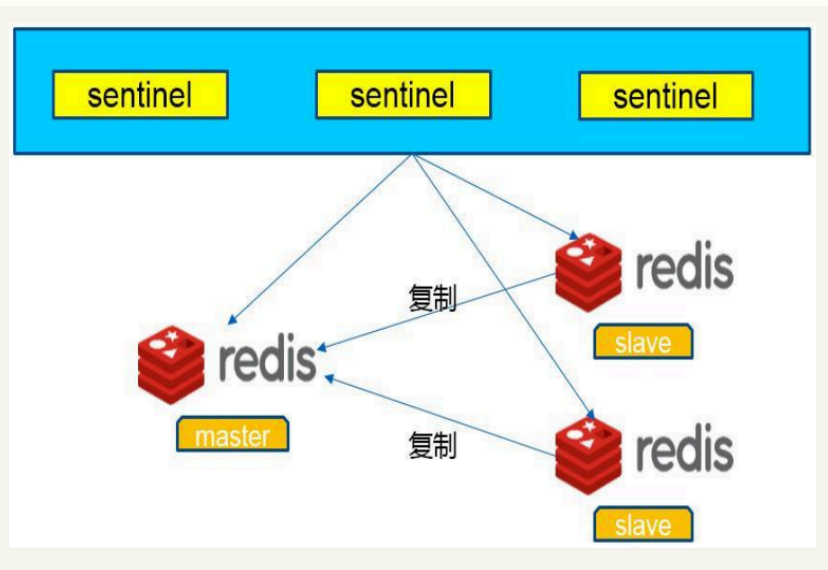
          在存储层与服务器层之间加一个虚拟化层，就是存储虚拟化
      - 实现方式 练习题
        - 基于主机
          - 也即在本机上通过对已有存储设备的操作进行虚拟化
          - 比如对电脑上的C盘进行拆分，实际上还是一个盘，只是用户看到两个盘
          - 性价比不错，但性能、可扩展性差
        - 基于存储设备
        - 基于网络
          - 基于网络设备，通过网络进行存储虚拟
          - 好处就是性能、效果、安全都不错，但一旦路由器等网络设施出故障就会导致数据无法访问
    - 网络虚拟化
      - 在底层网络与网络用户之间增加一个抽象层，对网络进行虚拟化操作
      - 虚拟化分层
        - 核心层
          - 具备超大规模数据交换能力
        - 接入层
          - 支持灵活部署与新以太网技术，即为网络走线的抽象层
        - 虚拟机
          - 使用网络的机器，即用户
      - 实现 练习题
        - VLAN网络分段
        - SDN
        - NFV
    - 桌面虚拟化
      - 基于中心服务器的计算机运作模型，叫桌面虚拟化的原因大概是可以在同一机器的桌面上同时使用多种不同的操作系统
      - 一代
        - 同一硬件平台上并行运行多个操作系统
      - 二代
        - 提升性能，降低管理成本
  - OpenStack 掌握
    - 一个数据中心云计算开放平台，提供管理云计算资源的能力
    - 结构
      - 

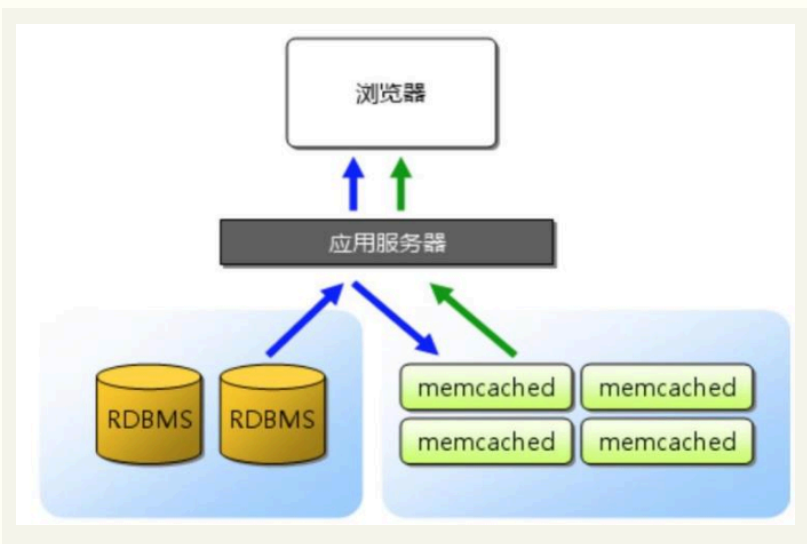
        大概长这样
    - 功能
      - 计算资源管理
        - 规划、管理大量虚拟机
        - 允许按需提供计算资源
      - 存储资源管理
        - 为云服务/应用提供所需的存储资源
      - 网络资源管理
        - 提供高扩展、高自动化管理网络的能力
    - 组件 练习题
      - Nova计算资源管理
      - Glance镜像管理
      - Cinder块存储服务
      - Neutron网络服务
      - Swift对象存储服务
      - Heat编排服务
  - 超融合技术 了解
    - 一种技术架构，通过软件定义实现计算、存储、网络融合，进而实现以虚拟化为中心的软件定义数据中心
    - 产生背景
      - 传统架构没有统一的集中化IT构建策略
      - 传统架构的存储系统难以升级
      - 传统架构中网络成为了传输的瓶颈
    - 优势 练习题
      - 硬件成本低
      - 部署快速
      - 资源利用率高
      - 运维、升级简便
- 容器
  - 架构 掌握
    - 

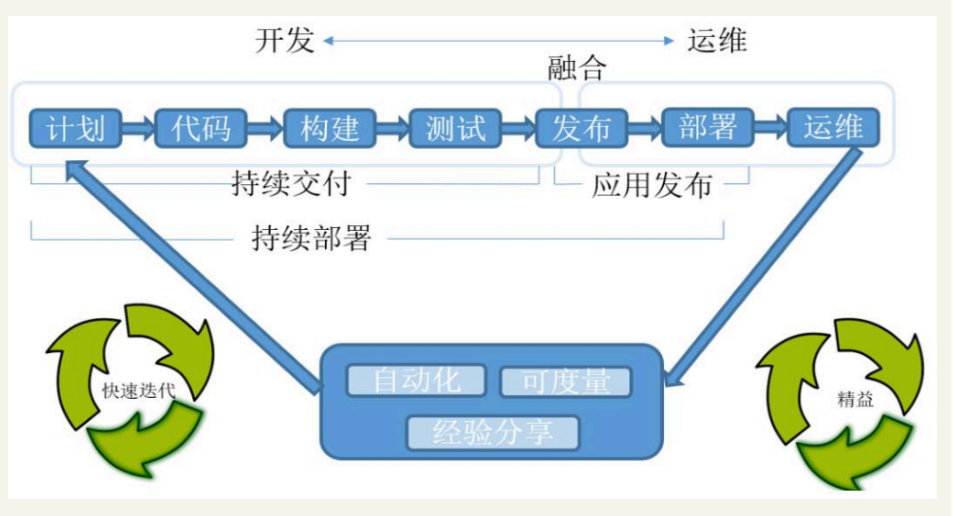
      架构长这样
      - 应用层就是运行在容器上的应用程序，比如MySQL
      - 集群管理就是管理容器集群的，比如K8S
      - 运行引擎是常见的容器系统，比如Docker
      - 资源管理就是对服务器、操作系统这些资源进行管理的组件，比如Linux Kernel
      - 虚拟机层就是容器运行的环境了，比如Linux
  - 技术组成 掌握
    - 底层技术 练习题
      - 命名空间
        - 每个容器都有自己的命名空间，互不影响
      - 控制组
        - 控制组能够对容器进程进行隔离、记录等操作
      - 切根
        - 切根的意思是改变程序运行时参考的根目录
        - 切根能够将容器的进程限定在指定目录里运行，从而实现基本文件系统隔离
    - 关键技术
      - 镜像
        - 包含操作系统、应用本身的文件
      - 引擎
      - 编排工具
        - 对容器进行编排，包括启动、部署、升级等操作
      - 集群
        - 将多物理设备抽象为单一实体，从而以集中化方式管理资源
      - 服务注册、发现
        - 容器之间的互相发现用到的技术
      - 热迁移
        - 将容器运行时状态完整复制，并能在其他主机上恢复运行
  - 对比虚拟机 掌握 练习题
    - 启动速度在秒级
    - 复杂度低
    - 性能与原生环境几乎一致
    - 简单可控
    - 体积小，通常MB
    - 并发性高，能同时启动几百至几千个容器
    - 资源利用率高
  - Docker 了解
    - 一个实现容器技术的产品
    - 基本概念
      - 镜像
      - 容器
      - 仓库
    - 架构 练习题
      - 

        基本长这样，很熟悉了
    - 组件
      - Docker daemon
        - 守护进程
        - 管理宿主机的容器、镜像
      - Docker client
        - 客户端
        - 与守护进程通信以进行相关操作
      - Docker registry
        - 镜像仓库
        - 存储、分发镜像的服务器
      - Docker images
        - 镜像
        - 容器的只读模板，包含运行环境，用以创建容器
      - Docker containers
        - 容器
        - 镜像运行后的实例，有独立的文件系统等环境
  - Rocket 了解
    - 一个实现容器引擎技术的产品
    - 执行模式
      - 直接运行
        - 容器与宿主机共享PID与命名空间
      - 系统容器
        - 容器运行在一个隔离开的系统服务中
        - 支持完整的进程管理，类似虚拟机
      - KVM
        - 容器在轻量虚拟机中运行
        - 安全性、隔离性都很高
  - 容器编排平台 了解 练习题
    - Apache Mesos
      - 特性
        - 节点数上限高
        - ZooKeeper实现容错
        - 使用Linux实现本地隔离
        - 可通过Web查看集群状态
      - 底层级调度器，支持多种框架，包括K8S与Swarm
    - Google K8S
      - 特性
        - 声明式设计
        - 简单化
        - 以服务为中心
      - 需要对应用重新设计以适配K8S，但支持失败冗余与系统扩展
    - Docker Swarm
      - 特性
        - 声明式服务模式
        - 去中心化
        - 集群管理、Docker集成
      - 使用简单、容易集成到现有系统、但是难以支持复杂调度
- 云原生
  - 微服务 掌握
    - 概念 练习题
      - 将大型复杂软件应用拆分成多个简单应用，每个简单应用描述着一个小业务，系统中的各个简单应用可被独立部署，这样的拆分中，每个简单应用是一个微服务
    - 架构 练习题
      - 侵入式架构
        - 即要求业务代码必须继承框架父类的深耦合架构
        - Spring Cloud
          - 

            基于Spring Boot的微服务框架
          - 简化分布式基础设施的开发
      - 非侵入式架构
        - 与侵入式架构相反
        - Dubbo
          - 
            长连接
短连接
Dubbo Registry
MySQL Database
2. 启动时订阅服务地址列表
3. 变更时推送服务地址列表
1. 启动时以注册提供的服务
4. 随机调用一个服务地址，失败重试另一地址
Dubbo Invoker Service Consumer
Dubbo Exporter Service Provider
5. 后台定时采集服务调用次数和调用时间等信息
Dubbo Monitor
            阿里巴巴公司的开源服务框架
          - 功能
            - 高性能RPC调用
            - 服务自动注册/发现
            - 高可扩展性
            - 负载均衡
            - 运行中流量调度
            - 可视化运维
        - Service Mesh服务网格
          - 服务间通信的基础设施层
          - 应用间通信的中间层
          - 提供服务间消息的可靠传递
  - 敏捷基础设施
    - 概念 掌握
      - 别名基础设施即代码/可编程基础设施，意味着敏捷基础设施可以将基础设施配置完全当作软件编程来操作
      - 目标
        - 标准化
        - 自动化
        - 可视化
        - 可追溯
        - 可替换
        - 快速
    - 消息中间件服务 掌握
      - 

        支持在分布式系统中发送和接收消息的硬件或软件基础设施，消息中间件就是一个系统，提供可靠的分布式系统间通信能力
      - 目的
        - 解决分布式系统之间的消息传递问题
      - 主流类型 练习题
        - Kafka
        - ActiveMQ
        - RabbitMQ
        - RocketMQ
      - 模型分类 练习题
        - 点对点模型
        - 发布/订阅模型
    - 分布式缓存服务 掌握
      - 分布式缓存服务将数据分散存储在多个节点的内存中，通过集群协同提供高性能读写、弹性扩展与高可用性的缓存服务
      - 目的
        - 解决数据库服务器和Web服务器之间的瓶颈
      - 主流类型
        - 

          Redis
        - 

          Memcached
  - DevOps 了解
    - 概念 练习题
      - 一组过程、方法与系统的统称，打破开发Dev与运维Ops的隔阂并减少变更范围、加强发布协调、自动化部署流程
    - 目的
      - 统一软件开发和软件维护操作
      - 缩短开发周期，增加部署频率，更可靠的发布
    - 常用工具
      - Ansible
      - SaltStack
      - Jenkins
      - Terraform
      - Packer
  - CI/CD流程 了解 练习题
    - 持续交付CD
      - 

        持续交付流程
      - 概念
        - 一种软件工程手段，旨在缩短软件开发周期，确保软件可以随时、可靠地发布
      - 技术
        - TeamCity
        - octopus
        - NCrunch
        - Azure
    - 第一步
    - 持续集成CI
      - 概念
        - 一种在开发周期的早期阶段进行集成的实践技术，为软件开发提供频繁地测试、合并操作环境
    - 第二步
    - 持续部署
      - 概念
        - 持续部署则是持续交付的下一步，代码通过评审，自动化部署到生产环境
  - 应用场景 了解
    - 深度学习
      - 以容器服务为核心，以云原生技术作为基础架构的深度学习，整合了云的计算、存储、负载均衡等服务
    - 区块链
      - 使用容器封装区块链节点，通过容器集群实现区块链网络的编排、创建、运维和资源管理
    - 边缘计算场景
    - 传统行业互联网化应用