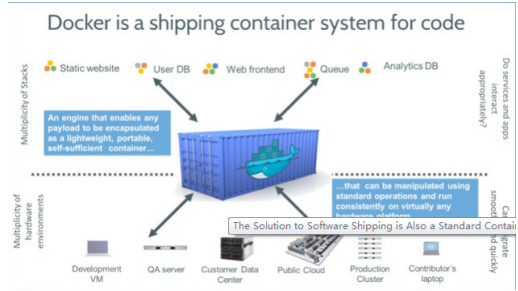
应用 *Docker* 相关技术实现容器即服务的探索与实践

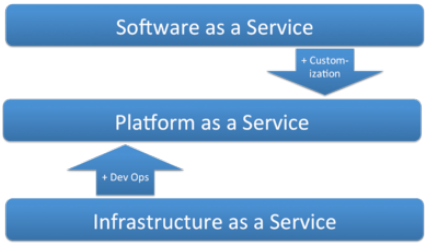
TenxCloud时速云团队 – 王磊

目录

DockOne.io
Community of Docker

Docker技术带给我们的遐想





- 基于容器提供类似IaaS的功能
  - 优势
    - ☑ 价格低，可以打价格战，性价比不错
    - ☑ 速度快，定制灵活，支持多租户共享资源
  - 缺点
    - ☐ 资源的隔离效果
    - ☐ 安全性
    - ☐ 用户需要时间对容器技术的理解和接受
- DevOps - 代码构建、简化配置、保证开发测试环境一致性、持续集成等
  - 比较合适
- 微服务，服务编排与集成
  - 擅长领域，但是需要一个较长的转变过程

镜像管理

Docker 镜像管理

- Registry server类型
  - private registry
  - mirror registry
  - sponsor / vendor registry
- 快速上手
  - docker run -p 5000:5000 -v <HOST_DIR>:/tmp/registry-dev registry:<2.0.1>
  - 配置文件参考 config/config_sample.yml 及 config_mirror.yml，也可以通过各种环境变量修改
  - 需要修改docker daemon

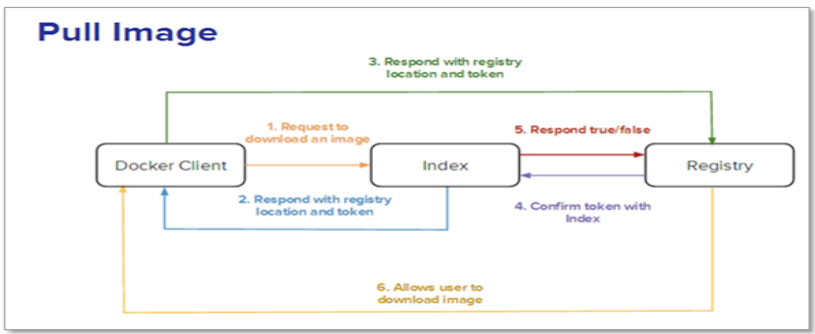
    --insecure-registry <REGISTRY_HOSTNAME>:5000

- 增加基本用户验证
  - 在registry server前面放一个nginx server

```
location / {
  auth_basic                "Restricted";
  auth_basic_user_file      docker-registry.htpasswd;
  include                   docker-registry.conf;
}
```

- Redis cache 和 MySQL search database, 默认使用 SQLAlchemy
- Registry 无状态，多个server可以共享存储

- 添加 index server
  - 权限和角色认证
  - 扩展服务支持 - Webhook等
  - 链接多个Registry server - *X-Docker-Endpoints*



- Registry v2
  - Go 语言替换原来的python，实现上更高效
  - 下载和上传速度更快一些 (多线程下载)
  - 简化部署过程
  - 内嵌Webhook 通知系统
  - 系统状态检查

持续集成

- GitHub等代码托管服务的集成
  - 通过GitHub、BitBucket的OAuth接口获得客户授权（token）
  - 通过token获取用户的项目列表，和CI系统集成

- 客户端支持
  - 无需关联代码托管服务
  - 如同使用本地 docker 一样的体验
  - 不需要打包源代码文件

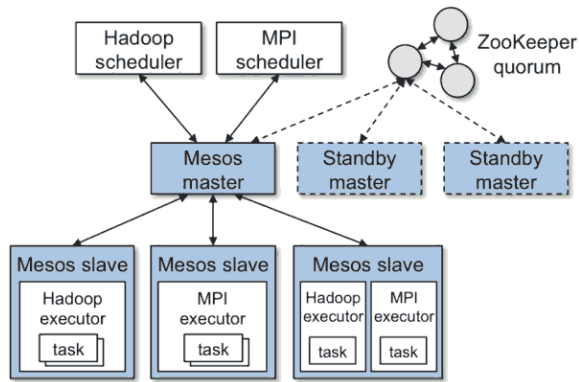
- 构建Docker 镜像 (Docker in Docker)
  - 构建过程不影响 host 主机的docker daemon
  - 镜像构建过程放到容器中，方便管理
- 添加代理优化
  - Proxy 路由，用于 “翻墙”
  - 缓存下载内容，提高构建速度

容器及集群管理

- 容器资源管理 - 需要LXC、存储等Linux底层支持
  - CPU
    - cpu-shares=1024
    - cpuset.cpus=1,2-4
  - Memory
    - memory
    - memory-swap
  - lxc-conf
  - Storage
    - aufs
    - Device-mapper
    - overlay
    - btrfs

容器集群管理、调度 - Marathon + Mesos (Apache)

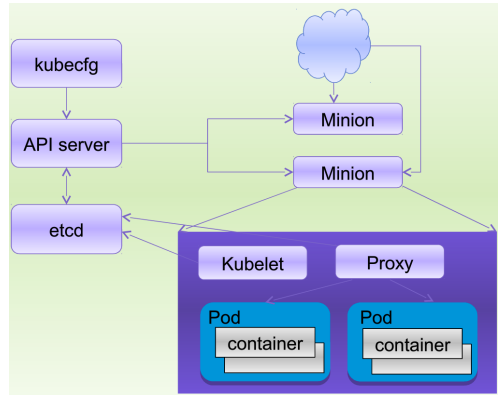
- Marathon是Mesos 的一个调度和服务管理组件。它配合mesos来控制长时间运行的服务，并为进程和容器管理提供Web界面。
- Mesos - 管理资源
  - 分布式应用的资源隔离和共享
- Marathon
  - 集群范围的进程监管
  - 服务发现、负载均衡
  - 应用部署和伸缩



容器集群管理、调度 - Kubernetes （Google）

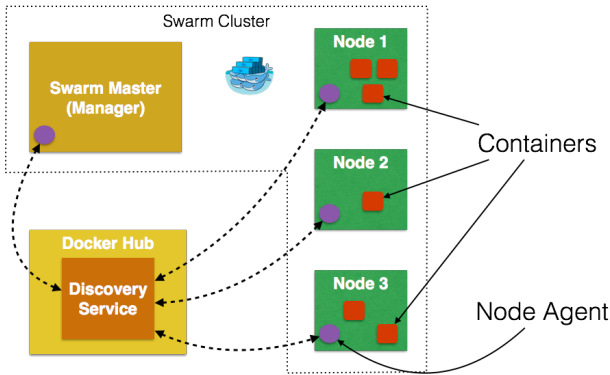
- Kubernetes：高级调度器，kubernetes提供对容器更多的控制。容器可以被打标签、分组和配置通信子网。

- 以集群的方式运行
- 解决Docker 跨主机容器之间的通讯问题
- 自我修复机制使得容器集群总是运行在用户期望的状态
- 良好的扩展性，支持多种容器实现
- 丰富的定制化参数



容器集群管理、调度 - Swarm (Docker)

- 采用Docker原生句法使得可以在宿主机上启动容器和进行供应
- 为调度和编排框架提供通用接口



主机管理

DockOne.io
Community of Docker

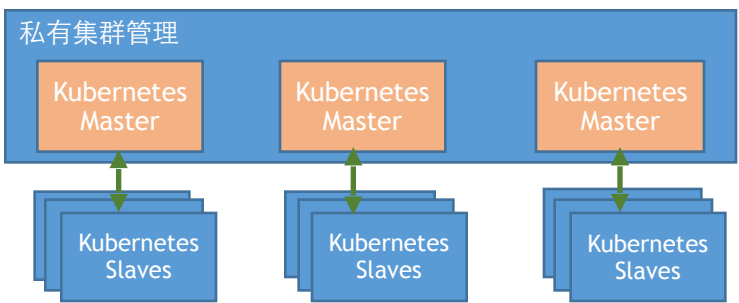
主机管理

国内首个Docker容器主机管理混合云服务正式上线!

欢迎大家试用!

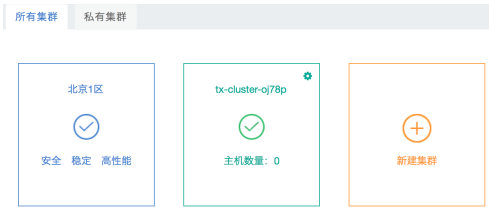
私有主机加入集群管理

- 方便用户管理和整合自己的实体机、虚拟机、云主机等物理资源
- 开发者不需要考虑机器和资源的问题，更专注于自己的应用、编排集成
- 提供丰富的云端服务，直接部署到私有集群
- 提供统一的管理和监控平台

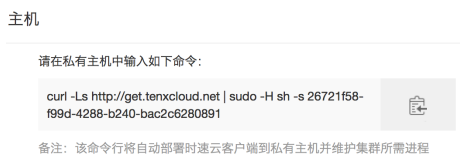


TenxCloud 平台上创建私有集群

1. 创建集群节点 - 集群的master端

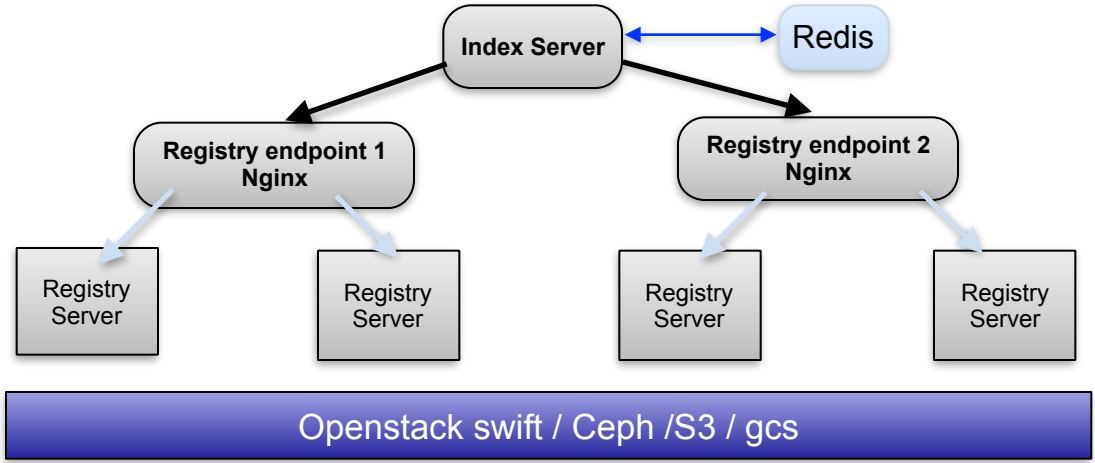


2. 添加自有主机（slave）到集群管理中

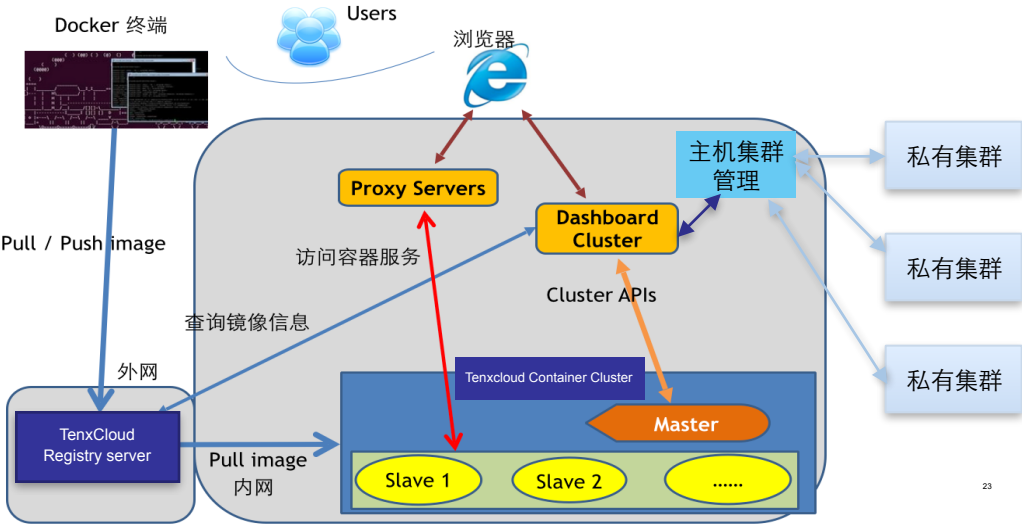


架构实践

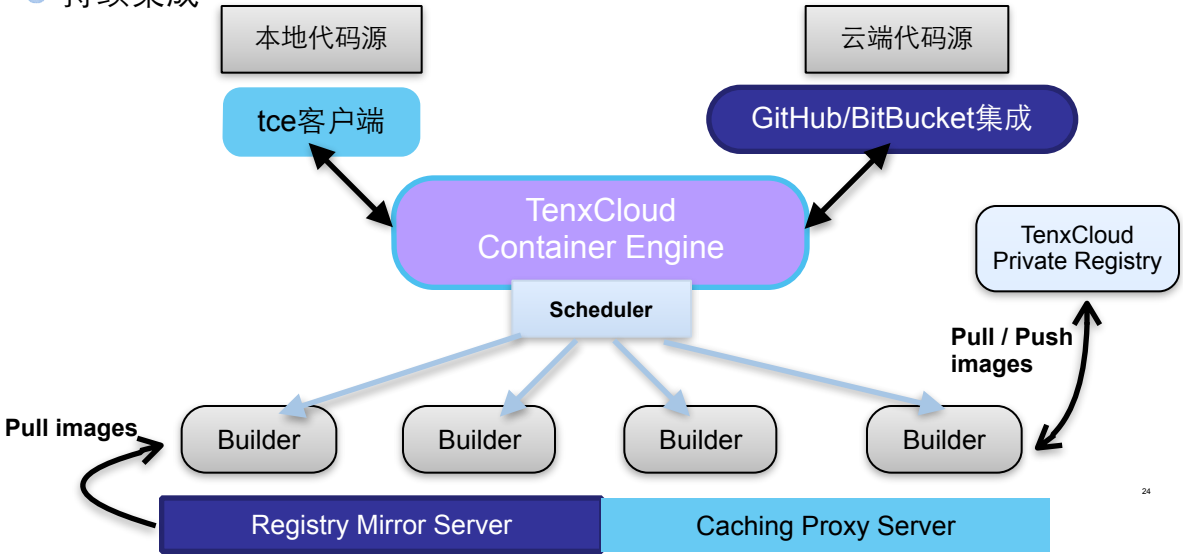
镜像管理 - TenxCloud Docker Registry Service



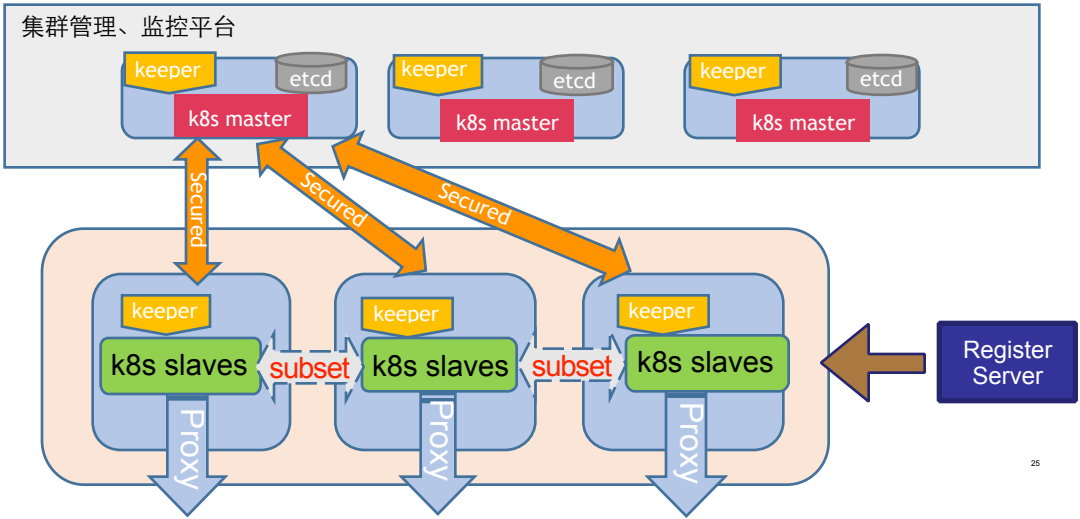
容器、集群管理


Docker 终端
Users
浏览器
Proxy Servers
Dashboard Cluster
主机集群管理
私有集群
私有集群
私有集群
Pull / Push image
访问容器服务
查询镜像信息
Cluster APIs
外网
Tenxcloud Container Cluster
Master
TenxCloud Registry server
Pull image
内网
Slave 1
Slave 2
.....

- 持续集成


本地代码源
云端代码源
tce客户端
GitHub/BitBucket集成
TenxCloud
Container Engine
Scheduler
TenxCloud
Private Registry
Pull / Push
images
Pull images
Builder
Builder
Builder
Builder
Registry Mirror Server
Caching Proxy Server

私有主机集群


集群管理、监控平台
keeper
etcd
k8s master
keeper
etcd
k8s master
keeper
etcd
k8s master
Secured
Secured
Secured
keeper
k8s slaves
subset
keeper
k8s slaves
subset
keeper
k8s slaves
Proxy
Proxy
Proxy
Register Server

将于6月10日正式开启公测

专注互联网应用的容器云平台

Thank You！

下次再见