# 工作报告

**注**:本文档和后续文档均会托管到github.com/oncecloud/document上。

当前我主要关注以下三方面的工作：

* Docker源码解析
* Docker在线迁移与高可用
* BeyondSphere代码精简与重构

## Docker架构

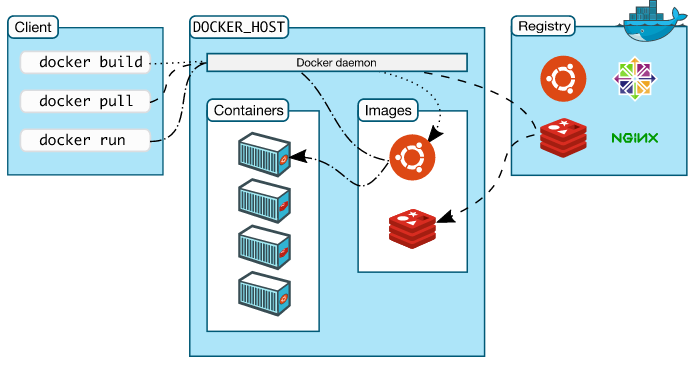


图 1 Docker架构简图

（图片来源：<https://docs.docker.com>）

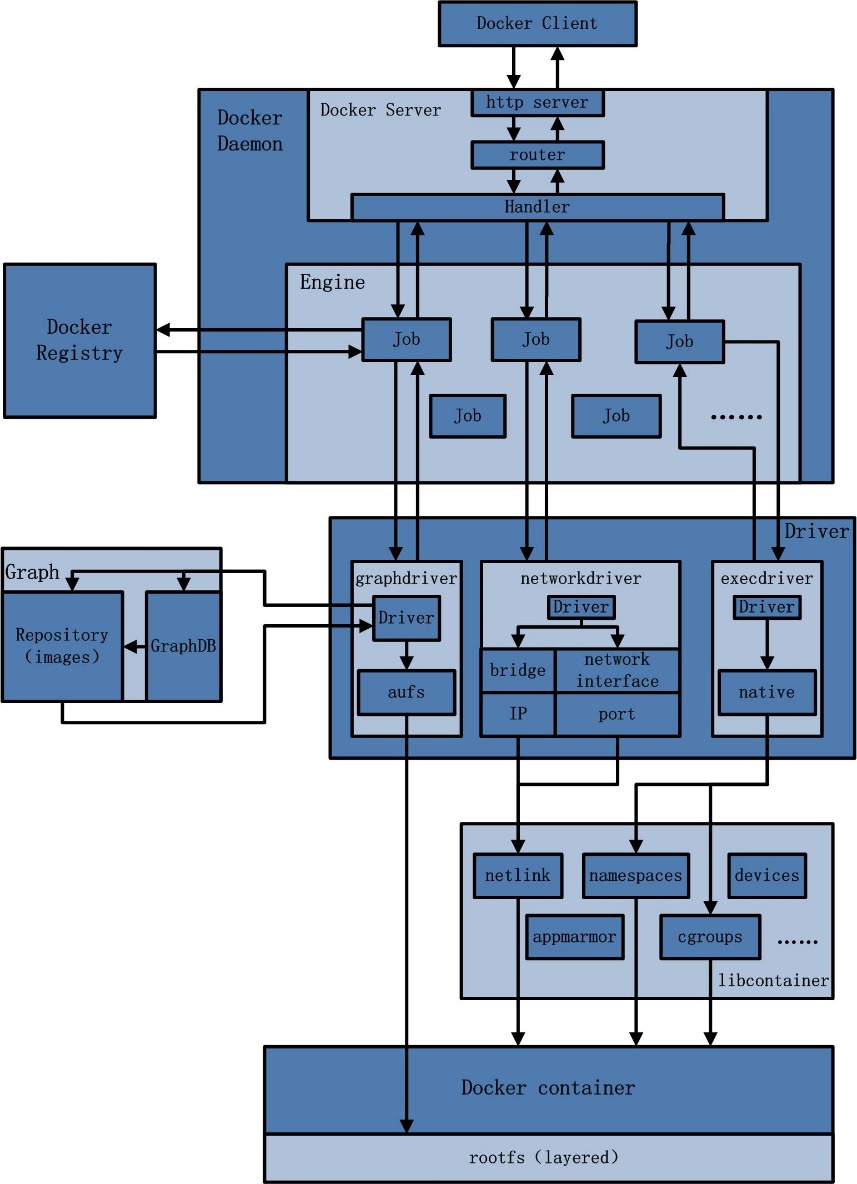


图 2Docker详细架构图

(图片来源：<http://www.infoq.com/>)

Docker采用C/S架构，如图1所示，主要包括Docker daemon和Docker client。

* Docker Daemon

Docker Daemon是Docker架构中一个常驻在后台的系统进程，接受并处理Docker Client发送的请求。该守护进程在后台启动了一个Server，Server负责接受Docker Client发送的请求；接受请求后，Server通过路由与分发调度，找到相应的Handler来执行请求。

* Docker Client

Docker Client是Docker架构中用户用来和Docker Daemon建立通信的客户端。用户使用的可执行文件为docker，通过docker命令行工具可以发起众多管理container的请求。

Docker client可通过以下三种方式与Docker daemon建立连接:

* tcp://host:port
* unix://path\_to\_socket
* fd://socketfd
* Docker三个重要组件
* Docker images
* Docker registry

Docker Registry是一个存储容器镜像的仓库。

* Docker containers

Docker container是Docker架构中服务交付的最终体现形式。

* Docker底层技术
* Namespaces
* Control groups
* Union file systems

AUFS, btrfs, vfs, and DeviceMapper

* Container format

**更为详细的Docker架构参考图2（Docker详细架构图），Docker架构的详细描述见《Docker源码解析》文档。**

例: docker run 命令的运行流程

* **Pulls the** **image**
* **Creates a new container**
* **Allocates a filesystem and mounts a read-write** *layer*
* **Allocates a network / bridge interface**
* **Sets up an IP address**
* **Executes a process that you specify**
* **Captures and provides application output**

## Docker Swarm的高可用

Swarm是Docker社区正在开发的Docker集群管理工具，Swarm架构如图3所示。

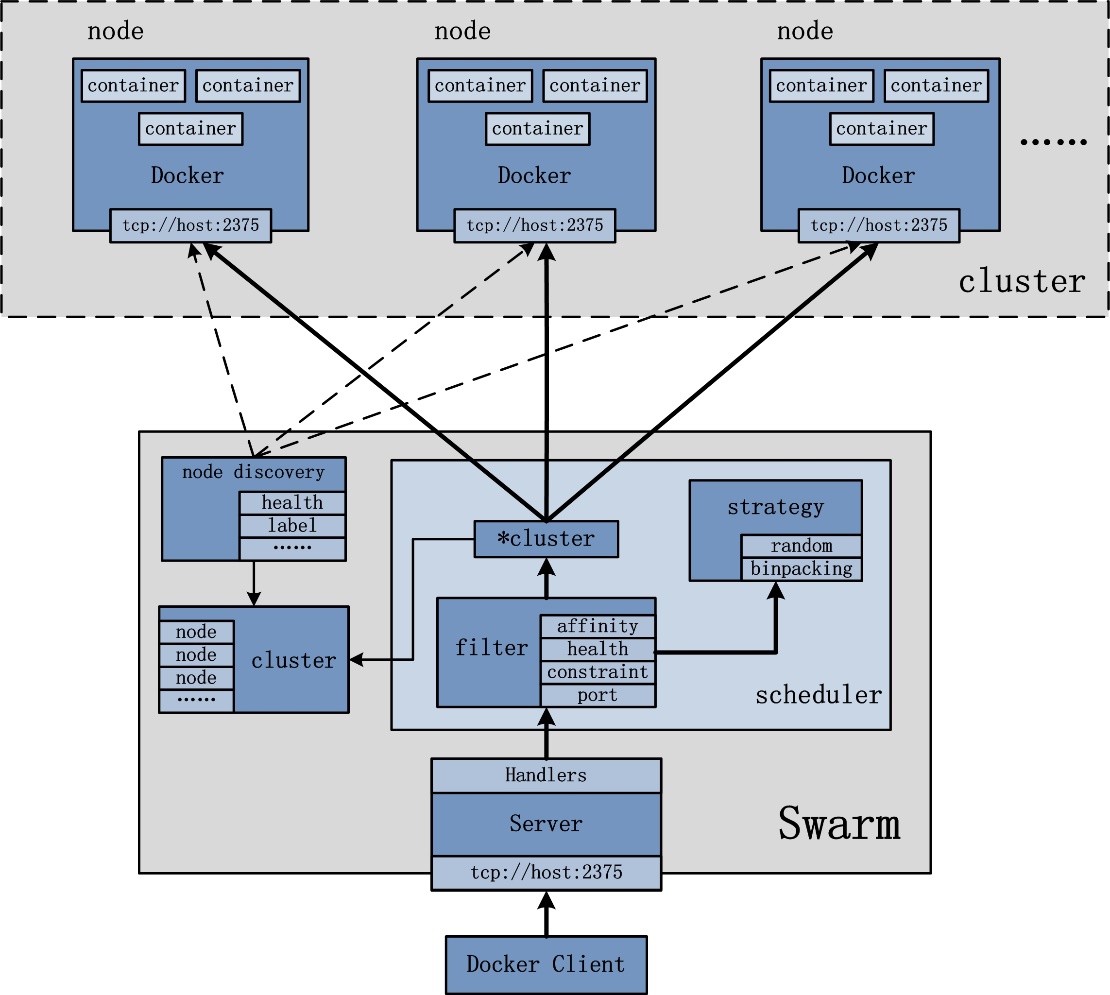


图 3 Swarm架构

（图片来源: <http://blog.daocloud.io/>）

Swarm当前采用了一种针对管理节点的高可用方案：

如图4所示，主要策略是对管理节点做一个冗余备份，如果主要节点失效，则从备份节点中重新选举一个主管理节点来管理Docker集群。



图 4 Swarm高可用

## Docker在线迁移

基于共享存储的方式已经做过实验（郝庭毅），仍存在一些问题，可行性有待进一步实验验证。

## BeyondSphere

当前BeyondSphere项目采用Maven管理依赖，并用Maven module的方式对项目大幅度解耦，使项目结构清晰，有利于协同开发，但仍然存在以下问题：

* 部分代码冗余，耦合度高，逻辑不清晰
* 没有标准的测试框架，项目可靠性难以保障
* 项目构建步骤繁琐，影响开发效率
* 安装、部署困难

对此，我准备做以下工作：

* 代码精简与重构
* 加入TestNG测试框架
* 采用Hudson进行持续集成
* 采用Cargo进行自动化部署
* 包装成Docker image(如果需要)