# 基础设施层

## 关键文献阅读

### 1. Docker Swarm and Kubernetes in Cloud Computing Environment

#### 研究内容

主要是对Docker Swarm和Kubernetes进行性能对比。感觉可以往对业务调整比较少靠一靠。

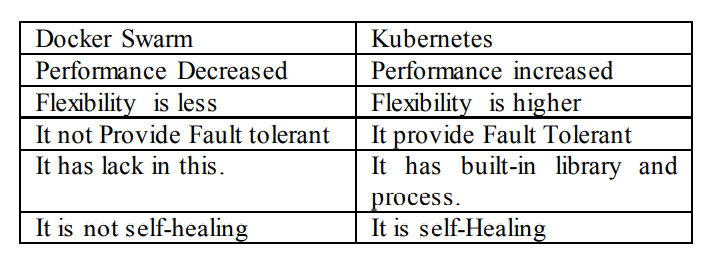


Figure : Difference between docker swarm and Kubernetes

#### 可参考点

该论文仅仅分析了CPU和内存等低级指标，没有涉及到业务，可引到对业务调整比较少。

### 2. Elixir: An Agent for Supporting Elasticity in Docker Swarm

#### 研究内容

采用了Elixir去管理Docker swarm，对容器进行弹性伸缩。由于Docker swarm本身对于其没有管理。

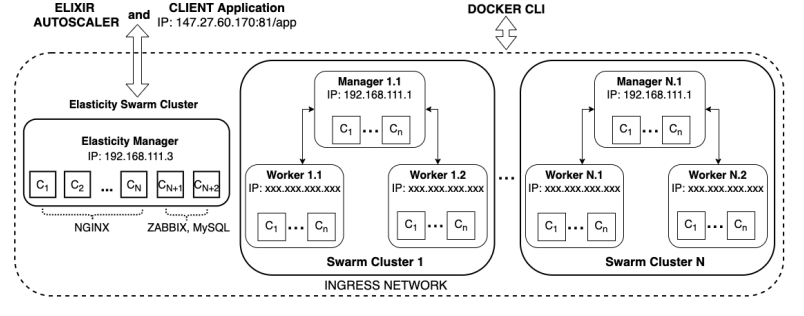


Figure : Docker Swarm deployment with Elixir agent

#### 可参考点

仍然是没有对业务进行调整，仅仅是针对数据检测。尽管通过了一个第三方的Monitor处理，但仍然仅仅是对各个swarm的数据进行监测。

### 3. Building Modern Clouds: Using Docker, Kubernetes & Google Cloud Platform

#### 研究内容

标题三者的对比，比较有价值的内容就是下图。

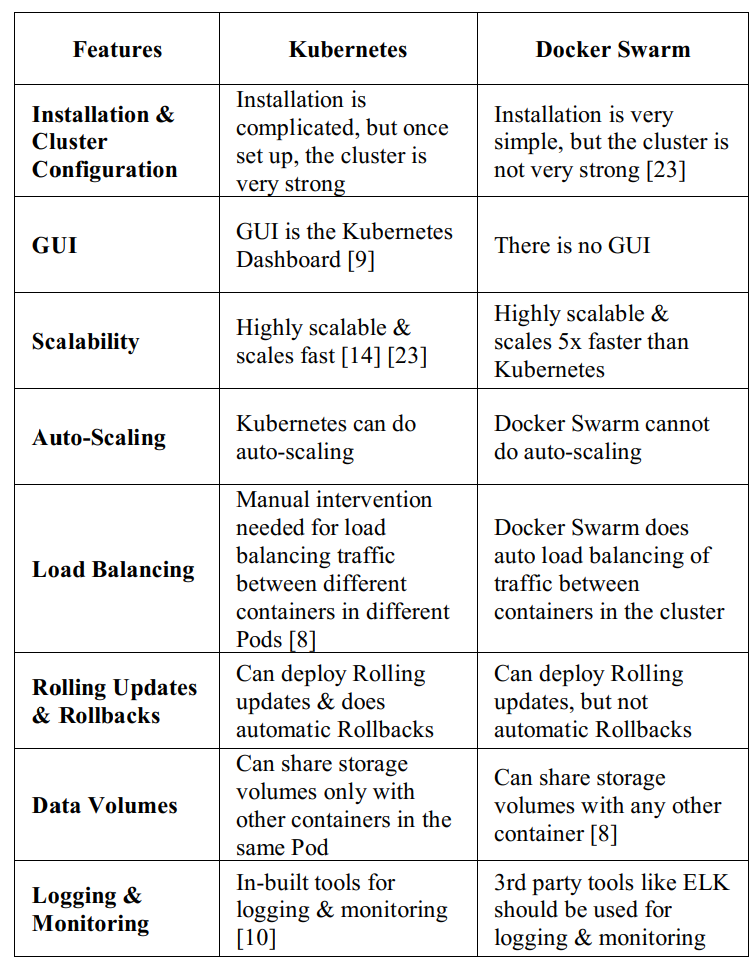


Figure : Comparison of docker and k8s

#### 可参考点

由上表可以看出，k8s目前对于负载均衡仍需要人为的干预，在devop阶段，而docker swarm支持的更少，可以作为重点引入。

### 4. Self-adaptive K8S Cloud Controller for Time-sensitive Applications

#### 研究内容

通过对k8s的controller添加了一个self-adaptive机制，属于对基础设施层的增强工作。

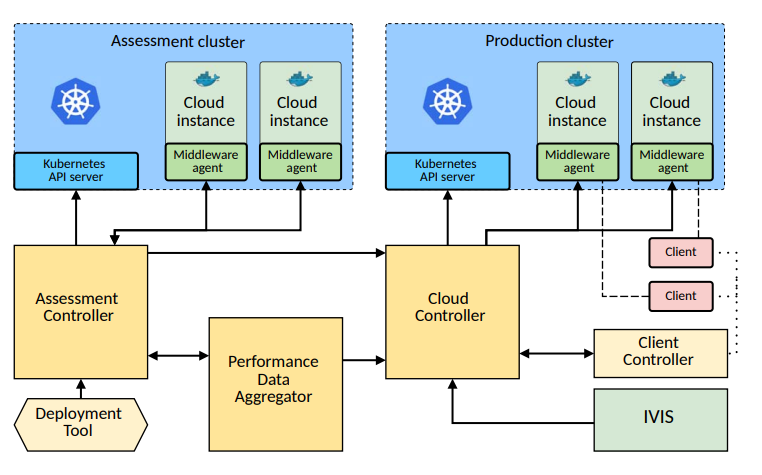


Figure : High-level overview of the framework architecture

#### 可参考点

仍然是通过增强基础设施层的部分功能以实现自适应功能，虽然能够处理adaptation,但是还是没有考虑业务，针对的是单一的问题，解决实时性问题。

### 5. Impact of etcd deployment on Kubernetes, Istio, and application performance

#### 研究内容

一篇比较好的文章，介绍etcd在k8s和Istio的性能情况。

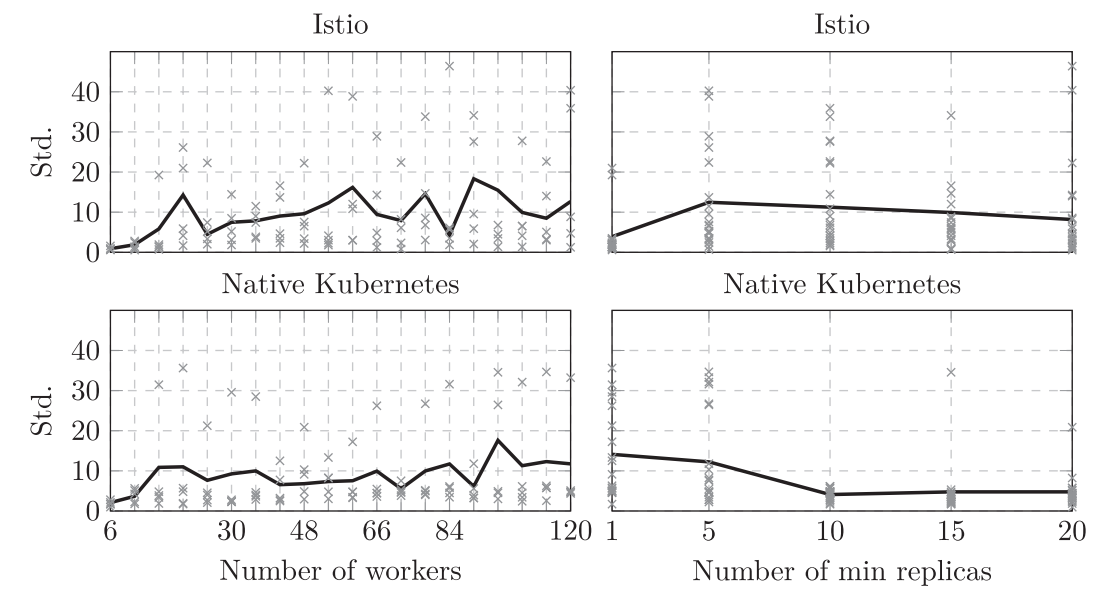


Figure : k8s and istio

#### 可参考点

Istio本质就是在k8s上的升级，其实仍然是通过将流量透明，使得更好实现服务管理。但仍然没能解决核心的问题：

* 业务调整
* AIOps

### 6. Programming framework and infrastructure for self-adaptation and optimized evolution method for microservice systems in cloud-edge environments

此文提出了一种新的架构，现有的微服务编程范式和集装箱技术无法完全支持微源的自主演变。例如将不同SLA的服务实例（什么是SLA？）上的资源限制调整为一个组成一个服务，这改进了QoS，演变的微服务包括添加新实例，删除现有实例，移动实例，调整现有实例上的资源约束，合成和拆分服务。

其次，尽管每个微服务本身都可以发展，但是在复杂的服务系统中有大量的组成微服务，必须建立一种机制以确定进化的最合适的时间并协调多个单独的微服务的演变。虽然kubernetes 等现有方法可以根据性能问题自动缩放系统中的服务，但他们无法处理来自用户的功能要求，只能通过部署新服务来满足。它们还缺乏通过从所有情况收集必要的运行时信息并分析QoS和SLA的变化来演化系统的方法。有必要拥有一个常见的基础设施，支持自适应对用户需求的变化和用户感知QoS的劣化。

### 7. Hierarchical Edge-Cloud Microservice Infrastructure for Scientific Instruments’ Lifetime Connectivity

介绍了一种边缘云微服务基础设施，通过边缘增强了现有的基于cloud的基础设施，解决了通过网络连接到云时面临的一些问题。

### 8. Towards Automated Inter-Service Authorization for Microservice Applications

微服务之间的通信通过网络暴露出来会产生潜在的攻击面。仅仅依靠网络边界来提供全面的安全保护是不现实的。当一个微服务受到威胁时，它可以向其他微服务发送恶意请求以发起攻击或窃取数据。因此，除了使用 SSL/TLS 来保护微服务之间的通信外，Kubernetes 和 Istio等流行的微服务基础设施还提供了服务间授权机制来指定微服务可以发起哪些调用。例如，被入侵的日志服务可能会与后端服务对话以获取敏感信息，管理员可以指定只有前端服务才能访问后端服务以防御这种攻击。

## 思路

对于基础设施层的现状的思路，个人认为可以从以下几点出发：

* 首先是目前的基础设施层对于**业务的调整**是比较少的。由关键论文1-3可以看出，目前的基础设施层往往提供的是底层的功能，例如弹性伸缩和负载均衡，都是针对容器（Pod）本身进行调控的，没有对业务涉及。尽管有些研究通过增加外部控制加强了这些功能，但仍然没有涉及到对业务的调整。
* 其次是针对数据检测，从基础设施层目前仅仅只是针对一些网络需求、QoS处理。可以与数据检测挂钩。
* 同时，阅读了一些与自适应相关的结合基础设施层操作的论文，目前解决的问题都是针对特定领域的，例如IoT领域，以及实时领域需求。并没有从运维的角度出发，全面地考虑该问题。
* 关于针对未知情况，目前常见的基础设施层处理的方式较少，这点现状难以靠近，可不可以从服务熔断这种出发，k8s的服务降级与熔断也是比较单一的指标。