Kubernetes 可视化管理平台

赵旭杰,梁正和
(河海大学 计算机与信息学院,江苏 南京 211100)

摘 要:为了方便云平台的管理,摆脱使用 Kubetcl 命令行方式操作 Kubernetes 集群的繁琐,前台采用 React 框架构建 UI 界面,后台使用 Fabric8 框架调用 Kubernetes 云平台的功能接口,前后端数据的交互借助 Axios 框架来完成。最终成功完成了基于 Docker 的 Kubernetes 可视化管理平台的开发。通过该平台,管理人员可以操作 Kubernetes 集群中 Pod、Namespaces、Service 等核心组件。并且在此基础上,该文借助时序数据库 InfluDB 和可视化监控工具 Grafana,实现了平台资源的监控。通过集成 Harbor 管理页面,可以对 Docker 私有镜像仓库进行管理。最终,Kubernetes 可视化管理平台实现的功能主要包括:物理节点信息显示,Pod、Controller、Service、Namespace 的增删改查,平台资源监控和私有镜像管理。这种所见即所得的管理方式,可以充分发挥容器集群自动化部署、自动化扩缩容、自动维护的特性。同时,也使得容器集群的管理更加方便,极大地提高了平台管理的效率,降低了管理人员的负担。

关键词:Kubernetes;Docker;React;容器云;可视化管理
中图分类号:TP391 **文献标识码**:A **文章编号**:1673-629X(2021)02-0106-05
doi:10.3969/j.issn.1673-629X.2021.02.020

Kubernetes Visual Management Platform

ZHAO Xu-jie,LIANG Zheng-he
(School of Computer and Information,Hohai University,Nanjing 211100,China)

Abstract:In order to facilitate the management of the cloud platform and get rid of the cumbersome operation of Kubernetes cluster by using Kubetcl command line,the React framework is used in the foreground to build the UI interface,while the Fabric8 framework is used in the background to call the functional interface of Kubernetes cloud platform,and the interaction between the front and rear data is completed with the help of Axios framework. Finally,the development of Kubernetes visual management platform based on Docker is successfully completed. Through this platform,managers can operate core components such as Pod,Namespaces,and Service in Kubernetes cluster. On this basis,we realize the monitoring of platform resources with the help of the timing database InfluDB and the visual monitoring tool Grafana. By integrating Harbor management page,Docker private image warehouse can be managed. Finally,the functions realized by Kubernetes visual management platform mainly include physical node information display,addition,deletion and revision of Pod,Controller,Service,Namespace,and platform resource monitoring and private image management. This management method can make full use of container cluster automation deployment,automatic expansion,automatic maintenance. At the same time,it also makes the management of container cluster more convenient,greatly improving the efficiency of platform management and reducing the burden of managers.

Key words:Kubernetes;Docker;React;container cloud;visual management

0 引 言

随着计算机技术的快速发展,传统的应用部署方式已经无法满足开发者的需求。现如今,既要避免应用与操作系统的绑定,又要改善虚拟机技术十分笨重,不利于移植的特点。

Docker 这种应用容器引擎的出现填补了这个空缺。相对于传统方式的虚拟机,它占用的空间更少,启动更快,并且不需要虚拟出整个操作系统,只需要虚拟出一个小规模的环境。每个容器之间互相隔离,并且有着自己的文件系统。容器之间可以通过共用网络资源栈的方式进行交互[1]。任何应用都可以被打包成镜像,在发布成功后,供其他人拉取使用[2]。但是,要将 Docker 应用于具体的业务实现,还是存在着很大的困难。人们需要一种基于容器的集群管理平台,来解决 Docker 在编排、管理和调度等方面上存在的问题。

Kubernetes 的出现填补了这个空缺,它是开源的

收稿日期:2020-04-08 **修回日期**:2020-08-11
基金项目:国家自然科学基金项目(61272543)
作者简介:赵旭杰(1996-),男,硕士,研究方向为分布式系统;梁正和,博士,研究方向为分布式系统。

容器集群管理系统,可以实现容器集群的自动化部署、自动扩缩容、自动维护等功能[3-4]。并且因为它是基于容器的技术,在移植性、扩展性和自动化方面也具有很大的优势。Kubernetes 可以满足生产环境中的很多常见需求,比如:多个进程的协同工作、存储系统挂载、应用实例的复制、日志访问、负载均衡等[1]。通过 Kubernetes,可以直接管理云平台中多个主机上的容器化的应用。但与此同时,它也带来了一些操作上的麻烦。目前,大多使用 Kubectl 命令行的方式,对平台应用进行管理。而这种方式,很大程度上降低了开发的效率,并且提高了运维的成本。

为了改善这种情况,该文使用集成开发平台 Fabric8 实现对 Kubernetes 集群的操作,并且使用前端框架 React 和组件库 Ant Design 实现平台的可视化。通过该平台,管理人员可以对 Kubernetes 中的物理节点、Pod 节点、Replication Controller、Namespaces、Service 直接进行操作,很好地取代了命令行的方式。

1　系统集成相关技术

1.1　Kubernetes

Kubernetes 是一种面向应用的容器集群部署和管理的系统,前身源自 Google 公司内部的 Borg 系统,有着强大的开源支撑和 10 年以上的技术沉淀[5]。相对于 Apache 下的 Mesos 和 Docker 下的 SWARM,Kubernetes 具备的功能更加齐全,部署也更加方便。Kubernetes 在具有强大的集群管理、防护策略和自我修复能力的同时,还支持弹性伸缩和负载均衡,并且提供便捷的服务升级和完善的管理工具[6]。Kubernetes 架构如图 1 所示。

Kubernetes 由管理节点和工作节点组成。可以简单理解为它们之间是主从的关系[7]。为了提高平台的可靠性和决策能力,一般会部署奇数数量的管理节点,而工作节点也会部署多个,具体数目由实际资源情况决定。采用这种策略,可以有效地避免在某个节点宕机后,对整个集群产生影响。

管理节点中的核心组件有 APIServer、Etcd、Controller Manager、Scheduler 等。APIServer 是资源操作的唯一入口,有着身份认证、授权和访问控制等机制,也可以用于 API 注册和发现[1,8-9]。管理人员借助 Kubectl 命令行工具,可以实现对于 Kubernetes 集群的管理。Etcd 是一个可信赖的分布式键值存储服务,是 Kubernetes 实现存储化的方案,里面保存着整个集群的状态[9-10]。而 Scheduler 则负责资源的调度,可以将 Pod 调度到相应的物理机器上[11]。在调度的过程中,它会首先与 APIServer 进行交互,然后通过 APIServer 将请求写入到 Etcd 中。Controller Manager 负责控制器的管理,使得集群达到期望中的状态。Controller Manager 中的 Replication Controller、Replica Sets 和 Deployment 都可以直接或间接控制 Pod 的副本数量,使其达到期望的副本数目。并且 Controller Manager 具备故障检测、自动拓展和滚动更新等功能。

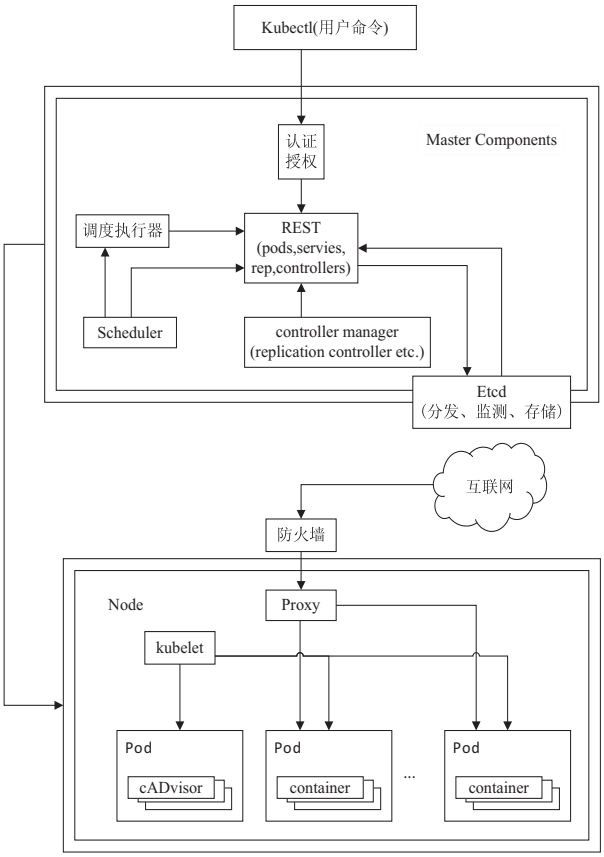

图 1　Kubernetes 架构

工作节点中主要运行着 Kubelet、Kube-proxy 和 Pod。Kubelet 负责管理容器的生命周期,可以对节点中的容器进行删除和添加[8,12]。为了实现容器间文件的持久化和共享,需要借助 Volume(CVI),而这也是由 Kubelet 进行管理的[1,13]。除此之外,Kubelet 还负责容器的网络管理和监控等。因为节点中容器有着大量的副本,所以 Kubernetes Service 想要访问其中对应的副本,需要借助于 Kube-proxy。并且 Kube-proxy 通过 IPVS 的模式,可以让 Service 与 Pod 之间实现负载均衡。Pod 既是工作节点中的主要成分,也是 Kubernetes 的最小单位,它的生命周期取决于其内部容器的状态[14]。

1.2　Fabric8

Fabric8 是一个基于 Docker、Kubernetes 和 Jenkins 的开源微服务平台。通过 Continuous Delivery 管道可以很方便地创建、编译、部署和测试微服务,并且通过 Continuous Improvement 和 ChatOps 可以对这些微服务进行运行和管理[15]。

在该项目中,会通过 Fabric8 调用 Kubernetes 的功

能接口,使用 Java 语言实现可视化界面中后端接口的开发。

1.3 React

React 是一个用于构建页面 UI 的库,它以组件化的思想开发网站。开发者从功能的角度出发,把 UI 拆分为不同的组件,各组件只负责自己部分的 UI 和逻辑,彼此互相独立,不同的组件可通过组合或嵌套的方式和其他组件一起使用[16]。

该文通过 React 技术,将 Kubernetes 中每个应用拆分为各个组件。可以在菜单栏中,选择想要操作的功能,进入相应的组件。通过各个功能的结合,可以达到在网页中,直接操作 Kubernetes 的目的。

2 平台设计

2.1 总体设计

Kubernetes 容器云管理平台分为四层架构(见图 2),分别为:资源层、容器层、平台层和管理层。

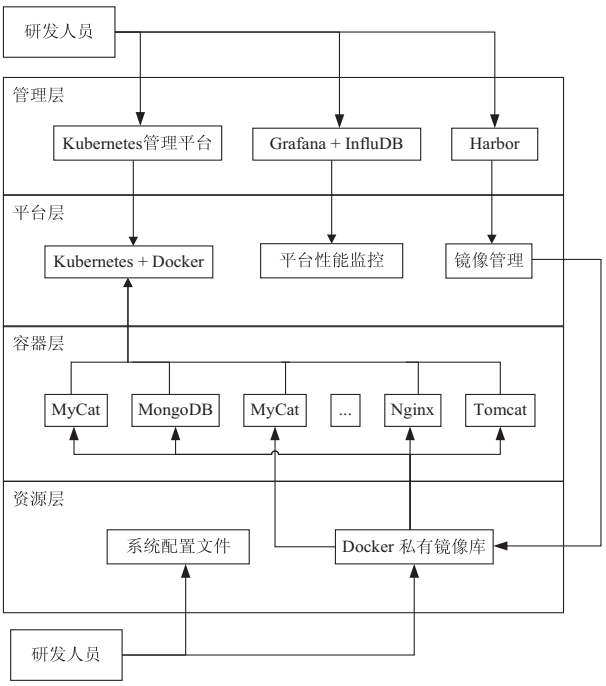

图 2 层次架构

资源层是平台的最下面一层,它为容器层提供容器镜像和一些系统配置文件。研发人员可以直接对其进行修改配置,也可以通过管理层中的应用对其进行管理。

容器层是 Kubernetes 集成平台的基础,由开发人员自身创建的私有镜像库提供。可以根据实际需求,自由组装平台需要的运行环境。

平台层是系统的核心,其他各层都是为其服务或者对其进行配置。其中最重要的是 Docker 和 Kubernetes 这两个组件。Docker 为容器提供了运行环境,Kubernetes 则负责统一编排和调度 Docker。

管理层主要是配置平台层中的应用,InfluDB 加上 Grafana 可以对平台的状态进行监控,而 Harbor 会被用来管理私有镜像库。

2.2 平台架构描述

使用 React 作为前端开发框架,使用 Axios 用于前后端接口的交互,并且使用 React 生态圈的新特性 React Hook 进行开发。借助 React Hook 可以把所有的组件都定义为函数,而不使用原来继承类的形式,这样可以使得组件的复用性得到很大的增强。

借助 React Router 来保持 UI 与 URL 间的同步,并且使得向应用中添加视图和数据流更加便捷。Axios 向后端端口请求到的数据,会在页面跳转后,被渲染到对应的 UI 组件中。

项目后台由 Spring 和 Spring Boot 框架开发而成(见图 3),该层使用 RequestMapping 来完成请求地址的映射,使用框架 Fabric8 完成与云集成平台的交互,从而可以通过前端页面来完成对于 Kubernetes 集成平台的管理。

2.3 功能模块描述

平台主要是为了使得原本的命令行管理方式,变得所见即所得。所以要把原本的功能,通过页面的形式给复现出来。如图 4 所示,主要包括以下几种功能:

(1)物理节点。

需要知道部署在云平台上的主机节点和从属节点的 IP 地址。

(2)命名空间。

为了解决在集群下管理对象时的复杂性问题,Kubernetes 使用命名空间的概念。实际使用的时候,需要做到查看和创建命名空间的功能。

(3)Service 管理。

Service 是 Kubernetes 中的核心资源对象,它将运行在一组 Pods 上的应用程序公开为网络服务的抽象方法。所以,既要获取所有的 Service 服务名称,也要知道每个 Service 下的具体信息。并且,也需要根据需求,添加或删除相应的 Service。

(4)Replication Controller 管理。

RC(Replication Controller)的主要作用是:确保容器应用的副本数,始终保持在用户定义的数量。它可以根据情况,创建或回收 Pod。在该模块中,要实现对于所有 RC 的查询,并且可以查询每个 RC 的详细信息。除此之外,还可以根据需要,手动增加或删除相应的 RC,也可以设置容器应用中的副本数。

(5)Pod 节点操作。

Pod 是 Kubernetes 中能够创建和部署的最小单位,里面包含一个或多个容器。所以,需要清楚地知道里面的各种属性。

(6)中间件。

这里的中间件,主要是实现了对于 MyCat 的配置和查询。可以通过前端页面,直接配置相应的 MyCat 文件,而不需要在服务器端进行配置。

(7)资源监控。

该功能采用 InfluDB(时序数据库)加上 Grafana 的形式。InfluDB 是一种开源分布式时序、时间和指标数据库,可以按照时间维度索引数据。Grafana 是一款美观、强大的可视化监控指标展示工具,可以很好地展示 InfluDB 监控到的数据。通过 InfluDB 将 Kubernetes 集群平台的资源使用情况记录下来,再通过 Grafana 进行图形化展示,可以很好地监控集群平台的状态。

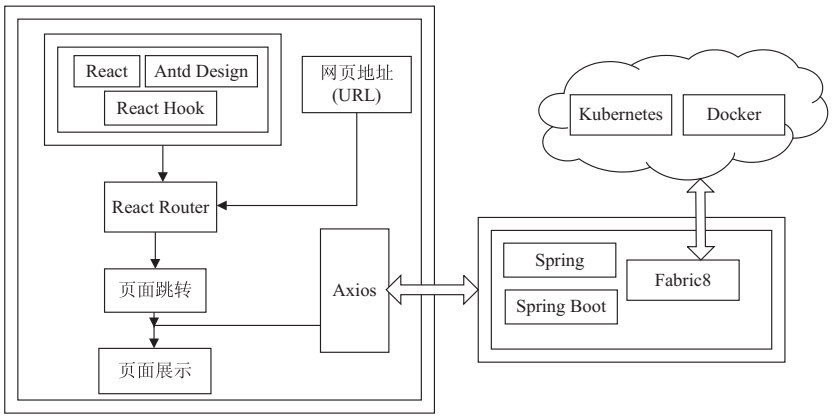

图 3　前端各框架的关联

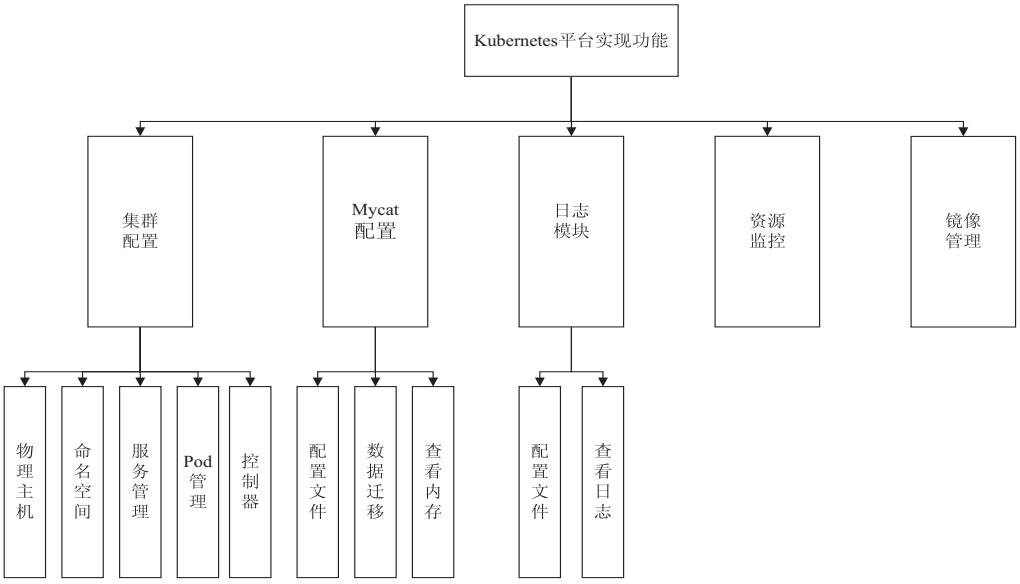

图 4　功能模块

(8)镜像管理。

使用 Harbor 作为 Docker 私有镜像仓库。Harbor 提供了友好的 Web UI 界面,也可以进行用户权限管理等功能。将 Harbor 集成到该平台之中,可以很方便地对镜像进行操作。

(9)日志模块。

平台中的日志模块,主要实现了对于操作的记录,以方便日后的还原和查询。与此同时,也包含一些配置文件的读取和操作。

3　实现细节

3.1　前端组件实现

前端组件的组合,得益于 React Router 的作用。React Router 使得组件之间可以借助统一资源定位符,完成嵌套和地址的映射。这些组件中,有负责导航的类,比如:NsIndex、AdminIndex、ServiceIndex 等。也有负责添加、删除、展示的类,比如:NodeShow、ServiceShow、AddService 等。它们借助于 React Hook 中的生命周期函数和 Axios 框架,可以实现与后端程序的交互。

3.2　后台端口实现

后台通过 RequestMapping 完成请求地址的映射,前台的请求会首先在自身创建的 SysGeneratorControlle 类中进行处理,然后调用对应的业务逻辑类。在进行业务逻辑处理的时候,会使用到一些自定义的类,比如:nodeSelf、podSelf、serviceSelf,

这些类全部继承自 Fabric8 框架中的原有类。该文在原来的基础上,定义了一些公共方法,来完成一些特定的功能,比如:选择性展示、添加、删除、搜索等。

4 结束语

主要阐述了 Kubernetes 可视化管理平台的构建方式。主要是通过 React 框架来完成前台页面的渲染,通过 Fabric8 进行 Kubernetes 接口的调用,通过 Axios 框架实现前后端数据的交互。相比于通过指令控制的方式,这种方式更加简便,对操作人员的要求也更低。

该文构建的可视化管理界面,实现了一些核心功能,但也存在着一些不足和缺漏。在此基础上,可以进一步完成功能上的开发,也可以进行安全上的完善。

参考文献:

[1] 王 敏. 基于 Docker 的数据科学虚拟化实验平台构建[J]. 实验室科学,2019,22(3):104-106.

[2] 丁海斌,崔 隽,陆 凯. 基于 Docker 的 DevOps 系统设计与实现[J]. 指挥信息系统与技术,2017,8(3):87-92.

[3] 陈建娟,刘行行. 基于 Kubernetes 的分布式 ELK 日志分析系统[J]. 电子技术与软件工程,2016(15):211-212.

[4] TAHERIZADEH S, GROBELNIK M. Key influencing factors of the Kubernetes auto-scaler for computing-intensive microservice-native cloud-based applications[J]. Advances in Engineering Software, 2020, 140(C): 102734.

[5] BURNS B, GRANT B, OPPENHEIMER D, et al. Borg, Omega, and Kubernetes[J]. Communications of the ACM, 2016, 59(5): 50-57.

[6] 翁涅元,单杏花,阎志远,等. 基于 Kubernetes 的容器云平台设计与实践[J]. 铁路计算机应用,2019,28(12):49-53.

[7] 金子威. 基于 K8S 的 Docker 分布式容器自动化运维系统的设计与实现[D]. 武汉:中南民族大学,2018.

[8] 王伟军. 基于 Kubernetes 的容器云平台建设[J]. 电脑知识与技术,2019,15(36):47-48.

[9] PAN Y, CHEN I, BRASILEIRO F, et al. A performance comparison of cloud-based container orchestration tools[C]//2019 IEEE international conference on big knowledge (ICBK). Beijing, China: IEEE, 2019: 191-198.

[10] 余昌发,程学林,杨小虎. 基于 Kubernetes 的分布式 TensorFlow 平台的设计与实现[J]. 计算机科学,2018,45(11A):527-531.

[11] 周佳威. Kubernetes 跨集群管理的设计与实现[D]. 杭州:浙江大学,2017.

[12] 翟雅荣,于金刚. 基于 Filebeat 自动收集 Kubernetes 日志的分析系统[J]. 计算机系统应用,2018,27(9):81-86.

[13] MERCL L, PAVLIK J. Public cloud kubernetes storage performance analysis[C]//Computational collective intelligence. Cham: Springer, 2019.

[14] MEDEL V, TOLOSANA-CALASANZ R, BAÑARES J Á, et al. Characterising resource management performance in Kubernetes[J]. Computers & Electrical Engineering, 2018, 68: 286-297.

[15] Red Hat. Fabric8 Documentation[EB/OL]. 2017. http://fabric8.io/guide/overview.html.

[16] 张志鹏,黄素娟,周永圣,等. 基于 React 技术的单页 APP 的设计与实现[J]. 微型电脑应用,2019,35(10):71-74.