Journal of Computer Applications
计算机应用, 2021, 41(4): 956 - 962
ISSN 1001-9081
CODEN JYIIDU
2021-04-10
http://www.joca.cn

文章编号:1001-9081(2021)04-0956-07
DOI:10.11772/j.issn.1001-9081.2020111977

基于Kubernetes的Fabric链码管理及高可用技术

刘宏宇,梁秀波*,吴俊涵
(浙江大学 软件学院,浙江 宁波 315048)
(*通信作者电子邮箱 xiubo@zju.edu.cn)

摘　要: 区块链即服务(BaaS)平台的核心在于如何将区块链网络部署在云计算平台上。Fabric部署可以按照组件启动时间分为静态组件和动态链码两部分,而链码部署是Fabric云化最核心、最复杂的部分。因为Fabric本身没有针对Kubernetes开发接口,所以业界当前的方案均是通过一系列辅助技术实现链码部署,而这些方案并没有将链码随静态组件一起纳入到Kubernetes管理环境中。针对当前BaaS方案存在的问题,主要做了如下几项工作:1)比较全面地研究了底层基础设施,尤其是生产环境下的高可用性Kubernetes平台;2)设计并实现了Fabric在Kubernetes上的云化部署,尤其是链码部分通过一个全新的容器控制插件实现了对Kubernetes在代码级别上的支持,并完成了将链码纳入Kubernetes环境管理的目标;3)用函数计算服务来管理Fabric链码的部署,从而实现了一个全新的链码执行模式,即从"启动-等待-调用-等待"的模式改变为"启动-调用-退出"的模式。上述在Fabric云化部署尤其是链码部署管理方面的工作,对基于Fabric和Kubernetes的BaaS平台优化有一定的参考价值。

关键词: 区块链;区块链即服务;Hyperledger Fabric;Kubernetes;链码;无服务器函数计算
中图分类号: TP311.5　　**文献标志码:** A

Kubernetes-based Fabric chaincode management and high availability technology

LIU Hongyu, LIANG Xiubo*, WU Junhan
(*School of Software Technology*, *Zhejiang University*, *Ningbo Zhejiang* 315048, *China*)

Abstract: The core of the Blockchain as a Service (BaaS) platform is how to deploy the blockchain network on the cloud computing platform. Fabric deployment can be divided into static components and dynamic chaincodes according to the component startup time, and chaincode deployment is the core and the most complex part of Fabric cloudification. Because the Fabric has no interfaces for Kubernetes, the current solutions in the industry implement chaincode deployment through a series of auxiliary technologies, but these solutions do not incorporate the chaincodes into the Kubernetes management environment along with static components. In response to the existing problems of BaaS scheme, the following works were mainly done: 1) a comprehensive study of the underlying infrastructure, especially of the Kubernetes platform with high availability in the production environment; 2) the cloud deployment of Kubernetes on Fabric was designed and implemented, especially in the chaincode part, a brand-new container control plug-in was used to realize the support for Kubernetes at the code level and complete the goal of incorporating chaincodes into Kubernetes environment management; 3) the functional computing service was used to manage the Fabric chaincodes to realize a brand-new chaincode execution mode, which means changing from the "start-wait-call-wait" mode to the "start-call-exit" mode. The above works in Fabric cloud deployment, especially in chaincode deployment management, have certain reference value for the optimization of the BaaS platform based on Fabric and Kubernetes.

Key words: blockchain; Blockchain as a Service (BaaS); Hyperledger Fabric; Kubernetes; chaincode; serverless function computing

0　引言

区块链即服务(Blockchain as a Service, BaaS)是一种创建、管理和运维企业级区块链网络及应用的云服务平台,它可以为开发者提供便捷、高性能的区块链服务,并降低开发成本[1]。按照云计算服务的划分,BaaS有平台即服务(Platform as a Service, PaaS)和软件即服务(Software as a Service, SaaS)两种形态,目前绝大多数以PaaS为主,对外提供一种打包封装好的区块链技术平台[2]。BaaS平台的发展需要云计算和区块链两个方向的技术能力,具有较高的技术门槛,云计算巨头公司提供了大多数的BaaS平台。

IBM在2016年2月宣布推出区块链服务平台,支持用户通过Bluemix的区块链服务完成创建、部署、运行和监控区块链应用程序的任务;微软在2016年8月正式开放其Azure云

收稿日期: 2020-12-16;**修回日期:** 2021-01-14;**录用日期:** 2021-01-15。
基金项目: 浙江省教育厅一般科研项目(Y202044224);浙江省研究生教育学会项目(2020-002);浙江省文物保护科技项目(2019009)。
作者简介: 刘宏宇(1994—),男,河北邯郸人,硕士,主要研究方向:容器云、区块链;　梁秀波(1983—),男,山东莱阳人,副研究员,博士,CCF会员,主要研究方向:区块链、人工智能、自然人机交互、移动互联网;　吴俊涵(1998—),男,浙江丽水人,硕士研究生,CCF会员,主要研究方向:区块链。

平台的BaaS服务，支持多种区块链网络（Hyperledger Fabric、Ethereum、Corda等），提供快速构建、管理和扩展区块链网络的能力，Azure用户可以专注于业务开发，实现高效上链；AWS（Amazon Web Services）在2018年4月，发布了区块链模板服务，AWS用户可以选择Ethereum、Hyperledger Fabric等常见的开源框架，快速创建和部署区块链网络[3-4]。

中国互联网公司先后推出BaaS平台：2017年4月，腾讯发布了区块链白皮书和自主研发的TrustSQL区块链平台；2018年2月，华为云推出BCS（BlockChain Service）区块链解决方案，旨在通过与人工智能（Artificial Intelligence，AI）、物联网（Internet of Things，IoT）、5G等前沿技术的结合构建一个坚实的技术底座，使能各行业数字化高效转型；2018年7月，阿里云商业产品区块链服务（公测）发布，该服务可以帮助用户快速构建稳定、安全的生产级区块链网络，减少在区块链部署、运维、开发等方面的挑战，让用户专注于核心业务创新，实现快速上链；2018年8月，京东区块链开放平台正式发布，该平台可以帮助客户快速构建、管理和使用区块链应用[5-6]。

几乎所有的云计算巨头及区块链创业公司都提供了BaaS平台，但当前的平台有一个显著的同质化的问题，产品功能甚至底层实现方案大同小异，没有明显的差异化。事实上，大多数的云计算公司只是简单地提供了一种开源区块链平台的一键化部署管理方案，没有深入到区块链技术的底层，而区块链公司关注更多的是区块链技术本身，这就导致目前的BaaS底层仍有许多改进和优化的地方。另外，BaaS研发的门槛较高，只有大型公司和高收入的公司有能力负担，无论国内外，BaaS几乎都是由商业巨头把控，行业的马太效应明显[7]。商业BaaS平台虽然提供了对开源区块链平台的支持，但BaaS本身是闭源的，对于BaaS发展和需要构建自己私有BaaS平台的企业来说都是不友好的。

针对上述问题，本文选择基于开源的Hyperledger Fabric联盟链和Kubernetes容器编排技术研究构建BaaS的底层核心技术，通过对Fabric源代码的解析和改造更好地适配Kubernetes。此外学术界和产业界也在积极探索Serverless Function新技术和区块链的融合应用[8-9]，本文跟踪了相关内容，并提出一种基于函数计算思想改造Fabric的BaaS平台方案。

1　相关技术

1.1　BaaS核心技术

BaaS实质上是一种利用各种云计算技术打包封装指定区块链平台并对外提供一键部署和管理区块链能力的云服务。BaaS有PaaS和SaaS两种类型，PaaS提供部署指定区块链平台的能力，SaaS则可以直接提供某种类型的区块链应用服务[10-11]。PaaS类型的区块链服务是业界当前主流的区块链云化方案。在区块链服务层，本文调研了一些主流云计算公司的BaaS平台对区块链的支持情况，详细情况如表1所示。

表1　主要公有云的BaaS平台

Tab. 1　BaaS platforms for major public clouds

BaaS平台	区块链平台
AWS区块链服务	Hyperledger Fabric，Ethereum
Azure区块链服务	Ethereum
Google Cloud Platform	不支持
IBM区块链即服务	Hyperledger Fabric
阿里云区块链服务	Hyperledger Fabric，蚂蚁区块链，Ethereum
腾讯云区块链服务TBaaS	Hyperledger Fabric，FISCO BCOS，Tencent TrustSQL
华为云区块链服务BCS	Hyperledger Fabric

1.2　Hyperledger Fabric

Hyperledger Fabric（简称Fabric）是一个带有准入机制的企业级联盟链项目，它的前身是IBM的OpenBlockchain项目。目前Fabric已经成为企业区块链平台的典范，也是绝大多数厂商构建BaaS平台的首要选择。Fabric是一种许可网络，而不是在没有身份的公共网络中建立分散的信任；Fabric各参与方可以仅仅将需要共享的数据分享给他人，实现保密交易；Fabric采用可插拔的架构设计，各行各业可以针对特定需求量身定制各种插件[12]。

1.3　Kubernetes容器编排技术

Kubernetes是一个用于自动部署、扩展和管理容器的开源平台，吸收了Borg系统的许多功能特性[13]。Kubernetes字面意思为“舵手”，其图标对应为一个方向舵，另外Kubernetes单词中间有8个字母故也简称K8s。Google在大规模工作负载的运维管理上经历十几年先后构建了Borg、Omega以及Kubernetes等系统，并在2014年开源了Kubernetes系统。2017年Docker公司宣布在其核心产品中内置Kubernetes服务，2018年Kubernetes和容器成为所有云厂商的既定标准，以“云”为核心的软件研发思想逐步形成[14]。

1.4　Serverless Function技术

Serverless本意是指一种不需要专门服务器的技术，例如对等网络（Peer-to-Peer，P2P）无需设置服务器，因为每个节点既是服务器又是客户端；但在云环境下Serverless已经转变为一种开发人员不必关心服务器的设计思想、架构模式[15]。Serverless有两种类型：1）后端即服务（Backend as a Service，BaaS），主要是指承载数据存储的服务，例如消息队列、内容分发网络、云对象存储等；2）函数即服务（Function as a Service，FaaS），是指一种将用户自定义代码运行在无状态的计算容器中并由事件触发的计算服务[16]。

2　核心系统概要设计

2.1　系统功能

基于相关技术需求，设计了核心系统的相关功能，如表2所示。系统功能主要分为底层云计算平台、Fabric静态组件部署管理、动态组件链码部署管理以及基于函数计算思想改造链码管理几个项目，每个项目下又划分了若干个功能点。

底层云计算平台是区块链网络的实际运行环境，它提供了计算、存储等资源。计算资源主要由Kubernetes平台管理，存储资源由分布式文件系统提供。作为底层支撑平台，它必

须满足高可用的特性，保证在部分节点失效的情况下仍可有效提供服务，并能通过一定的运维手段恢复失效节点。上层区块链平台数据不断增长的存储需求，要求底层的存储资源能够动态扩展，本文使用分布式文件系统管理存储资源，通过增加存储设备实现存储资源的在线扩展。存储资源管理系统独立于容器计算平台，这使得底层平台还要为 Kubernetes 提供一个存储接入的功能，从而方便地为上层业务提供持久化存储。

BaaS 平台一个基本的要求就是将区块链平台部署在云计算平台上。Fabric 组件根据不同的启动时间，可以分为静态组件和动态组件两部分。静态组件指的是证书颁发机构（Certificate Authority，CA）、Peer、Orderer 以及共识算法所依赖的外部组件，这些组件随区块链网络的启动而启动。Fabric 动态组件链码和其他组件的不同点在于它是通过实例化命令启动的。Fabric 项目提供了 Docker-Compose 的部署样例，只需做适当修改即可将静态组件部署到 Kubernetes 上。在实验环境中，链码和 Peer 服务部署在同一个宿主机上，链码镜像的编译、存储在同一个 Docker 服务中，可以方便地安装与实例化链码。但是在云计算环境下，链码可能会被调度到不同的计算节点上，所以跨节点实例化链码需要构建链码镜像的分发能力，实现链码镜像从 Peer 节点到部署节点的传输。最后一个项目仍是针对链码的，链码的函数计算改造，主要涉及对 Fabric 链码原有生命周期的改造，使其在调用前启动容器，调用完成即退出，让出计算资源。

表 2　核心系统功能

Tab. 2　Core system functions

项目名称	功能细节
底层云计算平台	高可用 Kubernetes 平台 Kubernetes 接入网络存储
Fabric 静态组件部署管理	Peer、Orderer、CA 组件部署、共识算法依赖组件
动态组件链码部署管理	Kubernetes 部署管理链码 链码镜像存储、分发功能
函数计算思想改造	链码生命周期改造 链码容器编译功能 容器镜像分发功能

2.2　系统架构

BaaS 底层的区块链平台云化管理，可以分为云计算和区块链两层结构。本文所涉及的核心系统同样可以分为两层，底层为高可用 Kubernetes 平台和分布式文件系统，上层为 Fabric 区块链平台，系统架构如图 1 所示。

底层高可用 Kubernetes 系统是上层业务的托管平台，承载了上层所有的服务压力。作为一个容器管理平台，Kubernetes 要为上层 Fabric 提供高效的容器编排管理能力，将应用组件合理地分散到计算集群中健康的节点上。Kubernetes 具有高效的服务治理能力，利用系统提供的应用容器健康检测功能，可以选择相关策略自动恢复失效组件。凭借这些能力，开发者可以在 Kubernetes 平台上很方便地实施 Fabric 相关组件的高可用管理。同时，Kubernetes 的高可用也保证了在部分控制节点失效的情况下，仍能提供容器编排能力，确保上层服务的可靠运行。

区块链网络层位于高可用 Kubernetes 平台之上，这一层完成区块链所有的业务逻辑。Fabric 有静态组件和动态链码两部分：静态是指相关组件在部署网络时启动；动态主要指链码在区块链业务运行阶段实例化时启动。动态和静态的划分，关系到 Fabric 网络在 Kubernetes 上的部署设计。在链码管理上，该架构增加了函数计算管理模块，改造后的链码容器共享计算节点，调用任务结束立即退出容器，让出计算资源。

该架构上还有一个独立于 Kubernetes 计算集群的分布式文件系统（Distributed File System，DFS）集群。以 HDFS（Hadoop Distributed File System）为例介绍外部存储。HDFS 是一个分布式文件系统，它提供一种高可靠的、可动态扩展的数据存储能力。HDFS 接入到 Kubernetes 系统，可以为上层的区块链服务提供可动态扩展存储空间的持久化存储服务。Fabric 的账本数据最终存储在了可靠的分布式文件系统上。

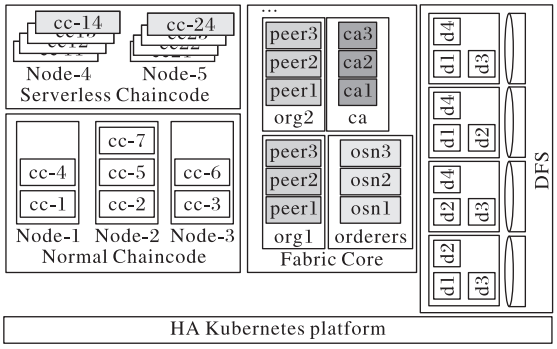

图 1　BaaS 底层架构

Fig. 1　BaaS underlying architecture

2.3　系统流程

本文在链码部署管理上引入了函数计算思想，改变了链码容器原有的生命周期，部署流程如图 2 所示。引入函数计算的系统在链码实例化消息发出后，Fabric 静态组件 Peer 会完成链码容器镜像的编译，并将镜像上传到存储中心，但不会启动链码容器。函数计算下的链码容器是按需启动的，在区块链系统发起一笔交易产生链码调用时，才会触发链码容器的启动。链码容器在完成计算返回结果后，会立即结束其生命周期让出计算资源。

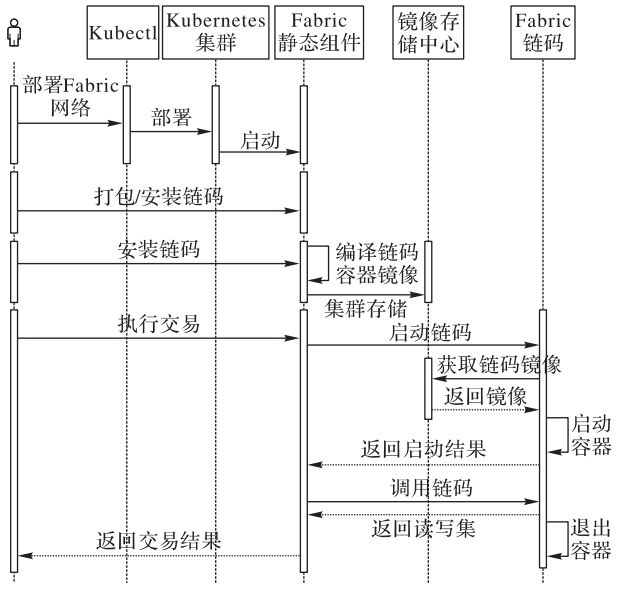

图 2　引入函数计算的 Fabric 链码部署流程

Fig. 2　Fabric chaincode deployment process with function calculation

3　核心系统具体实现

3.1　高可用架构设计

Kubernetes 分为 Master 和 Node(Slave)两个部分,其中 Node的高可用由系统本身完成,保证将 Pod 调度到健康的工作节点上。Kubernetes高可用主要是指 Master组件的高可用。本文所设计的高可用架构如图3所示,可以发现集群外还增加了一个高可用负载均衡器,这个均衡器主要承担 Apiserver 的高可用任务。整个架构按照集群内外分为两部分:集群内多Master节点的高可用和集群外基于主备切换的负载均衡器高可用。Kubernetes 上 Kubelet 是通过 Systemctl 服务管理的,除它之外所有组件都可以直接部署在 Kubernetes 上,本文借用这种能力来设计Kubernetes的高可用方案。

Master节点上的Etcd组件本身就是一个基于Raft协议的高可用分布式键值数据库,只需为它配置多个实例即可。Master上的其他组件,根据有无状态分为两种部署方案:无状态组件 Apiserver采用"多实例+负载均衡器"的高可用设计;有状态组件Scheduler和ControllerManager采用"多实例+竞争锁"的高可用设计。集群外的负载均衡器采用"虚拟IP+双节点"的主备设计,保证主代理服务失效后备份服务器能够立刻提供服务。同时负载均衡器还应具备主动健康检测功能,保证将流量分配到健康的服务上。

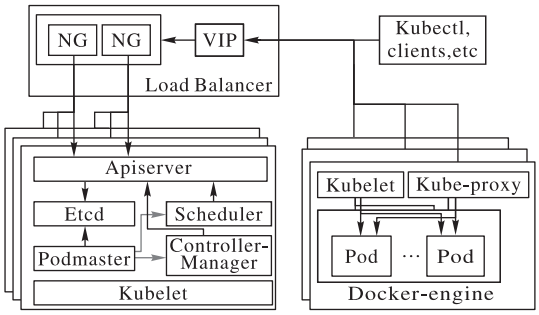

图3　Kubernetes高可用架构

Fig. 3　High availability architecture for Kubernetes

图3中的Apiserver高可用依赖一个外部高可用负载均衡器,本节基于虚拟IP、Keepalived、Nginx等技术设计了一种双节点主备自动切换高可用负载均衡器,系统架构如图4所示。每台物理服务器上都需要安装 Keepalived 和 Nginx 服务。主备服务器上Keepalived需要配置物理网卡、虚拟IP、state、认证信息及优先级,其中主服务器的state设置为MASTER,备份服务器设置为BACKUP。Keepalived 和虚拟IP完成Nginx主备切换的高可用,Keepalived基于虚拟路由冗余协议实现了虚拟IP和服务器的自动绑定,初始状态主节点绑定虚拟IP提供服务。主节点失效,虚拟IP会自动漂移到备份服务器,此时由备份节点提供服务。负载均衡器还需要主动探测代理的Apiserver健康状态,保证将流量转发到可用的服务上。官方版本的Nginx提供被动健康检测方法,即目标服务异常,下次将流量分配到其他服务上,后端Apiserver异常,仍有机会被Nginx转发过来流量。本文基于ngx_healthcheck_module插件为Nginx提供了主动健康检测能力,保证将流量分配到健康的Apiserver上。

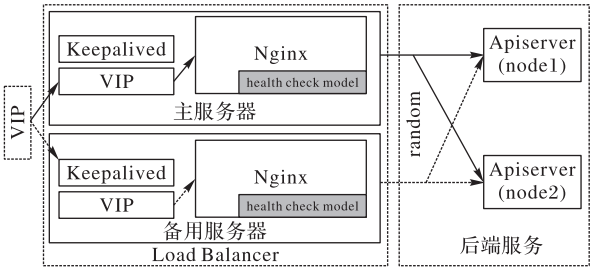

图4　高可用负载均衡器架构

Fig. 4　High-availability load balancer architecture

3.2　Fabric静态组件部署设计

Fabric区块链平台是一个复杂的分布式系统,云化部署复杂度较高。本文将Fabric的组件分为了静态和动态两种,静态组件如CA、Orderer、Peer等构成了分布式系统的主要的基本框架。静态组件在Fabric网络部署时就可以启动完毕(不包括运行中动态加入节点)。图5描述了一个具备两个组织的部署架构,该架构中Fabric所有组件部署在Kubernetes上,所有资源通过Kubernetes进行管理。图5中的O1代表Orderer1。

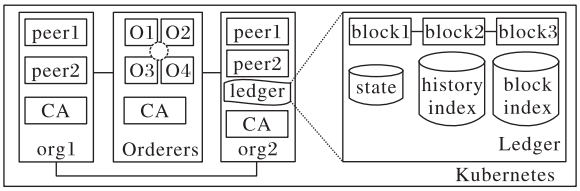

图5　Fabric云化部署架构

Fig. 5　Fabric cloudification deployment architecture

CA是Fabric提供信任的基础,主要提供签发、管理证书以及身份识别等功能。CA是一个C/S架构的服务,Server端可以实现高可用扩展,但是必须共享存储以统一管理身份证书信息。Orderer是Fabric的核心,也是最为复杂的组件之一。Fabric的插件化设计支持多种算法,Orderer是实现插件化共识的基本框架。Peer节点是Fabric执行交易和账本管理的组件。交易执行意味着链码管理,Fabric系统存在系统链码和用户链码两种类型,系统链码有CSCC(Configuration System ChainCode)、LSCC(Lifestyle System ChainCode)、QSCC(Querier System ChainCode)、ESCC(Endorser System ChainCode)、VSCC(Validator System ChainCode)五种分别完成通道配置、链码生命周期管理、交易查询、交易背书、交易验证等功能。

3.3　Fabric链码部署设计

链码是区块链的核心,也是整个部署方案中最具挑战性的地方。将链码纳入到Kubernetes的环境中是最理想的部署方案,但社区和业界的方案均没有实现这一点,这主要是因为Fabric的代码本身没有对这块内容的支持。本节所要介绍的部署方案就是要通过解构、二次开发Fabric代码将链码纳入Kubernetes环境。

Peer服务在链码管理上完成了链码安装、Docker镜像编译、创建容器、启动容器以及健康检测等工作。原生Peer服务,启动时要指定好Docker Daemon服务地址,随后链码实例化过程的镜像编译和创建均在同一个宿主机上进行,但是Kubernetes是一个分布式的计算集群,链码容器会被调度到不同的节点上,也就是获得部署链码容器任务的宿主机可能没有链码镜像。通过链码镜像编译和容器创建分离,镜像编译仍在Peer服务绑定的Docker Daemon中进行,然后再将镜像分发到执行创建任务的节点上。

集群内镜像分发可以构建一个镜像仓库实现，高可用镜像仓库采用的Harbor的主从复制模式构建，同时为了降低集群内的网络带宽占用和提高镜像分发速度，基于Dragonfly构建P2P镜像下载加速器，相较于原生方式，可将容器分发速度提高57倍，网络出口流量降低99.5%[17]。引入镜像仓库后，Peer和部署节点之间增加了向仓库Push和Pull镜像的两步操作，完整的工作流如图6所示。原生Peer通过Docker SDK向Docker Daemon发起镜像编译和创建容器的任务，现在链码实例化分为镜像编译和创建两个过程：镜像编译仍通过Peer完成，但是要将镜像Push到集群内的镜像仓库；容器创建，则需要在Peer中集成Kubernetes的SDK完成容器的部署，获得部署任务的工作节点从远程仓库拉取链码镜像并创建链码实例。

链码部署管理的改造涉及客户端部分：Peer服务端实现了多种链码容器管理插件，部署链码时需要客户端指明使用哪一个插件。Peer客户端提供解析配置文件的形式确定链码部署环境，具体是客户端配置文件的vm.type参数。客户端将链码使用的虚拟执行环境类型存储在了CDS（ChaincodeDeploymentSpec）结构中，而CDS是在链码安装时确定的，所以还需要在链码安装部分加上对Kubernetes的支持。链码部分的改造实现拟全部封装在k8sontroller包中，除了K8sDockerVM接口的实现外，还有如表3所示的一些Kubernetes客户端接口的封装实现。这些函数完成链码部署时的相关资源的创建。

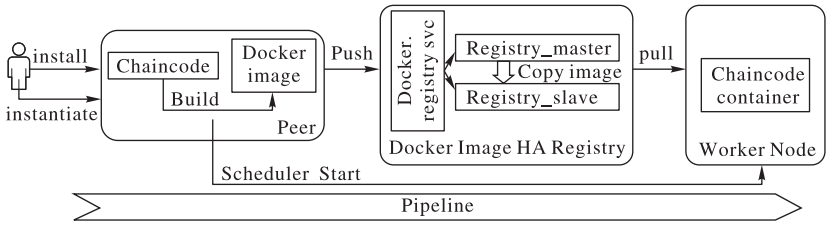

图6 Kubernetes环境中链码部署流

Fig. 6 Chaincode deployment flow in Kubernetes environment

表3 K8sDockerVM中的K8s接口

Tab. 3 K8s interfaces in K8sDockerVM

主要功能	函数名
创建ConfigMap	func (vm*K8sDockerVM)createConfigMap(clients *k8sClient. Clientset,name,ns string,data map[string]string)error
删除ConfigMap	func (vm*K8sDockerVM)deleteConfigMap(clients*k8sClient. Clientset,name,ns string,opts*metav1. DeleteOptions)error
创建Deployment	func (vm*K8sDockerVM)createDeployment(clients*k8sClient. Clientset,deployment* appsv1. Deployment)error
删除Deployment	func (vm*K8sDockerVM)deleteDeployment(clients*k8sClient. Clientset,name,ns string,opts*metav1. DeleteOptions)error
创建Service	func(vm *K8sDockerVM)createSVC(clients *k8sClient. Clientset,svc *apiv1. Service)error
删除Service	func(vm *K8sDockerVM)deleteSVC(clients *k8sClient. Clientset,name,ns string,opts *metav1. DeleteOptions)error

3.4 函数计算管理链码设计

根据区块链和函数计算服务的相似性，链码是Fabric上基于函数计算改造最具潜力的组件。Fabric中一笔交易的完成需要多数节点达成共识，并落到区块中，部分链码容器失效并不影响区块链整体网络。当有新的交易需要执行计算时，Fabric会再次拉起容器。Fabric的这项能力为实现链码容器函数计算化管理提供了可能。Fabric链码是一个只提供计算服务的组件，它的所有数据均来自于Peer服务，这种特性满足函数计算服务对代码无状态性的要求。Fabric链码的生命周期如图7所示，完成链码调用任务需要依次经历链码安装、实例化的过程。Peer服务端的链码容器在实例化后一直处于Running状态等待调用，只有在链码升级或者链码本身内部错误才会结束链码容器。而函数计算服务管理链码，首先要做的就是改变链码的生命周期，要求链码容器在调用时再实例化启动，调用结束即退出。

无论函数计算服务还是区块链服务，部署用户代码都需要解决代码的编译问题。Fabric链码的编译是一个复杂的过程，不能像函数计算服务一样提前准备好代码和依赖包直接上传，目前业界或者社区的函数计算服务无法完成Fabric链码的编译。鉴于实际情况，本文采用模拟函数计算服务的方法进行实验。函数计算服务的用户代码部署可以分为镜像编译、分发、部署等过程，以Knative函数计算服务为例，用户代码被编译成Docker镜像存储在远程镜像仓库中，事件触发计算节点从仓库中拉取镜像并启动实例，完成任务后退出。Fabric Peer本身具备链码镜像的编译能力，本文设计的Kubernetes部署方案也完成了集群内链码镜像的分发，所以只需改造链码实例的调度任务即可。

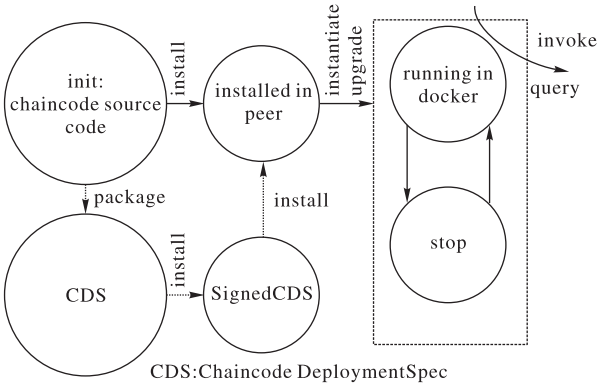

图7 Fabric链码的生命周期

Fig. 7 Fabric chaincode lifecycle

通过在CDS结构中增加了Serverless字段和在Chaincode Install子命令下增加serverless bool全局参数，实现改变链码容器的运行模式和完成调度任务。引入Serverless参数的链码启动流程如图8所示。引入Serverless的链码调用，会增加单次链码调用的处理时间，但它在空闲时让出了计算资源，可以极大地提高资源的利用率，多租户网络共享计算池可以显著降低区块链的部署运维费用，降低用户上链的费用门槛，同时计算资源池对用户是透明的，共享资源的同时又保证了数据隐私性。

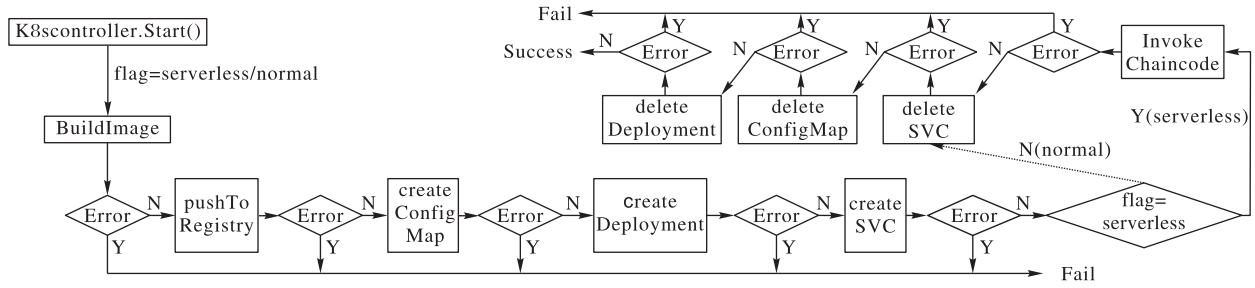

图8　链码启动简化流程

Fig. 8　Simplified flowchart of chaincode activation

3.5　链码性能测试

Kubernetes环境和普通Docker环境中链码在镜像管理和运行环境方面存在区别。本节通过相同任务量下任务完成时间的长短来评估改造后的链码调用性能,测试基准为基于Docker运行环境的链码完成时间,测试环境为双组织单节点部署环境,共识采用Solo算法。

测试方案通过脚本封装链码调用,通过参数指定调用次数,并输出调用开始和结束的时间,每项测试5次,并求取平均值,实验结果如表4~5所示。根据两个表中的数据,可以看出Docker和Kubernetes环境下,Kubernetes部署的链码调用性能跟Docker链码相近。此外,统计了Docker和Kubernetes链码在100次调用下成功调用次数的平均值分别为18.4和19.0,两者调用的成功率也相近。为了不影响记时,测试脚本并未进行Sleep操作等待区块链网络数据的同步,所以链码调用的成功率普遍低一些。

表4　Docker链码调用时间测试　　单位:s

Tab. 4　Docker chaincode call time test　　unit:s

测试次数	调用次数							
	1	2	3	4	5	10	50	100
1	0	1	1	2	2	4	21	44
2	1	1	1	2	2	4	20	42
3	0	1	2	3	3	4	23	49
4	1	1	1	2	3	5	21	44
5	0	1	1	2	2	4	21	43
平均	0.4	1.0	1.2	2.2	2.4	4.2	21.4	44.4

表5　Kubernetes链码调用时间测试　　单位:s

Tab. 5　Kubernetes chaincode call time test　　unit:s

测试次数	调用次数							
	1	2	3	4	5	10	50	100
1	0	1	1	2	2	4	21	44
2	0	0	1	1	2	4	22	43
3	0	1	1	2	3	4	23	49
4	0	0	1	1	2	4	21	42
5	1	1	3	3	4	6	23	45
平均	0.2	0.6	1.4	1.8	2.6	4.4	22	44.6

此外,也对Serverless模式下的链码进行了测试,测试结果如表6所示。函数计算有一个明显的特点是冷启动时间较长,在这种模式下,链码在有调用需求时才会启动容链码容器的创建、启动花费了较长的时间。Serverless链码调用时间较长,为区块链数据同步提供了比较充足的时间,实验测得链码调用成功率要比Docker和Kubernetes高,100次调用下成功调用次数的平均值为68.8。

表6　Serverless链码调用时间测试　　单位:s

Tab. 6　Serverless chaincode call time test　　unit:s

测试次数	调用次数							
	1	2	3	4	5	10	50	100
1	16	32	48	65	73	158	813	1 530
2	7	22	38	46	61	125	785	1 600
3	8	16	31	47	62	125	747	1 596
4	8	24	39	47	62	155	835	1 574
5	7	23	38	69	76	156	766	1 611
平均	9.2	23.4	38.8	54.8	66.8	143.8	789.2	1 582.2

4　结语

基于业界典型的BaaS方案,本文选择了Fabric和Kubernetes两项基本技术研究BaaS的底层实现。在Kubernetes方面,主要研究底层技术设施的高可用性;在Fabric方面,主要研究基于高可用Kubernetes的云化部署技术。

Fabric链码的部署管理是本文研究的核心之一,也是Fabric云化部署方案中最复杂的一部分。现有的链码管理技术均没有将链码纳入Kubernetes环境中。本文通过对Fabric Peer的二次开发,通过新增一个K8sDockerVM控制器插件,实现了对Kubernetes在代码级别上的支持。此外,本文也通过函数计算思想对链码运行机制进行了一个创新尝试,实现了链码实例执行模式的改变,将实例化后链码容器常驻等待调用改变为有调用需求时启动。

函数计算引入区块链网络,可以为它带来几个能力:链码高度弹性部署,链码按需部署,用完即释放资源,可以为用户大幅降低费用。但函数计算服务的冷启动时间较长,由Serverless链码调用性能测试结果也可以看到,调用时间比常规部署方式的链码调用要久。对区块链网络来说,混合部署模式可能更加合适:高频应用,链码常驻提供高效的服务;低频应用,按Serverless模式部署,按需调用降低费用。混合部署及高频、低频应用区分方案是接下来区块链云化部署的研究方向。

参考文献(References)

[1] 可信区块链推进计划. 区块链即服务平台BaaS白皮书(1.0版)[R/OL]. [2019-11-17]. http://www.trustedblockchain.cn/schedule/detail/2989.(Trusted Blockchain Initiatives. Blockchain-as-a-Service (BaaS) platform white paper (Version 1.0)[R/OL]. [2019-11-17]. http://www.trustedblockchain.cn/schedule/detail/2989.)

[2] ONIK M M H, MIRAZ M H. Performance analytical comparison of Blockchain-as-a-Service (BaaS) platforms[C]// Proceedings of the 2019 International Conference for Emerging Technologies in

Computing, LNICST 285. Cham: Springer, 2019: 3-18.
[3] 刘楠,刘露. 区块链与云计算融合发展BaaS成大势所趋[J]. 通信世界, 2017(17): 61-62. (LIU N, LIU L. Blockchain and cloud computing integration development BaaS becomes a major trend[J]. Communications World, 2017(17): 61-62.)
[4] 才丽. 面向BaaS平台的资源调度算法研究与实现[D]. 杭州:浙江大学, 2018: 2-4. (CAI L. The BaaS platform orientated research and implementation of scheduling algorithm [D]. Hangzhou: Zhejiang University, 2018: 2-4.)
[5] 工业和信息化部信息中心. 2018年中国区块链白皮书[R/OL]. [2019-11-17]. http://www.miit.gov.cn/n1146290/n1146402/n1146445/c6180238/part/6180297.pdf. (Information Center of Ministry of Industry and Information Technology. China blockchain white paper in 2018[R/OL]. [2019-11-17]. http://www.miit.gov.cn/n1146290/n1146402/n1146445/c6180238/part/6180297.pdf.)
[6] 刘曦子. 2019年中国区块链发展形势展望[J]. 网络空间安全, 2019, 10(1): 31-35. (LIU X Z. Prospects for the development of blockchain situation in 2019 [J]. Information Security and Technology, 2019, 10(1): 31-35.)
[7] 郝方舟,李雪婷,孔繁星. 2018年BaaS(区块链即服务)平台研究报告[R/OL]. [2019-11-17]. https://www.odaily.com/post/5135837. (HAO F Z, LI X T, KONG F X. BaaS (Blockchain as a Service) platform research report in 2018[R/OL]. [2019-11-17]. https://www.odaily.com/post/5135837.)
[8] CHEN H, ZHANG L. FBaaS: functional blockchain as a service [C]// Proceedings of the 2018 International Conference on Blockchain, LNCS 10974. Cham: Springer, 2018: 243-250.
[9] 段红. 一种基于区块链的高性能FAAS系统: 201811430601.0[P]. 2019-03-22. (DUAN H. A high-performance FAAS system based on blockchain: 201811430601.0[P]. 2019-03-22.)
[10] SINGH J, MICHELS J D. Blockchain as a Service (BaaS): providers and trust[C]// Proceedings of the 2018 IEEE European Symposium on Security and Privacy Workshops. Piscataway: IEEE, 2018: 67-74.
[11] WAN Z, CAI M, YANG J, et al. A novel blockchain as a service paradigm[C]// Proceedings of the 2018 International Conference on Blockchain, LNCS 10974. Cham: Springer, 2018: 267-273.
[12] CACHIN C. Architecture of the Hyperledger blockchain Fabric [EB/OL]. [2020-01-16]. https://www.zurich.ibm.com/dccl/papers/cachin_dccl.pdf.
[13] OLIVEIRA C, LUNG L C, NETTO H, et al. Evaluating Raft in docker on Kubernetes[C]// Proceedings of the 2016 International Conference on Systems Science, AISC 539. Cham: Springer, 2016: 123-130.
[14] 林琳. 为什么说2019年是云原生的关键节点[J]. 计算机与网络, 2019, 45(19): 38-40. (LIN L. Why 2019 is a critical point for cloud native [J]. Computer and Network, 2019, 45(19): 38-40.)
[15] FOX G C, ISHAKLAN V, MUTHUAMY V, et al. Status of serverless computing and Function-as-a-Service (FaaS) in industry and research[R/OL]. [2019-09-10]. https://arxiv.org/ftp/arxiv/papers/1708/1708.08028.pdf.
[16] 刘畅,毋涛,徐雷. 基于无服务器架构的边缘AI计算平台[J]. 信息通信技术, 2018, 12(5): 45-49. (LIU C, WU T, XU L. Edge AI computing platform based on serverless architecture[J]. Information and Communications Technologies, 2018, 12(5): 45-49.)
[17] dragonflyoss. 什么是Dragonfly[EB/OL]. [2020-05-19]. https://d7y.io/zh-cn/docs/overview/what_is_dragonfly.html. (dragonflyoss. What is Dragonfly[EB/OL]. [2020-05-19]. https://d7y.io/zh-cn/docs/overview/what_is_dragonfly.html.)

This work is partially supported by the General Research Project of the Department of Education of Zhejiang Province (Y202044224), the Zhejiang Graduate Education Association Project (2020-002), the Zhejiang Cultural Relic Protection Science and Technology Project (2019009).

LIU Hongyu, born in 1994, M. S. His research interests include container cloud, blockchain.

LIANG Xiubo, born in 1983, Ph. D., associate research fellow. His research interests include blockchain, artificial intelligence, natural human-computer interaction, mobile internet.

WU Junhan, born in 1998, M. S. candidate. His research interests include blockchain.