阿里云 容器服务Kubernetes版

解决方案

文档版本: 20190912

为了无法计算的价值 | [-] 阿里云

<u>法律声明</u>

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读 或使用本文档,您的阅读或使用行为将被视为对本声明全部内容的认可。

- 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档,且仅能用于自身的合法 合规的业务活动。本文档的内容视为阿里云的保密信息,您应当严格遵守保密义务;未经阿里云 事先书面同意,您不得向任何第三方披露本手册内容或提供给任何第三方使用。
- 未经阿里云事先书面许可,任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分 或全部,不得以任何方式或途径进行传播和宣传。
- 3. 由于产品版本升级、调整或其他原因,本文档内容有可能变更。阿里云保留在没有任何通知或者 提示下对本文档的内容进行修改的权利,并在阿里云授权通道中不时发布更新后的用户文档。您 应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
- 4. 本文档仅作为用户使用阿里云产品及服务的参考性指引,阿里云以产品及服务的"现状"、"有缺陷"和"当前功能"的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引,但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的,阿里云不承担任何法律责任。在任何情况下,阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害,包括用户使用或信赖本文档而遭受的利润损失,承担责任(即使阿里云已被告知该等损失的可能性)。
- 5. 阿里云网站上所有内容,包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计,均由阿里云和/或其关联公司依法拥有其知识产权,包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意,任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外,未经阿里云事先书面同意,任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称(包括但不限于单独为或以组合形式包含"阿里云"、Aliyun"、"万网"等阿里云和/或其关联公司品牌,上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司)。
- 6. 如若发现本文档存在任何错误,请与阿里云取得直接联系。

通用约定

格式	说明	样例
•	该类警示信息将导致系统重大变更甚至 故障,或者导致人身伤害等结果。	禁止: 重置操作将丢失用户配置数据。
A	该类警示信息可能导致系统重大变更甚 至故障,或者导致人身伤害等结果。	▲ 警告: 重启操作将导致业务中断,恢复业务所需 时间约10分钟。
Ê	用于补充说明、最佳实践、窍门等,不 是用户必须了解的内容。	道 说明: 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 确定。
courier 字体	命令。	执行 cd /d C:/windows 命令,进 入Windows系统文件夹。
##	表示参数、变量。	bae log listinstanceid Instance_ID
[]或者[a b]	表示可选项,至多选择一个。	ipconfig [-all -t]
{}或者{a b }	表示必选项,至多选择一个。	<pre>swich {stand slave}</pre>

目录

法律声明I
通用约定I
1 区块链解决方案1
1.1 概述
1.2 版本历史和升级注意事项1
1.3 快速开始
1.4 环境准备13
1.5 访问区块链解决方案主页15
1.6 配置部署区块链16
1.7 查看区块链网络日志25
1.8 访问区块链概述
1.9 配置公网IP和外部端口34
1.10 以CLI方式访问区块链36
1.11 以应用程序方式访问区块链
1.12 以管理监控工具方式访问区块链 39
1.13 清理区块链环境
1.14 区块链网络重启和数据恢复 42
2 深度学习解决方案44
2.1 基于helm chart的深度学习解决方案 44
2.1.1 概述
2.1.2 创建 GN5 型Kubernetes 集群45
2.1.3 TensorFlow模型预测48
2.1.4 TensorFlow 分布式模型训练53
2.1.5 创建 NAS 数据卷59
2.1.6 TensorFlow模型开发62

1区块链解决方案

1.1 概述

阿里云容器服务区块链解决方案为基于 Hyperledger Fabric 区块链应用和解决方案的开发、测试 提供了一种简便和灵活的通用型基础解决方案。

针对区块链配置部署所要求的专业技能较高、过程繁琐耗时且容易出错等问题,该解决方案为开 发者提供了图形化的区块链网络配置向导,开发者只需填写关键配置参数,通过一键自动化配置 部署功能,便可在数分钟内完成复杂的配置文件生成、以及在容器集群上创建基于 Hyperledger Fabric 的区块链网络。此外,为方便开发者使用区块链应用程序和区块链管理监控工具访问区块链 网络,该解决方案提供了预生成的适配应用和工具的配置文件,方便开发者一键下载以提供给区块 链应用和工具使用。

阿里云容器服务区块链解决方案具备以下特性:

- ・标准:支持 Linux 基金会开源区块链技术 Hyperledger Fabric V1.4 主要功能,包括 Peer、 CouchDB、Orderer、Kafka 和 ZooKeeper、CA 等标准区块链节点类型。
- ・简便: 图形化向导, 简化区块链配置, 屏蔽底层复杂的参数设置、工具调用、配置分发等过 程, 显著降低出错概率。同时支持内置部署Hyperledger Blockchain Explorer。
- ・成熟:底层阿里云容器服务集群历经多年企业级大规模生产应用的检验,稳定可靠,并集成了云端应用的最佳实践和 Docker 技术研发的专业经验。
- ・丰富:可无缝对接阿里云强大和丰富的企业级应用服务能力,例如存储、网络、路由、安全、镜像、日志、监控等等,为区块链上层应用提供全方位的支持。未来区块链解决方案将全面涵盖公共云、专有云、混合云等多种部署形态。

阿里云容器服务区块链解决方案目前为公开测试阶段,我们欢迎感兴趣的阿里云用户进行试用以及 提供宝贵的意见和建议。我们将根据市场和客户的需求不断改进和丰富该解决方案,帮助开发者和 合作伙伴打造出更多更好的业务创新应用和行业解决方案。

1.2 版本历史和升级注意事项

本文档记录了阿里云容器服务区块链解决方案的版本历史和对应的功能变更,以及升级过程中的注 意事项。版本号为应用目录中区块链解决方案ack-hyperledger-fabric的chart版本号。

版本历史

版本 0.2.3

- Hyperledger Fabric 升级到1.4.0
- · Explorer 升级到0.3.8

版本 0.2.2

· 提升NAS文件系统挂载方式的高可用性

▋ 说明:

从此版本开始,需在每次创建区块链网络时输入NAS挂载地址,不再需要在环境准备环节 将NAS挂载到ECS。

- ・提升区块链节点部署的高可用性
- · 区块链日志级别可参数化配置
- ·提升数据清理、Explorer启动、Chaincode执行超时等方面的稳定性
- · 统一部分Pod的命名规范

版本 0.2.1

- · Chart更名为ack-hyperledger-fabric
- ・ Bug修复

版本 0.2.0

- ・支持Hyperledger Fabric 1.1.0
 - 支持Node.js类型的chaincode功能以及相关示例
 - 支持chaincode级别的账本数据加密功能以及相关示例
 - 支持connection profile
 - 代码级别优化以进一步提升性能和水平扩展性
 - 其他1.1.0新功能的支持
- ・集成阿里云日志服务
- · 内置部署Hyperledger Blockchain Explorer
- · 支持阿里云神龙弹性裸金属实例
- · 优化区块链网络删除过程的数据目录清理

版本 0.1.0

· 支持在阿里云容器服务Kubernetes集群上部署Hyperledger Fabric 1.0.0

升级注意事项

·从容器服务控制台应用目录界面安装的区块链解决方案为当前最新版本

・ 对于使用命令行 helm install 安装的方式,因为helm repo在本地可能有老版本的缓存(如 过去曾经安装过区块链解决方案),所以可以通过以下命令查看当前版本:

helm search hyperledger

如需更新本地repo缓存,可运行以下命令,以获得最新版的区块链解决方案:

helm repo update

1.3 快速开始

本文档提供了从零开始实现区块链部署的快速入门指南,其中大部分设置均采用默认值或者示例 值。如需了解更全面的配置方式,请参阅后续章节的详细介绍。

限制条件

- ・需要注册账号, 开通容器服务, 用户账户需有 100 元的余额并通过实名认证, 否则无法创建按 量付费的 ECS 实例和负载均衡。
- · 容器集群和 NAS 文件系统必须位于相同的地域,请确保所选地域能同时支持 kubernetes 集群 模式和 NAS 文件系统(取两者交集)。关于 NAS 文件系统支持地域列表请参见 产品与服务 > 文件存储 > NAS > 文件系统列表。
- · 文件系统的使用方式仅适用于区块链相关应用和解决方案的开发、测试阶段,如需在生产环境部 署,请联系我们进一步探讨具体的业务和技术需求,以共同决定最适合的方式。

步骤1创建Kubernetes集群

- 1. 登录 容器服务控制台。
- 2. 在 Kubernetes 菜单下,单击左侧导航中的 集群,再单击右上角的 创建 Kubernetes 集群。

容器服务 - Kubernetes ▼	集群列表	您最多可以创建 5 个集群,每个约	東群最多可以添加 40 个节点 刷新	申请使用Serverless Kubernetes集群	创建 Kubernetes 集群 🗸
概览	常见问题: 🔗 如何创建集群	⊗ 扩容和缩容集群 🛛 ⊗ 通过 kubecti 连接 Kubernetes 集群	第 🔗 通过命令管理应用		2
▼ 集群	名称 ▼				
集群	集群名称/ID	集群类型 地域 (全部) 👻 网络类型	集群状态 创建时间	Kubernetes 版本	攝作

- 3. 设置集群的基本信息。本示例相关配置如下:
 - · 输入集群名称,例如: k8s-blockchain。
 - ・地域:选择 华东1。
 - ・可用区:选择 华东1可用区A。
 - · 网络类型:单击 专有网络。如没有现成的 VPC 专有网络,则单击 自动创建。
 - ・ 设置 登录密码 和 确认密码。
 - · 设置实例规格和数量。区块链网络部署需要占用较多资源,推荐使用默认配置。
 - · SSH登录:勾选开放公网SSH登录。
 - ・最后单击 创建集群。等待集群创建完成(约需要数分钟)。
- 4. 返回集群列表页面, 选择刚创建的 k8s-blockchain 集群, 单击 管理。

容器服务 - Kubernetes ▼	集群列表	您	最多可以创建 5	个集群,每个集群最多可	以添加 40 个	节点 刷新 申请	使用Serverless Kubernetes集群	创建	Kubernetes 集	群 🔸
概览	常见问题: 8 如何创建集群 8 扩容和	宿容集群 🔗 通	过 kubectl 连接 K	(ubernetes 集群 🛛 Ø 通)	立命令管理应用	Ð		集群名称	你更新成功	
▼ ^{集群}	名称 ▼									
集群	集群名称/ID	集群类型	地域 (全部) 👻	网络类型	集群状态	创建时间	Kubernetes 版本	2		操作
节点	k8s-blockchain	Kubernetes	华东1	虚拟专有网络 vpc-bp1kd7yn4qn	●运行中	2018-07-03 09:55:34	1.10.4	管理	查看日志 集群伸縮	控制台 更多▼
存储										

5. 在集群的基本信息页面 ,记录 Master 节点 SSH 连接地址 (此为公网地址)。

<	集群:k8s-blockchain										
基本信息	基本信息										
控制台	集群ID:	虚拟专有网络	●运行中	地域: 华东1	节点个数 4	集群伸缩					
	连接信息										
	API Server 公网连接通点										
	API Server 内网连接端点	API Server 内网连接骑兵									
	Master 节点 SSH 连接地址	10.000.00									

6. 在节点列表页面,记录当前集群对应的节点(Master 和 Worker)的 IP 地址(此为内网地址)。

容器服务 - Kubernetes ▼	节点列表							刷新	标签管理	集群伸缩	添加已有节点
概党	常见问题: 🖉 排										
▼ 集群 1	集群: k8s-blo	ckchain 🔻	标签过滤 ▼								
集群	IP地址	角色	实例ID/名称	配置	容器组(已分配量)	CPU (请求量 使用量)	内存 (请求量 使用量)	更新时	间	操作	
节点 2		Master	10010-004	按量付费 ecs.n4.xlarge	8	21.25% %	2.56% %	2018-0	06-27 18:01:36	详情 监控 :	修除 调度设置
存储卷命名空间		Master	10000	按量付费 ecs.n4.xlarge	7	21.25% %	2.56% %	2018-0	06-27 18:01:36	详情 监控 :	修除 调度设置
▼ 应用	1.00	Master	20120	按量付费 ecs.n4.xlarge	9	27.75% %	3.96% %	2018-0	06-27 18:01:36	详情 监控 :	修除 调度设置
部署容器组		Worker	1000	按量付费 ecs.n4.xlarge	10	2.75% %	2.81% %	2018-0	06-27 18:01:37	详情 监控 :	修除 调度设置

步骤 2 为 Worker 节点绑定弹性公网 IP

- 1. 登录 IP 管理控制台。
- 2. 在左侧导航栏,单击弹性公网 IP,然后单击右上角的 申请弹性公网IP。

- 3. 选择 k8s-blockchain 集群所在地域,对于其他设置,根据需要自行选择,然后单击 立即购买。
- 4. 开通完成后,等上述购买的 IP 实例出现在列表后,单击 绑定。

IP管理控制台	弹性公网IP列表 华北1 华北2 华北3	华北5 华东1 华东2	华南 1 香港 亚太东北 1 (东	(新加坡) 亚太东南 1 (新加坡)	亚太东南 2 (悉尼) 亚太东南3 (吉隆	城)				
弹性公网IP	美国东部1(弗吉尼亚)	第四百部1(硅谷) 中东东部1	(迪拜) 欧洲中部 1 (法兰克福)			€ 刷新	申请弹性公网IP			
	弹性公网IP地址 Y 请输入弹性公网IP地址进	推公网IP地址 ▼ 诺金入弹性公网IP地址进行精确查词 證素								
	□ 实例 ID IP地址	监控 带宽	付费类型(全部) ▼	状态(全部) ▼ 共享帯宽	绑定实例	实例类型	操作			
	eip-bp1ihxg830f9qk674n8xa	▶ 按使用流量计费 1Mbps	接量付费 2017-12-25 16:35:30 创建	可用 -		-	绑定 解绑 更多操作▼			

- 5. 在 ECS 实例下拉列表中,选择一个 worker 节点(名称是以 node 结尾的实例),单击 确定。
- 6. 完成绑定后,回到弹性公网 IP 列表,记录上述新建实例的 IP 地址。

步骤3创建文件系统和添加挂载点

- 1. 登录 文件存储 NAS 控制台。
- 2. 在页面顶部地域列表中, 单击 华东1。单击右上角的 创建文件系统。



3. 在弹出的 创建文件系统 对话框进行配置。

创建文件系统		×
SSD性能型文件系统存	储容量上限1PB,容量型文件系统存储容量上限10PB。	
* 地域:	华东 1	
* 存储类型:	容量型 ◆	
*协议类型:	NFS(包含NFSv3和NFSv4) ◆	
* 可用区:	华东1可用区B ● 同一地域不同可用区之间文件系统与计 算节点互通	
存储包:	不绑定 绑定一个现有空闲存储包,没有则不绑 定	
	确定工具	2消

- · 地域:选择华东1。选择与容器集群相同的地域。
- ・存储类型:本示例选择 容量型。
- ・协议类型:选择 NFS。
- ·可用区:选择华东1可用区B。同一地域不同可用区可以互通。
- ・ 单击 确定。
- 4. 单击 点击前往。



5. 配置 NAS 存储包。

NA	S存储包								
	区域	华东1 亚太东南 1 (新加坡)	华北2 华北3	华东2 欧洲中部 1 (法兰克福)	华南1 香港	华北1 美国东部 1 (弗吉尼亚)	华北5	当前配置 区域: 文件系统ID: 句	华东1 创建新文件系统并绑定存储
悪犯る	文件系统ID 存储类型	创建新文件系统并绑定存 存储包必须与一个文件系 新文件系统并绑定存储包 SSD性能型	存储包 ▼ 统绑定,且一个文件 容量型	系统同一时刻只能绑定一 NASPlus智能缓存型	-个存储包,您可以在:	此选择一个已经存在的文	件系统ID或选择【创建	存储类型: 协议类型: 容量: 可用区: 购买时长:	容量型 NFS 500GB 华东1可用区 B 1个月
基本	协议类型容量	性能型、容量型和NAS Pi NFS 500GB	us智能缓存型之间的 SMB 1TB	6区别,请点击查看:【名 5TB	序储类型选择】 10TB	ЗОТВ	50TB	▲ 0.0 ¥ 0.0 省¥ 150.00	0
	可用区	100TB 华东 1 可用区 B	200TB					立即购到	χ
购买量	购买时长	1个月	6个月	1年					

- ・存储类型:本示例选择 容量型。
- ·购买时长:本示例选择1个月,请根据实际需要选择。
- ・ 单击 立即购买。
- 6. 如果在创建 Kubernetes 集群的时候,选择的是 使用已有 专有网络,请忽略本步骤,继续下 一步;如果在创建 Kubernetes 集群的时候,选择的是 自动创建 专有网络,请前往 VPC控制 台,将 VPC 网络名称修改为容易识别的标志,例如, blockchain_huadong1。

创建专有网络	刷新	自定义					实例名称	✓ 请输入名称或Ⅱ)进行精确查询	Q
实例ID/名称			目标网段	状态	默认专有网络	路由表	Š	を换机	操作	
vpc-2ze1w19wawvgya blockchain	dhkyok0		192.168.0.0/16	● 可用	否	1	1		管理 删除	

7. 在新创建的文件系统的操作列表中,单击添加挂载点。在弹出的添加挂载点对话框中进行配置。

添加挂载点		×
挂载点是云服务器 每个挂载点必须与	访问文件系统的入口,当前支持专有网络和经典网络挂载点, ;一个权限组绑定。	
Linux客户端在默认可以 <u>参考该文档</u> 进	人情况下对NFS请求并发数做了限制,若遇到性能较差的情况, 行配置。	
文件系统ID:	035bb49518	
* 挂载点类型:	专有网络	
* VPC网络:	vpc-bp1vvet9csb16vac2jqc ▼ 点击前往VPC控制创建VPC网络	
* 交换机:	vsw-bp1ikhyb5ogzdblu1xci 🔻	
* 权限组:	VPC默认权限组(全部允许 ▼	
	确定取消	

- · 挂载点类型:选择 专有网络。
- · VPC网络:选择在创建容器集群环节的 VPC 网络。
- · 交换机:选择在创建容器集群环节的虚拟交换机。
- · 权限组:选择 VPC 默认权限组。
- ・ 単击 确定。
- 8. 在文件系统的操作列表中, 单击 管理。在文件系统详情页面记录 挂载地址。

挂载点						添加挂载点	^
挂载点类型◆	VPC 🗢	交换机◆	挂载地址◆	权限组	状态◆		操作
专有网络	vpc- bp1vvet9csb16vac2jqoc	vsw- bp1ikhyb5ogzdblu1xci8	1518-pja63.cn-hangzhou.nas.aliyuncs.com	VPC默认权限组 (全部允许)	可用	修改权限组 激活 禁用	删除

步骤4配置部署区块链网络

1. 在 Kubernetes 菜单下,单击左侧导航中的市场 > 应用目录。在右侧列表区域找到 ackhyperledger-fabric,并单击该区域。



2. 在 ack-hyperledger-fabric 应用界面右侧区域,选择部署集群,并填写发布名称。

应用目录 - ack-hyperledger-fabric	
Ack-hyperledger-fabric Incubator Hyperledger Fabric Heim chart for Kubernetes on Albaba Goud Container Service	
· 伊朗 参数	创建
Hyperledger Fabric on Kubernetes of Alibaba Cloud Container Service	
Hyperledger Fabric is one of the most popular blockchain infrastructures in the world, which is open sourced and hosted by Linux Foundation.	仅支持 Kubernetes 版本 1.8.4 及以上的集群。对于 1.8.1 版本的集群,您可以在集群列 表中进行 集群升级 操作。不支持ServerlessKubernetes集群。
Introduction	集群 k8s-blockchain
This chart implements a solution of automatic configuration and deployment for Hyperledger Fabric. The solution is deployed on Kubernetes cluster of Alibaba Cloud Container Service. And the Hyperledger Fabric network can be accessed by CLL applications and explorers within or outside the Kubernetes cluster. Blockchain Explorer is now integrated inter the backing report.	命容空间 default ・
A NAS (NFS protocol) shared file storage is needed for: 1. distribution of crypto and configurations; 2. data persistence for most services.	发布名称 ack-hyperledger-fabric-default
Currently v1.4.0 of Hyperledger Fabric is supported.	台湾
You can also refer to the documentation for blockchain solution of Alibaba Cloud Container Service.	
Prerequisites	[
A Kubernetes cluster of Alibaba Cloud Container Service has been created. Refer to guidance document. NAS file system with mounting address of Alibaba Cloud has been created. Refer to guidance document (File system related section)	版本 02.3 项目主页
Installing the Chart	https://hyperledger.org/projects/fabric
You can use either Helm client or Application Catalog Dashboard of Alibaba Cloud Container Service to install this chart. In either way, please ensure NAS is set during installation. Below we will introduce how to install via Helm client.	链接 • https://github.com/hyperledge/fabric

- 3. 单击参数标签,查看或者修改相关部署参数。
 - sharedStorage: 输入步骤3创建文件系统和添加挂载点中记录的NAS文件系统挂载地址(必填,否则将部署失败)。
 - dockerImageRegistry:根据部署所在区域(中国或海外),从注释中的可选项中选择对
 应的容器镜像仓库地址填入。
 - ・externalAddress: 输入上文 Worker 节点绑定的弹性公网 IP, 用于生成connection profile。

一应	7用目录 - ack-hyperledger-fabric	
	ack-hyperledger-fabric Incubator Hyperledger Fabric Helm chart for Kubernetes on Allbaba Cloud Container Service	
73	Att A	 (2)注 (交互等 Kubernetes 版本 1.84 及以上的集群, 万于 1.8.1 版本的集群, 您可以在集群列 素地拉行 集群升级 強作, 不交增ServerleasKubernetes集群, 集群 (Bit blockham ◆ 会交回 default ▼ Set ServerleasKubernetes集群, ▼ # Control and the set of the set o
	Big Starsalkadress: 11.3.1.4 1 = H laws of thypelladger fahrlin factuorit 2 = # 3 = # laws of thypelladger fahrlin factuorit 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 3 = # 4 = # 4 = # 4 = # 5 = # 4 = #	tri建 版本 0.2.3 项目主页 https://hypeifedger.org/projects/fabric 脱股 ・ https://github.com/hyperfedger/fabric

4. 单击 创建。



5. 进入部署集群的控制台界面,查看区块链网络相关 pods 的状态,直到全部变为 Running。

容器服务 - Kubernetes	-	容器组 (Pod)										周新
概范	1	集群 k8s-blockchain v 命名空间 default v								输入名称查询		٩
▼ 集群	L	名称 ◆	状态 ⇔	重启次数	Pod IP 🖨	节点 ◆	创建时间 🗢	CPU (核)	内存 (字节)			
集群 节点	L	network01-ca1-deployment-6dc494d7d8-tqxxv	●运行中	0	172.16.2.73	192.168.0.45	2019-03-11 12:08:45	0		17.125 Mi	详情	更多 🗸
存储器	L	network01-ca2-deployment-6f6989f9d6-t4ncq	●运行中	0	172.16.1.189	192.168.0.44	2019-03-11 12:08:45	0		16.984 Mi	洋情	更多 🗸
命名空间	L	network01-explorer-deployment-db55499fb-nfdmn	●還行中	0	172.16.1.187	192.168.0.44	2019-03-11 12:08:45	0.005		234.648 Mi	详情	更多 🗸
授权管理 ▼ 应用	L	network01-fabric-cli	●运行中	0	172.16.1.186	192.168.0.44	2019-03-11 12:08:45	0		11.012 Mi	评情	更多▼
无状态	L	network01-fabric-image-downloader-daemonset-525gh	●运行中	3	172.16.2.69	192.168.0.45	2019-03-11 12:08:45	0		92 Ki	洋情	更多 🗸
有状态		network01-fabric-image-downloader-daemonset-fsfbs	●运行中	3	172.16.2.195	192.168.0.46	2019-03-11 12:08:45	0		92 Ki	详情	更多,
45 45	L	network01-fabric-image-downloader-daemonset-mpw89	●运行中	3	172.16.1.188	192.168.0.44	2019-03-11 12:08:45	0		96 Ki	详情	更多 🗸
定时任务	L	network01-fabric-int-pod	●运行中	0	172.16.2.71	192.168.0.45	2019-03-11 12:08:45	0.001		7.492 MI	洋情	更多 🗸
容器组	ł	network01-fabric-network-generator-deployment-5559f5dc96-kStf2	●运行中	0	172.16.2.72	192.168.0.45	2019-03-11 12:08:45	0.001		13.504 Mi	洋情	更多,
发布	L	network01-fabric-utils-pod	●還行中	0	172.16.2.194	192.168.0.46	2019-03-11 12:08:45	0		608 Ki	洋情	更多 🗸
▼ 路由与负载均衡	1	network01-kafka1-deployment-99559999b-2gnk8	●运行中	0	172.16.2.197	192.168.0.46	2019-03-11 12:08:45	0.004	/	705.699 Mi	详情	更多 🕶
服务路由		network01-kafka2-deployment-795c979c9d-twpsh	●运行中	0	172.16.2.70	192.168.0.45	2019-03-11 12:08:45	0.008		1.12 Gi	详情	更多 🗸
▼ 应用配置		network01-kafka3-deployment-9dcddcfc6-jwd2w	●运行中	0	172.16.1.190	192.168.0.44	2019-03-11 12:08:45	0.007		1.104 Gi	详情	更多,
配置项		network01-kafka4-deployment-797db96654-gsr2b	●运行中	0	172.16.2.196	192.168.0.46	2019-03-11 12:08:45	AAA/ 0.004	~~~	575.496 Mi	详情	更多,

容器控制台的 Kubernetes 服务也支持通过 helm 部署区块链网络,具体请参见#unique_7。

步骤 5 用 CLI 测试区块链网络

1. 在 Kubernetes 集群的 master 节点上,执行以下命令进入 CLI 容器。

kubectl exec -it <fabricNetwork>-fabric-cli bash

2. 执行以下命令开始 CLI 测试。

./cli-test.sh

3. 测试过程中,按任意键继续下一步。

4. 如测试过程中没有任何错误,并且最后出现如下字样,说明测试已经成功完成。

步骤 6 访问区块链浏览器

1. 登录 容器服务控制台。

 在Kubernetes菜单下,单击左侧导航栏中的应用>服务,选择所需的集群和命名空间,找 到<网络名称>-explorer服务并访问其外部端点。

您也可在 Kubernetes 集群的 master 节点上运行 kubectl get svc 命令,或者登录容器 服务控制台,进入 kubernetes 集群的控制台,单击左侧导航栏中的 服务,然后查看 <网络名称>-explorer 服务的 EXTERNAL-IP(外部端点)。

_										
	容器服务 - Kubernetes ▼	服务 (Service)							刷新	0.92
	授权管理	常见问题:	. 0					10 1 / / / / / / / / / / / / /		0
	* 应用							制八百分里用		×.
	无状态	名称	後型	创建时间	集群IP	内部論点	外部豌点			撮作
	有状态	kubernetes	ClusterIP	2019-03-08 11:24:08	172.19.0.1	kubernetes:443 TCP	-	详情 更新	查看YAML	删除
	守护进程集	network01-ca1	NodePort	2019-03-11 12:08:45	172.19.5.96	network01-ca1:7054 TCP network01-ca1:31054 TCP		详情 更新	查看YAML	一型除
	任务 定时任务	network01-ca2	NodePort	2019-03-11 12:08:45	172.19.5.58	network01-ca2:7054 TCP network01-ca2:31064 TCP		洋街 更新	查看YAML	一動除
	2 2 40	network01-explorer	LoadBalancer	2019-03-11 12:08:45	172.19.3.86	network01-explorer:80 TCP network01-explorer:32645 TCP	47.302.14.199:80	洋街 更新	│ 查看YAML	影除
	存储声明	network01-fabric-network-generator-svc	ClusterIP	2019-03-11 12:08:45	172.19.3.228	network01-fabric-network-generator-svc:8080 TCP		洋情 更新	查看YAML	删除
		network01-kafka1	ClusterIP	2019-03-11 12:08:45	None	network01-kafka1:9092 TCP		详情 更新	查看YAML	删除
		network01-kafka2	ClusterIP	2019-03-11 12:08:45	None	network01-kafka2:9093 TCP		详情 更新	查看YAML	動除
	路由	network01-kafka3	ClusterIP	2019-03-11 12:08:45	None	network01-kafka3:9094 TCP	-	详情 更新	查看YAML	删除
•	 应用配置 	network01-kafka4	ClusterIP	2019-03-11 12:08:45	None	network01-kafka4:9095 TCP		洋情 更新	查看YAML	翻除
	配置项	network01-orderer1	NodePort	2019-03-11 12:08:45	172.19.12.199	network01-orderer1:7050 TCP network01-orderer1:31050 TCP	-	详情 更新	查看YAML	一般除
	保密字典	network01-orderer2	NodePort	2019-03-11 12:08:45	172.19.11.118	network01-orderer2:7050 TCP network01-orderer2:31060 TCP		洋情 更新	∃ 查看YAML	制除
	R\$	network01-orderer3	NodePort	2019-03-11 12:08:45	172.19.6.118	network01-orderer3:7050 TCP network01-orderer3:31070 TCP		洋情 更新	查看YAML	制除
	编排模板 应用目录	network01-peer1	NodePort	2019-03-11 12:08:45	172.19.4.215	network01-peer1:7051 TCP network01-peer1:31051 TCP		洋情 更新	查看YAML	制除
	服务目录	network01-peer2	NodePort	2019-03-11 12:08:45	172.19.12.75	network01-peer2:7051 TCP network01-peer2:31061 TCP		洋街 更新	查看YAML	制除

3. 在网络浏览器中访问上述 EXTERNAL-IP(外部端点)。



步骤 7 删除区块链网络

在 Kubernetes 菜单下,在左侧导航栏单击应用 > Helm,选择区块链网络对应的发布名称、单击删除。

-								
容器服务 - Kubernetes +	发布							周新
1000 CONTR	分批发布 Helm							
· 应用 1	無群 k8s-blockchain v							
无状态	发布名称	状态	命名空间	Chart 各称	Chart 版本	应用版本	更新时间	操作
有状态	ack-hyperledger-fabric-default	 E## 	default	ack-hyperledger-fabric	0.2.3	1.4.0	2019-03-11 12:08:44	¥ i 4
守护进程集								
任务								
定时任务								
87861								
存储声明								
发布 2 Ξ								

2. 在弹出的 删除应用 对话窗口,单击确定。

删除应用		×
•	确定要删除应用 network01 ? ☑ 清除发布记录	
		确定取消

至此我们便完成了环境准备、区块链配置部署、测试区块链、删除区块链的一系列基本操作。对 于后续的开发测试,可以复用区块链配置部署、测试区块链、删除区块链的步骤;或者根据实际需 要、按照产品文档的指引,进一步定制区块链网络环境。

1.4 环境准备

在开始使用阿里云容器服务区块链解决方案之前,我们需要先完成相应的环境准备工作,主要包括:

- ・ 创建 Kubernetes 集群
- · 给 worker 节点绑定弹性公网 IP
- · 创建文件系统和添加挂载点

本文档将对相关准备过程进行说明。

创建 Kubernetes 集群

区块链解决方案是基于云服务器 ECS 所构建的 Kubernetes 集群进行部署的。有关如何创建 Kubernetes 集群,请按照#unique_9文档说明进行操作。在创建 Kubernetes 集群过程中,请 进行以下配置,以确保区块链解决方案的成功部署:

- ・地域: Kubernetes 集群和 NAS 文件系统必须位于相同的地域。关于NAS文件系统支持地域列
 表请参见 产品与服务 > 文件存储 > NAS > 文件系统列表。
- · 网络类型:选择专有网络。
- · SSH登录:为方便管理,可勾选开放公网SSH登录。
- ・ 节点配置: 推荐采用默认设置(如3台 Master 和3台 Worker),或根据需要自行设置。因为
 区块链网络部署的软件、服务、容器数量较多,请确保集群资源满足需要。

提交创建集群后,将需要一定的时间(如数分钟,取决于 ECS 实例数量)完成 Kubernetes 集群 包括云服务器 ECS 的创建。

Kubernetes 集群创建完成后,在集群列表中,单击对应集群的 管理,在 基本信息 > 连接信息中,记录 Master 节点 SSH 连接地址的 IP(此为公网地址),接下来将作为外部访问地址(externalAddress)使用。

此外,单击 Kubernetes > 节点,进入节点列表,记录当前集群对应的节点(Master 和 Worker)的 IP 地址 (此为内网地址),留待后续步骤使用。

为 worker 节点绑定弹性公网 IP

此部分主要是为从集群外访问区块链网络所进行的先决条件配置,为 Kubernetes 集群的任一worker 节点创建并绑定弹性公网 IP。

操作步骤

- 1. 登录 IP 管理控制台。
- 2. 在左侧导航栏,单击弹性公网IP。
- 3. 单击 申请弹性公网IP。

专有网络	弹性公网IP 2	⑦ 弹性公网 IP 介绍
专有网络	曲調發性公開中 指走中地址申请 局新 导出 批量续费 自定义	弹性公网IP地址 > 请编入名称或ID进行精确查询 Q
交换机		達帝宽/全球 绑定实例 实例类型(全 操作 速 節) \?
共享带宽 共享流量包	eip-bp198c5u 100 Mbps 后付费 wweghvhf/wf6	1)入共享带宽 ngw-bp17buj7i NAT 网关 绑定 解却 v16x4s0rtazz 夏多操作〜
 ▼ 弹性公网IP 3 3 (清度形级监控) 	elp-bp1wuyf9c 100 Mbps 后付畫 b4kmz8dxktt2	Hbp1gckpmdc wpy6p299/2 554466025921 40545606984 e4445245-nod e2

- 4. 选择与 Kubernetes 集群相同的地域,对于其他设置,根据需要自行选择,然后单击 立即购买。
- 5. 开通完成之后, 稍等片刻, 等上述购买的示例出现在列表中, 单击 绑定。
- 在 ECS 实例下拉列表中,选择一个 worker 节点(名称是以 node 结尾,而不是 master 结尾),单击确定。

7. 返回弹性公网 IP 列表,记录上述新创建实例的 IP 地址。

创建文件系统和添加挂载点

在区块链解决方案中,文件系统主要用于:存储和分发区块链的证书、密钥、配置;区块链主要节 点的数据持久化存储。

蕢 说明:

以上文件系统的使用方式仅适用于区块链相关应用和解决方案的开发、测试阶段,如需在生产环境 部署,请联系我们进一步探讨具体的业务和技术需求,以共同决定最适合的方式。

有关如何创建文件系统,请按照 创建文件系统 文档说明进行操作。在创建文件系统的过程中,请务 必选择以下关键配置,确保区块链解决方案的成功部署。

- · 地域:选择与容器集群相同的地域。
- ・协议类型:选择 NFS 协议类型。

*协议类型:	NFS (包含NFSv3和NFSv4)	۳
	SMB协议邀请测试中,敬请关注	

有关如何添加挂载点,请按照 <mark>添加挂载点</mark> 文档说明中的 "添加 VPC 类型挂载点"章节进行操作。

完成添加挂载点后,选择文件系统的管理选项。

文件系统ID/名称	存储类型	协议类型	存储量	所在可用区	已绑存储包	挂载点数目		操作
035bb49518 035bb49518	容 <u>量型</u>	NFS	224.75 MB	华东 1 可用区 B	是	1	添加挂载点 管理	删除

然后记录挂载点的挂载地址。

挂載点						添加挂戴点	^
挂载点类型◆	VPC 🗢	交换机◆	挂载地址◆	权限组	状态◆		操作
专有网络	vpc-	VSW-	-ety45.cn-hangzhou.nas.aliyuncs.com	VPC默认权限组 (全部允许)	可用	修改权限组 激活 禁用 ;	删除

1.5 访问区块链解决方案主页

阿里云容器服务的区块链解决方案以 Helm chart 的形式发布在容器服务 Kubernetes 的应用目 录中。您可以通过以下方式查看区块链解决方案的介绍和配置参数等信息。

操作步骤

1. 登录 容器服务管理控制台。

2. 在 Kubernetes 菜单下,单击左侧导航栏中的 应用目录。

3. 在应用列表中, 单击 ack-hyperledger-fabric。

容器服务 - Kubernetes +	应用目录			
授权管理 ▼ 应用 无状态	Øo	Øo	Øo	Ø ₀
有状态	ack-arms-pilot 1.0.1 incubator	ack-consul 0.5.0 incubator	ack-hyperledger-fabric 1.4.0 incubator	ack-istio 1.0.5 incubator
任務 定时任務 容難相	P ^O _O	Øo	Øo	Øo
存储声明 五	ack-istio-certmanager 1.0.5 stable	ack-istio-coredns 1.0.5 stable	ack-istio-customgateway 1.0.5 stable	ack-istio-egressgateway 1.0.5 stable
 路由与负载均衡 服务 	¢o o	Øo	Øo	Øo
→ 应用配置 配置项	ack-istio-ingressgateway 1.0.5 stable	ack-istio-init 1.0.5 stable	ack-istio-remote 1.0.5 incubator	ack-knative-build 0.3.0 incubator
(2) (1) (2) (2) (2) (2) (2) (2) (2) (2) (2) (2	¢°	Øo	Øo	Øo
線線 編排機板 血目录 2	ack-knative-eventing 0.3.0 incubator	ack-knative-init 0.3.0 incubator	ack-knative-serving 0.3.0 incubator	ack-knative-sources 0.3.0 incubator

 进入 ack-hyperledger-fabric 详情页,可查看区块链解决方案的介绍、部署条件、部署命 令、测试命令、调试命令和配置参数等信息。

1.6 配置部署区块链

在完成环境准备工作之后,我们接下来可以开始区块链网络的配置和部署。区块链网络是基于 Hyperledger Fabric,由以下几种标准类型节点所组成的一套区块链运行环境。

- · Orderer:用于将区块链的交易组合成一个个区块。从可扩展性角度考虑,区块链解决方案采用 Kafka 类型的 Orderer 服务。
- · Kafka 和 ZooKeeper: 以集群形式为 Orderer 提供底层服务。
- · Peer:用于保存和维护账本、创建运行 chaincode 容器、为交易进行背书等等。从高可用性角 度考虑,区块链解决方案为每一个组织(organization)创建 2 个 Peer 节点。
- · CouchDB:用于保存 Peer 的状态数据库 (State Database)。区块链解决方案为每一个 Peer 创建一个 CouchDB。
- ・ CA:用于为应用提供 PKI 证书服务。区块链解决方案为每一个组织(organization)创建一 个 CA 节点。

更多详细介绍可参考Hyperledger Fabric官方文档。

为满足企业级应用的需求,区块链解决方案为主要节点提供了数据持久化存储,使用的是 在#unique_12环节中所创建的共享文件系统。

区块链网络部署运行于阿里云容器服务 Kubernetes 集群之上。用户可选择在同一个 Kubernetes 集群里部署多套区块链网络(用 namespace 隔离),也可以在每一个 Kubernetes 集群里部署 一套区块链网络。 区块链网络的配置部署支持两种方式:图形化界面和 Helm 命令行。下面将对这两种方式进行介绍。

📋 说明:

目前暂不支持在已有区块链网络上动态增加新的 organization 或 peer。因此若您想变更区块链网络的配置,需要删除原有区块链网络重新创建。

配置参数说明

区块链解决方案为大部分字段均提供了默认值,以降低用户配置的复杂度。如需对区块链进行定制,可参考以下字段说明进行参数设置。

参数项	说明
sharedStorage	NAS文件系统的挂载地址。创建区块链网络必须提供的参数,否则将导致创建失败
storageCapacity	Kubernetes中对NAS的Persistent Volume Claim(PVC)的初始大小,NAS支持动态扩容。 默认值:1Gi
dockerImageRegistry	Docker 镜像仓库 URL:区块链解决方案的镜 像仓库。请根据 Kubernetes 集群所在区域选 择对应的镜像仓库。
	 · 国内区域: registry.cn-hangzhou. aliyuncs.com/blockchain-solution · 海外区域: registry.ap-southeast-1. aliyuncs.com/blockchain-solution
fabricNetwork	区块链网络名称(必填项):区块链网络会作为 容器服务的一个应用部署,此区块链网络名称即 为应用名称。请避免使用与已部署应用相同的名 称。此外区块链网络名称也会在共享文件系统中 作为保存配置和数据的根目录名称。
fabricChannel	区块链网络通道名称:即 Hyperledger Fabric 的 channel 名称,区块链解决方案将在 部署时自动创建指定名称的通道。
externalAddress	外部访问地址(必填项):如用户希望利用部署 于容器集群之外的应用程序或管理监控工具访 问区块链网络,则需要提供所在 Kubernetes 集群的一个节点的公网地址或者 Kubernetes 集群的负载均衡的公网地址作为外部访问 地址。相关配置方法可参考#unique_12/ unique_12_Connect_42_section_ph2_355_w

参数项	说明
ordererDomain	Orderer 域:即 Hyperledger Fabric 中 orderer 的 domain,可根据实际需要指定。
ordererNum	Orderer 数量:使用 Kafka 类型(非 Solo 类型)的 orderer 服务。指定希望部署的 orderer 节点数量。如需更改ordererNum ,请确保同时修改ordererExternalPortList ,以保证节点数量和外部端口数量一致,否则将 导致区块链部署失败。
peerDomain	Peer域:即 Hyperledger Fabric 中 peer 的 domain,可根据实际需要指定。
orgNum	组织数量:即Hyperledger Fabric中 organization的数量,区块链解决方案为每一 个组织创建两个peer,以保证高可用性以及业 务扩展的需求。可根据实际需要指定组织的数 量,实际部署的peer节点数量=组织数量 x 2 。如需更改orgNum组织数量,请确保同时修 改peerExternalGrpcPortList、caExternal PortList,以保证节点数量和外部端口数量一 致,否则将导致区块链部署失败。
ordererExternalPortList	Orderer 外部端口列表:如希望使用集群外部 的应用访问 orderer 服务,可指定 orderer 节 点所使用的外部端口或者使用默认端口。请注意 避免不同区块链网络之间占用相同端口可能导致 的区块链部署失败。同时请保证列表中端口的数 量要与 ordererNum 数量保持一致,否则也将 导致区块链部署失败。
caExternalPortList	CA 外部端口列表:如希望使用集群外部的应 用访问CA服务,可指定 CA 节点所使用的外 部端口或者使用默认端口。请注意避免不同区 块链网络之间占用相同端口可能导致的区块链 部署失败。同时请保证列表中端口的数量要与 orgNum 数量保持一致,否则也将导致区块链 部署失败。
peerExternalGrpcPortList	Peer gRPC 外部端口列表:如希望使用集群外部的应用访问 peer 服务(默认基于 gRPC 协议),可指定 peer 节点所使用的外部端口或者使用默认端口。请注意避免不同区块链网络之间占用相同端口可能导致的区块链部署失败。同时请保证列表中端口的数量要与(orgNum x 2)数量保持一致,否则也将导致区块链部署失败。

参数项	说明
imagePullPolicy	镜像拉取策略:此为 Kubernetes 参数,一般 用于开发测试目的。
hyperledgerFabricVersion	Hyperledger Fabric 版本号: 目前支持 1.4.0 ,无需设置。
thirdPartyImageVersion	Hyperledger Fabric 包含的第三方软件(如 CouchDB、Kafka、ZooKeeper等)镜像的 版本号:目前支持 0.4.14(与Hyperledger Fabric 1.4.0 对应),无需设置。
explorer.enabled	是否自动部署Hyperledger Explorer。在部署 过程中同时会创建负载均衡实例,并通过80端 口提供基于Web UI的区块链浏览器功能。默认 为true。
logService.enabled	是否开启对阿里云日志服务的支持。默认 为false。关于阿里云日志服务的详情可参 考#unique_13。
logService.region	如开启对阿里云日志服务的支持,则指定日志服 务项目所在的地域。请根据实际地域指定。关于 阿里云日志服务的详情可参考#unique_13。
logService.userID	如开启对阿里云日志服务的支持,则指定阿里云 主账号的用户ID。关于阿里云日志服务的详情 可参考#unique_13。
logService.machineGroup	如开启对阿里云日志服务的支持,则指定日志服 务项目的机器组。关于阿里云日志服务的详情可 参考 #unique_13。
logLevel	Hyperledger Fabric不同类型节点(Peer, Orderer, CouchDB)的日志级别。可选值为: CRITICAL ERROR WARNING NOTICE INFO DEBUG;默认值: INFO

使用控制台界面部署区块链

1. 在 Kubernetes 菜单下,单击左侧导航中的 应用目录,在右侧列表区域找到并单击 ackhyperledger-fabric。



2. 在 ack-hyperledger-fabric 应用界面右侧区域,选择部署 集群,并填写 发布名称。

应用目录 - ack-hyperledger-fabric				
Ack-hyperledger-fabric Incubator Hyperledger Fabric Helm chart for Kubernetes on Albaba Cloud Container Service				
2006 100	创建			
Hyperledger Fabric on Kubernetes of Alibaba Cloud Container Service				
Hyperledger Fabric is one of the most popular blockchain infrastructures in the world, which is open sourced and hosted by Linux Foundation.	仅支持 Kubernetes 版本 1.8.4 及以上的集群。对于 1.8.1 版本的集群,您可以在集群列 家中进行"集群升级"操作。不支持ServerlessKubernetes集群。			
	朱祥			
Introduction	k8s-blockchain 🔻			
This shart implements a solution of automatic configuration and dealayment for kinetical or Fahir. The solution is dealayment or Kikehe Claud Contains:	命名空间			
The Chair imperiences a solution of automatic computation and deployment or ryperleager solution is deployed on observed on autoences duste or Autoad Codu Container Service. And the Hyperleager Fabric network can be accessed by CLL applications and explorers within or outside the Kubernetes cluster. Blockchain Explorer is now integrated in the inclusion as wall.	default			
mu the solution as well.	发布名称			
A NAS (NFS protocol) shared file storage is needed for: 1. distribution of crypto and configurations; 2. data persistence for most services.	ack-hyperledger-fabric-default			
Currently v1.4.0 of Hyperledger Fabric is supported.	6570			
You can also refer to the documentation for blockchain solution of Alibaba Cloud Container Service.	UDBA			
Prerequisites				
A Kuhematas cluster of Alihaha Cloud Container Service has been created Defer to quidence document	版本			
 A Addemetes usate or Analysis close contents service has been created, keer to guidance document (File system related section) NAS file system with mounting address of Alibaba Cloud has been created. Refer to guidance document (File system related section) 	0.2.3			
	项目主页			
Installing the Chart	https://hyperledger.org/projects/fabric			
You can use either Helm client or Application Catalon Dashboard of Alibaba Cloud Container Service to install this chart. In either way, please ensure NAS is set during	链接			
installation.	https://github.com/hyperledger/fabric			
Below we will introduce how to install via Helm client.				

- 3. 单击参数标签,可以查看或者修改相关部署参数。
 - sharedStorage: 输入环境准备环境记录的NAS文件系统挂载地址(必填,否则将部署失败)。
 - dockerImageRegistry:根据部署所在区域(中国或海外),从注释中的可选项中选择对
 应的容器镜像仓库地址填入。
 - ・externalAddress: 输入上文 Worker 节点绑定的弹性公网 IP, 用于生成connection profile。

応知目录 - ack-hyperledger-fabric Incubator Hyperledger Fabric Helm chart for Rubernetes on Albaba Cloud Container Service	
<pre>P30 P30 1 • 64 NS Thereof file system monthing address 2 # P4 TRETROTOR: This must be provided by user, otherwise to installation of this chart will full 2 # P4 TRETROTOR: This must be provided by user, otherwise to installation of this chart will full 2 # P4 TRETROTOR: This must be provided by user, otherwise to installation of this chart will full 2 # P4 TRETROTOR: This must be provided by user, otherwise to installation of this chart will full 2 # P4 TRETROTOR: This must be provided by user, otherwise to install with the part of the</pre>	分注
36 #* 37 for CiClampel: bankchannel 38 ## Number of orderers 39 ## 31 ## Ordererban: 3 33 ## Data load or otherers 34 ## This domain will be used mainly for certification generation 36 ## This domain will be used mainly for certification generation 36 ## This domain will be used mainly for certification generation 37 ## Domain of pers 38 ## This domain will be used mainly for certification generation 41 ## This domain will be used mainly for certification generation 42 ## Number of pers organizations	版本 0.2.3 项目主页 http://hyperledger.org/projects/fabric 链接 • http:://github.com/hyperledger/fabric

4. 单击 创建。

说明:

如参数中的sharedStorage未作配置,部署过程将会报 spec.nfs.server: Required value 的错误。出现此错误后,需要先删除对应的发布,然后填写sharedStorage参数 值,并重新部署。

5. 进入部署集群的控制台界面,查看区块链网络相关 pods 的状态,直到全部变为 Running。

容器服务 - Kubernetes ▼	容器组 (Pod)									周新
概范	集群 k8s-blockchain ▼ 命名空间 default ▼							输入名称查询		Q
▼ 集群	名称 令	状态 ♠	重启次数	Pod IP 🜩	节点 🗢	创建时间 🜩	CPU (核)	内存 (字节)		
集群 节点	network01-ca1-deployment-6dc494d7d8-tqxxv	●运行中	0	172.16.2.73	192.168.0.45	2019-03-11 12:08:45	0	17.125 Mi	洋情	更多,
存储带	network01-ca2-deployment-6/6989/9d6-t4ncq	●运行中	0	172.16.1.189	192.168.0.44	2019-03-11 12:08:45	0	16.984 Mi	洋情	更多 •
命名空间	network01-explorer-deployment-db55499fb-nfdmn	●运行中	0	172.16.1.187	192.168.0.44	2019-03-11 12:08:45	0.005	234.648 Mi	详情	更多 ▼
授权管理 ▼ 应用	network01-fabric-di	●运行中	0	172.16.1.186	192.168.0.44	2019-03-11 12:08:45	0	11.012 Mi	详情	更多 •
无状态	network01-fabric-image-downloader-daemonset-3z5gh	●运行中	3	172.16.2.69	192.168.0.45	2019-03-11 12:08:45	0	92 Ki	洋情	東多・
有状态	network01-fabric-image-downloader-daemonset-fsfbs	●运行中	3	172.16.2.195	192.168.0.46	2019-03-11 12:08:45	0	92 Ki	洋情	更多 •
69-20129e	network01-fabric-image-downloader-daemonset-mpw89	●运行中	3	172.16.1.188	192.168.0.44	2019-03-11 12:08:45	0	96 Ki	详情	● 更多 ▼
定时任务	network01-fabric-init-pod	●运行中	0	172.16.2.71	192.168.0.45	2019-03-11 12:08:45	0.001	7.492 MI	详情	更多,
容器组	network01-fabric-network-generator-deployment-5559f5dc96-k5tf2	●运行中	0	172.16.2.72	192.168.0.45	2019-03-11 12:08:45	0.001	13.504 Mi	洋情	US •
发布	network01-fabric-utils-pod	●运行中	0	172.16.2.194	192.168.0.46	2019-03-11 12:08:45	0	608 Ki	详情	更多 •
▼ 路由与负载均衡	network01-kafka1-deployment-99559999b-2gnk8	●运行中	0	172.16.2.197	192.168.0.46	2019-03-11 12:08:45	0.004	705.699 Mi	洋情	更多 •
服务路由	network01-kafka2-deployment-795c979c9d-twpsh	●运行中	0	172.16.2.70	192.168.0.45	2019-03-11 12:08:45	0.008	1.12 Gi	详情	更多 •
▼ 应用配置	network01-kafka3-deployment-9dcddcfc6-jwd2w	●运行中	0	172.16.1.190	192.168.0.44	2019-03-11 12:08:45	0.007	1.104 Gi	洋情	
配置项	network01-kafka4-deployment-797db96654-gsr2b	●运行中	0	172.16.2.196	192.168.0.46	2019-03-11 12:08:45	A A A A 0.004	575.496 Mi	洋情	更多,

使用 Helm 命令行部署区块链

如需了解容器服务 Kubernetes 集群的 Helm 部署应用的更多功能,请参考#unique_14。

1. 用 SSH 工具登录 Kubernetes 集群的 master 节点,使用 root 账户和在创建 Kubernetes 集 群时设置的密码进行登录。

- 2. 运行 Helm 命令部署区块链网络。
 - ·如需使用默认配置参数部署区块链网络,可执行如下命令:

```
helm install --name blockchain-network01 --set "sharedStorage
=987a6543bc-abc12.cn-hangzhou.nas.aliyuncs.com" incubator/ack-
hyperledger-fabric
```

其中 --name 为区块链网络对应的 Helm release 名称,用户可自行设置。

·如需将区块链网络部署于指定的 namespace(例如 network01),可执行如下命令:

```
helm install --namespace network01 --name blockchain-network01 --
set "sharedStorage=987a6543bc-abc12.cn-hangzhou.nas.aliyuncs.com"
incubator/ack-hyperledger-fabric
```

其中 --namespace 为区块链网络部署的目标 namespace 名称,用户可自行选定。

・如需设置简单或数量较少的配置参数部署区块链网络,可用 set 参数将配置参数传入,例

如:

```
helm install --name blockchain-network01 --set "fabricChannel=
mychannel,sharedStorage=987a6543bc-abc12.cn-hangzhou.nas.aliyuncs.
com" incubator/ack-hyperledger-fabric
```

·如需设置复杂的或数量较多的配置参数部署区块链网络,可用 yaml 文件将参数值传入,例 如:

```
helm install --values network01.yaml --name blockchain-network01
incubator/ack-hyperledger-fabric
```

其中 --values 指定的是自定义配置参数的 yaml 文件,示例如下,用户可自行定义。

```
# sample values yaml
sharedStorage: 987a6543bc-abc12.cn-hangzhou.nas.aliyuncs.com
fabricNetwork: network01
fabricChannel: tradechannel
orgNum: 3
ordererNum: 4
ordererDomain: shop
peerDomain: shop
externalAddress: 11.22.33.44
caExternalPortList: ["31054", "31064", "31074"]
ordererExternalPortList: ["31050", "31060", "31070", "31080"]
peerExternalGrpcPortList: ["31051", "31061", "31071", "31081", "
31091", "31101"]
```

检查区块链网络的 Helm release 部署成功。执行如下命令,确保区块链网络对应的 release 状态为 DEPLOYED。

3. 执行如下命令,检查区块链网络的所有节点的 pod 是否均成功运行,确保区块链网络对应的 pod 状态均为 running。

kubectl get pod

・如指定了namespace,如network01

kubectl get pod -n network01

·如需要以watcher模式监听pod状态变化

```
kubectl get pod -w
```

4. 执行如下命令, 查看区块链网络部署的创建状态。如果显示状态为 DEPLOYED, 说明部署成功。

helm list

至此我们便完成了一套区块链网络的配置和部署。

区块链网络各节点服务的命名规则

对于 Hyperledger Fabric 的标准节点类型,相应的服务命名规则如下:

```
<区块链网络名称>-<节点类型><序号>
```

例如:

```
network01-peer1
network01-peer2
network01-orderer1
network01-ca1
```

需要说明的是,尽管在 Kubernetes 集群内可用 namespace 来区分不同的区块链网络,但上述 服务命令规则中仍使用了区块链网络名称作为前缀,主要原因是为了保持与 swarm 集群的区块链 解决方案一致。

此服务名称与区块链证书、密钥中的节点名称保持一致。对于部署于同一 Kubernetes 集群内的 区块链应用程序或 CLI,均可使用此类服务名称(加上服务端口)直接访问,无需使用外部访问地 址。

问题诊断

针对区块链的配置、部署、访问等过程中可能发生的各类问题或错误,我们在这里将介绍一些常用 的问题诊断思路、方法和工具。

首先,建议检查#unique_12 文档中所要求的各项准备工作是否均已正确完成。

其次, 善用 Kubernetes 相关命令进行部署事件和输出日志等信息的查看, 例如: kubectl describe pod, kubectl logs, kubectl get pod -o yaml等。

此外,为辅助故障诊断和问题排查,区块链解决方案在区块链网络中一同部署了自定义的 fabric -utils 容器,里面集成了常用的基础工具,如 telnet、ping、nslookup、curl 等。用户可在 master 节点上,通过以下 kubectl 命令进入fabric-utils 容器中使用适合的工具进行诊断分析。 例如:

kubectl exec -it fabric-utils-pod bash

最后,针对 Hyperledger Fabric 相关的问题或错误,可尝试通过搜索 Hyperledger Fabric 官 方文档、StackOverFlow、Google/Bing/Baidu 等查找相关线索或解决方案。

1.7 查看区块链网络日志

区块链网络运行期间的日志是通过其中的 peer、orderer、CA、Kafka、ZooKeeper 等类型节 点的容器日志方式输出的。容器服务区块链解决方案支持以多种方式查看这些日志信息,包括容 器服务控制台、Kubernetes命令、阿里云日志服务等。本文档将对这几种方式的使用方法进行介 绍。

使用容器服务控制台

容器服务控制台提供了便捷的图形化界面的日志查看方式。在部署了区块链网络之后,可以通过以 下操作步骤查看对应节点的容器日志。

操作步骤

- 1. 登录 容器服务管理控制台。
- 2. 在 Kubernetes 菜单下,单击左侧导航栏中的 集群,再单击目标集群的 控制台。

容器服务 - Kubernetes ▼	集群列表		您最多	:可以创建 5 个集群,每个	集群最多可以消	添加 40 个节点 刷新	创建Serverless Kubernetes集群	创建Kubernetes 集群 🛛 👻
概党	常见问题: Ø 如何创建集群 Ø 扩容和缩	容集群 ♂ 通过 kubectl 连	接 Kubernetes 集群	𝔇 通过命令管理应用				
▼ 集群	名称 ▼							
集群	集群名称/ID	集群类型	地域 (全部) 🔻	网络类型	集群状态	创建时间	Kubernetes 版本	2 🕫
节点 存储卷	k8s-blockchain	Kubernetes	华东1	虚拟专有网络 vpc-bp1kd7yn4qn	●运行中	2018-06-27 17:48:29	1.9.7	管理 查君日志 控制台 集群仲缩 更多▼

3. 在左侧导航栏中单击 容器组, 再单击 日志 图标。



4. 查看日志详细信息。

三日志						+ 创建
容器日志 network01-zookeeper3 * 位于 network01-zookeeper3-deployment-846464b8b8-9qp6r *	A	Тт	Ō	Z	0	ŧ
2018-03-28 08:19:32,281 [syid:3] - IMFO [Thread-3:NIOServerCoundHOO8] - Closed socket connection for client /172.20.3.33:47574 (no session est 2018-03-28 08:19:34,847 [syid:3] - WANN [VerkerSender[mpid:3]:QuorumCoudManager8400] - Cannot open channel to 2 at election address network01-z java.net.SockerLineoutEcoption: connect timed out	ablish ookeep	ed for er2/172	client) .21.14.3	5:3888		
at java net. PlainŠooketlapl. sooketConnect (Native Method) at java net. AbstractPlainŠocketLapl. do Connect (AbstractPlainŠooketLapl. java: 350) at java net. AbstractPlainŠocketLapl. connectToAddress (AbstractPlainŠocketLapl. java: 206)						
at java net Abstractfianbocketingi, connect Ubstractfianbocketingi, java 1889 at java net Sockstocketingi, connect (Sockstoketingi, java 392) at java net Sockst, connect (Sockst java 588) at java net Sockst, connect (Sockst java 588)						
at og spache rokstepr. av tru gov med dor sekninger. I obre de som get skannen som sekninger. Jøre 507 at org spache roksepr. server. govrum. PastLeaderIlectionMlessager. Noënd (DyorumChallmanger. jøre. 508 at org spache roksepre. server. govrum. FastLeaderIlectionMlessamerSM forkerSender. process(FastLeaderElection. jøre. 453) at org. spache roksepre. server. govrum. FastLeaderIlectionMlessamerSM forkerSender. process(FastLeaderElection. jøre. 433)						
at javalang Thread run(Thread java:748) 2018-03-26 09:19:34,848 [myid:3] - IMFO [WorkerSender[myid=3]:QuorumPeerSQuorumServer@149] - Resolved hostname: networkOl-rookeeper2 to addres 2018-03-26 09:19:34,650 [myid:3] - IMFO [WorkerReceiver[myid=3]:FastLeederElection@600] - Notification: 1 (message format version), 3 (n.leade 100TUNE (n.estate) 3 (n. cid) DeD (n. moerPronch) 100TUNE (m. citate)	s: net r), Ox	work01- 0 (n. zx	zookeepe id), Ox1	r2/172.21 (n. round	14. 3 I),	5
2018-03-28 08 19:34,882 [syid:3] - DRF0 [NIDServerCan.Factory:0.0.0.0/0.0.0.0:2181:NIDServerCannFactory8192] - Accepted socket connection from 2018-03-28 08:19:34,882 [syid:3] - DRF0 [NIDServerCan.Factory:0.0.0/0.0.0.0:2181:NIDServerCannM827] - Processing stat command from /172.20.3 2018-03-28 08:19:34,882 [syid:3] - DRF0 [NIDServerCan.Factory:0.0.0/0.0.0.0:2181:NIDServerCannM827] - Processing stat command from /172.20.3 2018-03-28 08:19:34,882 [syid:3] - DRF0 [NIDServerCan.Factory:0.0.0/0.0.0.0:2181:NIDServerCannM827] - Processing stat command from /172.20.3	/172.: .38:53 ablish	20.3.38 472 ed for	:53472 client)			
2016-03-26 09:19:34, 942 [ayid:3] - IMFO [QuorumFeer[ayid:3]/0.0.0.0.2181;FastLeaderElection0852] - Netification time out: 400 2016-03-26 09:19:34, 947 [ayid:3] - IMFO [NIOServerCan Factory 10.0.0.0/0.0.0.2181;NIOServerCannFactory49[2] - Accepted a socket connection from 2016-03-26 09:19:34, 975 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0.2181;NIOServerCannFactory49[2] - Processing stat command from /12:20.3 2016-03-26 09:19:34, 975 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0.2181;NIOServerCannFactory49[3] - Processing stat command from /12:20.3 2016-03-26 09:19:34, 984 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0.2181;NIOServerCannFactory49[3] - Processing stat command from /12:20.3 2016-03-26 09:19:34, 984 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0.2181;NIOServerCannFactory49[3] - Processing stat command from /12:20.3 2016-03-26 09:19:34, 984 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0.2181;NIOServerCannFactory49[3] - Processing stat command from /12:20.3 2016-03-26 09:19:34, 984 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0.2181;NIOServerCannFactory40] - Processing stat command from /12:20.3 2016-03-26 09:19:34, 984 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0.2181;NIOServerCannFactory40] - Processing stat command from /12:20.3 2016-03-26 09:19:34, 984 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0.0] - Processing from /12:20.3 2016-03-26 09:19:34, 984 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0.0/0.0] - Processing from /12:20.3 2016-03-26 09:19:34, 985 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0] - Processing from /12:20.3 2016-03-26 09:19:34, 985 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0] - Processing from /12:20.3 2016-03-26 [ayid:3] - Processing from /12:20.3 2016-03-26 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0] - Processing from /12:20.3 2016-03-26 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0] - Processing from /12:20.3 2016-03-26 [ayid:3] - IMFO [NIOServerCan Factory:0.0.0] - Processing from /12:20.3 2016-03-26 [ayid:3] - Processing from /12	/172.: .37:34 ablish	20.3.37 948 ed for	:34948 client)			
2018-03-26 09:19:35,154 [myid:3] - INFO [WorkerReceiver[myid=3]:FastLeaderElection8600] - Notification: 1 (message format version), 1 (n.leade LOOKING (n.state), 1 (n.sid), OnO (n.peerEpoch) LOOKING (my state) 2018-03-26 09:19:35,155 [myid:3] - INFO [WorkerReceiver[myid=3]:FastLeaderElection8600] - Notification: 1 (message format version), 3 (n.leade 100TUS (n.state), 1 (n.sid), OnO (n.peerEpoch) LOOKING (my state)	r), Ox r), Ox	0 (n. zx 0 (n. zx	id), 0x1 id), 0x1	(n. round (n. round		

5. 此外,您也可以单击某个容器组,在顶部菜单栏中单击 日志,查看日志详细信息。



使用 Kubernetes 命令

用户也可以通过标准的 kubectl logs 命令查看区块链网络的容器服务日志。

操作步骤

- 1. 在容器服务控制台的 Kubernetes 集群列表,选择区块链部署所在的集群,单击 管理,获得 Master 节点 SSH 连接地址。
- 2. 通过 SSH 方式登录 Kubernetes 集群的 master 节点,输入用户名 root 和创建集群时设置的 密码。
- 3. 运行 kubectl get pod 获得 pod 列表,并选择需要查看日志的 pod 名称。
- 4. 运行 kubectl logs pod名称 命令来查看日志信息。
- 5. 如一个 pod 内包含多个容器,可运行 kubectl logs pod名称 容器名称 命令来查看某个容器 的日志信息。

使用阿里云日志服务

使用容器服务控制台的日志功能或者 Kubernetes 命令两种方式基本可以满足常见的日志查看的需求。但对于企业级需求来说,如果需要日志存储、实时索引查询和分析、报警、可视化报表等高级功能的话,可结合 阿里云日志服务 进行扩展。

容器服务区块链解决方案支持与阿里云日志服务进行整合。下面对基本的操作步骤进行介绍,如需了解更详细的关于在容器服务 Kubernetes 集群集成阿里云日志服务的信息,可参考#unique_17。

使用阿里云日志服务可能会产生一定费用,详情请参考日志服务 计费说明。

操作步骤:

- 1. 登录 日志服务控制台,并按照提示开通日志服务。
- 2. 单击左侧导航栏中的 Project管理,单击右上角的 创建Project。

3. 输入 Project 名称,选择区块链网络所在的地域,然后单击确认。

创建Project	×
* Project名称: blockchain-network-logging	
注释:	
不支持<>"最多包含64个字符 * 所属地域: 华东 1 ▼	
确认	取消

4. 弹出对话框,询问您是否创建 Logstore,单击 创建。

创建Proj	ect		×
0	您已成功创建Project, 请创闢	Logstore用于日本政略存储	
		008	取消

5. 在 创建 Logstore 窗口中,输入 Logstore 名称,其他设置可根据实际需要进行调整,完成后单 击 确认。

IELogstore		
• Logstore名称:	blockchain-network-logstore	
Logstore属性		
• WebTracking:		
	WebTracking功能支持快速采集各种浏览器以及iOS/Android/APP 访问信息、默认关闭(帮助)	
•数据保存时间:	30	
	目前Loghub保存时间和索引已经统一、数据生命周期以Loghub设 重为准(单位:元)	
• Shard数目:	2 0 什么是分区 (Shard) ?	
• 计费:	參考计費中心说明	
	_	

6. 新建 Logstore 后, 会弹出提示对话框, 单击 数据接入向导。

创建	\times
•	您已成功创建Logstore,请使用数据接入向导快速设置采集、分析等使用 方式
	数据接入向导取消

7. 选择 Docker 标准输出,然后单击下一步。

blockchain-network-logs 全返回日志库列表				
1.选择数据关型	2.数据源设置	\rightarrow	3.查询分析 & 可视化	> 4.投递 & ETL
云产品日志				
	MNS 消息服务	API网关	SLB 负载均衡	
自建软件				
NGINX访问日志. NGINX追	拉 MYSQL BINLOG	9 MYSQL	Ei励结果 Docker文件	÷ ● Dockert記書出

8. 在插件配置框内,填入以下示例配置。配置详情可参考 说明文档。完成后单击 下一步。

{	
"inputs": [
{	
"type": "service docker s	stdout",
"detail": {	· · · · ·
"Stdout": true,	
"Stderr": true.	
"IncludeLabel": {	
"ExcludeLabel" · {	
J l	
ے ا	
} ٦	
}	

ockchain-network-logs 全返回日志库列表			
1.选择数据类型	2.数据源设置 3.查询分析 & 可视化	د ک	4.投递 & ETL
 記幣々約 	docker-stdout-config		
的是包约	, docker-addode-coning		
* 插件配置	: 溫馨提示:本功能只支持Linux系统,并且需要基于最新版本Logtail,如何升级 Logtail请参考链接		
	IncludeLabel表示只有包含对应的key/value才会被采集,为空表示全部采集, ExcludeLabel包含对应的key/value不会被采集,如何使用清参考链接		
	{		
	"inputs": [
	"type": "service_docker_stdout",		
	"detail": {		
	"Stdout : true, "Stderr": true.		
	"IncludeLabel": {		
), 		
	"ExcludeLabel": {		
	}		
	}		
	1		

9. 单击 创建机器组。

在创建机器组窗口,填入自定义的机器组名称,在机器组标识下拉框中选择 用户自定义标 识,在用户自定义标识编辑框填入和机器组名称一致的内容,最后单击 确认。示例如下:

创建	【机器组		×
	* 机器组名称:	blockchain-network01	
	机器组标识:	用户自定义标识 ▼	
		如何使用用户自定义标识	
	机器组Topic:		
		如何使用机器组Topic ?	_
*	用户自定义标识:	blockchain-network01	
		确认	取消

10.勾选刚创建的机器组,单击应用到机器组,再单击下一步。

blockchain-network-logs 全返回日志库列表					
1.选择数据关型	2.数据源设置		3.查询分析 & 可视化	\rightarrow	4.投递 & ETL
应用到机器组					
		+ 创建机器组			
					*
9					2
					应用到机器组

11.可根据需要,添加键名称用于建立索引,例如 _pod_name_,完成配置后,单击下一步。

		3.查询分析 & 可视化	4.扮進 & ETL
义 2			
全文察引属性:			预览
大小写敏感 分词符		时间/IP	内容
false ▼		封无数境 1. 如果使用LogtalI采集日志,请确认机器组心跳星否正常(调查文档),正常后仍无数据点击诊断 2. 如果使用API/SDK,请查署程序输出日志	
建名称 + 类型 别名 大小写敏感 分词符 开启统计	删除		
_pod_name_ text v container false v , ";=()[]{}?@&<	×		
全文素引屬性和鍵值素引屬性必须至少启用一种 素引使因为long/doublest,大小写敏感和分词符属性无效			

12.单击 确认,根据页面引导,完成剩余步骤。至此我们完成了阿里云日志服务的创建和初始化配置。

13.接下来,利用区块链解决方案部署一套新的区块链网络,与日志服务集成的相关参数在方案主页 的参数页面进行设置。



您需要将 enabled 参数设为 true,表示启用 logservice 服务,此外,需要将 machineGro up 参数设置为机器组中配置的自定义用户标识,本例中即是 blockchain-network01。

Region 的设置参见 Linux 安装 logtail,查找相关安装命令,从而查找 region ID。例如 cn_hangzhou,表示从杭州地域的阿里云内网写入日志服务,不消耗公网带宽。

账号管理	安全设置
安全设置	登录账号 :
基本资料	账号ID:
联系人管理	注册时间:
实名认证	修改头像

UserID 的设置参考下面截图:
14.要开始利用阿里云日志服务,在日志服务控制台的 Logstore 列表中,单击目标 Logstore 右侧的 查询。

Logstore列表							查看End	point创建
请输入Logstore名进行模糊图 搜索								
Lonstore名称	数据接入向导	数据接入向导 监控	日志采集模式		日志消费模式			操作
LUGSCOLUM	RYMIN (1943			日志消费	and the second se	日志投递	查询分析	DRTP
blockchain-network-logstore	8	⊭	Logtail配置(管理) 诊断 更多-	预览		MaxCompute OSS	查询	修改 删除
						共有1条,每	页显示:10条 《	< 1 > »

区块链网络的日志信息示例如下:

<	B blockchain-network-log	store (雇于)				分享 查询分析	属性 另存为快速查询	月存为告誓
标签列表	请输入关键字进行搜索 6k				@ 1	15分钟 🗸 2	2018-03-27 14:46:53 ~ 2018-0	13-27 15:01:53	搜索
🗟 blockchain-n	0 46⊖56€ 47⊕45€	489 3 458)	49;3458) 50;3458)	5194580 5294580 5894580	549458 559458	569458	57(6458) 58(6458)	59;34580 00;34580	01:94118
新建标签 🥝	原始日志 统计图	表		日志总条数:15,294 查询	状态:结果精确				
	快速分析	<	时间 🖛	内容▼					⊎ ⊚
	_pod_name_	1 Q	03-27 15:01:47	sourcetraz.os.z teaghostemanologial-r/bbj containeranamonetwork01-oeer1 manoregistry.cn-hangzhou.allyuncs.or 11 namespacedefault nodnamenetwork01-peer1-deployment-Sct nodnamenetwork01-peer1-deployment-Sct nodnamenetwork01-peer1-deployment-Sct nodnamename_name nodnamename_name nodname_name_name nodname_name_name nodname_name_name nodname_name_name nodname_name_name nodname_name_name nodname_name_name nodname_name_name name_name_name name_name_name name_name_name name_name_name name_name_name name_name_name name_name_name name_name_name name_name_name name_name name_name name_name_name name_name na	com/cos-solution/fabric-; 5b679ff8-vx47r Qe5Def endorser] ProcessPropos	-peer@sha256:57417696 bsal -> DEBU 1511-(0m E	9ddf50c5ebd47a9a2cc74c032 Exit: request from%i(EXTRA s	4fbba0281eb1104b9ddd0 tring=172.20.3.13:42208)	35a67776b0
		2 Q	03-27 15:01:47	Source_int2.20.3.2 Logital_robustlessure_interview. Logital_robustlessure_interview. Logital_robustlessure_interview. Logital_robustlessure_interview. Source_isterview. Logital_robustlessure_interview. Source_isterview. Logital_robustlessure_interview. Logital_robustle	com/cos-solution/fabric-; 5b679ff8-vx47r 0e50ef endorser] simulatePropo	-peer@sha256:57417699 psal -> DEBU 15f0-[0m [9ddf50c5ebd47a9a2cc74c032 [[e169b90e] Exit	4fbba0281eb1104b9ddd0	05a67776b0

15.进一步的,日志服务支持进行复杂的查询,更多查询语法和其他日志服务的高级功能,请参见 查询语法。

	_		
network01-ca1 or network01-ca	12		② 15分钟
200			
55分19秒 56分15秒	57分15秒	58分15秒 59分15秒	003/158 013/158 023/158 033/158 043/158 053/158 063/158 073/158 083/158 093/158 103048
			日志总会数 174 查询状态: 結果精确
原始日志 统计图录	Ę		
快速分析	<	时间 🔺	内容▼ 🕘 @
_pod_name_ network01-peer1-dep 50.34% network01-explorer-d 11.97% network01-peer2-dep 7.83% network01-peer3-dep entwork01-peer3-dep	1 Q	03-27 14:58:20	source
network01-peer4-aep 6.49% network01-orderer1-d 5.21% network01-orderer2-d 1.3.51% network01-orderer3-d 1.3.47% network01-akfka3-de 1.4.62%	2 Q	03-27 14:58:20	source
approx_distinct	3 Q	03-27 14:58:20	source: 172.20.3.2 tag:_hostname: logtail-tj6bj topic: _container_mame_: network01-ca1

1.8 访问区块链概述

在阿里云上完成了区块链网络的配置和部署之后,区块链的开发者或管理员便可以开始访问区块链 网络并使用不同的区块链节点服务。常见的访问方式如下:

- · 开发者或管理员远程连接到区块链节点的容器上,以 CLI 命令方式运行区块链测试,或者进行 区块链的管理工作。
- · 区块链应用程序连接区块链网络的 CA、Orderer、Peer 等服务,进行基于区块链的交易和服务调用。
- · 区块链管理监控工具连接区块链网络,对区块链网络以及各节点进行图形化或自动化的管理和监 控操作。

这些区块链的访问方式和相关程序、代码可由用户根据自身的业务和技术需求进行开发和部署,部 署形式既能以容器应用的方式与区块链网络一起部署在容器集群中,也可部署在用户自有的环境 中,从容器集群外部访问区块链网络。对于第二种方式,我们需要提前为区块链网络创建公网IP或 者负载均衡,以及为外部端口(NodePort)配置安全组规则,以允许外部的访问连接。

在本文档中,我们为这几种主要的区块链访问方式提供了简单的示例和使用说明,以帮助用户更好 地理解和开发相关的应用和工具。同时,区块链解决方案提供了可一键下载的配置文件(包含连接 区块链服务所需的证书、密钥、区块链网络配置文件等),以加速区块链应用和管理监控工具的开 发、测试流程。

- #unique_22
- #unique_23
- · 以应用程序方式访问区块链
- #unique_25

1.9 配置公网IP和外部端口

前提条件

如需要将区块链网络开放给容器集群外的应用程序和管理监控工具访问,需要具备以下几项前提条 件:

- 有外部可达的公网地址。实现方式既可以通过给 worker 节点绑定弹性公网 IP,也可以创建负载均衡实例(后端服务器添加 worker 节点),使用其提供的公网IP。在#unique_12 文档中我们已经对第一种方式进行了介绍,请参阅其中的#unique_12/unique_12_Connect_42_section_ph2_355_vdb章节。
- · 允许外部端口列表中的 NodePort 的入方向访问,实现方式是通过 ECS 的安全组规则设置。

操作步骤

- 1. 登录 云服务器 ECS 管理控制台。
- 2. 在左侧导航栏,单击网络和安全 > 安全组。
- 3. 在安全组列表中,单击选择你的 Kubernetes 集群所在的地域。
- 4. 在列表中, kubernetes 集群的安全组名称带有 k8s_sg 字样, 请据此选择对应的安全组

ID,单击 配置规则。

云服务器 ECS	安全组列表 华北1 华北2 华	d北 3 华北 5 华 东 1 华东 2	华南1 香	清港 亚太东	R北 1 (东京) 亚太东南	1 (新加坡)		c	创建安全组
概览 实例	亚太东南 2 (悉尼) 欧洲中部 1 (法兰売福	亚太东南3 (吉隆坡) 美国东部 1 3) 3	(弗吉尼亚)	美国西部 1	(硅谷) 中东东部 1 (迪	拜)			
▼ 存储	安全组ID · 输入安全组ID精确	查询,多个用","隔开	liter 📎	标签					2 ?
云盘	■ 安全组ID/名称	所属专有网络	相关实例	网络类型	创建时间	描述	标签		4 ^{//E}
共享块存储 文件存储 NAS	sg-bp1hu2a9ydcs76kiivnc	vpc-	6	专有网络	2017-12-26 10:49:19	-	ros-aliyun-created:k8s	修改 克隆 管理实例	还原规则
▼ 快照和镜像	sg-bp1he4m3a5sz45azq1cy alicloud-cs-auto-creat	vpc-	0	专有网络	2017-12-08 13:40:24	security group of ACS		修改 克隆 管理实例	· 还原规则 · 配置规则
快照列表	sg-bp1gnm9vvm42o31vpyht sg-bp1gnm9vvm42o31vpyh	vpc-	0	专有网络	2017-12-06 21:55:40	System created securit		修改 克隆 管理实例	还原规则
自动快照策略	sg-bp10ay4bt2m4kddat7j8 alicloud-cs-auto-creat 🖍	vpc-	1	专有网络	2017-12-05 17:11:32	security group of ACS		修改 克隆 管理实例	还原规则 配置规则
镜像	sg-bp1fvnzjphkag22vtcem alicloud-cs-auto-creat	vpc-	2	专有网络	2017-12-05 14:55:30	security group of ACS		修改 克隆 管理实例	· 还原规则 配置规则
▼ 网络和安全 1 弹性网卡	sg-bp12r6zsuwx47pm91tpr alicloud-cs-auto-creat	vpc-	0	专有网络	2017-12-05 13:52:38	security group of ACS		修改 克隆 管理实例	· 还原规则 配置规则
安全组 2	sg-bp12ma406e6eii7v72ty alicloud-cs-auto-creat	vpc-	2	专有网络	2017-12-04 15:26:49	security group of ACS		修改 克隆 管理实例	还原规则

 如在入方向列表中没有符合要求的安全组规则,则可单击添加安全组规则,在弹出的添加安全 组规则对话窗口进行设置。以下是一个参考设置示例:

添加爱	全组规则		? ×
	网卡类型:	内网	
	规则方向:	入方向	
	授权策略:	允许 •	
	协议类型:	自定义 TCP V	
	★ 端口范围:	30000/32767	
	优先级:	1	
	授权类型:	地址段访问 ▼	
	*授权对象:	0.0.0/0	
	描述:		
			确定取消

- ・ 为 规则方向 选择 入方向。
- ·为端口范围根据区块链网络的外部端口 NodePort 范围选择合适的端口范围区间。
- ・为 授权对象 根据实际访问需要选择合适的地址范围。
- 6. 单击 确定 之后,安全组规则便被成功添加到列表内,并且立即生效。

1.10 以CLI方式访问区块链

背景信息

区块链解决方案在部署区块链网络过程中创建了 CLI 容器,主要用于以命令行方式连接到区块链网络的 Peer 和 Orderer 节点,执行 Hyperledger Fabric 支持的 CLI 命令,可以满足对区块链网络的测试、管理等需求。

登录 CLI 容器的方式主要是在支持 kubectl 命令的环境中,运行 kubectl exec -it < fabricNetwork>-fabric-cli -n <namespace名称> bash 命令进入 CLI 容器。在该 CLI 容器中,我们提供了基于 Hyperledger Fabric 的标准端到端 CLI 测试脚本,如果用户需要修改该测试脚本,可从任意一个 ECS 节点上的文件路径 /data/fabric/<########>/cli/cli-test.sh 找到该文件。

操作步骤

- 1. 用 root 账户以 SSH 方式登录 Kubernetes 集群的 master 节点 (获取地址方式请参 见#unique_12),或者在一个支持 kubectl 远程管理 Kubernetes 集群的环境。
- 2. 运行命令进入CLI 容器: kubectl exec -it -n <namespace名称> <fabricNetwork>fabric-cli bash,例如: kubectl exec -it -n network01 <fabricNetwork>fabric-cli bash。
- 3. 运行以下测试脚本: ./cli-test.sh, 然后 CLI 测试便开始运行。
- 测试过程中在每一步完成后会暂停,以方便用户查看执行过程和结果,然后在用户按下任意键之后继续测试步骤。当测试脚本成功执行完毕后,可以见到类似以下的输出信息:

▋ 说明:

对同一套区块链网络,CLI示例和应用程序示例无法同时运行,请为每一套区块链网络仅选择 一种类型的示例运行。

1.11 以应用程序方式访问区块链

前提条件

- · 在Kubernetes集群上完成区块链网络的配置和部署。
- ・如应用程序部署于容器集群外,则需要完成 配置公网IP和外部端口

背景信息

利用阿里云容器服务区块链解决方案创建了区块链网络之后,用户可使用基于 Hyperledger Fabric SDK 的区块链应用程序访问区块链网络上的服务。此外,区块链解决方案从Hyperledger Fabric 1.1版本开始支持Connection Profile功能。

- · 区块链应用程序可与区块链网络一同部署于阿里云容器集群上,在这种模式下,应用程序可直接
 通过区块链各个服务的名称加上服务端口来实现服务的连接访问。
- · 区块链应用也可以部署于阿里云容器集群之外,在这种模式下,应用程序可通过区块链网络的外部访问地址加上各个服务的外部端口来实现服务的连接访问。

本示例使用了上述的第二种方式进行演示,提供的应用程序是基于 Hyperledger Fabric的 balance transfer 转账应用进行适配性调整而成的。

用户可直接使用本示例提供的应用程序,也可以使用 Hyperledger Fabric 的官方示例应用程序(如 fabric-samples)或者自己开发的区块链应用程序,并参考本示例应用的源代码进行适配 性的调整,即可实现对阿里云容器服务区块链网络的访问。

对已有的区块链应用程序的适配性调整包括:

- 直接使用示例代码中提供的脚本 download-from-fabric-network.sh,实现从新部署的区 块链网络一键自动化下载区块链网络配置,包括证书、密钥、区块链网络配置文件(主要是 config.json和network-config.yaml);
- ·确保区块链应用程序可以正确加载到上述区块链网络配置;
- 如区块链应用程序中直接使用了区块链通道(channel)名称、区块链外部访问地址、节点名称
 或域名等信息,请按照配置文件(config.json和network-config.yaml)的参数进行替换以
 保持与目标区块链网络的一致。

操作步骤

1. 下载示例区块链应用程序源代码到本地开发环境。命令如下:

git clone https://github.com/AliyunContainerService/solutionblockchain-demo.git

2. 按照示例区块链应用程序的README文档,完成后续操作。

中文版: https://github.com/AliyunContainerService/solution-blockchain-demo/ blob/master/balance-transfer-app/README.cn.md

英文版: https://github.com/AliyunContainerService/solution-blockchain-demo/ blob/master/balance-transfer-app/README.md



对同一套区块链网络,CLI示例和应用程序示例无法同时运行,请为每一套区块链网络仅选择一种类型的示例运行。

1.12 以管理监控工具方式访问区块链

前提条件

- ·使用容器服务区块链解决方案创建阿里云上的区块链网络。详情请参考#unique_29文档。
- 使用应用程序的方式或者CLI方式,访问区块链网络,完成端到端测试。详情请参 考#unique_23 或以应用程序方式访问区块链文档。

背景信息

利用阿里云容器服务区块链解决方案创建了区块链网络之后,用户可使用基于 Hyperledger Fabric SDK 的区块链管理监控工具(以下简称"管控工具")访问区块链网络上的服务。

- ·管控工具可与区块链网络一同部署于阿里云容器集群上,在这种模式下,管控工具可直接通过区 块链各个服务的名称加上服务端口来实现服务的连接访问。
- ·管控工具也可以部署于阿里云容器集群之外,在这种模式下,管控工具可通过区块链网络的外部 访问地址加上各个服务的外部端口来实现服务的连接访问。

本示例使用了上述的第二种方式进行演示,提供的管控工具是基于 Hyperledger Explorer 进行适配性调整而成的。在区块链网络部署的同时将默认一同部署 Hyperledger Explorer 到 Kubernetes 集群上。

您可以直接使用本示例提供的管控工具,也可以使用 Hyperledger Explorer 的官方版本、自己 开发的管控工具、或者第三方的管控工具,并参考本示例管控工具的源代码进行适配性的调整,即 可实现对阿里云容器服务区块链网络的连接访问。

对已有的区块链管控工具的适配性调整包括:

- • 直接使用示例代码中提供的脚本 download-from-fabric-network.sh,实现从新部署的区块 链网络一键自动化下载区块链网络配置,包括证书、密钥、区块链网络配置文件(主要是config .json和network-config.yaml)。
- ·确保区块链管控工具可以正确加载到上述区块链网络配置。
- 如区块链管控工具中直接使用了区块链通道(channel)名称、区块链外部访问地址、节点名称
 或域名等信息,请按照配置文件(config.json和network-config.yaml)的参数进行替换以
 保持与目标区块链网络的一致。

操作步骤

在 Kubernetes 集群的 master 节点上运行 kubectl get svc 命令,或者登录 容器服务控制台,进入 kubernetes 集群的控制台,单击左侧导航栏中的 服务,然后查看 <网络名称>-explorer 服务的 EXTERNAL-IP (外部端点)。

6)	管理控制台	产品与服务 🗸					Q 搜索	<u> </u>	费用	工单	备案	企业	支持	yushan.ys		体中文
Ξ	命名空间		■ 服务发现与负载均衡 服	务 netw	ork01-explorer								编辑	∎ 删除	+	创建
-	network01 👻															
=	概况		详情													
¥	工作负载		名称: network01-explorer				连接									
*	定时任务		命名空间: network01				集群 IP: 172	.19.2.3		101 00 700						
	守护进程集	1	创建时间: 2018-03-20110:54 01C 标签选择器: app: network01-exp	lo			内部編点: ne	etwork01-e: etwork01-e:	xpiorer.netwo	ork01:80 TCP ork01:30006 T	CP					
<i></i>	部署		类型: LoadBalancer				外部端点: 🚛	100.00.0	2							
4	任务		保持会话: None													
*	容器组		滞占													
	副本集		AND INC								_					
භ	副本控制器		王利		端口(名标、端口、协议)		市点	aanhan i w	=0:1 <i>flue</i> 1 = 7line	ant/Coo	El	1533省				_
*	有状态副本集		1/2		<01581>, 6060, 10P		CII-SI	Ienznen.i-w	29111601271311	IIIZKOBB	u	ne				
8	服务发现与负载均衡		容器组													咨询
4,5	访问权		名称		节点	状态		已重启	已创建	CPU (核)		内	存(字节)			· 建
© 	服务		network01-explorer-deployme	nt-59df95859f-	cn-shenzhen.i-	Running		0	2018-03- 20		0.022		94	.344 Mi	=	: 2
	配置与存储				wz911tkp1z/ljmnzk6ee	-			18:54:24							

2. 在网络浏览器中访问上述 EXTERNAL-IP(外部端点)。

GER DRER	DASHBOARD	NETWORK BLOCKS TRANSACTIONS CHAINCODES CHANNELS bankchannel -
S BLOCKS	5 TRANSACTIONS	7 1 NODES CHAINCODES
Peer Name	Status	BLOCKS / HOUR BLOCKS / MIN TX / HOUR TX / MIN
network01-peer1	RUNNING	10
network01-peer2	RUNNING	6
network01-peer3	RUNNING	3
network01-peer4	RUNNING	0 S00 PM 800 PM 11:00 PM 2:00 AM 3:00 AM 8:00 AM 11:00 AM 3:00 PM
		Transactions by Organization
Channel Name: bankchannel Datatash: 0458d937e466tc1a4933ae52d46822 Number of Tic 1 3 hours ago	450afc4a880c1dc6075809cea06cc9cf57	
Block 3 Channel Name: bankchannel Datahash: 2dda76e925ab5f7efa352d34dec77b	> (c) <	
	Hyperledger Explorer Client Ve	ersion: 0.3.7 Fabric Compatibility: v1.2



说明:

Hyperledger Explorer 仍处于项目孵化(incubation)阶段,因此其功能仍有待完善,用 户可自行关注其 官方项目 进展以获得其后续功能和版本的更新。

1.13 清理区块链环境

在完成相应开发、测试任务后,如不再需要已部署的区块链网络,或者需要重新部署区块链网络,可参照以下操作步骤对相应的区块链环境进行清理。

在发布界面删除区块链网络

- 1. 登录容器服务管理控制台。
- 2. 在 Kubernetes 菜单下,在左侧导航栏单击应用 > Helm,选择所需的集群,选择区块链网络 对应的发布名称,单击删除。

容器服务 - Kubernetes +	发布							周川新
授权管理	分批发布 Helm							
<u>• 应用</u>	無群 k8s-blockchain ▼							
无状态	发布名称	状态	命名空间	Chart 名称	Chart 版本	应用版本	更新时间	操作
有状态	ack-hyperledger-fabric-default	E##	default	ack-hyperledger-fabric	0.2.3	1.4.0	2019-03-11 12:08:44	洋橋 4 题题
守护进程集								
任务								
定时任务								
容器组								
存储声明								
发布 2 Ξ								

3. 在弹出的删除应用对话窗口,单击确定。

删除应用			\times
•	确定要删除应用 ack-hyperledger-fabric-default ?		
		确定	取消

用 Helm 删除区块链网络

- 1. 用 root 账户以 SSH 方式登录 Kubernetes 集群的 master 节点。
- 2. 运行 helm list 查看区块链网络对应的 Helm release 名称。
- 3. 运行 helm delete --purge <区块链网络的Helm release名称> 删除区块链网络,例如 helm delete --purge blockchain-network01。

等待数分钟(取决于区块链网络节点数量)直到 helm delete 命令完成并返回。这样便完成了将区 块链所有节点对应的服务和容器从 Kubernetes 集群中删除的操作。此外,我们也同时自动删除了 所有 worker 节点上与此区块链网络相关的chaincode容器。

关于区块链数据目录

在上述区块链网络删除的过程中,区块链网络在共享文件存储上的数据目录已经被自动清理,以便 于再次创建区块链网络。为安全起见,清理的方法是在原目录名基础上添加"-deleted-时间戳"的 后缀,例如:"-deleted-2018-03-17-160332"。这样在未来如有需要,我们仍可以通过删除后缀 的方式来重用此数据。如需彻底删除,我们可采用手动rm命令或者结合自动化脚本的方式做定期清 理以释放存储空间。

如需访问或清理区块链数据目录,可使用如下命令示例将NAS文件系统挂载到ECS中。

```
mkdir /data
mount -t nfs -o vers=4.0 987a6543bc-abc12.cn-hangzhou.nas.aliyuncs.com
:/ /data #注意替换为您的NAS挂载地址
```

1.14 区块链网络重启和数据恢复

容器服务区块链解决方案支持对区块链网络进行重启、同时可复用原有区块链配置和账本数据等信息。

前提条件

- · 您已创建一个Kubernetes集群,参见#unique_32。
- · Kubernetes集群中已有一套运行中的区块链网络,参见#unique_33。
- · 您已通过SSH连接到Kubernetes集群,参见#unique_34。

操作步骤

- 1. 用root账户以SSH方式登录Kubernetes集群的Master节点。
- 执行helm delete --no-hooks --purge <区块链网络的Helm Release名称>命令,删除
 区块链网络。



这里的--no-hooks参数可避免删除原有数据目录。

```
helm delete --no-hooks --purge network01
#本例中Helm Release名称为network01
```

```
release "network01" deleted
```

 使用同样的区块链网络名称(即相同的fabricNetwork变量值),在容器服务控制台或Helm命 令行创建新的区块链网络,参见#unique_35,这样便能复用原有的数据目录。本例中 以Helm命令行为例。

```
helm install --name network01 --set "sharedStorage=029bb489d2-ikw80.
cn-hangzhou.nas.aliyuncs.com" incubator/ack-hyperledger-fabric
#替换为您的NAS挂载地址
```

如需备份区块链数据目录,可使用如下命令示例将NAS文件系统挂载到ECS中,这里假设区块链 网络名称为network01。

mkdir /data mount -t nfs -o vers=4.0 987a6543bc-abc12.cn-hangzhou.nas.aliyuncs. com:/ /data #注意替换为您的NAS挂载地址

```
📋 说明:
```

现在/data/fabric/network01即为您需要备份的数据目录。

使用以上方法,可实现对一套已有区块链网络的数据备份、迁移、恢复的目的。

2 深度学习解决方案

2.1 基于helm chart的深度学习解决方案

2.1.1 概述

本文主要为您介绍TensorFLow 深度学习解决方案。

背景信息

基于阿里云强大计算能力的深度学习解决方案,为您提供一个低门槛、开放、端到端的深度学习服 务平台。方便数据科学家和算法工程师快速开始利用阿里云的资源(包括 ECS 云服务器、GPU 云 服务器、高性能计算 HPC、文件存储 NAS、Elastic MapReduce、负载均衡等服务)执行数据准 备、模型开发、模型训练、评估和预测等任务。并能够方便地将深度学习能力转化为服务 API,加 速与业务应用的集成。

TensorFlow 是业界最流行的深度学习框架,但是如何将 TensorFlow 真正运用于生产环境却并不简单,它面临着资源隔离,应用调度和部署,GPU 资源分配,训练生命周期管理等挑战。特别是 大规模的分布式训练场景,单靠手动部署和人力运维已经无法有效处理。特别启动每个模块都需要 指定好分布式集群的 clusterSpec。

阿里云 Kubernetes 深度学习解决方案有效应对在负载均衡、弹性伸缩、高可用性以及滚动升级方面的挑战,利用 Kubernetes 的内置自动化能力,将极大地降低 TensorFLow 应用的运维成本。

具体而言,该深度学习解决方案具备以下特性:

- · 简单: 降低构建和管理深度学习平台的门槛。
- · 高效:提升 CPU、GPU 等异构计算资源的使用效率,提供统一的用户体验。
- · 全周期:提供基于阿里云强大服务体系构建端到端深度学习任务流程的最佳实践。
- ・服务化:支持深度学习能力服务化,与云上应用的轻松集成。

开始使用

- 1. 环境准备。
 - · 创建 GN5 型Kubernetes 集群。
 - ・#unique_40,本示例使用 NAS 数据卷。
- 2. #unique_41, 开发模型。
- 3. 运行TensorFlow 分布式模型训练,导出模型。

4. 利用导出的模型,执行 TensorFlow模型预测。

2.1.2 创建 GN5 型Kubernetes 集群

下面将介绍如何在阿里云容器服务上创建 Kubernetes GPU 集群。

前提条件

您需要申请一个按量付费的GPU计算型gn5。请 提交 ECS 工单申请开通。

背景信息

Kubernetes 深度学习解决方案支持使用云服务器 ECS 的 Kubernetes 集群或者 GPU 服务器 Kubernetes 集群。本文档以 GPU 服务器容器集群为例进行说明。



有关如何创建 ECS 容器集群,参见#unique_44。

而 Kubernetes 全新的 GPU 调度方案基于 Nvidia 官方的设备插件和 nvidia-containerruntime,和之前社区方案相比,最终用户所要做的配置更少。

基于该方案,客户可以将应用程序利用容器技术构建镜像,结合 Kubernetes+GPU 运行机器学 习,图像处理等高运算密度等任务,无需安装 nvidia driver 和 CUDA,就能实现一键部署和弹性 扩缩容等功能。

使用限制

- ・目前,gn5型 GPU 云服务器只支持专有网络(VPC)。
- ·用户账户需有100元的余额并通过实名认证,否则无法创建按量付费的ECS实例和负载均衡。
- · Kubernetes 深度学习解决方案要求 Kubernetes 集群的版本在 1.9.3 及以上。

操作步骤

- 1. 登录容器服务管理控制台。
- 2. 在 Kubernetes 菜单下,单击左侧导航栏的集群 > 集群,进入集群列表页面。
- 3. 单击页面右上角的 创建 Kubernetes 集群。

容器服务 - Kubernetes ▼	集群列表	您最多可以创建 5 个集	群,每个集群最多可以添加 40	个节点 刷新	申请使用Serverless Kubernetes集群	创建 Kubernetes 集群 🛛 👻
概览	常见问题: & 如何创建集群 & 扩容和缩	集群 🔗 通过 kubecti 连接 Kubernetes 集	群 🔗 通过命令管理应用			2
* ^{集群}	名称 ▼					
集群	集群名称/ID	集群类型 地域(全部) ▼ 网络类	2 集群状态	创建时间	Kubernetes 版本	操作

4. 配置集群基本信息。

* 集群名称	test-gpu				
	名称为1-63个字符,可	包含数字、汉字、英文	字符 , 或"-"		
地域	华北 1	华北 2	华北 3	华北 5	华东 1
	华东 2	华南 1	香港	亚太东北 1 (东京)	亚太东南1(新加坡)
	亚太东南 2 (悉尼)	亚太东南 3 (吉隆坡)	亚太东南 5 (雅加达)	亚太南部 1 (孟买)	美国东部 1 (弗吉尼亚)
	美国西部1(硅谷)	中东东部 1 (迪拜)	欧洲中部 1 (法兰克福)		
可用区	华东 1 可用区 B		•		
专有网络	自动创建	使用已有			
节点类型	按量付费	包年包月			

5. 为 worker 节点选择实例系列,本例中选择 GPU 计算型 gn5。

MASTER 配置	
实例规格	4 核 8 G (ecs.n1.large) ▼ ▼ 数量 3台
系统盘	SSD云盘 ▼ 40 GiB ◆
Worker 实例	新增实例 添加已有实例 添加已有实例 总目前可以通过 ECS 管理控制台将按量付费实例转换成包年包月实例。查看详情
WORKER 配置	
实例规格	2 核 8 G (ecs.gn5i-c2g1.large) 🔹 🔻 🕉 🛣
系统盘	高效云盘 ▼ 40 GiB 🜩
挂载数据盘	
登录方式	设置密钥 设置密 码
* 登录密码	••••••• 8 - 30 个字符,且同时包含三项(大、小写字母,数字和特殊符号)
* 确认密码	••••••

6. 勾选开放公网 SSH 登录,这样就可以通过 SSH 登录 Kubernetes 的 Master 节点。

SSH登录	✓ 开放公网SSH登录
	选择不开放时,如需手动开启 SSH 访问,请参考 SSH 访问 Kubernetes 集群
门 说明:	
其他集群参数的	D详细配置信息,请参见 #unique_44。

7. 完成配置后, 单击 创建集群, 等待一段时间, 新建的 GPU 集群会出现在集群列表中。

容器服务 - Kubernetes ▼	集群列表	您	最多可以创建 5 个集群,每个	、集群最多可以添加 40	个节点 刷新	申请使用Serverless Kubernetes集群	创建 Kubernetes 集	₩ -
概览	常见问题: ③ 如何创建集群 ⑧ 扩容和缩	容集群 🔗 通过 kubect	」 连接 Kubernetes 集群 🔗	通过命令管理应用				
集群	名称 ▼							
集群	集群名称/ID	集群类型 地域 (全部) 👻 网络炭型	集群状态	创建时间	Kubernetes 版本	2	操作
节点	test-gpu 3	Kubernetes 华东1	虚拟专有网络 vpc-bp1kd7yn4	qn ●运行中	2018-08-06 15:36:05	1.10.4	管理 查看日志 集群伸缩	控制台 更多▼

8. 单击集群右侧的 管理,进入集群基本信息页面,查看 Master节点 SSH 连接地址。

<	集群 : test-gpu						
基本信息	基本信息	基本信息					
	集群ID: c794156934f14479c8e4aa62f190cb2e3	虚拟专有网络	●运行中	地域: 华东1			
	连接信息						
	API Server 公网连接端点	https://					
	API Server 内网连接端点	https://					
	Master 节点 SSH 连接地址	47.97.2					
	服务访问域名	*.c7941	-hangzhou.alicontainer.com				

9. SSH 登录到 Master节点,可通过执行如下命令,查找集群下的 GPU 节点。

```
$ kubectl get nodes -l 'aliyun.accelerator/nvidia_name' --show-
labels
 . . .
NAME
                                       STATUS
                                                 ROLES
                                                           AGE
VERSION
         LABELS
cn-hangzhou.i-bp12xvjjwqe6j7nca2q8
                                      Ready
                                                 <none>
                                                           1h
            aliyun.accelerator/nvidia_count=1,aliyun.accelerator/
   v1.9.3
nvidia_mem=16276MiB,aliyun.accelerator/nvidia_name=Tesla-P100-PCIE-
16GB, ..
```

10.查看 GPU 节点的详细状态信息。

```
$ kubectl get node ${node_name} -o=yaml
...
status:
   addresses:
   - address: 172.16.166.23
   type: InternalIP
   allocatable:
     cpu: "8"
     memory: 61578152Ki
     nvidia.com/gpu: "1"
```

```
pods: "110"
capacity:
cpu: "8"
memory: 61680552Ki
nvidia.com/gpu: "1"
pods: "110"
```

至此, 创建的 Kubernetes 的 GPU 集群已经创建完毕。

2.1.3 TensorFlow模型预测

本文将介绍如何利用 Kubernetes 的官方包管理工具 Helm 在阿里云容器服务上准备模型,部署 TensorFlow Serving,并且进行手动扩容。

背景信息

TensorFlow Serving 是由谷歌开源的机器学习模型预测系统,能够简化并加速从模型到生产应用 的过程。它可以将训练好的机器学习模型部署到线上,使用 gRPC 作为接口接受外部调用。更给人 惊喜后的是,它还提供了不宕机的模型更新和版本管理。这大大降低了模型提供商在线上管理的复 杂性,可以将注意力都放在模型优化上。

TensorFlow Serving 本质上也是一个在线服务,我们需要考虑它的部署时刻的安装配置,运行时 刻的负载均衡,弹性伸缩,高可用性以及滚动升级等问题,幸运的是这正是 Kubernetes 擅长的 地方。利用 Kubernetes 的内置自动化能力,将极大地降低 TensorFLow Serving 应用运维的成 本。

准备工作

在运行模型预测任务之前,请确认以下工作已经完成:

- · 创建包含适当数量弹性计算资源(ECS或EGS)的Kubernetes集群。创建步骤请参考#unique_46。
- 如果您需要使用 NAS 文件系统保存用于模型训练的数据,您需要使用相同账号创建 NAS;然后 在上面的 Kubernetes 集群中创建持久化数据卷(PV),动态生成 PVC 作为本地目录挂载到 执行训练任务的容器内。参见#unique_47。
- · SSH 连接到 Master 节点连接地址,参见 #unique_48。

步骤1 准备模型

由于 TensorFLow Serving 需要用持久化存储加载预测模型,这里就需要准备相应的存储。在阿 里云容器服务里,您可以选择 NAS、OSS和 云盘,具体可以参考#unique_49。

本文以 NAS 存储为例介绍如何导入数据模型。

1. 创建 NAS 文件存储,并且设置 VPC 内挂载点。

请参见 #unique_50/unique_50_Connect_42_section_9q8_owp_z6n, 查看添加挂载

点,这里假设挂载点为 xxxxxx.cn-hangzhou.nas.aliyuncs.com。

2. 利用集群内的一台阿里云 ECS 云服务器准备模型数据,请执行如下命令,创建文件夹。

```
mkdir /nfs
mount -t nfs -o vers=4.0 xxxxxx.cn-hangzhou.nas.aliyuncs.com:/ /nfs
mkdir -p /nfs/serving
umount /nfs
```

3. 下载预测模型并且保存到 NAS 中。

```
mkdir /serving
mount -t nfs -o vers=4.0 xxxxxx.cn-hangzhou.nas.aliyuncs.com:/
serving /serving
mkdir -p /serving/model
cd /serving/model
curl -0 http://tensorflow-samples.oss-cn-shenzhen.aliyuncs.com/
exports/mnist-export.tar.gz
tar -xzvf mnist-export.tar.gz
rm -rf mnist-export.tar.gz
cd /
```

此时可以看到预测模型的内容,检查后可以 umount 挂载点。

步骤2 创建持久化数据卷

以下为创建 NAS 的 nas.yaml 样例。

1. 创建并拷贝以下内容到nas.yaml文件中。

```
apiVersion: v1
kind: PersistentVolume
metadata:
  labels:
    model: mnist
  name: pv-nas
spec:
  persistentVolumeReclaimPolicy: Retain
  accessModes:

    ReadWriteMany

  capacity:
    storage: 5Gi
  flexVolume:
    driver: alicloud/nas
    options:
      mode: "755"
```

path: /serving/model/mnist
server: xxxxxx.cn-hangzhou.nas.aliyuncs.com
vers: "4.0"

2. 执行以下命令,创建持久化数据卷。

```
kubectl create -f nas.yaml
```

```
persistentvolume "pv-nas" created
```



注意这里需要指定 label 为 model: mnist,该标签对于 pvc 选择 pv 绑定非常重要。另外和 NAS 相关的具体配置可以参见#unique_51。

您也可以通过控制台创建数据卷,请参见#unique_52。

步骤3 通过 Helm 部署 TensorFlow Serving 应用

- 1. 登录容器服务管理控制台。
- 2. 在 Kubernetes 菜单下,单击左侧导航栏中的 市场 > 应用目录,进入应用目录列表。
- 3. 单击 ack-tensorflow-serving, 进入对应的 chart 页面。



4. 单击参数,对 ack-tensorflow-serving 的参数进行配置,最后单击创建。

容器服务	应用目录 - ack-tensorflow-serving	
Kubernetes Swarm		
▼ 集群	→ O ack-tensorflow-serving	
集群	incubator	
节点	TensorFlow Serving Helm chart for Kubernetes on Alibaba Cloud Container Service	
存储		
▼ 动田		
	1 + Default valuer for activerantian	创建
服务	 a Poliatic values for access relation representation. a Philip Star VALL-formatted file. b Declare variables to be passed into your templates. 	仅支持 Kubernetes 版本 1.8.4 及以上的集群。对于 1.8.1 版本的 集群,您可以在集群列表中进行"集群升级"操作
发布	5 - ## Kubernetes configuration 6 ## support NodePort, LoadBalancer	集群
配置项	7 ## 8 serviceType: LoadBalancer	test-gpu 🔻
★ #K	9 10 ## expose the service to the groc client	命名空间
1040	11 port: 9090 12 reolicas: 1	default 🔻
应用目录	13 = 14 - ## Tensorflow server image version	发布名称
服务目录	15 ## if gpuCount>0, the default image is registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tensorflow-serving:1.4 .8-devel-enu in the regions of China excent Honekone.	ack-tensorflow-serving-default
	<pre>16 ## registry.us-east-1.aliyuncs.com/tensorflow-samples/tensorflow-serving:1.4.0-devel-gpu for other regions including Honekong.</pre>	创建
	17 ## if gpuCount=0, the default image is registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tensorflow-serving:1.4 .8-devel in the regions of China except Honekone.	
	18 ## registry.us-east-1.aliyuncs.com/tensorflow-samples/tensorflow-serving:1.4.0-devel for other regions including Honorong	2
	19 ## You can also bring up your own docker image. 20 image: "registry on-bangthup aligners com/tensorflow-samples/tensorflow-serving:1 4 0-devel-gou"	版本
	21 imagePullPolicy: "IfNotPresent" 22 ## the gau recourse to claim for you, change it to 0	0.1.0
	23 gpuCount: 1 24	项目主页
	25 26 ## The command and args to run the nod	链接
	<pre>27 command: ["/usr/bin/tensorflow_model_server"] 28 acres: ["port=9999". "model_page=mist" "model_base_path=/serving/model/mnist"]</pre>	
	29 ## the mount path inside the container 30 mountPath: /serving/model/mitt	
	31 32 persistence:	
	33 - ## The request and label to select the persistent volume	
	35 storage: 56i	
	37	

· 创建支持 GPU 的自定义配置参数:

```
serviceType: LoadBalancer
 ## expose the service to the grpc client
port: 9090
 replicas: 1
 image: "registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/
tensorflow-serving:1.4.0-devel-gpu"
    imagePullPolicy: "IfNotPresent"
 ## the gpu resource to claim, for cpu, change it to 0
gpuCount: 1
## The command and args to run the pod
command: ["/usr/bin/tensorflow_model_server"]
args: [ "--port=9090", "--model_name=mnist", "--model_base_path=/
serving/model/mnist"]
 ## the mount path inside the container
mountPath: /serving/model/mnist
persistence:
 ## The request and label to select the persistent volume
    pvc:
       storage: 5Gi
       matchLabels:
         model: mnist
```

· 创建支持非 GPU 的自定义配置参数:

```
image: "registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/
tensorflow-serving:1.4.0-devel"
imagePullPolicy: "IfNotPresent"
mountPath: /serving/model/mnist
persistence:
    mountPath: /serving/model/mnist
    pvc:
        matchLabels:
            model: mnist
            storage: 5Gi
```

也可以登录到 Kubernetes master 运行以下命令。

```
helm install --values serving.yaml --name mnist incubator/acs-
tensorflow-serving
```

步骤4 查看TensorFlow-serving的应用部署

登录到 Kubernetes 的 master 上利用 helm 命令查看部署应用的列表。

# helm list						
NAME	REVISION	UPDATED			STATUS	CHART
	NAM	ESPACE				
mnist-deploy	1	Fri Mar 16	19:24:35	2018	DEPLOYED	acs-
tensorflow-ser	ving-0.1.	0 default				

利用 helm status 命令检查具体应用的配置。

```
# helm status mnist-deploy
LAST DEPLOYED: Fri Mar 16 19:24:35 2018
NAMESPACE: default
STATUS: DEPLOYED
RESOURCES:
==> v1/Service
NAME
                                     TYPE
                                                   CLUSTER-IP
EXTERNAL-IP
               PORT(S)
                               AGE
mnist-deploy-acs-tensorflow-serving LoadBalancer 172.19.XX.XX 139.
195.XX.XX 9090:32560/TCP 5h
==> v1beta1/Deployment
                      DESIRED
                               CURRENT
                                        UP-TO-DATE AVAILABLE
NAME
                                                               AGE
mnist-deploy-serving 1
                               1
                                        1
                                                    1
                                                               5h
==> v1/Pod(related)
                                       READY
                                              STATUS
                                                       RESTARTS
NAME
                                                                 AGE
mnist-deploy-serving-665fc69d84-pk9bk 1/1
                                              Running
                                                       0
                                                                 5h
```

TensoFlow Serving 的对外服务地址是*EXTERNAL-IP*: 139.195.1.216, 端口为 9090 对应 部署的 deployment 是 mnist-deploy-serving, 这个信息在扩容时刻是需要的。 查看 tensorflow-serving 的下 pod 的日志,发现 mnist 的模型已经加载到内存里,并且 GPU 已经正常启动。

kubectl logs mnist-deploy-serving-665fc69d84-pk9bk

2.1.4 TensorFlow 分布式模型训练

本文是一个利用 Helm 运行端到端的分布式模型训练示例。

背景信息

TensorFlow 是业界最流行的深度学习框架,如何将 TensorFlow 真正运用于生产环境却并不简 单,它面临着资源隔离,应用调度和部署,GPU资源分配,训练生命周期管理等挑战。特别是大规 模的分布式训练场景,单靠手动部署和人力运维已经无法有效处理。特别启动每个模块都需要指定 好分布式集群的 clusterSpec,使得实现更加困难。

在 Kubernetes 集群上运行分布式 TensorFlow 模型训练,可以依靠 Kubernetes 本身在应用调度,GPU资源分配,共享存储等方面的能力,实现训练任务和参数服务器的调度以及生命周期的管理。同时利用共享存储查看训练的收敛程度,调整超参。

但是手动写部署 Yaml 对于最终用户来说过于繁杂,阿里云容器服务提供了基于 Helm 的 TensorFlow 分布式训练解决方案:

- ・同时支持 GPU 和非 GPU 的集群。
- · 不再需要手动配置 clusterspec 信息,只需要指定 worker 和 ps 的数目,能自动生成 clusterspec。
- · 内置 Tensorboard 可以有效监控训练的收敛性,方便快速调整参数 epoch、batchsize、 learning rate。

准备工作

在运行模型训练任务之前,请确认以下工作已经完成:

- ・ 创建包含适当数量弹性计算资源(ECS 或 EGS)的 Kubernetes 集群。创建步骤请参 考#unique_46。
- 如果您需要使用 NAS 文件系统保存用于模型训练的数据,您需要使用相同账号创建 NAS;然后 在上面的 Kubernetes 集群中创建持久化数据卷(PV),动态生成 PVC 作为本地目录挂载到 执行训练任务的容器内。参见#unique_40。
- · SSH 连接到 Master 节点连接地址,参见#unique_48。

步骤1 准备数据

 在前面的准备环节,已经创建了一个 NAS 文件系统,并且设置 VPC 挂载点,参 见#unique_40。本例的挂载点为 xxxxxxx.cn-hangzhou.nas.aliyuncs.com。 2. 配置名为 /data 的数据文件夹。

```
mkdir /nfs
mount -t nfs -o vers=4.0 xxxxxxx.cn-hangzhou.nas.aliyuncs.com:/ /
nfs
mkdir -p /nfs/data
umount /nfs
```

步骤2 创建持久化存储

1. 以下为创建 NAS 的 nas.yaml 样例,实际上您也可以创建云盘或者 OSS 等持久化存储。

这里需要指定 label 为 train:mnist,该标签对于 pvc 选择 pv 绑定非常重要。其他和 NAS 相关的具体配置可以参考#unique_49。

```
apiVersion: v1
kind: PersistentVolume
metadata:
   labels:
     train: mnist
   name: pv-nas-train
spec:
   persistentVolumeReclaimPolicy: Retain
   accessModes:
     - ReadWriteMany
   capacity:
     storage: 5Gi
   flexVolume:
     driver: alicloud/nas
     options:
mode: "755"
       path: /data
       server: XXXX.cn-hangzhou.nas.aliyuncs.com
       vers: "4.0"
```

2. SSH 连接到 Master 节点,运行 kubectl 命令创建 pv。

```
$ kubectl create -f nas.yaml
persistentvolume "pv-nas-train" created
```

3. 部署完成后,可以通过 Kubernetes Dashoard 检查运行状态。

<	三 集群 持久化存储	诸卷								+ 创建
集群	持久化存储卷									Ŧ
#名空间 节点	名称 🗢	总量	访问模式	回收策略	状态	素取	存储类	原因	已创建 ≑	
持久化存储卷	🔮 pv-nas-train	5Gi	ReadWriteMany	Retain	Available	-	-	-	2018-04-17 17:49:48	:
角色										
存储类										
命名空间										
default 👻										

步骤3 通过 Helm 部署 TensorFlow 分布式训练的应用

- 1. 登录容器服务管理控制台。
- 2. 在 Kubernetes 菜单下,单击左侧导航栏中的 市场 > 应用目录,进入应用目录列表。

C

- | 应用目录 容器服务 - Kubernetes ▼ 概览 $\mathbf{O}_{\mathrm{O}}^{\mathrm{O}}$ \mathbf{Q}_{O}^{O} \mathbf{Q}_{O}^{O} $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ 集群 集群 节点 ack-hyperledger-fabric ack-istio ack-istio-remote ack-openmpi 1.1.0 incub 1.0.0 incubat 1.0.0 incuba 3.1.0 incub 存储卷 命名空间 $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ **0**0 \mathbf{Q}_{0}^{O} $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ 应用 部署 容器组 ack-springcloud-configserver ack-springcloud-eureka ack-springcloud-hystrix ack-springcloud-turbine 1.5.13.RELEASE incuba 1.5.13.RELEASE incu 1.5.13.RELEASE inc 1.5.13.RELEASE incut 服务 (= 路由 $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ 存储声明 helm ack-tensorflow-serving ack-tensorflow-dev 发布 ack-springcloud-zipkin ack-springcloud-zuul 1.5.13.RELEASE incu 1.5.13.RELEASE incu 1.5.0 incuba 1.4.0 incubato 配置项 保密字典 $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ $\mathbf{O}_{\mathbf{O}}^{\mathbf{O}}$ $\mathbf{O}_{\mathrm{O}}^{\mathrm{O}}$ 市场 1 镜像 ack-tensorflow-training 编排模板 ceph chartmuseum consul 1.4.0 incubato 0.7.1 stable 0.8.3 stable incubato 应用目录 2
- 3. 单击 ack-tensorflow-training, 进入对应的 chart 页面。

4. 单击参数,对 ack-tensorflow-training 的参数进行配置,最后单击部署。

↓	
默认配置下,使用 GPU 进行模型训练。	
应用目录 - ack-tensorflow-training	
Ack-tensorflow-training incubator TensorFlow Training Helm chart for Kubernetes on Alibaba Cloud Container Service	
说明 ● 数 1	部署
<pre>stage: stage: stage:</pre>	(又党特 Kubernetes 該本 1.8.4 及以上的集群, 对于 1.8.1 版本的集群, 空可以在集 群列集中进行"集群升级"强作 第月 第月 ● </th
<pre>image: tensorbaard: tensorbaard: # registry.us-sast-1.aliyons.com/tensorflow-samples/tensorbaard:1.1.8 in the regions of China except Hongkong, # registry.us-sast-1.aliyons.com/tensorflow-samples/tensorbaard:1.1.8 # registry.cn-hagrbo.aliyons.com/tensorflow-samples/tensorbaard:1.1.8 # registry.cn-hagrbo.aliyons.com/tensorflow-samples/tensorflow-samples/tensorbaard:1.1.8 # registry.cn-hagrbo.aliyons.com/tensorflow-samples/tensorbaard:1.1.8 # registry.cn-hagrbo.aliyons.com/tensorflow-samples/tensorflow-samples/tensorflow-samples/tensorflow-samples/tensorflow-samples/</pre>	2 版本 0.1.0 項目主页 链接
55 matchladels: 56 krain: mnist	

以下为支持 GPU 的自定义配置参数的 yaml 文件。

worker: number: 2

```
gpuCount: 1
   image: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tf-
mnist-k8s:gpu
   imagePullPolicy: IfNotPresent
   port: 8000
ps:
   number: 2
   image: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tf-
mnist-k8s:cpu
   imagePullPolicy: IfNotPresent
   port: 9000
hyperparams:
   epochs: 100
   batchsize: 20
   learningrate: 0.001
tensorboard:
   image: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/
tensorboard:1.1.0
   serviceType: LoadBalancer
mountPath: /data
persistence:
    pvc:
       storage: 5Gi
       matchLabels:
         train: mnist
                                                   ##与前面创建的pv的标
签一致
```

如果您运行的 Kubernetes 集群不含有 GPU,可以使用以下配置 yaml 文件。

```
worker:
number: 2
gpuCount: 0
# if you'd like to choose the cusomtized docker image
image: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tf-mnist
-k8s:cpu
imagePullPolicy: IfNotPresent
ps:
number: 2
# if you'd like to choose the cusomtized docker image
image: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tf-mnist
-k8s:cpu
imagePullPolicy: IfNotPresent
tensorboard:
image: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/
tensorboard:1.1.0
serviceType: LoadBalancer
hyperparams:
epochs: 100
batchsize: 20
learningrate: 0.001
persistence:
mountPath: /data
pvc:
 matchLabels:
    train: mnist ##与前面创建的pv的标签一致
```

storage: 5Gi

这里镜像的参考代码来自于: https://github.com/cheyang/tensorflow-sample-code。

您也可运行 helm 命令部署。

```
helm install --values values.yaml --name mnist incubator/acs-
tensorflow-tarining
helm install --debug --dry-run --values values.yaml --name mnist
incubator/acs-tensorflow-tarining
```

5. 部署完成后,单击集群右侧的 控制台,进入 Kubernetes Dashboard,查看应用运行状态。

<	≡ 概況					+ 1932
集群	任务					-
命名空间	名称 ≑	标签	容器组	已创建 ≑	请像	
节点	🔮 ps-0	app: mnist-dist-tr	1/1	2018-04-17 20:35:33	registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tf-mnist-k8	. ≣
持久化存储举	🔮 ps-1	app: mnist-dist-tr	1/1	2018-04-17 20:35:33	registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tf-mnist-k8:	₽ :
角色	🛛 worker-0	app: mnist-dist-tr name: worker	1/1	2018-04-17 20:35:33	registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tf-mnist-k8:	₽ :
存储类						
命名空间	發展过					
default +	名称 \$	节点	状志 🗣	已重启 建 ^{CD} 的 ≑ CPU(核)	内存(字节)	
概況	🔮 ps=0-btkrj	cn-hangzhou.i-bp12xvjjwqe6j7nca2q8	Running	2018-04- 0 17 -	169.75 Mi	≡ :
工作负载	-			20:35:33 2018-04-		
定时任务	S ps-1-75pz2	cn-hangzhou.i-bp12xvjjwqe6j7nca2q8	Running	0 17 - 20:35:33	175.258 Mi	₽ :
守护进程集 部 署	tensorboard-5c785fbd97-g97s4	cn-hangzhou.i-bp12xvjjwqe6j7nca2q8	Running	2018-04- 0 17 - 20:35:33	36.930 MI	₽ :
任务 容難組	S worker-0-jxfwr	cn-hangzhou.i-bp12xvjjwqe6j7nca2q8	Running	2018-04- 0 17 - 20:35:33	856.672 MI	≡ :
副本康	Sack-tensorflow-dev-default-84c7fbb5dc-p7v8g	cn-hangzhou.i-bp12xvjjwqe6j7nca2q8	Running	0 17 0.013 16:49:50	1,348 GI	≡ :
BU-9-17 (7) BF						

步骤4利用 helm 命令查看部署的信息

1. 登录到 Kubernetes 的 master 节点上,利用 helm 命令查看部署应用的列表。

```
# helm list
NAME REVISION UPDATED STATUS
CHART NAMESPACE
mnist-dist-train 1 Mon Mar 19 15:23:51 2018 DEPLOYED acs
-tensorflow-training-0.1.0 default
```

2. 利用 helm status 命令检查具体应用的配置。

```
# helm status mnist-dist-train
LAST DEPLOYED: Mon Mar 19 15:23:51 2018
NAMESPACE: default
STATUS: DEPLOYED
RESOURCES:
==> v1/ConfigMap
                 DATA
NAME
                        AGE
tf-cluster-spec
                 1
                        7m
==> v1/Service
             TYPE
                         CLUSTER-IP
                                        EXTERNAL-IP
                                                       PORT(S)
                                                                  AGE
NAME
worker-0
             ClusterIP
                                                       8000/TCP
                         None
                                         <none>
                                                                  7m
             ClusterIP
                         None
                                         <none>
                                                       9000/TCP
                                                                  7m
ps-1
tensorboard
            ClusterIP
                         172.19.XX.XX
                                         106.1.1.1
                                                       80/TCP
                                                                  7m
             ClusterIP
                         None
                                                       9000/TCP
                                                                  7m
ps-0
                                         <none>
worker-1
             ClusterIP
                         None
                                         <none>
                                                       8000/TCP
                                                                  7m
==> v1beta1/Deployment
NAME
             DESIRED CURRENT
                                UP-TO-DATE AVAILABLE
                                                        AGE
tensorboard
                       1
                                1
                                             1
                                                        7m
             1
==> v1/Job
          DESIRED
                    SUCCESSFUL
NAME
                                AGE
ps-1
          1
                    0
                                7m
```

worker-0	1	Θ	7m			
ps-0	1	Θ	7m			
worker-1	1	Θ	7m			
==> v1/Po	d(related)				
NAME			READY	STATUS	RESTARTS	AGE
tensorboa	rd-5c785f	bd97-7cwk2	1/1	Running	Θ	7m
ps-1-lkbt	b		1/1	Running	Θ	7m
worker-0-	2mpmb		1/1	Running	Θ	7m
worker-0- ps-0-ncxc	2mpmb h		1/1 1/1	Running Running	0 0	7m 7m
worker-0- ps-0-ncxc worker-1-	2mpmb h 4hngw		1/1 1/1 1/1	Running Running Running	0 0 0	7m 7m 7m

这里可以看到 Tensorboard 的对外IP是106.1.1.1,可以在训练过程中查看 cost 的收敛程

度。

3. 检查任务运行状况,此时 worker 都处于运行状态。

# kubectl	get job						
NAME	DESIRED	SUCCESSF	UL /	AGE			
ps-0	1	Θ	ŗ	5m			
ps-1	1	Θ	ŗ	5m			
worker-0	1	Θ	ŗ	5m			
worker-1	1	Θ	ŗ	5m			
# kubectl	get po						
NAME	U .		READ	Y	STATUS	RESTARTS	AGE
ps-0-jndp	d		1/1		Running	Θ	6m
ps-1-b8zg	z		1/1		Running	Θ	6m
tensorboa	rd-f78b4d57	b-pm2nf	1/1		Running	Θ	6m
worker-0-	rqmvl	-	1/1		Running	Θ	6m
worker-1-	7pgx6		1/1		Running	Θ	6m

4. 检查训练日志。

kubectl logs --tail=10 worker-0-rqmvl Step: 124607, Epoch: 24, Batch: 1600 of 2750, Cost: 0.8027, AvgTime: 6.79ms Step: 124800, Epoch: 24, Batch: 1700 of 2750, Cost: 0.7805, AvgTime: 6.10ms

5. 通过 watch job 状态,可以监视到 job 已经完成。

# kubectl	get job		
NAME	DESIRED	SUCCESSFUL	AGE
ps-0	1	Θ	1h
ps-1	1	Θ	1h
worker-0	1	1	1h
worker-1	1	1	1h

此时再查看训练日志,发现训练已经完成。

```
# kubectl logs --tail=10 -f worker-0-rqmvl
Step: 519757, Epoch: 100,
                            Batch: 2300 of 2750, Cost: 0.1770,
AvgTime: 6.45ms
Step: 519950, Epoch: 100,
                            Batch: 2400 of 2750,
                                                  Cost: 0.2142,
AvgTime: 6.33ms
                            Batch: 2500 of 2750,
Step: 520142, Epoch: 100,
                                                  Cost: 0.1940,
AvgTime: 6.02ms
Step: 520333, Epoch: 100,
                            Batch: 2600 of 2750,
                                                  Cost: 0.5144,
AvgTime: 6.21ms
```

```
Step: 520521, Epoch: 100, Batch: 2700 of 2750, Cost: 0.5694,
AvgTime: 5.80ms
Step: 520616, Epoch: 100, Batch: 2750 of 2750, Cost: 0.5333,
AvgTime: 2.94ms
Test-Accuracy: 0.89
Total Time: 1664.68s
Final Cost: 0.5333
done
```

步骤5 访问 Web 站点观察训练结果

通过 Tensorboad 查看训练效果,前面已经获得了 Tensorboard 的外部 IP 106.1.1.1,直接通 过浏览器访问 http://106.1.1.1/,就可以观测到训练的效果。

← → C ①	
TensorBoard	SCALARS IMAGES AUDIO GRAPHS DISTRIBUTIONS HISTOGRAMS EMBEDDINGS
Write a regex to create a tag group X Split on underscores Data download links Tooltip sorting method: default Smoothing 0.6	accuracy accuracy 100 0.0000 0.0000 0.0000 0.000 0.000 0.000 0.
STEP RELATIVE WALL	cost
Runs Write a regex to filter runs	cost 100 600 400 200 0,000 40,00k 80,00k 120,0k C3

2.1.5 创建 NAS 数据卷

本文为您介绍如何创建NAS数据卷。

前提条件

准备一个与 Kubernetes 集群位于相同地域的存储包。您可使用已有的存储包,或者购买一个新的存储包,请参见#unique_54。

操作步骤

1. 登录 NAS控制台。

2. 默认进入文件系统列表页面,选择与 GPU 集群相同的地域,单击右上角 创建文件系统。

	NAS文件系统管理 级化1 级化2 级板1 级板2 级雨1 香港 亚大东南1(新加坡) 美国东部1(靖吉尼亚) 欧洲中部1(法兰克福) 亚大东南3(吉隆坡) 级1 大南部1(孟买) 亚大东南5(淮加达) 亚大东南3(吉隆坡) 级1 西港 亚大东南5(淮加达)			
 NAS 		€刷新	购买存储包	创建文件系统
文件系统列表				
权限组	重磅:云上超级存储引擎并行文件系统CPFS邀请测试中,您可以在此申请测试资格。 申请 查询申请状态			2
存储包	重磅:针对高IOPS、低时延业务的NASPlus极速IO型邀请测试中,您可以在此申请测试资格。 申请 查询申请状态			
文件同步	产品更新: NAS现已开通用户级监控,可以实时监控NAS的多项IO指标,您可以在此申请开通资格。 申请 查询申请状态			
	温馨提示:创建文件系统后,您需要为文件系统添加一个挂载点,挂载点是文件系统的访问入口。如何创建挂载点			
	文件系统10/名称 存储类型 协议类型 存储量 所在可用区 已鄉存儲包 挂载点数目			操作

3. 在弹出的对话框中,进行配置,创建一个 NAS 文件系统。

创建文件系统		\times
SSD性能型文件系	统存储容量上限1PB,容量型文件系统存储容量上限10PB。	
* 地域:	华东 1 ▼ 不同地域文件系统与计算节点不 互通	
* 存储类型:	容量型 ▼	
*协议类型:	NFS(包含NFSv3和NFSv4〕▼	
* 可用区:	华东1可用区B ▼ 同一地域不同可用区之间文件系 统与计算节点互通	
存储包:	不绑定 绑定一个现有空闲存储包,没有 则不绑定	
	确定取消	肖

- · 地域:选择与容器集群相同的地域。本示例选择 华东1。
- ・存储类型:本示例选择 容量型 。
- ・协议类型:选择 NFS。
- ·可用区:选择华东1可用区B。同一地域不同可用区可以互通。
- · 存储包:选择一个准备好的存储包。
- 4. 单击确定后,新建的文件系统出现在列表中。

5. 单击新建文件系统右侧的 管理,进入文件系统详情页面。

	NAS文件系统管理 华	北1 华北2 华北3 华 太东南3(吉隆坡) 华北5	东1 华东2 华丽 亚太南部1(孟买)	南 1 香港 亚太 :) 亚太东南5 ((东南 1 (新加坡) 美国东部 1 (弗雷 雅加达)	吉尼亚) 欧洲中部	31(法兰克福)			
 NAS 								こ刷新	购买存储包	创建文件系统
文件系统列表										
权限组	重磅: 云上超级存储引	擎并行文件系统CPFS邀请测	试中,您可以在此	申请测试资格。	申请 查询申请状态					
存储包	重磅:针对高IOPS、但	彩时延业务的NASPlus极速IO	型邀请测试中,您可	可以在此申请测试	资格。 申请 查询申请状态					
文件同步	产品更新: NAS现已开	通用户级监控,可以实时监	控NAS的多项IO指标	示,您可以在此申	请开通资格。 申请 查询申请》	术态				
	温馨提示: 创建文件系统后, 您需要为文件系统添加一个挂载点, 挂载点是文件系统的访问入口。 如何创建挂载点									
	文件系统ID/名称	存储类型	协议美型	存储量	所在可用区	已绑存储包	挂载点数目			操作
	035bb49518 035bb49518	容量型	NFS	447.92 MB	华东 1 可用区 B	是	1		添加挂载点	管理 創除

6. 单击 添加挂载点, 添加 VPC 类型的挂载点, 参见 #unique_50/

添加挂载点		\times
挂载点是云服务器 每个挂载点必须与	访问文件系统的入口,当前支持专有网络和经典网络挂载点 一个权限组绑定。	Ξ,
Linux客户端在默认 可以参考该文档进	、情况下对NFS请求并发数做了限制,若遇到性能较差的情况 行配置。	æ,
文件系统ID:	035bb49518	
* 挂载点类型:	专有网络	
* VPC网络:	VPC172 (172.16.0.0/16) ▼ 点击前往VPC控制创建VPC网络	
* 交换机:	testk8s (172.16.166.0/24) 🔻	
* 权限组:	VPC默认权限组(全部允许 ▼	
	确定	収消

 $unique_50_Connect_42_section_9q8_owp_z6n_{\circ}$

7. 添加挂载点完毕后,查看挂载点地址。

035bb49518								
基本信息							删除文件图	統 ^
文件系统ID: 035bb49518	地域: 华东1	地域: 华东1			可用区: 华东1可用区 B			
存储关型: 容量型	协议类型: NFS (NFSv	3及NFSv4.0)			文件系统用量: 447.92 MB			
创建时间: 2017-10-27 14:01:59 +0800								
存储包								^
存借包ID: naspackage-035bb49518-48dd8c 存储包容量: 500.00 GB 升	级	超始时间: 201 7-1 0	-27 14:01:59 +0800		有效期至: 2017-	11-28 00:00:00 +0800 🦛	F. C.	
種數点						\$	四何挂戴文件系统 液加挂動	瀌 ^
挂载点类型 ● VPC 交换机 ● 挂载地址 ●			权限组	状态●				操作
专有网络			VPC默认权限组(全部允 许)	可用			修改权限组 激活 禁門	1 前除

2.1.6 TensorFlow模型开发

TensorFLow 是深度学习和机器学习最流行的开源框架,它最初是由 Google 研究团队开发的 并致力于解决深度神经网络的机器学习研究,从2015年开源到现在得到了广泛的应用。特别是 Tensorboard 这一利器,对于数据科学家的工作非常有用。

Jupyter notebook 是强大的数据分析工具,它能够帮助快速开发并且实现机器学习代码的共 享,是数据科学团队用来做数据实验和组内合作的利器,也是机器学习初学者入门这一个领域的好 起点。

利用 Jupyter 开发 TensorFLow 是许多数据科学家的首选,但是如何能够快速从零搭建一套这样的环境,并且配置 GPU 的使用,同时支持最新的 TensorFLow 版本,对于数据科学家来说既复杂,同时也浪费精力。在阿里云的 Kubernetes 集群上,您可以通过简单的 Web 界面,一键式创建一套完整的 tensorFlow 实验环境,包括 Jupyter Notebook开发模型,利用 Tensorboard 调整模型。

准备Kubernetes环境

阿里云容器服务Kubernetes 1.9.3目前已经上线,但是购买按量付费的GPU计算型服务器需要申 请ECS工单开通。创建包含适当数量弹性计算资源(GPU 服务器)的 Kubernetes 集群,可以参 见#unique_46。

需要登录 Master节点,执行相关命令,参见#unique_48。

准备数据

- 1. 创建NAS文件存储,并且设置vpc内挂载点。可以参考管理文件系统。并且查看挂载点,这里假 设挂载点为xxxxxx.cn-shanghai.nas.aliyuncs.com
- 2. 准备名字为/data的数据文件夹。

```
mkdir /nfs
mount -t nfs -o vers=4.0 xxxxxx.cn-hangzhou.nas.aliyuncs.com:/ /nfs
mkdir -p /nfs/data
umount /nfs
```

创建 persistent volume

以下为创建NAS的nas.yaml样例,实际上也可以创建云盘或者OSS等持久化存储

```
apiVersion: v1
kind: PersistentVolume
metadata:
   labels:
      train: mnist
   name: pv-nas-train
spec:
   persistentVolumeReclaimPolicy: Retain
   accessModes:
      - ReadWriteMany
```

```
capacity:
  storage: 5Gi
flexVolume:
  driver: alicloud/nas
  options:
    mode: "755"
    path: /data
    server: xxxxxx.cn-hangzhou.nas.aliyuncs.com
    vers: "4.0"
```

📕 说明:

这里需要指定label为model: mnist,该标签对于 pvc 选择 pv 绑定非常重要。另外和NAS相关的具体配置可以参考#unique_56。

运行kubectl命令创建。

```
kubectl create -f nas.yaml
persistentvolume "pv-nas" created
```

通过应用目录部署 TensorFlow 应用

- 1. 登录 容器服务控制台。
- 2. 在 Kubernetes 菜单下,单击左侧导航栏中的市场 > 应用目录,进入应用目录列表。
- 3. 单击 ack-tensorflow-dev, 进入对应的 chart 页面。



4. 单击参数, 就可以通过 Web 界面配置参数, 然后单击 部署, 最后单击 Kubernetes 控制台。



默认的配置下,是使用 CPU 进行计算。

应用目录 - ack-tensorflow-dev ack-tensorflow-dev incubator Setup TensorFlow development environment for Kubernetes on Alibaba Cloud Container Service	
送用 参数 ① 1 単 Default values for ack-tensorflow-dev. 2 ■ This is a VML-formatted file. 3 ■ Declare variables to be passed into your templates. 4 □uotate variables to be passed into your templates.	部署 仅支持 Kubernetes 版本 1.8.4 及以上的集群,对于 1.8.1 版本的集 群,您可以在集群列集中进行"集群升级"强作
<pre>5 ## Tensorflow server image version 5 ## Tensorflow server image version 6 ## Tensorflow server image service services services com/tensorflow-service (instant in the periods of (hims </pre>	集群
except Hongkong,	test-gpu 🔻
<pre>7 ## registry.us-east-1.allyuncs.com/tensor+low-samples/jupyter for other regions including Hongkong. 8 ## if nvidia.com/gpu =0, the tag is 1.5.0-devel, nvidia.com/gpu =0, the tag is 1.5.0-devel-gpu</pre>	命名容问
9 · ## You can also bring up your own docker image. 10 · image:	default v
<pre>repository: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/jupyter</pre>	Conduct
12 tag: 1.5.6-devel 13 oullPolicy: Always	发布名称
14 password: tensorflow	ack-tensorflow-dev-default 3
15 resources: {} 16 # We usually recommend not to specify default resources and to leave this as a conscious	
17 # choice for the user. This also increases chances charts run on environments with little	部署成功,请前往控制台查看部署进度: Kubernetes 控制台
18 # resources, such as Minikube. If you do want to specify resources, uncomment the following 19 # lines adjust them as precessary, and remove the curly braces after 'resources'.	部署
20 # limits:	
21 # cpu: 100m	2
22 * memory: 1201 23 * nvidia.com/gpu: 1	
24 # requests:	版本
25 # cpu:100m 26 # memory: 128Ni	010
27 # nvidia.com/gpu: 1	0.1.0
28 - tensorboard:	项目主页
<pre>29 Image: 30 repository: registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tensorboard</pre>	
31 tag: 1.1.0	链接
32 pullPolicy: Always 33 service:	
34 type: LoadBalancer	

本示例使用 GPU 节点进行 tensorflow 开发,具体的配置如下:

```
jupyter:
   image:
     pullPolicy: Always
     repository: registry.cn-hangzhou.aliyuncs.com/tensorflow-
samples/jupyter
     tag: 1.5.0-devel-gpu
   password: tensorflow
   resources:
     limits:
       nvidia.com/gpu: 1
     requests:
       nvidia.com/gpu: 1
service:
   type: LoadBalancer
tensorboard:
   image:
     pullPolicy: Always
     repository: registry.cn-hangzhou.aliyuncs.com/tensorflow-
samples/tensorboard
     tag: "1.1.0"
persistence:
   enabled: true
  mountPath: /data
   pvc:
     matchLabels:
        train: mnist
```

storage: 5Gi

您也可以登录到 Kubernetes master 节点,运行以下命令进行部署。

\$ helm install --name ack-tensorflow-dev-default incubator/acktensorflow

5. 进入 Kubernetes Dashboard 页面, 查看 tensorflow 应用启动的状态。

<	三 縣兒							+ 创建		
集群 会内交回	CPU 使用率				内存使用率 ①					
节点	0.004		_	85.8 M				_		
持久化存储器	₩ 0.003 ₽ 0.002			£ 57.2 M						
存储类	0,001			老 19.1 M						
命名空间 default ~	18.07	18.07 8916	16.08	10.06		10.08	18.07 8514	16.08		
概况	工作负载									
工作负载	1f2\$\$%5									
守护进程集										
部署	100.00%		100.0	0%			100.00%			
容難迫	854		88	ia.			副本类			
副本集 副本控制器	部署							Ŧ		
有状态副本集	名称 🗧	标签	容器组		E	12 ÷	洗 漆			
服务发现与负载均衡	ack-tensorflow-dev-default	app: ack-tensorflow-d chart: ack-tensorflow-dev-0 heritage: Till release: ack-tensorflow-dev-defa	1/1		201	8-04-17 16:05:31	registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/jupyter:1.5. registry.cn-hangzhou.aliyuncs.com/tensorflow-samples/tensorboarc	:		
IES-	容器组							Ŧ		
配置 与存储 配置字典	名称 ≑	节点 状:	5 ¢		日重用	B创 ≑ CPU(核)	内存(字节)			
持久化存储整素取	ack-tensorflow-dev-default-7bc846484-86gbn	cn-hangzhou.i-bp12xv/jjwqe6j7nca2q8 Ru	unning		0	2018-04- 17 16:05:31	74.270 Mi	≓ :		
任密享具										

至此, ack-tensorflow-dev 应用成功运行。

使用 TensorFlow 实验环境

1. 首先通过 ssh 登录 Kubernetes 集群,查看 tensorflow 应用列表。

```
$ helm list
NAME REVISION UPDATED
STATUS CHART
NAMESPACE
ack-tensorflow-dev-default 1 Tue Apr 17 16:05:31
2018 DEPLOYED ack-tensorflow-dev-0.1.0 default
```

2. 利用 helm status 命令检查应用配置。

```
$ helm status ack-tensorflow-dev-default
LAST DEPLOYED: Tue Apr 17 16:05:31 2018
NAMESPACE: default
STATUS: DEPLOYED
RESOURCES:
==> v1/Service
NAME
                             TYPF
                                           CLUSTER-IP
                                                         EXTERNAL-IP
   PORT(S)
                                 AGE
ack-tensorflow-dev-default LoadBalancer 172.21.7.225 120.55.137.
46 6006:32203/TCP,80:30795/TCP 7m
==> v1beta2/Deployment
NAME
                             DESIRED CURRENT UP-TO-DATE AVAILABLE
 AGE
ack-tensorflow-dev-default 1
                                      1
                                               1
                                                           1
 7m
NOTES:
1. Get the application URL by running these commands:
```

这里可以看到外部 SLB 的 IP 是 120.55.137.46, Jupyter Notebook 的端口为 80,

Tensorboard 为 6006。

3. 通过 Jupyter 访问端点登录,本示例中 Jupyter 的访问地址是 http://120.55.137.46,输入前面设定的密码,单击 Log in,在本示例中我们设定的是 tensorflow。

① 不安全 120.55.137.46/login?next=%2Ftree%3F	×	
	💭 Jupyter	
Password	d:	Log in

4. 单击 New > Terminal, 进入 Terminal 页面。

📁 jupyter	Logout
Files Running Clusters	1
Select items to perform actions on them.	Upload New -
	Notebook:
The notebook list is empty.	Python 2
	Other:
	Text File
	Folder
2	Terminal
· · · · · · · · · · · · · · · · · · ·	

5. 在 bash 执行环境下,在 Terminal 内执行 nvidia-smi,可以看到 GPU 的配置。

# bash root@ack-tensorfl Tue Apr 17 08:	ow-dev-default-84c7fbb5dc-p7v8g; 53:47 2018	# nvidia−smi			
NVIDIA-SMI 38	4.111	Driver Version: 384.11	1		
GPU Name Fan Temp	Persistence-M Bus Perf Pwr:Usage/Cap	Id Disp.A Memory-Usage GPU	Volatile Uncorr. ECC -Util Compute M.		
0 Tesla N/A 34C	P100-PCIE On 0000 P0 25W / 250W	0000:00:08.0 0ff 0MiB / 16276MiB	0%	0 Default	
			+		
Processes: GPU	PID Type Process	name		Usage	GPU Memory
No running	processes found		+		

6. 通过 git 命令下载tensorflow样例代码。

```
$ git clone https://code.aliyun.com/kubernetes/Tensorflow-Examples.
git
Clone into "Tensorflow-Examples"...
remote: Counting objects: 885, done.
remote: Total 885 (delta 532), reused 885 (delta 532)
Receiving objects: 100% (885/885), 17.89 MiB | 0 bytes/s, done.
Resolving deltas: 100% (532/532), done.
Checking connectivity... done.
```

root@ack-tensorflow-dev-default-84c7

- 7. 查看mount进来的/data路径。
 - ls /data
- 8. 返回 Jupyter 主页您就能看到 Tensorflow-Examples 已经下载到您的工作目录。

O //tree?#	
💭 jupyter	Logout
Files Running Clusters	
Select items to perform actions on them.	Upload New - 2
	Name 🔶 Last Modified
C Tensorflow-Examples	3分钟前

- 9. 进入 /Tensorflow-Examples/notebooks/4_Utils/tensorboard_basic.ipynb 目
 - 录,运行程序。

🗒 说明:

如果您需要用Tensorboard观测训练效果,请将日志记录到/output/training_logs下。



10.以下为训练结果输出。

Epoch:	0001	cost=	1.184048782				
Epoch:	0002	cost=	0.665532489				
Epoch:	0003	cost=	0.552896572				
Epoch:	0004	cost=	0.498722185				
Epoch:	0005	cost=	0.465545912				
Epoch:	0006	cost=	0.442538375				
Epoch:	0007	cost=	0.425477976				
Epoch:	0008	cost=	0.412196293				
Epoch:	0009	cost=	0.401395965				
Epoch:	0010	cost=	0.392410899				
Epoch:	0011	cost=	0.384823679				
Epoch:	0012	cost=	0.378181933				
Epoch:	0013	cost=	0.372421673				
Epoch:	0014	cost=	0.367311774				
Epoch:	0015	cost=	0.362693607				
Epoch:	0016	cost=	0.358600763				
Epoch:	0017	cost=	0.354853889				
Epoch:	0018	cost=	0.351446943				
Epoch:	0019	cost=	0.348336726				
Epoch:	0020	cost=	0.345467410				
Epoch:	0021	cost=	0.342783512				
Epoch:	0022	cost=	0.340270793				
Epoch:	0023	cost=	0.337918402				
Epoch:	0024	cost=	0.335781661				
Epoch:	0025	cost=	0.333670841				
Optimi	Optimization Finished!						
Accura	Accuracy: 0.914						
Run the	е соли	nand 1:	ine:				
> tensorboardlogdir=/output/training_logs							
11.这时您可以登录 Tensorboard 查看训练效果,本示例中Tensorboard的地址为 http://120.55.137.46:6006。这里您可以看到模型的定义和训练的收敛趋势。

TensorBoard	SCALARS IMAGES AUDIO GRAPHS DISTRIBUTIONS	
Write a regex to create a tag group X Split on underscores Data download links Tooltip sorting method: default	accuracy 0.900 0.800 0.700	
Smoothing 0.6	0.600 0.500 0.400 0.000 4.000k 8.000k 12.00k	
Horizontal Axis	loss	
Runs Write a regex to filter runs ✔	2.20 1.80 1.40 1.00 0.200 0.000 4.000k 8.000k 12.00k	

