K8S 1.18版本 (https://www.kubernetes.org.cn/tags/kubernetes1-18) 1.17 (https://www.kubernetes.org.cn/tags/kubernetes1-17) 1.16 (https://www.kubernetes.org.cn/tags/kubernetes.org.cn/tag

表达的中心中心 wp-login.php?cms=k8s) | 加入我们 (https://www.kubernetes.org.cn/%e8%81%94%e7%b3%bb%e6%88%91%e4%bb%ac)

开发实践 (https://www.kubernetes.org.cn/practice) 行业动态 (https://www.kubernetes.org.cn/news) 入门教程 (https://www.kubernetes.org.cn/course) 安装教程 (https://www.kubernetes.org.cn/course) 文档下载 (https://www.kubernetes.org.cn/%E6%96%87%E6%A1%A3%E4%B8%8B%E8%BD%BD) QQ/微信群 (https://www.kubernetes.org.cn/kubernetes.org.cn/kubernetes.org.cn/kubernetes.org.cn/kubernetes.org.cn/www.kubernetes.org.cn/video) 活动 (https://www.kubernetes.org.cn/meetup) 中文文档 (https://docs.kubernetes.org.cn) **Q**

Kubernetes CKA实战培训 突击CKA全球认证

官方认证CKA讲师、实操环境实战、现场答疑互动

开班城市:北京/上海/深圳/武汉/成都/杭州 以及线上直播班

点击查看

(https://www.kubernetes.org.cn/peixun)

您目前处于: 社区首页 (https://www.kubernetes.org.cn) > 微服务 (https://www.kubernetes.org.cn/microservices) > 微服务与单体应用,该选谁

微服务与单体应用,该选谁

(https://www.kubernetes.org.cn/8014.html)

2020-07-14 22:26 阅读(642) 王延飞 (https://www.kubernetes.org.cn/author/fly) 分类: 微服务 (https://www.kubernetes.org.cn/microservices) 评论(0)

越来越多的组织开始放弃单体应用,逐步转向微服务的架构模式-将业务流程分为多个独立的服务。

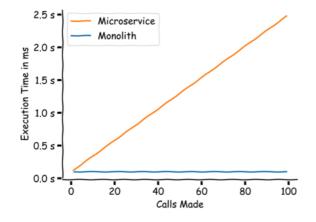
例如,在一个机票预订中,就可能涉及许多个单独的过程:在航空公司预订机票,付款,并在机票成功预订后向客户发送确认信息。

微服务架构,就是将各个流程按照业务拆分为独立的服务。在上面的示例中,机票预订服务可以被拆分为 机票预订,付款和确认,拆分后的微服务可以通过接口相互通信。

那么,微服务与单体应用,究竟有什么不同?

对比1: 网络延迟

当涉及微服务时,有一个基本的物理定律在起作用,每当微服务通过网络调用另一服务时,字节就通过网络发送,这涉及将字节转换为电信号或脉冲光,然后将这些信号转换回字节。根据模拟结果 (https://github.com/Marcuccio/Network-overhead-in-microservices-architecture),微服务调用的等待时间至少为 24ms。如果我们假设实际处理大约需要100毫秒,则总处理时间如下所示:



网络延迟-微服务与单体应用

Kubernetes 1.18 版本

我们很高兴宣布Kubernetes 1.18的交付,这是我们2020年的第一版!Kubernetes 1.18包含38个增强功能:其中15个功能已趋于稳定,beta版本中有11个,alpha版本中有12个。 (https://www.kubernetes.org.cn/tags/kubernetes1-18)



关注「K8S中文社区」微信公众号

回复"文档" 获取K8S文档下载链接

回复"加群" 加入K8S微信技术交流群

最新文章



Dubbo-go 发布 1.5 版,朝云原 生迈出关键一步

2020-07-29 评论()

(https://www.kubernetes.org.cn/8125



演示视频: 在K8S上备份和恢复 MySQL

MysQL

2020-07-27 评论()

(https://www.kubernetes.org.cn/8121.



Porter 进入 CNCF 云原生全景图, 新版本即将发布!

2020-07-27 评论()

(https://www.kubernetes.org.cn/8118.



掌门教育微服务体系 Solar I 阿里巴巴 Nacos 企业级落地上篇

2020-07-27 评论()

(https://www.kubernetes.org.cn/8117.

热门推荐



你的微服务还差个容错机制

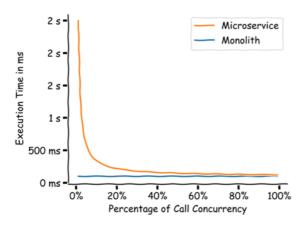
2020-03-26

(https://www.kubernetes.org.cn/7060.

Kubernetes 1.18GA, 15个稳定 11个beta, 引入kubectl debug命 令

2020-03-26

假设在理想情况下,所有调用执行可以同时发生,并且彼此之间不依赖–这称为扇出模式(fan-out pattern)。 下图显示了随着越来越多的调用同时执行,总时间如何减少。



同时执行多个调用意味着总执行时间减少

并行执行所有调用, 意味着最长的调用执行完, 服务将返回给使用者。

从上图可以看出,单体应用没有网络延迟,因为所有调用都是本地调用。即使在完全可并行化的世界中,单体应用仍会更快。而,微服务由于需要多个服务间通信,即使并行调用,也是需要一定的网络延迟。

这一次,单体应用胜利了。

对比2: 复杂性

考虑复杂性时,有许多因素在起作用:开发的复杂性和运行软件的复杂性。

由于开发的复杂性,在构建基于微服务的软件时,代码库的大小会快速增长。因为微服务,涉及多个源代码,使用不同的框架甚至不同的语言。由于微服务需要彼此独立,因此经常会有代码重复。

另外,由于开发和发布时间不一致,因此不同的服务可能会使用不同版本的库。

对于日志和监控方面,在单体应用中,日志记录就像查看单个日志文件一样简单。但是,对于微服务,跟 踪问题可能涉及检查多个日志文件。不仅需要查找所有相关的日志输出,而且还需要以正确的顺序将它们 放在一起。

在Kubernetes集群中运行微服务时,复杂度进一步增加。虽然Kubernetes启用了诸如弹性伸缩等功能,但它并不是一个易于管理的系统。要部署单体应用,简单的复制操作就足够了。要启动或停止单体应用,通常只需一个简单的命令即可。还有与单体应用相比,事务还增加了运行微服务架构的复杂性。跨服务的调用,很难保证数据是同步的。例如,执行不当的调用,重试可能会执行两次付款。

这一次,单体应用又胜利了。

对比3: 可靠性

在微服务中,如果A服务通过网络以99.9%的可靠性调用B服务(这意味着在1000个调用中,有一个将由于网络问题而失败),这时B调用再C服务,我们将获得99.8%的可靠性。



(https://www.kubernetes.org.cn/7055.



英国Monzo银行,用K8s管理 1600个微服务实践

2020-03-21

(https://www.kubernetes.org.cn/7001.



Java vs. Go 微服务 – 负载测试 (复赛)

2020-03-20

(https://www.kubernetes.org.cn/6988.



与时俱进 – 为什么要使用云原 生数据库?

2020-03-15

(https://www.kubernetes.org.cn/6953.



2020年Service Mesh 三大发展方向

2019-12-13

(https://www.kubernetes.org.cn/6255.

社区标签

BoCloud博云

(https://www.kubernetes.org.cn/tags/bocloud%e5%8d%9a%e

CI/CD (https://www.kubernetes.org.cn/tags/cicd)

CNCF

(https://www.kubernetes.org.cn/tags/cncf)

DevOps

(https://www.kubernetes.org.cn/tags

Docker

(https://www.kubernetes.org.cn/tags/c

etcd (https://www.kubernetes.org.cn/tags/etcd)

GO (https://www.kubernetes.org.cn/tags/go)

Helm

(https://www.kubernetes.org.cn/tags/he

Istic

(https://www.kubernetes.org.cn/tags/istio)

Jenkins

(https://www.kubernetes.org.cn/tags/jenk k8s代码解读

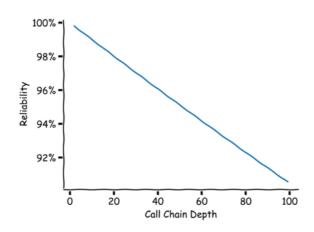
(https://www.kubernetes.org.cn/tags/k8s%e4%bb%a3%e7

kubeadm

(https://www.kubernetes.org.cn/tags/kube

kuberenetes

 $(https://www.kubernetes.org.cn/tags/kuberen\varepsilon$



随着调用时间的延长, 可靠性下降

因此,在设计微服务架构时,要考虑网络会在某个时刻断开。微服务提供了一些解决此问题的解决方案。Spring Cloud (https://spring.io/projects/spring-cloud)提供了负载均衡和网络故障处理,诸如Istio (https://istio.io/)之类的服务网格还能够处理多种编程语言的服务。当微服务集群中的服务失败时,集群管理器给出替代方案。这就使得微服务架构具有高度的弹性。

Netflix创建了一个名为Chaos Monkey (https://github.com/netflix/chaosmonkey)的工具,该工具可以模拟随机终止虚拟机和容器。微服务的开发者,可以使用Chaos Monkey (https://github.com/netflix/chaosmonkey)的工具在测试环境模拟网络断连和网络故障等问题,这样,他们就可以确保系统能够处理生产环境中的停机故障。

单体应用中的所有调用都是在本地完成,因此很少发生网络故障,虽然如此,然而单体应用在云环境却无 法满足弹性伸缩的需求。

最后, 微服务取得了胜利。

对比4:资源使用

一般来说,微服务会比单体应用使用更多的资源。即使在Docker中运行时,基准测试 (https://community.centminmod.com/threads/centos-6-6-memcached-1-4-22-docker-image.2348/#post-11009https://community.centminmod.com/threads/centos-6-6-memcached-1-4-22-docker-image.2348/%23post-11009)发现,虽然服务连接数量下降了8%,但是容器编排还将消耗资源,日志聚合和监视也将消耗资源。

但是,微服务使我们可以更聪明地使用资源。由于集群管理器可以根据需要分配资源,因此实际的资源使 用量可能要低得多。

在软件中,20%的代码一般会完成80%的工作。如果单体应用的一个实例使用8GB,则两个实例使用16GB,依此类推。使用微服务后,我们可以把单体应用中负责主要职能的20%代码提取成一个服务,因此对于两个实例,我们的RAM使用量为降低到了9.6GB左右。

下图显示了资源使用情况的差异。

Kubernetes

(https://www.kubernetes.org.c

Kubernetes1.5

(https://www.kubernetes.org.cn/tags/kubernetes1-5)

Kubernetes1.6

(https://www.kubernetes.org.cn/tags/kubernetes1-6)

Kubernetes 1.7

(https://www.kubernetes.org.cn/tags/kubernetes1-7)

Kubernetes 1.8

(https://www.kubernetes.org.cn/tags/kubernetes1-8)

Kubernetes 1.9

(https://www.kubernetes.org.cn/tags/kubernetes1-9)

Kubernetes1.10

(https://www.kubernetes.org.cn/tags/kubernetes1-10)

OpenStack (https://www.kubernetes.org.cn/tags/openstack)

PaaS

(https://www.kubernetes.org.cn/tags/paas)

Pod (https://www.kubernetes.org.cn/tags/pod)

Prometheus

(https://www.kubernetes.org.cn/tags/prometheus)

Rainbond (https://www.kubernetes.org.cn/tags/rainbond)

Rancher

(https://www.kubernetes.org.cn/tags/rancher)

Serverless

(https://www.kubernetes.org.cn/tags/serverles

Service (https://www.kubernetes.org.cn/tags/service)

service mesh

(https://www.kubernetes.org.cn/tags/service-mesh)

云原生

(https://www.kubernetes.org.cn/tags

云计算

(https://www.kubernetes.org.cn/tags/%e4%ba%91%e8%ae%a1%e7%ae%97)

企业案例

(https://www.kubernetes.org.cn/tags/%e4%

存储

(https://www.kubernetes.org.cn/tags/%e5%ad%

安全

(https://www.kubernetes.org.cn/tags/%e5%ae%899

容器

(https://www.kubernetes.org.cn/tags/%

容器云

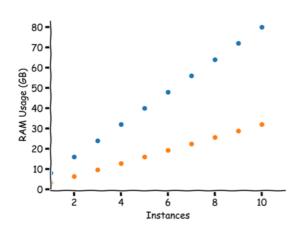
(https://www.kubernetes.org.cn/tags/%e5%ae%b9%e5%99%a8%e 容器云平台

(https://www.kubernetes.org.cn/tags/%e5%ae%b9%e5%99%a8%e4%ba%91%

微服务

(https://www.kubernetes.org.cn/tags

日ま



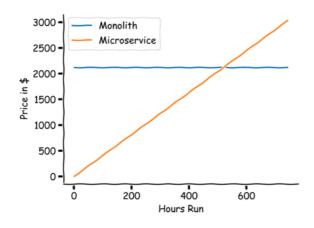
随着越来越多的实例在运行,单体应用比微服务需要更多的资源

资源使用率方面,微服务胜利了。

对比5: 扩展的精确性

单体应用的扩展有多种办法,运行多个实例,或运行多个线程,或者使用非阻塞IO。对于微服务架构,这 三个也都是适用的。

但是,面对客户端越来越多的请求,由于微服务架构更精细,因此扩展单个服务也更加精细。所以,对于 微服务来说,扩展既简单又精确。而且,由于微服务的资源消耗较少,又可以节省资源。



相比单体应用,微服务精确的扩展和更少的资源使用,是一个明显的胜利。

对比6: 吞吐量

让我们再看一个性能指标--吞吐量。在微服务架构体系中,数据需要在不同服务之间发送,从而会产生一定的开销。如果微服务还不是一个分布式架构,那么他的吞吐量还不如一个单体应用高。

对比7: 部署时间

人们选择微服务架构的原因之一就是-能够节省部署时间,满足快速迭代。

由于微服务的职责单一原则,因此对其进行的任何更改都有很明确。然而,修改一个单体应用的功能,可能会"牵一发动全身"。

此外,微服务更易于测试。由于微服务仅覆盖有限的一组功能,因此代码依赖性低,便于编写测试并且运行得快。

灵雀云

(https://www.kubernetes.org.c

灵雀云 Jenkins

(https://www.kubernetes.org.cn/tags/%e7%81%b5%e9%9b% jenkins)

监控

(https://www.kubernetes.org.cn/tags/%e7%9b%

网络

(https://www.kubernetes.org.cn/tags/%e7%bd

_

(https://www.kubernetes.org.cn/tags/%e8%b4%9f%e8%bd%bd%e5%9d%87%

(https://www.kubernetes.org.cn/tags/%e8%bf%90%e7%bb%l

Kubernetes 版本资讯

- Kubernetes v1.18 正式版已发布 (https://www.kubernetes.org.cn/7055.html)
- Kubernetes v1.17 正式版已发布 (https://github.com/kubernetes/kubernetes/releases/tag/v1
- Kubernetes v1.16 正式版已发布 (https://www.kubernetes.org.cn/5838.html)
- Kubernetes v1.15 正式版已发布 (https://www.kubernetes.org.cn/tags/kubernetes-1-15)
- Kubernetes v1.14 正式版已发布 (https://www.kubernetes.org.cn/5204.html)

最新评论

翻译错误太多5天前说:

强烈建议网站开发一个读者可编辑的功能,读者发现翻译问题后,提交修改,后台审核通过后更新,大家一起共建,让社区更美好! (https://www.kubernetes.org.cn/k8s#comment-1490)

Cc360428 6天前说:

为什么我看不到日志,我kuber-service 改成了 default (https://www.kubernetes.org.cn/4278.html#comment-1489)

zoozer 1周前 (07-22)说:

高可用是不是少了 VIP 默认集群安装没办法高可用把 (https://www.kubernetes.org.cn/7315.html#comment-1488)

没牙的蚂蚁 1周前 (07-21)说:

使用token登陆报错,Unauthorized (401): Invalid credentials provided,这个怎么解决啊

(https://www.kubernetes.org.cn/7189.html#comment-1487)

还有,微服务的资源消耗较少,并且可以按比例扩展。这就使微服务可以无感知部署,例如,可以先在集群一部分节点上启动微服务的新版本,然后迁移一部分用户到新版本,如果有问题,这可以快速回滚到旧版本。

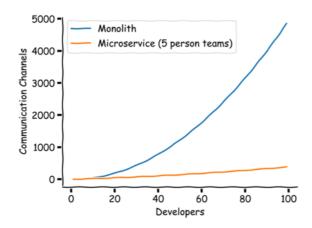
胜利归功于微服务。

对比8:沟通

在微服务诞生之前,弗雷德·布鲁克斯(Fred Brooks)撰写了开创性的著作《人月神话》(https://en.wikipedia.org/wiki/The_Mythical_Man-Month),本书的其中一项内容是,沟通渠道的数量随着团队成员的数量而增加。由两个人组成的团队,只有一个沟通渠道。如果有四个人,则最多可以访问六个频道。通信通道数的公式为n(n – 1)/2。由20位开发人员组成的团队拥有190个可能的沟通渠道。将这些开发人员分成两个团队,就可以大大减少沟通渠道的数量。

我们以拥有20个开发人员的团队为例,将其分为四个微服务团队(每个团队五个人),则每个团队有10个沟通渠道。四个团队之间的沟通渠道只有六个。沟通渠道的总数为46,大约占20个人团队的四分之一。

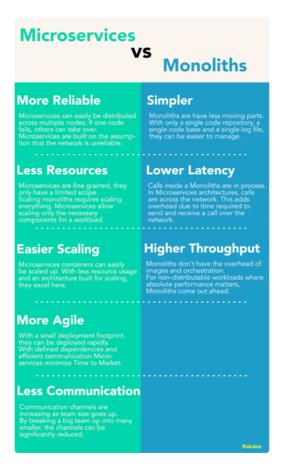
下图显示了,一个大团队的通信渠道数量,和单个微服务团队的通信渠道数量的对比。



因此,将10个以上的开发人员分成几个较小的团队,可以为任何开发项目提供更高的沟通效率。

这是微服务的另一个明显胜利。

谁是赢家?



单体应用获得了3场胜利,微服务获得了5场胜利。

但是,在查看此图表时,请记住它是相对的。微服务并不是解决所有开发问题的万能药。

例如,一个由5个开发人员组成的小型团队可能会倾向于选择单体应用。因为,单体应用不仅更易于管理,同时如果软件产品每秒仅有几个访问量,那么单体应用可能就足够了。

以下是一些迹象,表明微服务架构可能是一个合适的选择:

- 需要7*24的可靠性
- 精确的扩展
- 峰值和正常负载明显不同
- 超过10个开发人员的团队
- 业务领域可以被细分
- 方法调用链路短
- 方法调用可以使用REST API或队列事件。
- 几乎没有跨服务的事务

译文链接: https://thenewstack.io/microservices-vs-monoliths-an-operational-comparison/(https://thenewstack.io/microservices-vs-monoliths-an-operational-comparison/)



关注微信公众号,加入社区

(http://tpe/yidhotnet/lisabateshatidate

-- ## (http://www.kuberneles.org.cg/7958.htm)
%E5%B0%86%H25%H3%H3%H3%HB486H244F4%H7%B3%SH194H2604H3%HB48B7%SH1944F9%H348H348H2604H3



%E5%B0%86%E4%B8%9A%E5%8A%A1%E6%B5%81%E7%A8%8B%E5%88%86%E4%B8% 从这里开启您的 Kubernetes 之旅 100% 开源 Web UI 高线部署 一戦打容 应用商店 %E5%B0%86%E4%B8%9A%E5%8A%A1%E6%B5%81%E7%A8%8B%E5%88%86%E4%B8%

(https://github.com/KubeOperator/KubeOperator)

标签: 微服务 (https://www.kubernetes.org.cn/tags/%e5%be%ae%e6%9c%8d%e5%8a%a1)

相关推荐

- 掌门教育微服务体系 Solar | 阿里巴巴 Nacos 企业级落地上篇 (https://www.kubernetes.org.cn/8117.html)
- 2020 年, 从架构谈起, 到 Mesh 结束 (https://www.kubernetes.org.cn/8037.html)
- BoCloud博云获中电基金、蔚来资本C+轮战略投资 (https://www.kubernetes.org.cn/8073.html)
- 微服务和服务网格有什么区别, Istio告诉你 (https://www.kubernetes.org.cn/8067.html)
- 你问我答:容器篇 (1) (https://www.kubernetes.org.cn/7996.html)
- 提高微服务安全性的11个方法 (https://www.kubernetes.org.cn/7893.html)
- 确保微服务最佳运行的5个方面 (https://www.kubernetes.org.cn/7889.html)
- 5种主要的软件架构模式 (https://www.kubernetes.org.cn/7863.html)

评论 抢沙发



© 2020 Kubernetes中文社区 粤ICP备16060255号-2 (http://www.miitbeian.gov.cn/) 版权说明 (https://www.kubernetes.org.cn/版权说明) 联系我们 (https://www.kubernetes.org.cn/广告投放) 法律声明:本网站不隶属于谷歌或 Alphabet 公司 | kubernetes kubernetes 标识及任何相关标志均为 Google LLC 公司的商标。