

## 14 | 云上容器服务：从Docker到Kubernetes，迎接云原生浪潮

2020-04-03 何恺铎

深入浅出云计算

[进入课程 >](#)




讲述：何恺铎

时长 16:26 大小 15.06M



你好，我是何恺铎。

容器，毫无疑问是近年来的又一个技术热词。容器化技术的诞生和兴起，以及它所催生的微服务架构、DevOps、云原生等技术理念，都对软件行业产生了深远的影响。

容器的优点有很多了，完善的封装、便捷的部署、轻量的启动和调度，这些都是容器技术受到欢迎的原因。与编排系统配合后，它能让我们的应用程序容易管理和迭代，即便是再复杂的系统也不在话下。同时呢，容器应用还能做到非常好的可迁移性，环境中只要有符合  定义的容器运行时就可以顺利运行。

我相信你对容器其实有一定的了解，也知道 Docker 和 Kubernetes 分别是容器技术和容器编排的事实标准。甚至，不少同学已经有过一些实践的经验。

那么在容器这一讲中，我们主要关心什么问题呢？我认为，你需要重点搞清楚两个问题：

**容器和云是什么关系呢？**

**在云上运行容器有哪些方式，它们各自又有什么特点呢？**

让我们顺着容器上云的发展历程，来了解这两个问题的答案。

## 容器上云：从 Docker 到 Kubernetes

轻量的容器和富有弹性的云计算，互相之间其实是非常契合的。容器对于运行环境的极强适应性和快速启动的能力，配合云上动态扩展的庞大资源规模，让云端的容器应用可以在短时间内拓展到成千上万个实例。所以，**云可以说是容器应用的最佳载体，容器应用也非常适合在云上运行和扩展。**

其实在 Docker 技术家喻户晓之前，云厂商已经在研究和类似容器的技术了，因为云本身是多租户的，需要**运行环境的隔离性**。所以云本身也是容器技术的用户和受益者，只是部分厂商会考虑进行自研，未必直接使用 Docker 而已。

补充：比如说第 11 讲提到过的 AWS 的 Elastic Beanstalk，它就使用了类似容器的私有技术，以实现 Web 应用之间的隔离。当然后来呢，它也直接支持了 Docker 容器封装的应用。

而当 Docker 兴起之后，各大公有云都不约而同地开始对外提供容器相关的标准 PaaS 服务，并持续地进行改进。如果我们梳理容器 PaaS 服务的发展脉络，就会发现它经历了一系列的发展阶段。

在初期，云上容器平台把 Docker 容器在云上的**顺畅运行**，作为首要的目标。产品服务的主要目的是帮助用户创建底层虚拟机集群，免去了用户自己手动管理虚拟机的麻烦。然后呢，随着容器应用的复杂化，**编排**逐渐成为了用户最急迫的需求，所以各厂商又纷纷推出和加强容器编排方面的解决方案。

在那个编排框架群雄并起的年代，有的厂商选择了多点开花，比如微软当时的 Azure Container Service，可以支持 Docker Swarm、Apache Mesos (DC/OS) 和 Kubernetes 等多种编排系统；也有的厂商呢，选择了自己的编排方式，以便更好地和自己其他的云服务集成，比如 AWS 的 Elastic Container Service (ECS)。

当然，后来编排框架大战的结果你已经都知道了，Kubernetes 最终一统天下，成为了事实标准。所以各大厂商又很快地调转方向，纷纷为云上 Kubernetes 的支持加码，推出面向 Kubernetes 的专属服务了，比如 AWS 的 Elastic Kubernetes Service (EKS) 和 Azure 的 Azure Kubernetes Service (AKS)。阿里云呢，同样也逐渐停止了旗下容器服务对 Swarm 的支持，而把发展重点聚焦在容器服务 Kubernetes 版 (ACK) 上。

你看，云对容器技术的支持，是伴随着容器生态发展而发展的，所以很多时候，云也是容器生态重要的参与者和推动者。就像 Google Cloud 中的 GKE (Google Kubernetes Engine)，由于“根正苗红”，也一直是云上 Kubernetes 服务的标杆之一。

补充：如果说之后的技术潮流还有什么变化，我想你在云上也一样会看到领域内的最新进展。比如 Service Mesh (服务网格)，也有越来越多的云服务正在探索和提供相关的支持与服务。这也是云与时俱进的魅力所在。

所以，就现在最新的形势而言，如果要容器上云，那我想你几乎不用犹豫，直接选择各大云上最新的针对 Kubernetes 的服务即可。


关于 Kubernetes 本身，它是一个非常庞大的技术体系，我建议你通过专门的课程来系统学习。但在这里你需要重点了解一下，相对于自建 Kubernetes 集群，云上 Kubernetes 服务的几个独有特点。

首先，很多云上的 Kubernetes 服务，**由于云端的多租户特性，可以免除你在 Master 节点方面的开销**。换句话说，你只需要创建 Worker 节点，并为之付费就行了。云平台会统一为你提供和托管 Master 节点，降低你的资源和运维成本。

另外，我们都知道 Kubernetes 虽然复杂性较高，但抽象设计出色，**能够支持大量灵活的扩展**。所以云厂商在这个方面花了很多功夫，能够让很多云平台上的 IaaS 或 PaaS 功能组件，渗透到 Kubernetes 的体系中来，这样可以让两边有一个更紧密的集成。

比如说，在常用来引导外部流量的 **Ingress Controller**（入口控制器）方面，就有 AWS 的 ALB Ingress Controller，和 Azure 的 AKS Application Gateway Ingress Controller 等基于云上负载均衡器的控制器实现。它们会创建相应的 PaaS 服务实例，来为 Kubernetes 集群服务。

再比如，我们可以在 Kubernetes 中定义动态存储卷分配策略的 StorageClass 层面，指定使用云端的**块存储服务**，来按需创建和挂载持久化存储。下面的配置文件片段（注意它的 provisioner 和 parameters 字段），就展示了一个使用 AWS EBS 服务来提供的 SSD 云硬盘的例子。

 复制代码

```
1 apiVersion: storage.k8s.io/v1
2 kind: StorageClass
3 metadata:
4   name: standard
5 provisioner: kubernetes.io/aws-ebs
6 parameters:
7   type: gp2
```

云和 K8s 集成的方面还有很多，权限认证、日志集成、私有网络等许多方面，这里就不一一展开讨论了。这些集成共同构成了 K8s 在云上运行，以及和云全方位融合的坚实保障。

目前，AWS 还正在积极地研发推进 AWS Service Operator for Kubernetes，它把 S3、RDS 等有**状态**的 AWS 云服务，以**自定义资源**的形式纳入到了 Kubernetes 中。这让我们能够通过使用云服务来扩展 Kubernetes 的能力，并反过来使用 Kubernetes 来管理这些云资源。我们可以预期，类似 Service Operator 这样的服务，未来将会不断走向成熟，也能够进一步加快容器和云一体化架构的发展。

从**架构灵活性**的角度来看，云上 Kubernetes 服务还带来了另一个好处：**多集群**。

由于建立 K8s 集群的门槛大大降低了，如果业务间的关联较小，你是可以考虑为不同的业务单独创建 K8s 集群的。这样，不同的集群之间就有了更好的隔离性，也可以单独地扩展。

## 容器镜像服务



容器方面的另一种常见而且又很重要的云服务，就是**容器镜像服务**（Container Registry）。我们知道，容器的镜像是容器化程序封装后的基本单位，在云上，你肯定需要一个可以存储和管理自己程序镜像的地方，就像 Docker Hub 管理了很多公开镜像一样。

所以，在大多数云上都提供了自己的容器镜像服务，如 AWS 的 ECR、Azure 的 ACR 等。它们能让你建立私有的镜像仓库，支持镜像的推送和拉取，还可以进行版本管理等操作。镜像服务虽然看上去简单，却是云上容器体系中不可或缺的一环，容器的运行肯定要和它打交道。

## 全托管的容器实例服务

上面我们所讨论的容器服务，一般还是能够看到虚拟机的，在云虚拟机的层面都有相应的集群被创建。这其实是一种半托管的 PaaS 模式。

**那么，有没有更加方便易用，不用关心底层基础设施的容器服务呢？**


答案是肯定的，这也是近期云计算在容器领域的另一个特点和趋势：**容器实例服务**。常见的有 AWS 的 Fargate、阿里云的弹性容器实例、Azure 的 Azure Container Instance 等等。它们都是“全托管”思想在容器服务上的体现。

如果你只是有一个容器镜像，想要尽快地在云上跑起来，那么这类服务很可能就是你的最佳选择。因为它简便易行，成本低、速度快，而且你不需要操心底层的虚机和集群，也可以绕开复杂的编排系统，只需要关注 Pod 运行层面的目标就可以了。这是容器实例类云服务很大的卖点，尤其对于无状态的应用非常适合。

接下来，我就举一个实际的例子，把 [🔗 第 11 讲](#) 中，我们计算斐波那契数列的 Node.js 程序容器化，并且把它搬到云上的容器实例服务。我们这里使用 **Azure 云上的容器实例**（Azure Container Instance）来完成这个实验。

首先，要把我们之前的 Node 程序用 Docker 封装起来，相关的 Dockerfile 如下：

```
1 FROM node:10
2 WORKDIR /usr/src/app
3 COPY package.json ./
4 COPY app.js ./
```

 复制代码

```
5 RUN npm install
6 ENV PORT=80
7 EXPOSE 80
8 CMD [ "node", "app.js" ]
```

可以用一个简单的打包命令，来获得本地的 Docker 镜像，我们就叫它 **fiboapp**：

复制代码

```
1 docker build --rm -f dockerfile -t fiboapp:1.0.0 .
```

然后，我们在 Azure 上新建一个容器注册表，也就是前面提到的容器镜像服务。云上会为我们分配一个镜像服务器的域名：

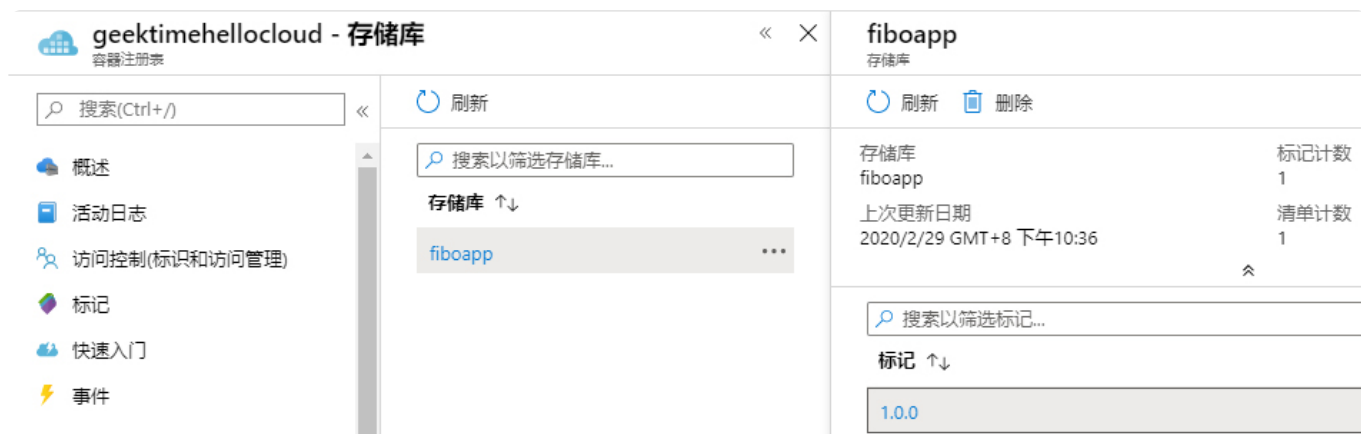


接着我们就可以用标准 Docker 命令登录，并且将镜像上传到这个私有的镜像仓库：

复制代码

```
1 docker login geektimehellocloud.azurecr.io
2 docker tag fiboapp:1.0.0 geektimehellocloud.azurecr.io/fiboapp:1.0.0
3 docker push geektimehellocloud.azurecr.io/fiboapp:1.0.0
```

推送完成后，界面上就显示出了这个镜像的信息。



然后，我们就可以创建一个容器实例，并且指向相关的镜像了：

## 创建容器实例

选择订阅以管理已部署资源和成本。使用资源组(如文件夹)组织和管理所有资源。

订阅 \* ⓘ

资源组 \* ⓘ

容器详细信息

容器名 \* ⓘ

区域 \* ⓘ

映像类型 \* ⓘ

映像 \* ⓘ

映像注册表登录服务器 \* ⓘ

映像注册表用户名 \* ⓘ

映像注册表密码 \* ⓘ

OS 类型 \*

大小 \* ⓘ

新建

geektime-hellocloud

fiboapp

(亚太) 日本东部

☐ 公共 ☒ 私有

geektimehellocloud.azurecr.io/fiboapp:1.0.0

geektimehellocloud.azurecr.io

geektimehellocloud

.....

☒ Linux ☐ Windows

2 vcpu, 4 GiB 内存, 0 gpu

更改大小

随后通过一些特别简单的配置，我们就可以让容器在云上跑起来了。它还贴心地“赠送”给我们一个域名：*fiboapp.japaneast.azurecontainer.io*。

fiboapp

容器实例

搜索( Ctrl + / )

开始 重新启动 停止 删除 刷新

概述

活动日志

访问控制(标识和访问管理)

标记

设置

资源组 (更改)

geektime-hellocloud

状态

正在运行

位置

日本东部

订阅 (更改)

OS 类型

Linux

IP 地址

52.140.236.134

FQDN

fiboapp.japaneast.azurecontainer.io

容器数

1

这就大功告成了。我们用 curl 工具测试一下运行在云容器中的斐波那契服务，一切正常：

复制代码

```
1 client@clientVM:~$ curl http://fiboapp.japaneast.azurecontainer.io/fibo/30
2 Fibo(30) = 1346269
3 Computed by wk-caas-3fa7a6b99b-7703b0c1f33ab2xxxxxxxxx with private ip 10.244.3:
```

至此，我们把斐波那契数列应用就成功地迁移到了容器实例服务上。你可以看到，这个服务为我们准备好了容器运行所需的一切环境，我们需要做的，就只是简单地把镜像打包上传而已。

## 课堂总结与思考

从 Docker 到 Kubernetes，容器生态不断的发展，云原生的技术浪潮已经袭来。不单是我们开发者要学习和拥抱容器技术，各个云计算厂商也都想把自己变成运行容器的最佳场所。所以，云平台们推出了各种各样的容器相关服务，以争夺云上的容器用户。

相信通过今天的介绍，你能够回答这一讲开头提出的两个问题了。**容器和云是相辅相成的，云承载着容器的运行，容器生态也驱动着云的发展。**从运行方式上来看，你既可以轻量方便地在云上运行容器，也可以在云上 Kubernetes 服务的帮助下创建集群，进行较大规模的编排和部署。

我这里还整理了各大云的容器相关服务名称和缩写，你可以参考下面的表格：



服务名称	AWS	阿里云	Azure
K8s容器服务	EKS	ACK	AKS
自有容器服务	ECS	容器服务	ACS
容器镜像服务	ECR	容器镜像服务	ACR
无服务器服务	AWS Fargate	弹性容器实例 (ECI)	ACI

不知道你有没有注意，其实容器与云还有一层微妙的关系，那就是容器与厂商力推的一些云服务，**存在一定的竞争和替代关系**。

就像部分 PaaS 的功能，能够使用 IaaS 来实现一样，容器由于它极高的构建自由度和便捷的封装部署机制，也可以一定程度地替代部分云上的可复用组件（就像我们的实验中，用容器完成了🔗第 12 讲中类似效果的应用部署）。而且，它可以作为系统的一部分参与编排，还是避免厂商绑定的“神器”。

注：从资源编排的角度来看同样如此，K8s 的各种 yaml 配置，和🔗第 8 讲中提到过的 ARM Template 和 AWS CloudFormation 等私有的资源描述方式，同样存在能力交集。

这就是为什么你会发现，像 Google 这样在云计算领域相对后发的厂商，会更热衷于云原生生态的建设，并积极创立和发展 CNCF（Cloud Native Computing Foundation，云原生计算基金会）这样的组织。因为以厂商中立为特点的云原生阵营若能崛起，有助于挑战已经在云计算产品体系中占据优势的老玩家。这是有商业考量的。

当然，不论背后的商业用意如何，业界从碎片化的私有技术到统一的技术标准上面来，这本身就是云原生带来的一种进步，有着非常积极的意义。对我们开发者也是好事情，它帮我们保证了应用的可迁移性。

所以尽管容器一定程度地威胁到了部分云服务，不过云计算再一次体现了技术中立性，云平台们都大大方方地支持和承载了容器的运行，甚至作为重点服务来进行发展。把选择权留给用户，这是云的胸怀所在。

K8s 还在快速地发展，云上各种 IaaS/PaaS 服务也是实力雄厚，两者亦敌亦友，又相互渗透。未来的走向究竟如何，十分值得关注，让我们拭目以待。

**好了，今天同样留给你两个思考题，欢迎你参与讨论：**

后半篇所讲的容器实例类服务，与前半篇讲到的云上 Kubernetes 编排服务，其实并不是割裂的关系。通过集成和调度，不少云上 Kubernetes 服务中的 Pod 能够在容器实例服务中运行。你知道它是通过 K8s 中什么机制来完成的吗？

对于最后讨论的容器与云的微妙关系，你是怎么看的？你觉得未来更多是以 K8s 为中心、云原生吞噬一切，还是云自身的 IaaS/PaaS 产品体系更为强大，K8s 只是云中的一个服务呢？

这一讲我们就到这里。如果你觉得有收获，欢迎你把这篇文章分享给你的朋友。感谢阅读，我们下期再见。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 13 | 云上大数据：云计算遇上大数据，为什么堪称天作之合？

下一篇 15 | 无服务器计算：追求极致效率的多面手

## 精选留言 (7)

写留言



何恺铎 置顶

2020-04-05

[上讲问题参考回答]

1. 在Hadoop的黄金时代，就近访问是诞生在当时网络传输速度远远低于本地硬盘IO的大背景下的，所以它的作用非常大。随着数据中心高速网络技术的发展，网络传输得以不断接近本地IO，反而是算力容易成为瓶颈，所以使用对象存储时计算存储分离等优势就凸显出来了。...

展开 ∨



3



Helios

2020-04-03

第一个问题，可不可以简化为一个k8s集群的POD能够调度到另一个k8s集群上，然后这个

POD还归属与前者集群？如果是这样的话就不晓得了～

第二个问题，随着k8s的越来越成熟，以后所有的PAAS都能跑在k8s上，就像现在都是跑在操作系统上一样，我对云原生吞噬一切持有乐观态度，但是k8s还有很多问题要解决，比...  
展开 ∨

💬 1

👍 3



**leslie**

2020-04-03

关于第一个问题我记得在张磊的课程中看到过，生产没用容器故而也就没钻了。  
第二个问题我倒是看到过相关报道：尤其是Google在开发K8第二代时提及过其中引入一些ACID中的元素去解决一些问题；个人觉得更加偏向是云厂商中的一种服务形式。这就像现在几乎一提及设计就是分布式，可是我记得不少老师在其架构课中都有提及“分久必合，合久必分”；容器化的根本还是要一体机啊，Docker的产生并没有真正的代替VMware...  
展开 ∨

💬

👍 3



**leaf**

2020-04-07

请问云原生是怎么定义的，k8s就是云原生吗？  
展开 ∨

💬

👍



**Michael Yang**

2020-04-03

K8S只会是云服务的一种！  
展开 ∨

💬

👍



**Bora.Don**

2020-04-03

所以这就是serverless的实现方式？  
展开 ∨

💬

👍



**八哥**

2020-04-03

大多数云计算公司容器镜像服务是拿开源得Habor改的。感觉未来serverless服务可能是未来主流模式，期待。  
展开 ∨

