DOIT

2020

# 行业云原生应用报告指南

## Cloud Native Application Guide

# 序

从“互联网+”开始，互联网企业开始取代传统IT企业成为了数据时代的领军者，以开源技术作为“营养液”；借助微服务化、容器、迭代、DevOps、AIOps等一系列原生化应用操作，互联网企业对传统行业/企业进行了无情碾压。

对于传统行业企业来说，惟有奋起直追，加速完成应用的云原生化改造以及创新。从目前的状况看，传统行业企业或者通过公有云，或者通过私有云，基本完成了业务应用的云迁移，接下来的重点应该就是业务应用的云原生改造。为了帮助传统行业企业加速完成云原生化改造，百易传媒联合中国计算机学会存储专委会、武汉光电国家研究中心，特邀Intel、戴尔科技集团、VMware、阿里云、腾讯云、百度云、浪潮、联想凌拓、金山云、华云数据、红帽、灵雀云、博云、青藤云安全、Rancher、青云QingCloud、焱融科技、XSKY、杉岩数据、Portworx、UCloud、才云、京东云、网易云、亚信科技、时速云、七牛云、CSphere希云、云徙科技等国内外领先的云原生应用厂商的专家，共同组织编纂了《行业云原生应用报告指南》，希望能够为传统行业企业提供帮助。

2020年夏

## 特此鸣谢参加本次白皮书的编纂的行业和企业专家。

**谢长生** 华中科技大学武汉光电研究中心教授

**吴　非** 华中科技大学光电国家研究中心研究员、博导

**许　渊** Intel 数据中心产品架构师

**吴　栋** 浪潮集团云计算高级架构师

**林小引** 戴尔科技集团售前系统工程部解决方案架构师

**张利俊** 戴尔科技集团企业技术战略架构师

**傅纯一** VMware 大中华区高级产品经理

**郦　达** 联想凌拓数据与云架构顾问

**李　云（至简）** 阿里云云原生高级技术专家

**刘　尧** 百度智能云云原生和区块链产品负责人

**陈一苇** 腾讯云云原生高级技术专家

**彭德华** 金山云云原生解决方案总经理

**朱　琅** 京东智联云 PaaS 产品负责人

**吴　涛** 华云数据标准总监

**李　鲁** 网易云资深解决方案架构师

**贺洪龙** 灵雀云资深解决方案架构师

**康　洋** 博云解决方案架构师

**梁　胜（西瓜哥）** 高端存储知识 CHO

**薛　浩** 亚信科技平台产品研发与交付中心总监

**Grace Shen** 青藤云安全容器安全技术专家

**苗远强** 时速云解决方案架构师

**庄怀轩** 大连华信技术总监（原 Portworx 亚太区首席架构师）

**周　磊** 云徙科技 i-DP 产品总监

**邵国健** XSKY 系统架构师

**王鹏飞** 焱融科技首席技术官

**黄雨婷** 才云科技智能容器云数字营销经理

**俞仁杰** 蚂蚁金服金融云产品专家

**花　瑞** 杉岩数据存储专家

**于　爽** KubeSphere 容器平台产品经理

**叶理灯** UCloud 私有云及容器产品线负责人

**侯亚辉** 云途腾容器产品总监

**宋家雨** 百易传媒（DOIT）编辑部

**谢世诚** 百易传媒（DOIT）编辑部

**朱朋博** 百易传媒（DOIT）编辑部

**崔欢欢** 百易传媒（DOIT）编辑部

**张妮娜** 百易传媒（DOIT）编辑部

# 目录

# FOREWORD

# 前言

有数据显示：2018 年我国云计算整体市场规模达 962.8 亿元，增速为 39.2%。预计未来几年复合增速在 30% 左右，到 2022 年我国云计算整体市场规模接近 3000 亿元。

国外知名分析机构 Gartner 的数据显示：云计算渗透率将大幅提升，2019 年云计算的市场渗透率将首次突破 10%，达到 11.3%；到 2021 年全球云计算市场渗透率将达到 15.3%。

数据是冰冷的，会有分类和统计上差异，如云计算渗透率专指公有云吗？包括私有云在内吗？尽管如此，并不影响我们对整体趋势的判断。

传统行业 / 企业应用需要迁移到云平台，过程不是一蹴而就的，而是遵循演进阶段的划分，大致可以分为如下 4 个阶段：

**阶段 1：**虚拟化整合 +IaaS 阶段：这个阶段以 IaaS 基础设施虚拟化应用为主。

**阶段 2**：IaaS+ 简单 SaaS 应用阶段：在虚拟化资源基础上，开始对外提交简单 SaaS 服务。在这个阶段，传统行业 / 企业用户部分业务部门，会脱离开 IT 部门的纳管，尝试公有云服务，尝试局部业务的创新。

**阶段 3：**丰富 SaaS 应用 +PaaS 平台阶段：为了能够提供统一的 IT 服务，增强对于 IT 基础设施的把控能力，更多传统行业 / 企业通过构建私有云，对于 SaaS 服务进行集中管理，开始构建符合行业企业发展需要的 PaaS 平台。

**阶段 4：**全面云化、自动化管理和服务的阶段。

既然如此，问题来了！

如今，中国传统行业 / 企业业务云化已经发展到了哪个阶段？

本白皮书观点表明，传统行业 / 企业云计算应用如今已经进入到了第三阶段的 PaaS 平台阶段，这里 PaaS 平台技术非常明确，以平台为基础的云原生化开发，实现对传统应用逐步改造。

从技术上看，基于容器 + 新型 PaaS 平台具有敏捷部署、弹性伸缩、灵活调度的优点，结合、DevOps 和微服务三驾马车，企业可以快速走上云原生转型之路。有数据显示，在美国，容器化部署占到了生产应用部署的 48%，2020 年，超过 50% 的全球组织将在生产环境中运行容器化应用程序，到 2022 年将超过 75%。在中国，截止到 2018 年底，已有 96% 的 IT 企业在生产环境部署容器化应用。

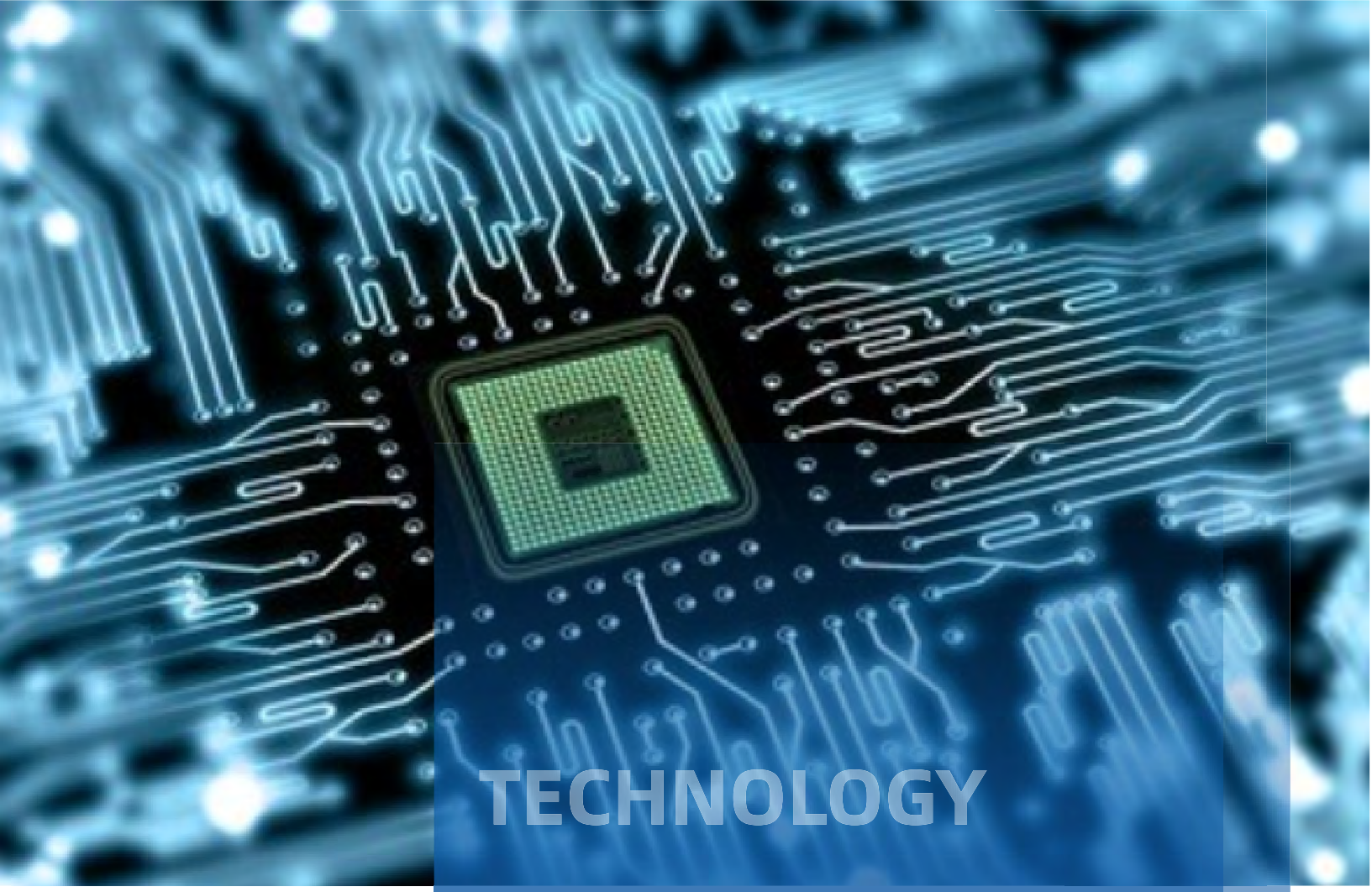
白皮书也强调：构建平台是起点，不是目的，是手段、工具，但不是业务应用，用好容器 + 新型 PaaS 平台加快云原生化部署才是当务之急。与此同时，传统行业 / 企业用户可以直接选用云厂商提供的云原生服务，也可以通过 ISV 等合作伙伴继续沿用服务外包方式，依靠第三方服务商的产品和技术力量，不必事事亲力亲为。

按照白皮书的判断，行业 / 企业传统应用向云上的迁移基本完成，解决了传统应用在云环境下的使用问题，也能够享有 IaaS 基础设施的资源弹性和易部署的红利，但效果是有限的，与类似“双十一”的互联网业务支撑系统存在明显差距。如果说，传统应用上云解决“能用”的问题；云原生化改造要解决的就是“好用”的问题。

业界经常用“稳态”和“敏态”来概括行业 / 企业应用的特征，但这种概括是相对意义上的，是暂时的，因为并不存在绝对意义上的“稳态”，业务创新要打破的就是“稳态”，使其走向“敏态”，走向云原生化应用。

毫不夸张地说，云原生化改造会成为传统行业 / 企业新的分水岭，将决定传统行业 / 企业在未来市场上的“生”与 “死”，依靠政策壁垒，也许传统行业 / 企业还能够生存，但也仅仅是生存，甚至“生不如死”，终究会被时代洪流所吞没。

传统行业 / 企业需要有足够的危机意识！



TECHNOLOGY

# 二、云原生化是“新基建”和“互联网+”的技术表达

## 2.1 传统行业面临的挑战

### 2.1.1 传统行业 / 企业被互联网无情“碾压”

传统行业 / 企业面临互联网企业无情碾压，这已经是铁一样的事实，也是传统行业 / 企业每个从业者的真实感受；新动能替代旧动能，“互联网 +”或者“+ 互联网”已是传统行业 / 企业新课题。

遗憾的是，几年时间过去了，很多传统行业 / 企业仍然没有找到“互联网 +”的节奏。从金融行业的 Open Banking，到电信运营商避免被“ 管道化”，从新能源、新制造到新零售，传统行业 / 企业仍然在摸索。新基建部署也提出了新的需求。

传统行业 / 企业相比互联网企业的业务特点不同，传统行业 / 企业注重 ROI，强调投入产出比。相比互联网企业的特点是投入大，风险大，收益量化有不确定性。但是从技术需求上，“互联网 +”实质是数字化转型的问题，是在 “数据 + 算法” 定义的世界中，以智能数据服务的流动，化解复杂系统的不确定性，优化资源配置效率，构建企业新型竞争优势。

补足云原生应用上的短板，是传统行业 / 企业应对互联网冲击的首要课题。

### 2.1.2 Made in Internet（新零售、新制造、新能源、新技术、新物流）

新基建和“互联网 +”或者说“+ 互联网”从概念上并不难理解，就是利用先进的技术对传统产业进行改造，方向就是所谓新零售、新制造、新能源、新技术、新物流，也就是“Made in Internet ”。

“Made in Internet”应该怎么实现呢？在“天猫”、“京东”、“淘宝”、“拼多多”上开个网店，通过互联网接受订单，利用支付宝、微信支付等电子支持手段结算，这算不算“互联网 +”呢？

答案当然算，但不高级。

对于传统行业 / 企业来说，电商、网店就是一种新销售渠道，因其信息更加透明、物流配送方便，导致销售占比不断提高，如今新旧渠道并存是一个基本现状。但传统行业 / 企业也应该意识到无论是电商渠道，还是传统分销渠道，无形中都割裂了企业和最终消费者间的信息传递。

这种“互联网 +“仅仅触及了传统行业 / 企业销售这一个环节，更何况，传统渠道也在积极探索电商新渠道，因此并不高级。“Made in Internet”要改变的是从研发、市场、制造、销售到服务的全部环节。按照理想化的设计，“Made in Internet”并不需要电商渠道，全部业务流程应该完全构建在互联网平台上，这才是真正的“互联网 +”，云原生化应用是首要待解决的问题。

### 2.2 云原生（Cloud Native）实现

互联网企业应用生来就是云原生化，相比传统行业 / 企业应按照 Client/Server 或者 Browser/Server 的模式构建，本质是一种集中式计算，没有办法支持高并发，也没有办法支持快速迭代。

传统集中式系统，版本更新以年为单位计算，主要依赖传统 IT 产品供应商，二者之间是一种 IT 服务外包方式。相比以分布式为特征的云原生应用，使用微服务化的架构，模块之间呈现一种松耦合的模式。其中，局部修改并不引发全局，让软件开发以“迭代”方式呈现，以此为依托，创新业务以持续“试错”的方式，完成与消费和使用者的磨合。

在迭代模式下，消费者既是使用者，也是新需求和服务的贡献者，软件开发人员根据消费者的需求不断修改，迭代式创新服务，而不是传统研发人员的单向驱动，这种新型消费伙伴关系是传统模型无法比拟的。

以迭代为结果的微服务化云原生应用开发，将涉及研发体系和结构的变化，从单体结构走向微服务化。容器是目前使用最为普遍的开发技术，Kubernetes 是最为普遍的容器编排和管理平台，是笑到最后的技术，事实上的市场标准。目前容器可以部署在物理机，也可以部署在虚拟机上。

与虚拟机相比，容器部署的数量更多，颗粒度更细。微服务化、DevOps、容器带来了云原生应用研发和管理新方法，让传统行业 / 企业能够更好与开源社区最新的软件技术进行结合，支持业务创新和迭代。

### 2.3 云原生与企业竞争力塑造

业内流传着这样一句话：所有企业都是软件企业，实际上，间接回答了云原生与企业竞争力关系的问题。

为什么都是软件企业？

因硬件产品趋于工业标准化，更能够体现企业差别的应该是软件的能力，以互联网厂商为例，他们就是通过充分使用开源技术，重新塑造了软件能力。

传统行业 / 企业多采用购买商业软件套件，采用 IT 服务外包的模式，传统模式的特点是初始购买成本高，软件版本更新迭代缓慢。相比，互联网企业初始购买成本低，软件版本更新速度快，几乎每天都有软件更新，可以随时引入新技术，产品技术创新能力强。

未来，企业的核心竞争能力更多体现在软件能力上。

依据企业中企业应用的不同特征，有些专业分析机构和企业将其区分为“稳态”和“敏态”，划分的主要依据是软件使用者数量的划分，“稳态”业务没有海量并发访问的需求，举例如ERP，主要局限在生产线相关人员使用，这些业务以往是部署在小型机环境中，强调系统的可靠性和稳定性；“敏态”业务，类似“双十一”这样的促销活动，预计会有波峰、波谷的剧烈变动，要求系统具有足够的弹性、敏捷性和灵活性。有人习惯上将传统业务应用称为“稳态”，而将互联网创新业务应用称为“敏态”。

这样的业务划分有一定的现实意义。

但是站在云原生业务应用的角度上，需要注意到所谓“稳态”和“敏态”只是一个相对的概念，并非是一成不变的。以新制造为例，与标准工业化、流程化为特点的传统批量化制造相比，新制造更加强调个性化，从批量化被动的使用者，要变成为个性化产品制造的创造者和参与者，消费者需要参与到产品制造过程中。从技术的角度，意味着传统 ERP 的各个环节需要面向最终消费者开放，消费者希望了解共享物料、设计、制造以及物流等各个环节信息。

在新制造的环境下，业务需求的变化，也将带来新的需求，对于系统来说意味着海量的并发访问，从某种意义说，新制造就是要打破四平八稳“稳态”的格局，所谓不破不立。互联网企业能够走通的路，传统企业没有走不通的理由，这恐怕也是“互联网 +”希望传递的信息。

### 2.4 云原生与企业管理

从单体结构到微服务化到业务应用创新，软件更新迭代快，一天 72 变，意味着没有办法一次性购买终身免疫，云原生能力获得需要方方面面的改变。传统行业 / 企业以往所采用的 IT 服务外包模式没有办法应对随时出现的调整和变化，没有办法更多从“开源”技术的世界中吸取能量，为业务创新所使用。

从根本上说，云原生应用的本质不是产品 / 方案，而是一种管理开发体系的改变，是对行业 / 企业研发体系的重构。

为了实现重构，传统行业 / 企业可以像互联网一样，公开招聘，广纳人才，以金融、电信为例，他们的研发队伍规模、实力并不逊色于互联网厂商，但对于政府、能源、制造、医疗等行业而言，其技术队伍的规模、实力比较薄弱，在“全民软件、全民研发”的大环境条件下，面对面真金白银去抢人，对于传统行业 / 企业难度不小。

传统行业 / 企业可以继续沿用传统的 IT 服务外包的方式，但是应该探索新型合作伙伴关系，以满足和适应云原生化转型和变化的需要。新型合作伙伴关系，需要新的服务和付费方式。对

于现有 IT 人员也提出了更高的要求，需要随时把握开源技术最新动向，并能够与业务创新加以结合，需要协调合作伙伴技术力量，构建新的管理和付费方式。

在中国，企业发展问题都可以归结为现代化企业管理，加之云原生应用牵涉企业应用开发管理体系的重建，因而应该从企业管理的高度重新审视云原生应用和业务创新的问题。

但是企业管理问题也是一个系统庞杂的问题，类似“一把手”工程，需要企业管理者运筹帷幄。将云原生应用上升到企业管理的高度是恰当的，应该有这样一个全局高度的把握，然而过度强调“一把手”工程也会束缚云原生化应用推动的步伐。

目前，公有云、ICT 产品供应商、创新企业等都提供了很多云原生应用技术产品和服务，可以帮助传统行业实现业务创新，从而让云原生化的问题变得没有那么复杂。

不要被企业管理束缚住手脚。这也是白皮书希望阐述的观点之一。

TECHNOLOGY

# 三、容器云原生技术基础（企业云原生转型的技术实施）

云原生应用起源应该追溯到 2006 年谷歌 cgroups 容器技术，也有观点认为这个概念是 Pivotal 公司的 Matt Stine 于 2013 年首次提出的，2013 年 Docker 正式发布让更多人看到了容器，类似于集装箱技术对运输业革命，没有容器就没有云原生应用，容器技术催化了云原生应用，但是云原生并不等于容器（详情参见 3.2 容器和微服务化）。

容器应用自带环境，可将一致容器化应用运行在各种环境中，它便于调试、开发、部署、运维、迁移、扩容，从而造福程序员自己。容器这些特性与云弹性能力相结合，可最大化发挥云的效能，发挥云的价值。

云原生的软件应用生于云上，迭代成长在云上，在云上工作，最后也销毁在云上。为了高效管理容器，2014 年，Kubernetes 项目开源。2015 年，谷歌和红帽牵头成立了 CNCF 基金会，并将 Kubernetes 捐献给了 CNCF。Kubernetes 让容器技术生态迅速成型，云原生由此进入原子弹爆炸的状态。

## 3.1 云原生应用的技术内涵

### 3.1.1 容器化

从以往单体应用到微服务架构应用，部署和管理应用的复杂性大大增加，在 2013 年 Docker 镜像出现以后，容器变成了黑盒子，使用者只会关心它能做什么，需要一个服务的时候，启动一个或者若干容器即可。

容器可以提供很多不一样的服务，也可以提供很多一样的服务，也就是横向扩展，提供控制系统的弹性伸缩。容器化解决了效率问题，如应用开发、测试、部署、迭代等都发生了天翻地覆的变化。以互联网企业为代表，开发人员普遍采用了容器化手段和开发方法。

传统行业 / 企业用户的数字化转型也应该采用容器化的技术，传统应用需要进行容器化改造。容器技术有共享内核和独立内核两种技术，默认都是共享物理机内核，独立内核的比如 Kata Container。

### 3.1.2 容器化与虚拟化

容器化需要首先部署虚拟化吗?

这个问题始终存在争论。有舆论认为，容器化替代虚拟化，容器化是虚拟化的大敌。

虚拟化早于容器化，以 VMware vSphere、微软 HyperV、开源 KVM 等软件为代表，传统行业 / 企业普遍采用虚拟化技术。虚拟化技术重点解决了 CPU 计算资源利用率不高的问题，通过构建虚拟机，提高计算资源的利用效率。虚拟化技术是云计算技术的基础，这也是为什么云计算初期被称为虚拟化的原因。

在 Intel 处理器支持下，CPU 芯片直接部署虚拟化，虚拟化虚拟物理设备，所谓虚拟机，其上安装操作系统，部署应用。

容器化和虚拟化在功能上存在一定程度重合，本意都是屏蔽 CPU 资源调度技术的复杂性和差异性。

从目前的情况看，容器可以直接部署在物理主机上，也可以部署在虚拟机上，特别对于传统行业 / 企业用户而言，虚拟化技术普遍采用，因此需要同时管理容器和虚拟化，相互之间存在交集，虽然虚拟化和容器本质上都是一种计算资源隔离技术，但在隔离程度上，虚拟化更高，安全程度也更高。

云原生应用可以直接部署物理硬件的操作系统中，由于没有虚拟化的开销，更加具有成本效率，它们凭借 Kubernetes 平台对集群中的容器进行管理，这也是有舆论认为容器化将取代虚拟化的动议来源。

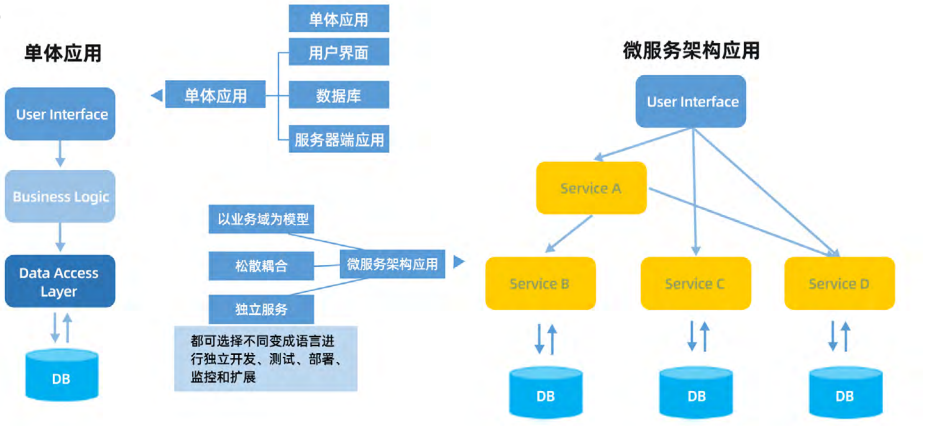
但是这种开销也并非绝对。以虚拟化为例，虚拟化内核与跟 Kubernetes 紧密结合在一起，也就是说，在虚拟化就有对容器调度的 mini 资源嵌入。如此部署容器应用的时候，根本不用考虑是部署在虚拟机上，还是部署在物理服务器上，根本不用关心底层的基础设施。

有厂商数据显示：超过 80% 的容器部署是运行在虚拟机上，其余部署在物理服务器上，很多用户希望物理服务器资源的调配管理能够通过容器编排软件实现。也有第三方的测试显示：借助新的平台对虚拟机和 Kubernetes 进行管理，相比物理机裸机设备部署有 8% 的性能提升，这主要得益于虚拟化对于物理资源的高效调配。很多企业已经基于虚拟化技术建立了整套的运维管理流程，为容器应用建立一套基于物理服务器的运维流程是他们轻易不肯尝试的，这也是他们继续选择在虚拟化技术平台上运行容器应用的原因。

### 3.1.3 微服务：单体化应用和微服务架构的变化

微服务是相对于大型单体应用而言的。

传统的应用开发模式下，随着用户对应用陆陆续续提交新的需求，开发人员需要在原来的应用上不断添加新的功能，这就相当于在一个船上不断修修补补的过程，它的功能越多，也就意味着下次改动需要注意的地方越多，一次改动可能会带来很大的影响。简单说，这是一种紧密耦合的架构。



微服务架构与此前的垂直架构、SOA 架构的一脉相承。

通俗地说，微服务是将单体应用拆分为多个组件，如用户界面、消息队列、数据库访问等，以电商交易为例，拆分为在线支付、订单生成、订单跟踪等多个微服务，其中，每个微服务也可以进一步拆分，微服务之间通过接口进行调用，如此，就构建成了松耦合的架构。

微服务彼此独立，修改、更新、迭代不会牵涉到其他微服务的模块。

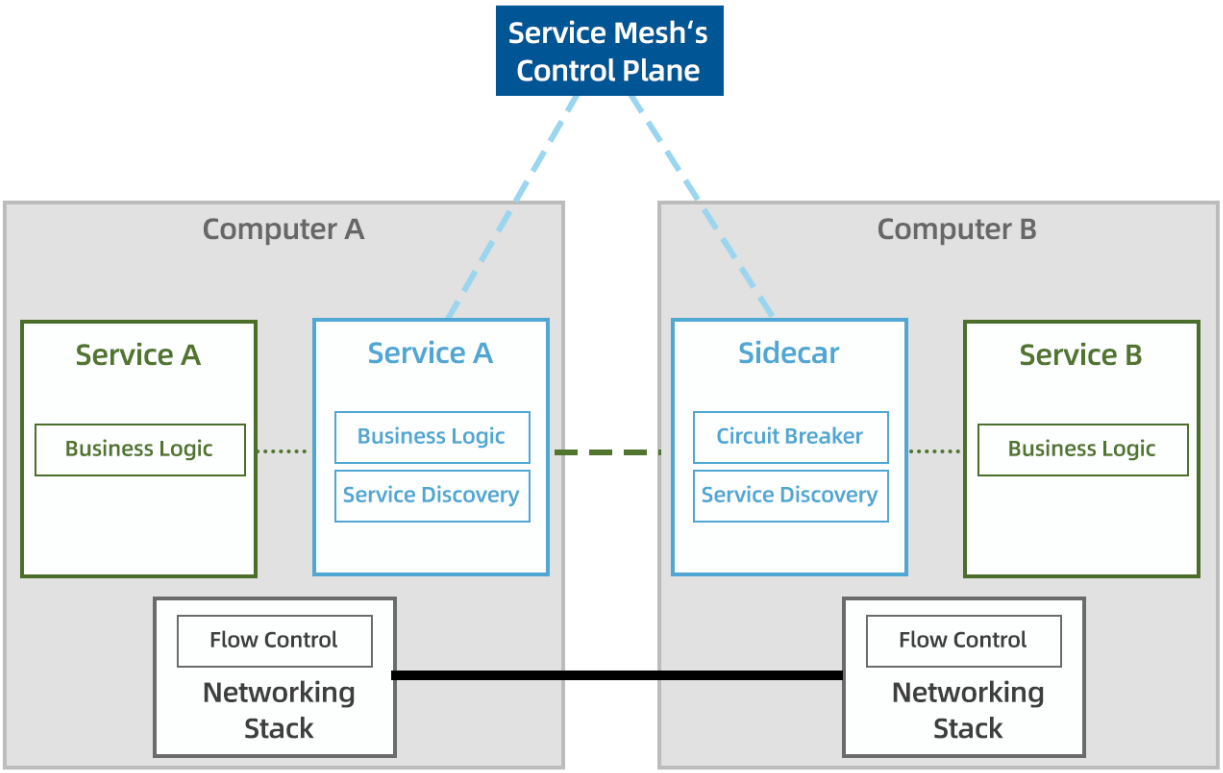
从开发效率来讲，传统单体应用开发模式下，许多人共同面向一个大的应用，就像一群工人围在一起组装一台汽车一样，工序之间相互影响，相互干扰，效率不高。

微服务化则把汽车生产变成了工业流水线，一个工人只负责其中一部分，每个工人只需要熟悉自己负责的那部分就好，每一部分可被独立完成，独立部署和更新。

微服务之间通过 RESTful API 进行连接。由于微服务之间以及微服务与客户端之间的连接要很快速，而且消耗要尽可能的低，REST API 就非常合适。

Service Mesh 也是如今谈论和使用最多的技术，Service Mesh 是用于处理服务间通信的基础设施层，用于在云原生应用复杂的服务拓扑中实现可靠的请求传递。

Service Mesh 与传统基础设施层不同之处在于，它形成了一个分布式的互联代理网络，以 sidecar 形式部署在服务两侧，服务对于代理无感知。



目前发部分 Service Mesh 只是开源项目，比较知名的项目有：Linkerd、Istio 和 HashiCorp Consul 等。

• Linkerd：2016 年发布，是这些项目中最早的。Linkerd 是从 Twitter 开发的 library 中分离出来的。在这一领域另一位重量型选手——Conduit，已经进入了 Linkerd 项目并构成了 Linkerd 2.0 的基础。

• Envoy：由 Lyft 创建，为了能够提供完整的 service mesh 功能，Envoy 占据“数据平面”的部分，与其进行匹配。

• Istio：由 Lyft、IBM 与谷歌联合开发，Istio 可以在不修改微服务源代码的情况下，轻松为其加上如负载均衡、身份验证等功能，通过控制 Envoy 等代理服务来控制所有的流量。此外，Istio 提供容错、金丝雀部署、A/B 测试、监控等功能，并且支持自定义的组件和集成。

Rancher 2.3 Preview2 版本上开始支持 Istio，用户可以直接在 UI 界面中启动 Istio 并且可以为每个命名空间注入自动 sidecar。Rancher 内置了一个支持 Kiali 的仪表盘，简化 Istio 的安装和配置。这一切让部署和管理 Istio 变得简单而快速。

• HashiCorp Consul：与 Consul 1.2 一起推出了一项名为 Connect 的功能，为 HashiCorp 的分布式系统添加了服务加密和基于身份的授权，可用于服务发现和配置。

这里重点说说 Istio，它被视为 Service Mesh 最流行的实践，Istio 一些关键性功能包括：帮助微服务之间建立连接，帮助研发团队更好的管理与监控微服务，并使得系统架构更加安全。

Istio 帮助微服务分层解耦，解耦后的 proxy 层能够更加专注于提供服务发现、负载均衡、故障恢复、服务度量、服务监控、A/B 测试、灰度发布、限流限速、访问控制等基础架构的能力。

使用微服务需要治理平台支撑，经过微服务化拆分之后，服务从本地调用变为远程方法调用，各种不确定因素变多了，必须有微服务治理平台来应对，常见的有 Spring Cloud 、Apache Dubbo、Spring Cloud Alibaba、Apache Service Comb 等等，其中包括服务发现、服务订阅、服务监控、服务追踪、服务日志等。

在选择微服务框架时需要有几个简单的考虑原则：工具栈丰富度（微服务技术栈的深度和广度）、通用性（社区活跃程度、使用比例等）、开放性（与其他开源系统的兼容性和互操作性）和可移植性（业务代码无侵入或低侵入）等。

当一切就绪，平台上发布了成百上千个服务，每个服务或微服务相互连接成网状，如何让这个网络有序、有逻辑便非常重要，一来便于修改和维护，二来保证稳定和性能。

如何让网络变得有序有逻辑呢？

治理平台首先梳理系统对外开放的服务化接口、服务分组以及动态路由等等，然后对拆散的服务进行归类、界定，确定服务领域和服务边界，最后通过服务迁移、切换，对同一领域的服务接口进行服务整合，提供统一的服务出口，实现服务编排。

领域驱动设计（Domain-Driven Design，DDD）是软件行业最新的方法，当拿到一个需求的时候，DDD 强调应该首先考虑要解决什么问题，它的问题域是什么，而不是先考虑用哪种技术去实现。这里出现了域的概念，域可以简单地理解为微服务化拆分后的又一个整合，域是同类功能或者问题的一个集合。

从领域驱动设计的角度来看，探讨需求首先应该从问题域出发，探讨关于业务边界的事情，梳理要实现哪些功能和需求。下一阶段拓展到工作边界，探讨如何分工协作，确定团队合作的方式。最后，到达应用边界，即技术实现的解决问题领域。

微服务也是对大数据、AI、物联网的有效支撑。

开发运营平台收集处理数据，用包括数据库、数据仓库、数据湖的方式存储海量数据，采用各种先进的大数据和人工智能 AI 技术对数据进行挖掘，从数据中心获取洞察力，为产品优化，降本增效提供支撑。

大数据、AI、物联网等都需要各种不同的运算、存储以及网络传输能力，而基于 Kubernetes 提供容器微服务可以动态提供资源和平台支撑，微服务在可管理性上、成本、提升效率和质量上都有明显优势。另一方面业务架构的微服务化打破了数据孤岛，更方便获取业务数据用于大数据分析。

### 3.1.4 DevOps

DevOps 是 Development（开发）和 Operations（运维）的组合词，DevOps 是一组过程、方法与系统的统称，通过一系列自动化流程能使构建、测试、发布变得更加快捷、频繁和可靠，可用于促进软件开发、技术运营和质量保障（QA）部门之间的沟通和协作，最终为软件产品和服务的及时交付提供保障。

DevOps 领域最明显可见的在工具层面，常用的工具包括：监控工具 Prometheus、Zabbix、Nagios，性能分析 / APM 工具 Newrelic、Oneapm、Pinpoint、Zipkin，批量 + 自动化运维工具 Puppet、Ansible、Chef、Saltstack，日志分析工具 Graylog、Nagios、Elastic Stack、LOGalyze、Fluentd，持续集成 / 发布工具 Jenkins, 中国自己的开源工具 Cyclone。

AWS 对 DevOps 的定义是：DevOps 集文化理念、实践和工具于一身。可以提高组织交付应用程序和服务的能力。与使用传统软件和基础设施管理流程相比，能够帮助组织更快地发展

和改进产品。这种速度使组织更好地服务其客户，并在市场上高效地参与竞争。

DevOps 追求的是一种理想化的协作状态，涉及开发、测试、产品、项目管理、运维人员等等，在 DevOps 中，开发人员专注于业务应用的生命周期管理，运维人员专注于自动化环境资源的维护，QA 人员专注于自动化业务运行环境的供给和质量跟踪保证。许多人认为，所有能在保持质量的前提下提高交付效率的方法和方法论都属于 DevOps 的范畴。

无论国内还是国外，行业内还没有一个能广为接受的 DevOps 实践模型来作为参考，DevOps 本身有很高的实践性和灵活性的特点，在实践中，很多企业也确实从 DevOps 中获益良多，但对于更多企业来说，没有成熟的参考模式做起来就会比较困难，这要求企业能提供试错的空间，也就意味着会出现意料之外的成本。

一个比较容易接受的路径是，由实际的业务来驱动，当企业开发一些新兴业务的时候，就以 DevOps 来进行，一味的对原有系统进行改造是得不偿失的。

那么如何构建业务流程梳理与度量体系完善 DevOps 呢?

**通过职责梳理。**确定流程与职责的对应关系，在实践中，通过对照制度发现各个部门在职责方面的问题，做出弥补和修正。

**通过流程诊断实现流程优化。**依据制度对流程进行审核，发现流程与制度之间的不一致后，要识别改进或优化。流程的跨部门审核、各种管理制度审查完善，结合控制要求进行流程诊断，根据诊断结果完成流程优化工作。流程优化工作，要在整个架构的框架下审视流程，既要保证流程完整顺畅又要关注流程之间工作步骤的衔接关系。

DevOps 度量体系可以帮助负责人了解团队的生产力，可以帮助了解目前应用的状况，可以回答三天后能否上线的问题，而不是“大概”“可能”这类回答，帮助团队认清并提升团队的生产力。

度量体系中的引领性指标定义了一些达成目标关系最为重要的行动，在可以驱动的指标事情上倾斜资源，为实现滞后性指标提供支撑。引领性指标关注目标如何完成，而滞后性指标关注目标是否完成，滞后性指标的数据比较容易获得。

### 3.1.5 持续交付、频繁交付、快速交付、降低发布风险 CI/CD

CI 的英文名称是 Continuous Integration（持续集成）。CI 中，开发人员将会频繁地向分支提交代码，提交的代码在经过编译和自动化测试后，最终合并到主干。持续集成的目标是快速确保提交的修改是不是好的，确定新代码能不能和原有代码集成起来。

CD 可以是指持续交付 Continuous Delivery，也可以是在说持续部署 Continuous Deployment。

持续交付：持续交付负责将 CI 之后的代码发布到存储库，持续交付维护了一个可随时部署到生产环境的代码库，可供终端用户使用。持续部署是将持续交付的代码自动发布到生产环境。

持续交付表示的是一种能力，而持续部署表示的则一种方式。持续部署是持续交付的最高阶段，上文提到，所有能在保持质量的前提下提高交付效率的方法和方法论都属于 DevOps 的范畴，所以，CI/CD 是 DevOps 的必不可少的部分。

在实践中，Jenkins 是最为常见的 CI/CD 工具。

微服务对系统的扰动很小，每次改动也很小，如果做一个比喻的话，单体应用就好比射箭，需要远距离瞄准发射，开弓没有回头箭，射出去的箭很难再调整方位，而微服务则像近身肉搏，拳脚都可以快速出击，而且可以随时调整出拳的方位和力道。

同样的，应用代码通过持续频繁快速的交付，可以降低发布的风险。

### 3.2 容器和微服务化

了解了微服务化，也了解了容器，微服务化和容器之间是什么关系呢?

可以说，微服务与容器没有直接关系，没有容器其实也可以有微服务。那么，为什么人们将微服务和容器化放在一起谈呢?

因为容器作为基础设施为微服务提供了很大便利，我们知道微服务是应用软件架构设计模式，推崇单一职责、服务自治、轻量通信和接口明确等原则， 而容器由于其 Build Once，Run Everywhere 的特性，使得微服务易于开发和维护、按需伸缩等。

其次，Kubernetes 的编排调度能力，使得管理为数众多的微服务不再痛苦，特别是 CRD 的引入，让 Kubernetes 具备了理解和管理各种类型的业务架构的能力。如今越来越多的技术开发人员都通过容器打包应用，使用 Kubernetes 平台部署和管理微服务。

如果说微服务化是目标，那容器和 Kubernetes 则是实现这一目标的手段和方法，这样的说法未必严谨，但确实是当下的现实。

### 3.3 容器和 Kubernetes

Kubernetes 是什么?

Kubernetes 是一个跨主机集群的、开源的容器调度平台，它可以自动化应用容器的部署、扩展和操作，提供以容器为中心的基础架构。

简而言之，Kubernetes 具有强大的容器编排能力，如果没有 Kubernetes，容器面对大规模容器时完全无能为力。

Kubernetes 是谷歌在 2014 年开源的，2017 年，Kubernetes 战胜 Docker Swarm 和 Apache Mesos，成为容器管理与调度编排领域的首选平台和事实标准，Kubernetes 被称为云计算时代的 Linux 操作系统，是全球顶级开源项目。

Kubernetes 无论是技术还是生态都获得了各方支持，几乎所有的公有云服务商都提供了 Kubernetes 方案，允许在云上快速构建 Kubernetes 集群，进行各种配置和管理。所有的私有云以及混合云提供商也都在以各种方式对容器应用提供支撑，容器化的趋势是真正的大势所趋。

### 3.4 技术应用热点

TensorFlow、Jenkins、ElasticSearch、WordPress、Spark、MySQL 等这些经常见诸报端的词汇，也是互联企业经常使用的技术，它们也让云原生世界更加富于技术的色彩，让世界变得精彩。

对于行业企业来说，什么时候会使用到这些技术呢？最好的办法就是了解这些的概念和内涵，根据需要加以选择和使用。

**TensorFlow：**是一个采用数据流图 (data flow graphs)，用于数值计算的开源软件库，简单点说，就是一个机器学习框架，可以用它来训练模型并推理模型。类似的，国内有百度 paddlepaddle。TensorFlow 有个针对 Kubernetes 和云原生的对标开源项目 Kubeflow。

**Jenkins：**是一款开源 CI&CD 软件，用于自动化各种任务，包括构建、测试和部署软件。Jenkins 是使用的最为普遍的软件之一。

**ElasticSearch：**是一种流行的企业级搜索引擎，基于 Lucene，它提供了一个分布式多租户能力的全文搜索引擎，具有搜索、分析和探索的能力，可以用于搜索各种文档。

**Spark：**Apache Spark 是为大规模数据处理而设计的快速通用的计算引擎。Spark 拥有 Hadoop MapReduce 的优点，但有所不同，与 Hadoop 相比，在某些工作负载方面表现得更加优越，算是对 Hadoop 的补充。

**MySQL：**MySQL 是非常著名的开源关系型数据库管理系统，是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL 是最好的关系数据库管理系统应用软件之一。

**WordPress：**是一个开源的内容管理系统（CMS），可以用来搭建个人博客或者新闻网站，安装简单易用，WordPress 有丰富的插件和模版，有强大的生态支持和用户群体。

### 3.5 云原生应用有关思考

#### 3.5.1 从 Web 访问接入开始更加稳妥吗？

从谨慎的角度出发，传统行业用户习惯从边缘业务入手，采取农村包围城市的策略。这样的方法有助于熟悉技术、锻炼队伍，也是很多用户乐于采用的。

在云原生应用问题上是否应该也采取类似的策略呢?

类似于公有云应用中出现过的现象：有些业务部门会绕开 IT 部门采购公有云服务，特别一些创新业务，很容易上手，有立竿见影的作用。但是对于 IT 部门来说，这些应用脱离了监管，给行业合规管理带来了风险。

对于云原生应用来说，如果从访问端开始入手，着重解决并发访问的问题，对于改善用户体验会有一定的帮助，同时也可以锻炼队伍，但是缺点也是显而易见的，核心访问数据库仍然是瓶颈，没有从根本上解决问题。

从接入端入手，给人的感觉是头痛医头，脚痛医脚，没有从战略高度，统筹规划业务进程。前面说过，云原生业务改造和创新，并不仅仅是引入新技术方案，而是涉及到研发体系到企业管理，以及业务创新管理的大问题，对此，需要从战略高度审视问题。

### 3.5.2 数据库原生化改造

针对高并发访问，传统数据库也提供了分库、分表等解决方案，有数据显示，Oracle 18c 数据库的极致性能每秒可以支持 523202 笔交易，这几乎就是“双 11”的峰值交易量。换句话说，18c 也可以支持“双 11”交易的需要。18c 自治数据库也采用了松耦合的模式，由数据层面和管理层面构成，用于应对以往紧耦合带来的问题。

如今也有很多用户开始尝试开源数据库。对于用户来说，历来视数据库为核心业务，更加强调稳态业务应用，牵一发动全身，用户通常不愿涉险。

从互联网厂商的实践来看，通过并行化、分布式部署，实现了对“双 11”规模交易的处理，取得了云原生化改造的成功。除了 MySQL 等各种开源数据库之外，华为、金山云、青云、蚂蚁金服、AWS 等也推出了新的数据库产品，支持海量并发访问的需要。

技术上，分布式数据库存在 CAP 定理又称 CAP 原则，指的是在一个分布式系统中，Consistency（一致性）、 Availability（可用性）、Partition tolerance（分区容错性），最多只能同时具备三个特性中的两个，三者不可兼得。

由于 CAP 原则存在，仅从技术角度解决问题是不可行的，还需要从业务角度进行拆分，以银行的转账交易为例，就可以将集中模式下的“同一数据库本地事务”拆分为分布式模式下的分库模式，将扣款、收款不用数据库来完成，如此就会带来扣款 / 收款数据不一致的风险，对此，可以借助冲正和对账等业务手段解决问题。

此外，分布式数据库可以通过容器进行部署，但容器部署大多应用在前端的访问接入上，后端数据库交易处理大多还是部署在虚拟机的环境中。如今，云原生数据库有很多实现的方案，需要用户结合实际需求，合理进行选择。

### 3.5.3 选 SaaS 服务，还是自建?

对于传统行业用户来说，选择 SaaS 服务方式也是一个行之有效的方法，通过选择 SaaS 服务，屏蔽对于底层基础设施复杂管理，将基础设施等技术问题交给服务商，集中精力于业务创新，这也是“世界只有一种云，就是公有云”论断提出来的原因。

当用户尝试和探索一些创新业务，可以先从 SaaS 的方式开启云原生的大门，在尝试中感受云原生与自身业务或者未来业务发展方向的契合度进行下一阶段的计划，考虑部署从云上到云下的云原生基础架构。

云原生是一套技术体系和流程体系，市场上有各种方案和工具，用户可以从云服务厂商，也可以从第三方专业厂商获得产品和服务，这些产品方案和工具体现了设计和管理流程思想，但云原生应用仍然需要业务的结合，需要用户调整组织与业务流程，在实践与摸索中找出一条云原生创新路径。

### 3.5.4 应用微服务化拆分的一般原则

微服务拆分是将应用程序拆分为松耦合、可独立部署的一组服务的过程。拆分过程无统一标准，但是可以通过以下方法来提升拆分的质量，避免因拆分不当造成的形式上是分布式部署但是服务之间耦合紧密的分布式单体应用。

分解业务问题，面对复杂性，需要将问题分解成可管理的块。不必考虑问题的所有细节，而是将问题抽象地分解成几个关键部分，然后寻找这些部分之间存在的关系。在微服务架构中，需要将业务问题分解成代表离散活动领域的块。这些块封装了与业务域特定部分相关联的业务规则和数据逻辑。微服务不会封装执行单个事务的所有业务规则，当遇到需要跨业务领域不同部分的一组微服务来完成整个事务时，需要通过查看数据域中那些不适合放到一起来划分微服务的服务边界。可以通过以下方式来识别潜在的微服务：

1. 描述业务问题，特别注意用来描述问题的名词。在描述问题时，反复使用的同一名词常意味着它们是核心业务领域并且适合创建微服务。

2. 注意动词，动词突出了动作，通常代表问题域的自然轮廓。

3. 寻找数据内聚，将业务问题分解成离散的部分时，要寻找彼此高度相关的数据。如果在一次业务逻辑处理过程中，突然读取或更新与迄今为止所调用的内容完全不同的数据，那么就可

能还存在其它候选服务。微服务应完全拥有自己的数据。

确定服务粒度，构建微服务架构时，粒度的问题很重要，可以采用以下方法来确定正确的粒度大小：

1. 在微服务设计之初可以让微服务涉及的范围更广泛一些，然后将其重构到更小的服务，在微服务构建之初，容易出现的一个极端情况就是将所有的业务都变成微服务。但是将问题域分解为小型的服务通常会导致过早的复杂性，导致微服务变成了细粒度的数据服务。

2. 重点关注服务如何相互交互，这有助于建立问题域的粗粒度接口。从粗粒度重构到细粒度是比较容易的。

3. 随着对问题域的理解不断增长，服务的职责将随着时间的推移而改变，通常来说，当需要新的应用功能时，微服务就会承担起职责。最初的微服务可能会发展为多个服务，原始的微服务则充当这些新服务的编排层，负责将应用的其他部分的功能封装起来。

定义服务接口即应用程序中的微服务该如何彼此交互。使用微服务构建业务逻辑时，服务的接口应该是直观的，开发人员应该通过学习应用程序中的一两个服务来获得应用程序中所有服务的工作节奏。可使用以下方法来定义服务接口：

1. 拥抱 REST 的理念，REST 对服务的处理方式是将 HTTP 作为服务的调用协议并使用标准 HTTP 动词 (GET、PUT、POST 和 DELETE)，围绕这些 HTTP 动词对基本行为进行建模。

2. 使用 URI 来传达意图，用作服务端点的 URI 应描述问题域中的不同资源，并为问题域内的资源的关系提供一种基本机制。

3. 请求和响应使用 JSON--JavaScript 对象表示法 ( JavaScript Object Notation,JSON) 是一个非常轻量级的数据序列化协议，并且比 XML 更容易使用。

使用 HTTP 状态码来传达结果—HTTP 协议具有丰富的标准响应代码，以指示服务的成功或失败。

TECHNOLOGY

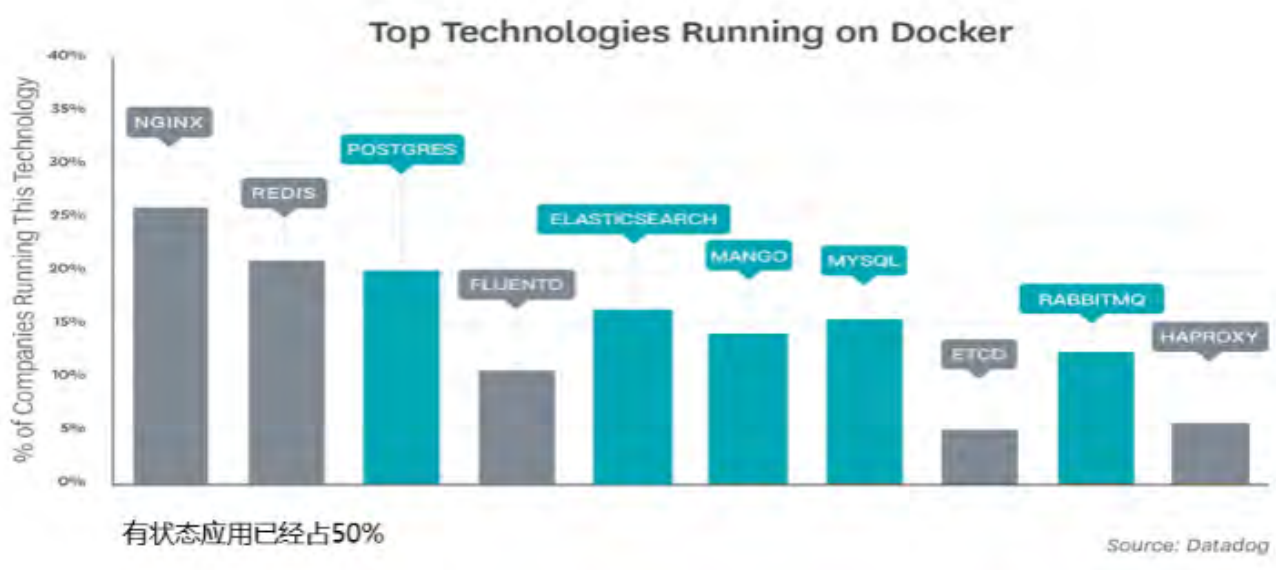
# 四、云原生应用对数据基础设施的影响

云原生应用的浪潮不可避免会对 IT 数据基础设施带来一系列影响和变化，最先接触的首先是存储的变化。

如今越来越多的容器被部署到生产环境中，或者部署在物理裸机，或者部署在虚拟机（VM）中。根据 Sysdig 2018 年收集的数据，生产环境部署的容器数量已经达到了 90000 个。

容器的特点是召之即来，挥之即去。对于无状态应用的如此，但是对有状态的应用，有数据显示，有状态的应用已经达到 50%，如 MySQL、PostgreSQL、MangoDB、RabbitMQ、ElasticSearch 等，应用已经比较广泛，但问题也随之而来，当容器“死了”，可以快速重建，但是数据呢？也能够快速重建吗？这也是容器持久化存储的来源。

## 4.1 存储的变化




### 4.1.1 容器数据持久化

Docker 容器在设计之初，为了实现容器实例重启后，业务运行行为和镜像最初设定的行为一致，设计了一个基于分层镜像来运行的架构，每个镜像都是只读模式，容器在运行时会在指定的镜像上附加一层“临时”可写的文件系统层，容器实例运行时写入的任何数据都只是临时的，实例重启后，这些写入的数据都会被新的“临时”可写层覆盖。

由于这个特点，对于需要数据保持的应用而言，为了保持业务连续性，要求能够确保在容器重启之后数据不丢失，从技术上说，要为容器提供持久化存储支持。如今，业界普遍采用 Kubernetes 对于容器进行编排和调度，Kubernetes 已经成为事实上的标准。

Kubernetes 定义了 Flex Volume、CSI 两种存储接口方式，可以通过 API 调用外部存储为容器提供持久化数据访问服务。可供选择的外部存储，如 CephRBD、Ceph iSCSI 等，通过

yaml 文件创建 PV（Persistent Volume，持久化卷），为容器提供存储。

### 4.1.2 CSI 和容器存储?

但是支持 Flex Volume、CSI 两种存储编排接口就可以称为容器存储了吗？从编排接口实现方式看，FlexVolume 对 Kubernetes 平台有一定侵入性，需要修改一些 Kubernetes 代码，CSI 对此进行了改进，只需要在 Kubernetes 的 Master、计算节点及其他相关节点以容器的形式运行 CSI 插件，即可使 Kubernetes 通过 CSI 编排和使用底层存储。因此，CSI 编排接口现在被广为推崇，在 Kubernetes CSI 驱动的目录中，从 A 到 Z 排列，可以看到很多存储产品，这些产品对于容器持久化存储的支持，效果都是一样的吗？用户可以任意选择吗？

应该说，这些产品解决了容器平台与存储的对接问题，让容器可以使用存储资源，但是持久性存储更加重点要解决的是容器计算故障节点情况下，数据快速恢复访问的问题。一个故障计算节点往往会包含数十或者百个容器，传统的基于块设备方式为容器提供存储的方案，需要平台为每个容器 attach 或 detach 才能连接或断开块设备，在计算节点故障时，如果每个容器都需要借助 detach 并重新 attach 到新节点，加上，attach、detach 不仅耗时且容易出错，这样的操作在现实应用中，效果不佳。

为了解决 attach、detach 块设备的效率问题，最好的办法就是要使 Kubernetes 集群中的各个计算节点可以实时共享访问各个 PV（持久化卷），业界通过采用共享文件系统或类似手段作为容器存储达到这个效果。

目前 Kubernetes 定义了三种访问 PV 的读写模式：ReadWriteOnce(RWO)、ReadOnlyMany(ROX)、ReadWriteMany(RWX)，其中 RWX 读写模式不可或缺，通过 RWX 读写模式的支持，可以使多个容器业务共同读写同一个 PV 内的数据，实现数据的共享。

当采用共享文件系统作为容器存储时在性能方面，设计的关键是 MDS（MetaData Service，元数据管理）不能出现访问瓶颈。以机器学习 AI 容器应用为例，越来越多的传统行业依靠 AI 进行业务改造，而将 AI 应用运行在容器环境下是业界主流趋势，AI 业务数据训练中需要依赖大量的小文件，规模会达到数十亿级别，这种规模下容器存储的性能是否有衰减，很大程度上考验的是元数据的能力；此外高性能计算也将涉及到海量文件的处理，高性能计算业务的 IO 特点是浪涌式 IO，就是瞬时 IO 很高，要求底层文件系统不仅需要处理好元数据瓶颈，还需要提供高 IOPS 和大带宽，能承接上层应用数据。其中就涉及到 MDS 管理瓶颈的问题，好的解决方案利用可水平扩展的元数据集群架构，采用合理的元数据管理放置、检索和管理算法，来确保 MDS 的访问效率。技术上有动态子树、元数据分片，或者通过高性能硬件集中管理等方式。

此外，存储 IO 性能方面，可以采用文件切片来保证文件读写性能；并通过支持 RoCE 或 InfiniBand 网络，借助 RDMA 访问协议让性能提升。

热点目录也是造成性能瓶颈的来源之一，特别对于采用分布式文件系统设计的容器存储而言，当大量业务集中访问某个单一目录下的文件，就会容易形成热点目录，热点目录并行处理能力不足，就会导致系统崩溃，或者性能低下。为此，业界一些容器存储会采用类似虚拟目录的设计，将热点目录的元数据分散至更多元数据节点，一方面增强热点目录并行处理能力；另一方面保持上层应用的目录结构和访问方式不变，做到对容器应用的透明。

### 4.1.3 容器存储可视化和管理

容器和虚拟机的一大差别是隔离粒度更精细了，虚拟机隔离的单位是整个虚拟操作系统，而容器则精细到为每个服务或组件隔离出一个逻辑上独立的运行环境，因此，容器平台上运行的容器数量要远远多于虚拟机平台上运行的虚拟机数量，大多数容器至少需要一定数量的 PV 用于存放日志或其他持久化数据，如何管理、监控、观察这些 PV，也是容器平台和容器存储区别于虚拟机平台的不同之处。

容器数据持久化之后，在实际应用的过程中，管理数据也是用户应该关注的问题，好的容器存储产品应该能够一方面满足应用平稳运行的需求，另一方面满足系统管理员日常管理的需求，如提供对故障感知的调度策略、QoS、PV IO 压力跟踪和定位、Prometheus 监控、以及对容器存储 PV 的细粒度监控和管理等功能。

以故障感知为例，当 Kubernetes 在创建和调度新的具有数据持久化需求的 Pod 时，通过该功能，Kubernetes 调度器能够自动过滤掉 CSI 插件容器异常或与容器存储集群连接异常的工作节点（Worker Node）。QoS 能帮助系统控制应用对有限资源（如 IOPS、带宽等）的使用，能够对其进行设置与管理，从而避免有些容器应用因为资源不足，被“活活饿死”的现象。此外，在 PV 数量倍增的背景下，跟踪、监控、定位 IO 压力大的 PV，能帮助容器云平台用户更好的低管理容器应用。管理员在容器存储平台上，快速查看 PV 内数据层次以及数据温度，进行分析和调整；弹性伸缩 PV 大小，界面化显示 Pod、PV、PVC 之间的关联关系甚至 Pod 的运行数据，这种管理调度及监控的能力，对于容器系统平稳运行保障，效果显著。

Prometheus 作为 Cloud Native Computing Foundation（CNCF）中的重要一员，其活跃度仅次于 Kubernetes，现在已经成为主流监控系统。大多数客户会将容器存储中的监控数据通过 Prometheus 进行收集及后期展示。

### 4.1.4 海量数据和容灾备份

在容器化的应用中，不同的 Pod 共享访问海量数据属于其中一种典型场景（例如 Drupal、WordPress 等内容管理系统，或图片识别、视频编码、视频渲染等应用）。

这些应用场景需要同时满足容器持久化数据共享访问、高性能、低成本三方面要求，其中，高性能和低成本本身就是一对矛盾。以闪存为例，闪存性能高但是价格贵，相比，磁盘便宜但性能差，因此兼顾高性能和低成本，就需要系统设计就具有数据智能分层的能力。

统计数据显示，80% 的数据具有较为明显的访问周期性特征，超过一定周期后，数据会逐步趋冷，随后，应用对这些冷数据的读写变得极少。因此，容器存储可以将热数据保存在具有 SSD 的高性能数据层中，冷数据层直接接入任何第三方提供的具备 S3 标准接口的对象存储。

容器应用在访问数据时，应该做到对数据所处的层次完全无感知。用户可自定义冷热数据策略及冷数据自动执行跨层迁移的时间，数据在冷热数据层之间迁移的过程中仍然可读可写，将数据跨层流动的影响做到最小。

在有状态容器应用高可用方面，有状态 Pod 所在计算节点故障，Kubernetes 将 Pod 跨节点重建后，容器存储应该让 Pod 能在秒级快速完成对数据的访问，从而保证当容器计算节点出现故障时业务的连续性。比容器应用更高层次的是数据中心级别的高可用，跨数据中心容器双活的问题本质是数据的双活问题，容器存储可以通过副本放置策略，在两个数据中心分别放置副本，副本间保持强一致，如果能做到数据中心内的应用优先读取本地副本，即可在保证数据中心双活的基础上，进一步降低数据读延时，这种容器存储双活的方案，可供企业选用容器存储时参考。

#### 4.1.5 存储趋势变化促进容器发展

存储趋势的变化也会带来容器应用的变化。如今，以容器应用为核心的云原生应用是通过 IO 接口访问数据持久性存储，SSD、HDD 还是主要的存储设备，未来 SCM（Storage-Class Memory，存储级内存 ) 发展也会原生应用带来更好的性能，以及数据访问的方式。以 3D Xpoint 为代表 Optane 的两种产品形式，Optane Memory 和 Optane SSD 为用户带来的新的选择，未来的数据存储完全可以存储在 Optane Memory 这种非易失内存存储介质上，带来优于 SSD 的性能表现。

这种进步更多表现在存储硬件及软件层面的变化，其影响不限于云原生应用，但是云原生应用也会因为存储技术的进步，有很多新的应用方式，带来新的表现。以互联网企业为代表，新的存储软硬件技术正在不断走入实际应用过程中。

### 4.2 计算、网络的变化

云原生应用发展也对网络传输和 CPU 计算提出了新的需求。

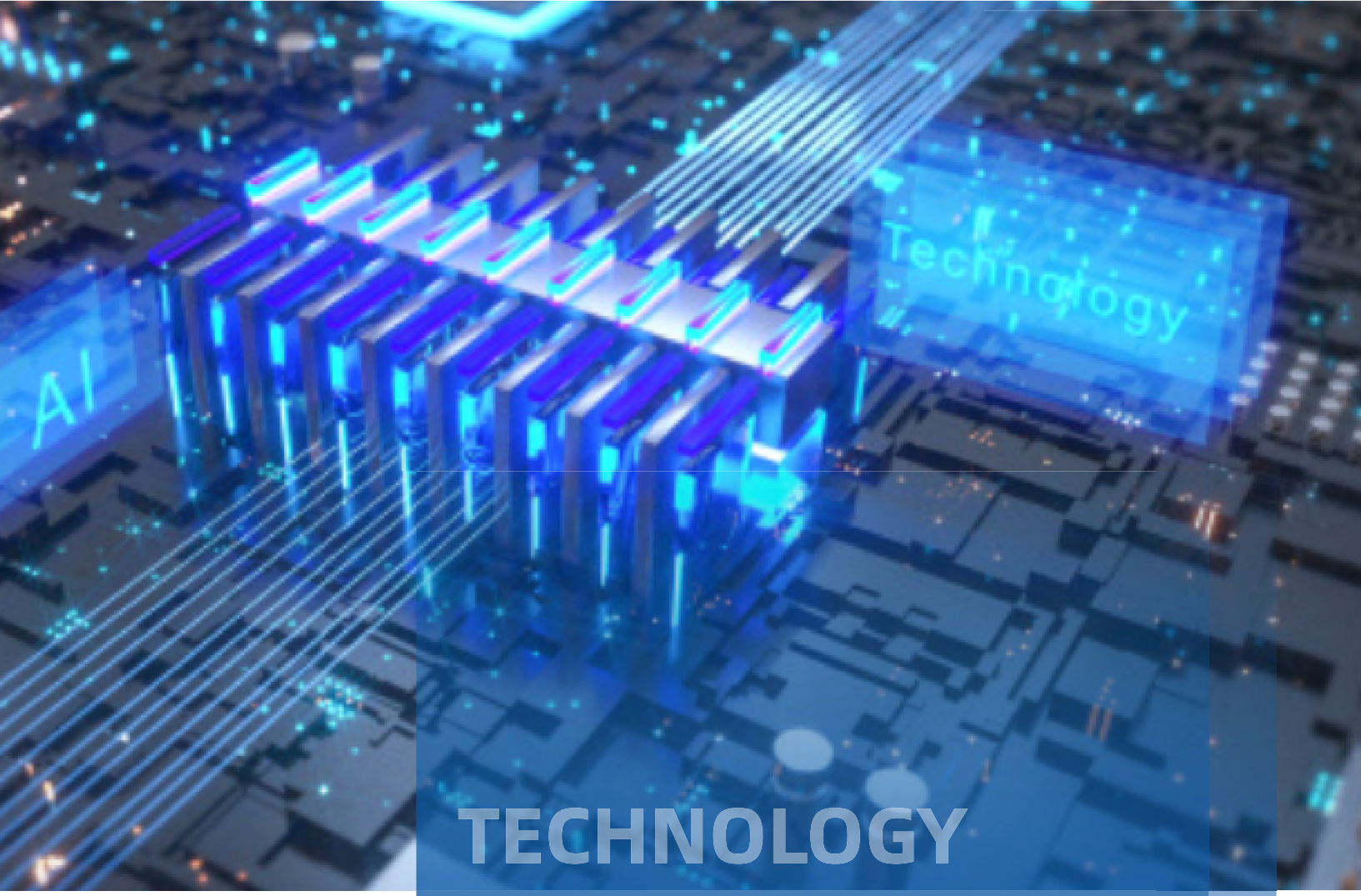
云原生应用要求网络传输更加迅速，类似 DPDK、SROV、Multus 这样的技术得到广泛的应用，其中，DPDK 作用在于加速，它通过类似用户态的网络协议栈，减少了网络对于 NPU 计算资源的中断，从而呈现更好的网络加速的效果。SROV 的作用在于网络流量的隔离，以提供更加安全的网络传输效果。

相比，开源 Multus 提供一个网络端口多接入网络资源的访问。在骨干广域网、城域网普遍采用的 NFV 技术，很多也运行在容器平台环境下。云原生应用与网络传输深度结合。

同样在计算领域，很多 CPU 技术也对云原生应用发挥了重要的作用。以容器隔离为例，因为都运行在 Linux 内核上，如果某些容器应用侵入到内核，势必危害云原生应有的安全运行，这里就需要类似 Intel-VT 这样的技术。针对容器之间存在的资源争夺问题，RDT 这样的技术，也被用于资源的管理和分配。

从本质上说，云原生应用的微服务化，存进了迭代式应用，以及 DevOps 的开发方式，但大量微服务的调用，也增加管理的复杂性，这种复杂性就需要借助 AIOps 技术保驾护航，也因为计算核心内置 AI 加速技术，以及 TPU、GPU 等技术的进步，催生了 AIOps 技术进步，也为云原生应用创造了良好的外部应用环境。

从实际使用者的角度来看，其实随着云原生时代的到来，基础设施在企业中变得越来越不重要、越来越边缘化，因为使用容器的大多是研发的角色，他们更关注怎么用，而不是基础设施运维。因此云原生时代，企业更关注的是业务，而不是基础设施。




TECHNOLOGY

# 五、云原生化应用安全的话题

以容器为核心构建云原生应用安全吗？行业企业 IT 管理和决策人员都有这样的疑虑：“安全吗？”、“我们可以信任容器中的应用程序吗？”

安全的问题要引起高度重视，但也不要因噎废食，不要让对于安全的担忧影响了云原生应用改造和创新的步伐。之所以有这样的结论，因为安全是会伴随信息化系统的全生命周期，始终处于攻防对抗过程中，并没有绝对意义上的安全。

## 5.1 安全风险与挑战

### 5.1.1 镜像风险

**镜像漏洞问题：**由于镜像实际上是静态存档文件，其中包含运行给定应用的所有组件，因此镜像中的组件可能缺乏重大安全更新或已过时。

**镜像配置缺陷：**除了软件缺陷，镜像也可能存在配置缺陷。例如，镜像没有使用特定用户账户进行配置来运行，因而，运行时所需的权限更高。

**嵌入式恶意软件：**由于镜像只是打包在一起的文件集合，其中可能会有意或无意地包含恶意文件。此类恶意软件与镜像中的任何其他组件具有相同能力，因此可以用来攻击环境中的其他容器或主机。

**嵌入式明文秘密钥：**许多应用需要密钥才能实现组件之间的安全通信。嵌入式秘钥的示例包括连接字符串、SSH 私钥和 X.509 私钥。将一个应用被打包成镜像时，这些密钥可以直接嵌入到镜像文件系统中。然而，这种做法造成了安全风险，因为能够访问镜像的任何人都可以轻易对其进行分析，获取密钥。

**镜像不可信:** 在任何环境中，一个最常见的高风险场景是运行不可信软件。由于容器可移植、易于重复使用，更可能诱使团队运行可能无法妥善验证、或不可信的外部来源的镜像。

### 5.1.2 镜像仓库风险

**连接不安全：**镜像中经常包含敏感组件，例如组织机构的专有软件和嵌入式密钥。如果与镜像仓库的连接通过不安全的通道进行，镜像的内容会与用明文传输任何其他数据一样存在同样的保密性风险。还会提高中间者攻击的风险，导致可能会截取用于镜像仓库的网络流量，偷取流量中开发人员或管理员的凭证，给编排工具等提供虚假的或过时的镜像等。

**仓库中的镜像过时：**因为镜像仓库通常是组织机构部署的所有镜像的来源位置，随着时间的推移，这些镜像可能包含许多有漏洞的过期版本。虽然这些有漏洞的镜像并不会因为仅仅存储在镜像仓库中就直接构成威胁，但他们增加了意外部署已知晓含有漏洞版本的可能性。

**认证和授权限制不足：**因为镜像仓库中可能包含用于运行敏感或专有应用以及访问敏感数

据的镜像，认证和授权要求不足可能导致知识产权被盗，或将应用的大量技术细节暴露给攻击者。

### 5.1.3 Kubernetes 安全风险

**管理访问权限不受限制:** 历来，编排工具在设计时都假设，与其交互的所有用户都是管理员，并且管理员应具有对整个环境的控制能力。然而，在许多情况下，单个编排工具可以运行许多不同的应用，每一个由不同的团队管理，并具有不同的敏感性级别。如果提供给用户和不同分组的访问权限不局限于其特定需求，恶意或粗心的用户可能会影响或破坏编排工具管理的其他容器的运行。

**未经授权的访问：**编排工具通常有自己的身份鉴别目录服务，这可能与组织机构内部已经使用的常见目录不同。这可能导致编排工具中帐户管理不到位。容器使用的数据存储卷通常是由编排工具管理的，而且不是主机特定的数据存储卷。由于容器可以在集群中任一给定节点上运行，而该容器内应用所需的数据必须是容器可用的数据，无论它运行在哪台主机上。与此同时，许多组织机构管理的数据在静态下是必须加密的，以防止未经授权的访问。

**容器间网络流量隔离效果差：**在大多数容器化环境中，各个单独节点之间的流量是通过虚拟覆盖网络路由。该覆盖网络通常由编排工具管理，通常对现有的网络安全和管理工具是不透明的。传统的网络过滤器看不到从 Web 服务器容器发送到另一台主机上的数据库容器的数据库查询，只能看到两个主机之间流动的加密数据包，看不到实际容器终端内部的活动，也看不到发送的流量。更严重的是，共享同一虚拟网络的不同应用存在流量风险。如果不同敏感级别的应用，诸如面向公众的网站和内部财务管理应用，使用同一虚拟网络，敏感的内部应用面临网络攻击的风险更大。例如，如果面向公众的网站被入侵，攻击者可能会利用共享网络来攻击财务应用。

**混合不同敏感度级别的工作负载：**编排工具通常重点关注工作负载规模和密度的管理。这意味着，在默认情况下，编排工具可以将不同敏感度级别的工作负载置于同一主机上。例如，在默认配置下，编排工具可能将一个运行面向公众的 Web 服务器的容器与运行处理敏感财务数据的容器置于同一台主机上。若 Web 服务器有严重漏洞，这可能会提高处理敏感财务数据的容器面临的入侵风险。

**编排工具节点受信问题：**编排工具是最基本的节点。编排工具配置不当可能让编排工具和所有其他容器技术组件面临更大风险。维护环境中各节点之间的信任时需要特别小心。

### 5.1.4 容器风险

**运行时软件中的漏洞：**虽然运行时软件中的漏洞比较少见，但是，如果因为漏洞导致“容

器逃逸”等情况的发生，恶意软件就可以攻击其他容器以及主机操作系统本身的资源，这就特别危险了。

**容器的网络访问不受限制：**在大多数容器运行时的默认状态下，各容器都能够通过网络访问其他容器以及主机操作系统。如果容器被入侵并采取恶意行为，那么，允许存在这种网络流量可能会使环境中的其他资源面临危险。由于容器之间大部分是虚拟化连接，因而，在容器化环境下管理对外网络访问更加复杂。因此，从一个容器到另一个容器的数据流在网络中可能只显示为封装的数据包，而没有直接表明其根本来源、目的地或有效载荷。不能感知容器的工具和操作流程无法检查这些该流量，也无法确定其是否存在威胁。

**容器运行时配置不安全：**容器运行时通常会给管理员提供多种配置选项。容器运行时配置不当会降低系统的相对安全性。如果容器在特权模式下运行，则可以访问主机上的所有设备，从而让其本质上成为主机操作系统的一部分，并影响在主机操作系统上运行的所有其他容器。运行时配置不安全的另一个示例是允许容器在主机上装载敏感目录。如果遭到入侵的容器可以更改这些路径，那么，也可以被用来提权并攻击主机本身以及主机上运行的其他容器。

**应用漏洞：**如果容器运行的应用存在缺陷，容器仍可能受到入侵。这不是容器本身的问题，而只是容器环境中典型软件缺陷的表现。当容器遭到入侵时，容器可能会通过多种方式被滥用，例如允许非授权访问敏感信息，或实现对其他容器或主机操作系统的攻击。

**流氓容器：**流氓容器是环境中计划之外或未经批准的容器。这可能是一个常见现象，尤其是在开发环境中，应用开发人员可能会启动容器，以此来测试其代码。如果这些容器没有经过严格的漏洞扫描和适当配置，很有可能更容易被利用。

### 5.1.5 主机操作系统风险

**攻击面大：**每个主机操作系统都有一个攻击面。任何可访问网络的服务都为攻击者提供了潜在的入口点，增大了攻击面。攻击面越大，攻击者就越有可能找到并利用漏洞，从而导致主机操作系统以及在主机操作系统上运行的容器遭到入侵。

**共享内核：**虽然容器提供了功能强大的软件级别的资源隔离功能，但使用共享内核无疑会导致相对于虚拟机管理器甚至容器专用操作系统而言更大的攻击面。换言之，容器运行时提供的隔离级别不像虚拟机管理器所提供的那样高。

**主机操作系统组件漏洞：**所有主机操作系统，甚至是容器专用的操作系统，都提供基本的系统组件。与其他任何软件一样，这些组件也会有漏洞，而且由于这些组件存在于容器技术架构的底层，所以会影响运行在这些主机上的所有容器和应用。

**用户访问权限不当：**由于交互式用户登录应该很少，因此容器专用操作系统通常没有进行优化，以支持多用户场景。当用户不通过编排层，直接登录到主机来管理容器时，组织机构就会面临风险。

**篡改主机操作系统文件系统：**容器配置不安全可能会让主机中文件被篡改的风险增大。例如，如果允许容器在主机操作系统上装载敏感目录，则该容器就可以更改这些目录中的文件。这些更改可能会影响主机及主机上运行的所有其他容器的稳定性和安全性。

### 5.2 安全结论

以上所罗列的安全问题并不完全针对云原生和容器应用，有些问题也是信息系统所共有的问题。云原生应用和安全的话题是相伴相随，既相互影响和促进，也彼此独立平行发展，因为安全问题完全是一个单独学科，需要专业技术人员提供专业性的服务。

还是一句话，要重视云原生安全问题，目前市场有专业化公司提供专业化的服务，但是从应用的角度，也不必因噎废食。

这也是这个白皮书能够给出的建议。

SUMMARY AND OUTLOOK

# 六、总结和展望

结合目前传统行业 / 企业面临互联网巨大冲击的现状，白皮书编委会专家从技术的角度进行了剖析，给出了“云原生技术是关键抓手”的结论，对微服务化、容器化、虚拟化、DevOps、CI/CD、Kubernetes 等技术解决的问题，以及带来业务价值进行深入分析。对 TensorFlow、Jenkins、ElasticSearch、WordPress、Spark、MySQL、Service Mesh、Istio 等技术应用热点进行了介绍，拓宽行业用户的知识和视野。

结合传统行业 / 企业 IT 管理模式和现状，针对“买产品解决方案”，还是“买人（专业人才）”；“买产品方案自力更生”，还是“买服务”的问题进行分析。白皮书指出：云原生应用并不单纯是技术问题，而是涉及企业研发管理的大问题。这样的问题，制造业在引入 ERP 应用的过程中遇到过，类似“上 ERP 是找死，不上 ERP 是等死”的窘境。如今，传统行业 / 企业再一次面临关键的抉择。

作为企业竞争力的核心抓手，云原生应用迫在眉睫，这是传统行业 / 企业领导者必须有的意识，也是本次白皮书编纂的核心观点。花钱能够解决的问题都不是问题，对于云原生应用而言，恰恰就不是一个花钱就能够解决的问题，对此，应该引起高度关注。

对于传统行业 / 企业而言，践行云原生应用不是一个容易的事情，新型合作伙伴关系模型的欠缺也增加了实施的复杂性，尽管如此，也没有必要悲观，因为事物发展就是循环往复不断向前的。

传统行业 / 企业要发展进步，要摆脱目前的被动局面，云原生应用是必须跨越的障碍，市场的发展和崛起仅仅是时间的问题。一万年太久，只争朝夕！

加油！中国的 IT 产业！

APPENDIX

# 七、附录：容器云原生技术产品和服务

## 公司简介

BoCloud 博云，云计算解决方案服务商，以 PaaS 和多云管理技术推动数字化转型。

通过容器云、微服务、DevOps 等组成的 PaaS 技术中台产品体系，为客户提供面向应用管理的解决方案；通过多云管理平台、自动化运维等产品，为客户提供跨越传统 IT 及云计算环境的资源管理解决方案。

博云以产品 + 服务 + 咨询的方式，支撑企业应用版本周级快速迭代、秒级服务扩缩容、业务 99.99% 高可用性保障、服务中台化及微服务化等要求，并实现了多云异构场景下的资源获取管理和应用管理的大规模自助化、自动化、可视化能力，显著降低企业开发运维成本。

BoCloud 博云目前已为包括银行、证券、保险、支付、基金、互联网金融、能源、政务、电信、先进制造、航空、交通、零售、媒体、软件开发、医疗等十多个领域的上百家大型企业客户提供了专业的云计算解决方案服务，支撑着众多客户的生产系统稳定高效运行，为客户的数字化转型保驾护航。博云的典型客户包括中国农业银行、民生银行、华泰证券、东方证券、江苏银行、新华社、中国航信、江苏电力、南方电网、中国移动、中国联通、中国石油、京东数科、安邦保险、中国人寿保险、吉利汽车、海信集团等众多大型标杆性企业。

## 产品矩阵


BoCloud 博云产品矩阵

### 面向应用管理——服务企业应用，助力业务创新

**PaaS** 技术中台

博云对中台的定义是“企业级能力复用平台”，所以技术中台的定义就是“企业级的技术能力复用平台”。通过中台化的建设，可以让后台服务更简单更稳定、让前台服务更灵活、让中间层服务的建设统一化从而避免重复投资建设以及技术资产的积累。博云的技术中台解决方案，抽象出了 IT 资源服务、产品研发管理服务、中间件技术服务、产品运营服务等方面的服务维度，最终在“技术能力中台化”的过程中利用平台化手段发现、沉淀与复用企业级 IT 能力。

**BeyondContainer** 容器云

BeyondContainer 是博云基于容器技术的企业级 PaaS 解决方案，利用微服务思想和 DevOps 理念，基于 Docker 和 Kubernetes 提供对应用开发态、部署态、运行态的应用全生命周期管理能力，利用可视化、可配置、自动化持续交付流水线结合 DevOps 咨询，帮助企业 DevOps 落地，通过容器化和服务治理对微服务架构业务落地进行支撑，帮助企业实现应用云化、能力平台化、管理互联网化。

**BeyondMicroService** 微服务治理

BeyondMicroService 微服务治理平台集成多种微服务架构和运行组件，以其庞大的兼容性，平滑接入客户已有的微服务应用；并借助于对微服务技术的熟悉和经验，为客户实现微服务转型、应用开发、微服务框架落地提供解决方案。

**BeyondDevOps** 平台

BeyondDevOps 提供从“需求 -> 开发 -> 测试 -> 发布 -> 运维 -> 运营”端到端的开发运营一体化平台解决方案，覆盖项目管理、研发管理、测试管理和运行管理的协同服务和研发工具支撑，将线下 IT 生产过程转变为线上高度自动化、可视化的 IT 生产线，提升产品研发效率，快速响应业务需求，保障工作质量，并通过度量分析、风险预判，持续提升 IT 运营能力。

**面向资源管理——提高 IT 资源效率，解决全新运营问题**

**BeyondCMP** 一体化云管理平台

BeyondCMP 一体化云管理平台为企业提供云和非云资源的统一纳管，帮助客户实现多云资源的调配和管理。通过对企业云架构系统中人员、事件、云计算中的基础设施资源、数据资源、业务资源的统一管理，从而提供对企业部门之间资源统一协调和整体运筹的一体化解决方案。对企业未来云架构系统的运维、管理和可持续性等方面实现一体化管理，帮助企业云架构系统持续优化。

**BeyondBSM** 自动化运维

BeyondBSM 自动化运维平台由博云和京东云联合开发，依托京东云丰富互联网运维经验和京东云翼自动化运维平台，为运维的自动化操作提供一体化解决方案，助力企业提升运维效率，降低运维成本，规范化、标准化运维工作，支撑企业业务快速稳定发展。

**BeyondCube** 超融合

BeyondCube 是由博云自主研发的超融合云计算平台。以高性能的分布式为基础，将计算、存储、网络和管理融合到 标准 x86 服务器中，实现基础设施以积木式部署和横向扩展。数据中心所有的关键功能都以软件定义的形式实现。 从而帮助用户以最好的性价比快速构建安全、高效、可靠的新一代云计算数据中心。

# Dell Technologies Cloud Platform 戴尔科技云平台

## 对传统和云原生应用程序的企业级支持

作为一种最新的软件架构，云原生正随着 Kubernetes 的迅速普及而得到了广泛的推广，但很多用户在落地云原生的过程中，依然面对着两大挑战，主要体现在：

一方面，大部分企业过去的基础架构管理与应用管理是脱节的，无法提供一致性和一站式的管理能力；另一方面，对于企业来说，随着越来越多的业务操作和应用程序扩展到多云混合状态，就要求基础设施架构能力必须不断发展，以支持任何地方的创新，同时提供无缝的开发、部署和持续的管理经验。

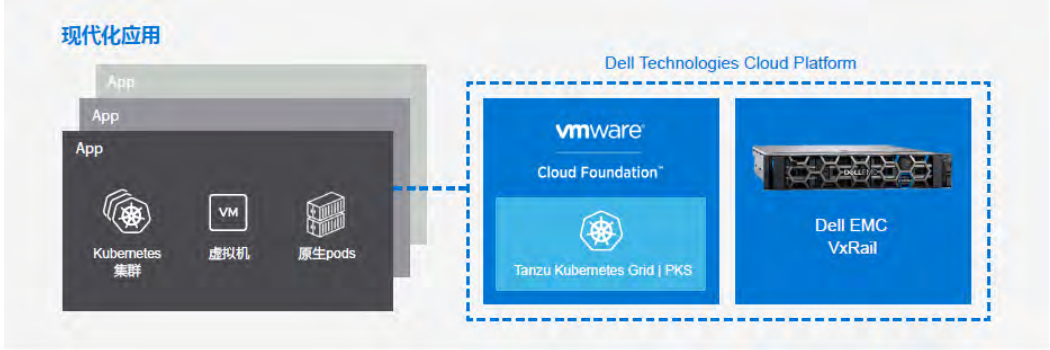
正是洞察到这一新的市场需求和变化，戴尔科技推出了云原生平台，这是一款软硬件一体交付的工程化解决方案，它把全新的云原生能力以端到端交付的新模式提供给企业客户，是一个真正的混合云就绪的企业级云原生平台，同时也是企业进行数字化转型和实现云原生之旅的理想选择，具体来看：


首先，戴尔科技云原生平台的软件核心是 VMware Tanzu，VMware Tanzu 是 VMware 最新推出的云原生系列产品家族，可以帮助企业管理现代化应用以及将现代化应用交付到任何云环境中。

目前，通过 Project Pacific，可引入 VMware Tanzu 的功能和服务，企业用户就可以实现容器和虚拟机的融合，将 VMware vSphere 转换为 Kubernetes 原生平台，最大化的帮助客户释放虚拟机的潜力。更为重要的是，还可以借助 Tanzu Kubernetes Grid 和 VMware PKS 高效地部署、运行和管理 Kubernetes，快速获得生产力，并为传统和云原生工作负载提供一致的集成式使用体验。

由此带来的价值是，IT 运维与开发者是在一个统一的平台上管理各自的工作，企业可以在戴尔科技提供的一套统一平台下，把虚拟机资源与云原生应用完成对接和实现统一管理，最终让虚拟机和容器完美的“融合”在一起，并彻底打通 IaaS 和 PaaS。

其次，戴尔科技云原生平台的现代化应用及数据的底座 , 则是 VMware Cloud Foundation on VxRail（简称 VCF on VxRail），这是一套建立在 VxRail 上的业界领先的超融合软硬件一体化的基础架构解决方案。



VCF on VxRail 由 VMware vSphere、vSAN、NSX 与 Dell EMC VxRail 超融合基础架构紧密集成，具有从计算、存储、网络虚拟化到云计算管理平台的所有功能，同时还结合了公有云的简易性、灵活性和经济性，以及与本地基础架构的安全性、控制力和超强性能。

VCF on VxRail 的优势主要在于，企业可以根据自己业务的需要来进行灵活选择，由于基础架构的一致性，企业现有的工作负载无需任何修改，就可以灵活地在私有云和公有云之间、多个公有云之间进行迁移；同时，这种负载迁移同时也是双向的，既可以从私有云到公有云，也可以从公有云到私有云，或者是从一种公有云到另一种公有云，给用户提供最大的便捷性。

随着 2020 年 4 月 VMware vSphere 7 和 VMware Cloud Foundation 4（VCF 4.0）的发布，Dell EMC VxRail 也同步升级到 7.0 版本，其升级重点包括三个方面：其一，VxRail 7 和 Vsphere 7 同时支持传统虚拟机和云原生应用；其二，戴尔科技集团于 2020 年 5 月 20 日发布，VCF 4.0 Consolidated Architecture 可以部署在 VxRail 的 4 个节点上，有效减少客户的初期投入成本；其三，VCF 4.0 和 VxRail 7 进一步的深度集成，也实现了从 VxRail 硬件到 VCF 4.0 软件的全自动化安装、配置、升级，极大降低 IT 架构的运维难度，同时更快速地支持云原生应用的开发和维护。


戴尔科技云原生平台在落地的过程中，戴尔科技还提供了包括战略研讨，规划设计，实施交付以及持续运维四个步骤的服务方式，帮助企业落地云原生系列解决方案。

在此过程中，戴尔科技通过和企业相关部门人员一起召开的战略研讨会先摸清企业 IT 的基本情况，并了解利益相关方的诉求，最终达成企业的 IT 现状和未来的云原生及微服务平台如何发展的策略共识，接下来通过规划设计，包含整合企业原有的多云基础架构、多云服务管理，云业务创新模式的演进路线，并提供应用的现代化架构规划及应用迁移评估帮助企业实现微服务化的应用转型。

不仅如此，戴尔科技也为企业的应用和数据提供容灾及业务连续性解决方案，满足等保 2.0 的要求，通过对运维现状的评估，提出运维优化方案，如让企业实施更加满足云原生应用的 DevOps 体系，真正帮助企业从容构建一个敏捷的云原生平台，为企业的持续创新提供源源不断的新动力。

值得一提的是，戴尔科技云原生平台还即将提供全新的“云消费”模式，即通过支持订购模式的 Dell Technologies Cloud 平台，就可在短短 14 天内部署出企业所需要的混合云架构。而且，还可借助 Dell Technologies ON Demand（DTOD）提供基于消费和服务的交付模型，为企业提供更大的选择空间、灵活性和可预测的结果。

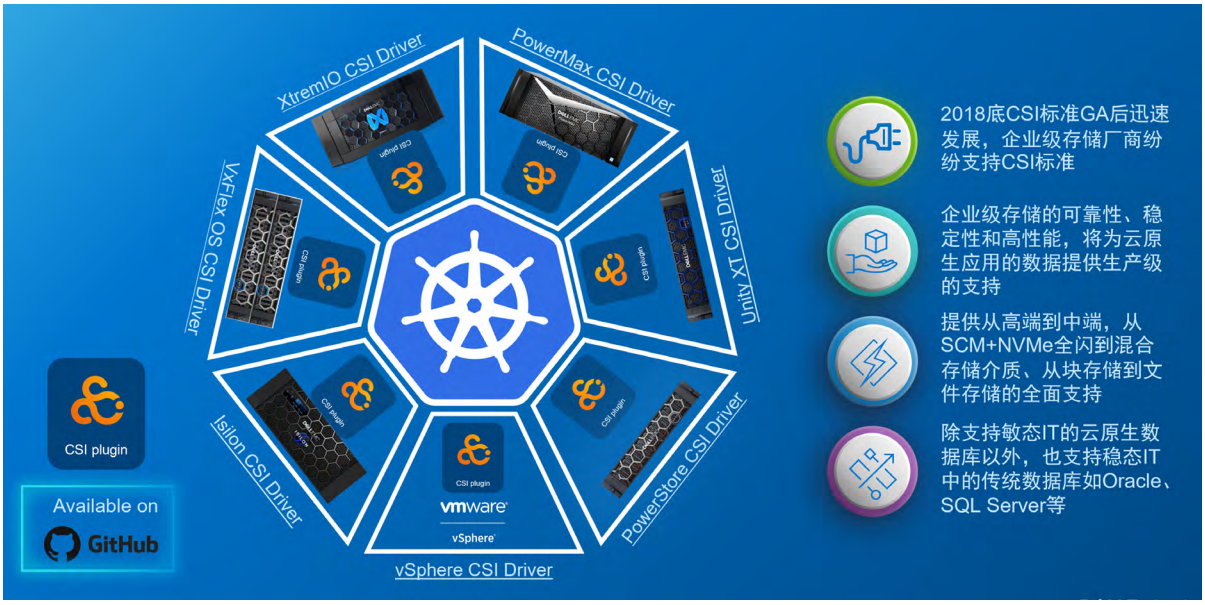
由此可见，戴尔科技云原生平台通过软硬件一体交付的工程化解决方案，以及创新的端到端交付新模式，真正为企业提供了一个包括软件和硬件层，且包容传统应用和云原生应用的各

方面需求的一体化平台，不仅为跨多云的技术融合和业务升级提供了支撑，保证各种核心应用业务的稳定安全性、企业级一致性和业务合规；同时，更打破原有硬件环境相互间的桎梏，更联通私有云、公有云、混合云的边界，保证了企业数字化转型需求，以及云原生升级改造所需的业务升级能力。

### 戴尔科技提供支持容器化应用的存储

软件架构正迅速从紧密耦合的单体应用向微服务的方式转变，通过基于微服务的这些模块，可以构建小型的、自包含的和独立的功能部件，并且能以更好的敏捷性、更大的规模和更高的可用性来进行部署。微服务部署为一个或多个容器。容器是一种包含微服务及其依赖的轻量级包，运行这些服务不需要完整的操作系统镜像。

作为对容器化工作负载的共享存储进行供应和管理的事实标准，容器存储接口（CSI）正在迅速发展。Dell EMC 业界领先的 PowerMax、PowerStore、XtremIO、Unity/Unity XT 、VxFlex OS 和 Isilon 存储平台都有 CSI 插件来支持在 Kubernetes 上运行的容器化工作负载，提供从高端到中端、从 SCM+NVMe 全闪到混合存储介质、从块存储到文件存储、从集中式存储到分布式存储的全面 CSI 支持。




CSI 插件是 Kubernetes 中的逻辑卷（PV）与存储中的卷或 LUN 之间的接口。存储类（Storage Class）为底层存储阵列的不同特性指定一组唯一的参数。在静态资源调配工作流中，存储管理员可以使用 Kubernetes 群集中可用的特定存储类（Storage Class）来创建 PV。在动态资源调配中，pod 配置文件可以通过指定特定存储类（Storage Class）的卷声明（PVC）来创建 PV。

### VMWare 容器存储支持的解决方案

• Container Storage Interface (CSI) for vSphere

Kubernetes 集群通过 CSI 插件使用底层基础设施的存储资源， vSphere CSI 是一个 out-of-tree 插件，它将 vSphere 存储公开给容器编排器（如 Kubernetes）上的容器化工作负载。该插件支持 vSAN 和其他类型的 vSphere 存储。vSphere CSI 与 vCenter 服务器上的 CNS 控制平面通信，以执行所有存储资源调配操作。

• Cloud Native Storage 控制平面

CNS（Cloud Native Storage）控制平面位于 vCenter 服务器中。它是 vCenter 服务器管理的扩展，实现了容器卷的供应和生命周期操作。在供应容器卷时，它与一级磁盘（FCD）功能交互，以创建支持卷的存储对象。此外，CNS 控制平面与基于存储策略的管理（Storage Policy Based Management - SPBM）部件进行通信，以确保对磁盘提供所需的服务级别。CNS 还执行查询操作，允许您通过 vCenter Server 管理和监视容器卷及其备份存储对象。

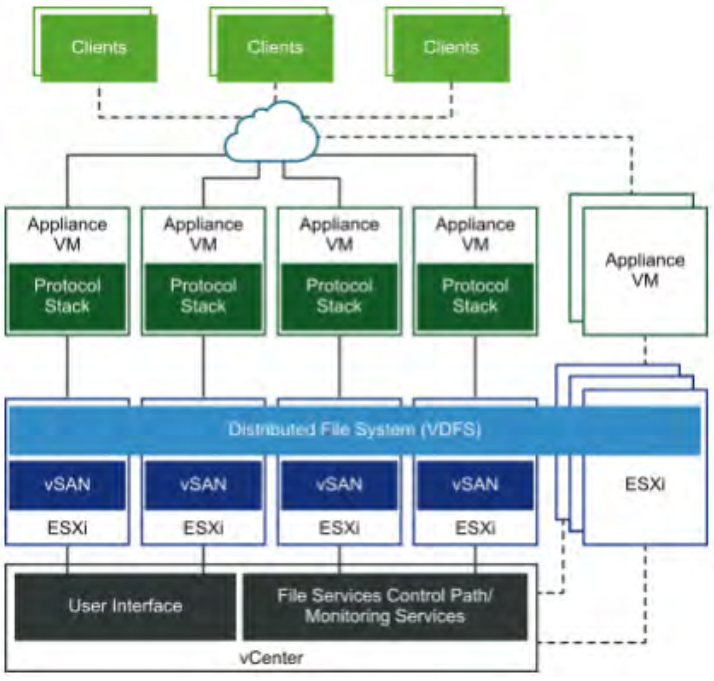
• First Class Disk (FCD)

也称为改进的虚拟磁盘。它是一个与虚拟机（VM）无关的虚拟磁盘。这些磁盘位于 VMFS、NFS 或 vSAN datastore 上，支持访问模式为 ReadWriteOnce 的容器卷。FCD 执行与 PV 相关的生命周期操作，FCD 的生命周期在 VM 或 Pod 生命周期之外。如果 VM 是一个 Kubernetes 节点，在它上面运行了多个基于容器的应用程序，并且这些应用程序使用了 PV 和虚拟磁盘，那么 CNS 有助于在容器和 PV 粒度上执行生命周期的操作。

• vSAN File Service

vSAN File Service 提供了直接在 vSAN 集群上承载 NFS 共享的功能。使用 vSAN File Service 可以在 vSAN 的 datastore 中创建文件共享，客户机工作站或 VM 可以访问该文件共享。存储在文件共享中的数据可以从任何具有访问权限的设备进行访问。

vSAN File Service 是位于 vSAN 之上提供文件共享的层。它目前支持 NFSv3 和 NFSv4 文件共享。vSAN File Service 由 vSAN 分布式文件系统（vDFS）组成，vDFS 通过聚合 vSAN 对象，提供了底层的可扩展文件系统和存储服务平台，该平台提供弹性文件服务器端点以及用于部署、管理和监视的控制平面。文件共享被集成到现有的基于 vSAN 存储策略的管理（SPBM）中，并且策略管理是基于每个共享的。



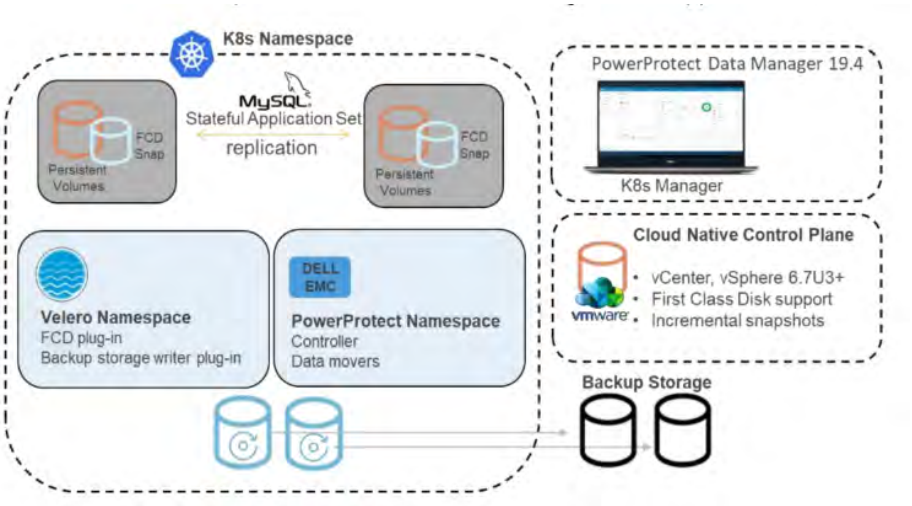
• Storage Policy Based Management (SPBM)

基于存储策略的管理是 vCenter Server 的一个服务，它支持根据指定的存储要求配置 PV。存储卷被供应后，该服务将监视供应的卷是否符合所需的策略特征。

### 戴尔科技提供对 Kubernetes 应用负载的数据保护

根据 Gartner 的调查报告显示，到 2022 年，超过 75% 的全球组织将在生产环境中运行容器化应用程序。考虑到数据丢失和对数据的跟踪能力仍然是使用容器和 Kubernetes 的主要关注点，如果组织想要释放云原生技术的收益，就不能把数据保护当作事后考虑。Dell EMC PowerProtect Data Manager 是第一个也是唯一一个同时针对虚拟机、应用程序和 Kubernetes 的企业级保护解决方案。PowerProtect Data Manager 支持在 Kubernetes 环境中发现、保护和管理生产工作负载。通过对生产和开发 / 测试工作负载的保护，以确保数据易于备份和恢复。PowerProtect Data Manager 与 VMware 合作，利用为 Kubernetes 环境开发的 Project Velero 项目中的 kubernetes 原生体系结构，将其直接集成到用户界面中。数据和应用程序所有者获得的好处是有一个直观、易于使用的用户界面（UI），同时 IT 运维人员可以独立于开发进行集中治理。

最近，VMware Tanzu 的产品和服务组合改变了虚拟化环境管理员在 Kubernetes 上构建应用程序的方式。作为此策略的一部分，VMware 发布了 vSphere 7.0，其中包含原生 Kubernetes 功能和 VMware Cloud Foundation 4.0。与此相结合，Dell Technologies 在 VMware 云原生控制平面中实现了 Tanzu Kubernetes Grid（TKG）的数据保护功能。Project Velero，作为 VMware 和 Dell EMC 创新合作伙伴关系的一部分成果出现。Velero 被设计为一个基本工具，可以与 Tanzu 一起工作以安全备份、恢复和迁移 Kubernetes 群集资源和 PV。Dell Technologies 基于 Velero 的控制器功能构建，并在 PowerProtect Data Manager 中提供企业级功能。



PowerProtect Data Manager 的用户可以通过直接将受保护的数据放入 PowerProtect DD 或 Data Domain 来增强保护，并从具有无与伦比的效率、重复数据消除、性能和可扩展性的第二存储中获得收益。Data Domain 系统可在系统运行时无缝扩展，只需在运行中添加额外的托架，而无需中断操作。巨大的可扩展性意味着组织需要管理的设备更少，需要的基础设施更少，实现更高重复数据消除率，因为 Data Domain 具有全局重复数据消除的能力。Data Domain 高效的在线可变长重复数据消除技术成为经济高效的“无磁带”灾难恢复的支持技术。Data Domain 系统仅将唯一数据复制到远程站点，并在备份仍在进行时就开始复制。在 PowerProtect Data Manager 和 PowerProtect DD 或 Data Domain 上进行投资的组织将随着时间的推移而从经济效益、性能的提高以及最重要的是从保护成本的降低中受益。

### 重定义云原生之旅

总的来说，戴尔科技云原生平台加速重新定义了企业的云原生之旅，解决了用户在云原生基础架构方面的痛点，也构建了面向未来的混合多云管理新模式，其价值远不止于眼下，更在于未来。

第一，戴尔科技云原生平台依托 VMware Tanzu 带来的强大能力，可以从容帮助企业实现云原生的升级和改造，尤其是更大的传统用户无需在虚拟机和 Kubernetes 容器环境之间做出选择，从而能自由在戴尔科技云原生平台上进行现代应用程序开发和运营，同时继续利用现有的技术、工具和技能组合投资，最大化降低云原生转型之路上的风险和成本。

第二，让企业大大减轻了日常基础架构运营带来的种种“制约”。换句话说，未来企业的服务模式将都转变为云的模式，而且和应用是深度融合的，因此通过戴尔科技云原生平台赋能企业基础架构的运营，就可以让企业无需关注底层基础架构带来的难题，并构建起以应用为中心管理新模式，推动业务价值的提升。

第三，是帮助企业实现了混合多云环境下管理的统一。企业在未来无论是采用私有云、混合云、还是公有云，这些不同的云都是企业建设资源、管理资源的一种方式和选择。对企业来说，通过戴尔科技云原生平台提供的统一的用户体验，并在安全可控的基础上“为我所用”，最终为企业实现资源优化和降本增效。

### 关于戴尔科技集团

戴尔科技集团是一个独特的企业集团，致力于为企业、组织及个人构建数字化未来，实现工作和生活方式转型。公司为客户提供业界最广泛和创新的科技和服务组合，覆盖从边缘计算到核心数据中心到云。戴尔科技集团包括 Dell, Dell EMC, Secureworks, Virtustream 和 VMware 等。

# 联想凌拓容器存储解决方案 Trident：用于容器的动态永久存储流程编排程序

**联想凌拓容器存储解决方案 Trident 主要优势：**

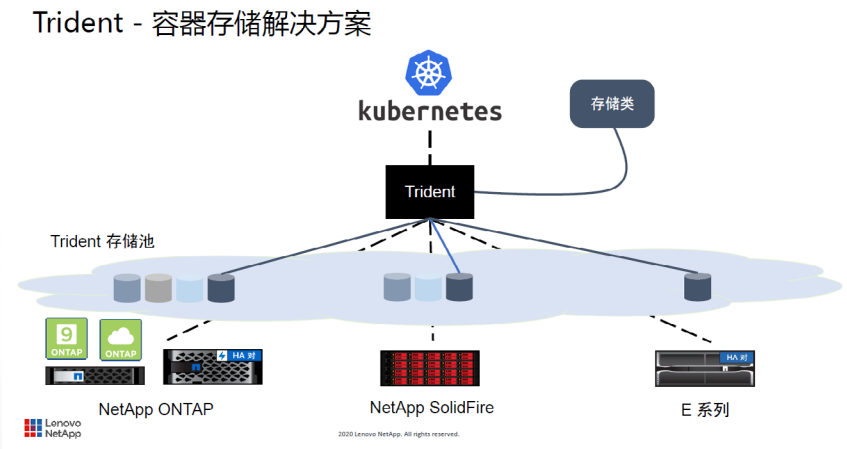
- 为保留时间超过容器生命周期的数据编排永久存储
- Trident 会自动将对永久存储的请求分配给可满足所需服务类别的联想凌拓存储目标
- 提高数据一致性，即使容器生命周期不可预测也不例外
- 轻松交付灵活且安全的永久存储

对于加快云原生应用程序的开发和交付速度以及推进现有应用程序的现代化而言，应用程序容器正变得越来越重要。单个容器可能是短暂、无状态的，但是它们在生命周期中使用或生成的数据呢？这些应用程序通常具有永久数据需求，企业在部署容器框架时必须考虑这些需求。虽然容器流程编排程序已执行一些基本步骤，帮助自动配置实现永久性所需的存储，但是开发人员和基础架构团队仍需执行一个耗时的手动流程。

**联想凌拓提供的容器存储解决方案 Trident** 作为外部配置程序控制器实施，本身以 POD 形式运行，可监控卷并全面实现配置流程自动化。此外，通过引入 Kubernetes 存储类（Kubernetes StorageClasses），降低了支持用于容器的永久存储的复杂性。

**一些常见使用情形包括：**

- 希望加快持续集成 / 持续交付 (CI/CD) 流程的开发运营团队
- 对内部部署的现有企业级应用程序进行现代化改造（升级和转变）或将它们迁移到云环境
- 云原生应用程序和微服务




## 利用强大的功能提高 DevOps 效率 >>

除了基本的永久卷集成以外，Trident 还支持内置于联想凌拓提供的 NetApp 产品组合中的

固有高级数据管理功能，所有这些功能都旨在帮助企业为容器化应用程序灵活部署存储。企业可获得以下优势：

• 涵盖联想凌拓提供的完整 NetApp 产品组合的支持服务， 其中包括 NetApp HCI、NetApp Cloud Volumes、NetApp ONTAP、NetApp SolidFire 和 NetApp E 系列。

• 由企业级存储与数据管理服务管理和保护的应用程序数据

• 能够同时使用多个存储后端。每个后端均可采用不同的配置进行部署，因而 Trident 能够配置和使用具有不同特性和成本的存储。Trident 还可以通过一种简单直接的方式向容器化工作负载提供可组装的基础架构，不会增加复杂性。

### 支持容器生态系统 >>

不断增长的应用程序容器生态系统可以提高可用性并扩大部署范围。Trident 可以支持许多常用应用程序容器平台和流程编排程序，例如 Kubernetes、Red Hat OpenShift、Docker Enterprise Edition 及 Rancher。

### 释放速度和灵活性 >>

容器是多云工作负载的未来发展方向，将应用程序从底层操作系统抽象出来，在软件开发中实现了可移植性和灵活性，并提高了开发团队和基础架构团队的效率。Trident 可帮助企业不断前行，全面迎接云原生未来。无论哪种使用情形或工作负载，均可利用 Trident 让软件生命周期变得更快、更灵活。

## >> 联想凌拓容器存储备份解决方案 <<

Docker 容器正成为开发和部署应用程序的一种有效方式。Gartner 预计，2018 年，至少在应用程序生命周期的一个阶段，超过 50% 的新工作负载将部署到容器中。多年前，大多数容器都被设计为暂时的或短暂的，因为容器化应用程序没有创建或修改以后需要的数据。

然而，随着容器越来越流行，越来越多的容器化应用程序被设计来创建和修改持久数据。容器使用的这种演变导致了数据完整性方面的问题，并突出了在生产环境中支持这些容器化应用程序的成熟备份和恢复过程的需要。是否需要备份所有的容器，这取决于容器中的应用程序是如何设计的，以及它们是否创建或修改持久数据。

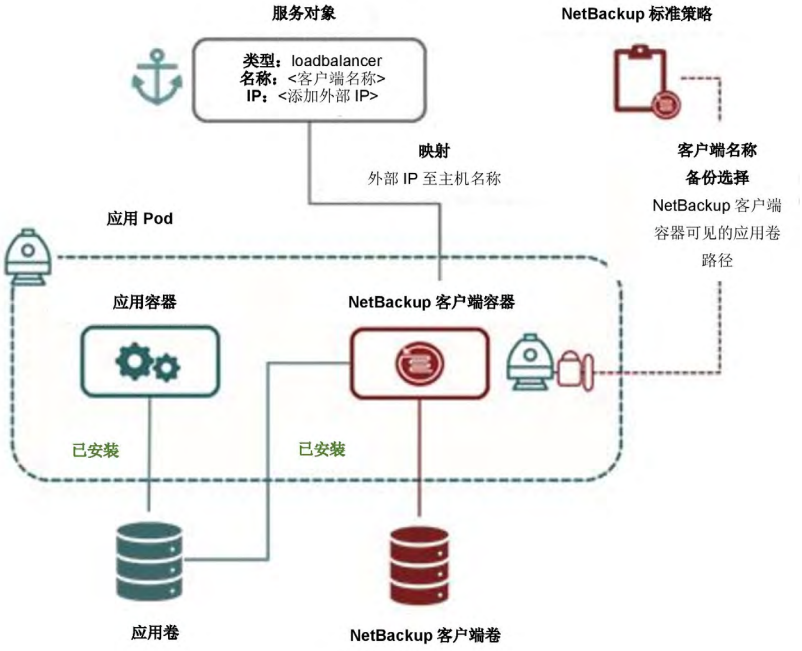
企业需要与开发人员讨论此主题，以了解哪些容器可能需要保护、哪些容器可以安全地从备份过程中排除。有了经过全面测试和认证的 Docker 容器数据保护解决方案，开发人员就可以自由探索使用持久数据实现容器的新方法。

为了最大限度的灵活性，联想凌拓容器存储备份解决方案提供了三种不同的方式来保护容器，通过使用 Kubernetes 容器编排，联想凌拓 NetBackup 容器客户端可以：

• 在单个 Kubernetes pod 中与应用程序容器一起自动部署，也可以独立部署

• 用于保护共享或转储（shared or dump）卷

备份方式：在应用程序 Pod 中部署 NetBackup Client Container




同时，也有 Kubernetes 用户希望环境中运行的应用可以实现崩溃持续备份，备份元数据和连接到 Kubernetes 中运行的 pod 的持久化卷，通过简单的管理界面快速设置保护并在多个不同的目标环境中存储 Kubernetes 备份。

联想凌拓可以做到（by vProtect）：

- Kubernetes 环境中运行的部署的应用崩溃一致性备份
- 备份连接到 Kubernetes pods 的元数据和持久化卷
- 自动暂停运行中的部署，以实现 Kubernetes 一致的备份
- 排除特定持久性卷的选项
- 在还原过程中，从备份的元数据中重新创建应用程序
- 重新命名恢复的应用程序
- 支持将数据导出到现有的备份环境中进行整合和长期存储。可与其他备份软件协同工作
- Kubernetes 备份到 Amazon S3、Google 云存储（Nearline、Coldline）、Microsoft Azure、OpenStack Swift、Neverfail HybriStor、ExaGrid、IBM Object Cloud Storage、Ceph 或 Data Domain Boost 等的 Kubernetes 备份
- Kubernetes 备份到任何安装的文件系统（本地、NFS、iSCSI、SMB）
- Kubernetes 备份到 Red Hat 虚拟数据优化器 (VDO)，其中包括重复数据删除和压缩
- 用于第三方软件集成（REST API）以及命令行的开放 API
- UI - 支持中文界面

欲了解联想凌拓容器解决方案详细信息，请拨打热线 400-116-0099 咨询。欢迎访问公司官网 http://www.lenovonetapp.com/ 及关注公司官方微信，获取更多联想凌拓产品及解决方案资讯。

# KubeSphere 容器平台，助力企业一步跨入云原生时代

KubeSphere 是一款基于 Kubernetes 构建的企业级容器平台，具有强大且完善的网络与存储能力，并通过极简的人机交互提供完善的多集群管理、CI / CD 、微服务治理、应用管理等功能，帮助企业在云、虚拟化及物理机等异构基础设施上快速构建、部署及运维容器架构，实现应用的敏捷开发与全生命周期管理。


### 一步跨入云原生应用时代 >>

• 即点即用 DevOps 功能：可视化程度更高的 DevOps 平台，拖拽编辑 CI / CD 流水线，内置流水线模版，DevOps 功能即点即用。

• 完善的微服务治理：深度整合 Istio 与 SpringCloud 微服务框架，支持 Java, Python, Go 等多种开发语言，同时，可视化界面支持灰度发布、熔断等完善的治理功能。

• 应用跨平台一键部署：一键部署应用到任意基础设施平台上，并可实施统一的运维操作，从而实现应用标准化管理，大幅降低应用运维复杂度。

### 健壮的容器基础设施 >>

• 商业验证的 SDN 能力：可通过 QingCloud CNI 插件对接 QingCloud SDN，获得更安全、更高性能的网络支持。

• 商业验证的 SDS 能力：无缝对接 QingStor NeonSAN ，数百万 IOPS 时延亚毫秒之内，快速响应核心业务需求。

• 全栈容器云：可与 QingCloud 云平台无缝对接，调度 QingCloud 全栈 IaaS 能力。

### 企业级特性增强 >>

• 极简：向导式图形化的 UI 全方位覆盖调度、管理、运维监控等功能，低学习成本高效使用。

• 安全：基于微服务级别细粒度的多租户权限管理，完美实现资源隔离，保障数据安全性。

• 运维友好：可视化、自动化的统一运维，以及全方位、立体化的秒级频率监控，极大程度降低运维复杂度。

• 兼容企业传统 IT Legacy：尊重企业 IT 管理规范，兼容企业既有 IT 管理流程，可平滑整合到现有 IT 体系中。

• 商用企业级技术沉淀：源自青云 QingCloud 技术领先的产品团队，通过企业用户大规模生产验证。

### 典型案例——本来生活网借助 KubeSphere 零门槛打造 DevOps 极致体验

本来生活网虽然是一家互联网电商公司，但很早就停止了烧钱模式，追求盈利，对 IT 建设也提出了尽量平衡成本、开源节流的要求。所以，本来生活网迫切需要重构基础设施，建设一套更为灵活、更为敏捷的 IT 架构，以优化开发运维流程，最大程度提高应用开发效率并降低 IT 生产环境运维成本。

此外，本来生活的应用发布由测试团队完成，但测试人员缺乏一定的开发运维经验，无法快速上手 Kubernetes 实现版本快速迭代。想要打通开发、测试与运维的 DevOps 一体化流程，需要一个统一的平台配合应用开发和上线发布的整套流程。本来生活网通过大量调研，最终选择了 KubeSphere。

#### 解决方案 >>

本来生活网通过 KubeSphere，逐步把生产环境从虚拟化迁移到 Kubernetes 之上。基于 KubeSphere 向导式的交互，让测试团队在还不熟悉 Kubernetes 的情况下，也能对应用进行持续发布。KubeSphere 自动采集应用与基础设施的日志，可以方便测试团队进一步调试，从而实现统一的 DevOps 管理。

物理机上部署的 Kubernetes 集群不提供类似 LoadBalancer 服务暴露的功能。为此，本来生活网选择了 KubeSphere 的子项目 —— Porter（一款适用于物理机部署 Kubernetes 的负载均衡器，提供用户在物理环境暴露服务和在云上暴露服务一致性体验的插件。），作为在物理环境下暴露 Kubernetes 服务的解决方案。

#### 用户证言 >>

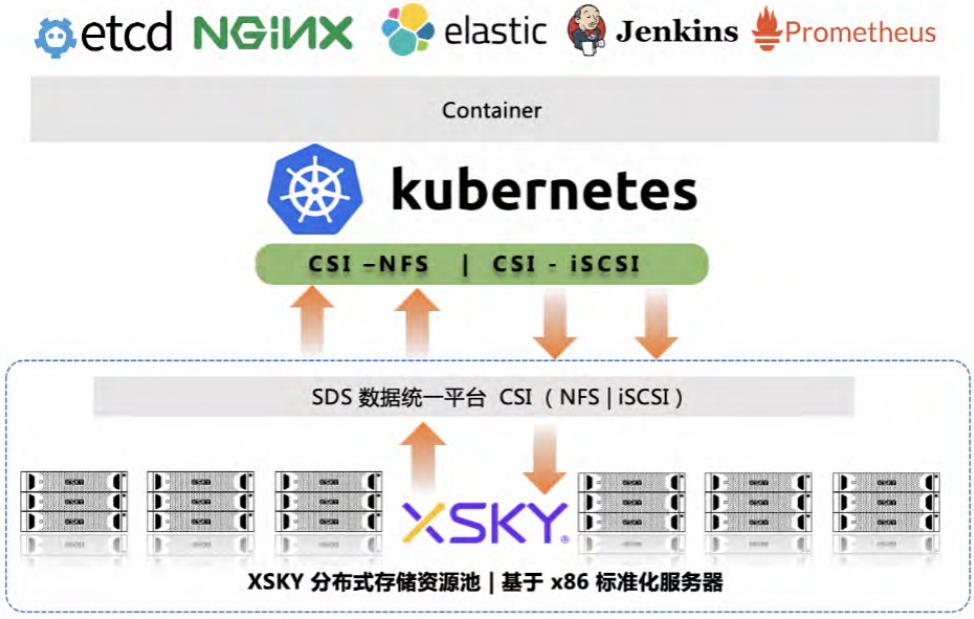
KubeSphere 有良好的用户体验，能提升容器部署的便捷性，测试效率的提升也非常明显。同时，KubeSphere 还极大降低了运维团队的工作复杂度，缩短了应用从开发到上线发布的时间。本来生活网不需要再花费大量时间去开发一个可视化平台，并且还能实现自定义的 CI/CD 流程，这对于技术团队来说至关重要。

**申请商用支持**
https://kubesphere.qingcloud.com/

**访问开源社区**
https://kubesphere.io

# XSKY CSI 容器存储解决方案，助力企业云原生转型

XSKY 是国内率先通过 K8s CSI 认证的 SDS 厂商，同时也是加入 CNCF Landscape 全景图的 SDS 厂商之一。XSKY 基于 CSI 的容器存储方案架构如下：




XSKY CSI 在容器存储场景有如下优势：

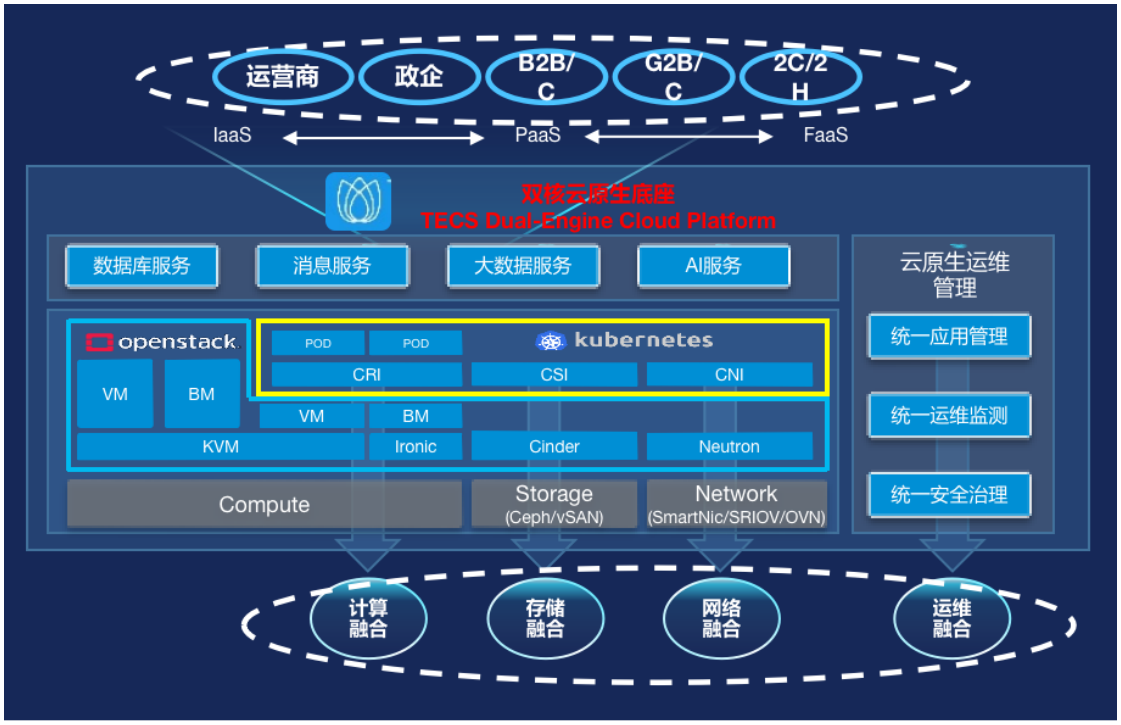
- 标准接口：实现标准的 K8s CSI 接口，无缝支持 K8s 生态
- 统一存储：支持 iSCSI 和 NFS 两种接口，实现块和文件的统一存储
- 存算分离：将计算和存储分离，实现资源隔离，可独立对存储集群进行扩容
- 稳定可靠：基于数千个企业级生产案例的最佳实践，XSKY 分布式存储为容器平台提供稳定可靠的保障
- 性能卓越：可根据应用需要提供全 SSD、混合盘以及全 HDD 等存储资源，通过 XSKY SDS IO 全路径的性能优化能力，获得卓越性能
- 特性丰富：除了支持裸设备、卷扩容、卷克隆、卷快照和 QoS 等基础特性，也支持延展集群和 MPIO 等高级特性

除了 CSI 插件外，XSKY 同时支持利用 K8s CRD 实现 XSKY SDS 集群的快速编排和部署，帮助您将存储资源纳入到现有的 K8s 编排系统中，从而实现对计算和存储等资源的统一管理。

欲了解 XSKY CSI 容器存储解决方案详细信息，请拨打热线 400-016-6101 咨询。欢迎访问公司官网 https://www.xsky.com 及关注公司官方微信，获取更多 XSKY 产品及解决方案资讯。

# 中兴通讯 TECS 双核云原生底座，助力 5G 行业创新

中兴通讯“TECS 双核云原生底座”提供给运营商统一的边缘云管理和服务视图，同时通过融合 OpenStack 和 Kubernetes 为边缘侧平滑引入丰富的云原生技术栈，辅助边缘应用更快速的迭代和获得一致的边云体验。




## 全栈能力 Full Stack >>

中兴通讯“TECS 双核云原生底座”提供给运营商统一的边缘云管理和服务视图，同时通过融合 OpenStack 和 Kubernetes 为边缘侧平滑引入丰富的云原生技术栈，辅助边缘应用更快速的迭代和获得一致的边云体验。

- 统一基础设施，避免七国八制，支持虚机容器、裸机容器、虚机、裸机共平台部署管理，灵活调整适应不同资源规模
- 支持基于微服务容器和虚拟化的 CT 能力内建，支持大数据和 AI 服务的平台赋能，助力 DICT 数字中台实现

## 异构计算 Hyper Computing >>

- 支持异构硬件平台部署，从嵌入式设备、定制化服务器到通用服务器
- 支持异构基础设施平台部署，从私有云（裸机、KVM、vSphere）到公有云
- 支持异构位置部署 Any Position，云管边端不同位置，支持大规模部署、轻量化部署和拉远部署
- 支持异构业务属性部署，从 CT 网元、通用 IT APP 到高性能 / 低时延处理业务等

### 高性能云原生 Cloud-Performance Native >>

• 基于云原生架构，100% Upstream API 方便集成，支持经过 CNCF 认证的 Kubernetes 平台，支持容器应用秒级启停、支持策略弹性伸缩，支持 ServiceMesh 和 Serverless 框架功能

• 统一技术框架，资源管理高效融合，支持 2 节点起步，最小 4C 平台资源开销、提升资源效率 75%

• 软硬件优化，提供高性能 FullMesh 硬件平台，转发全硬件加速和灵活的 GPU 资源供给方式，融合 IP 和 IT 的能力，优化成本

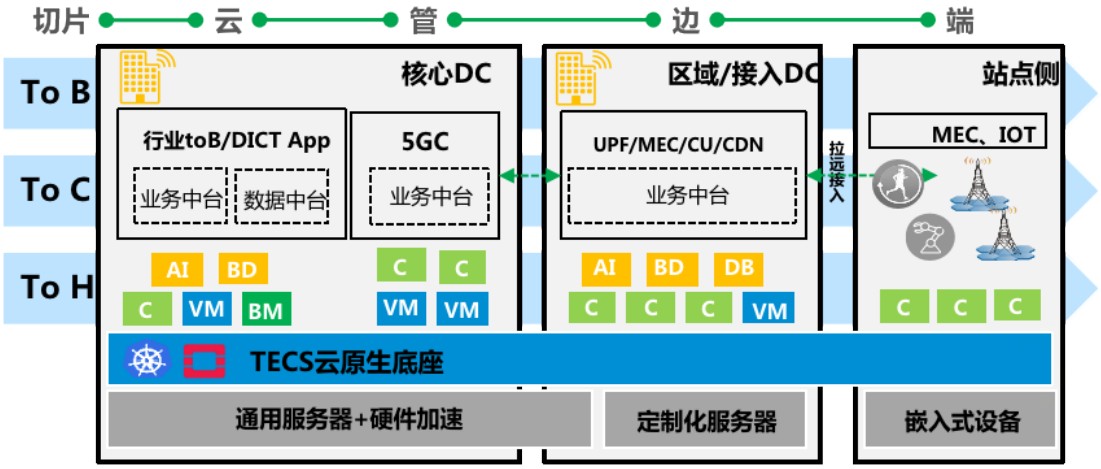
### 极简运维 Simple Self-Management >>

• 双引擎共享资源，降低 CAPEX，继承现有云基础设施和管理方式，可一键式升级为双核云原生基础设施，确保演进过程中一致的 SLA 体验，支撑业务平滑演进

• 统一管理视图，降低 OPEX，运维视图微服务化可定制，统一安全治理框架

• 支持无人值守、开箱即用、远程部署，部署效率提升 60%

• 支持多云统一管理和混合云管理




“TECS 云原生底座”已应用于中兴通讯边缘计算相关产品，正与国内国际数十家运营商和生态合作伙伴开展合作，在 5G、MEC 、CDN、IOT、OSS 等领域为运营商边缘计算的按需部署提供更灵活、更低成本、更具演进性的边缘云解决方案。

让传统企业也获得互联网公司的能力！

Traditional businesses can also acquire the capabilities of Internet enterprises!

欢迎关注微信

回复***白皮书***，看大图