阿里云
专有云Agility版

技术白皮书

产品版本：V1.0.4

文档版本：20180315

法律声明

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。

1. 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。
2. 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

表 -1: 格式约定

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。
	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。
	用于警示信息、补充说明等，是用户必须了解的内容。	 **说明**： 导出的数据中包含敏感信息，请妥善保存。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **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。
斜体	表示参数、变量。	bae log list --instanceid *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。	swich *{stand | slave}*

目录

法律声明.....I
通用约定.....I
1 容器服务..... 1
1.1 什么是容器服务.....1
1.2 功能特性..... 1
1.3 产品优势..... 3
1.4 产品架构..... 3
1.5 环境要求..... 4
1.6 应用场景..... 5
1.6.1 容器化应用上云..... 5
1.6.2 容器化DevOps.....5
1.6.3 微服务支持..... 6
1.6.4 混合云.....6
1.7 行业场景..... 7
1.7.1 电商..... 7
1.8 成功案例..... 8
1.8.1 某跨国企业迁云..... 8
1.8.2 某大型互联网公司DevOps与微服务转型..... 8
1.9 基本概念..... 9
1.10 容器技术..... 10
1.10.1 应用场景..... 10
1.10.2 容器技术与虚拟化.....12
1.10.3 容器技术的优势..... 13
2 对象存储OSS.....14
2.1 什么是对象存储OSS..... 14
2.2 产品架构..... 14
2.3 功能特性..... 16
3 块存储EBS.....18
3.1 产品概述..... 18
3.2 技术架构及特点.....18
3.2.1 技术架构..... 18
3.2.2 技术特点..... 19
3.2.2.1 高可用.....19
3.2.2.2 高可靠.....19
3.2.2.3 规模性能..... 20
3.2.2.4 易用性.....20

3.3 技术原理..... 20
3.3.1 数据组织..... 20
3.3.2 多副本机制..... 21
3.3.3 数据重建..... 22
3.3.4 掉电保护..... 22
3.3.5 精简配置..... 23
3.3.6 快照原理..... 23
3.4 产品分类..... 24
3.5 技术指标..... 25
4 表格存储Table Store.....27
4.1 什么是表格存储.....27
4.1.1 技术背景.....27
4.1.2 表格存储技术.....28
4.2 功能特性..... 29
4.2.1 用户和实例..... 29
4.2.2 数据表.....30
4.2.3 数据分片.....30
4.2.4 表的常用命令与函数.....31
4.2.5 授权与权限控制..... 31
4.3 产品优势.....32
5 文件存储NAS.....34
5.1 什么是文件存储.....34
5.2 系统架构..... 34
5.3 基本功能..... 35
5.4 产品优势..... 36

1 容器服务

1.1 什么是容器服务

容器服务是面向企业中小规模专有云场景的敏捷云应用平台，支持Docker企业版。可以直接部署在企业已有的x86等硬件基础之上并加以管理，支持研发运维一体化、云原生应用架构和机器学习等场景，支持混合云管理，允许应用在公共云和自有数据中心物理机统一部署管理，支持应用无缝迁云、弹性伸缩应对突发流量等场景。阿里双十一大规模验证，Docker公司战略合作，提供真正企业级的安全和稳定服务保障。

1.2 功能特性

资源调度

- 支持大规模集群的统一资源池化管理，支持管理用户的物理机、VMware等现有IAAS环境。
- 支持根据应用需求动态调度容器，可选择多种维度的调度策略，比如资源维度（CPU、内存、GPU等）、可用性要求维度、应用拓扑的亲和性维度等等。

微服务

- 内置通用的服务注册、发现、路由、负载均衡等机制，对开发语言和中间件无特殊需求。
- 提供声明式方式配置，无需编码。
- 支持Spring Cloud等开源微服务框架。

DevOps

- 内置容器化DevOps最佳实践，可以实现一键式从代码提交到应用变更上线的全自动流程。
- 支持与三方、开源CI/CD方案整合。
- 提供不间断发布、蓝绿发布、灰度发布等发布机制，支持灵活、可控的服务更新。

日志与监控

- 提供企业级日志采集和输出方案。无缝集成容器日志采集，支持采集标准输出或指定目录的日志。
- 容器服务集成了ELK 第三方开源日志框架，并做了功能扩展，支持企业对容器日志服务的需求。
- 提供容器级别、应用级别和宿主机级别的多维度监控，提供服务和应用视角聚合数据。
- 支持脚本、URL等自定义监控。

- 容器服务集成了Grafana、InfluxDB、Elasticsearch等三方开源监控工具，并做了功能扩展，支持灵活的自定义监控Dashboard能力。

安全

- 与企业用户目录无缝集成，支持统一用户认证管理。
- 支持基于角色的授权模型，对集群资源灵活控制。

开放的接口

- 兼容Docker原生工具链和API。
- 兼容Docker Swarm和Compose编排能力。
- 兼容三方Docker工具。
- 无厂商锁定。

容器存储

- 支持本地和分布式数据卷管理，支持企业已有NAS和SAN存储。
- 可以通过插件扩展机制对接更多存储实现。

容器网络

- 支持容器间高性能跨宿主机网络通信。
- 支持与企业现有网络方案对接。
- 支持打通云上云下，混合管理。
- 提供插件扩展机制支持更多网络方案。

混合云

- 与阿里云无缝实现混合云。
- 支持对自有数据中心和公共云容器集群的统一资源、镜像、应用和安全管理。
- 支持工作负载动态迁移。
- 支持云突发、云灾备、异地多中心管理等典型场景。

自动化运维与管理

- 基于阿里云飞天基础架构管理平台。
- 提供硬件资产管理、自动化监控、运维能力。
- 支持基础网络服务管理。

无缝对接阿里云公共云

- 阿里云无缝实现混合云。
- 提供通过专线、VPN等网络互通方式。
- 复用阿里云提供数据复制机制，可实现数据迁移、异地灾备等多种混合云业务应用场景。

1.3 产品优势

Docker Engine

- 经过双十一大规模线上业务验证。
- 阿里和Docker战略合作；提供OS和插件的安全认证。
- 提供专业化技术支持团队。

敏捷

- 应用秒级启动。
- 打造软件研发流水线，应用迭代速度提升13倍，加速企业业务迭代。
- 良好的隔离性，可以多种应用类型混合部署，系统利用率提升5~10倍。

弹性

- 密切监控应用性能，根据负载情况自动伸缩资源，从容应对突发流量。

开放

- 商业化产品和Docker社区开源保持一致。
- 开放的产品技术路线。
- 活跃的生态。

1.4 产品架构

容器服务完全兼容Docker原生编排基础，兼容Docker Swarm Mode集群管理，全兼容Docker Compose模板编排应用。使用DTR存储和管理Docker镜像。支持通过图形化界面和Open API管理集群和容器应用。

容器服务部署在客户自有数据中心里，所有数据存储在客户本地，完全由客户控制。适配客户已有的资源，无需额外采购，充分利用现有硬件资源和软件资源。

保证用户应用的低成本上云，容器服务实现了兼容标准Docker API的程序接口，兼容所有的Docker镜像和Compose模板，支持应用无缝迁云。为第三方能力扩展，实现了灵活可定制的扩展机制。

容器服务的概念架构简略图示如图 *1-1:* 产品架构所示。

图 1-1: 产品架构

1.5 环境要求

硬件要求

- **规模**：5台起步，单集群可扩展到1,000台以上。
- **硬件**：对服务器无依赖，支持X86服务器和GPU加速方案；可适配传统的本地存储、集中式存储；千兆、万兆等网络。

软件要求

操作系统：

- Centos 7.3
- Ubuntu 16.04
- Suse 12 SP1

1.6 应用场景

1.6.1 容器化应用上云

传统运行在数据中心的应用如果要向云上迁移面临的挑战是应用的运行环境的适配以及应用部署方式的改变。原有数据中心的环境和云环境的差异越大，迁移的成本越高。

由于容器技术具有跨平台移植的特性，可以解决不同环境下应用运行差异问题。无论用户的应用是何种语言，何种技术栈，都能够通过容器技术封装成为跨平台可用的Docker镜像。从而将原来繁重的应用迁移变成一系列标准化的操作。

一般情况下，在数据中心里运行的虚拟机镜像无法直接迁移到云上环境，原来的镜像构建和部署脚本都要重新开发。采用容器技术可以使用容器镜像替代虚拟机镜像。利用容器技术将原有的传统应用镜像化，可以在尽量减少或者没有改变的原有代码的情况下快速上云，并保证部署的一致性。

利用Docker Compose部署模版，可以描述完整的应用栈和所需云资源。容器服务扩展了Compose模板支持服务治理：包括服务调度约束、服务注册/发现策略、弹性伸缩等；以及容器与外部资源的动态集成，如负载均衡，网络存储，RDS数据库服务等等。这样可以采用应用为中心的管理方式，大大提升了部署、运维效率。

这个方案的特点是：成本低、上线速度快，并且在完成迁移后可以利用容器服务的微服务支持进行进一步微服务化改造。

1.6.2 容器化DevOps

开发和运维中交接的代码，由于同一代码在不同环境中的差异性，往往导致很高的调试成本，软件的交付和测试周期长。开发运维团队需要花费很大的精力来屏蔽系统差异，标准化交付手段。

容器技术作为软件的交付手段，具有同一应用在多个平台的可移植性，和不同应用在同一环境中的相互隔离的特点。利用这个特性，使容器成为开发环境、测试环境、生产环境的标准的代码交付手段。在任何环境中运行的都是同一个容器，这样保证了由于环境不同所带来的差异被最大限度地屏蔽了。利用这种标准的交付手段，可以大大提高开发和运维的效率。

在容器服务上，更提供了容器化Jenkins的一键式部署模版，可以快速灵活地搭建一套CI/CD环境。Jenkins有着非常活跃的生态圈，我们也提供了Jenkins插件来充分发挥容器服务的能力，比如多种发布模式，保证新系统和服务上线的可靠。比如通过灰度发布，可以灵活控制新老版本的流量切换，快速、可靠地回滚。

1.6.3 微服务支持

微服务作为云应用常见的架构模式正在被越来越多地接受。但是随着服务实例数量的快速增长，对服务治理的要求会越来越高。容器服务提供了和语言无关的微服务治理能力，用户不限定语言和开发框架，不用改变应用逻辑，就可以利用轻松应对微服务的管理和伸缩。所有这些能力都可以通过在编排模版中以声明的方式指定，或者在容器服务控制台中可视化地指定。

从对SLA保证角度，容器服务提供了利用自定义命令/HTTP等健康检查的能力，可以细粒度地评估应用的健康状况。容器服务还可以按照资源约束调度和再平衡，当节点故障自动重调度、服务亲和性和跨可用区调度约束等机制保证可用性。

服务发现是微服务平台的一个重要能力，所有服务的启动和停止都能够自动注册到平台，服务之间可以非常容易地进行服务发现。更进一步，配合Routing Mesh，简单路由服务可以为应用提供灵活的4层、7层路由方式。

对于任何运维团队来说，日志的集中和分析，云资源的集中监控都必不可少的。容器服务提供了声明式日志、监控采集方案，只需要简单的声明要采集的日志目录，就可以自动收集日志。支持对接多种日志、监控存储。

配合监控服务，在用户自定义的弹性伸缩策略后，容器服务对应用的指标进行监控，在触发条件是按照策略进行应用弹性扩容和所容。容器服务同时也支持集群弹性伸缩。

1.6.4 混合云

容器服务支持计算能力跨界迁移，通过容器服务可以管理本地专有云和公共云容器集群，提供了对所有环境完全一致的管理方式，用户无需针对不同的其他云服务商或者IDC机器寻求特定的管理方案。使用同一套镜像和编排模板让应用可以无缝在云间迁移。

混合云的典型场景包括：

- 场景1：应用快速在云端部署伸缩，应对大促突发流量峰值
- 场景2：异地、同城灾备
- 场景3：本地的机器开发测试，云端预发生产；或反之

针对混合云，为了提供最佳的网络实现，我们提供了不同的网络方案来适配不同环境。对于IDC与阿里公共云容器互通的需求，提供基于VPC的混合云互连方案，使用VPC+专线/VPN，可以实现IDC机器、IDC容器、ECS机器、ECS容器的全部互通。容器跨宿主机互联方案包括：Overlay（VXLAN）/Layer 3（Calico）等。如下图 *1-2:* 架构方案所示。

图 1-2: 架构方案

1.7 行业场景

1.7.1 电商

针对电商行业大促常态化，响应快速，流量峰值高等特点对基础架构所带来的挑战，很多电商用户开始采用容器技术解决动态管理的问题。基础架构容器化所带来的优势包括：快速扩容/缩容、提升资源使用率、提升动态响应能力。实际上，很多电商应用都可以容器化，比如Web，后台交易系统，移动端（无线），但这里带来的挑战包括如何进行全链路监控、全链路压力测试等。

针对电商行业的秒杀和抢红包，我们开发了可以基于容器技术的可以应对瞬时高并发的实例应用，显示了容器技术结合底层弹性云能力所带来的巨大潜力。在大促期间本地资源不足的情况下，可以利用容器服务的混合云部署模式，线上线下统一管理容器集群，并且利用公共云、专有云、数据中心资源等进行计算能力的跨界迁移，保证应用的稳定可靠和高并发处理能力。

1.8 成功案例

1.8.1 某跨国企业迁云

客户是一个跨国互联网企业，应用环境从其他云厂商迁移到阿里云。海外环境包含几十个应用，在线/离线处理，多语言栈混合。在原云厂商上的系统由虚拟机镜像和基于puppet自行研发的部署脚本。

这里客户遇到的挑战是：原来的虚拟机镜像无法直接移植到阿里云上，在原来云厂商上的自动化部署脚本无法运行在阿里云上，需要重新开发。客户的现状是在国内开发运维团队人数非常少，需要在不改变原有代码的情况下快速上线。

阿里云和客户合作的解决方案是利用容器技术将原有的传统应用镜像化，并部署到容器服务上。这个方案的特点是：

- 应用零修改；全自动构建、上线。把原有虚拟机的脚本通过Dockerfile构造一样的镜像，支持镜像持续构建、持续集成。
- 通过*docker-compose.yml*描述多容器应用一键部署。
- 集成日志服务进行日志处理。

项目的实际执行情况是客户用非常少的人力，没有改变代码的情况下，在1个多月的时间内顺利把几十个应用全部迁移到了阿里云上。

阿里云容器化方案为客户带来的价值包括：学习成本低，人工投入少，客户可自助完成迁移工作；原有应用不必做改动，极大减少部署脚本工具的开发量，降低由于代码改动出错带来的成本。除了成本低外，上线速度也要比传统的迁移快。容器化完成后可以进行微服务化改造，能够逐步将传统应用升级为云应用。

1.8.2 某大型互联网公司DevOps与微服务转型

客户在业务快速发展的同时，IT系统规模增长迅速，对IT系统稳定性、成本（资源+人工）带来巨大挑战。转型迁系统都在第三方云上，有将近千台服务器规模，但云厂商最近稳定性问题突出，所以决定迁移到阿里云。

客户所面临的挑战包括：运维部门人手不够，现行运维模式导致工作繁重，效率低下。亟待通过践行DevOps来尝试解决此问题。产品线拓展，业务系统强耦合的现状成为新业务拓展的瓶颈，也加重了开发、运维的负担，决定向微服务架构转型。

客户的团队对容器服务几乎0基础，在DevOps方面及微服务架构领域均处于探索阶段，并没有太多经验。

阿里云和客户共同指定的方案是在迁移阿里云的同时对原有服务进行规划分析和服务化的工作，利用容器技术作为载体，在迁移的同时完成向微服务架构转型。

1.9 基本概念

容器服务管理Docker集群所需的重要信息，包括集群、节点、容器、镜像、编排模版、服务和应用。这些元信息是支撑容器服务的重要概念。

集群

一个集群指容器运行所需要的云资源组合，关联了若干服务器节点、负载均衡、专有网络等云资源。

节点

一台服务器（可以是虚拟机实例或者物理服务器）已经安装了Docker Engine，可以用于部署和管理容器；容器服务的Agent程序会安装到节点上并注册到一个集群上。集群中的节点数量可以伸缩。

容器

一个通过Docker镜像创建的运行时实例，一个节点可运行多个容器。

镜像

Docker镜像是容器应用打包的标准格式，在部署容器化应用时可以指定镜像，镜像可以来自于Docker Hub，阿里云容器镜像服务，容器服务的DTR，或者用户的私有Registry。镜像ID可以由镜像所在仓库URI和镜像Tag（缺省为latest）唯一确认。

编排模板

编排模板包含了一组容器服务的定义和其相互关联，可以用于多容器应用的部署和管理。容器服务支持Docker Compose模板规范并有所扩展。

服务

一组基于相同镜像和配置定义的容器，作为一个可伸缩的微服务。

1.10 容器技术

容器技术是一种轻量级的操作系统级虚拟化技术。用户通过容器镜像来交付应用，其中包含了应用程序及所需的运行时依赖。容器镜像具有良好的可移植性，可以在不同环境下保证部署的一致性。容器之间运行时相互隔离，具有相当好的安全性。

容器技术避免了不同应用在同一个环境中可能存在的版本冲突，以及同一个软件在不同环境中可能存在的运行环境不一致的问题。所有容器共享宿主机的操作系统内核，这使得容器比虚拟机更轻量级，可以快速启动，并进行细粒度的资源控制。

1.10.1 应用场景

容器技术的特点是敏捷、可移植、可控。

敏捷

简单快速是容器技术吸引开发人员的重要特性，一致性的交付能力使得使得软件的交付更快，开发企业更敏捷。

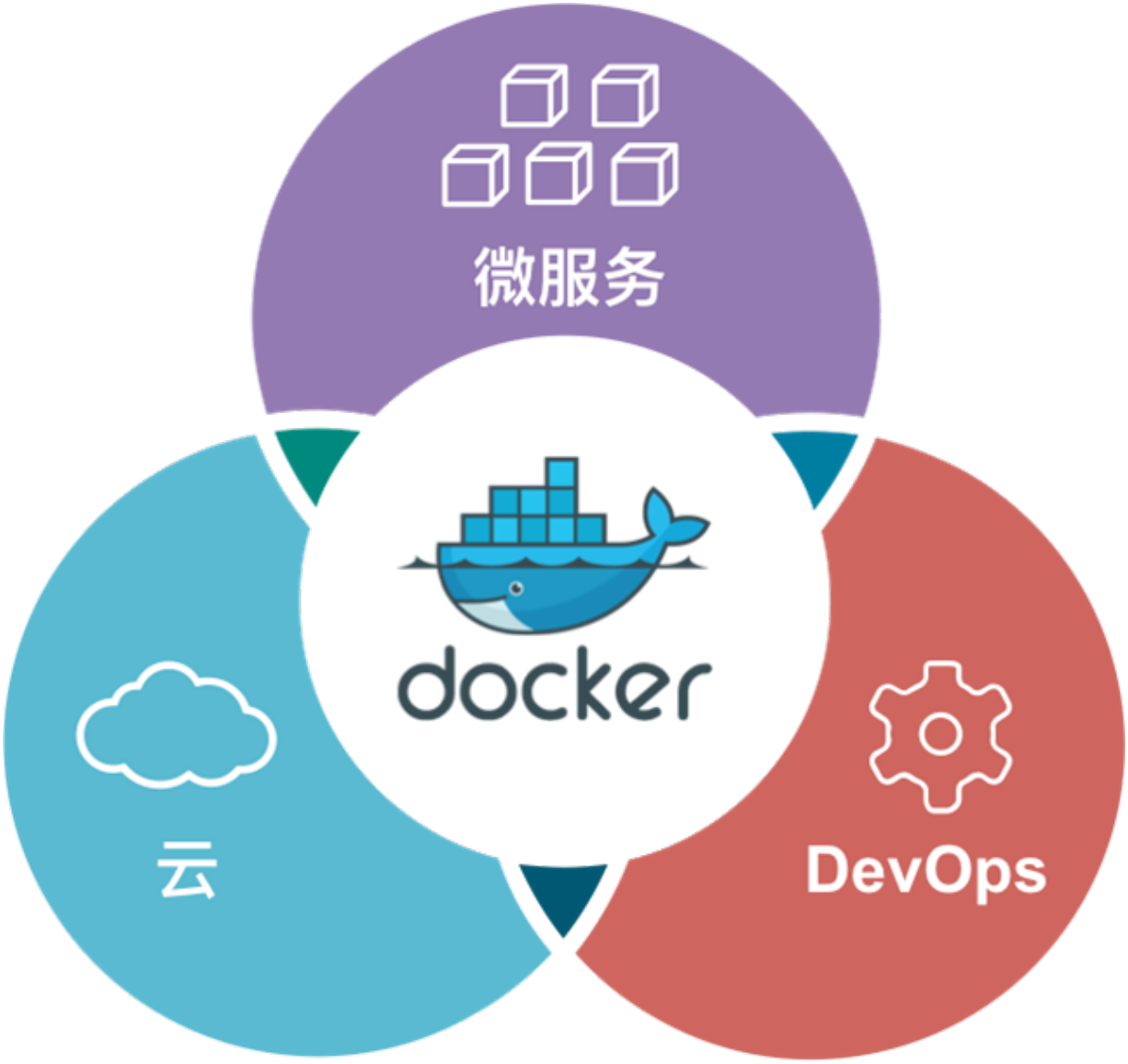
可移植性

开发人员可以把容器化的应用从开发转移到测试，最终到生产环境。在这个过程中同样的镜像运行结构一致。这意味着计算能力可以可以跨越数据中心边界进行部署，从而让混合云中的计算能力迁移真正可行。

可控

生产环境的应用需要保证SLA，要有完善的管理能力，安全和监控能力。容器技术应用环境标准化了，开发人员可以利用自动化工具来管理基础架构和应用，保证了所有操作自动化，可控，可回溯。

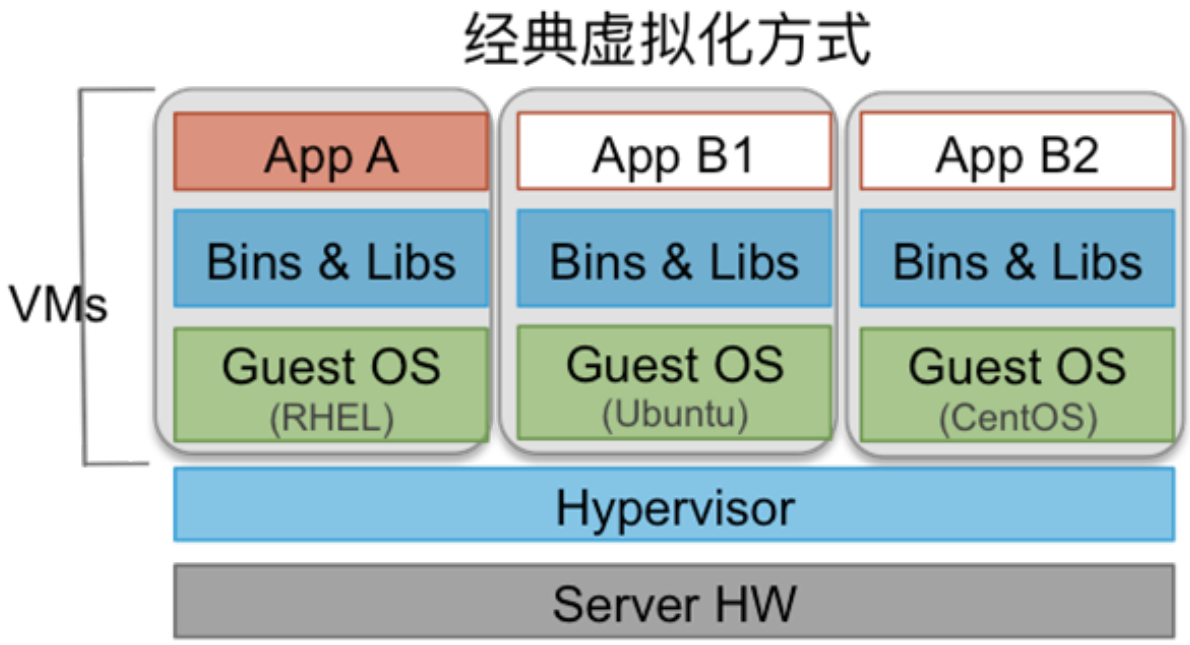
容器技术可以应用在非常多的场景。不过针对这三点要求很高的DevOps、云应用管理和微服务的应用场景讨论得比较多，研究也比较充分。


微服务
docker
云
DevOps

1.10.2 容器技术与虚拟化

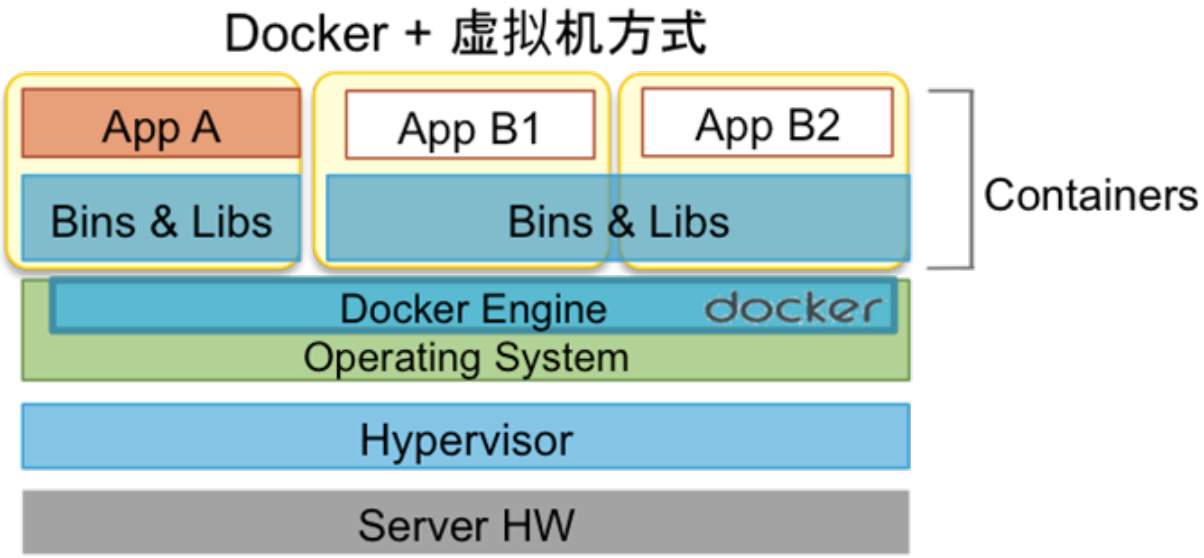
容器技术和传统的虚拟化并不冲突。传统的虚拟化方式是将操作系统到应用所有要素全部包含在一起。

图 1-3: 经典虚拟化方式



容器只将应用的代码和运行环境打包，镜像分层可以同同样环境的复用非常简单。

图 1-4: Docker + 虚拟机方式



结合容器和虚拟化技术，可以利用虚拟机提供弹性基础架构，提供更好的安全隔离，动态热迁移能力；同时还可以利用容器技术实现简化应用部署、运维，实现弹性应用架构。

1.10.3 容器技术的优势

容器技术作为轻量级的虚拟化技术，可以极大地提升资源使用率，实现资源的安全隔离。配合DevOps、微服务等方案，完成传统应用到云应用的升级，提高应用的开发、运维和管理水平，实现敏捷开发快速应对外部需求变化。

容器的打包技术还保证了二进制级别跨平台的可移植，镜像的版本管理可以让应用的管理流程全程可控，满足大规模企业及组织的需求。

2 对象存储OSS

2.1 什么是对象存储OSS

对象存储服务（Object Storage Service，简称 OSS）提供海量、安全、低成本、高可靠的云存储服务。它可以理解为一个即开即用，无限大空间的存储集群。相比传统自建服务器存储，OSS 在可靠性、安全性、成本和数据处理能力方面都有着突出的优势。使用 OSS，您可以通过网络随时存储和调用包括文本、图片、音频和视频等在内的各种非结构化数据文件。

OSS 将数据文件以对象/文件（object）的形式上传到存储空间（bucket）中。 您可以进行以下操作：

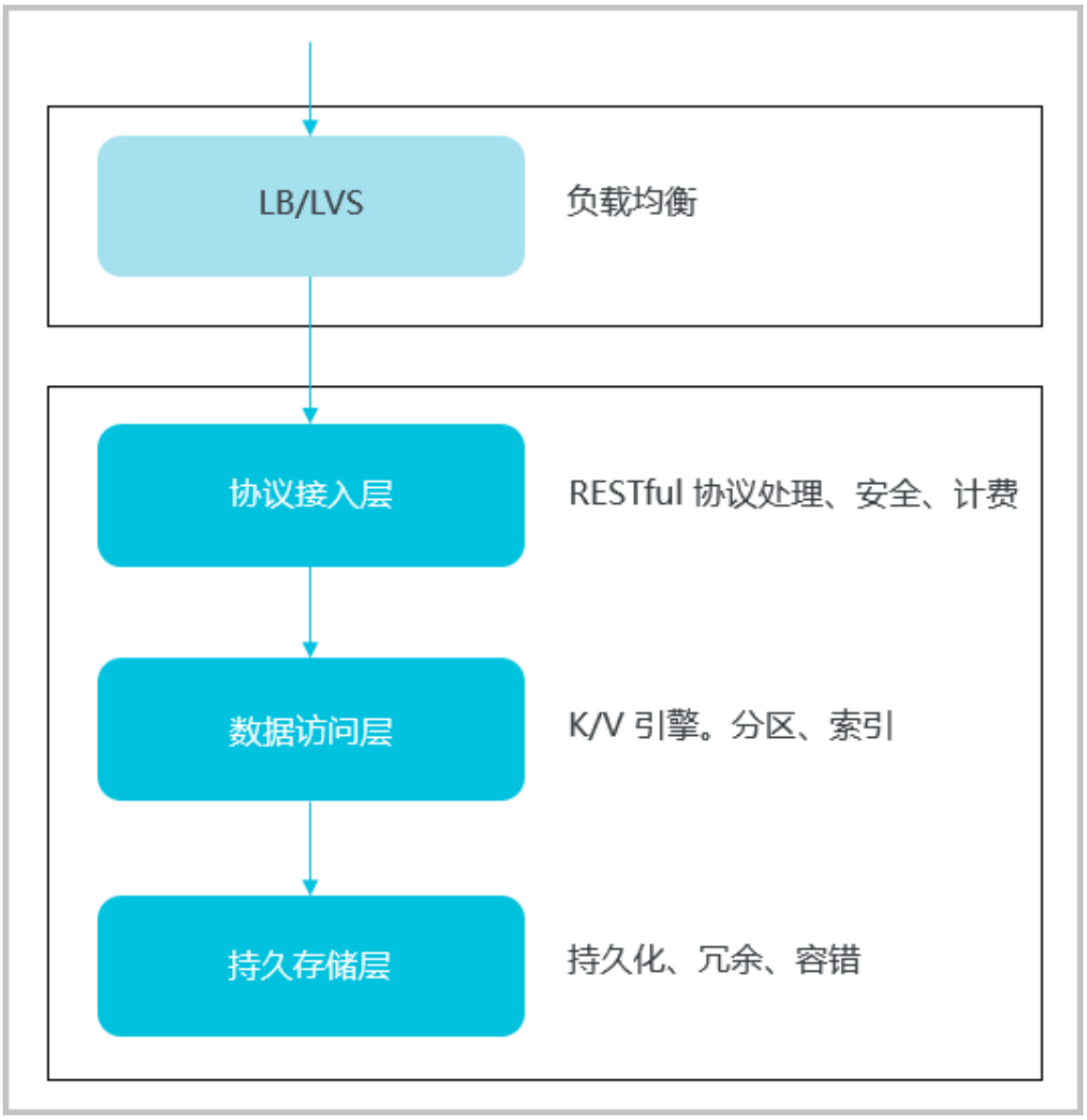
- 创建一个或者多个存储空间。
- 每个存储空间中添加一个或多个文件。
- 通过获取已上传文件的地址进行文件的分享和下载。
- 通过修改存储空间或文件的属性或元信息来设置相应的访问权限。
- 通过云控制台执行基本和高级 OSS 任务。
- 通过开发工具包 SDK 或直接在应用程序中进行 RESTful API 调用执行基本和高级 OSS 任务。

2.2 产品架构

对象存储OSS是构建在阿里云飞天平台上的一种存储解决方案。其基础是飞天平台的分布式文件系统、分布式任务调度等基础设施。该基础设施提供了OSS以及其他阿里云服务所需的分布式调度、高速网络、分布式存储等重要特性。

OSS的架构如图 *2-1: OSS架构图*所示。

图 2-1: OSS架构图



- 最上层是协议接入层，负责接收用户使用RESTful协议发来的请求，进行安全认证。如果认证通过，用户的请求将被转发到Key-Value引擎继续处理；如果认证失败，直接返回错误信息给用户。
- 数据访问层负责数据结构化处理，即按照Key来查找或存储数据，并支持大规模并发的请求。当协调服务集群变更导致服务被迫改变运行物理位置时，可以快速协调找到接入点。
- 最底层是持久存储层，即大规模分布式文件系统。元数据存储在Master上，Master之间采用分布式消息一致性协议（Paxos）保证元数据的一致性。从而实现高效的文件分布式存储和访问，保证数据在系统中有3个备份以及在软硬件错误发生以后的故障恢复。OSS系统的这一设计提供了不低于99.9% 可用性和99.99999999% 数据可靠性。

2.3 功能特性

存储空间概览

展示请求者所拥有的所有Bucket，在通过HTTP访问OSS服务地址时将默认展示您所拥有的所有Bucket。

设置并查询Bucket访问权限

- Private（私有权限）：只有该存储空间的创建者或者授权对象可以对该存储空间内的文件进行读写操作，其他人在未经授权的情况下无法访问该存储空间内的文件。
- Public-read（公共读，私有写）：只有该存储空间的创建者可以对该存储空间内的文件进行写操作，任何人（包括匿名访问）可以对该存储空间中的文件进行读操作。
- Public-read-write（公共读写）：任何人（包括匿名访问）都可以对该存储空间中的文件进行读写操作，所有这些操作产生的费用由该存储空间的创建者承担，请慎用该权限。

创建/删除 Bucket

一个用户默认最多创建 10 个 Bucket，当 Bucket 创建数量超过 10 时将返回错误信息。新创建的 Bucket 命名要符合Bucket命名规范，否则返回错误标志。如果创建的 Bucket 不存在，系统按照 Bucket 名称创建 Bucket，并返回成功标志；如果要创建的Bucket已存在，且请求者是所有者，则保留原来 Bucket，并返回成功标志；如果要创建的 Bucket 已存在，且请求者不是所有者，则返回失败标志。Bucket删除成功的条件有如下几个：Bucket 存在，访问者对 Bucket 有删除权限，Bucket 为空。

列出 Bucket 中的所有 Object

根据 Bucket 名称列出此 Bucket 下的所有 Object 信息，访问者必须具有对相应 Bucket 的操作权限，当访问的 Bucket 不存在时返回错误信息。

OSS 支持前缀查询，可以设置一次最大返回的文件数量（最大支持设置1000）。

上传/删除 Object 文件

上传 Object 到指定的 Bucket 空间下。在满足如下条件下 Object 会上传成功：Bucket 存在、访问者拥有对 Bucket 相应的操作权限。当 Bucket 中存在同名 Object 文件时将会覆盖掉原来的 Object 文件。根据 Object 名称删除某个特定 Object，访问者必须有此 Object 相应的操作权限。

获取 Object 文件或元信息

取得 Object 文件内容信息或者元信息，访问者需要对 Object 有相应的操作权限。

访问 Object

OSS 支持通过 URL 方式访问某文件。

日志及监控操作

用户可以选择开启 Bucket 的日志记录功能，一旦开启，OSS 会按照小时粒度推送日志。用户可以通过 OSS 控制台查询存储空间、流量、请求等信息。

3 块存储EBS

3.1 产品概述

弹性块存储（Elastic Block Service，简称EBS）是基于分布式文件系统实现的一个高可靠、高可用的块设备存储服务。EBS数据通过多份冗余存放在不同的机架下，在硬件出现故障时，提供可靠的数据安全保护能力。

EBS卷（EBS Volume）即一块虚拟的块存储设备，用户可以像使用物理硬盘一样来使用，可以格式化，可以挂载，可以执行I/O操作。

EBS卷有多种类型，不同类型的卷提供不同等级的I/O能力，以适应不同的工作负载。

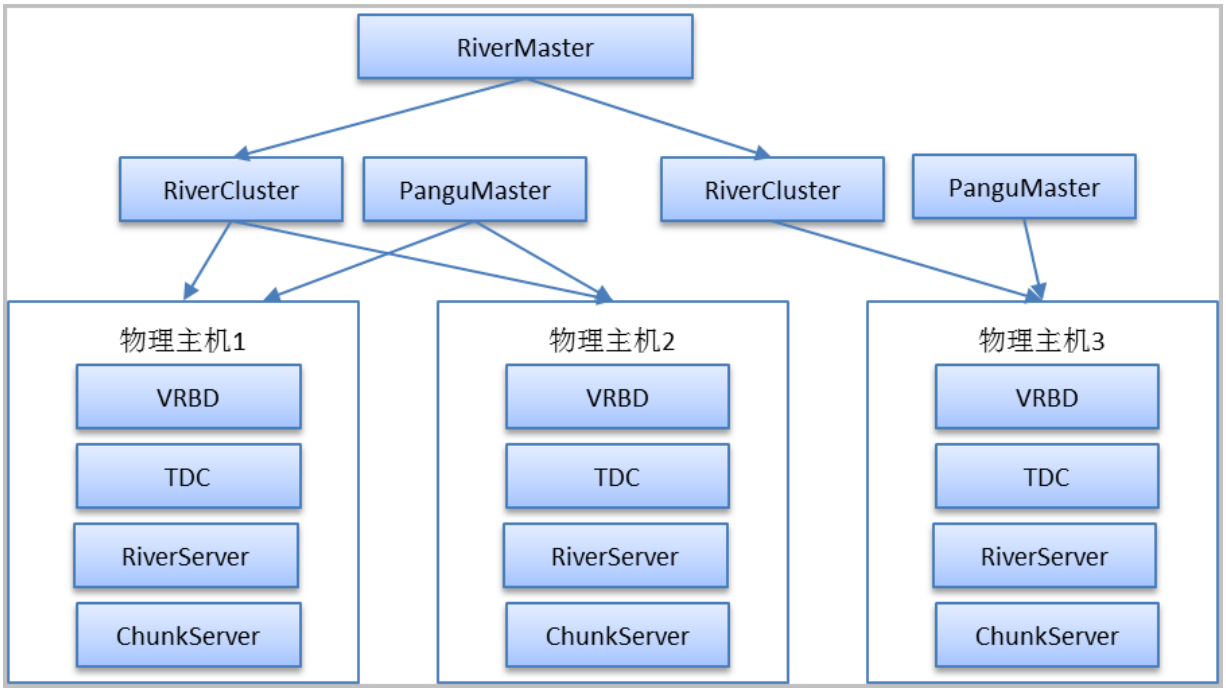
EBS快照（EBS Snapshot）是EBS卷某一时刻的数据备份，EBS快照所备份的数据按照固定大小的数据块存储在对象存储中。快照主要用于数据灾备、数据审计以及制作基础镜像等。

3.2 技术架构及特点

3.2.1 技术架构

块存储的技术架构如图 *3-1: EBS软件架构*所示。

图 3-1: EBS软件架构



其中，各组件及其功能如表 *3-1:* 各组件功能说明所示。

表 3-1: 各组件功能说明

组件	功能
VRBD	虚拟块设备驱动，部署在计算节点，通过VRBD能创建虚拟的块设备，能够被挂载到物理主机上，能进行IO读写，完全和物理介质的磁盘一样的使用体验。
TDC	TDC负责将后端存储和前端组件如VRBD桥接起来，承载块设备IO，同时提供形态丰富的CLI接口，如磁盘的创建、销毁，快照的创建与回滚等。
PanguMaster	盘古中最重要的角色，维护了文件和数据块之间的映射quota数据、chunkserver元数据、和checkpoint等。盘古通过多master机制保证master的高可用性，同时使得盘古具备热升级的能力。master之间log的同步和主master的选举采用了Paxos算法来保证其一致性。
ChunkServer	运行在数据物理服务器上，主要的职责是管理本地的硬盘和支撑对硬盘的读写删操作。
RiverMaster	在块存储系统中管控角色，负责磁盘在集群见的调度，包括磁盘创建的的调度、动态负载调度等，以及相关的管理工作，如存储库存水位管理等。
RiverCluster	负责管理一个集群，RiverCluster通过心跳维护RiverServer的状态。当RiverServer不可服务时，负责将RiverServer的磁盘动态迁移到另一台RiverServer上。
RiverServer	运行在每台物理主机上，RiverServer负责块层的IO读写处理，同时将IO请求转化后投递到Pangu文件层。

3.2.2 技术特点

3.2.2.1 高可用

块存储具备高可用特性：

- 提供多master机制，保证服务的高可用性和元数据的安全性。
- 多集群支持。

3.2.2.2 高可靠

块存储具备高可靠性：

- 用户通过minCopy和maxCopy分别指定盘古文件的副本数。

  盘古保证数据最少有minCopy份副本；尽可能的有maxCopy份副本。

  一般情况下，minCopy和maxCopy取值是2和3。
- 实现端到端的CheckSum，盘古在IO整个链路的各个环节中，都会对数据进行checksum校验。

3.2.2.3 规模性能

块存储具备如下性能：

- 提供用户自定义的数据聚簇方式，用户根据应用场景可以选择数据打散存放，或本地优先扎堆存放，又或者所有副本都扎堆存放。
- 提供优先级控制，保证后台复制的过程中，不影响前端的读写服务。
- 提供数据rebalance操作，保证数据均衡。
- 支持慢盘规避，pangu会定期检查物理磁盘的性能，并识别出性能异常的磁盘，隔离这些性能异常的磁盘，从而保证服务中的物理磁盘性均能符合预期。
- 支持SATA、混合存储及SSD。

3.2.2.4 易用性

块存储具备良好的易用性：

- 支持平滑热升级与回滚，对上层业务无影响。
- 损坏磁盘自动运维。

3.3 技术原理

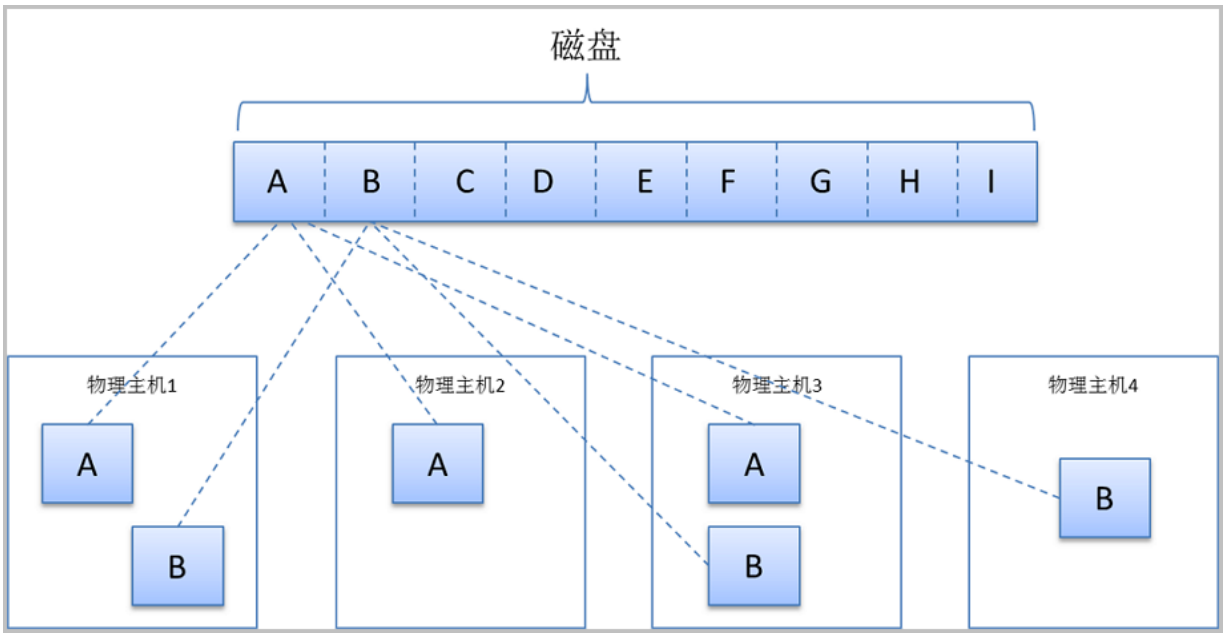
3.3.1 数据组织

EBS卷的数据组织具有如下几个特点：

- EBS卷全局按照一定的大小进行切片，一般都是基于64MB为单位进行分割。
- 切片的数据块，可以根据用户数据安全性的要求，存储多份冗余，一般是3个冗余。
- 不同的数据库冗余，分布在不同的物理主机上，防止单台主机异常后，EBS卷不可访问。

EBS卷的磁盘数据组织如图 *3-2: 磁盘数据组织*所示。

图 3-2: 磁盘数据组织

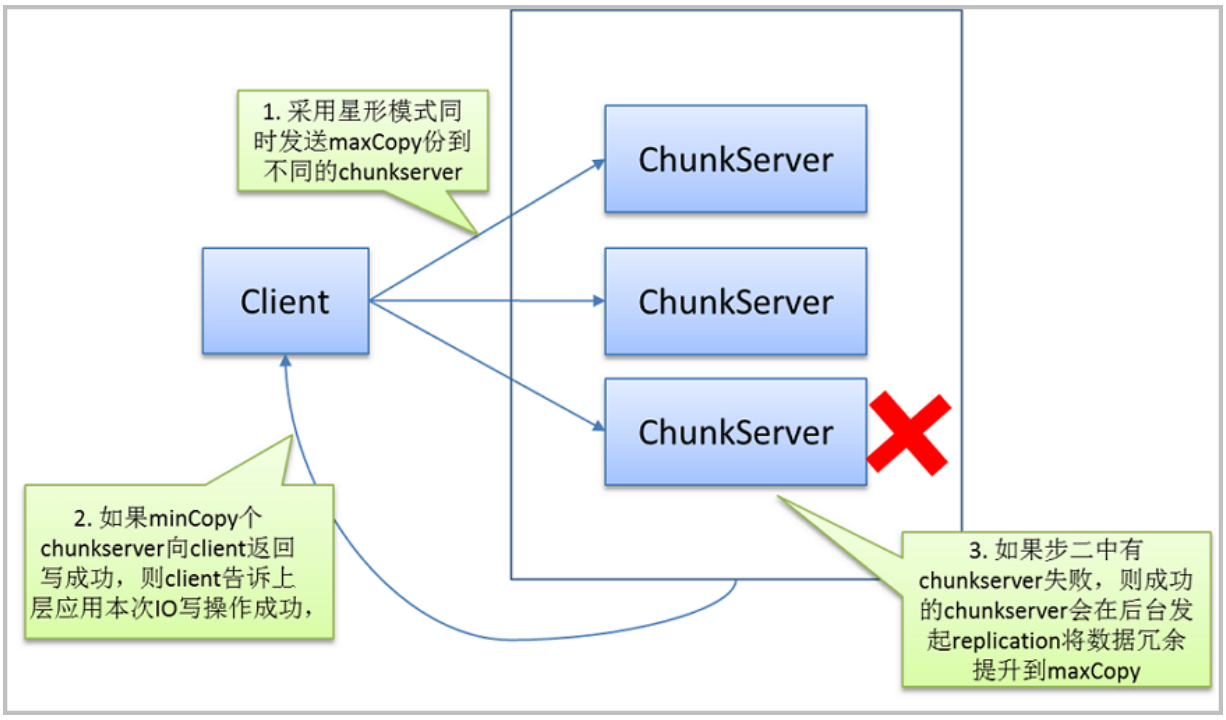


3.3.2 多副本机制

如图 *3-3:* 多副本机制所示，盘古多副本机制的内部实现原理如下：

1. Clinet采用星形模式，同时向服务器发送maxCopy份副本到不同的chunkserver。
2. 如果minCopy个chunkserver向Clinet返回写成功，则Client报告上层应用，本次IO写操作成功。
3. 如果上一步中有chunkserver失败，则成功的chunkserver会在后台发起replication，将数据冗余提升到maxCopy。

图 3-3: 多副本机制



3.3.3 数据重建

盘古具备强大的数据保护和智能恢复机制。

数据存储时已被切片打散到多个节点上，这些分片数据通过算法被分配在不同的存储节点、不同机柜之间以提供最大数据安全保障，同时数据存储时采用多副本技术，支持两副本或三副本，数据会自动保存多份，每一个分片的不同副本也被分散保存到不同的存储节点上。

在硬件发生故障导致数据不一致时，盘古一方面通过异步机制来保证服务的可用性，另一方面通过优化的自检机制，能够快速比较不同节点上的副本分片，自动发现数据故障，并在发现故障后启动数据修复机制。

在后台修复数据，由于数据被分散到多个不同的存储节点上保存，数据修复时，在不同的节点上同时启动修复，每个节点上只需修复一小部分数据，多个节点并行工作，配合精准的流控机制不仅能充分利用剩余网络资源，还可以有效避免单个节点修复大量数据所产生的性能瓶颈，对上层业务的影响做到最小化。

3.3.4 掉电保护

系统运行过程中，可能会出现服务器突然下电的情况，盘古在内存的元数据和写缓存数据会随着掉电而丢失，需要使用NVDIMM或SSD Cache来保存和恢复元数据和缓存数据。

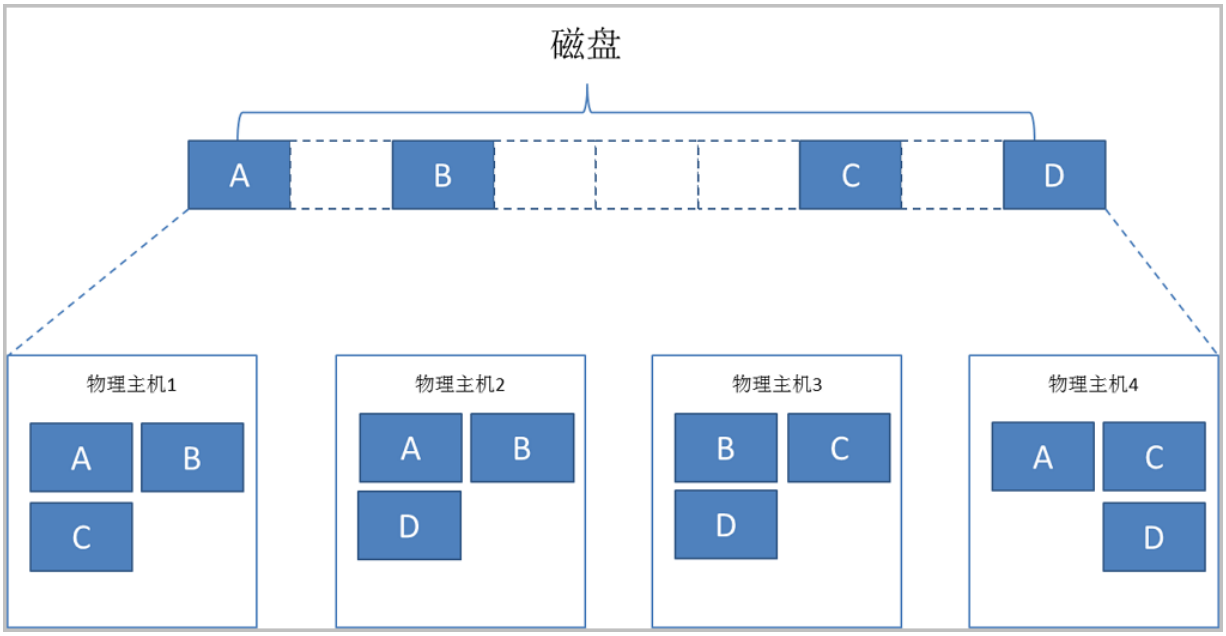
部署盘古软件的每一台服务器上要求配备SSD Cache，服务器掉电时，会把元数据和缓存数据写入SSD Cache中，上电后会自动把数据还原到内存中。盘古能够识别出系统中的SSD Cache，并根据用户对成本、性能的需求提供不同的存储服务，若需要提供掉电保护的支持，需要在配置表中明确设置。

3.3.5 精简配置

阿里云EBS块服务，默认支持精简配置，以降低存储的成本，提高资源利用率。

如图 *3-4:* 精简装置所示，用户磁盘真正写入的只有四个块，虽然磁盘的容量是9个块的容量。所以，落在底层盘古，也并不是按照9个块空间进行分配与存储，而只是存储了真实有数据的块，并进行索引维护，从而实现了精简配置。

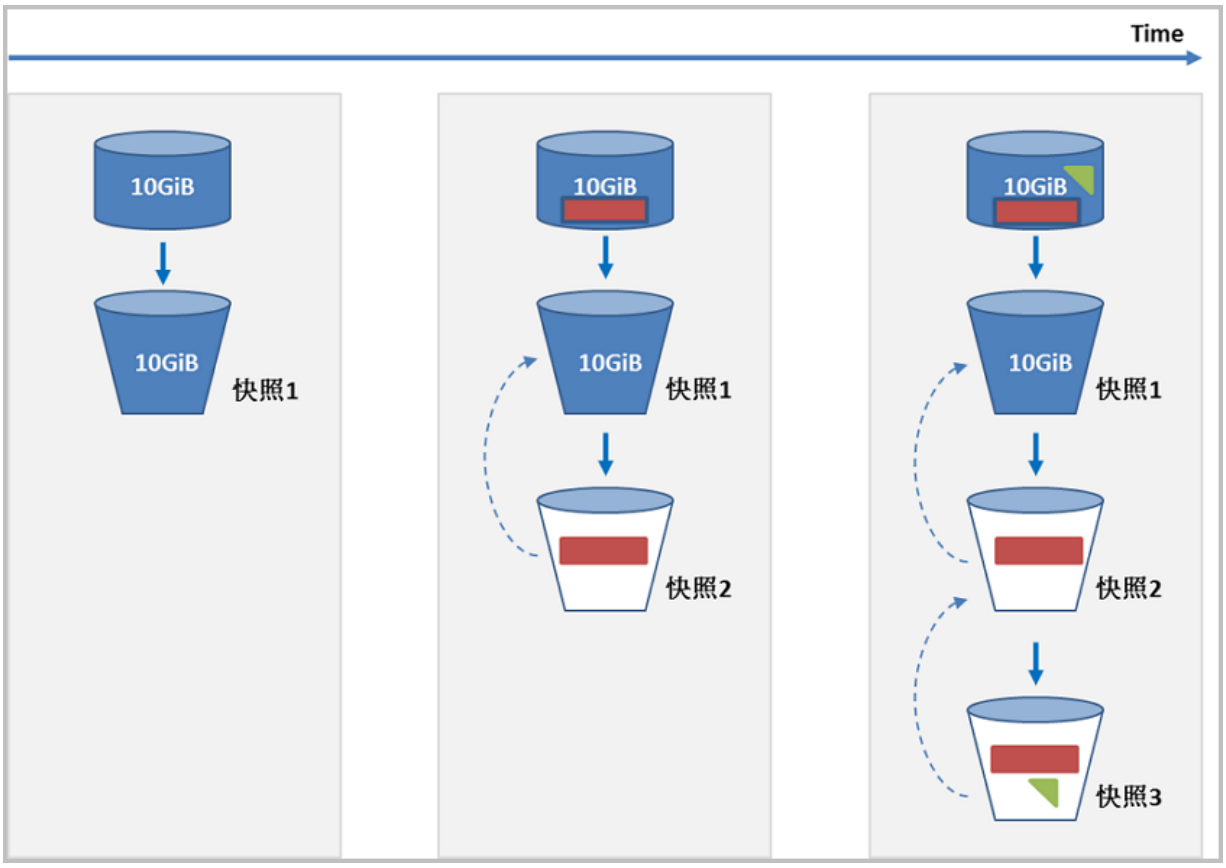
图 3-4: 精简装置



3.3.6 快照原理

EBS卷的快照原理如图 *3-5:* 快照原理所示。

图 3-5: 快照原理



EBS卷的快照具备如下特点：

- 快照通过增量技术实现，创建的第一个快照是卷的数据的全量备份，后续创建的快照均是自上次卷的数据量，进行增量备份。
- 快照是按照一定大小进行切割，并将数据有变化的数据块压缩后，备份到对象存储系统中。
- 快照的指令下发是秒级完成，快照数据同步是在系统后台进行的，不需要停止卷的运行状态。
- 快照删除后，不会影响后续快照的数据完备性，被删除的快照的数据块，只有引用计数为0时，才会被回收。

3.4 产品分类

EBS卷的分类如表 *3-2: EBS卷类型及其使用场景*所示。

表 3-2: EBS卷类型及其使用场景

EBS卷	主要场景	存储目标
SSD卷	• I/O密集型应用	• 单卷IOPS能力上万能力

EBS卷	主要场景	存储目标
	• 中大型关系数据库 • NoSQL数据库	• 单卷BPS能力上百MB能力
高效卷	• 开发与测试业务 • 系统盘 • 小型负载数据库	• 单卷IOPS能力数千能力 • 单卷BPS能力数十MB能力

3.5 技术指标

EBS弹性块存储的产品技术指标如表 *3-3:* 技术指标所示。

表 3-3: 技术指标

项目	描述
存储介质	支持HDD机械硬盘和SSD固态硬盘
部署模式	支持计算存储分离部署架构
硬件要求	支持通用X86架构服务器
单集群最大规模	10000
单集群最大硬盘数量	120000
单集群最大管理容量	480000 TB
单集群最大逻辑卷数量	500000
单卷最大容量	10TB
单计算节点挂载逻辑卷最大数量	200
资源管理和监控	• 支持块存储Volume的创建、挂载、卸载、删除、扩容 • 支持对资源池的利用率、性能等进行实时监控
存储访问协议	块设备访问协议
QOS	支持不同用户之间的资源隔离和QOS性能保障
冗余能力	支持块设备的多副本部署，副本数量可调
可靠性	支持服务器和机架级别可靠性
兼容性	支持Openstack、KVM、容器等虚拟化技术

项目	描述
GuestOS	支持Linux、Windows等主流操作系统
高级存储服务	• 支持快照功能，支持手动和自动快照 • 支持镜像功能 • 支持多租户 • 支持自动精简配置
管理接口	支持RestfulAPI、CLI及图形化管理界面

4 表格存储Table Store

4.1 什么是表格存储

4.1.1 技术背景

DT 时代下的数据特点

随着移动互联网的普及并深入到各个行业和领域中，互联网应用呈现出如下几个非常显著的特点和趋势：

- 应用需要存储和处理的数据量接近指数级增长，比如微博、社交事件、图片、访问日志等。
- 诸如手机等移动设备的普及以及物联网设备的增加，结构化数据的存储将面临越来越高的写入并发。
- 需要处理的数据没有严格的 schema，更趋向于半结构化，数据的字段会动态的变化。
- 用户的访问存在明显的热点和高峰，比如各种大促期间，应用的用户访问量会在瞬间达到非常高的值。
- 由于移动互联网无时无刻都在接入用户，用户对互联网应用的可用性要求也非常高，很难接受故障导致的服务不稳定甚至是计划中的服务停机。
- 大量的数据信息对计算分析提出了更高的要求。

传统 IT 软件解决方案的挑战

使用传统 IT 软件解决方案很难面对这些新的趋势和挑战，主要体现在如下几个方面：

- 规模可扩展

  传统的软件如关系型数据库很难处理这样快速增长的数据量，不管是在数据写入的吞吐率还是在数据量变大之后的访问效率上，都存在巨大的瓶颈。于是使用传统数据库的方案不得不进行手工和静态的分库分表策略，而这个策略也意味着很大的系统维护代价，特别是在增加节点进行扩容的时候，需要对已有的数据进行重新的切分和迁移，在这个过程中服务的性能、稳定性和可用性都很难得到很好的保证，而且整个过程是非常复杂的。

- 数据模型变更

  传统数据库处理的数据都具有严格的 schema，数据中包含的列数通常是固定的，很少去修改。频繁修改表 schema 和列数的设置，会对服务的可用性产生较大的影响，因此传统方案在面对结构越来越松散的互联网应用的数据时显得很力不从心。

- 快速伸缩

  传统的解决方案中，业务的访问压力是比较平稳的，系统不会经常面临资源需要快速调整（扩容和减容）的情况，因此一旦发生这种情况，就需要很大的代价，比如对数据进行重新切分，对切分之后的数据进行迁移，一旦业务压力下降之后，为了避免资源利用率低的问题，又要对多余的机器进行下线处理，又会经历数据的再一次搬迁，整个过程极其复杂且效率低下。

- 运维保障

  使用传统的软件方案需要专门来处理机器硬件（网络、磁盘）等设备发生故障时的服务恢复，需要处理硬件的更换，需要处理软件的版本升级、配置调优和更新，要让这些过程对应用透明、不影响服务的可用性，需要有专门的运维保障和系统工程师团队才能达到。不管是从人才招聘还是从成本投入上来说，这些工作对于快速发展的企业都是巨大的挑战。

- 计算瓶颈

  在现有的业务系统中，我们通常使用的是 OLTP （OnLine Transaction Processing，联机事务处理）系统来对数据进行处理和分析，如 MySQL、Microsoft SQL Server 等关系数据库系统。这些关系数据库系统擅长事务处理，在数据操作中保持着严格的一致性和原子性，能够很好支持频繁的数据插入和修改，但是，一旦需要进行查询或计算的数据量过大，达到数千万甚至数十亿条，或需要进行的计算非常复杂，OLTP 类数据库系统便力不从心了。

4.1.2 表格存储技术

表格存储是构建在阿里云分布式操作系统飞天之上的 NoSQL 数据存储服务，通过将数据表进行分区并且将数据分区调度到不同的节点上进行服务，从而提供可扩展的能力。在单机的硬件出问题时，表格存储服务通过心跳机制快速发现有问题的节点，并且把该节点上数据分区快速迁移到健康的节点上继续服务，从而达到服务的快速恢复能力。

数据分区和负载均衡

表中每一行主键的第一列称为数据分片键（Partition Key），系统根据数据分片键的范围将表切分为多个分区，这些分区被系统均匀地调度到不同的存储节点上。当单个数据分区内的数据不断增加到一定程度时，分区会自动分裂成两个更小的分区，从而将数据和访问 load 分散到两个分区上，这两个分区会被调度到不同的节点上，从而将访问 load 分散到不同的节点上，最终达到了单表数据规模和访问压力的线性扩展。

技术指标：表格存储支持的单表最大能够到 PB 级别，最多能够提供百万级别的并发读写能力。

单机故障自动恢复

在表格存储的存储引擎中，每个节点都会服务一批不同表的数据分区，这些分区的分布和调度信息由一个 Master 节点来负责管理，并且这个 Master 节点也会监控每个服务节点的健康状态。当发现某个服务节点出现问题时， Master 就会将原先分配给这个服务节点的数据分区调度到其他健康的节点上。由于数据分区的迁移只是逻辑上的迁移，不涉及实际数据的移动，所以当出现单机故障时，服务能够在很短的时间内恢复。

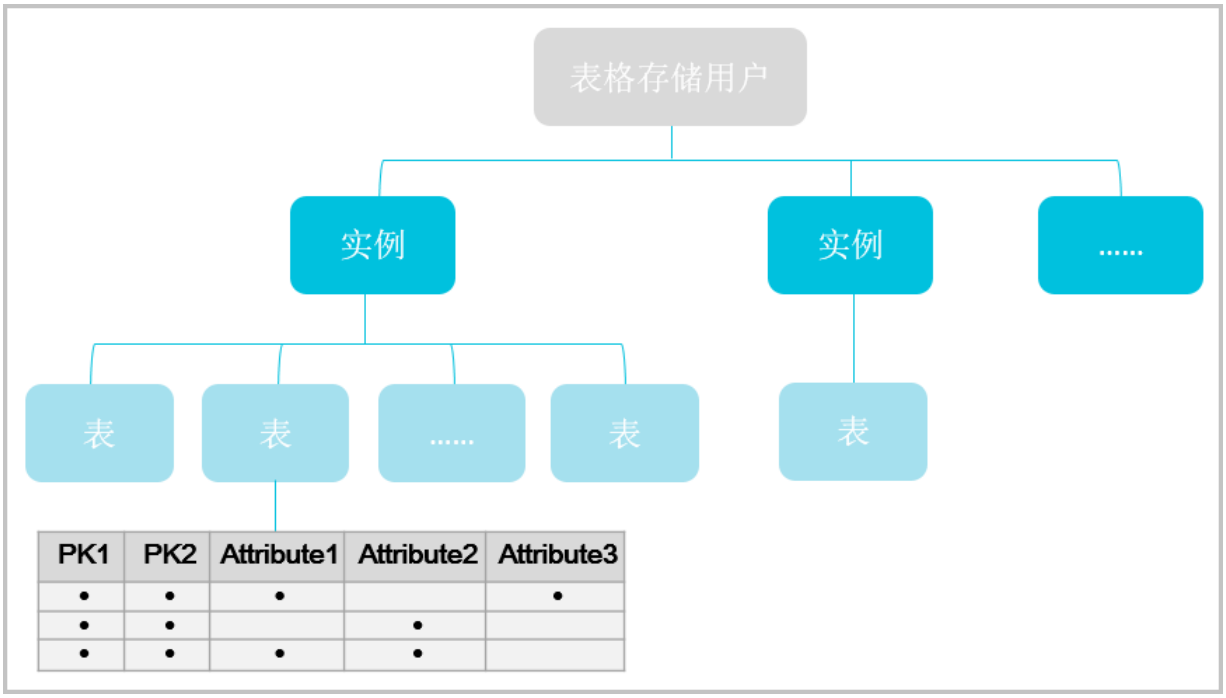
技术指标：单个机器节点的故障只影响部分数据分区的服务，并且能够在分钟级别恢复。

4.2 功能特性

4.2.1 用户和实例

用户和实例架构图如图 *4-1:* 用户和实例架构图所示。

图 4-1: 用户和实例架构图

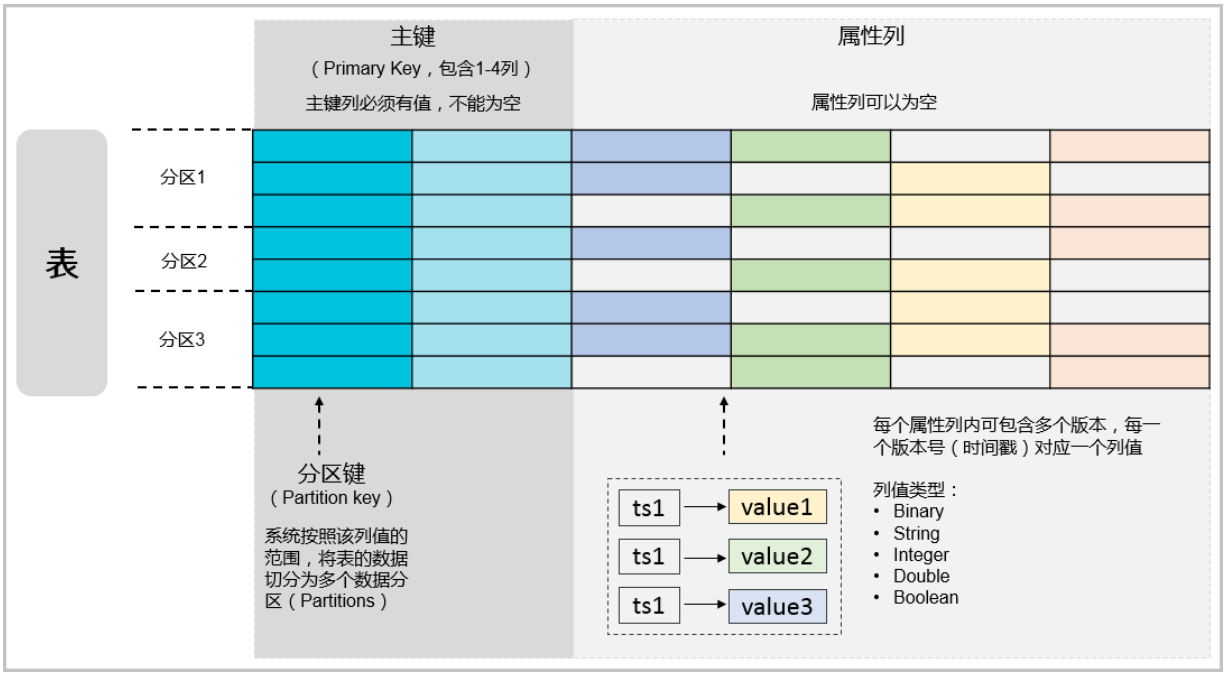


- 通过云账号进行登录。
- 用户的操作均可被细粒度审计。
- 用户通过实例来组织资源，一个用户可以创建多个实例，每个实例可以创建、管理多张数据表。
- 实例是多租户隔离的基本单位。
- 不同的用户可以授予不同的权限。

4.2.2 数据表

数据表结构图如图 *4-2:* 数据表结构图所示。

图 4-2: 数据表结构图



- 数据表是资源分配的最小单元。
- 表是行的集合，行由主键和属性组成。
- 表根据第一个主键列大小对数据进行分片。
- 表中的所有行都必须包含相同数目和名称的主键列。
- 每行包含的属性列的数目、名字和数据类型也可以不同。
- 每行属性列的个数没有限制，但一次请求写入的属性列的个数不能超过1024列。
- 单表可支持千亿行甚至更多数据。
- 单表数据规模可达到 PB 级别。

4.2.3 数据分片

- 表根据第一个主键列大小对数据进行分片。
- 第一个主键列值在同一个分片范围内的行会被分配到同一个数据分片。
- 表格存储服务会根据特定的规则对分片进行分裂和合并，以达到更好的负载均衡。
- 同一个分片键下的数据建议不超过 10 GB。

4.2.4 表的常用命令与函数

操作表的常用命令

- ListTable：列出实例下所有的表。
- CreateTable：创建数据表。
- DeleteTable：删除表。
- DescribeTable：获取表的属性信息。
- UpdateTable ：更新表的预留读/写吞吐量配置。
- ComputeSplitPointsBySize：将全表的数据在逻辑上划分成接近指定大小的若干分片，返回这些分片之间的分割点以及分片所在机器的提示。

操作表中数据的常用函数

- GetRow：读取单行数据。
- PutRow：新插入一行数据。
- UpdateRow：更新一行数据。
- DeleteRow：删除一行数据。
- BatchGetRow：批量读取一张或者多张表的多行数据。
- BatchWriteRow：批量插入、更新、删除一张表或者多张表的多行数据。
- GetRange：读取表中一个范围内的数据。

4.2.5 授权与权限控制

表格存储的权限

表格存储支持以下权限控制：

- 表级别的授权操作。
- API 粒度的权限控制。
- 支持 IP 限制、https、MFA（多因素认证）、访问时间限制等多种鉴权条件。
- 临时授权访问（STS）。

云控制台

- 支持云平台账号登录与鉴权。
- 提供图形化的实例创建、管理和删除的功能。
- 提供图形化的数据表的创建、管理、调整预留读写吞吐量和删除的功能。

- 提供表级别的监控信息展示。

4.3 产品优势

扩展性

- 动态调整预留读/写吞吐量

  在创建表的时候，应用程序可以根据业务访问的情况来配置预留读/写吞吐量。表格存储根据表的预留读/写吞吐量进行资源的调度和预留。在使用过程中，还可以根据应用情况动态修改预留读/写吞吐量。

- 无限容量

  表格存储中表的数据量没有上限，随着表数据量的不断增大，表格存储会进行数据分区的调整，从而为该表配置更多的存储。

数据可靠性

表格存储将数据的多个备份存储在不同机架的不同机器上，并会在备份失效时进行快速恢复，提供了极高的数据可靠性。

高可用性

通过自动的故障检测和数据迁移，表格存储对应用屏蔽了机器和网络的硬件故障，提供了高可用性。

管理便捷

应用程序无需关心数据分区的管理、软硬件升级、配置更新、集群扩容等繁琐的运维任务。

访问安全性

表格存储对应用程序的每一次请求都进行身份认证和鉴权，以防止未授权的数据访问，确保数据访问的安全性。

强一致性

表格存储保证数据写入强一致，写操作一旦返回成功，应用就能立即读到最新的数据。

灵活的数据模型

表格存储的表无固定格式要求，每行的列数及不同行同名列的类型可以不相同，支持多种数据类型，如 Integer、Boolean、Double、String 和 Binary。

监控集成

您可以从表格存储云控制台实时获取每秒请求数、平均响应延时等监控信息。

5 文件存储NAS

5.1 什么是文件存储

阿里云文件存储（Network Attached Storage，简称NAS）是面向阿里云ECS实例、HPC和Docker等计算节点的文件存储服务，提供标准的文件访问协议，您无需对现有应用做任何修改，即可使用具备无限容量及性能扩展、单一命名空间、多共享、高可靠和高可用等特性的分布式文件系统。

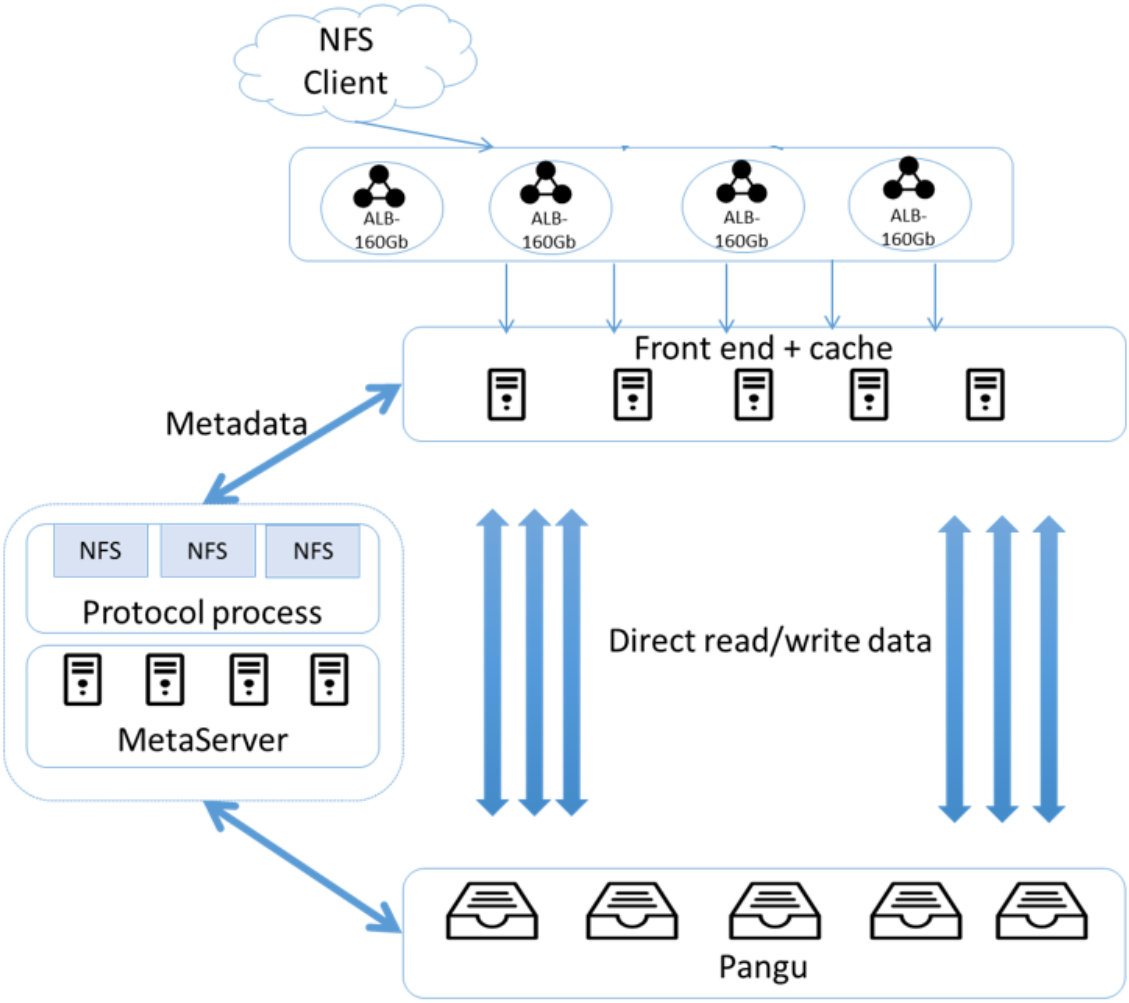
您创建NAS文件系统实例和挂载点后，即可在ECS、HPC和Docker等计算节点内通过标准的NFS协议挂载文件系统，并使用标准的Posix接口对文件系统进行访问。多个计算节点可以同时挂载同一个文件系统，共享文件和目录。

5.2 系统架构

NAS文件系统后端基于阿里云盘古分布式存储，数据3副本分布存储于多台盘古节点上。前端访问节点接受NFS客户端的连接请求和提供Cache功能，自身是无状态和分布式部署，保证前端的高可用。文件系统的Metadata数据保存在MetaServer上，前端机的IO请求在用MetaServer获得文件系统的Metadata后，User Data的读写直接到后端盘古数据节点，如图 *5-1:* 系统架构图所示。

架构上前后端可以单独弹性扩展，在保证高可用的前提下，做到IO吞吐的高并发和低时延。

图 5-1: 系统架构图



5.3 基本功能

NAS支持NFS（v3.0/v4.0）协议，应用无需修改即可以使用。它们都可以满足用户业务对文件存储的需求，比如业务数据文件共享，OA系统后端文件存储，企业数据库备份的目标存储，业务系统日志的存储和分析，Web站点数据的存储和分发，业务系统开发和测试数据的存储等场景。NAS支持的基本功能如图 5-2: 基本功能所示。

图 5-2: 基本功能

5.4 产品优势

NAS的产品特性：

- 共享的文件系统

  支持1万个Client同时通过NFS v3.0/4.0协议挂载同一个文件系统，实现数据共享。

- 高性能

  集群的最大吞吐可达到20 GBps，IOPS可以达到2万以上。

- 弹性

  业务数据弹性增长，按需购买存储空间。最大单文件系统可达10 PB的存储空间，每个文件系统支持最多10亿文件，单文件最大32 TB。

- 高可用

  基于3副本的盘古分布式系统，可以达到高达99.99999999%的可靠性，保护用户数据安全。

- 安全

  支持ACL、主子账号等安全特性，保障用户数据隔离。

- 全局命名空间

  文件系统数据分布在整个NAS集群上，提供单一命名空间。