卓越架构技术框架

卓越架构技术框架与实践

文档版本 01

发布日期 2024-07-18

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 卓越架构技术框架简介

卓越架构技术框架（Well-Architected Framework）聚焦客户业务上云后的关键问题的设计指导和最佳实践。

以华为公司和业界最佳实践为基础，以韧性、安全性、性能效率、成本优化与卓越运营五个架构关注点为支柱，打造领先的卓越架构技术框架，支撑客户完成云架构设计、云架构治理体系建设、研发生产力提升、现代化应用构建及云营运维体系建设等关键问题解决。

架构支柱

- **韧性支柱：**

  旨在帮助企业构建具有高可用的应用系统架构，提高工作负载的韧性，使之在面对各种异常场景时仍能提供和维持可接受的服务水平。韧性支柱结合了华为公司韧性设计经验和业界最佳实践，总结并提炼出一系列设计原则与最佳实践，用以帮助企业利用华为云平台基础设施达到高可用、面向各种故障场景进行韧性设计，并具备一定的灾备能力；同时通过规范化变更、部署及应急恢复等处理流程，减少业务中断时长，提升可用性。

- **安全性支柱：**

  旨在确保业务的安全、可信、合规，通过一系列华为云架构的最佳实践保护工作负载免受各种安全威胁，降低安全风险。安全性支柱涉及保护云上系统、资产、数据的机密性、完整性、可用性以及合法、合规使用数据，保护用户隐私的一系列最佳实践。

- **性能效率支柱：**

  聚焦于如何设计出高性能的架构。作为基本的质量属性，性能的重要性和性能失败后果的严重性是无须质疑的。性通效率支柱为性能设计、性能优化提供一些技术方法和手段，可以用于系统的软件性能工程，也可用于指导性能调整和优化。

- **成本优化支柱：**

  专注于帮助企业高效地使用云服务来构建工作负载，面向工作负载的整个生命周期不断完善和改进，减少不必要的开支并提升运营效率，让云上应用始终最具成本效益。成本优化支柱结合了华为公司云成本运营经验和业界最佳实践总结提炼出的体系化实践建议。

- **卓越运营支柱：**

  融合了这些优秀实践，聚焦如何正确地构建软件，高效地运维软件，持续提供卓越的客户体验，包含：组织团队、设计工作负载、大规模运营工作负载和随时间变化改进工作负载的最佳实践。

应用场景

- **云架构治理体系建设**

  云平台将虚拟化、数据库与中间件、大数据与AI等技术融合业界最佳实践，以托管云服务的方式提供企业使用。随着业务上云，企业将不受限于自身的技术能力使用先进IT技术，企业可以基于先进的云平台与WA方法论，构建现代化架构治理体系，使能组织、流程、工具和产品，让企业在数字化时代处于领先地位。

  云架构治理体系不同于传统IT架构治理体系，通过现代化云平台及轻量化治理体系，使能业务安全、强韧性、资源高效、成本最优、敏捷创新。

- **云架构设计**

  由于云平台封装了底层软件技术的复杂度，让企业可以更聚焦业务应用设计。云架构设计鼓励以领域驱动设计(DDD)为架构设计起点，结合不同视角的架构视图，融入韧性、安全性、性能效率、成本和运营支柱，真正将云架构关注点融入到架构设计过程中。

- **云架构审视**

  随着业务需求和技术发展的变化，系统的架构也需要不断演进和优化。通过对照卓越架构技术框架的最佳实践，架构师对工作负载的架构进行全面、系统的评估，确保架构符合最新的需求、规范，符合最新的云上最佳实践。架构审视是一个持续的过程，建议在关键里程碑点进行审视或定期例行（如每半年一次）审视。

- **研发生产力提升**

  基于云的应用研发，技术、工具和工程实践都有很高的成熟度。业务上云后，基于云最佳实践升级工具链，改造研发流程，提升研发团队基于云的研发能力，引入先进的DevSecOps体系和确定性运维体系将大幅度提升企业的生产力，真正做到业务敏捷。基于华为公司20年的数字化实践和数百万企业客户的服务经验，华为云吸收业界先DevSecOps理念精华，提炼出DevSecOps质量效能管理体系典型特征，同时以价值流创造为核心，摸索出了一套行之有效的质量效能方法论和最佳实践。

- **构建高韧性、高可用的应用程序**

  华为公司结合内部韧性设计经验和业界最佳实践，总结并提炼出一系列体系化设计原则与最佳实践：

  - 帮助客户利用华为云平台基础设施达到高可用、面向各种失败场景进行设计，并具备一定的灾备能力。
  - 通过规范化变更、部署及应急恢复等处理流程，减少业务中断时长，提升可用性。

- **安全合规体系建设**

  云安全已经成为多维度的全球性挑战，华为云卓越架构技术框架结合业界先进的云安全理念和积累的网络安全经验和优势，参考世界领先的 CSP 优秀安全实践、摸索出了一 整套行之有效的云安全战略和实践。并且已经构建起多维立体、纵深防御和合规遵 从的基础设施架构，用以支撑并不断完善涵盖了 IaaS、 PaaS 和 SaaS 等具有优良安 全功能的常用云服务。

- **确定性运维体系建设**

  IT运维行业正在面临着颠覆性的变化，我们正在从保障设备稳定的防守型运维转向支撑业务敏捷的进攻型运维，从关注自身网络转向关注客户应用，从系统维护工程师转向研发工程师，这个转型的过程对运维提出艰巨挑战的同时，也给每个组织和个人提供了难得的发展机会。华为云SRE过去构建了一些能力，也还在持续解决新的挑战，我们已经构建了一套质量管理机制、一套运维平台、一支全球专家队伍，更重要的是，我们已经和很多客户一起开展了面向应用视角的稳定性提

升工作，助力客户提升应用稳定性，从应用层到平台底层，在成本、质量、效率中寻找最优方案。

- **云财务体系(FinOps)建设**

  FinOps是“Finance”和“DevOps”的结合，目的是解决企业管理云成本难题。FinOps基金会将FinOps定义为“不断发展的云财务管理纪律和文化实践，通过帮助工程、财务、技术和业务团队在数据驱动的支出决策上进行协作，使组织获得最大的业务价值”。企业云资源消费贯穿用云的整个过程，管理云成本也需要持续迭代优化。

  FinOps框架提出三阶段(可视、优化、持续运营)实践模型，指导企业持续优化。在优化时，FinOps指导企业找到成本、质量与效率的平衡，避免企业为了极低成本导致业务效率和稳定性受影响。在一个公司内部业务团队众多，各团队实践FinOps进展不一，不同团队可能处于不同的阶段。FinOps指导企业通过多团队协作和基于数据决策，精细化管理云成本。各业务团队成本可视，主动控制不超支不浪费；企业基于数据决策云投资，保障企业核心业务和战略业务方向的支出。企业应用FinOps后，持续降低单位业务成本。

- **应用优化**

  当前，企业大量的存量应用逐渐成为业务发展的阻碍，老旧、复杂、僵化的系统难以更新，昂贵的基础设施维护成本高，繁杂的部署过程也给发布加上了沉重的枷锁，导致发布缓慢，现有的架构和技术无法很好地适应现代软件开发，这些问题都对企业的发展带来新的挑战。但对于大多数企业来说，这些应用仍然是公司价值链的重要组成部分，为企业提供核心功能和数据。对负责存量应用处理的开发和运营人员来说，同样面临诸多挑战： 日益复杂的 IT 环境、不断增加的“技术债务”、有限的技能以及安全风险等，这些问题都将成为企业无法快速创新和实现业务目标的潜在风险。

  卓越架构技术框架（Well-Architected Framework）将为企业提供优化建议，企业结合实施策略，有选择有节奏的优化应用，以提升存量应用的韧性、安全性、性能及资源利用率，适应现代化软件开发，降低运营成本。

- **伙伴能力标签认证**

  华为云合作伙伴能力标签（简称能力标签）是华为云合作伙伴达到能力标准后获得的标识，华为云定义并维护能力标签的全集。

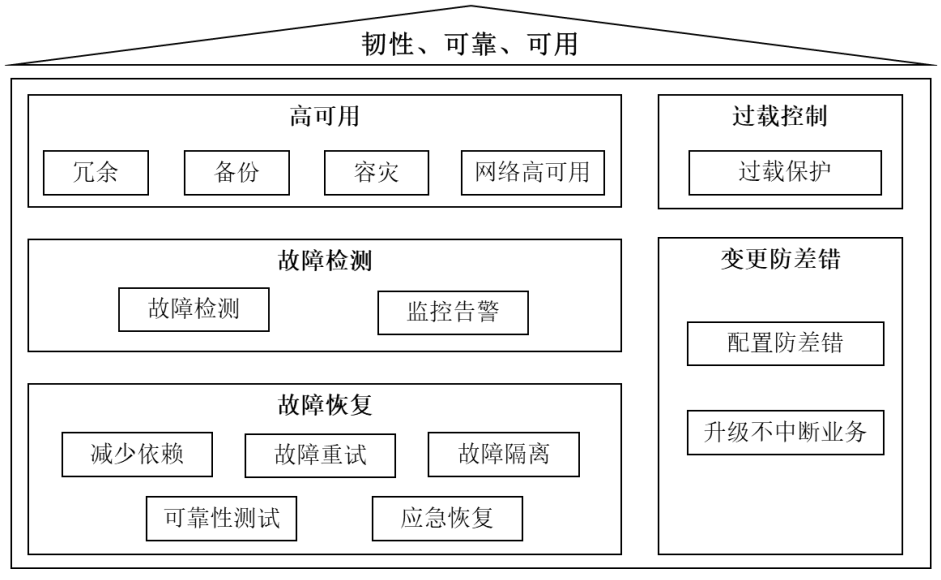
  合作伙伴通过学习卓越架构技术框架（Well-Architected Framework），理解并参考各支柱的云上最佳实践，以获取更专业的云架构设计知识。在构建解决方案或给客户提供专业服务的过程中，合作伙伴应用这些最佳实践，持续提升架构设计质量、持续完善工作负载。合作伙伴提交实际的客户案例并经过华为云审核通过后，可获得相应领域、场景或行业的能力标签认证。

2 韧性支柱

2.1 韧性支柱简介

韧性支柱旨在帮助企业构建具有高可用的应用系统架构，提高工作负载的韧性，使之在面对各种异常场景时仍能提供和维持可接受的服务水平。韧性支柱结合了华为公司韧性设计经验和业界最佳实践，总结并提炼出一系列设计原则与最佳实践，用以帮助企业利用华为云平台基础设施达到高可用、面向各种故障场景进行韧性设计，并具备一定的灾备能力；同时通过规范化变更、部署及应急恢复等处理流程，减少业务中断时长，提升可用性。

华为云韧性支柱的设计框架如下图所示：



2.2 基本概念

2.2.1 概念表

概念	解释
韧性 (Resilience)	系统从故障中保持在已知运行状态（甚至降级）的能力。在遭遇故障后快速恢复核心功能和数据，且在业务需要的时间窗内恢复到有效运行状态。
可靠性 (Reliability)	产品在规定的条件下和规定的时间内完成规定功能的能力。它的概率度量称为可靠度。
可用性 (Availability)	产品在任意随机时刻需要和开始执行任务时，处于可工作或可使用状态的程度。它的概率度量称为可用度
云服务指标 SLI	Service level Indicator，面向服务的指标，如：请求响应成功率
云服务目标 SLO	Service Level Object，面向服务的目标，如：一定时间范围内的请求响应成功率大于XX%，或正常运行时间的百分比
云服务协议等级 SLA	Service Level Agreement，面向用户的协议等级，涉及不满足时的补偿
数据恢复点目标 RPO	Recovery Point Objective，主要指的是业务系统所能容忍的数据丢失量
恢复时间目标 RTO	Recovery Time Objective，主要指的是所能容忍的业务停止服务的最长时间，也就是从灾难发生到业务系统恢复服务功能所需要的最短时间周期。

业界对韧性没有统一的定义。狭义韧性，指的是自动或快速从故障中恢复运行的能力；而广义韧性，除了从故障中恢复运行的能力外，还包括故障容忍能力。故障容忍（**fault tolerance**，简称“容错”），是使系统在其某些组件中出现一个或多个故障时能够继续提供服务的能力，从客户的角度来看，该服务仍能完全正常运行，或可能降级运行。

而可靠性同样分为狭义可靠性与广义可靠性。狭义可靠性工程的目标是提高系统无故障运行的能力，即提高可靠性。而广义可靠性工程的目标除了提高可靠性外，还包括提高从故障中恢复运行能力，即维修性（maintainability），同时还包括其他围绕故障展开的各种能力，如可用性（availability）、保障性（supportability）等。

因此，从广义韧性与广义可靠性的定义来看，并没有显著区别。只是可靠性和韧性的侧重点不同。可靠性工程的目标是尽可能减少系统中的故障，保证系统无故障运行。而韧性工程，接受故障总会发生的现实，关注的是如何降低故障带来的损失以及如何从故障中恢复。

2.2.2 什么是应用韧性

应用韧性是应用系统在运行过程中面对各种异常场景，如基础设施故障（如数据库异常）、外部攻击（如网络DDoS攻击超出预定限额流量）、外部依赖故障（如依赖系统访问超时或不可用）、地域灾难（如大面积停电、洪水）等，仍能提供和维持可接受的服务水平的能力，对系统至关重要。

系统韧性设计主要涉及以下两个方面：

- 确保系统具有高可用的架构，如无单点故障
- 各种故障场景下的恢复能力，如数据丢失、设备或站点故障等场景均能恢复

相对于传统数据中心，华为云可以提供具备高可用、弹性伸缩、自动备份、跨AZ容灾、跨Region容灾等高可用能力的基础设施与云服务，便于客户构建高可靠的系统。例如：

- EVS云硬盘、OBS对象存储采用分布式存储，可避免单个硬盘、单个服务器或单个机架等硬件故障的影响。
- RDS数据库提供自动数据备份、跨AZ和跨Region的数据复制与切换。

不过，即使应用系统利用云平台能力具有了这些高可用能力，要实现较高的可用性，仍需要构建针对各种偶发故障下的恢复能力，如：

- 由于硬件故障导致的高可用切换或跨AZ切换过程中，导致瞬时链接中断，需要应用系统具备链接中断重试的功能。
- 由于外部流量突发导致业务过载，需要应用系统具备流量控制的能力。
- 部分强依赖于硬件的负载，如依赖本地硬盘、GPU等，由于硬件故障导致服务中断，需要应用系统自身构建高可用的能力。

不同的应用系统，可用性要求可能不同，采用的韧性恢复方案会有差异。

2.2.3 责任共担模式

云上应用系统的韧性，依赖于云基础设施及应用系统本身的韧性，任何一方故障，都可能会导致云上应用系统故障；因此需要华为云与客户共同承担责任，来保障应用系统的韧性。

- 华为云责任：华为云提供高可用的基础设施，包括运行华为云服务的硬件、软件和机房设施，并确保服务可用性满足SLA服务等级协议。
- 客户责任：客户可以从华为云选择合适的产品并进行可靠性配置以符合应用韧性目标，并参考本白皮书中的设计原则与最佳实践，充分考虑各种异常场景的检测和恢复能力，来构建高可用应用系统。

2.2.4 可用性目标定义

可用性是衡量可靠性和韧性的综合性指标。

2.2.4.1 可用度及 SLO

可用性目标用于衡量应用系统的运行时间和停机时间，其表现形式为应用系统正常运行的时间占总时间（通常是一个月或一年）的百分比（如99.9%），即：

可用度 = 可用时间 / 总时间 * 100%

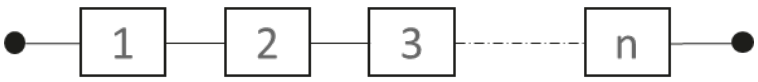
常见的简单表达方式用“9”的数量或“9”的数量加“5”表示，如“三个9”表示“99.9%”，而“三个9一个5”表示“99.95%”。

系统可用性目标通过服务等级目标（SLO）定义。不同的应用系统对可用性目标是不同的，明确应用系统的可用性目标，对于衡量应用系统的韧性至关重要。常见IT系统SLO示意如下：

SLO	每年最大不可用时间	典型IT服务
99%	3.65天	批处理，后台任务，数据抽取
99.9%	8.76小时	内部知识管理系统，项目跟踪系统
99.95%	4.38小时	客户账户管理，信息管理
99.99%	52.56分钟	电商，B2B web服务，大流量媒体/内容网站
99.999%	5.26分钟	银行，投资，金融，政府，电信，关键企业应用

系统的可用度依赖于系统内各业务单元的可用度。各业务单元之间典型的可靠性模型有两类：

- 串联模型：组成系统的所有单元中任一单元的故障都会导致整个系统故障的称为串联系统。



$$R_S = \prod_{i=1}^{n} R_i$$

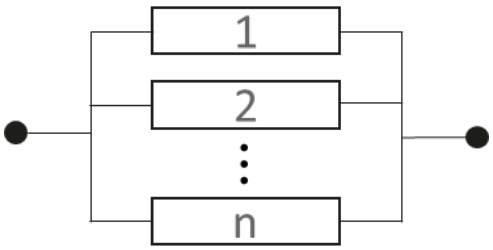
可靠性数学模型：

举例：假定系统存在2个串联单元，每个单元的可用度均为99.9%，则系统可用度为 Rs = 99.9% * 99.9% = 99.8%。

串联系统中系统可用度低于串联系统中任一单元的可用度。为提高系统可用度，设计时需考虑：

  - 尽可能减少串联单元数目
  - 提高单元可靠性，降低其故障率

- 并联模型：组成系统的所有单元都发生故障时，系统才发生故障的成为并联系统。



$$R_P = 1 - \prod_{i=1}^{n} (1 - R_i)$$

可靠性数学模型：

举例：假定系统存在2个并联单元，每个单元的可用度均为99.9%，则系统可用度为 Rs = 1 -（1 - 99.9%）*（1 - 99.9%）= 99.9999%。

并联可显著提高系统可用度，典型的并联技术有：主备、集群、双活或多活等。

应用系统要达到可用性目标，需对应用系统内组件及依赖组件进行可用性要求分解，包括：

- **对依赖组件的可用性要求：**通常关键依赖组件需要比其他服务提高一个9的SLO目标，如应用系统SLO目标为99.9%，则关键依赖组件SLO目标要求达到99.99%。
- **应用系统SLO分解：**综合系统SLO、故障频次、云服务SLA，分解得出应用组件的中断时长要求，进一步分解得出故障检测、人工介入、干预恢复的时长要求。
- **针对应用系统内薄弱环节进行增强：**
  - 当云服务SLA无法满足要求时，需要应用层进行额外的保护和增强。
  - 通过冗余提升可用度：包括组件冗余（负载均衡集群），故障回退冗余（fail-back，例如使用DMS访问失败时暂时切换到SMN）。

2.2.4.2 RTO 与 RPO

灾难场景通常采用RTO和RPO目标定义：

- 恢复时间目标RTO：指灾难发生后应用不可用的最长时间。RTO决定了应用容灾整体架构，是采用数据备份，还是冷备、温备、热备。
- 恢复点目标RPO：指灾难发生后应用数据丢失的最大时间。RPO决定了数据备份频率或复制方式，是在线备份还是离线备份，是同步复制还是异步复制。

国家标准《信息系统灾难恢复规范》（GB/T 20988-2007）中灾难恢复等级与RTO/RPO的关系如下：

灾难恢复能力等级	能力要求	RTO	RPO
1	基本支持：基本支持备份介质并场外存放	2天以上	1天至7天
2	备用场地支持：有备份场地，能调配所有资源	24小时以上	1天至7天
3	电子传输和设备支持：关键数据定时传送，备用网络部分就绪	12小时以上	数小时至1天
4	电子传输及完整设备支持：少量数据丢失，备用数据系统就绪，数据定时传送，备用网络就绪	数小时至2天	数小时至1天
5	实时数据传输及完整设备支持：数据丢失趋于0，备用数据系统就绪，远程数据复制，备用网络就绪	数分钟至2天	0至30分钟
6	数据零丢失和远程集群支持：数据零丢失，自动系统故障切换，远程磁盘镜像，备用网络active	数分钟	0

2.2.4.3 数据持久度

数据持久度是指数据不丢失的概率，即存储在预计周期内不出现数据丢失的概率，可以用于度量一个存储系统的可靠性。其只表示数据是否丢失的概率，不体现数据丢失多少；数据持久度的预计周期，一般按一年进行预计。

影响存储数据持久度的主要因子有：冗余数、磁盘失效率与数据修复时间。其中每多一个冗余，数据持久度通常可增加2~3个9；云上常用的对象存储，一般采用3副本冗余，通常可提供11~12个9的数据持久度。

2.2.5 可用性需求

根据“常见IT系统SLO示意”中的表格可以得知，不同的IT系统，SLO目标是存在差异的，不是所有的应用系统都需要达到最高可用性要求。

当系统可用性目标要求升高时，所需的成本也通常会增加，因此在可用性目标制定时，需要对韧性与成本进行权衡，确定真正的可用性需求。

在系统的可用性目标明确后，可参考以下韧性最佳实践来优化系统，使之满足可用性目标要求。

2.3 设计原则

由于故障不可避免，如硬件故障、软件错误、网络延迟、突发流量等，因此在设计高可用应用系统时，必须考虑所有的硬件及系统包括的软件都可能会失效，包括IaaS、PaaS、SaaS及应用系统本身。韧性设计的目标不是试图防止这些故障的发生，而是为了在这些故障发生时，能最大程度地减轻故障对系统造成的影响，并持续稳定地运行，建议遵循以下设计原则。

高可用设计

单点故障会导致整个系统崩溃、主要功能受到影响、任务延误的系统轻度损坏或存在较大的故障隐患，因此系统的高可用设计非常关键。

高可用设计的主要手段是冗余，甚至是多级冗余的组合，包括异地容灾方式保证灾难情况下无单点：

- 冗余机制：只要条件允许，需要考虑关键组件的冗余，甚至是多级冗余的组合（例如：1+1冗余、n+1冗余、N-Way冗余等）
- 异地容灾：例如，两地三中心，保证灾难的情况也可以提供业务。
- 数据冗余：可以通过定期备份和多副本备份等方式实现以提高数持久度，并确保数据一致性。

冗余的增加，意味着成本的增加；因此在应用高可用设计时需要综合考虑冗余对成本的影响。

故障全面检测

故障检测是故障管理的前提，检测全面与检测快速都很重要，通常情况下故障检测全比故障检测快重要。

故障检测涉及以下方面：

- 检测范围：识别并跟踪检测所有组件，有重大影响的故障模式需要重点检测。

  - 亚健康检测：对不引起系统故障却导致系统或服务KPI下降的亚健康异常需要能检测，如网络时延变大、磁盘变慢、内存泄露等亚健康故障。
  - 备用检测：冗余系统中，主备用模块的故障都需要检测，避免静默故障。
  - 有特殊寿命器件：应及时监控有特殊寿命（如本地硬盘）要求的期间健康状态，通过提前预警采取维护错误，避免故障的突然发生造成严重影响。
- 检测速度：需要根据业务综合要求，确定合适的检测速度。
- 检测影响：故障定时检测的周期，需综合考虑对CPU占用率的影响和检测延迟对业务恢复速度的影响。
- 检测模块要简单：故障检测系统、模块要比被检测系统、模块简单。

在检测到问题后，需要通过监控系统及时发现，迅速处理。

故障快速恢复

故障恢复指恢复产品执行规定功能的能力，一般情况下恢复越快影响越小。

结合业务情况，综合考虑技术实现难度、技术方案复杂度、成本等设计合适的故障恢复方案：

- 自动恢复：对于影响业务的故障，系统应尽可能自动恢复自愈，如保护倒换、局部复位或系统服务等。
- 优先恢复：优先对故障发生概率高、故障影响大的故障进行恢复。
- 分级复位：提供分级复位设计，尽可能在更小级别进行复位，以减少对业务的影响。
- 无耦合恢复：尽可能做到系统局部故障或各部件启动顺序不影响系统成功启动。
- 分层保护：系统故障保护要考虑网络分层，下层的故障保护倒换要比上层灵敏，防止系统出现乒乓倒换。

通过检测系统运行状态，或监控系统载关键指标，来判断系统是否发生故障，并针对故障可进行自动恢复处理。

可以通过故障分析方法分析各种故障模式、影响及危害，设计对应的可靠可用方案，提供冗余、隔离、降级、弹性等能力；并通过故障注入测试（FIT）验证可靠可用方案的有效性，最大程度提高业务的可靠性和可用性。

对于某些故障，即使通过各种技术手段进行冗余和自动恢复处理，但仍会导致业务中断，需要人工干预，如备份恢复或灾难恢复处理，因此需要建立高效的故障应急恢复处理流程和平台，以便在故障发生时，能快速恢复业务，减少故障影响。

过载控制

在系统请求超过系统容量时，会由于资源饱和而导致系统请求失败，在云中，可以监控系统和工作负载的利用率，来自动添加或删除资源，以维持最佳级别来满足业务需求，而无需过度配置或配置不足。

控制业务流量一般通过动态资源管理来实现，不建议简单的使用静态门限来达到防过载的目的，有可能造成资源大量浪费，过载设计应该考虑以下方面：

- 动态限流：根据系统资源消耗情况动态调整流控门限。
- 弹性扩缩容：自动检测系统资源利用率，自动进行添加或删除资源。

- 先负载均衡后流控：多个并行处理单元场景下，优先考虑负载均衡，避免单个处理单元资源受限导致业务受损；然后进行过载控制保护，使得整个系统的处理能力最大化。
- 及早控制：系统过载时，应尽可能在业务流程处理前端或业务处理较早的处理模块或底层协议层次上控制业务接入，避免中间控制带来不必要的性能消耗。
- 优先级保障：系统过载时保证高优先级的业务能够优先获得资源，优先得到处理，从而保证社会效益最大化。

变更防差错

当对系统进行升级部署、配置变更时，需要防止变更过程中由于人因差错导致系统和业务受损或失效。

通常采用防呆的方式来减少人因差错。防呆是一种预防矫正的行为约束手段，运用防止错误发生的限制方法，让操作者不需要花费注意力、也不需要经验与专业知识，凭借直觉就可准确无误地完成操作，在许多场景下可以提升效率和使用体验，也防止损坏更换的成本，因此优良的产品中防呆设计极为基础而普遍。

变更防差错通常采用以下方案：

- 角色约束：通过权限控制设计预防对不同角色的配置范围进行约束，避免越权配置导致错误。
- 查改分离：通过产品界面设计将配置界面分层分级，查看与修改分离等降低人为配置失误风险。
- 配置校验：通过配置生效机制设计确保在配置生效前进行必要的检查，避免错误配置生效。通过使用自动化方式进行配置变更处理，可减少人因输入错误的可能。
- 删除保护：在删除资源时增加保护机制，防止误删，如：删除前运行状态检查保护，资源锁定防止误删除，回收站机制等。

2.4 问题和检查项

企业在进行应用韧性设计的过程中，推荐使用如下问题寻找自身可以改进的点，并参考检查项/最佳实践进行改进，以下所有检查项，也是最佳实践建议，将在下一章节进行详细描述。

问题	检查项/最佳实践
RES01 您如何使用冗余技术确保应用系统的高可用?	1. 应用组件高可用部署 2. 应用组件多位置部署 3. 云服务器反亲和
RES02 您如何备份应用程序中的关键数据?	1. 识别和备份应用中所有需要备份的关键数据 2. 自动数据备份 3. 定期进行备份数据恢复

问题	检查项/最佳实践
RES03 您如何对应用程序进行跨AZ灾难恢复?	1. 集群跨AZ部署 2. 跨AZ数据同步 3. 对接容灾仲裁，支持自动切换 4. 支持容灾管理
RES04 您如何对应用程序进行跨Region或跨云灾难恢复?	1. 定义应用系统的容灾目标RPO与RTO 2. 部署容灾系统以满足容灾目标 3. 容灾恢复过程自动化 4. 定期进行容灾演练，以检查恢复能否满足容灾目标
RES05 您如何保证网络高可用?	1. 网络连接高可用 2. 避免暴露不必要的网络地址 3. 不同流量模型业务的网络共享带宽隔离 4. 预留IP资源以便扩展和高可用
RES06您如何进行故障检测处理?	1. 故障模式分析 2. 面向所有故障进行检测 3. 支持亚健康检测
RES07 您如何监控应用系统资源?	1. 定义关键指标与阈值并监控 2. 日志统计监控 3. 监控到异常后发送消息通知 4. 监控数据存储和分析 5. 端到端跟踪请求消息
RES08 您如何减少依赖影响?	1. 减少强依赖项 2. 依赖采用松耦合 3. 减少被依赖项故障的影响
RES09 您如何进行重试?	1. API以及命令调用需要设计为可重试 2. 客户端需要根据综合评估是否需要重试 3. 重试需要避免造成流量压力
RES10 您如何进行故障隔离?	1. 应用控制平面与数据平面隔离 2. 应用系统多位置部署 3. 采用Grid架构 4. 健康检查与自动隔离
RES011 您如何进行可靠性测试?	1. 混沌测试 2. 压力负载测试 3. 长稳测试 4. 灾难演练 5. 红蓝攻防

问题	检查项/最佳实践
RES012 您如何进行应急恢复处理?	1. 组建应急恢复团队 2. 制定应急预案 3. 定期应急恢复演练 4. 出现问题后尽快恢复业务 5. 应急恢复回溯
RES013 您如何进行过载保护以适应流量变化?	1. 采用自动弹性扩缩容 2. 应用系统负载均衡，避免流量不均匀 3. 过载检测与流量控制 4. 支持主动扩容 5. 资源自动扩容考虑了配额限制 6. 压力负载测试
RES14 您如何进行配置防差错?	1. 变更防呆检查 2. 自动化变更 3. 变更前数据备份 4. 提供runbook进行标准化变更
RES15 您如何进行升级不中断业务?	1. 自动化部署和升级 2. 自动化检查 3. 自动化回滚 4. 灰度部署和升级

2.5 高可用设计

具有高可用的系统必须避免单点故障，以防由于某个节点故障而导致整个系统不可用。

2.5.1 RES01 冗余

2.5.1.1 RES01-01 应用组件高可用部署

应用系统内的所有组件均需要高可用部署，避免单点故障。

- **风险等级**

  高

- **关键策略**

  应用系统内各组件需要根据其具体能力，采用不同的高可用部署方案：

  - 使用原生高可用实例：当云服务既支持单节点资源，又支持主备或集群资源时，应用的关键节点应使用主备或集群资源，如CCE高可用集群、RDS主备实例、DDS集群、DCS主备或集群实例等。对于运行在CCE集群上的工作负载，也需要配置多个，以避免单个节点故障就导致业务中断。

- 单节点实例通过多实例实现高可用：当云服务只支持单节点发放，则需要应用层来实现多个节点之间的主备或负载均衡，如ECS实例，用户可以通过构建ELB+多ECS实例，来实现无状态业务在多实例之间的负载均衡和自动切换，或从应用层实现两个ECS实例的主备等。
- 硬件依赖实例从应用层实现高可用：当ECS使用本地硬盘、直通FPGA、直通IB网卡等物理服务器强相关的硬件资源时，当硬件故障时会导致ECS故障，且无法通过虚拟机HA功能自动恢复；针对此类问题，需要应用系统在设计时就必须要预料到偶发故障，尽可能避免使用，若必须用时需要从应用层来实现高可用，以便在所依赖的硬件故障时业务能快速恢复。
- 虚拟机HA：当ECS不依赖于特殊资源时，可以支持虚拟机故障自动恢复功能，在其所在物理服务器故障的情况下，可以自动在其他物理服务器上重启；对于部署在这种ECS中的工作负载，需要支持虚拟机重启后业务自动恢复的功能，并能容忍虚拟机HA期间业务处理性能短暂下降或中断。

对已部署的应用系统，改造为支持高可用能力的实施步骤：

a. 确定应用系统的关键组件；所谓关键组件是指一旦故障，会导致整个应用系统或其中的关键功能受损。

b. 针对关键组件，检查其高可用能力，即在其故障的情况下，是否能自动故障转移，进行业务恢复。

c. 针对未支持高可用的关键组件，进行如下优化处理：

- 若云服务实例为单节点实例，如ECS，则通过申请多个实例承载相同业务，并利用ELB实现负载均衡和自动故障切换，或由应用层实现多实例的自动故障切换能力，来实现高可用。
- 对于不依赖于特殊资源的ECS，支持故障自动恢复功能，在ECS所在物理服务器故障的情况下可以自动在其他物理服务器上重启；对于部署在这种ECS中的工作负载，需要检查ECS重启后业务是否能自动恢复。
- 对于依赖特殊资源的ECS，如本地盘、直通FPGA卡、直通IB卡等，不支持故障自动恢复，针对此类ECS需要检查是否可以替换为不依赖于这些特殊资源的ECS，以提高ECS的可用性。
- 对于ECS、BMS、MRS等实例，在使用本地盘时，由于磁盘存在使用寿命上的限制，长时间使用后出现故障的概率会比较高，需要避免使用，而尽可能使用具有高可用能力的EVS磁盘；若必须使用时，则建议使用RAID提升本地盘的可用性，并从应用层实现高可用，以便在一个实例故障时，应用可以自动故障切换和恢复业务。

- **相关云服务和工具**
  - 弹性云服务器 ECS
  - 裸金属服务器 BMS
  - 弹性负载均衡 ELB
  - 云容器引擎 CCE
  - 文档数据库服务 DDS
  - 分布式缓存服务 DCS
  - MapReduce服务 MRS

2.5.1.2 RES01-02 应用组件多位置部署

应用组件需要部署在多个数据中心，以避免单个数据中心故障而导致业务中断。

- **风险等级**

高

- **关键策略**

  可根据不同需求，将应用的数据和资源部署在多个位置：

  - 应用多AZ部署：应用应尽可能部署在多个可用区，避免由于单个可用区故障而导致所有业务中断。
  - 应用多Region部署：对于可用性要求高的应用系统，需要考虑多Region部署，避免由于单个Region故障而导致所有业务中断。
  - 在多AZ部署能满足需求的情况下，应优先使用多AZ部署。大多数工作负载的可用性目标都可通过在单个Region内多 AZ 部署来实现，只有工作负载具有极高的可用性要求或者其他业务目标时，才考虑多Region架构。

2.5.1.3 RES01-03 云服务器反亲和

应用内相同业务的ECS需要分散到多台物理服务器，避免运行到同一台物理服务器上，当发生这种情况时，可能会由于一台物理服务器故障而导致业务中断。

- **风险等级**

  高

- **关键策略**
  - 针对多个承载相同业务的ESC，需要配置主机组反亲和，从而可以将相同业务的ECS调度到不同物理服务器上，以避免由于单台物理服务器故障而导致所有业务不可用的场景。
  - 若ECS通过AS进行弹性伸缩时，则需要AS配置云服务器组反亲和，以避免AS自动创建的ECS运行在同一个物理服务器上。
  - 若CCE集群节点或节点池采用弹性云服务器ECS时，建议配置云服务器组反亲和，以避免CCE集群中的ECS节点运行在同一个物理服务器上。
- **相关云服务和工具**
  - 弹性云服务器 ECS：云服务器组
  - 弹性伸缩服务 AS
  - 云容器引擎 CCE

2.5.2 RES02 备份

对于应用系统中的重要数据，需要提供备份功能，以便在病毒入侵、人为误删除、软硬件故障等场景，能够快速将数据恢复到备份点。

由于容灾通常对数据采用实时复制且没有多备份点，在主数据被误删或误改的情况下，错误数据会同步到备端，从而无法达到数据备份的效果，因此通常不能使用容灾来代替备份。

备份恢复时的RPO指标（即数据丢失量），与最近一个备份时间点相关；不同类型的数据，允许丢失数据量可以不同，即RPO不同；为了保证数据备份的RPO目标，需要采用定期自动备份，而不要依赖人工进行手工备份。

2.5.2.1 RES02-01 识别和备份应用中所有需要备份的关键数据

不同数据的重要性不一样，针对应用系统内的所有数据，需要明确其重要性及对应的RPO/RTO指标要求。比如对于重要数据，通常允许数据丢失的时间会比较少，从而需要更频繁的备份；对于一般的数据，允许数据丢失的时间比较长，可以使用较低的备份频率；对于一些不重要的数据，其数据丢失对业务没有影响，则不需要进行备份。

- **风险等级**

  高

- **关键策略**

  - 识别应用系统中的所有数据。数据可以存储在多种资源中，如ECS/BMS中的卷、RDS/DDS等数据库、SFS文件系统、OBS对象存储等。
  - 根据重要性对数据进行分类。应用系统内的不同数据具有不同的重要程度，对备份的要求也不同；如对一些重要数据，RPO要求接近0，需要实时备份；而对另外一些数据，重要性不高，可以容忍数据丢失，可以不做备份；此外还存在一些比较重要的数据，数据丢失的容忍程度各有不同，需要设计不同的备份策略。
  - 针对需要备份的数据设计备份方案以满足其RPO/RTO指标要求。

2.5.2.2 RES02-02 自动数据备份

对于需要备份的数据，可根据该数据的RPO指标要求，设置定期备份策略进行自动备份。

- **风险等级**

  高

- **关键策略**

  使用华为云备份服务或第三方备份软件对数据进行备份，并可根据RPO要求设置自动备份频率。CBR云备份服务可对ECS/BMS/EVS/SFS Turbo以及文件目录等进行备份；大多数云服务，如RDS、DDS、DCS等具备原生的创建备份功能；云商店也有不少备份软件可以支持各种数据的备份。

  华为云云服务提供了备份工作负载数据的功能，典型的备份有：

  - 云备份CBR服务：CBR提供对磁盘（EVS）、服务器（ECS、HECS、BMS）基于快照的备份和恢复能力，SFS Turbo文件系统备份，云服务器部署的MySQL或SAP HANA等数据库备份，以及云上同步和管理线下备份软件OceanStor BCManager和VMware虚拟机的备份数据。CBR支持一次性备份和周期性备份两种配置方式。目前备份时间只支持整点，可以同时选择多个整点进行备份，即最小RPO=1小时，用户需要根据数据重要性选择合适的备份周期。
  - 数据库自动备份：RDS、DDS、GaussDB等数据库服务提供了缺省自动备份功能，实例每5分钟自动进行一次增量备份，以保证数据库的可靠性。
  - DCS备份：DCS服务针对非单机实例提供了自动备份和手工备份功能，建议设置自动备份策略进行备份。

  此外，用户也可使用第三方备份软件进行备份。

  华为云中云服务的数据备份到OBS存储中，可高度保障用户的备份数据安全。

- **相关云服务和工具**

  - 云备份 CBR
  - 云数据库 RDS
  - 分布式缓存服务 DCS

2.5.2.3 RES02-03 定期进行备份数据恢复

通过定期恢复测试，可以验证备份数据的完整性与恢复处理过程是否可用，且数据丢失时间以及恢复时间符合数据的RPO与RTO指标要求。

- **风险等级**

  高

- **关键策略**

  - 定期执行备份数据恢复，以验证备份的完整性。
  - 为了避免备份恢复对生产业务造成影响，可以构建一个测试环境，并使用已有的备份数据进行恢复处理。

  华为云云服务提供了手工恢复功能，用户可定期执行恢复操作，以进行恢复测试。

- **相关云服务和工具**

  - 云备份 CBR
  - 云数据库 RDS
  - 分布式缓存服务 DCS

2.5.3 RES03 跨 AZ 容灾

为了预防单可用区故障，可借助华为云多可用区（Availability Zone，简称AZ）能力，应用可以用较小成本来完成容灾架构部署。应用系统可设计为使用分布在多个可用区中的资源池，并利用云服务实例本身具备或应用自身支持的跨AZ数据复制与切换能力，在多个AZ之间复制数据、负载均衡和跨AZ故障切换，从而使应用系统具备应对可用区故障的能力。

2.5.3.1 RES03-01 集群跨 AZ 部署

应用内所有组件均采用跨AZ容灾部署，以避免单AZ故障时业务中断。

- **风险等级**

  高

- **关键策略**

  - 云服务实例具备跨AZ高可用实例时，优先使用云服务实例自身的跨AZ高可用实例。
  - 云服务实例只支持发放单AZ实例，不支持跨AZ高可用实例时，需要借助其他云服务或应用层实现跨AZ容灾；以ECS为例：

    - 对于无状态ECS实例，可利用AS弹性伸缩服务的跨AZ伸缩能力，或ELB跨AZ负载均衡能力，实现跨AZ高可用，在一个可用区故障时能自动快速切换。
    - 对于有状态ECS实例，或BMS实例，建议从应用层实现跨AZ容灾，支持跨AZ自动切换或通过容灾管理工具实现自动化容灾切换，减少灾难发生时的人工操作。

  对于已部署的应用系统改造为跨AZ实例的实施步骤：

  a. 确定应用系统的关键组件；所谓关键组件是指一旦故障，会导致整个应用系统或其中的关键功能受损。
  b. 针对关键组件，检查其跨AZ高可用能力，即在一个AZ故障的情况下，是否能自动故障转移到另外一个AZ，进行业务恢复。
  c. 针对未支持跨AZ高可用的关键组件，可进行如下优化处理：

- 若云服务实例支持跨AZ高可用实例且支持由单AZ高可用实例改造为跨AZ高可用实例，如RDS、DDS、DCS实例，则直接原地由单AZ实例改造为跨AZ实例；
- 若云服务实例支持跨AZ高可用实例但不支持由单AZ高可用实例改造为跨AZ高可用实例，如独享ELB、CCE集群、DMS、OBS桶等，则需要新申请跨AZ高可用实例替换原来的单AZ高可用实例。
- 若云服务实例为单节点实例，如ECS，则通过申请多个AZ的多个实例承载相同业务，并利用跨AZ的ELB实现跨AZ的负载均衡和自动故障切换，或由应用层实现跨AZ多实例的自动故障切换能力，来实现跨AZ高可用。

- **相关云服务和工具**

  华为云大部分云服务支持创建多可用区实例，可实现在一个可用区故障时能自动快速切换，不影响实例对外提供服务，如ELB负载均衡、AS弹性伸缩、CCE容器集群、DCS实例、DMS消息服务、RDS数据库、GaussDB数据库等。

2.5.3.2 RES03-02 跨 AZ 数据同步

针对有状态业务，需要进行跨AZ的数据同步，以便在一个AZ故障的情况下，数据不丢失；对于无状态业务不涉及。

- **风险等级**

  高

- **关键策略**
  - 当应用组件对应的云服务实例支持跨AZ高可用实例时，可采用云服务实例自身的跨AZ数据同步；如RDS数据库、DCS实例、OBS桶等。
  - 当应用组件对应的云服务实例不支持跨AZ高可用实例，但提供了同步服务进行跨AZ数据同步时，可利用该服务进行跨AZ数据同步；如存在有状态数据的ECS实例不支持跨AZ高可用，但可通过SDRS服务进行跨AZ数据同步。
  - 当应用组件对应的云服务实例不支持跨AZ高可用实例，且不支持跨AZ数据同步或不使用跨AZ数据同步服务时，则需要由应用层进行数据复制；如存在有状态数据的BMS实例。

- **相关云服务和工具**
  - 存储容灾服务 SDRS
  - 弹性云服务器 ECS
  - 云数据库 RDS
  - 分布式缓存服务 DCS
  - 对象存储服务 OBS

2.5.3.3 RES03-03 对接容灾仲裁，支持自动切换

针对有状态的主备类型业务，在跨AZ部署并支持自动切换时，需要对接容灾仲裁，以避免出现双主或双备，从而在AZ间链路中断的情况下，业务能自动切换到一个AZ提供服务而不受影响；对于集群类业务不涉及。

- **风险等级**

  高

- **关键策略**
  - 面向有状态主备类型业务提供容灾仲裁，站点间链路中断不双主，不破坏数据完整性。

- 应用内所有相关组件对接一致性仲裁，在链路中断的情况下所有组件均能切换到同一个站点，实现端到端的业务可用性

2.5.3.4 RES03-04 支持容灾管理

提供容灾管理功能，实现容灾状态及RPO监控，及异常场景下的业务切换。

- **风险等级**

  高

- **关键策略**
  - 实时监控容灾状态，了解容灾运行状态。
  - 支持应用级数据校验，比较AZ间数据同步差异，监控及PO指标。
  - 典型确定性故障场景下自动容灾或切换，无需人工接入，业务不受影响，满足RPO/RTO指标。
  - 典型亚健康故障场景，支持业务降级或主动切换，业务不持续受损。
- **相关云服务和工具**
  - 多活高可用服务 MAS

2.5.4 RES04 跨 Region/跨云容灾

为了预防区域级灾难发生，或业务跨云容灾需求，需要构建容灾系统提供较为完善的数据保护与灾难恢复能力，以便在站点级灾难发生时，可以保证生产系统的数据尽可能少的丢失，业务系统能在最短时间内由灾备中心接替，恢复业务系统的正常运行，将损失降到最小。

对于跨Region容灾场景，应用系统可在多个Region中部署，并将数据从一个Region复制到另一个Region，以便在发生地区级服务中断或数据丢失时可进行灾难恢复。

对于跨云容灾场景，当应用系统已部署在IDC或其他云中，可以在华为云中另外部署一套系统并将数据从IDC或其他云复制到华为云中，以便在发生整IDC或整朵云服务中断或数据丢失时可以进行灾难恢复。

2.5.4.1 RES04-01 定义应用系统的容灾目标 RPO 与 RTO

在进行容灾设计前，需要根据应用系统的重要性，明确其容灾目标，通常以RPO和RTO指标来定义：

- RPO：允许的数据丢失量，与数据的周期性复制周期或连续性复制延时相关。
- RTO：允许的业务恢复时长，即业务中断时长，与灾备端业务的部署与切换方式相关。
- **风险等级**

  高

- **关键策略**

  不同的业务系统重要性不一样，针对应用系统内的各种业务，需要明确其重要性及对应的RPO/RTO指标要求。比如对于核心业务，通常需要保障业务的连续性，允许业务中断的时间会比较少，从而需要保障故障场景下的业务快速恢复，可采用双活/多活容灾；对于重要业务，允许一定的业务中断时间，可采用主备容灾；对于一般业务，允许中断的业务时间可达到天级，则可采用远程备份；对于一些不重要的业务，其业务中断对外部客户没有影响，则不需要进行容灾。

2.5.4.2 RES04-02 部署容灾系统以满足容灾目标

针对不同应用系统的容灾目标，需要综合考虑中断概率、容灾成本等因素，来决定采用什么样的容灾方案来实现这些目标。

- **风险等级**

  高

- **关键策略**

  面向跨Region/跨云容灾场景，可基于不同的可用性目标要求，采用不用的容灾方案，如远程备份、主备容灾、双活容灾等，其中生产站点根据场景不同可能为其他云或IDC或华为云Region：

  - 远程备份：生产站点内的重要数据，备份到异地华为云灾备Region，当生产站点发生灾难时，需要在异地灾备Region新部署一套业务系统并使用最新备份数据恢复数据，并恢复业务。
  - 主备容灾：生产站点与华为云灾备Region各部署一套业务系统，并将生产站点的重要数据异步复制到灾备Region；平常只有生产站点提供业务，当生产站点发生灾难时，将灾备Region提升为主，并将业务流量切换到灾备Region并由其提供业务。
  - 双活/多活容灾：生产站点与华为云灾备Region各部署一套业务系统，并将各自站点的重要数据异步复制到其他站点；每个站点都同时提供业务，通过全局负载均衡器进行流量分发；当一个站点发生灾难时，则将业务流量全部分发到其他站点来接管其业务。

  以跨Region主备容灾为例，对于已在一个Region部署应用系统后，增加支持跨Region主备容灾能力的实施步骤建议如下：

  a. 选择另一个Region作为灾备Region，部署一套相同的应用系统，包括工作负载、数据库实例等。
  b. 针对应用系统内的关键数据，利用云服务或应用系统自身实现跨Region的数据复制。
     - 若云服务实例支持跨Region容灾，则配置生产站点与灾备Region之间的复制，如对于RDS数据库实例，需申请DRS实例对主Region与灾备Region的数据库进行实时复制；对于OBS桶，需要配置主Region中的OBS桶到灾备Region中OBS通的复制。
     - 若云服务实例不支持跨Region容灾，但数据比较关键，则需要应用层实现跨Region的数据双写，以进行数据同步。
  c. 接入侧主Region与灾备Region各自申请外部IP，并通过DNS域名解析到主Region，在主Region故障时，将DNS域名对应IP地址修改为灾备Region中的外部IP。
  d. 申请MAS多活高可用服务，进行容灾编排，以便在灾难场景快速主备切换恢复业务。

- **相关云服务和工具**
  - 云备份 CBR：支持跨区域复制与恢复
  - 数据复制服务 DRS：支持RDS for MySQL、GaussDB for MySQL等数据库的实时灾备，支持跨Region/跨云容灾场景
  - 对象存储服务 OBS：支持跨区域复制与双活

2.5.4.3 RES04-03 容灾恢复过程自动化

由于容灾恢复场景涉及容灾站点的业务恢复、数据库的主备切换、业务到容灾站点的流量切换等，恢复过程比较复杂，因此需要提供容灾管理功能，实现容灾状态及RPO监控，以及灾难场景下的一键式自动切换，减少人工干预。

- **风险等级**

  高

- **关键策略**
  - 实时监控容灾状态，了解容灾运行状态。
  - 支持应用级数据校验，比较AZ间数据同步差异，监控及PO指标。
  - 灾难场景下的一键式自动切换，减少人工干预，满足RPO/RTO指标。
  - 支持容灾恢复流程编排、容灾演练等功能。
- **相关云服务和工具**
  - 多活高可用服务 MAS

2.5.4.4 RES04-04 定期进行容灾演练，以检查恢复能否满足容灾目标

通过定期的容灾演练，可以验证灾备系统是否可用，且数据丢失时间以及恢复时间符合数据的RPO与RTO指标要求。

- **风险等级**

  高

- **关键策略**
  - 每年至少进行一次容灾演练；通过演练可提升操作人员的熟练程度。
  - 演练期间需要对恢复过程计时，以确定应用系统的RPO与RTO目标能否满足。
  - 演练期间可检查灾难恢复计划执行顺序及恢复时间并进行优化。
- **相关云服务和工具**
  - 多活高可用服务 MAS

2.5.5 RES05 网络高可用

应用系统对外或对内通信都依赖于网络，一旦网络异常将会导致业务中断，因此网络架构的高可用及容灾能力至关重要。在进行网络设计时，需要充分考虑应用系统对内和对外的网络连接、IP地址管理和域名解析等。

华为云中网络高可用主要涉及三个场景：

- 公有云网络：构建应用系统相关的公网网络连接的高可用，可减少由于网络连接中断而导致的业务中断。
- 混合云网络：对于自建本地数据中心（IDC）或使用其他云的用户，基于业务发展需要将部分业务部署到华为云时，将涉及到混合云网络互连；应用系统跨云部署时（如跨云主备容灾或双活），需要构建高可用的混合云网络连接，以减少由于网络连接中断而导致的业务中断。
- 云上网络之间访问：当业务系统涉及到多个部门或业务团队时，一般会使用多个VPC进行业务隔离，不同团队和部门之间需要相互访问，将会涉及不同VPC之间的网络连接。

2.5.5.1 RES05-01 网络连接高可用

应用系统对外提供服务时，需要确保对外网络连接的高可用，避免单个网络连接中断而导致业务不可用。

- **风险等级**

  高

- **关键策略**
  - 网络链路冗余：网络连接需要支持多路径，以实现高可用能力，以避免在一条网络路径中断的情况下，业务能切换到其他路径继续通信。
  - 网络链路快速倒换：需要定期检查网络链路的连通性，但检测到失败时需要尽快切换到正常路径。

  公有云组网场景可通过多EIP 弹性IP及DNS域名解析实现网络连接的高可用；对可用性要求较高的场景，需要支持智能DNS功能，能对EIP进行异常监控和自动切换；此外DNS自身也需要冗余容错，避免由于DNS故障而导致域名解析失败，业务中断。

  混合云组网场景链路冗余与倒换方案：
  - 双DC专线冗余：用户数据中心与华为云VPC之间采用两条DC专线互通；其中两条物理专线接入同区域的两个华为云专线接入点，并通过BGP路由协议接入同一个VPC，用户可设置虚拟接口的优先级以决定业务的主备链路。具体的方案参见“**用户通过双专线双接入点BGP协议访问VPC**”。
  - 双VPN冗余：用户数据中心与华为云VPC之间采用两条VPN连接保证可靠性；当其中一条VPN链接故障时，系统可以切换到另一条VPN连接，保证网络不中断。两条VPN连接可以是双活或主备部署。具体的方案参见“**通过VPN实现云上云下网络互通（双活模式）**”与“**通过VPN实现云上云下网络互通（主备模式）**”。
  - DC专线/VPN主备：用户数据中心与华为云VPC之间同时部署DC专线和VPN两条网络链路，互为主备，并通过企业路由器，可以实现DC和VPN主备链路的自动切换，不需要手工切换双联路，不仅避免业务受损，同时降低维护成本。具体的方案参见“**通过企业路由器构建DC/VPN双联路主备混合云组网**”。

- **相关云服务和工具**
  - 云专线 DC
  - 虚拟专用网络 VPN

2.5.5.2 RES05-02 避免暴露不必要的网络地址

网络地址对外暴露时，可能会引入安全风险，需要避免暴露不必要的网络地址。

- **风险等级**

  高

- **关键策略**
  - 通常对外网络地址需要尽可能集中管控，避免分散暴露，如使用网络服务ELB弹性负载均衡、公网NAT网关、Web云防火墙等作为公网访问入口。
  - 对外的IP地址需要通过安全组、NAT等限制网络端口访问，减少安全风险。

- **相关云服务和工具**
  - 虚拟私有云VPC：安全组

- 弹性负载均衡器 ELB
- NAT网关 NAT
- Web云防火墙 WAF

2.5.5.3 RES05-03 不同流量模型业务的网络共享带宽隔离

不同流量模型业务共享网络带宽享时，可能会导致流量抢占，相互影响，一个业务流量突然可能会导致其他业务不可用。

- **风险等级**
  高
- **关键策略**
  - 相同流量模型的业务，可共享网络带宽，带宽需要满足所有共享业务的需求
  - 不同流量模型的业务，为了避免相互干扰，建议使用各自独立的共享带宽实例
  - 不同特性的业务，建议使用各自独立的域名隔离。

2.5.5.4 RES05-04 预留 IP 资源以便扩展及高可用

云上网络需要满足可扩展以及高可用需求，以便在云上资源弹性伸缩或业务扩展时，有足够网络资源支撑业务发展。

- **风险等级**
  高
- **关键策略**
  云上网络规划设计应满足以下原则：
  - 针对每个Region，根据业务需要规划不同的VPC，每个VPC使用独立的地址空间；并需要预留IP地址空间用于新建VPC。
  - 针对每个VPC中，需要根据业务需要规划子网和IP地址空间；并需要预留IP地址空间用于新建子网。
  - 针对每个子网，需要预留IP地址空间用于网络扩容。
  - 当涉及与其他网络（如VPC、IDC或其他云）互连时，需要确保IP地址空间不重叠。

2.6 故障全面检测

高可用性系统必须具有完善的故障检测能力，以确保能够快速发现那些可能导致故障的事件、显示正在发展的故障、激活的故障，以及潜在的故障的事件。在几乎所有情况下，故障检测能力都是故障恢复的前提。

2.6.1 RES06 故障检测

2.6.1.1 RES06-01 故障模式分析

故障模式分析是在系统分析和设计过程，通过对各组成单元潜在的各种故障模式及其对产品功能的影响进行分析，并把每一种潜在故障模式按它的严酷度予以分类，找出单点故障和产品的薄弱环节，提出可以采取的预防改进措施，以提高产品可靠性的一种设计方法。

当应用系统部署在华为云中时，华为云提供了基础设施的故障管理，应用系统可减少对机房、电力、环境、计算服务器、存储设备、网络交换机等基础设施的故障模式的检测和恢复处理，但仍需考虑这些基础设施故障对应用系统的影响及对应的恢复措施，如机房发生灾难(AZ或Region级灾难)、计算服务器故障/重启、使用本地硬盘时硬盘故障/亚健康、网络通信中断/丢包等。而对于应用自身相关的故障模式，如软件系统类、数据类、通信类、负荷过载、人因差错等类型的故障，更需要充分分析并提供检测和恢复措施。

- **风险等级**

  高

- **关键策略**

  针对每种故障模式，分析其发生的频率以及造成的影响，以确定严酷度等级。对于存在单点故障的组件对应的故障模式，严酷度必须设置为高。云服务通用的故障模式有：CPU过载、内存过载、磁盘使用率过高、数据故障(被误删等)、AZ故障、Region故障等。

  a. 定义严酷度类别

     严酷度是度量故障给系统造成的最坏潜在后果，一般分为四个等级：Ⅰ类（严重）、Ⅱ类（较严重）、Ⅲ类（一般）、Ⅳ类（轻微）。

     - I类：这种故障会导致整个系统崩溃或主要功能受到严重影响；
     - II类：这种故障会导致系统主要功能受到影响、任务延误的系统轻度损坏或存在较大的故障隐患；
     - III类：系统次要功能丧失或下降，须立即修理，但不影响系统主要功能实现的故障；
     - IV类：部分次要功能下降，只须一般维护的，不对功能实现造成影响（一般告警或指示灯故障等）。

     其中，I～II类故障通常称为重大故障，也即“单点故障”，它们的区别主要是I类故障可能涉及到安全性问题，或者I类故障是所有/大部分功能丧失。II类故障指主要功能受影响。III类故障可简单理解为需要尽快修复的故障。

     通常来说，当一个故障不能被检测出来时，会认为这是一个故障“隐患”，相应的故障严酷度级别上升一级。

  b. 标识系统中的所有组件及功能模块

     明确应用系统涉及的所有组件，以及外部依赖项，如提供者、第三方服务等。

  c. 识别故障点

     对于每个组件，标识可能发生的潜在故障。单个组件可能具有多种故障模式，需要针对不同故障模式分别分析。故障模式的种类需要尽可能完备，若出现遗漏，可能导致该故障在设计中不被考虑，而没有进行监控和恢复处理。

  d. 故障影响范围分析(爆炸半径)

     针对每种故障模式，分析其发生的频率以及造成的影响，以确定严酷度等级。对于存在单点故障的组件对应的故障模式，严酷度必须设置为高。云服务通用的故障模式有：CPU过载、内存过载、磁盘使用率过高、数据故障(被误删等)、AZ故障、Region故障等。

  e. 提供故障检测和缓解措施

  f. 针对每种故障模式，需要分析如何检测和恢复，提出改进建议措施，并在系统复杂度和成本之间进行综合考虑，优先解决严酷度高的故障模式。

- **相关云服务和工具**
  - 云运维中心 COC：支持故障模式管理。

2.6.1.2 RES06-02 面向所有故障进行检测

针对所有故障场景，都需要能自动检测，以便及时发现和恢复故障。

- **风险等级**

  高

- **关键策略**
  - 所有故障都必须有检测。
  - 支持按不同维度进行故障检测，如Region、AZ、服务、方法、实例或容器ID等，检测维度与故障恢复方式对齐。
  - 检测到故障后需及时告警或自动恢复。

  针对具体故障进行检测时，根据检测的类型通常可以分为资源检测、功能检测和业务检测。
  - 资源检测：云环境中一般指虚拟化后的物理硬件资源及其对应的软件资源，具体包含CPU、内存、网络和磁盘资源等。
  - 功能检测：对组成产品系统的各个内部模块对象进行检测的过程，确定模块功能是否满足设计的需求。当产品系统的功能发生故障时，对外的呈现即为功能输出和预期不一致。在产品上线之前，通过功能相应接口，开发者和测试人员需要多次检测以保证模块功能的正确性。功能检测可以使用传统日志跟踪技术、调用链技术来进行检测，如华为云APM。
  - 业务检测：模拟用户的业务操作过程，获得完成业务的操作过程性能数据和操作结果数据；业务检测使用拨测技术来完成检测，由于拨测需要占用网络资源，对于长周期拨测，一般选择在空闲时间段进行，属于抽样检测，而如果是短周期拨测（如5分钟周期），则可例行进行；与功能检测的联系是，业务检测也可以采用调用链来完成。

  故障检测方法根据类型有很多种，下面是一些在高可用性系统中常用的故障检测方法。
  - 数值范围检查：在大多数应用中，一个操作的结果必须处于某个范围之内。对这些边界条件可以进行一些测试来验证数据是否满足预期要求。
  - 数据完整性检查：每当数据被从一个单元传递给另一个单元时，该数据可能会被破坏。对于在硬件单元间传递的数据尤其如此。然而，由于软件层可以隐藏本地内存传送和跨远程链路的传送间的差异，因此需要在多个点进行数据完整性检查。可以采用很多方法来验证数据的完整性，其中大多数方法都依赖于冗余或者包含在数据中的摘要信息。有些方法采用足够的冗余，不仅能检测错误，而且能纠正错误。但大多数方法中都只包括足够的额外信息来检测数据是否有效。典型的方法如奇偶校验和CRC（循环冗余校验）。
  - 比较测试：当系统具有冗余时，可以使两个系统并行进行计算，然后对结果进行比较，如果结果不匹配则认为发生了故障。这种概念也称为表决。比较可以在系统的任何层次上进行，包括在一条内存总线上的cycle by cycle的比较，到最终发送到网络上结果的比较。
  - 时间检测：时间检测是故障检测的一种简单形式。如果一个事件预期应在某个时间段内发生，而却没有在该时间段发生，就检测到了一个故障。时间检测的一种特殊方法通常称为心跳方法。它采用以某个规定的周期频率执行的某些类型的消息握手。该技术可以用于验证单元或子系统是否仍然能够维持某些等级的功能。

2.6.1.3 RES06-03 支持亚健康检测

系统内组件有可能完全故障，也有可能处于亚健康状态；亚健康是指系统整体业务未超标，但系统中局部实例业务超标。亚健康更多是个相对概念，相对历史表现的统计，或相对系统整体。因此针对亚健康的检测和判断有所不同。当处于亚健康状态时，系统也需要及时进行隔离或恢复处理，避免对业务造成影响。

- **风险等级**

  高

- **关键策略**

  亚健康检测通常用于根据亚健康症状来预测系统故障，典型的例子是内存泄漏，内存泄漏往往不会立刻导致系统失效，系统首先会因为Swap Memory不足变得运行缓慢，消耗内存量持续增加，因此通过监控实例内的内存占用率，在超过阈值的情况下及时告警，人工介入迅速恢复，可避免造成业务中断。

  典型的亚健康场景有：通信链路丢包/错包、硬盘性能下降、CPU/内存过载等，当应用系统内组件出现亚健康时，可能会导致应用系统对外业务成功率下降。

  由于亚健康并非故障，因此针对亚健康的检测一般是针对业务监控指标设置阈值，当指标超过阈值时进行告警和恢复处理。

2.6.2 RES07 监控告警

应用系统需要监控，以便维护人员能快速识别系统运行现状及问题。

2.6.2.1 RES07-01 定义关键指标与阈值并监控

对资源进行监控时，需要先定义资源的关键指标以及对应的阈值，以便快速有效的发现业务表现和系统状态，以便在异常状态下尽早干预恢复，或定位改进系统缺陷。

- **风险等级**

  中

- **关键策略**

  - 关键指标需要与系统内工作负载的关键性能指标相关，并能确定为系统性能下降的早期警告信号，如系统处理的API数量及成功率，相比CPU利用率、内存利用率等基础指标，能更真实的指示系统性能问题。
  - 从可用性保证出发，结合有效性和简化，建议应用系统至少从业务状态、服务状态、资源状态三个层面进行监控。根据业务规模，可以使用CES服务（侧重在I层服务）或AOM/APM服务（侧重在P层业务），也可以借助Prometheus、Zabbix、Zipkin等部件自行搭建，使用Grafana等部件进行界面展示和时序对齐。

  1、业务监控

  以下4个黄金指标，是针对大量分布式监控的经验总结，可以作为业务监控的参考，包括：

  - 延迟：注意需要区分请求成功的延迟和请求失败的延迟。
  - 流量：对系统业务负荷的监控。
  - 错误率：注意区分显示失败（如HTTP 500错误）和隐式失败（如HTTP 200中包含了错误内容）。
  - 饱和度：侧重在对系统中最为受限的瓶颈资源的监控。

  对于基于Java的应用系统，华为云用户可使用APM服务实现基于调用链的业务延迟和错误率监控。函数服务FunctionGraph、微服务引擎CSE提供了流量、延迟和

错误率监控能力。基于API网关暴露接口的应用，可使用APIG服务提供的流量、延迟和错误率监控能力。如果云服务现有能力不能满足系统要求，用户也可以自行埋点或基于Zipkin开源框架实现调用链跟踪、延迟和流量监控。

2、服务监控

由于服务实例的冗余配置和应用系统的容错保护，业务指标正常并不意味着服务实例状态一定正常。例如，在配置了ELB的虚拟机集群中，ELB会主动隔离异常节点，虽然业务会在正常节点上分担，但应用系统实际已损失了部分处理容量。因此，云服务状态监控必不可少。

云服务具体指标因功能特性而异。站在功能提供者的层面，通常同样需要重点关注延迟、流量、错误率、利用率等指标。此外，服务实例的动态伸缩、过负荷控制、故障自愈或迁移等可靠性关键事件也是服务健壮性的表征，如有异常需要预先干预。关键事件监控可以使用CTS服务，或自行搭建。

CES服务支持ECS、EVS、OBS、VPC、ELB、AS等IaaS服务，以及RDS数据库，DCS、DMS等高可用中间件的主要指标监控，支持用户上报自定义监控指标。如果用户自行搭建监控系统，也可以通过CES SDK获取指定服务的监控指标。

AOM服务提供了微服务应用和节点的关键指标监控能力。云容器工作负载关键指标在CSE服务中查看。函数服务关键指标在FunctionGraph控制台中查看。

3、资源监控

资源监控通常用于识别资源瓶颈分析系统性能问题。对应用系统资源进行监控时，需要先定义资源的关键指标以及对应的阈值，以便快速有效的发现业务表现和系统状态，以便在异常状态下尽早干预恢复，或定位改进系统缺陷。

关键指标需要与系统内工作负载的关键性能指标相关，并能确定为系统性能下降的早期警告信号，如系统处理的API数量及成功率，相比CPU利用率、内存利用率等基础指标，能更真实的指示系统性能问题。

常用USE方法（Utilization Saturation and Errors Method）对资源监控，包含：

- 使用率Utilization：覆盖系统资源，包括但不限于CPU、内存、网络、磁盘等。
- 饱和度Saturation：针对资源的饱和度，如CPU队列长度，注意与业务监控的黄金指标相区分。
- 错误Errors：资源处理错误，如网络丢包率等。

CES主动监控提供了虚机细粒度的监控能力，其他服务监控指标也不同程度涉及到资源使用率和错误监控。如果云服务现有能力不能满足系统要求，用户可使用CES或AOM服务的自定义指标监控能力。用户若自行搭建监控系统，需要覆盖主机资源、网络设备和Apache、Java、MySQL等第三方组件，开源的Zabbix是常见选择。

- **相关云服务和工具**
  - 云监控服务 CES
  - 应用运维管理 AOM
  - 应用性能管理 APM

2.6.2.2 RES07-02 日志统计监控

应用系统需要收集日志，在必要时对日志进行统计分析，设置告警规则触发告警，统计分析的内容可以是统计一定时间段内某些关键字出现的次数。

- **风险等级**

  中

- **关键策略**
  - 日志关键字与出现次数阈值需要合理设置，以免监控信息不正确。
  - 日志信息（如关键字或出现频率）发生变化时，需要及时更新告警规则。
- **相关云服务和工具**
  - 云日志服务 LTS

2.6.2.3 RES07-03 监控到异常后发送消息通知

当对应用系统监控发现应用异常后，需要向相应的人员和系统发送实时通知消息和告警，以便及时处理。

- **风险等级**

  中
- **关键策略**
  - 采用实时快捷的消息通知方式，以便相关人员能及时得到消息。
  - 消息发送人员需要涵盖运维人员，以便及时恢复。
  - 运维人员需要有备份，避免单点风险。

  SMN消息通知服务可依据用户需求主动推送通知消息，方式可为短信、电子邮件等。CES、AOM、CTS、APM、LTS等服务均已经对接SMN消息通知服务，在阈值规则发生变化时，可以以邮件或短信等方式通知，以便您在第一时间发现异常并进行处理。
- **相关云服务和工具**
  - 消息通知服务 SMN
  - 云运维中心 COC：支持人员管理、排班管理和通知管理，可以根据通知规则自动将消息发送给要通知的人员。

2.6.2.4 RES07-04 监控数据存储和分析

监控数据包括统计和日志信息，均需要存储并进行生命周期管理，以满足数据监控的保留要求；并定期对其进行分析，以了解系统运行状态和趋势。

- **风险等级**

  中
- **关键策略**
  - 监控数据存储时长需要满足保留要求。
  - 监控数据需要定期分析，以便发现或预测系统故障，减少业务中断。
- **相关云服务和工具**
  - LTS云日志服务：支持日志分析与数据转储

2.6.2.5 RES07-05 端到端跟踪请求消息

端到端跟踪请求消息的处理流程，便于分析和调试问题，并提高处理性能。

- **风险等级**

  低
- **关键策略**

- 消息跟踪需要包含消息处理流程中所有组件，以便跟踪结果完整，从而进行准确分析和定位。

- **相关云服务和工具**
  - 应用性能管理 APM：支持调用链追踪，能够针对应用的调用情况，对调用进行全方面的监控，可视化地还原业务的执行路线和状态，协助性能及故障快速定位。
    - 在查询后的调用链列表中，单击待查看的调用链的链接，查看该调用链基本信息。
    - 调用链详情页面可以查看调用链的完整链路信息，包含本地方法堆栈和相关远程调用的调用关系。
    - 调用链与日志关联，提高用户体验。用户可以从调用链直接跳转LTS查看日志。

2.7 故障快速恢复

当应用系统采用华为云服务的高可用设计时，在云服务实例发生故障后，云服务能自动检测和恢复；但对于应用系统本身的故障，需要应用系统自身进行检测和快速恢复处理，以保证系统能够正常运行，从而提高系统的可靠性和稳定性。

2.7.1 RES08 依赖减少与降级

对于应用系统，需要识别和管理系统依赖项。应用系统设计人员需要维护对其他系统组件的依赖项的完整列表，包括系统内和系统外的所有依赖。

应用系统应尽可能减少关键依赖项，即减少由于该依赖项不可用而导致服务中断的组件。

2.7.1.1 RES08-01 减少强依赖项

系统内组件之间强依赖时，一个组件故障会对其他组件造成直接影响，影响系统可用性。

- **风险等级**

  中

- **关键策略**

  可以通过以下技术将强依赖项转换为非强依赖项：
  - 提高关键依赖项的冗余级别，降低该关键组件不可用的可能性。
  - 与依赖项的通信采用异步消息并支持超时重试，或发布/订阅消息功能将请求与响应分离，以便依赖项从短时故障中恢复。
  - 依赖项长时间无法访问时，应用程序应能继续执行其核心功能，以便将局部故障对整体系统功能的影响减到最小。如所依赖的数据丢失时，应用程序仍能运行，但可以提供稍微陈旧的数据、替代数据，甚至没有数据，应用仍处于可预测和可恢复的状态。
  - 避免启动依赖及循环依赖。若应用系统由于某些原因导致重启时，若依赖于其他依赖项启动或加载关键配置数据，可能会导致应用系统长时间停在启动状态而无法响应外部消息。针对这种情况，应用系统应该先使用缺省配置启动，再检查依赖项的状态或加载最新配置数据，以恢复正常运行。

2.7.1.2 RES08-02 依赖松耦合

系统内组件之间直接访问时，会产生紧耦合关系一个组件的状态变化会对其他组件产生直接影响，从而会导致所有组件的可用性均下降。而采用松耦合架构时，各个组件之间的依赖关系非常弱，它们可以独立地进行修改和扩展，而不影响其他组件；系统更加灵活，易于维护和升级，并且稳定性和可靠性也更强。

- **风险等级**

  中

- **关键策略**
  - 组件之间通过消息队列、消息缓存、负载均衡器等交互（即松耦合关系），可一定程度上屏蔽组件的状态变化，防止对其他组件造成影响
- **相关云服务和工具**
  - 弹性负载均衡服务 ELB
  - 分布式缓存服务 DCS
  - 分布式消息服务Kafka版
  - 分布式消息服务RabbitMQ版
  - 分布式消息服务RocketMQ版
  - 事件网格 EG

2.7.1.3 RES08-03 减少被依赖项故障的影响

被依赖项自身的可用性需要增强，以减少对依赖它的组件的影响。

- **风险等级**

  中

- **关键策略**

  对于被依赖项本身，为减少由于服务故障或运行缓慢对依赖它的组件的影响，需要考虑使用以下技术和原则：
  - 减少被依赖项本身的外部依赖。
  - 优化性能，减少消息响应时延和负载。
  - 使用优先队列，优先处理高优先级用户的请求，以便在流量过载时不影响应用系统的核心功能。
  - 流量过载时支持功能逐步降级。
  - 被依赖项本身的功能受损时，提供缺省处理，以便应用系统仍可继续正常运行；由于缺省处理可能与实际配置有差异，此时需要告警以便通知系统管理员解决问题。

2.7.2 RES09 故障重试

当应用系统部署在云中，虽然云具有一定的高可用和故障自动恢复能力，但对外仍会导致短时间的故障，需要应用系统能针对这种短时间故障进行适配处理，主要是采用重试机制。

云中故障需要重试的典型场景有：

1. 实例主备切换时可能会导致连接中断，如DCS、RDS实例由于某些原因主备切换时，会导致连接中断，需要客户端重试。

2. 实例由于故障重启可能会导致通信中断，如ECS所在物理服务器由于硬件原因故障时，ECS重启或在其他物理服务器中自动恢复，恢复过程中与ECS的通信会中断，需要重试。
3. 实例由于过载导致无法及时响应，需要重试。

2.7.2.1 RES09-01 API 及命令调用需要设计为可重试

在进行重试处理时，API及命令调用会重复发送，服务方会多次重复执行，需要保证重复执行多次的结果不变。

- **风险等级**

  高

- **关键策略**

  应用系统在设计时，应使操作具有幂等性，也就是允许一个操作连续执行两次或多次时，应该与单次调用产生的结果相同，从而保证重试安全；若不支持操作的幂等性，会导致客户端难以重试或重试的处理更复杂。

2.7.2.2 RES09-02 客户端需要根据综合评估是否要重试

当客户端请求超时或收到错误响应时，客户端需要决定是否重试；重试有助于客户端在请求失败时，通过重复消息来获得预期的结果，避免业务失败，但也会消耗更多的服务器时间来获取所需的成功响应。

- **风险等级**

  高

- **关键策略**
  - 请求超时，可能是链路闪断或其他临时性故障导致消息丢失，可以进行重试。
  - 根据错误响应码进行有针对性的重试；对于临时性故障，如错误码指示为系统繁忙时，可等待一段时间后重试，否则无需重试。
  - 请求SDK中内置了消息重试时，客户端无需重复重试。
  - 多层业务栈一般只在源端重试，避免逐层重试。

2.7.2.3 RES09-03 重试需要避免造成流量压力

对于链路闪断等原因导致的临时性故障，客户端进行一定的重试，可取得较好的效果；对于流量过载等原因导致的故障，重试可能会导致情况进一步恶化，因此需要避免这种影响。

- **风险等级**

  高

- **关键策略**

  客户端进行重试处理时，建议：
  - 增加指数回退和抖动方法，以避免对服务端造成流量压力；采用指数回退重试时，每次重试之间的间隔会逐渐延长，并在两次重试之间引入抖动，以随机调整重试间隔，避免同时出现造成重试峰值。
  - 限制最大重试次数或用时，避免由于消息积压而导致流量过载。

2.7.3 RES10 故障隔离

当系统某个单元发生故障时，如果不采取措施，故障可能会大规模扩散，从而造成整个系统失效。故障隔离技术的核心思想是将一个工作负载内的故障影响限制于有限数量的组件内，降低故障影响范围，防止产生级联故障。

通过划分故障隔离域，限制工作负载的影响，可有效进行故障隔离。

2.7.3.1 RES10-01 应用控制平面与数据平面隔离

通常应用的数据平面处理业务，比较重要，可用性要求比较高，而控制平面不直接处理业务，因此其故障时不应该影响业务系统。

- **风险等级**

  高

- **关键策略**
  - 应用控制平面与数据平面隔离，避免控制系统故障影响业务。
  - 数据平面所在业务系统的故障恢复可不依赖控制平面，避免由于控制平面故障而导致业务系统无法恢复。

2.7.3.2 RES10-02 应用系统多位置部署

通过将应用系统部署在多个位置，可以避免由于一个位置的基础设施故障而导致系统不可用。

- **风险等级**

  高

- **关键策略**
  - 将应用系统的数据和资源部署在多个AZ，可避免单个AZ故障影响业务。
  - 对于可用性要求较高的应用系统，可部署在多个Region，避免单个Region故障影响业务。
  - 当多AZ架构可以满足应用可用性需求时，无需采用多Region部署。

2.7.3.3 RES10-03 采用 Grid 架构

采用Grid架构，可将应用系统内的工作负载的故障影响限制在有限Grid业务单元中。

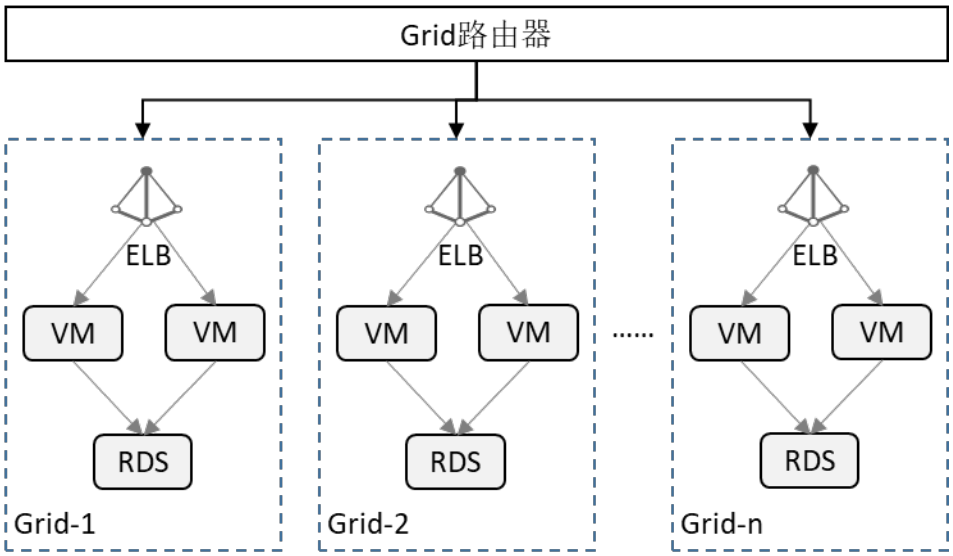
- **风险等级**

  高

- **关键策略**

  应用系统采用多个功能相同的Grid业务单元，每个Grid业务单元具备完整业务功能，处理整个业务负载中的一个子集，不涉及与其他Grid业务单元的交互；在一个Grid业务单元发生故障时，仅影响本Grid业务单元所处理的业务，对其他Grid业务单元没有影响，从而减少爆炸半径。

  应用系统典型Grid架构部署如下：



实施步骤：

a. 确定分区键。选择分区键应考虑：

- 选择分区键必须考虑匹配服务的“粒度”或者考虑以最小的方式跨分区互动。对于多用户系统，可使用用户ID作为分区键；而对于资源为对象的系统，则可以使用资源ID作为分区键。
- 所确定的分区键，必须在所有API或命令中都能直接包含或可通过其他参数间接转换得到，以便能使用该分区键进行分区处理。
- 按分区键进行分区处理时，需要确保对应分区能独立处理业务，尽可能避免或减少与其他分区的交互。

b. 确定分区数量与每个分区的大小，后续还存在增加分区的情况。需要综合考虑：

- 分区数量越多，对应分区会越小，爆炸半径也越小，运维定位简单，可用性高，但由于资源共享利用率低，所需的成本也越高。
- 分区数量越少，每个分区的资源多，更容易适合对资源要求较高的大客户，运维管理简单，且资源利用率越高，所需的成本低。

c. 确定分区映射算法。存在以下一些映射算法供参考：

- 原始除模：即使用分区键对分区数量取模，该算法分布均匀，但是不适配Grid增删场景，一旦增删需要进行业务迁移。
- Range-Hash/Hash：即使用分区键按范围分区后Hash或直接使用分区键Hash，元数据管理相对复杂一些。
- Full-Mapping：全映射，即针对分区键指定Grid，使用全映射会带来对映射表的严重读写依赖，读写一致性要求考虑，通常需要引入meta data service。
- 基于前缀和范围mapping：基于前缀和范围的映射，将键范围映射到Grid，并在提供灵活性的同时，弥补了Full-Mapping的不足。
- Mapping代替：强制将特定key分配给特定Grid，方便测试、隔离。

d. 进行Grid路由层设计。设计原则如下：

- 路由层是系统唯一的一个共享组件，因此需要尽可能的稳定，减少修改。
- 避免业务逻辑，保证尽可能的稳定，减少修改。
- 由于爆炸半径大，需要足够轻，足够简单，但是不能太简单。
- 某些情况，要考虑避免路由所有调用，有助于减少延迟，并减小路由层的规模。
- 支持横向扩展，避免路由层成为性能瓶颈。

e. 提供Grid迁移功能，以便在增加/删除Grid业务单元时，可以快速调整分区键对应的Grid业务单元。典型处理过程如下：

- 从分区键对应的旧位置拷贝数据到新位置。
- 更新Grid路由层路由，使分区键重定向到新位置。
- 从分区键旧位置删除数据。

f. Grid代码部署与更新：

- Grid代码部署可与跨AZ、跨Region结合，通过多层隔离，减少故障影响范围。
- Grid业务单元代码更新时，建议采用类似金丝雀部署（灰度发布）的方式进行更新，以减少由于版本问题而导致多个Grid业务单元同时故障的可能

2.7.3.4 RES10-04 健康检查与自动隔离

对应用组件进行健康检查，当发现故障后进行主动隔离，避免故障扩散。

- **风险等级**
  高
- **关键策略**
  - 对系统内组件需要定期进行健康检查，以判断其状态是否正常。
  - 对于异常组件，需要能支持自动隔离，避免对整体业务造成影响。
- **相关云服务和工具**
  - 弹性负载均衡器 ELB：支持健康检查，会定期向后端服务器发送请求以测试其运行状态，并根据健康检查来判断后端服务器是否可用，当判断为异常后就不会将流量分发给该异常后端服务器。
  - 云容器引擎 CCE：支持容器健康检查，容器运行过程中，可根据用户需要，定时检查容器健康状况。若不配置健康检查，如果容器内应用程序异常，Pod将无法感知，也不会自动重启去恢复。最终导致虽然Pod状态显示正常，但Pod中的应用程序异常的情况。

2.7.4 RES11 可靠性测试

可靠性测试是为了保证系统在规定的生命周期内，达到预期的可靠性目标；与通常的功能测试不同，可靠性测试需要在业务负荷叠加故障中进行，对测试环境和能力提出了更高要求。

可靠性测试和演练通过主动引入故障来充分验证软件质量的脆弱性，从而提前发现系统风险、提升测试质量、完善风险预案、加强监控告警、提升故障应急效率等方面做到故障发生前有效预防，故障发生时及时应对，故障恢复后回归验证。基于故障本身打造分布式系统韧性，持续提升软件质量，增强团队对软件生产运行的信心，减少业务运行中出现类似问题。

为了保证测试的有效性，测试环境需要与生产环境保持一致。

华为云提供了MAS-CAST故障注入服务、CodeArts PerfTest性能测试服务、MAS多活高可用服务，可用于故障注入测试、压力负荷测试、长稳测试以及灾难演练。

2.7.4.1 RES11-01 混沌测试

混沌工程（Chaos Engineering）是通过故障注入的方式，触发或模拟实际故障，验证系统的稳定性和容错保护能力。

- **风险等级**

  高

- **关键策略**
  - 在真实环境中测试。
  - 作为CI/CD管道的一部分例行执行。
  - 主动注入故障，以便在问题发生前提前发现并解决问题。
  - 以可控方式注入故障，减少对客户的影响。

  混沌工程度量指标：
  - 故障场景的覆盖率：分析故障场景的覆盖率，例如容灾场景覆盖 80%，过载场景覆盖 60%。
  - 故障场景的命中率：分析故障场景中，真实发生的比率。
  - 应急预案的质量：用于度量应急预案有效性和执行效率。
  - 风险发现个数与等级：定期评估分析（季度或年度）主动发现的风险数量和级别。
  - 风险消减个数、等级与类型：风险降级的数量，风险消减的数量，增加预案的数量，改进监控项的数量。
  - 故障恢复时长提升率：对应故障场景经过混沌工程演练，平均恢复速度提升的比率。
  - 故障数量相比上年减少数量：本年度故障数量相比上年度减少多少。
- **相关云服务和工具**
  - MAS-CAST故障注入服务：针对云应用提供测试工具和注入手段，支持故障和业务流程编排的可靠性评估测试、压力负荷测试、CHAOS随机故障注入、生产环境故障演练等能力。
  - 云运维中心 COC：支持混沌演练，为用户提供一站式的自动化演练能力，覆盖从风险识别、应急预案管理、故障注入到复盘改进的端到端的演练流程。

2.7.4.2 RES11-02 压力负载测试

通过施加超出系统容量的业务压力，验证云服务的过载保护、业务隔离和优雅降级等能力。为全面验证系统整体的容量规划和业务依赖，云服务应用通常采用全链路压测进行测试。

- **风险等级**

  高

- **关键策略**
  - 模拟大量接口消息进行压力测试。
  - 模拟各种业务场景进行压力测试。
  - 持续自动测试。
  - 性能发生偏差时自动告警，以便及时定位和处理。
- **相关云服务和工具**
  - 性能测试 CodeArts PerfTest：针对HTTP/HTTPS/TCP/UDP/HLS/RTMP/WEBSOCKET/HTTP-FLV等协议构建的云应用提供性能测试的服务，其支持快速模拟大规模并发用户的业务高峰场景，通过自定义报文内容、时序、多事务组合等复杂场景，帮助用户测试验证业务高峰下的服务表现。

2.7.4.3 RES11-03 长稳测试

基于用户使用场景构建业务模型，自动化构建覆盖系统容量规格70%的业务量，持续7*24小时进行长时间负载测试以评估系统稳定性。

- **风险等级**

  高

- **关键策略**
  - 模拟各种业务场景进行测试。
  - 持续自动测试。
  - 测试结果发生偏差时自动告警，以便及时定位和处理。

2.7.4.4 RES11-04 灾难演练

通过容灾演练，可以验证灾备系统是否可用，且数据丢失时间以及恢复时间符合数据的RPO与RTO指标要求。

- **风险等级**

  高

- **关键策略**

  灾难演练着重测试服务跨AZ或跨Region故障转移能力，验证系统的容灾能力以及面对灾难时的应对能力，涉及到多个团队间配合，通常作为专项开展。容灾演练可以帮助企业更好的验证RPO、RTO指标，及时发现和解决相关问题，提高系统的可用性和可靠性。

- **相关云服务和工具**
  - MAS多活高可用服务灾难演练：支持同城跨AZ灾备/双活、两地三中心及异地多活等场景下的业务高可用容灾管理、工作流编排及演练切换功能。

2.7.4.5 RES11-05 红蓝攻防

通过红蓝攻防，可以模拟各种复杂的攻击场景，帮助全面评估应用韧性，及时发现并解决潜在风险。

- **风险等级**

  高

- **关键策略**
  - 蓝军从第三方角度发掘各类脆弱点，并向业务所依赖的各种软硬件注入故障，不断验证业务系统的可靠性；而红军则需要按照预先定义的故障响应和应急流程进行处置。
  - 演练结束后，建议针对故障中的发现、响应、恢复三个阶段的时长和操作内容进行复盘，并梳理改进点进行优化，提升业务系统的稳定性。

2.7.5 RES12 应急恢复处理

应用系统无论如何精心设计，仍可能会出现无法恢复的故障，当此类故障发生后，需要进行应急恢复处理。

2.7.5.1 RES12-01 组建应急恢复团队

为了应对紧急故障场景，需要组建应急恢复团队，明确责任人，并进行培训。

- **风险等级**

  高
- **关键策略**
  - **组建应急恢复团队：**其中包括应急恢复主席及所有组件及关键依赖项的恢复责任人。
    - 应急恢复主席：在出现问题后及时组织应急恢复团队进行快速恢复处理。
    - 组件或关键依赖项运维责任人：负责问题定位和应急恢复处理。
  - **制定应急恢复管理方案：**所有应急恢复团队人员都需要进行应急恢复培训，熟悉应急恢复处理流程和恢复方法。

2.7.5.2 RES12-02 制定应急预案

针对常见问题现象，提供标准化的应急恢复指导，以便在出现问题后，可以有序的完成恢复操作，避免操作失误。

- **风险等级**

  高
- **关键策略**
  - 需要覆盖常用典型场景。
  - 应急恢复需要有标准的操作流程和动作，确保在事件发生时，相关干系人都能够明确自身职责和所需要采取的措施。
  - 每个恢复操作动作必须明确无歧义，可指导操作人员。
- **相关云服务和工具**
  - 云运维中心 COC：支持应急预案管理。

2.7.5.3 RES12-03 定期应急恢复演练

定期测试突发事件应急恢复处理，以便在出现问题后能进行高效的恢复处理。

- **风险等级**

  高

- **关键策略**
  - 每年至少进行一次应急恢复演练；通过演练可提升操作人员的熟练程度。
  - 演练期间严格按照应急预案进行恢复，以检验应急预案的准确性。
  - 演练结束后需要对恢复过程进行回溯，并优化应急预案。
- **相关云服务和工具**
  - 云运维中心 COC：支持混沌演练，为用户提供一站式的自动化演练能力，覆盖从风险识别、应急预案管理、故障注入到复盘改进的端到端的演练流程。

2.7.5.4 RES12-04 出现问题后尽快恢复业务

应用系统出现故障后，需要能尽快发现，尽快响应。

- **风险等级**

  高
- **关键策略**

  可以通过以下途径实现故障的快速发现：
  - 监控：应用系统需要提供业务监控信息，以便实时了解系统运行状态；维护团队需要有专人观测，并在发现故障发生时，需要及时响应。
  - 告警：应用系统在检测到故障后需要及时告警，并能通过短消息、邮件等方式发送给所有相关人员，确保使相关人第一时间得知故障信息，以便快速组织应急响应。
  - 预测：维护团队需要根据系统运行现状，通过数据分析、机器学习等方式，预测系统的风险情况，提前进行预防和处理。

  在进行应急恢复处理时，通常需要尽快缓解或恢复业务，快速结束业务中断对客户的影响，然后再启动问题定位和修复处理流程，以减少业务中断时间。
  - 组织协调：故障发生后，应急恢复主席需要迅速组织相关人员快速恢复业务。
  - 应急恢复处理：系统发生故障后需要快速问题分析并按照事先制定的应急预案进行恢复处理。

2.7.5.5 RES12-05 应急恢复回溯

在业务进行应急恢复处理后，需要对事件进行回溯并进行优化，以避免故障的再次发生。

- **风险等级**

  高
- **关键策略**
  - 对问题进行定位和修复，优化产品能力，减少同类事件的发生。
  - 针对应急恢复过程进行总结，优化恢复过程。

2.8 过载控制

系统内组件资源有限，在遇到突发流量时可能会造成资源耗尽，而导致业务受损。

2.8.1 RES13 过载保护

当系统流量超过一定阈值后，导致系统处于过载状态时，可能会导致部分请求失败，失败触发业务重试，会进一步增加系统的负荷，形成恶性循环，导致业务成功率远远低于系统的设计容量，甚至整体不可用。因此应用应该设计过载保护机制，使得在过载状态下依然可以保证一定比例设计容量的处理能力。

通过过载保护，可以缓解客户流量突增、泛洪攻击或重试风暴所造成的大量容量峰值情况，让工作负载能够继续正常处理支持的请求量，避免出现资源耗尽而导致所有请求都不能处理的情况。

2.8.1.1 RES13-01 采用自动弹性扩缩容

当系统突发流量时，通过自动弹性扩容，可减少业务中断影响。

- **风险等级**

  高

- **关键策略**

  弹性扩缩容需要通过业务处理逻辑与数据分离、状态外置等技术手段支撑系统处理能力的快速增加或减少。

  系统扩容和缩容的处理方式有两种，一种是改变单机的处理能力，包括CPU、内存、存储等，称之为纵向伸缩；另一种是单机节点处理能力不变，通过增加节点的数量来改变系统的处理能力，称之为横向伸缩。

  系统设计时一般建议采用横向伸缩。采用横向伸缩时，要求业务与数据解耦，即将系统的业务处理逻辑与数据分离、数据（状态）外置，以实现业务节点（含资源）无状态，按需快速增加或减少，从而实现系统业务处理能力的伸缩。

  当节点故障或资源不足时，系统需要自动检测和扩展节点，以实现自动横向扩缩容，自动增加资源容量，解决业务处理能力不足的问题，无需人工干预。

  华为云提供AS弹性伸缩服务，可以根据伸缩组内的负载情况，及伸缩规则，自动调整ECS实例、带宽等资源。当业务需求增长时，AS自动增加弹性云服务器（ECS）实例或带宽资源，以保证业务能力；当业务需求下降时，AS自动缩减弹性云服务器（ECS）实例或带宽资源，以节约成本。

  此外，华为云还提供了一些内嵌伸缩能力的云服务，对用户无感知或仅需简单配置：

  - OBS、SFS、FunctionGraph等服务会根据请求量自动扩展业务处理能力，用户无感知。
  - RDS服务最多支持5个只读副本，可在线扩展只读负载；一键规格变更实现CPU、内存扩容/缩容；在线存储容量扩容。
  - CCE服务支持配置自动扩容集群节点和工作负载，伸缩策略支持告警（按CPU或内存使用率触发）、定时、周期多种方式。

- **相关云服务和工具**
  - 弹性伸缩 AS
  - 云容器引擎 CCE
  - 云数据库 RDS
  - 对象存储服务 OBS
  - 弹性文件服务 SFS
  - 函数工作流 FunctionGraph

2.8.1.2 RES13-02 应用系统负载均衡，避免流量不均匀

针对无状态集群业务，通过负载均衡来保证业务均匀分发，可避免部分组件空闲，而部分组件过载而影响业务；同时还可以充分利用系统资源，提高系统性能，改善系统可靠性。

- **风险等级**

  高

- **关键策略**
  - 负载均衡分发业务粒度需避免过大，而导致部分组件过载。
  - 负载均衡分发时需检查后端节点的负载状态，并根据各节点的负载进行业务分发。
  - 在后端节点故障的情况下，需要自动将业务分发给其他健康节点处理，以避免业务失败。
- **相关云服务和工具**
  - 弹性负载均衡 ELB：支持业务负载均衡处理，还支持后端服务器健康状态检测，自动隔离异常状态的ECS。

2.8.1.3 RES13-03 过载检测与流量控制

当应用系统发生过载时，可能会导致系统疲于处理请求而无法有效提供服务，因此需要进行过载检测并进行流量控制。

- **风险等级**

  高

- **关键策略**

  过载控制(也称流控)指系统处于过载时，通过限流、降级、熔断、弹性伸缩等手段，使系统保证部分或者全部额定容量业务成功处理的控制过程；典型过载控制方法定义如下：
  - 限流：在系统过载时主动丢弃部分业务请求。
  - 降级：在系统过载时提供有损服务，通过减少非核心业务，降低业务质量等措施降低系统负载。
  - 熔断：在分布式系统中，应用调用第三方资源和服务时由于第三方资源和服务故障（包括过载）而失败，停止调用远程资源和服务，避免故障扩散。
- **相关云服务和工具**

  华为云提供了一些内嵌流控保护的云服务，用户可直接配置使用：
  - API网关 APIG：支持配置流控策略，用户可指定单位时间内的单个API、单个用户或单个APP的请求次数上限。
  - 微服务引擎 CSE：支持限流，用户可指定一定时间内可接受的请求次数上限。

2.8.1.4 RES13-04 支持主动扩容

当由于计划性活动而导致资源需求增加时，需要支持主动扩容，避免由于资源不足而导致业务受影响。

- **风险等级**

  高

- **关键策略**

  当发现应用系统业务需要更多资源时，可主动扩展资源以满足需求，而避免影响可用性。典型场景如产品促销前预测会有突发大流量，则可手工进行扩容处理。

  华为云服务实例支持主动横向或纵向扩容功能；如对于ECS实例可以通过创建多个ECS实例实现横向扩容，也可升级ECS规格实现纵向扩容；对于RDS实例可升级RDS实例规格实现纵向扩容。

2.8.1.5 RES13-05 资源自动扩容考虑了配额限制

当应用系统在资源不足自动扩容时，需要考虑配额的限制，若配额不足，会导致自动扩容失败。

- **风险等级**

  高

- **关键策略**

  华为云为防止资源滥用，限定了各服务资源的配额，对用户的资源数量和容量做了限制。如您最多可以创建多少台弹性云服务器、多少块云硬盘。在动态使用云服务资源时，需要了解云服务的限制，避免由于超过云服务配额限制而导致业务故障。当配置自动扩容时，需要确保自动扩容到最大时的规则不超过配额限制。

  在系统中也可配置资源使用超过一定限额后进行预警，避免配额超过限制后导致业务受影响。

- **相关云服务和工具**

  使用华为云“我的配额”，可以查询每个云服务不同资源类型的总配额限制和已用配额，可根据业务的需要申请扩大对应云服务指定资源的配额，也可配置配额预警，以便在配额达到预警阈值时可收到告警通知，以便提前申请提升配额。

  当应用系统中涉及到资源的弹性伸缩时，尤其需要关注弹性伸缩的配置是否会被限制，比如AS弹性伸缩服务中可以配置能创建的最大实例数量，而在过载情况下是否能真的创建出那么多实例，会依赖于ECS弹性云服务器配额、EVS云硬盘配额，当需要弹性公网IP时涉及弹性公网IP配额等，当配额不足时会导致无法创建工作负载进行业务分担，而业务受损。

2.8.1.6 RES13-06 压力负载测试

通过压力测试，可衡量系统的弹性扩容能力是否能满足业务要求。

- **风险等级**

  高

- **关键策略**

  参见“**RES11-02 压力负载测试**”章节。

2.9 变更防差错

在系统的运行过程中，配置变更是导致生产系统不可用的重要风险之一，如配置修改、工作负载手工增缩或补丁安装等。当变更失败时，可能会导致性能下降或业务中断等严重的问题。因此为了降低变更带来的业务风险，需要为工作负载或其环境的更改做好准备，实现工作负载的可靠操作。

变更操作属于运维的一部分，内容可参考卓越运营支柱部分“运维准备和变更管理”。

2.9.1 RES14 配置防差错

配置防差错是针对配置过程中因人输入了错误的配置数据导致系统和业务受损或失效场景下通过产品设计降低或避免配置错误产生的影响。

2.9.1.1 RES14-01 变更防呆检查

防呆是一种预防矫正的行为约束手段，运用防止错误发生的限制方法，让操作者不需要花费注意力、也不需要经验与专业知识，凭借直觉即可准确无误地完成的操作。

- **风险等级**

  高

- **关键策略**

  通过以下约束和检查，可减少配置差错：

  - 角色约束：通过权限控制设计预防对不同角色的配置范围进行约束，避免越权配置导致错误。
  - 查改分离：通过产品界面设计将配置界面分层分级，查看与修改分离等降低人为配置失误风险。
  - 配置校验：通过配置生效机制设计确保在配置生效前进行必要的校验，避免错误配置生效。
  - 删除保护：在删除资源时增加保护机制，防止误删，如：删除前运行状态检查保护，资源锁定防止误删除，回收站机制等。

2.9.1.2 RES14-02 自动化变更

自动化变更是指自动化提供并管理应用程序的环境（计算、存储、网络、中间件服务等）、安装、配置，实现Infrastructure as a Code；以解决手工部署中易于出错、依赖个人能力，手工配置中变更无法跟踪、难以回滚等难题。

- **风险等级**

  高

- **关键策略**

  - 使用配置管理工具进行变更：集中管理配置信息，发现和记录配置变化情况，快速识别变更影响范围。
  - 采用自动化变更流程：帮助组织规划和自动化变更流程，如预定义变更模板、审批变更流程、自动化测试和验证等，减少人工错误和延迟。
  - 进行变更评估和风险管理：评估变更影响范围，识别潜在风险和冲突，并采取相应的措施进行风险管理。
  - 自动化测试和验证：验证变更的正确性以及性能、可靠性影响，减少人工测试的错误和延迟。
  - 监控和审计变更过程：追踪和记录变更执行情况，及时发现和解决问题，提供透明度和可追溯性。

- **相关云服务和工具**

  - 云运维中心 COC：

    - 作业管理：提供用户自定义作业的创建、修改、删除以及在目标虚拟机上执行自定义作业的能力。通过该功能，用户可以通过自定义作业在目标实例（目前支持ECS）上执行操作。

- 变更中心：支持承载变更流程管理业务，以变更工单模式，从变更的申请、审批、执行三个大环节管控变更业务，为变更人员、变更管理人员提供统一管理平台。

2.9.1.3 RES14-03 变更前数据备份

通过配置数据事前备份与恢复设计，确保在出现配置错误时能够快速恢复到正确的配置数据状态。

- **风险等级**

  高

- **关键策略**
  - 进行全量数据备份，以防变更过程中数据被破坏，影响业务。
  - 异常回滚时，可使用备份数据进行恢复。

2.9.1.4 RES14-04 提供 runbook 进行标准化变更

runbook是指运行手册，是用来实现变更的详细操作过程。

变更前需提供标准化runbook用于变更和回退，变更过程中严格按照runbook执行，在变更失败时根据runbook进行回退。

- **风险等级**

  高

- **关键策略**
  - runbook需涵盖变更前检查、变更操作、变更后检查及变更失败回退操作。

2.9.2 RES15 升级不中断业务

软件版本在重新部署或升级过程中，需要尽可能避免业务中断，减少业务影响。

2.9.2.1 RES15-01 自动化部署和升级

部署和升级过程由代码实现，以固化部件间依赖、安装和配置过程，减少人工错误。

- **风险等级**

  高

- **关键策略**
  - 部署和升级过程自动化完成。

2.9.2.2 RES15-02 自动化检查

在部署或升级过程中集成基本测试功能，实现自动化检查，无需人工参与。

- **风险等级**

  高

- **关键策略**
  - 在部署或升级过程中集成基本测试功能，在部署或升级完成后自动进行检查和测试，以验证新部署的代码功能是否正确。

- 在部署或升级过程中集成故障注入测试功能，在部署或升级完成后自动注入故障进行测试，以验证新部署代码的韧性。

2.9.2.3 RES15-03 自动化回滚

在升级或部署过程中出现异常，或检查/测试失败时，支持自动回滚，减少人工干预，避免回滚失败。

- **风险等级**

  高

- **关键策略**
  - 检测到异常后，可一键式回滚。
  - 回滚过程自动化完成。

2.9.2.4 RES15-04 灰度部署和升级

原地升级和回滚时，升级和回滚过程中业务将会中断，中断时长受限于升级和回滚的时长，对业务影响比较大；而采用灰度部署和升级，可减少升级和回滚过程中的业务中断，提升系统可用性。

- **风险等级**

  高

- **关键策略**

  通过金丝雀部署、蓝绿部署等方式实现灰度升级或部署，逐步引入新版本部署范围或切换用户流量，配合自动回退以降低部署差错导致业务中断的风险。

  金丝雀部署（灰度发布）是将少量客户引导到新版本的做法，通常在单个服务实例（Canary）上运行；当检查到任何行为更改或错误时，可以将Canary中的流量删除，并将用户发回到以前的版本。如果部署成功，则可以继续以期望的速度进行部署，同时监控更改以便发现错误，直到所有部署完成。

  蓝绿部署与金丝雀部署类似，只是会并行部署一整套应用程序，形成两套生产环境：蓝环境和绿环境，蓝色是当前版本并拥有实时流量，绿色是包含更新代码的环境。当应用程序已经准备就绪，用户可以将所有流量都将路由到绿环境中，当出现问题时，可以快速将流量重新路由回蓝环境，进行故障恢复。

- **相关云服务和工具**
  - 部署 CodeArts Deploy：提供可视化、自动化部署能力，提供丰富的部署步骤，有助于用户制定标准的部署流程，降低部署成本，提升发布效率。
  - 微服务引擎 CSE：支持灰度发布。
  - 应用服务网格 ASM：支持灰度发布。

2.10 参考架构

2.10.1 概述

本章节以典型Web应用为例，介绍不同可用性目标要求下部署的典型架构示例。针对每种场景，从以下几个维度进行设计，来达成可用性目标。

类别	应用可用性影响
冗余	应用内组件的高可用能力，在应用内部分节点故障时业务自动恢复能力
备份	应用数据被破坏的情况下的恢复能力
容灾	在Region/AZ/IDC或其他云站点发生灾难的情况下的恢复能力
监控告警	应用系统故障后的检测和告警能力
弹性扩缩容	应用容量不足时的自动恢复能力
变更防差错	变更对应用业务中断的影响
应急恢复处理	应用在故障情况下的应急恢复能力

2.10.2 内部工具或公测类应用典型部署架构（99%）

内部工具类应用通常用于内部操作，且在故障时只会对内部员工造成影响，不可用时只会带来不方便，可以承受长时间的恢复时间和恢复点；公测类应用用于面向客户的实验性的工作负载，在必要时可以隐藏其功能；针对这些应用，其可用性目标通常要求不高，可达到99%，即每年中断时间可以为3.65天。

导致业务中断的时间包含故障中断时间及由于升级配置维护等导致的中断时间，假定分别中断时间如下：

- 故障中断：假定每年故障中断4次，每次应急恢复决策时长为1小时，应用负载重新部署、配置与数据恢复时长为2小时，则每年故障中断时长为12小时。
- 变更中断：假定应用离线更新，每年更新6次，每次更新时长4小时，则每年更新时长为24小时。

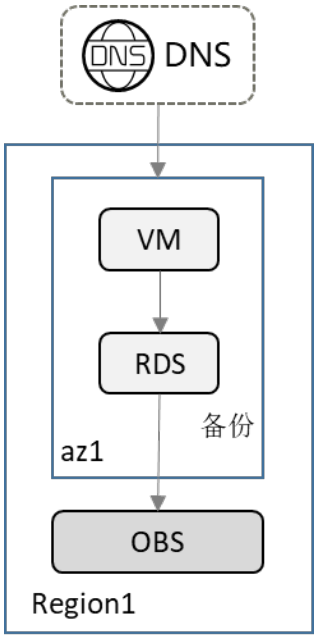
按照以上评估，每年应用系统不可用的时长是36小时，满足可用设计目标要求。

内部工具类应用典型架构为前端无状态应用层+后端数据库，其中前端无状态应用可采用ECS或CCE（以ECS为例），后端数据库基于不同业务类型可采用不同数据库，通常为RDS for MySQL；为满足对应的可用性目标，建议方案如下：

类别	实施方案
冗余	ECS与RDS单节点部署。
备份	RDS自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	不支持容灾部署，在站点故障的情况下，重新进行应用部署与备份数据恢复。
监控告警	进行简单的监控，检查应用系统是否能正常返回消息。
弹性扩缩容	提供常见故障处理runbook，以便在容量不足等场景可以手工扩容。

类别	实施方案
变更防差错	软件更新采用离线更新，安装和重启应用需要停机，根据runbook进行应用的部署与回滚。
应急恢复处理	指定应用系统责任人，在突发事件后能找到相关责任人进行恢复处理。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统部署在单Region单AZ。
- 为了保证数据的可靠性，RDS数据库的数据定期自动备份到OBS，在数据丢失时可以快速恢复。

2.10.3 内部知识管理类应用典型部署架构（99.9%）

内部知识管理类应用通常用于内部操作，且在故障时只会对内部员工造成影响，可以承受较长的恢复时间和恢复点，其可用性目标通常要求达到99.9%，即每年中断时间可以为8.76小时。

导致业务中断的时间包含故障中断时间及由于升级配置维护等导致的中断时间，假定分别中断时间如下：

- 故障中断：假定每年故障中断4次，每次应急恢复决策时长为30分钟，恢复处理时长为30分钟，则每年故障中断时长为240分钟。
- 变更中断：假定应用离线更新，每年更新8次，每次更新时长30分钟，则每年更新时长为240分钟。

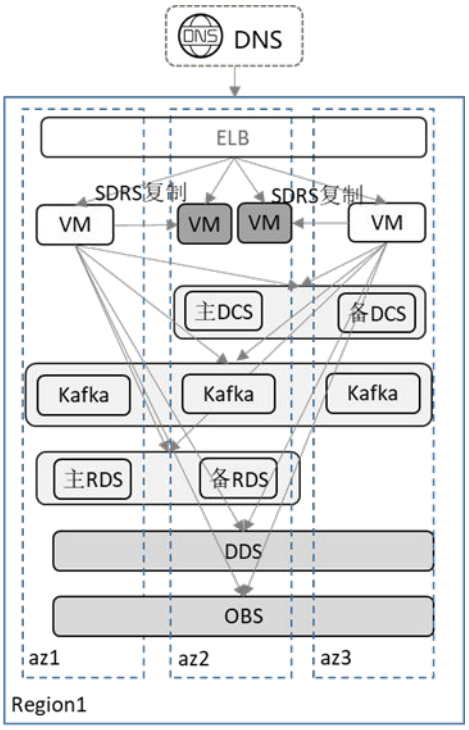
按照以上评估，每年应用系统不可用的时长是480分钟，满足可用设计目标要求。

内部知识管理类应用典型架构为前端无状态应用层+后端数据库，其中前端无状态应用采用ECS，后端数据库基于不同业务类型可采用不同数据库，通常为RDS for MySQL；

基于业务需要，通常还会使用DCS、Kafka等中间件及DDS文档数据库；为满足对应的可用性目标，建议方案如下：

类别	实施方案
冗余	ELB、RDS、DCS、Kafka、DDS等云服务实例均采用高可用部署。
备份	RDS、DDS数据库自动备份，有状态ECS通过CBR自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	应用使用支持跨AZ的服务进行跨AZ部署，ELB、RDS跨AZ部署，AZ故障时自动恢复。有状态ECS通过SDRS进行跨AZ容灾，在AZ故障时手工切换。
监控告警	进行站点运行状态检查，在发生故障时告警；针对ECS、RDS实例负载状态进行监控，在资源过载时需要告警。
弹性扩缩容	针对内部用户场景，资源足够，无需自动弹性伸缩；针对ECS，通过ELB实现ECS实例的故障检测与负载均衡，并可根据ECS监控情况随时添加和移除ECS实例来扩展应用系统的服务能力；针对RDS，可根据RDS负载监控情况，在维护时段更改实例类型或增加只读节点。
变更防差错	软件更新采用离线更新，在位替换，根据runbook进行应用的自动部署与回滚。每1~2个月更新一次软件。
应急恢复处理	制定应急处理机制，指定应急恢复人员，以便在突发事件后能快速决策和恢复；并提供常见应用、数据库问题以及升级部署失败的相关解决方案，以便在出现问题后可以及时恢复。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统采用有状态虚拟机+有状态数据库的分层部署架构。
- 该应用系统在华为云单个Region部署一套完整系统，采用跨AZ部署，其中有状态虚拟机采用跨AZ主备复制，可以实现云内应用层跨数据中心主备容灾。
  - 接入层（外部DNS）：通过外部DNS进行域名解析与流量负载均衡，单个AZ故障对业务没有影响。
  - 应用层（负载均衡器、应用软件及虚拟机）：对于有状态应用，通过SDRS服务实现跨AZ的虚拟机数据复制与容灾切换，并可通过CBR服务进行自动数据备份。
  - 中间件层：Redis、Kafka集群跨可用区高可用部署，单个AZ故障对业务没有影响。
  - 数据层：RDS与DDS数据库及OBS对象存储跨可用区高可用部署，单个AZ故障对业务没有影响。
- 为了保证数据的可靠性，RDS数据库的数据定期自动备份到OBS，在数据丢失时可以快速恢复。

2.10.4 信息管理类应用典型部署架构（99.95%）

信息管理类应用通常用于内部操作，且在故障时只会对内部员工造成影响，可以承受一定的恢复时间和恢复点，其可用性目标通常要求达到99.95%，即每年故障时长可以为4.38小时。

假定故障中断与变更中断的时长分别如下：

- 故障中断：假定每年故障中断4次，每次应急恢复决策时长为20分钟，恢复处理时长为10分钟，则每年故障中断时长为120分钟。
- 变更中断：假定应用支持离线更新与在线补丁，每年离线更新4次，每次更新时长30分钟，则每年更新时长为120分钟；在线补丁不影响业务。

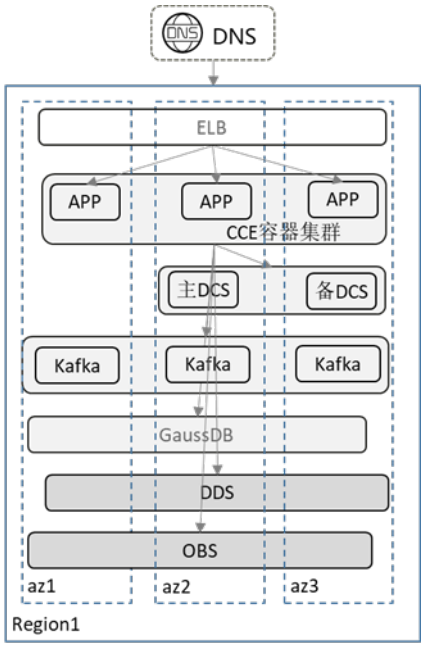
按照以上评估，每年应用系统不可用的时长是240分钟，满足可用设计目标要求。

信息管理类应用典型架构为前端无状态应用层+后端数据库，其中前端无状态应用可采用ECS或CCE（以CCE为例），通过ELB负载均衡；后端数据库基于不同业务类型可采用不同数据库，通常采用GaussDB提供更高性能与可靠性；基于业务需要，通常还会使用DCS、Kafka等中间件及DDS文档数据库；为满足对应的可用性目标，建议方案如下：

类别	实施方案
冗余	ELB、CCE、DCS、Kafka、GaussDB、DDS等云服务实例均高可用部署。
备份	GaussDB、DDS数据库自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	应用跨3AZ部署，AZ故障时自动恢复。
监控告警	支持业务运行状况、成功指标的检查，在发生故障时告警；支持云服务实例负载状态及资源故障切换等的监控，在负载超过阈值或状态异常时告警。

类别	实施方案
弹性扩缩容	针对内部用户场景，资源足够，无需自动弹性伸缩；针对CCE容器，通过CCE进行负载均衡与弹性伸缩；针对GaussDB，可根据GaussDB负载监控情况，自动扩缩规格或增删只读节点。
变更防差错	软件更新采用离线更新与在线补丁，根据runbook进行应用的自动部署与回滚。每1~2个月更新一次软件。
应急恢复处理	制定应急处理机制，指定应急恢复人员，以便在突发事件后能快速决策和恢复；并提供常见应用、数据库问题以及升级部署失败的相关解决方案，以便在出现问题后可以及时恢复。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统采用无状态应用+有状态数据库的分层部署架构。
- 该应用系统在华为云单个Region部署一套完整系统，采用跨AZ部署，可以实现云内应用层跨数据中心双活。
  - 接入层（外部DNS）：通过外部DNS进行域名解析与流量负载均衡，单个AZ故障对业务没有影响。
  - 应用层（ELB负载均衡器、应用软件及容器）：对于无状态应用采用跨AZ高可用部署，通过ELB负载均衡器进行故障检测与负载均衡，并可通过CCE容器进行负载监控和弹性伸缩。
  - 中间件层：Redis、Kafka集群跨可用区高可用部署，单个AZ故障对业务没有影响。
  - 数据层：GaussDB与DDS数据库及OBS对象存储跨3AZ高可用部署，数据分布式强一致，单个AZ故障对业务没有影响，数据零丢失。
- 为了保证数据的可靠性，GaussDB与DDS数据库的数据定期自动备份。

2.10.5 电商类应用典型部署架构（99.99%）

电子商务类应用用于外部客户，需要提供较高的可用性，并能承受组件故障，其可用性目标通常要求达到99.99%，即每年故障时间可以为52.56分钟。

假定故障中断与变更中断的时长分别如下：

- 故障中断：假定每年故障中断3次，每次应急恢复决策时长为10分钟，恢复处理时长为5分钟，则每年故障中断时长为45分钟。
- 变更中断：假定应用支持金丝雀部署或蓝绿部署，并自动完成，软件更新不中断业务。

按照以上评估，每年应用系统不可用的时长是45分钟，满足可用设计目标要求。

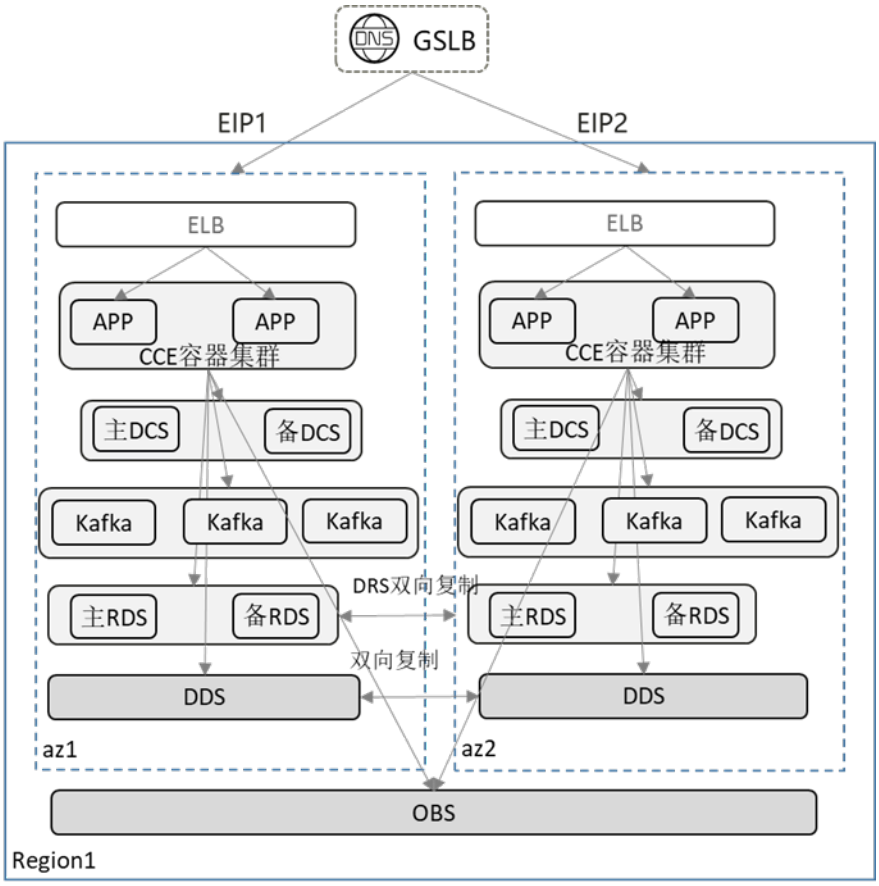
电子商务类应用典型架构为前端无状态应用层+后端数据库，其中前端无状态应用可采用ECS或CCE；后端数据库基于不同业务类型可采用不同数据库，通常采用RDS for MySQL；同时通常还会使用DCS、Kafka等中间件及DDS文档数据库；为满足对应的可用性目标，建议采用以下方案。

2.10.5.1 单 Region 方案

采用单Region时，前端以CCE为例，建议方案如下：

类别	实施方案
冗余	ELB、CCE、DCS、Kafka、RDS、DDS等云服务实例均高可用部署。
备份	RDS、DDS数据库自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	应用在两个AZ各部署一套，进行双向复制，双活容灾；AZ故障时自动恢复。
监控告警	进行站点运行状态检查，在发生故障时告警；针对CCE、DCS、kafka、RDS、DDS等实例负载状态进行监控，在资源过载时需要告警。
弹性扩缩容	CCE集群支持工作负载的自动弹性伸缩。
变更防差错	软件更新采用金丝雀或蓝绿部署，部署过程自动完成，在部署过程中出现问题时自动回滚。
应急恢复处理	制定应急处理机制，指定应急恢复人员，以便在突发事件后能快速决策和恢复；并提供常见应用、数据库问题以及升级部署失败的相关解决方案，以便在出现问题后可以及时恢复；定期进行演练，及时发现问题。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统采用无状态应用+有状态数据库的分层部署架构。
- 该应用系统在华为云一个Region两个AZ中各部署一套，提供同城容灾能力。
  - 接入层（外部GSLB）：通过外部GSLB进行域名解析与流量负载均衡，在单个AZ故障时自动将业务流量切换到另一AZ。
  - 应用层（负载均衡器、应用软件及容器）：对于无状态应用，通过负载均衡器进行故障检测与负载均衡，并可通过容器进行弹性伸缩。
  - 中间件层：每个可用区各部署一套DCS、DMS Kafka集群。
  - 数据层：每个可用区各部署一套RDS数据库，通过DRS数据复制服务实现跨AZ的双向数据库复制与容灾切换；并支持定期自动数据备份，在数据丢失时能快速恢复。OBS对象存储跨可用区高可用部署，单个AZ故障对业务没有影响。
- 为了保证数据的可靠性，RDS数据库的数据定期自动备份。

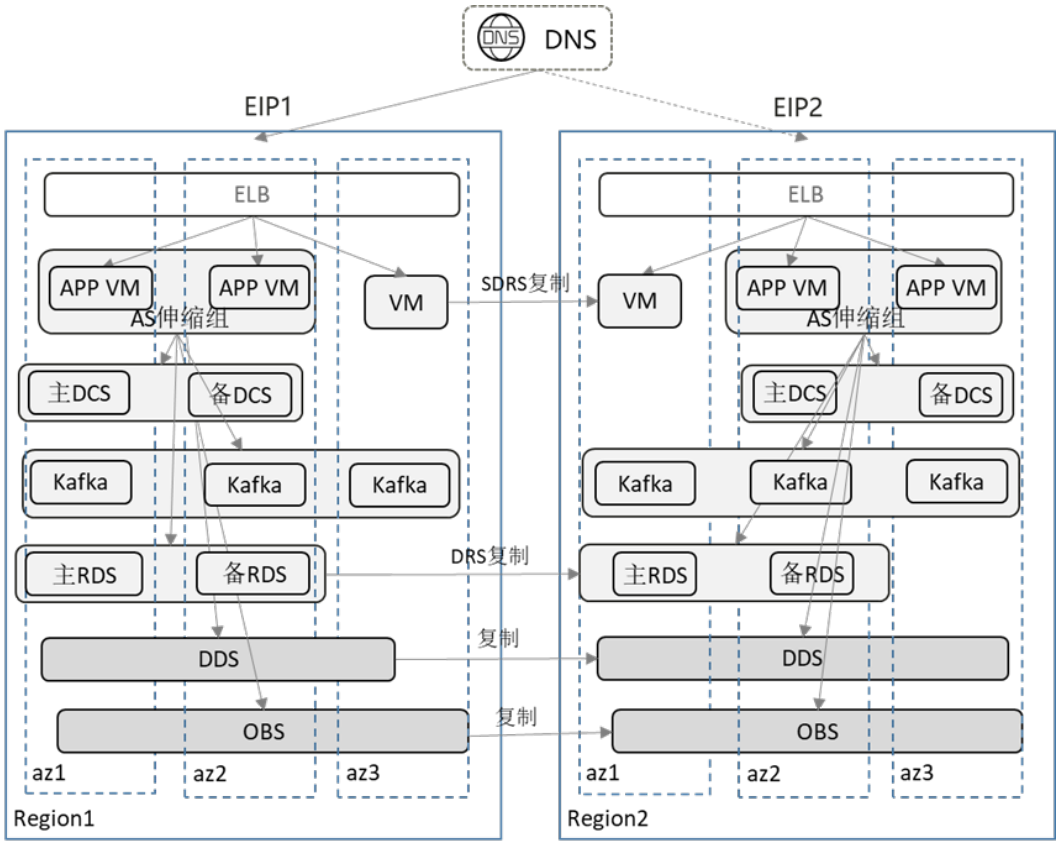
2.10.5.2 双 Region 方案

采用双Region时，前端以ECS为例，建议方案如下：

类别	实施方案
冗余	ELB、ECS、DCS、Kafka、RDS、DDS等云服务实例均高可用部署。

类别	实施方案
备份	RDS、DDS数据库自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	应用跨AZ部署，AZ故障时自动恢复；支持跨Region主备容灾，在出现Region级故障时可以快速在异地恢复业务。
监控告警	支持业务运行状况、成功指标的检查，在发生故障时告警；支持ECS、DCS、Kafka、RDS、DDS等实例负载状态及资源故障切换等的监控，在负载超过阈值或状态异常时告警。
弹性扩缩容	支持自动弹性伸缩；针对ECS，通过ELB实现ECS实例的故障检测与负载均衡，并可通过AS监控负载随时添加和移除ECS实例来扩展应用系统的服务能力；针对RDS for MySQL，可根据负载监控情况，自动扩缩规格或增删只读节点。
变更防差错	软件更新采用金丝雀或蓝绿部署，部署过程自动完成，在部署过程中出现问题时自动回滚。
应急恢复处理	制定应急处理机制，指定应急恢复人员，以便在突发事件后能快速决策和恢复；并提供常见应用、数据库问题以及升级部署失败的相关解决方案，以便在出现问题后可以及时恢复；定期进行演练，及时发现问题。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统采用无状态应用+有状态数据库/虚拟机的分层部署架构。
- 应用系统在主备Region各部署一套完整系统，主备Region间数据同步；Region内跨AZ高可用部署，提供同城跨数据中心双活能力；Region间数据支持数据异步实时同步，采用主备容灾，在一个Region故障的情况下能快速将业务恢复到另一个Region。
  - 接入层（外部DNS、API网关）：通过外部DNS进行域名解析，在一个Region故障时手工将业务流量切换到另一个Region。
  - 应用层（负载均衡器、应用软件及虚拟机）：对于无状态应用，通过ELB负载均衡器进行故障检测与负载均衡，并通过AS弹性伸缩服务监控负载进行弹性伸缩；对于有状态应用，通过SDRS服务实现跨云的虚拟机数据复制与容灾切换，并可通过CBR服务进行自动数据备份。
  - 中间件层：Redis、Kafka集群跨可用区高可用部署。
  - 数据层：MySQL数据库高可用，通过DRS数据复制服务实现跨云的数据库复制与容灾切换；并可定期自动备份数据，在数据丢失时快速恢复业务。OBS对象存储服务同样支持跨Region复制能力。
- 为了保证数据的可靠性，RDS数据库的数据定期自动备份到OBS，在数据丢失时可以快速恢复。

2.10.6 金融类核心应用典型部署架构（99.999%）

金融类核心应用通常比较重要，要求非常短的恢复时间和数据丢失量，其可用性目标通常要求达到99.999%，即每年故障时间可以为5.26分钟。

假定故障中断与变更中断的时长分别如下：

- 故障中断：由于要求的故障中断时间很短，要求尽可能自动恢复，没有手动触发的恢复，假定每年故障中断4次，每次自动恢复时长为1分钟，则每年故障中断时长为4分钟。
- 变更中断：假定应用支持金丝雀部署或蓝绿部署，并自动完成，软件更新不中断业务。

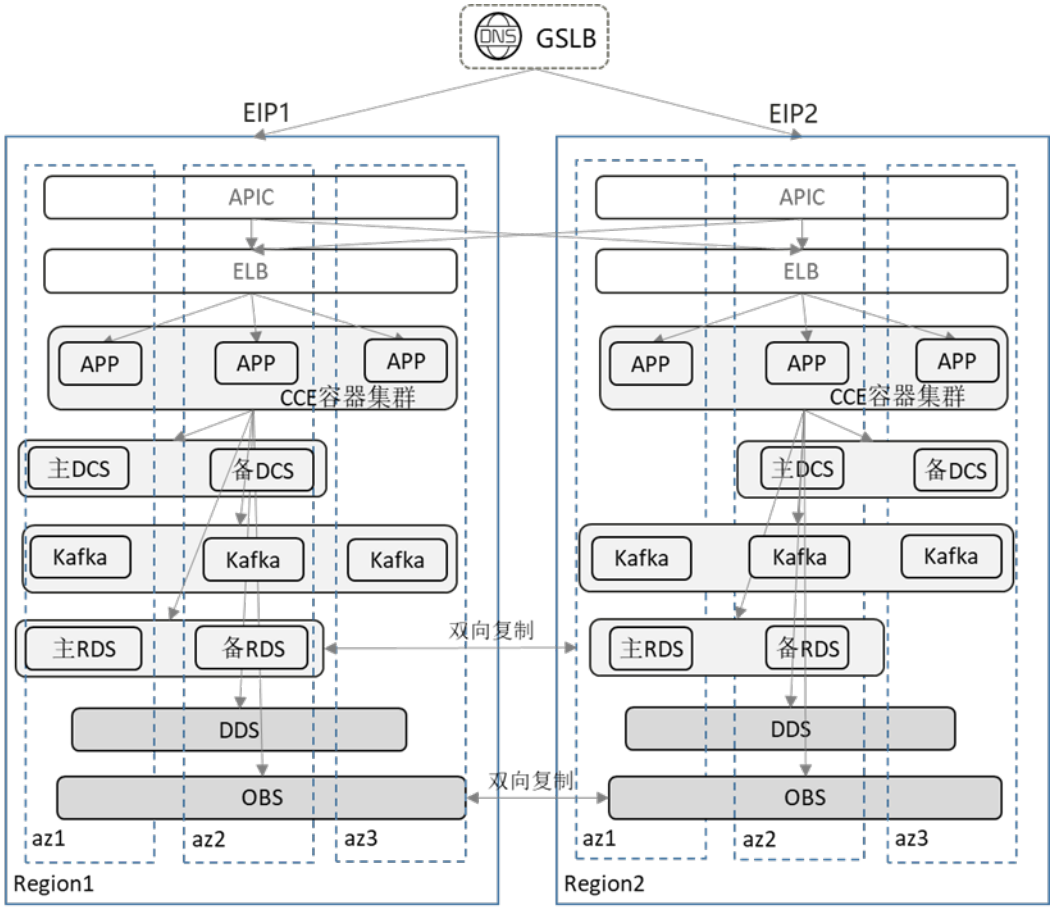
按照以上评估，每年应用系统不可用的时长是4分钟，满足可用设计目标要求。

金融类应用典型架构为三层架构：前端Web集群+后台应用集群+后端数据库集群，其中前端无状态应用可采用ECS或CCE（以CCE为例），后端数据库通常采用RDS for MySQL提供更高性能与可靠性；为满足对应的可用性目标，建议方案如下：

类别	实施方案
冗余	ELB、CCE、DCS、Kafka、RDS、DDS等云服务实例均高可用部署。
备份	RDS、DDS数据库自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	应用跨AZ部署，AZ故障时自动恢复；支持跨Region双活容灾，在出现Region级故障时可以自动切换在异地恢复业务。
监控告警	进行站点运行状态检查，在发生故障时告警；针对CCE、DCS、kafka、RDS、DDS等实例负载状态进行监控，在资源过载时需要告警。

类别	实施方案
弹性扩缩容	CCE集群支持工作负载的自动弹性伸缩。
变更防差错	软件更新采用金丝雀或蓝绿部署，部署过程自动完成，在部署过程中出现问题时自动回滚。
应急恢复处理	制定应急处理机制，指定应急恢复人员，以便在突发事件后能快速决策和恢复；并提供常见应用、数据库问题以及升级部署失败的相关解决方案，以便在出现问题后可以及时恢复；定期进行演练，及时发现问题。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统采用无状态应用+有状态数据库的分层部署架构。
- 应用系统在两个Region各部署一套完整系统，Region内跨AZ高可用部署，提供同城跨数据中心双活能力；Region间数据单元化部署，实现跨Region双活容灾，在任一Region故障的情况下能快速恢复业务。
  - 接入层（外部GSLB、API网关）：通过外部GSLB进行域名解析与流量负载均衡，两个Region同时提供服务，在单个Region故障时自动将业务流量切换到另一Region；API网关支持流量纠正，以便将业务路由到正确单元。

  - 应用层（负载均衡器、应用软件及容器）：对于无状态应用，通过ELB负载均衡器进行故障检测与负载均衡，并可通过容器进行弹性伸缩。
  - 中间件层：Redis、Kafka集群跨可用区高可用部署。
  - 数据层：MySQL数据库跨可用区高可用，通过DRS数据复制服务实现跨Region的双向数据库复制与容灾切换；并支持定期自动数据备份，在数据丢失时能快速恢复。OBS对象存储服务同样支持跨Region的双向复制能力。
- 为了保证数据的可靠性，RDS for MySQL、DDS数据库的数据定期自动备份。

2.10.7 跨云场景典型部署架构（99.99%）

2.10.7.1 跨云容灾方案

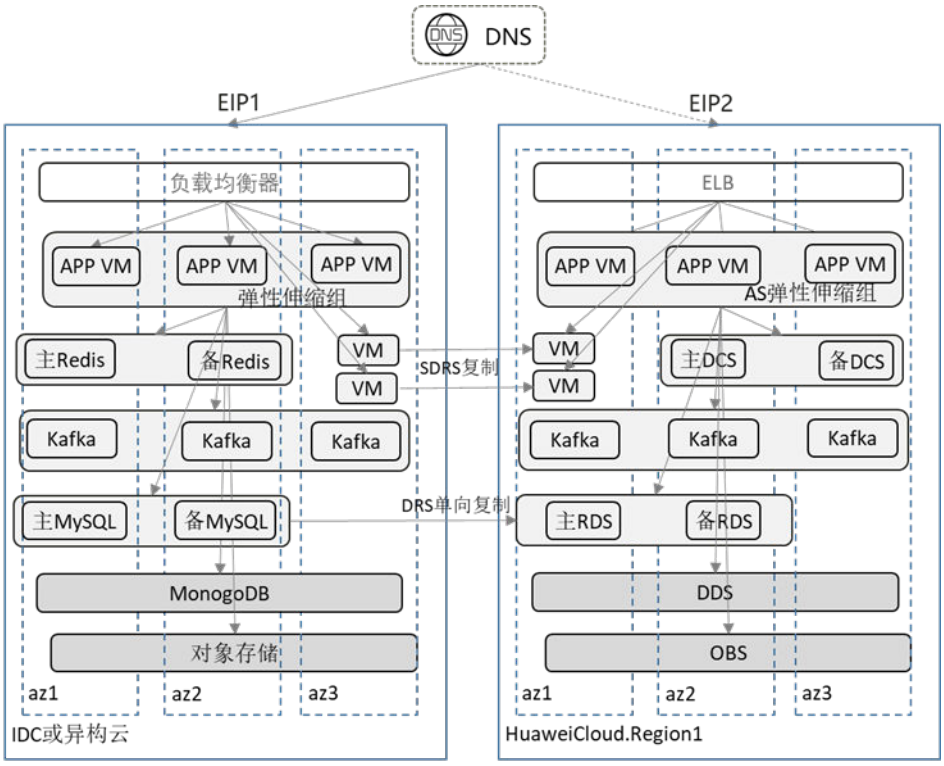
当重要应用系统已经在IDC或其他云上部署，并需要容灾到华为云，以提供高可用的容灾方案。假定应用系统在IDC或其他云上可以达到99.9%的可用性，则在容灾到华为云后，能提供99.99%的可用性。

跨云应用典型架构为前端无状态应用层+后端数据库，其中前端无状态应用可采用虚拟机或容器（以容器为例，华为云采用CCE），后端数据库通常要求采用通用MySQL数据库（华为云采用RDS for MySQL），以实现跨云容灾。

华为云上的应用部署建议方案如下：

类别	实施方案
冗余	ELB、CCE、DCS、Kafka、RDS、DDS等云服务实例均高可用部署。
备份	RDS、DDS数据库自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	应用跨AZ部署，AZ故障时自动恢复；支持跨云容灾，在IDC或其他云出现故障时可以快速切换到华为云。
监控告警	进行站点运行状态检查，在发生故障时告警；针对CCE、DCS、kafka、RDS、DDS等实例负载状态进行监控，在资源过载时需要告警。
弹性扩缩容	CCE集群支持工作负载的自动弹性伸缩。
变更防差错	软件更新采用金丝雀或蓝绿部署，部署过程自动完成，在部署过程中出现问题时自动回滚。
应急恢复处理	制定应急处理机制，指定应急恢复人员，以便在突发事件后能快速决策和恢复；并提供常见应用、数据库问题以及升级部署失败的相关解决方案，以便在出现问题后可以及时恢复；定期进行演练，及时发现问题。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统采用无状态应用+有状态数据库/虚拟机的分层部署架构。
- 应用系统在IDC/其他云与华为云中各部署一套完整系统；华为云采用跨AZ部署，可以实现云内应用层跨数据中心双活；云间数据支持将它云数据实时同步到华为云，采用主备容灾，在IDC/其他云故障的情况下能快速容灾切换到华为云。
  - 接入层（外部DNS、API网关）：通过外部DNS进行域名解析，在IDC/其他云故障时手工将业务流量切换到华为云。
  - 应用层（负载均衡器、应用软件及虚拟机或物理主机）：对于无状态应用，通过负载均衡器进行故障检测与负载均衡，在华为云上可通过AS弹性伸缩服务监控负载进行弹性伸缩；对于有状态应用，通过SDRS服务实现跨云的虚拟机数据复制与容灾切换，并可通过CBR服务进行自动数据备份。
  - 中间件层：Redis、Kafka集群跨可用区高可用部署。
  - 数据层：MySQL数据库高可用，通过DRS数据复制服务实现跨云的数据库复制与容灾切换；并可定期自动备份数据，在数据丢失时快速恢复业务。
- 为了保证数据的可靠性，数据库的数据定期自动备份，在数据丢失时可以快速恢复。

2.10.7.2 跨云双活方案

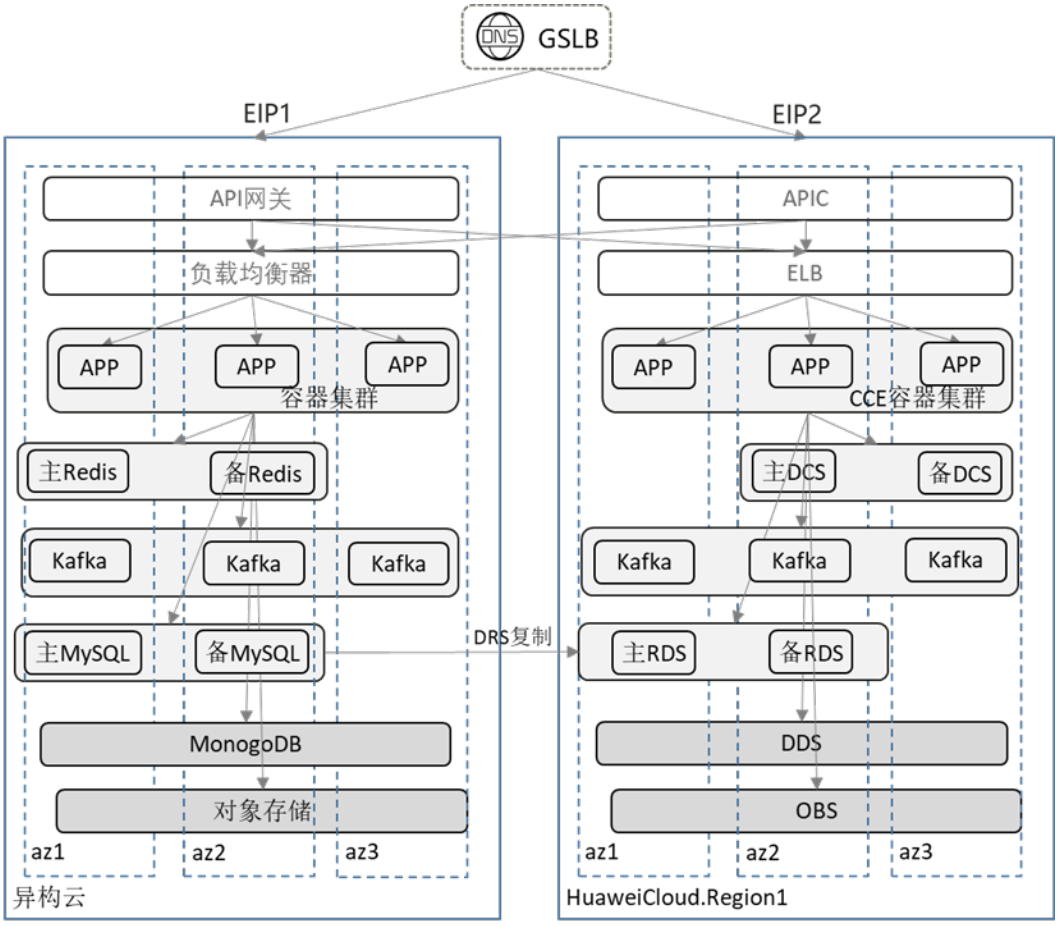
当重要应用系统已经在IDC或其他云上部署，并需在华为云上部署一套系统实现双活，以提供高可用的容灾方案。假定应用系统在IDC或其他云上可以达到99.9%的可用性，则在容灾到华为云后，能提供99.99%的可用性。

跨云应用典型架构为前端无状态应用层+后端数据库，其中前端无状态应用可采用虚拟机或容器（以容器为例，华为云采用CCE），后端数据库通常要求采用通用MySQL数据库（华为云采用RDS for MySQL），以实现跨云双活容灾。

华为云上的应用部署建议方案如下：

类别	实施方案
冗余	ELB、CCE、DCS、Kafka、RDS、DDS等云服务实例均高可用部署。
备份	RDS、DDS数据库自动备份，在数据故障时使用最新备份数据恢复，可以满足可用性目标要求。
容灾	应用跨AZ部署，AZ故障时自动恢复；支持跨云双活，在IDC或其他云出现故障时可以快速切换到华为云。
监控告警	进行站点运行状态检查，在发生故障时告警；针对CCE、DCS、kafka、RDS、DDS等实例负载状态进行监控，在资源过载时需要告警。
弹性扩缩容	CCE集群支持工作负载的自动弹性伸缩。
变更防差错	软件更新采用金丝雀或蓝绿部署，部署过程自动完成，在部署过程中出现问题时自动回滚。
应急恢复处理	制定应急处理机制，指定应急恢复人员，以便在突发事件后能快速决策和恢复；并提供常见应用、数据库问题以及升级部署失败的相关解决方案，以便在出现问题后可以及时恢复；定期进行演练，及时发现问题。

根据以上方案，典型部署架构如下：



该架构的主要特点包括：

- 应用系统采用无状态应用+有状态数据库的分层部署架构。
- 应用系统在其他云与华为云中各部署一套完整系统；华为与采用跨AZ部署，可以实现云内应用层跨数据中心双活；云间数据单元化部署，并支持将它云数据实时同步到华为云，实现双活容灾，在第三方云故障的情况下能快速容灾切换到华为云。
  - 接入层（外部GSLB、API网关）：通过外部GSLB进行域名解析与流量负载均衡，两朵云同时提供服务，在第三方云故障时自动将业务流量切换到华为云；API网关支持流量纠正，以便将业务路由到正确单元。
  - 应用层（负载均衡器、应用软件及容器）：对于无状态应用，通过负载均衡器进行故障检测与负载均衡，并可通过容器进行弹性伸缩。
  - 中间件层：Redis、Kafka集群跨可用区高可用部署。
  - 数据层：MySQL数据库跨可用区高可用，通过DRS数据复制服务实现跨云的数据库复制与容灾切换。
- 为了保证数据的可靠性，数据库的数据定期自动备份，在数据丢失时可以快速恢复。

2.11 云服务可靠性介绍

2.11.1 概述

本章节介绍常用云服务的可靠性功能与故障模式，以便应用系统能充分利用云服务提供的可靠性能力，提升应用系统的可靠性，并能针对云服务的常见故障模式，进行故障恢复处理，以便最大限度减少故障，并能从故障中恢复。

2.11.2 ECS 弹性云服务器

弹性云服务器（Elastic Cloud Server，ECS）是由CPU、内存、操作系统、云硬盘组成的基础的计算组件。弹性云服务器创建成功后，就可以像使用自己的本地PC或物理服务器一样，在云上使用弹性云服务器。

2.11.2.1 可靠性功能

数据备份和恢复

使用CBR云备份服务可对ECS的备份保护服务，支持基于多云硬盘一致性快照技术的备份服务，并支持利用备份数据恢复ECS数据。详见“**云备份概述**”。

故障自愈

当ECS支持自动恢复时，可以开启自动恢复能力，当物理服务器损坏时以冷迁移方式重启ECS实例，使弹性云服务器具备高可靠性和强大的动态迁移能力。当弹性云服务器所在的硬件出现故障时，系统会自动将弹性云服务器迁移至正常的物理机，保障业务受到的影响最小，该过程会导致云服务器重启。详见“**物理机故障时，弹性云服务器是否会自动恢复**”。

当检测到弹性云服务所在的硬件出现亚健康时，系统会自动化将弹性云服务器热迁移到其他物理服务器上继续运行，迁移过程中会导致业务处理性能下降，业务中断时间小于1s。

反亲和

通过云服务器组，支持创建ECS实例时尽量分散在不同主机上以提高业务的可靠性。详见“**管理云服务器组**”。

集群 HA

配合共享云硬盘，可以构建AZ内集群或HA关键应用。一块共享云硬盘最多可同时挂载至16台ECS。并需要搭建共享文件系统或类似的集群管理系统，例如Windows MSCS集群、Veritas VCS集群和CFS集群等。

为确保业务可靠性，建议将共享云硬盘挂载至位于同一个反亲和性云服务器组内的ECS。详见“**共享云硬盘及使用方法**”。

负载均衡

配合弹性负载均衡ELB服务，可以实现多ECS实例的负载均衡。

健康检查

通过弹性负载均衡ELB服务，可对ECS实例进行健康检查。详见“**修改健康检查配置**”。

弹性伸缩

配合动态伸缩AS服务，可以实现跨AZ的ECS实例重建或均衡。

跨 AZ 容灾

配合ELB服务，可以实现跨AZ的故障切换。

监控告警

配合CES服务，支持对ECS的CPU、内存、磁盘、网络等进行监控和告警。详见“**监控弹性云服务器**”。

2.11.2.2 常见故障模式

ECS 的 CPU /内存/磁盘容量/磁盘 IOPS 使用率过高

- 检测：通过CES监控CPU/内存/磁盘容量/磁盘IOPS使用率。
- 恢复：
  a. 根据业务情况，手工变更规格以扩展资源或增加ECS实例进行负荷分担。
  b. 对于无状态业务，启动AS弹性伸缩，自动扩展资源。
  c. 应用层进行过载保护，保障优先业务的运行。

连接后端 ECS 失败

- 检测：网络连接失败。
- 恢复：
  a. 至少部署2个后端ECS。对于无状态业务，配置ELB弹性负载均衡保障业务可靠性；对于有状态业务，由应用层实现多实例高可用。

b. 应用层进行重试，以应对暂时性故障，如ESC正在进行故障恢复时。应用故障重试处理可参考“**故障重试**”。

c. 当ECS由于过载导致网络限制时，可参考“ECS的CPU /内存/磁盘容量/磁盘IOPS使用率过高”的处理。

ECS 实例不可用或运行异常

- 检测：配置ELB弹性负载均衡器的后端服务器健康检查，以便定期检查后端服务器的运行状态。健康检查应检查关键功能是否能正确响应。
- 恢复：针对每个应用层，配置多个ECS实例，通过ELB弹性负载均衡器进行健康检查，当检测到某个ECS实例不可用时，ELB弹性负载均衡器停止向该实例发送业务请求。

ECS 实例或挂载的磁盘或数据被意外删除

- 检测：NA
- 恢复：对于无状态业务，使用模板快速发放新实例；对于有状态业务，使用CBR云备份服务对ECS进行定期备份，在数据被删除时使用备份数据快速恢复。

ECS 实例使用本地盘时本地盘故障

- 检测：应用层检测本地盘运行状态。
- 恢复：应用层采用RAID实现ECS内硬盘高可用，并实现跨ECS的数据复制与高可用，以便在本地盘故障时业务可快速恢复。建议非必须使用本地盘场景，尽可能使用EVS云硬盘，以提升硬盘的可靠性。

2.11.3 BMS 裸金属服务

裸金属服务（Bare Metal Server，BMS）是一款兼具弹性云服务器和物理机性能的计算类服务，为企业提供专属的云上物理服务器，为核心数据库、关键应用系统、高性能计算、大数据等业务提供卓越的计算性能以及数据安全。

2.11.3.1 可靠性功能

数据备份和恢复

使用CBR云备份服务可对BMS的所有云硬盘（系统盘和数据盘）进行备份，支持基于多云硬盘一致性快照技术的备份服务，并支持利用备份数据恢复裸金属服务器数据，最大限度保障用户数据的安全性和正确性，确保业务安全。详见“**备份裸金属服务器**”。

集群 HA

配合共享云硬盘，可以构建AZ内集群或HA关键应用。一块共享云硬盘最多可同时挂载至16台BMS。并需要搭建共享文件系统或类似的集群管理系统，例如Windows MSCS集群、Veritas VCS集群和CFS集群等。

详见“**共享云硬盘及使用方法**”。

负载均衡

配合弹性负载均衡ELB服务，可以实现多BMS实例的负载均衡。

健康检查

通过弹性负载均衡ELB服务，可对BMS实例进行健康检查。详见“**修改健康检查配置**”。

跨 AZ 容灾

配合ELB服务，可以实现跨AZ的故障切换。

监控告警

配合CES服务，支持对BMS的CPU、内存、磁盘、网络等进行监控和告警。详见“**监控指标说明**”。

2.11.3.2 常见故障模式

BMS 的 CPU /内存/磁盘容量/磁盘 IOPS 使用率过高

- 检测：通过CES监控CPU/内存/磁盘容量/磁盘IOPS使用率
- 恢复：
  a. 根据业务情况，更换规格更高的BMS实例或增加BMS实例进行负荷分担。
  b. 应用层进行过载保护，保障优先业务的运行。

连接后端 BMS 失败

- 检测：网络连接失败。
- 恢复：
  a. 至少部署2个后端BMS。对于无状态业务，配置ELB弹性负载均衡保障业务可靠性；对于有状态业务，由应用层实现多实例高可用。
  b. 应用层进行重试，以应对暂时性故障，如网络过载时；应用故障重试处理可参考“**故障重试**”。
  c. 当BMS由于过载导致网络限制时，可参考“BMS的CPU /内存/磁盘容量/磁盘IOPS使用率过高”的处理。

BMS 实例不可用或运行异常

- 检测：配置ELB弹性负载均衡器的后端服务器健康检查，以便定期检查后端服务器的运行状态。健康检查应检查关键功能是否能正确响应。
- 恢复：针对每个应用层，配置多个BMS实例，通过ELB弹性负载均衡器进行健康检查，当检测到某个BMS实例不可用时，ELB弹性负载均衡器停止向该实例发送业务请求。

BMS 实例或挂载的磁盘或数据被意外删除

- 检测：NA
- 恢复：对于无状态业务，使用模板快速发放新实例；对于有状态业务，使用CBR云备份服务对BMS云硬盘进行定期备份，在数据被删除时使用备份数据快速恢复。

BMS 实例物理服务器或本地盘故障

- 检测：应用层检测物理服务器和本地盘运行状态
- 恢复：应用层采用RAID实现BMS内硬盘高可用，并实现跨BMS的数据复制与高可用，以便在物理服务器或本地盘故障时业务可快速恢复。建议非必须使用本地盘场景，尽可能使用EVS云硬盘，以提升硬盘的可靠性。

2.11.4 CCE 云容器引擎

云容器引擎（Cloud Container Engine，简称CCE）提供高度可扩展的、高性能的企业级Kubernetes集群，支持运行Docker容器。借助云容器引擎，可以在云上轻松部署、管理和扩展容器化应用程序。

2.11.4.1 可靠性功能

集群 HA

CCE集群支持3个Master节点高可用部署，确保集群的可靠性。

数据备份和恢复

为满足数据持久化的需求，CCE支持将云硬盘（EVS）创建的存储卷挂载到容器的某一路径下；CCE通过云硬盘EVS服务提供针对云硬盘的快照功能，当数据丢失时，可通过快照将数据完整的恢复到快照时间点。详见“**快照与备份**”。

健康检查

健康检查是指容器运行过程中，根据用户需要，定时检查容器健康状况。若不配置健康检查，如果容器内应用程序异常，Pod将无法感知，也不会自动重启去恢复。最终导致虽然Pod状态显示正常，但Pod中的应用程序异常的情况。

Kubernetes提供了三种健康检查的探针：

- 存活探针：livenessProbe，用于检测容器是否正常，类似于执行ps命令检查进程是否存在。如果容器的存活检查失败，集群会对该容器执行重启操作；若容器的存活检查成功则不执行任何操作。
- 就绪探针：readinessProbe，用于检查用户业务是否就绪，如果未就绪，则不转发流量到当前实例。一些程序的启动时间可能很长，比如要加载磁盘数据或者要依赖外部的某个模块启动完成才能提供服务。这时候程序进程在，但是并不能对外提供服务。这种场景下该检查方式就非常有用。如果容器的就绪检查失败，集群会屏蔽请求访问该容器；若检查成功，则会开放对该容器的访问。
- 启动探针：startupProbe，用于探测应用程序容器什么时候启动了。 如果配置了这类探测器，就可以控制容器在启动成功后再进行存活性和就绪检查， 确保这些存活、就绪探针不会影响应用程序的启动。 这可以用于对启动慢的容器进行存活性检测，避免它们在启动运行之前就被终止。

详见“**设置容器健康检查**”。

反亲和

CCE支持节点反亲和，在创建节点池时，可以指定云服务器组以实现反亲和策略，在同一个云服务组中的云服务器分散在不同主机上，提高业务的可靠性。

CCE支持工作负载与节点之间，及工作负载之间的亲和/反亲和：

- 节点亲和：工作负载部署在指定节点/可用区或不部署在指定节点/可用区。
- 工作负载亲和/反亲和：负载部署在相同节点（就近部署就近路由降低网络消耗），或负载部署在不同节点（减少宕机影响）；

详见“**调度策略（亲和与反亲和）**”。

过载控制

CCE集群支持过载控制，在开启过载控制后，可根据控制节点的资源压力，动态调整请求并发量，维护控制节点和集群的可靠性。详见“**集群过载控制**”。

弹性伸缩

CCE支持工作负载弹性伸缩与节点弹性伸缩：

- 工作负载弹性伸缩：即调度层弹性，主要是负责修改负载的调度容量变化。例如，HPA是典型的调度层弹性组件，通过HPA可以调整应用的副本数，调整的副本数会改变当前负载占用的调度容量，从而实现调度层的伸缩。
- 节点弹性伸缩：即资源层弹性，主要是集群的容量规划不能满足集群调度容量时，会通过弹出ECS或CCI等资源的方式进行调度容量的补充。

两个维度的弹性组件与能力可以分开使用，也可以结合在一起使用，并且两者之间可以通过调度层面的容量状态进行解耦。

详见“**弹性伸缩概述**”。

跨 AZ 容灾

CCE服务支持跨AZ创建或扩展容器集群，工作负载自动在多个AZ间均匀分配。

监控告警

CCE支持配合AOM对集群进行全方位的监控，包括集群、节点、工作负载、容器实例POD等。详见“**监控概述**”。

2.11.4.2 常见故障模式

CCE 集群的 CPU /内存/磁盘容量使用率过高

- 检测：通过AOM监控CCE集群的CPU/内存/磁盘容量使用率。
- 恢复：
  a. 根据业务情况，手工变更集群规格或扩展资源。

CCE 节点的 CPU /内存/磁盘容量/磁盘 IOPS/GPU/GPU 缓存使用率过高

- 检测：通过AOM监控CCE节点的CPU/内存/磁盘容量/磁盘IOPS/GPU/GPU缓存使用率。
- 恢复：
  a. 根据业务情况，手工变更节点规格或增加节点数量。

CCE 工作负载的 CPU /内存/GPU/GPU 缓存使用率过高

- 检测：通过AOM监控CCE工作负载的CPU/内存/GPU/GPU缓存使用率。
- 恢复：
  a. 根据业务情况，手工调整工作负载的资源配额或增加工作负载个数。

2.11.5 ELB 弹性负载均衡

ELB弹性负载均衡是将访问流量根据分配策略分发到后端多台服务器的流量分发控制服务，支持独享型负载均衡与共享型负载均衡：

- 独享型负载均衡：独享型负载均衡实例资源独享，实例的性能不受其它实例的影响，可根据业务需要选择不同规格的实例。
- 共享型负载均衡：属于集群部署，实例资源共享，实例的性能会受其它实例的影响，不支持选择实例规格。

2.11.5.1 可靠性功能

集群 HA

ELB采用集群化部署，支持多可用区的同城多活容灾，无缝实时切换。

后端服务器健康检查

ELB弹性负载均衡支持定期向后端服务器发送请求以测试其运行状态。当判断后端服务器健康检查异常后，就不会将流量分发到异常后端服务器，而是分发到健康检查正常的后端服务器，从而提高了业务的可靠性。当异常的后端服务器恢复正常运行后，负载均衡器会将其自动恢复到负载均衡服务中，承载业务流量。

详见“**健康检查介绍**”。

跨 AZ 容灾

ELB采用集群化部署，支持多可用区的同城多活容灾，无缝实时切换。

ELB支持后端服务器多AZ部署，当某个AZ出现故障时，ELB仍可将流量转发到其他AZ的后端ECS处理，提高应用系统容灾能力。

监控告警

配合CES服务，支持对ELB的连接数、带宽、错误响应等进行监控和告警。详见“**监控指标说明**”。

2.11.5.2 常见故障模式

ELB 的并发连接数/新建连接数/带宽使用率过高

- 检测：通过CES监控ELB的并发连接数/新建连接数/带宽使用率。
- 恢复：
  a. 根据业务情况，采用独享型负载均衡器，并手工调整ELB负载均衡器规格。

2.11.6 AS 弹性伸缩

弹性伸缩（Auto Scaling，以下简称AS）是根据用户的业务需求，通过设置伸缩规则来自动增加/缩减业务资源。当业务需求增长时，AS自动增加弹性云服务器（ECS）实例或带宽资源，以保证业务能力；当业务需求下降时，AS自动缩减弹性云服务器（ECS）实例或带宽资源，以节约成本。AS支持自动调整弹性云服务器和带宽资源。

2.11.6.1 可靠性功能

负载均衡

配合弹性负载均衡ELB服务，可以对弹性伸缩组创建的弹性云服务器进行负载均衡。

健康检查

健康检查会将异常的实例从伸缩组中移除，伸缩组会重新创建新的实例以维持伸缩组的期望实例数和当前实例数保持一致，伸缩组的健康检查方式主要包括以下两种。

- 云服务器健康检查：是指对云服务器的运行状态进行检查，如关机、删除都是云服务器异常状态。伸缩组的健康检查方式默认是“云服务器健康检查”方式，指伸缩组会定期使用云服务器健康检查结果来确定每个云服务器的运行状况。如果未通过云服务器健康检查，则伸缩组会将该云服务器移出伸缩组。
- 弹性负载均衡健康检查：是指根据ELB对云服务器的健康检查结果进行的检查。仅当伸缩组使用弹性负载均衡器时，可以选择“弹性负载均衡健康检查”方式来做健康检查。如果将多个负载均衡器添加到伸缩组，则只有在所有负载均衡器均检测到云服务器状态为正常的情况下，才会认为该弹性云服务器正常。否则只要有一个负载均衡器检测到云服务器状态异常，伸缩组会将该弹性云服务器移出伸缩组。

以上两种健康检查方式，检查的结果均是将异常的云服务器从伸缩组中移除。详见“**弹性伸缩健康检查**”。

跨 AZ 容灾

AS支持后端服务器多AZ部署，当某个AZ出现故障时，AS可自动将云服务器创建到其他AZ，以快速恢复业务。

当选择多AZ部署时，可配置“多可用区扩展策略”为“均衡分布”或“选择优先”：

- 均衡分布：云服务器扩容时优先保证选择的可用区列表中各可用区下云服务器数量均衡，当无法在目标可用区下完成云服务器扩容时，按照选择优先原则选择其他可用区。
- 选择优先：云服务器扩容时目标可用区的选择按照选择的可用区列表的顺序进行优先级排序。

监控告警

配合CES服务，支持对弹性伸缩组的CPU、内存、磁盘、网络等进行监控和告警。详见“**监控指标说明**”。

2.11.6.2 常见故障模式

弹性伸缩失败

- 检测：查看弹性伸缩组的弹性伸缩活动历史。
- 恢复：
  a. 根据伸缩活动失败描述信息进行修复。

2.11.7 DCS 分布式缓存服务

分布式缓存服务（Distributed Cache Service，简称DCS）是华为云提供的一款兼容Redis的高速内存数据处理引擎，可提供即开即用、安全可靠、弹性扩容、便捷管理的在线分布式缓存能力，满足用户高并发及数据快速访问的业务诉求。

2.11.7.1 可靠性功能

集群 HA

DCS服务提供主备、Proxy集群、Cluster集群实例，通过节点冗余方式实现实例容灾，当检测到主节点故障后，快速切换到备节点并自动恢复，在异常检测和恢复期间，可能会影响业务，时间在半分钟内。

数据备份和恢复

DCS支持将当前时间点的实例缓存数据备份并存储到OBS中，以便在缓存实例发生异常后能够从备份数据进行恢复。DCS实例支持定时和手动两种备份方式，定时备份频率以天为单位，最多保存7天，但至少会保留一个数据备份文件；手动备份由用户触发，通常在执行业务系统维护、升级等高危操作进行，保存期限无限制。

DCS指定备份集恢复。恢复过程中，实例会有一段时间不能处理客户端的数据操作请求，当前数据将被删除，待恢复完成后存储原有备份数据。

详见“**备份与恢复说明**”。

跨 AZ 容灾

DCS提供的主备、Cluster集群、Proxy集群实例支持跨AZ容灾，当一个AZ异常时，另一个AZ节点不受影响，备节点会自动升级为主节点，对外提供服务。

监控告警

配合CES服务，支持对DCS的CPU、内存、磁盘、网络等进行监控和告警。详见“**支持的监控指标**”。

2.11.7.2 常见故障模式

DCS 的 CPU /内存/带宽/连接数使用率过高

- 检测：通过CES监控CPU /内存/带宽/连接数使用率。
- 恢复：

a. 根据业务情况，手工变更规格以扩展资源。
b. 应用层进行过载保护，保障优先业务的运行，如将部分性能要求不高的业务切回到原始数据源。

连接后端 DCS 失败

- 检测：连接失败。
- 恢复：
  a. 应用层进行重试，以应对暂时性故障，如DCS实例正在进行主备切换时；应用故障重试处理可参考“**故障重试**”。
  b. 当DCS实例由于过载导致网络限制时，可参考“DCS的CPU /内存/带宽/数据库连接数使用率过高”的处理。
  c. 对于非暂行性故障，应用层需要能回退到原始数据源进行处理，避免由于缓存故障而导致业务无法运行。

读写 DCS 概率性失败

- 检测：读写失败。针对低概率超时错误，是Redis使用的正常现象。Redis使用受到网络传输、客户端设置超时时间等因素影响，可能出现单个请求超时问题。
- 恢复：
  a. 应用层进行重试，以应对暂时性故障，如DCS实例正在进行主备切换时；应用故障重试处理可参考“**故障重试**”。
  b. 当DCS实例由于过载导致网络限制时，可参考“DCS的CPU /内存/带宽/数据库连接数使用率过高”的处理。
  c. 对于非暂行性故障，应用层需要能回退到原始数据源进行处理，避免由于缓存故障而导致业务无法运行。

2.11.8 DMS 分布式消息服务

DMS分布式消息服务支持以下各种消息类型：

- Kafka版：基于开源社区版Kafka提供的消息队列服务，向用户提供计算、存储和带宽资源独占式的Kafka专享实例。
- RabbitMq版：完全兼容开源RabbitMQ，提供即开即用、消息特性丰富、灵活路由、高可用、监控和告警等特性，广泛应用于秒杀、流控、系统解耦等场景。
- RocketMQ版：低延迟、弹性高可靠、高吞吐、动态扩展、便捷多样的消息中间件服务。

2.11.8.1 可靠性功能

集群 HA

Kafka实例通过副本冗余方式实现实例容灾，当检测到leader副本故障后，快速完成副本选主，保障Kafka实例持续提供服务。

RabbitMQ集群提供镜像队列，通过镜像在其他节点同步数据。单节点宕机时，仍可通过唯一的访问地址对外提供服务。

RocketMQ使用一主两备架构，备节点通过数据同步的方式保持数据一致。当节点故障时，通过Raft协议自动切换主备关系，保持数据强一致性。

跨 AZ 容灾

Kafka、RabbitMQ、RocketMQ实例支持跨AZ容灾部署，要求至少3个AZ，当一个AZ异常时，不影响实例持续提供服务。

监控告警

Kafka：配合CES服务，支持对Kafka实例、实例节点、实例主题、实例分区、实例分区的消费组、实例队列的消费组、实例的消费组等进行监控和告警。详见“**支持的监控指标**”。

RabbitMQ：配合CES服务，支持对RabbitMQ实例、实例节点、实例队列进行监控和告警等进行监控和告警。详见“**支持的监控指标**”。

RocketMQ：配合CES服务，支持对RocketMQ实例、实例节点、实例队列、实例消费组、实例队列消费组、实例的死信队列进行监控和告警。详见“**支持的监控指标**”。

2.11.8.2 常见故障模式

CPU /内存/磁盘/带宽使用率过高

- 检测：通过CES监控CPU /内存/磁盘/带宽使用率。
- 恢复：
  a. 当CPU/内存使用高时，可根据业务情况，手工修改代理规格或增加代理数量以扩展资源。
  b. 当磁盘使用率高时，可根据业务情况，修改实例存储空间支持更大存储空间。
  c. 当带宽使用率高时，可根据业务情况，变更规格以支持更大带宽。
  d. 应用层进行过载保护，保障优先业务的运行。

生产消息失败

- 检测：生产消息失败
- 恢复：
  a. 应用层进行重试，以应对暂时性故障；应用故障重试处理可参考“**故障重试**”。
  b. 当多次重试后仍无法写入成功，可将数据写入本地缓存，待服务可用后再写入实例。
  c. 当实例由于过载导致网络限制时，可参考“CPU /内存/带宽使用率过高”的处理。

2.11.9 RDS 云数据库

云数据库RDS（Relational Database Service，简称RDS）是一种基于云计算平台的稳定可靠、弹性伸缩、便捷管理的在线云数据库服务。

2.11.9.1 可靠性功能

集群 HA

RDS服务支持HA主备高可用架构，故障秒级自动切换。

数据持久性

RDS数据持久性高达99.9999999%，保证数据安全可靠，保护业务免受故障影响。

数据备份和恢复

RDS支持每天自动备份数据，备份都是以压缩包的形式自动存储在对象存储服务（Object Storage Service，简称OBS）。备份文件保留732天，支持一键式恢复。用户可以设置自动备份的周期，还可以根据自身业务特点随时发起备份，选择备份周期、修改备份策略。

支持按备份集和指定时间点的恢复。在大多数场景下，用户可以将732天内任意一个时间点的数据恢复到云数据库RDS新实例或已有实例上，数据验证无误后即可将数据迁回云数据库RDS主实例，完成数据回溯。

详见“**备份原理及方案**”。

存储自动扩容

RDS支持存储空间自动扩容，在实例存储空间达到阈值时，会触发自动扩容。详见“**存储空间自动扩容**”。

跨 AZ 容灾

RDS支持跨AZ高可用。当用户购买实例的时候，选择主备实例类型，可以选择主可用区和备可用区不在同一个可用区（AZ）。详见“**云数据库RDS支持跨AZ高可用吗**”。

RDS for MySQL也支持通过数据迁移服务DRS支持Region内跨AZ双主灾备，与跨Region容灾相同。

跨 Region 容灾

RDS支持使用数据复制服务（Data Replication Service，简称DRS）创建灾备任务，当主实例所在区域发生突发性自然灾害等状况，主节点（Master）和备节点（Slave）均无法连接时，可将异地灾备实例切换为主实例，在应用端修改数据库链接地址后，即可快速恢复应用的业务访问。数据复制服务提供的实时灾备功能，可实现主实例和跨区域的灾备实例之间的单主灾备（详见“**MySQL到MySQL单主灾备**”）或双主灾备（详见“**MySQL到MySQL双主灾备**”）。

监控告警

配合CES服务，支持对RDS的CPU、内存、磁盘、网络等进行监控和告警。详见“**支持的监控指标**”。

2.11.9.2 常见故障模式

RDS 的 CPU /内存/磁盘容量/磁盘 IOPS/数据库连接数使用率过高

- 检测：通过CES监控CPU /内存/磁盘容量/磁盘IOPS/数据库连接数使用率。
- 恢复：
  a. 根据业务情况，手工变更规格以扩展资源。
  b. 开启存储空间自动扩容，以便在磁盘容量不足时自动扩容。
  c. 应用层进行过载保护，保障优先业务的运行。

连接后端 RDS 失败

- 检测：连接失败。
- 恢复：
  a. 应用层进行重试，以应对暂时性故障，如RDS实例正在进行主备切换时；应用故障重试处理可参考“**故障重试**”。
  b. 当RDS实例由于过载导致网络限制时，可参考“RDS的CPU /内存/磁盘容量/磁盘IOPS/数据库连接数使用率过高”的处理。

2.11.10 GaussDB(for MySQL)云数据库

云数据库 GaussDB(for MySQL)是华为自研的最新一代企业级高扩展高性能分布式数据库，完全兼容MySQL。基于华为最新一代DFV存储，采用计算存储分离架构，128TB的海量存储，故障秒级切换，既拥有商业数据库的高可用和性能，又具备开源低成本效益。

2.11.10.1 可靠性功能

集群 HA

GaussDB(for MySQL)服务支持主节点+只读节点的高可用架构，当主节点故障时，系统会自动切换到只读节点，只读节点提升为主节点，原来故障的主节点也会自动恢复为只读节点。

GaussDB(for MySQL)服务还支持异构容灾实例(MySQL节点)，支持在极端场景，如社区未知bug、用户误操作、AZ级故障导致服务无法正常提供服务等场景，可以快速将服务切换到异构容灾实例继续提供服务。详见“**异构容灾实例**”。

数据备份和恢复

GaussDB(for MySQL)实例支持自动备份和手动备份，您可以定期对数据库进行备份，当数据库故障或数据损坏时，可以通过备份文件恢复数据库，从而保证数据可靠性。

GaussDB(for MySQL)支持同区域备份与跨区域备份；跨区域备份是将备份文件存放到另一个区域存储，某一区域的实例故障后，可以在异地区域使用备份文件在异地恢复到新的GaussDB(for MySQL)实例，用来恢复业务。

详见“**备份原理**”。

自动扩缩容

GaussDB(for MySQL)服务支持自动扩缩容，可自动扩缩规格和增删只读节点。详见“**设置自动变配（自动扩缩容）**”。

跨 AZ 容灾

GaussDB(for MySQL)实例支持将实例的节点分别部署在多个可用区。

GaussDB(for MySQL)也支持通过数据迁移服务DRS支持Region内跨AZ双主灾备，与跨Region容灾相同。

跨 Region 容灾

GaussDB(for MySQL)支持使用数据复制服务（Data Replication Service，简称DRS）创建灾备任务，当主实例所在区域发生突发性自然灾害等状况，主节点（Master）和备节点（Slave）均无法连接时，可将异地灾备实例切换为主实例，在应用端修改数据库链接地址后，即可快速恢复应用的业务访问。数据复制服务提供的实时灾备功能，可实现主实例和跨区域的灾备实例之间的单主灾备（详见“**GaussDB(for MySQL)到GaussDB(for MySQL)单主灾备**”），或双主灾备（详见“**GaussDB(for MySQL)到GaussDB(for MySQL)双主灾备**”）。

监控告警

配合CES服务，支持对GaussDB(for MySQL)的CPU、内存、磁盘、网络等进行监控和告警。详见“**支持的监控指标**”。

2.11.10.2 常见故障模式

GaussDB(for MySQL)的 CPU /内存/磁盘容量/磁盘 IOPS/数据库连接数使用率过高

- 检测：通过CES监控CPU /内存/磁盘容量/磁盘IOPS/数据库连接数使用率。
- 恢复：
  a. 根据业务情况，手工变更规格以扩展资源。
  b. 开启自动扩缩容，以便在过载时自动扩容规格和/或只读节点。
  c. 应用层进行过载保护，保障优先业务的运行。

连接后端 GaussDB(for MySQL)失败

- 检测：连接失败。
- 恢复：
  a. 应用层进行重试，以应对暂时性故障，如GaussDB(for MySQL)实例正在进行主备切换时；应用故障重试处理可参考“**故障重试**”。
  b. 当GaussDB(for MySQL)实例由于过载导致网络限制时，可参考“RDS的CPU /内存/磁盘容量/磁盘IOPS/数据库连接数使用率过高”的处理。

2.11.11 OBS 对象存储服务

对象存储服务（Object Storage Service，OBS）是一个基于对象的海量存储服务，提供海量、安全、高可靠、低成本的数据存储能力。

2.11.11.1 可靠性功能

数据持久性

OBS通过存储介质的慢盘/坏道检测、AZ内设备和数据冗余、AZ之间数据容灾、跨区域复制等技术方案，提供针对介质、服务器、机柜、数据中心和区域的多级可靠性保障。其数据持久性高达99.9999999999%（12个9），可用性高达99.995%，远高于传统架构。详见“**OBS的持久性和可用性如何?**”。

数据备份和恢复

OBS支持多版本控制，可以在一个桶中保留多个版本的对象，以便方便地检索和还原各个版本，在意外操作或应用程序故障时快速恢复数据。

跨 AZ 容灾

在创建桶时，数据冗余存储策略可选择多AZ存储，数据将冗余存储至多个AZ中，可靠性更高。

跨 Region 容灾

OBS支持跨区域复制，能够为用户提供跨区域数据容灾的能力，满足用户数据复制到异地进行备份的需求。

监控告警

配合CES服务，支持对OBS桶的请求、流量、时延和错误响应等进行监控和告警。详见“**监控对象存储服务**”。

2.11.11.2 常见故障模式

OBS 桶流量过载

- 检测：通过CES监控请求数、请求成功率、上传/下载带宽等流量指标。
- 恢复：
  a. 应用层调整批量业务，避免业务高峰期进行备份等业务；
  b. 应用层进行重试，以应对暂时性故障，如网络拥塞；应用故障重试处理可参考“**故障重试**”。
  c. 应用层进行过载保护，保障优先业务的运行。

OBS 对象上传/下载失败

- 检测：对象上传/下载失败。
- 恢复：
  a. 应用层进行重试，以应对暂时性故障，如网络拥塞；应用故障重试处理可参考“**故障重试**”。
  b. 当OBS桶由于过载导致网络限制时，可参考“OBS桶流量过载”的处理。

OBS 桶内数据被误删

- 检测：NA
- 恢复：针对OBS桶启用多版本控制，在数据被删除时使用历史版本快速恢复。

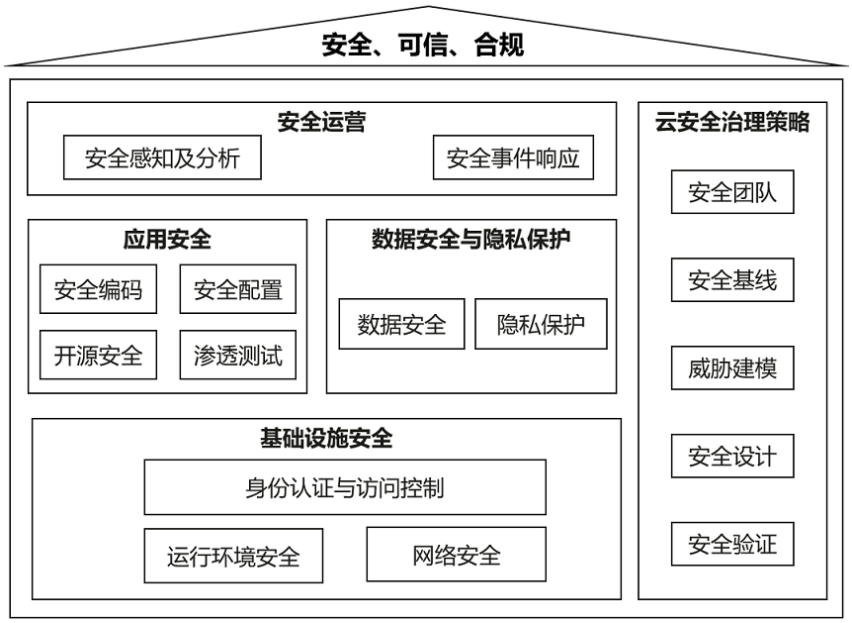
3 安全性支柱

3.1 概述

3.1.1 安全性支柱简介

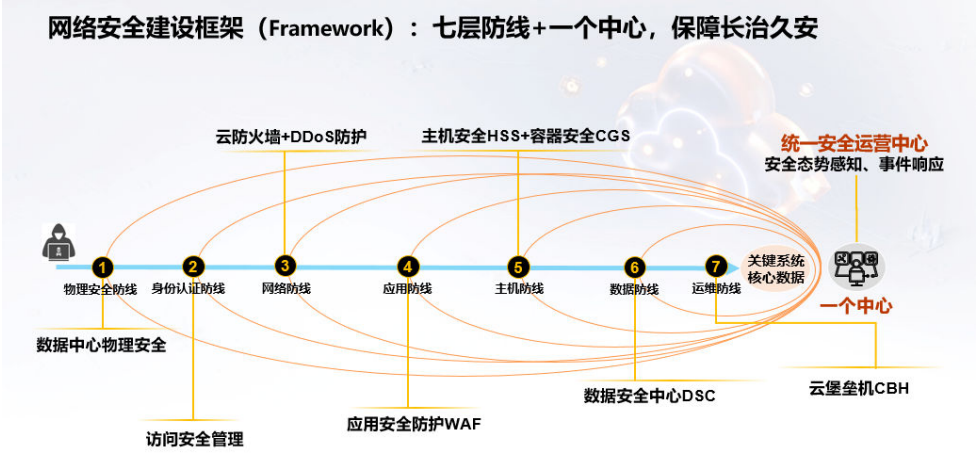
华为将安全及隐私保护作为公司的最高纲领。安全性支柱旨在确保业务的安全、可信、合规，通过一系列华为云架构的最佳实践保护工作负载免受各种安全威胁，降低安全风险。安全性支柱涉及保护云上系统、资产、数据的机密性、完整性、可用性以及合法、合规使用数据，保护用户隐私的一系列最佳实践。

安全性是现代应用程序的重要维度，需要成体系地考虑工作负载的安全。华为云安全性支柱的设计框架如下图所示：

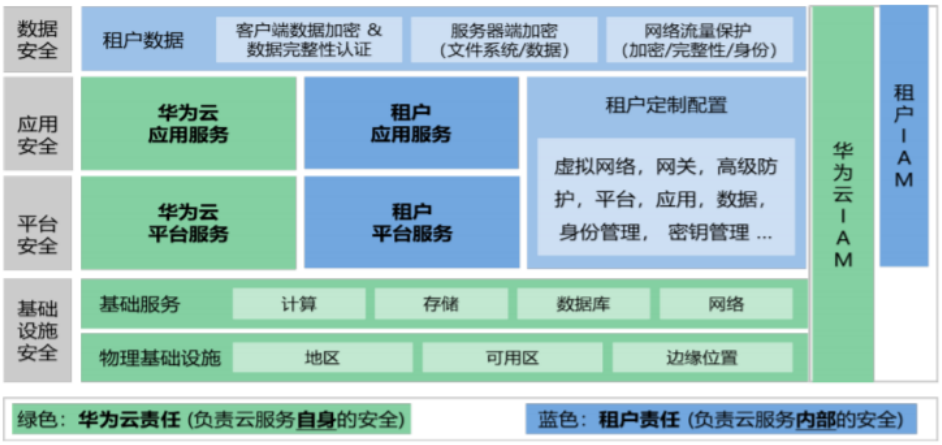


3.1.2 责任共担模型

基于华为在安全、合规、隐私及数据保护领域积累多年的技术和治理能力，华为云为您提供安全、可靠、可信赖的基础设施和服务。华为云提出“七层防线+一个中心”的网络安全建设框架，通过多重、多方面的安全防线来成体系保障云上业务的安全性。



华为云把安全合规作为首要任务，安全是华为云和您之间的共同责任。在云服务模式下，华为云与客户共同承担云环境的安全保护责任，为明确双方的责任，确定责任边界，华为云制定了责任共担模型。华为云负责云的安全性，华为云客户负责云上的安全性。



详细内容见：**华为云责任共担模型**

3.2 基本概念

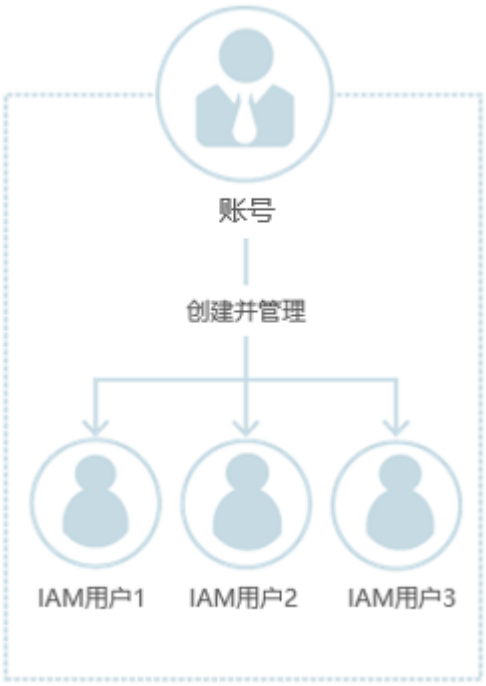
3.2.1 概念表

概念	解释
机密性	确保只有授权的用户可以访问系统中的敏感信息，防止未经授权的访问和泄露。机密性通常通过加密技术来实现，包括对数据进行加密和解密的过程，确保只有授权用户能够访问和理解数据内容。
完整性	确保数据在传输和存储过程中不被篡改，保持数据的完整性，防止数据被恶意篡改或损坏。完整性通常通过哈希函数和数字签名等技术来实现，确保数据在传输或存储过程中没有被篡改或损坏。
可用性	确保系统和数据在需要时可用，防止因攻击、故障或其他原因导致系统不可用。从安全的角度，可用性可通过负载均衡、弹性计算、事件监控和告警、防暴力攻击如DDoS防护等手段来实现。
可审计	系统或数据处理过程能够被有效地监视、记录和审计的能力。可审计性通常通过审计日志、审计跟踪、监控系统和审计工具等技术来实现，记录系统操作和事件，以便后续审计和监控。
不可抵赖性	在通信或交易过程中，一方无法否认已经发出的消息或行为，也无法否认接收到的消息或行为。不可抵赖性通常通过数字签名、公钥基础设施（PKI）、审计日志和审计跟踪等技术来实现，确保通信双方无法否认其行为或消息。
账号	帐号是您的华为云资源归属、资源隔离、计费的主体，对其所拥有的资源及云服务具有完全的访问权限。每个账号具有独立的身份验证、访问控制和资源隔离，账号之间默认相互隔离。
IAM	统一身份认证（Identity and Access Management，简称IAM）是华为云提供身份认证和权限管理的基础服务，可以帮助您安全地控制云服务和资源的访问权限。
IAM用户	由帐号在IAM中创建的用户，是云服务的使用人员，具有独立的身份凭证（密码和访问密钥），根据帐号授予的权限使用资源。 账号与IAM用户可以类比为父子关系，账号是资源归属以及计费的主体，对其拥有的资源具有所有权限。IAM用户由账号创建，只能拥有账号授予的资源使用权限，账号可以随时修改或者撤销IAM用户的使用权限。
多因素认证	多因素认证（Multi-Factor Authentication，简称MFA）是一种安全认证过程，需要用户提供两种及以上不同类型的认证因子来表明自己的身份，包括密码、指纹、短信验证码、智能卡、生物识别等多种因素组合，从而提高用户账户的安全性。
安全威胁	安全威胁指的是可能导致系统、网络或数据遭受损害、被破坏或被访问的潜在危险因素或事件。安全威胁可以是意外的，也可以是有意的，可能会导致系统遭受攻击或受到损害。安全威胁可以是外部的（如黑客攻击、恶意软件）或内部的（如员工疏忽、内部泄露）。
威胁建模	识别系统的潜在威胁以建立防护策略，构建安全的系统。

概念	解释
安全风险	安全风险是指在面临安全威胁的情况下，系统、网络或数据可能遭受损害或丧失机密性、完整性或可用性的概率和影响程度。安全风险通常由威胁的存在、系统漏洞、不恰当的安全措施或其他因素造成。安全风险通常通过风险评估来评估和管理，以确定风险的程度并采取相应的控制措施。
Playbook 处置剧本	安全响应中的处置剧本是一种预定义的操作指南，旨在帮助安全团队在面对特定的安全事件或威胁时，迅速且有序地采取行动。剧本通常包含详细的步骤、流程、工具和责任分配，以确保安全事件得到有效处理，减少潜在损失和影响。
数据主体	提供个人数据，可以通过个人数据识别或个人数据的组合识别的自然人，对个人数据有疑问时，有投诉或提出质询的权利。如：产品的最终用户，公司的雇员等。
数据控制者	单独或者与他人共同确定个人数据处理的目的和手段的自然人、法人、公共机构、政府部门或其他机构。对个人数据的处理有控制权，承担个人数据保护的主要责任。
数据处理者	代表数据控制者处理个人数据的自然人、法人、公共机构、政府部门或其他机构。数据处理者必须按照数据控制者的要求对个人数据进行充分的保护。
第三方（特指隐私保护中的第三方角色）	指数据主体、数据控制者、数据处理者以及根据数据控制者或者处理者的直接授权而处理数据的人之外的任何自然人或法人、公共权力机关、代理机构或其他机构。
个人数据	个人数据是指与一个身份已被识别或者身份可被识别的自然人（“数据主体”）相关的任何信息。身份可识别的自然人是指其身份可以通过诸如姓名、身份证号、位置数据等识别码或者通过一个或多个与自然人的身体、生理、精神、经济、文化或者社会身份相关的特定因素来直接或者间接地被识别。
敏感个人数据	高影响个人数据的一个子集。指在个人基本权利和自由方面极其敏感，一旦泄露可能会造成人身损害、财务损失、名誉损害、身份盗窃或欺诈、歧视性待遇等的个人数据。

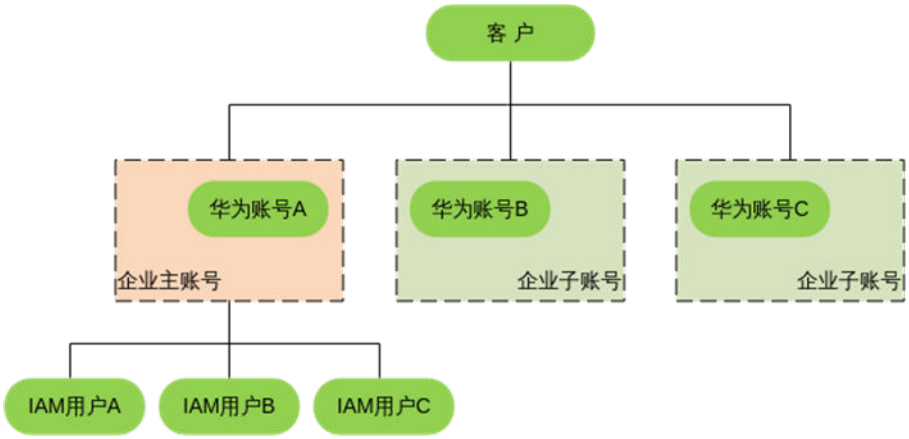
3.2.2 概念模型

华为云的客户在注册账号后，每个账号下可以创建多个IAM用户。



对于大型企业的客户，可能会管理多个账号，这些账号可以被统一管理。客户可通过一个企业主账号结合多个企业子账号来统一管理账号。

主账号与子账号中都可以再创建更小层级的IAM用户，这些IAM用户分别属于对应的账号，可以帮助账号管理资源。华为云企业中心提供了多个相互独立的华为账号之间形成企业主子账号关联关系的能力。



3.3 设计原则

国际标准化组织（ISO）对计算机系统安全的定义为：确保信息资产（包括硬件、软件、网络、数据等）受到保护，以确保其机密性、完整性和可用性。计算机系统安全的目标是保护信息系统免受未经授权的访问、使用、披露、破坏、修改、中断或不可用的威胁，同时确保信息系统能够持续地提供服务。

系统安全的基本要素包括机密性、完整性、可用性、可审计、不可抵赖性等。其中最基本的三个要素是机密性（Confidentiality）、完整性（Integrity）、可用性（Availability），简称CIA。

为实现系统安全所定义的基本要素，业界根据大量的实践，提炼出一些共性的安全设计原则：

- **零信任原则（Zero Trust）**
  - 零信任遵循“永不信任，始终验证”的安全理念，假设任何人或程序都不可信，无论是内部用户、外部用户还是网络设备。系统内的组件进行任何通信之前都将通过显式的验证，减少系统信任带来的攻击面。零信任把现有的基于实体鉴别和默认授权的静态信任模型（非黑即白），变成基于持续风险评估和逐次授权的动态信任模型。
  - 零信任不根据网络空间位置决定可信度，其重心在于保护资源，而不是网段。与传统安全理念对比，它将网络防御的重心从静态的、基于网络的边界转移到了用户、设备和资源上。所有的资源（如人/物/终端/应用/网络/数据/供应链）都需要进行持续身份验证和信任评估，从全局视角执行动态安全策略。零信任通过动态、持续性的实体风险评估，缩小受攻击面，保证系统安全。
- **纵深防御原则（Defense In Depth）**
  - 多点、多重的安全防护机制来分层保护组织的网络、资产和资源。
  - 不依赖单层安全防护能力，不因单一安全防护能力失效而完全暴露。
  - 假设系统受到攻击，系统有一定的韧性能力保持最小化系统运行，可以提供最小化服务。
- **最小化原则（Least Privilege）**
  - 最小化身份：尽可能减少非必要的系统管理员，定时清理过期的身份。
  - 最小化权限：给予用户或实体最小必要权限来执行其工作，以降低潜在的安全风险。
  - 最小化暴露面：对不同的访问区域和访问对象，仅暴露最小的服务端点和最少的服务应用接口。
  - 最小化凭证：尽量消除对长期的、静态凭证的依赖。
- **数据安全保护原则（Data Security）**
  - 数据分类分级，定义不同级别的数据防护措施。
  - 确保对数据进行适当的加密、备份和访问控制，以保护数据的机密性、完整性和可用性。
  - 维护个人隐私权利，保护隐私数据的机密性和完整性。
- **DevSecOps**
  - DevSecOps的核心理念是将安全性纳入到整个软件开发生命周期中，从需求分析、设计、开发、测试、部署、运维、运营的每个阶段都考虑安全性，以确保系统的安全性和稳定性。
  - 通过将安全性与DevOps的自动化流程相结合，DevSecOps可更快地检测和修复安全漏洞，并提高软件开发的效率和质量。

3.4 问题和检查项

问题	检查项/最佳实践
SEC01 如何整体考虑云安全治理策略?	1. 建立安全管理队 2. 建立安全基线 3. 梳理资产清单 4. 分隔工作负载 5. 实施威胁建模分析 6. 识别并验证安全措施
SEC02 如何管理人机接口和机机接口的身份认证?	1. 对账号进行保护 2. 安全的登录机制 3. 安全管理及使用凭证 4. 一体化身份管理
SEC03 如何管理人员和机器的权限?	1. 定义权限访问要求 2. 按需分配合适的权限 3. 定期审视权限 4. 安全共享资源
SEC04 如何进行网络安全设计?	1. 对网络划分区域 2. 控制网络流量的访问 3. 网络访问权限最小化
SEC05 如何进行运行环境的安全设计?	1. 云服务安全配置 2. 实施漏洞管理 3. 减少资源的攻击面 4. 密钥安全管理 5. 证书安全管理 6. 使用托管云服务
SEC06 如何进行应用程序安全设计?	1. 安全合规使用开源软件 2. 建立安全编码规范 3. 实行代码白盒检视 4. 应用安全配置 5. 执行渗透测试
SEC07 如何进行数据安全设计?	1. 识别工作负载内的数据 2. 数据保护控制 3. 对数据操作实施监控 4. 静态数据的加密 5. 传输数据的加密

问题	检查项/最佳实践
SEC08 如何进行数据隐私保护设计?	1. 明确隐私保护策略和原则 2. 主动通知数据主体 3. 数据主体的选择和同意 4. 数据收集合规性 5. 数据使用、留存和处置合规性 6. 向第三方披露个人数据合规性 7. 数据主体有权访问其个人隐私数据
SEC09 如何进行安全感知及威胁检测?	1. 实施标准化管理日志 2. 安全事件记录及分析 3. 实施安全审计 4. 安全态势感知
SEC10 如何进行安全事件的响应?	1. 建立安全响应团队 2. 制定事件响应计划 3. 自动化响应安全事件 4. 安全事件演练 5. 建立复盘机制

3.5 云安全治理策略

3.5.1 SEC01 云安全治理策略

企业安全的最终目标不会随着采用云服务而改变，但实现这些目标的方式将会改变。为了安全地操作、管理您的工作负载，您必须对安全性的各个方面进行总体策略上的考虑。企业的管理层和安全团队需要根据企业总体安全战略和业务战略制定云安全战略，并且需要在计划采用云服务时尽早考虑安全性。 云安全治理策略包括安全团队、安全基线、安全资产、安全建模以及核心的安全控制点。企业需尽早规划和思考如何使用云技术和云服务来实现安全治理的现代化，并通过实施合理的云安全策略，实现云上业务系统的安全、合规。

3.5.1.1 SEC01-01 建立安全管理团队

指定负责工作负载在云环境的安全性、合规性、隐私保护方面的关键角色，确保从责任主体上保障工作负载的安全性。

- **风险等级**

  高

- **关键策略**
  - 明确职责和角色：确定团队成员的职责和角色，包括安全架构设计、安全测试、安全运营等方面的角色。每个角色应清晰定义其职责范围和任务。
  - 跨职能团队：组建一个跨职能的安全管理团队，涵盖安全运营、安全架构、安全合规等不同领域的专业人员，以确保综合性的安全管理。

- 制定安全政策和流程：制定详细的安全政策和流程，明确安全管理的标准和规范。团队成员应遵守这些政策和流程，确保安全管理的一致性和有效性。
- 建立应急响应计划：开发和测试应急响应计划，以应对安全事件和紧急情况。团队应清楚知道如何应对安全威胁和处理安全事件。

3.5.1.2 SEC01-02 建立安全基线

建立符合合规性要求、行业标准和平台建议的安全基线，安全基线是团队内对安全的底线要求。根据基线定期衡量您的工作负载架构和运行情况，持续保持或改善工作负载的安全状况。

- **风险等级**

  高

- **关键策略**

  确定合规性要求：了解您的工作负载必须符合的组织、法律和合规性要求。

- **相关云服务和工具**
  - **华为云合规中心**
  - **华为云信任中心**
  - **华为云等保合规安全解决方案**：华为云依托自身安全能力与安全合规生态，为客户提供一站式的安全解决方案，帮助客户快速、低成本完成安全整改，轻松满足等保合规要求。通过华为30年安全经验积累，结合企业和机构的安全合规与防护需求，来帮助企业与机构满足国家及行业法律法规要求，同时实现对安全风险与安全事件的有效监控，并及时采取有效措施持续降低安全风险，消除安全事件带来的损失。

3.5.1.3 SEC01-03 梳理资产清单

梳理工作负载涉及的服务器、IP地址、域名、数据库、证书等全量云资源的资产清单，给资源打上标签，从而在出现安全事件时，能快速定位到有安全风险的资源。

- **风险等级**

  高

- **关键策略**
  - 设计态与运行态一致性：对照设计态的架构图、架构文档实施云服务资源。工作负载运行时的架构始终保持与设计态一致。
  - 自动化资产盘点：使用安全云服务或工具来自动发现和记录云上资源，包括主机、存储、数据库、网络等。这样可以确保资产清单的及时性和准确性。
  - 标签和元数据：使用标签和元数据来对云资源进行分类和描述，以便更好地组织和管理资源清单。通过标签可以快速识别和过滤资源，有助于监控和安全审计。

- **相关云服务和工具**
  - **解决方案工作台 InnoStageWorkbench**：使用解决方案工作台辅助进行云上架构的可视化设计。
  - 安全云脑 SecMaster：安全云脑支持对云上资产全面自动盘点，也可灵活纳管云外各种资产，点清所有资产，并呈现资产实时安全状态。
  - 配置审计 Config
  - 标签管理服务 TMS

3.5.1.4 SEC01-04 分隔工作负载

分隔工作负载是一种架构上进行分治的思想，通过将整个系统的工作负载分割成更小的部分，每个部分独立运行和管理，从而提高系统的安全性和可维护性。

- **风险等级**

  高

- **关键策略**

  一个企业特别是大型企业往往有多个不同类型（如生产环境、开发环境、测试环境）或不同组织单元（OU）下的工作负载，多个组织单元之间或多个工作负载之间要进行隔离。

  分隔工作负载在云环境中是非常重要的。从安全治理角度，主要基于以下几个理由：

  - 安全性：分隔工作负载可以降低潜在的安全风险。通过将不同的工作负载隔离在独立的环境中，可以减少一种工作负载受到攻击或故障时对其他工作负载的影响。
  - 合规性：在一些行业和法规中，对数据隔离和访问控制有严格要求。通过分隔工作负载，可以更容易地满足合规性要求，保护敏感数据和确保数据隐私。
  - 管理性：通过分隔工作负载，可以更轻松地管理和维护系统。每个工作负载都有独立的配置和管理需求，分隔可以简化管理流程并降低操作风险。
  - 灵活性：分隔工作负载可以提供更大的灵活性和可扩展性。组织可以根据需要调整和扩展不同工作负载的资源，而不会影响其他部分。

  华为云提供了以下几种工作负载的分隔机制：

  - 通过多VPC分隔工作负载：将不同的工作负载部署在不同的VPC中，每个VPC具有独立的网络空间，实现网络隔离。
  - 通过企业项目分隔工作负载：企业项目是云服务资源的逻辑集合，将工作负载部署在不同的企业项目中，实现资源的分组管理和权限控制。
  - 通过多账号分隔工作负载：将不同的工作负载部署在不同的华为云账号中，每个账号具有独立的身份验证、访问控制和资源隔离。这种方法可以实现更严格的隔离和安全性。为每个账号分配最小必要权限，避免权限过度赋予。这有助于减少潜在的安全风险和权限滥用。针对需要跨账号访问的情况，使用适当的身份验证和授权机制，如跨账号委托、资源共享等。
  - 多者结合：同时使用以上的两种或多种方式分隔工作负载。

- **相关云服务和工具**
  - 虚拟私有云 VPC
  - 企业项目 EPS
  - 统一身份认证服务 IAM
  - **华为云Landing Zone解决方案**
  - 组织 Organizations
  - 资源治理中心 RGC
  - 资源访问管理 RAM

3.5.1.5 SEC01-05 实施威胁建模分析

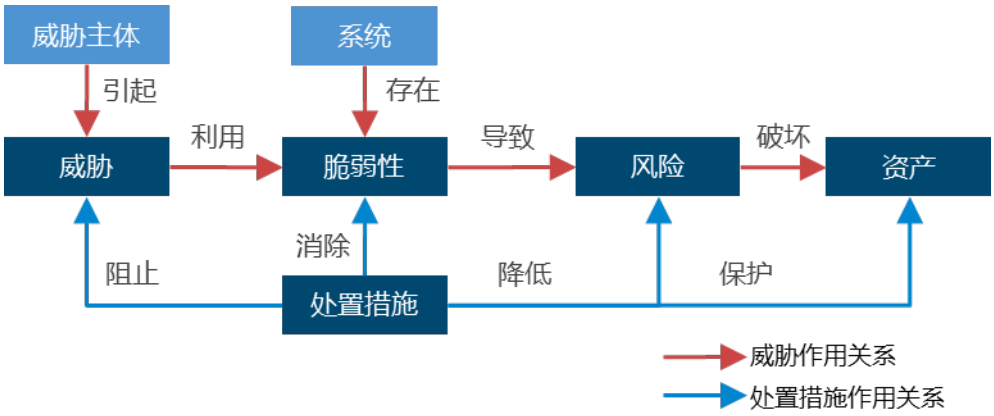
威胁建模是一种系统性的方法，用于识别和评估可能对系统或组织造成威胁的潜在威胁源、攻击路径和攻击手段。通过识别威胁理解系统的安全风险，发现系统设计中的安全问题，制定消减措施，降低系统风险，提升系统安全性和韧性。

- **风险等级**

  高

- **关键策略**

  以下是系统运行期间的威胁模型：



  该模型中涉及的概念如下：

  - 威胁主体：有企图的利用脆弱性的实体称为威胁主体；威胁主体可以是人、程序、硬件或系统。
  - 脆弱性：系统中允许破坏其安全性的缺陷，包括软件、硬件或过程或人为的缺陷。脆弱性的存在，说明了缺少应该使用的安全措施，或安全措施有缺陷。
  - 威胁：利用脆弱性而带来的任何潜在危险。
  - 风险：攻击者利用脆弱性的可能性以及相应的业务影响；风险将脆弱性、威胁和利用可能性与造成的业务影响联系在一起。
  - 资产： 任何对组织有价值的信息或资源，是安全策略保护的对象。
  - 处置措施：包括安全角度的消减措施和韧性角度的增强措施，能够消除脆弱性或者阻止威胁，或者降低风险的影响和保护资产。

  实施威胁建模，需要有攻击者思维，像攻击者一样思考，发现潜在的暴露面/攻击目标及可用的攻击方法，从而发现系统中潜在的安全威胁、 建立相应的消减措施。

  威胁建模的一般步骤如下：

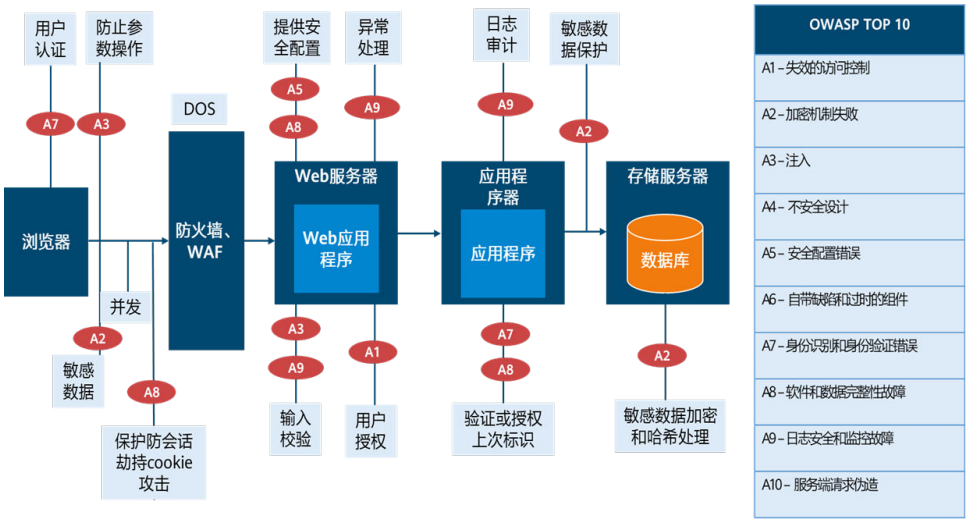
  a. 确定范围：明确要进行威胁建模的云上系统范围，包括云服务、数据存储、网络架构等。
  b. 收集信息：收集关于云上系统的信息，包括系统架构图、数据流程、访问控制策略等。
  c. 识别资产：确定在云上系统中的关键资产，包括数据、应用程序、虚拟机、存储等。
  d. 识别威胁源和攻击路径：确定可能对云上系统构成威胁的威胁源和攻击路径，考虑不同攻击者可能采取的攻击手段。

e. 评估威胁概率和影响：评估每种威胁的概率和可能造成的影响，包括数据泄露、服务中断等。

f. 制定安全对策：根据识别的威胁，制定相应的安全对策和控制措施，包括访问控制、加密、监控等。

g. 持续改进：定期检视和更新威胁模型，以反映新的威胁和安全风险，确保云上系统的安全性得到持续改进。

以下是OWASP总结的Web应用系统TOP10的威胁及处置措施：



- **相关云服务和工具**
  - **解决方案工作台 InnoStageWorkbench**：使用解决方案工作台辅助进行云上架构图的可视化设计，基于架构图进行威胁分析。

3.5.1.6 SEC01-06 识别并验证安全措施

根据团队制定的安全基线以及威胁建模分析的结果，对工作负载中涉及的安全措施进行验证，以确保它们按照预期方式运行并有效地保护系统，从而缓解或消除安全威胁。

- **风险等级**

  高

- **关键策略**
  - 依据系统的安全设计文档，通过验证确保安全措施被正确地集成到系统中，并符合最佳实践和标准。
  - 尽早检视系统的代码（此过程称为代码白盒安全检视），确保代码符合安全最佳实践，避免在后续阶段发现严重的安全漏洞。
  - 利用安全测试工具进行静态代码分析、动态代码分析、漏洞扫描等测试，以发现潜在的安全问题。
  - 使用模拟攻击工具或技术，尝试模拟攻击者的行为，以评估系统的安全性和弱点。

3.6 基础设施安全

3.6.1 SEC02 身份认证

3.6.1.1 SEC02-01 对账号进行保护

账号是华为云租户的账号体系中权限最高的用户，拥有对整个云环境的最高权限。一旦账号受到攻击或泄露，可能导致严重的安全问题和数据泄露。因此，身份认证的安全性首先要考虑对此账号进行保护。

- **风险等级**

  高

- **关键策略**
  - 强密码：使用强密码来保护账号，包括数字、字母、特殊字符的组合，并确保密码足够长且复杂。
  - 多因素认证（MFA）：启用多因素认证为保护账号提供了额外的安全层次。除了密码之外，MFA需要额外的身份验证信息，提高了账号的安全性。
  - 限制日常操作：避免直接使用账号进行日常操作，而是创建并使用IAM用户进行日常的管理操作。账号应仅用于关键操作，如创建新的IAM用户或修改权限。
  - 优先使用临时凭证并定期轮换凭证：定期更改账号的密码，并定期更新MFA设备。这有助于减少被猜测或盗用的风险。
  - 启用审计日志：启用审计日志功能，以监控账号的活动。审计日志可以帮助检测异常行为并及时采取措施。
  - 多账号管理场景：需指定一个账号作为中央账号（企业主账号），由这个账号再添加成员账号（企业子账号）。优先保证中央账号的安全，再考虑成员账号。
- **相关云服务和工具**
  - 统一身份认证服务 IAM
  - 组织 Organizations
  - 企业管理
  - 云审计服务 CTS

3.6.1.2 SEC02-02 安全的登录机制

将安全的登录机制用于账号、IAM用户以及对接第三方身份提供商。

- **风险等级**

  高

- **关键策略**
  - 除了账号，确保IAM管理员（有管理员权限的IAM用户）也开启MFA机制登录，避免登录凭证泄露带来的风险。
  - 配置IAM的登录验证策略，如会话超时策略、账号锁定策略、账号停用策略、最近登录提示等。
  - 配置IAM的网络访问控制策略。限制用户只能从特定 IP 地址区间、网段及 VPC Endpoint 访问华为云。
  - 多个账号或多个IAM用户间使用不同的密码。
  - 禁止将用户的密码共享给其他人，而是为每个管理或使用华为云资源的人创建一个单独的用户。

  - 修改新用户的默认密码。使用IAM创建新用户时，可通过邮件发送一次性登陆链接给新用户，新用户使用链接进行登陆时需要设置密码，另外在管理员自定义新用户的密码时可选择强制用户在激活后修改默认密码。
  - 集中的身份管控：
    - 使用单点登录：考虑使用单点登录解决方案，集中管理用户的身份认证信息，简化用户登录流程，提高安全性和用户体验。
    - 多账号场景，对账号的集中管控。
- **相关云服务和工具**
  - **IAM身份提供商**
  - **华为账号使用的安全最佳实践**
  - 应用身份管理服务 OneAccess：使用OneAccess与您组织的HR系统关联实现单点登录。

3.6.1.3 SEC02-03 安全管理及使用凭证

在进行身份验证时，首要选择使用临时凭证而非长期或永久性凭证，以减少或消除因凭证意外泄露、共享或被盗而带来的风险。

- **风险等级**

  高
- **关键策略**
  - 长期凭证如用户的登录密码、永久AK/SK，短期凭证如临时AK/SK、通过委托获取的权限等。禁止将长期凭证硬编码到代码中，以免泄露。优先使用临时凭证调用华为云的SDK或API。
  - 如果某些情况下不能选择临时凭证，才使用长期凭证。在此情况下，建议将长期凭证放置到代码之外的文件或由第三方托管，将长期凭证作为变量传入使用。要定期审计和实施凭证轮换，以帮助降低长期凭证相关风险。
  - 对您的身份提供者和IAM中配置的身份进行审计，这有助于验证只有经过授权的身份才能访问您的工作负载。
  - 使用数据加密服务DEW托管凭据。实现对数据库账号口令、服务器口令、SSH Key、访问密钥等各类型凭据的统一管理、检索与安全存储。
  - 使用数据加密服务DEW中的凭据管理服务（CSMS）定期轮换凭证。
  - 使用IAM委托。委托操作权限给云服务或者其它账号。
- **相关云服务和工具**
  - 数据加密服务 DEW
  - 统一身份认证服务 IAM

3.6.1.4 SEC02-04 一体化身份管理

在公司范围内构建统一的身份管理系统，统一管理私有云和公有云、公有云上多个账号的用户身份。

- **风险等级**

  中
- **关键策略**

  - 在公司范围内构建统一身份管理系统，集中存储用户身份信息。
  - 统一身份管理系统与私有云、公有云平台的IAM系统进行身份联邦，统一身份管理系统中的用户身份可以同时访问私有云和公有云平台。
  - 统一身份管理系统与公司的HR流程结合，当员工入职、调岗和离职时可以触发用户的创建、变更和删除。
  - 针对Landing Zone搭建的云上多账号环境，利用IAM身份中心集中管理多个账号的用户身份，并集中为这些用户配置能够访问多个账号下云资源的权限，无需在每个账号的IAM系统分别创建IAM用户并配置权限，简化多账号环境下身份权限管理的工作量。
  - 统一身份管理系统与IAM身份中心建立身份联邦，这样无需分别与每个账号的IAM系统进行身份联邦。
- **相关云服务和工具**
  - IAM身份中心 IAM Identity Center
  - 统一身份认证服务 IAM
  - 应用身份管理服务 OneAccess

3.6.2 SEC03 权限管理

3.6.2.1 SEC03-01 定义权限访问要求

明确定义哪些人员或机器应当有权访问哪个组件，选择用于进行身份验证和授权的适当身份类型和方法。

- **风险等级**

  高
- **关键策略**

  使用IAM角色来定义应用程序和组件对资源的访问权限。通过构建最低权限访问模型，确保只授予必要的权限。根据用户的角色和职责分配权限，确保用户只能访问其工作所需的资源。
- **相关云服务和工具**

  统一身份认证服务 IAM

3.6.2.2 SEC03-02 按需分配合适的权限

权限管理应遵循按需分配、最小授权、职责分离原则。需要根据工作职责限定人员对于关键业务系统的访问权限，以免非必要人员或非授权人员访问到关键系统和敏感数据。如需要临时权限，应仅向用户授予有限的时间段内执行特定任务的权限，并且在任务完成后，应撤销访问权限。

- **风险等级**

  高
- **关键策略**
  - 按照IT工作职能划分用户组，将用户加入到与其匹配的用户组中。用户组是IAM用户的集合，IAM可以通过用户组功能实现用户的授权。
  - 优先基于用户组授权，而不是基于用户授权。
  - “admin”为系统缺省提供的管理员用户组，具有所有云服务资源的操作权限。避免将所有用户都加入admin用户组。

- 遵循最小权限原则，仅授予用户组必要的最小权限，如某些用户组只能访问特定的云服务或者某些用户组仅有云服务资源的只读权限。
- 避免IAM自定义策略中包含“*:*”管理权限。
- 如果使用了企业项目，优先在企业项目中对用户组进行授权。如果需要针对账号内所有区域或特定区域内的所有资源进行统一授权，如将账号内所有企业项目的所有资源的访问权限授予统一资源管理组，则可以使用IAM项目进行授权，避免在各个企业项目中逐一授权，简化授权操作。

- **相关云服务和工具**
  - 统一身份认证服务 IAM
  - 企业项目 EPS
  - 云堡垒机CBH：使用CBH限制对运维账号的使用和访问。CBH可用于集中管控运维账号访问系统和资源的权限，对系统和资源的访问权限进行细粒度设置。
  - 组织 Organizations：多账号场景使用Organizations云服务的服务控制策略（SCP）。组织管理账号可以使用SCP指定组织中成员账号的权限边界，限制账号内用户的操作。服务控制策略可以关联到组织、组织单元和成员账号。当服务控制策略关联到组织或组织单元时，该组织或组织单元下所有账号受到该策略影响。

3.6.2.3 SEC03-03 定期审视权限

定期检视和更新权限，以避免权限蔓延，持续清理无用的权限。

- **风险等级**

  高
- **关键策略**
  - 使用IAM用户组控制人员的访问权限，并设置权限的到期时间。
  - 如果用户组的职责产生变化，应该及时调整用户组的权限。
  - 当账号委托给另一个账号时，设置到期时间。
  - 通过IAM用户的“最近一次登录时间”，判断该用户是否为长期未登录的用户，及时管理他们的身份凭证及权限。
- **相关云服务和工具**

  统一身份认证服务 IAM

3.6.2.4 SEC03-04 安全共享资源

大企业的不同组织、部门、团队之间需要安全共享资源。

- **风险等级**

  中
- **关键策略**
  - 大企业往往涉及多个组织单元、多个账号，需要对多账号之间进行共享资源。安全共享资源需遵循以下实践：
  - 使用资源标签。通过标签对资源进行分类和标记，以便于管理和应用策略。
  - 仅与可信实体共享资源。通过使用服务控制策略（SCP）限制权限，您可以限制组织内账户的权限，确保资源仅在组织内部共享。

- 创建专门的服务账号用于共享资源的访问。

- **相关云服务和工具**
  - 组织 Organizations
  - 资源访问管理 RAM：使用RAM为用户提供安全的跨账号共享资源的能力。如果您有多个华为云账号，您可以创建一次资源，并使用RAM服务将该资源共享给其他账号使用，这样您就不需要在每个账号中创建重复的资源。

3.6.3 SEC04 网络安全

3.6.3.1 SEC04-01 对网络划分区域

网络的分区是将网络划分为多个部分，以隔离不同敏感性要求的网络流量和资源，从而增加网络的安全性。

- **风险等级**

  高

- **关键策略**

  通过网络分区，可以实现以下目的：
  - 隔离敏感数据：将敏感数据和应用程序隔离在独立的网络分区中，以减少未经授权访问的风险。
  - 可扩展性：分区和分层可以帮助管理和扩展复杂的网络架构，使其更易于维护和扩展。
  - 限制网络流量：控制不同网络分区之间的通信流量，以确保只有经过授权的流量可以流动。
  - 提高性能和可用性：通过分区网络，可以优化网络性能和可用性，避免网络拥塞和单点故障的影响。

  定义每个分区的边界，并按照方便管理和控制的原则为各网络区域分配地址。例如，对于一个Web工作负载，划分Web区、App区、Data区等。最重要的边界是公共网络（互联网）与应用程序之间的边界，这个边界是您的工作负载的第一道防线。华为云的VPC和子网都可以作为每个网络分区的边界。
  - VPC划分：为VPC指定合适的CIDR范围，以确定VPC的IP地址空间。
  - 子网划分：在VPC中，创建多个子网，并将不同的资源部署在不同的子网中。

- **相关云服务和工具**

  虚拟私有云 VPC

3.6.3.2 SEC04-02 控制网络流量的访问

控制网络流量以确保网络分区之间的流量是可预期的、允许的。依据零信任原则，需在网络级别验证所有的流量出入。确保网络设备的业务能力、网络每个部分的带宽满足业务高峰期的需要。

- **风险等级**

  高

- **关键策略**
  - 在设计网络拓扑时，仔细检查每个组件的连接要求，例如是否需要互联网可访问性（入站和出站）、连接到VPC的能力、边缘服务和外部数据中心等。

除非资源必须接收来自公网的网络流量，否则不要将资源放置在VPC的公有子网中。

- 对于入站和出站流量，应采用深度防御方法。例如对入站流量进行入侵检测、防范恶意的网络攻击。对出站的流量使用NAT网关配置仅出站的单向连接。
- 流量过滤。使用防火墙、ACL控制内部和外部网络之间的访问流量以及内部网络中敏感区域的输入及输出流量，并对所有网络流量进行检查，阻止与已制定安全标准不相符的流量，以避免系统组件受到来自不可信网络的非授权访问。
- 使用应用负载均衡时，七层负载均衡更换为安全的证书。
- 启用VPC流量日志。VPC流日志功能可以记录虚拟私有云中的流量信息，帮助用户优化安全组和防火墙控制规则、监控网络流量、进行网络攻击分析等。关于安全日志更多见：**SEC09-01 实施标准化管理日志**

- **相关云服务和工具**
  - VPC、VPCEP
  - 企业路由器 ER
  - 云连接 CC
  - 云防火墙 CFW：提供云上互联网边界和VPC边界的防护，包括实时入侵检测与防御、全局统一访问控制、全流量分析可视化、日志审计与溯源分析等，同时支持按需弹性扩容、AI提升智能防御能力、灵活扩展满足云上业务的变化和扩张需求，极简应用让用户快速灵活应对威胁。云防火墙服务是为用户业务上云提供网络安全防护的基础服务。
  - EdgeSec：提供基于CDN边缘节点提供的安全防护服务，包括：边缘DDoS 防护、CC 防护、WAF 防护等功能。已使用华为云内容分发网络（CDN）或全站加速（WSA）的用户，购买服务后可为加速域名开启安全防护相关配置，全方位保障业务内容分发。
  - WAF：保护网站等Web应用程序免受常见Web攻击，保障业务持续稳定运行，满足合规和监管要求。
  - AAD：华为云DDoS防护提供全球化服务，以应对DDoS攻击挑战，可提供毫秒级攻击响应、多维度行为分析及机器学习、防御策略自动调优，精确识别各种复杂DDoS攻击，以保护您的业务连续性。用Anti-DDoS流量清洗服务提升带宽利用率。Anti-DDoS为弹性公网IP提供四到七层的DDoS攻击防护和攻击实时告警通知，提升用户带宽利用率，确保用户业务稳定运行。
  - NAT网关：NAT网关位于互联网与云上VPC之间，通过部署NAT网关可掩盖内部网络的IP地址，降低虚拟环境遭受攻击的风险。
  - ELB：对流量进行负载均衡到后端多个节点。

3.6.3.3 SEC02-03 网络访问权限最小化

确保只有必要的人员或组件可以访问特定的网络资源。

- **风险等级**

  高

- **关键策略**
  - 通过配置安全组和网络访问控制列表（ACL），控制进出云资源的网络流量，确保只有授权的流量能够访问特定的服务和端口。根据业务实际情况优化每个网络区域的ACL，并保证访问控制规则数量最小化。

  - 避免暴露多余的公网IP，同时不应对外开放或未最小化开放高危端口、远程管理端口。
  - 安全组仅开放业务所需的网段及端口，禁止设置成对所有IP(0.0.0.0/0)都可访问。
- **相关云服务和工具**
  - 虚拟私有云 VPC
  - NAT网关 NAT
  - 安全云脑 SecMaster：云服务基线核查

3.6.4 SEC05 运行环境安全

3.6.4.1 SEC05-01 云服务安全配置

安全配置是一个信息系统的最小安全保障，云服务安全配置是云环境最基本的安全保证，是开展安全防护的基础。正确配置云服务可以帮助防止安全漏洞和数据泄露，提高整体系统安全性。如果云服务没有达到安全配置基线要求，云上业务及资产将面临巨大安全风险。

- **风险等级**

  高
- **关键策略**
  - 遵循华为云安全配置基线指南，包括对不同服务的安全配置建议，例如：
    - 容器安全，例如容器安全配置，CCE里不安全的容器配置可能导致容器逃逸问题
    - 系统漏洞，例如操作系统的版本有没有升到最新版，使用版本是否存在漏洞
    - 开放必要的端口，例如系统是否对公网开放22，3306等高危端口
    - 禁止将重要业务数据所在的OBS桶设置为公开桶或者配置为公共可读。
  - 定期执行云服务安全配置的基线检查。
    - 全面性检查：确保基线检查覆盖所有关键的云服务配置项，包括身份认证、访问控制、网络安全等关键配置。
    - 定期与实时检查：设置定期自动检查计划，并提供实时检查功能，以便在需要时立即评估云服务的安全状态。
    - 风险评估：对检查结果进行风险评估，识别不同级别的风险资源，如致命、高危、中危、低危和提示。
- **相关云服务和工具**
  - **华为云服务的安全特性**：在云服务模式下，如何保障云上安全，成为大多数企业和客户的首要关注问题。华为云致力于保障其所提供的IaaS、PaaS和SaaS各类各项云服务自身的安全及基础设施安全，同时也为致力于为客户提供先进、稳定、可靠、安全的产品及服务。文档中说明了如何配置华为云服务以满足您的安全性目标。
  - **华为云安全配置基线指南**

- 配置审计 Config
- 安全云脑 SecMaster：使用安全云脑对云服务安全配置基线进行基线检查，持续保护客户的云服务安全。
- 企业主机安全 HSS：最新版本支持包含主机安全和容器安全（原CGS服务）的特性

3.6.4.2 SEC05-02 实施漏洞管理

漏洞管理有助于及时发现并修复系统中存在的安全漏洞，防范潜在的安全威胁和攻击。安全漏洞可能使他人非法获得系统访问特权，应通过可信渠道获取最新的安全情报。

- **风险等级**

  高

- **关键策略**

  安全漏洞可通过及时安装安全补丁的方式修复漏洞，以防恶意个人或软件非法利用从而破坏业务系统和数据。通过及时了解最新的华为云和业界的安全公告，实施对应消减建议，来保证工作负载的安全。

  - 及时了解最新的华为云和行业安全建议。**华为云安全公告**包含有关安全性的最新信息。
  - 漏洞扫描和识别：利用华为云云服务对系统、应用程序进行定期扫描，以发现潜在的漏洞和安全弱点。
  - 自动化扫描漏洞：使用自动化漏洞扫描工具对运行环境进行定期扫描，以发现潜在的漏洞和安全风险。
  - 漏洞修复和补丁管理：制定漏洞修复计划，及时修复已确认的漏洞，并管理安全补丁的发布和应用过程。
  - 在关键节点处检测和清除恶意代码：应在关键网络节点处对恶意代码进行检查和清除，并维护恶意代码防护机制的升级和更新。

- **相关云服务和工具**
  - 企业主机安全 HSS
  - 安全云脑 SecMaster
  - 漏洞管理服务 CodeArts Inspector
  - 威胁检测服务MTD：使用MTD持续发现恶意活动和未经授权的行为。MTD可识别各类云服务日志中的潜在威胁并输出分析结果，从而提升用户告警、事件检测准确性，提升运维运营效率。

3.6.4.3 SEC05-03 减少资源的攻击面

通过加固操作系统、减少未使用的组件和外部服务，以及使用工具加强云安全，减少资源的攻击面。

- **风险等级**

  高

- **关键策略**
  - 强化操作系统和减少组件：通过减少未使用的组件、库和外部服务，可以缩小系统在意外访问下的危险。这包括操作系统程序包、应用程序以及代码中的外部软件模块。

- 创建安全的虚拟机镜像或者容器镜像。
- 使用第三方工具进行安全性分析：使用第三方静态代码分析工具和依赖关系检查工具来识别常见的安全问题和漏洞，确保代码的安全性和合规性。
- 应用其他测试方法：除了工具的使用，还需要在应用程序级别进行测试，如使用模糊测试来查找和修复潜在的漏洞和错误。

- **相关云服务和工具**

  企业主机安全 HSS

3.6.4.4 SEC05-04 密钥安全管理

密钥的安全管理对于整个工作负载的安全性至关重要。如果使用不恰当的密钥管理方式，强密码算法也无法保证系统的安全。密钥的安全管理包括密钥的生成、传输、使用、存储、更新、备份与恢复、销毁等完整的生命周期流程。

- **风险等级**

  高

- **关键策略**
  - 生成密钥：
    - 分层管理密钥。最少把密钥分为两层，即：根密钥和工作密钥，根密钥为工作密钥提供加密保护。
    - 使用安全的随机数生成器来生成密钥，确保密钥的随机性和不可预测性。避免使用弱密钥或者固定密钥。
  - 传输密钥：
    - 使用安全的通信渠道传输密钥，如加密通道或者物理传输。
    - 确保传输过程中密钥不被窃取或篡改。
  - 使用密钥：
    - 最小化密钥的使用范围，避免在不必要的情况下暴露密钥。
    - 实施访问控制和权限管理，限制对密钥的访问。
  - 存储密钥：
    - 使用安全的存储设备或者加密存储来保存密钥。
    - 确保只有授权人员可以访问密钥存储。
  - 更新密钥：
    - 定期更新密钥以应对安全漏洞和攻击。
    - 使用安全的方式进行密钥轮换，确保服务的连续性。
  - 备份与恢复：
    - 定期备份密钥，并将备份存储在安全的地方。
    - 确保有可靠的恢复机制，以防止密钥丢失或损坏。
  - 销毁密钥：

        - 在密钥不再需要时及时销毁密钥。
        - 使用安全的密钥销毁方法，如加密删除或者物理销毁。
- **相关云服务和工具**

  数据加密服务 DEW

3.6.4.5 SEC05-05 证书安全管理

证书的常见用途包括传输数据的加密和系统间的身份认证场景。集中管理每个证书的用途、有效期等信息，并及时对证书替换。

- **风险等级**

  中
- **关键策略**
  - 集中管理证书：
    - 建立中心化的证书管理系统，用于存储、跟踪和管理所有证书。
    - 确保每个证书都有清晰的标识，包括用途、所有者、有效期等信息。
  - 有效期管理：
    - 定期检视证书的有效期，并确保及时对即将到期的证书进行更新或替换。
    - 避免使用过期证书，以防止安全漏洞和服务中断。
  - 安全存储：
    - 将证书存储在安全的位置，只允许授权人员访问。
    - 对私钥进行额外保护，如使用硬件安全模块（HSM）来存储私钥。
  - 加密传输：
    - 在证书的传输过程中使用加密通道，如SSL/TLS，以防止证书被篡改或窃取。
    - 避免在不安全的网络中传输证书，确保传输的安全性。
- **相关云服务和工具**

  云证书管理服务 CCM：CCM提供SSL证书的申请、签发、查询、吊销等一站式管理服务。

3.6.4.6 SEC05-06 使用托管云服务

将计算、数据库、存储等资源使用华为云云服务进行托管，避免自行构建增加的开发和运维成本。

- **风险等级**

  低
- **关键策略**
  - 实施用于托管资源的服务以便在责任共担模式中减少安全维护任务。例如使用华为云的数据库服务而不是自建关系型数据库的实例。

- 使用Serverless架构的云服务，将计算资源的安全交给华为云处理，减免了用户自行运维服务器带来的工作量和人为错误，减少了安全漏洞的风险。这样，用户能够将更多精力集中在业务逻辑和应用的安全性上。

- **相关云服务和工具**
  - 云数据库 RDS for MySQL
  - 云数据库 GaussDB
  - 函数工作流 FunctionGraph
  - 云容器实例 CCI
  - 事件网格 EG

3.7 应用安全

3.7.1 SEC06 应用安全性

3.7.1.1 SEC06-01 安全合规使用开源软件

开源软件在现代软件开发中的重要性不言而喻。越来越多的企业选择使用开源软件来开发和部署软件应用程序。开源软件的使用必须严格遵守合法合规的底线，包括开源软件的来源、漏洞管理、可追溯、归一化及生命周期管理等方面。

- **风险等级**

  高

- **关键策略**
  - 来源可靠。由于开源软件是公开的，因此黑客和攻击者可以更容易地找到其中的漏洞和安全隐患，从而进行攻击和入侵。确保引入的开源软件来源于正规社区官网、供应商官网或厂家官网。
  - 明确软件许可要求。确保引入的开源软件有明确的许可证或签订有相关使用协议。确保按许可要求使用开源，遵守相关的开源许可证和法律法规要求，避免知识产权、License带来的法律风险。应当履行开源义务，避免导致产品或企业的声誉受损。
  - 归一化管理。企业应进行开源软件归一化管理，对开源软件的引入进行归一，建立优选库、路标库，减少开源的种类和数量。牵引团队使用优选的开源软件，保障使用质量和安全。
  - 降低开源漏洞的影响。开源软件的安全漏洞传播快，影响大。一旦出现安全漏洞，快速排查受影响的产品并进行修复是降低影响的关键。
  - 可追溯。对开源软件的变更过程可控、有记录可查，建立产品版本与第三方软件及漏洞的关系。

3.7.1.2 SEC06-02 建立安全编码规范

应用安全涉及需求、设计、实现、部署多个环节，实现的安全是应用安全的重要一环。建立安全编码规范有助于团队编写更安全、更高质量的代码，减少甚至规避由于编码错误引入的安全风险。

- **风险等级**

  高

- **关键策略**
  - 发布团队常用编程语言的安全编码规范。通用的安全编码规范应包含程序输入校验、程序输出编码、身份验证、访问控制、安全加解密算法、异常处理、IO操作、文件上传、序列化、输出格式化等。
  - 对于在Web应用场景使用的语言如Java、Python，还要考虑安全会话管理、防SQL注入、防跨站脚本攻击XSS、防跨站请求伪造CSRF等编码规范。
  - 对于C/C++语言，要考虑缓冲区溢出漏洞、命令注入、危险函数、内存泄露、指针越界、数组读写越界等安全风险。
  - 对于JavaScript语言，要考虑容易受到XSS攻击的安全风险。

3.7.1.3 SEC06-03 实行代码白盒检视

代码白盒检视是一种软件质量保证方法，通过检视源代码的内部结构、逻辑和实现细节，以确保代码符合最佳实践、编程规范和安全标准。在代码白盒检视中，团队成员会检查代码的质量、安全性、可读性等方面，以发现潜在的问题和改进空间。

- **风险等级**

  中

- **关键策略**

  a. **制定检视计划**：

     i. 确定检视的频率和时间安排，以确保代码检视是持续的活动。

     ii. 确定检视范围，例如可以是每次提交、每个功能完成后，或者定期的大规模检视。

  b. **培训团队成员**：

     i. 提供培训以确保团队成员了解如何进行有效的代码检视。

     ii. 确保团队了解代码检视的目的和重要性，以及如何识别常见问题和潜在的安全漏洞，建议将常犯的TOP问题整理成清单，在开发人员编写代码后自检以及他人检视时进行对照。

  c. **选择合适的工具**：

     i. 使用代码检视工具来辅助检视过程，例如静态代码分析工具，以帮助发现潜在的问题。

     ii. 确保团队熟悉并能有效使用这些工具。

  d. **设定清晰的标准和准则**：

     i. 制定明确的代码检视标准和准则，以便检视者能够一致地评估代码质量。

     ii. 着重关注安全性方面。

  e. **分配角色和责任**：

     i. 确定谁将参与代码检视，例如开发人员、架构师、安全专家等。

     ii. 确保每个团队成员了解其在检视过程中的角色和责任。

  f. **记录检视结果**：

     i. 记录检视过程中发现的问题、建议和决定，以便后续跟踪和改进。

     ii. 确保问题得到适当的跟进和解决。

  g. **鼓励合作和讨论**：

i. 鼓励团队成员之间进行合作和讨论，分享经验和观点，以提高检视质量。
ii. 创建开放的氛围，使团队成员能够提出问题和建议，促进共同学习和成长。

h. **持续改进**：
i. 定期评估代码检视过程，收集反馈意见，并进行必要的调整和改进。
ii. 着眼于提高检视效率和质量，以确保团队不断提升代码质量和安全性。

3.7.1.4 SEC06-04 应用安全配置

对应用运行时的各项配置进行加固，以避免因安全配置错误而产生的安全漏洞和风险。

- **风险等级**

  高

- **关键策略**

  根据安全配置规范，对您工作负载中的应用，如Nginx、Tomcat、Apache、Jetty、JBoss、PHP、Redis等完成安全配置加固和Web攻击防护。

  - 系统越权，例如系统是否存在capability提权、suid文件提权、定时任务提权、sudo文件配置提权等系统提权问题。
  - 服务运行用户，例如服务运行的用户是否为最低权限用户，禁止使用root用户运行服务。
  - Web攻击，例如Web应用是否存在SQL注入、XSS跨站脚本、文件包含、目录遍历、敏感文件访问、命令、代码注入、网页木马上传、第三方漏洞攻击等常见Web威胁问题。

- **相关云服务和工具**
  - 企业主机安全 HSS
  - Web应用防火墙 WAF
  - 边缘安全 EdgeSec

3.7.1.5 SEC06-05 执行渗透测试

渗透测试是一种安全评估方法，模拟攻击者的行为，通过模拟真实的攻击场景来评估系统、应用程序或网络的安全性。渗透测试旨在发现系统中的安全漏洞、弱点和潜在的安全风险，以帮助组织改进其安全措施、加固防御，并保护系统免受真实攻击的威胁。

- **风险等级**

  高

- **关键策略**

  a. 建议在开发周期的后期执行渗透测试，使系统功能接近预期发布状态，但也要留有足够的时间来解决发现的问题。
  b. 采用结构化流程：使用结构化流程确定渗透测试的范围，基于威胁建模的模型保持场景相关性，以确保全面评估系统的安全性。
  c. 自动化测试：利用工具自动执行常见或可重复的测试，以加快渗透测试的速度，并提高效率。
  d. 分析测试结果：对渗透测试结果进行深入分析，以确定系统性安全问题，并为进一步的自动化测试和开发者培训提供有用信息。

e. 为构建者提供培训：提供培训，让开发者了解从渗透测试结果中可以期待获得什么，以及如何获取有关修复的信息，以促进问题的及时解决。

3.8 数据安全与隐私保护

3.8.1 SEC07 通用数据安全

3.8.1.1 SEC07-01 识别工作负载内的数据

通过业务流程、数据流动方向、数据分布、数据的所有者等维度，对照合规要求评估数据的敏感度，对数据分级分类。

- **风险等级**

  高

- **关键策略**

  遵循以下步骤梳理、识别数据：

  a. 业务流程分析。

     - 了解业务流程，对照业务流程图，明确在各个环节中产生、处理和存储的数据类型和用途。
     - 与业务部门、开发团队、运维人员等进行交流，获取关于数据的详细信息。

  b. 确定数据的分布：需要确定数据存储在哪里，例如云硬盘、数据库、对象存储等。

  c. 评估数据敏感度。

     - 确定数据的类型和内容，例如是否包含个人身份信息（如姓名、身份证号、地址等）、财务数据（如银行账号、交易记录等）、商业机密（如产品研发计划、客户名单等）或其他受法规保护的数据；
     - 考虑数据的潜在影响。如果数据泄露或被滥用，会对个人、组织或社会造成多大的危害，包括经济损失、声誉损害、法律责任等。
     - 参考相关的法律法规、行业标准和企业内部的合规政策。不同行业和地区对于敏感数据的定义和要求可能不同，例如医疗行业的患者数据、金融行业的客户交易数据等，都有特定的法规和标准来规范其保护。
     - 结合组织的业务战略和风险承受能力。对于关键业务相关的数据，即使其本身不属于常见的敏感类型，也可能因其对业务的重要性而被评估为高敏感度。

  d. 借助数据发现和分类工具，自动扫描工作负载以识别数据。自动识别和分类数据可帮助您实施正确的控制措施。

  e. 创建并维护数据清单。将分级分类后的数据整理成清单，包括数据的名称、描述、来源、分布情况、数据敏感度、所属分类级别等详细信息。

- **相关云服务和工具**

  数据安全中心 DSC：DSC可根据敏感数据发现策略来精准识别数据库中的敏感数据，并支持从海量数据中自动发现并分析敏感数据使用情况，基于数据识别引擎，对结构化数据和非结构化数据进行扫描、分类、分级，解决数据“盲点”。

3.8.1.2 SEC07-02 数据保护控制

针对数据分级分类结果，对每一类数据进行不同级别的数据保护控制，保护数据的机密性和完整性。

- **风险等级**

  高

- **关键策略**
  - 实施适当的数据保护措施，如加密和身份验证。
  - 管理数据访问权限。了解谁可以访问、修改和删除数据，有助于限制数据访问权限，减少数据泄露风险。验证只有获得授权的用户按照“最小化权限”原则访问数据，确保只有经授权的用户才能执行相关操作。
  - 在共享或公开数据之前，对敏感数据进行脱敏处理，防止敏感信息泄露。
  - 数据完整性保护。通过定期备份和版本控制来保护您的数据，防止数据被篡改或删除。将关键数据与其他数据隔离，以保护其机密性和数据完整性。
  - 确保存储了重要业务数据、敏感数据的OBS桶，配置为非公开可读，防止数据被非法访问。
  - 制定风险管理计划：了解数据被意外披露、更改或删除可能会带来的业务影响，有助于制定相应的风险管理计划。
- **相关云服务和工具**
  - 数据库安全服务 DBSS
  - 数据加密服务 DEW

3.8.1.3 SEC07-03 对数据操作实施监控

根据数据的分级分类，应对数据的修改、批量操作等行为实施限制措施或建立监控机制。

- **风险等级**

  高

- **关键策略**
  - 对数据的修改、批量操作等行为实施限制措施或建立监控机制。
  - 使用数据库安全服务DBSS对数据库行为进行审计。数据库安全审计提供旁路模式审计功能，通过实时记录用户访问数据库行为，形成细粒度的审计报告，对风险行为和攻击行为进行实时告警，对数据库的内部违规和不正当操作进行定位追责，保障数据资产安全。
  - 启用数据库安全审计告警。通过设置告警通知，当数据库发生设置的告警事件时，用户可以收到 DBSS 发送的告警通知，及时了解数据库的安全风险。
  - 使用云堡垒机服务CBH识别并拦截数据库高危命令。CBH提供数据库控制策略功能，用户可设置预置命令执行策略，动态识别并拦截高危命令（包括删库、修改关键信息、查看敏感信息等），中断数据库运维会话。同时自动生成数据库授权工单，发送给管理员进行二次审批授权。
- **相关云服务和工具**
  - 数据库安全服务 DBSS
  - 云堡垒机 CBH

3.8.1.4 SEC07-04 静态数据的加密

加密可以防止未经授权的人访问和窃取数据。应该默认对敏感的静态数据进行加密，以确保即使数据遭到未经授权访问或意外泄露，也能保持机密性。

- **风险等级**

  高

- **关键策略**
  - 启用默认加密。对云硬盘 EVS、关系数据库 RDS、对象存储服务 OBS、弹性文件服务 SFS等云服务配置默认加密，以自动加密存储的数据。启用RDS、DWS等数据库的加密，可降低拖库、数据泄露带来的安全风险。
  - 针对敏感数据，采取加密、掩码、匿名化等方式进行保护。这样，即使敏感数据被非法窃取，也可降低这类数据泄露的风险。
  - 应该监控加密和解密密钥的使用，并根据数据用途、类型和分类来选择不同的加密密钥。

- **相关云服务和工具**

  数据加密服务 DEW：DEW与OBS、云硬盘（EVS）、镜像服务（IMS）等服务集成，可以通过密钥管理服务（KMS）管理这些服务的密钥，并对云服务中的数据进行加密，还可以通过KMS API完成本地数据的加密。

3.8.1.5 SEC07-05 传输数据的加密

对传输中的数据进行加密处理，以确保数据在传输过程中不被未经授权的访问者所窃取、篡改或查看。

- **风险等级**

  高

- **关键策略**
  - 使用加密协议：确保在数据传输过程中使用安全的加密协议，以加密数据并保护其在传输过程中不被窃取或篡改。使用最新的TLS版本（如TLS 1.2或更高版本）以确保使用最强的加密标准。
  - 安全传输通道：确保数据传输的通道是安全的，避免使用不安全的网络或公共网络来传输敏感数据。
  - 确保敏感数据在云侧和客户端之间传输时是加密的状态，即使数据被窃取，也难以解密。
  - 端到端加密：采用端到端加密的方式，确保数据在传输的整个过程中都是加密的，从数据生成端到数据接收端都能保持加密状态。对于Web应用的API，必须使用HTTPS来加密客户端和服务器之间的通信。
  - 数据完整性验证：使用哈希函数、数字签名、消息认证码（MAC）等方法来验证数据的完整性，以确保数据在传输过程中没有被篡改。

- **相关云服务和工具**
  - 虚拟专用网络 VPN
  - 云专线 DC
  - 云连接服务 CC
  - 数据快递服务 DES
  - 云证书管理 CCM

3.8.2 SEC08 数据隐私保护

数据隐私保护是指采取一系列措施和技术，旨在确保个人数据和敏感信息在收集、存储、处理和传输过程中得到适当的保护，以防止未经授权的访问、使用或泄露。数据隐私保护旨在维护个人隐私权利，保护个人信息的机密性和完整性。

3.8.2.1 SEC08-01 明确隐私保护策略和原则

明确隐私保护策略和原则是指在处理个人数据时，明确规定和遵守的保护个人隐私数据的总体策略和原则。

- **风险等级**

  高

- **关键策略**

  明确个人数据的分级及影响。个人数据包括：自然人的email地址、电话号码、生物特征（指纹）、位置数据、IP地址、医疗信息、宗教信仰、社保号、婚姻状态等。个人数据按照影响等级分为高、中、低三种个人数据（分级描述如下表）：

个人数据分级	说明
高影响	被不正当地披露可能会违反法律，对公司声誉、财务或营运影响很大，对个人数据主体造成严重的不利影响，例如身份证号、指纹等
中影响	被不正当地披露可能会对公司有严重的不利影响，对个人数据主体造成较大的不利影响，例如账号、通信地址、年龄等
低影响	被不正当地披露对公司造成的影响可控，对个人数据主体造成的影响较小，例如性别等

  影响隐私风险级别的因素包括：个人数据级别（高、中、低）、是否能直接或间接的识别到数据主体、数据数量、数据属性（如疾病史比电话号码更敏感）、存储区域是否有其他数据可关联、个人数据收集/存储/处理/披露的目的（如统计分析、研究、税收管理、法律要求）、所属角色（控制者/处理者/设备供应者）等。

  - 严格保护敏感个人数据。通常情况下，敏感个人数据包括：生活信息如种族或血统、政治观点等；身份信息如身份证号、社会保障号等；财产信息如银行账号信息、存款信息等；健康信息如以往病史、诊治情况等；生物特征信息如指纹、虹膜等；以及其他信息如精准定位信息等。
  - 明确个人数据所涉及的角色，包括数据主体、数据控制者、数据处理者、第三方等角色。
  - 明确隐私保护原则，应遵循合法、透明、安全的原则。
    - 合法、正当、透明：个人数据应当以合法、正当、对数据主体透明的方式被处理。
    - 目的限制：个人数据应当基于具体、明确、合法的目的收集，不应以与此目的不相符的方式作进一步处理。
    - 数据最小化：个人数据应与数据处理目的相关，且是适当、必要的。尽可能对个人数据进行匿名或化名，降低对数据主体的风险。

- 准确性：个人数据应当是准确的，并在必要的情况下及时更新。根据数据处理的目的，采取合理的措施确保及时删除或修正不准确的个人数据。
- 存储期限最小化：存储个人数据不超过实现数据处理目的所必要的期限。
- 完整性与保密性：根据现有技术能力、实施成本、隐私风险程度和概率采取适度的技术或组织措施确保个人数据的适度安全，包括防止个人数据被意外或非法毁损、丢失、篡改、未授权访问和披露。
- 可归责：数据控制者须负责且能够对外展示遵从上述原则。

– 隐私保护是需要贯穿个人数据全生命周期，持续进行。个人数据全生命周期各个阶段皆应有相应的隐私保护要求，根据不同阶段面临的隐私风险制定对应的消减措施。

3.8.2.2 SEC08-02 主动通知数据主体

主动通知数据主体是指数据控制者主动向数据主体（个人）提供信息，告知其数据处理活动的相关信息，例如数据收集的目的、数据处理的方式、数据使用的范围、数据存储的期限等。这种通知通常以隐私政策、用户协议、提示信息等形式呈现。

- **风险等级**

  中

- **关键策略**
  - 主动通知数据主体的重要性在于：
    - 透明度和可控性：通过主动通知，数据主体可以了解数据处理者如何处理其个人数据，从而增加对个人数据的透明度和可控性，使其能够做出知情同意的决定。
    - 合规性：在许多隐私保护法规（如欧洲的GDPR）中，主动通知数据主体是一项法律要求，数据处理者需要向数据主体提供特定的信息，以确保数据处理活动符合相关法规
    - 建立信任：通过主动通知，数据处理者可以向数据主体展示其对隐私保护的重视，从而建立信任关系，增强数据主体对数据处理者的信任感。
    - 保护权利：主动通知数据主体有助于保护数据主体的权利，包括知情权、访问权、更正权等，使其能够行使自己的隐私权利。
  - 数据控制者必须提供隐私声明，隐私声明旨在告诉用户该产品的隐私和数据保护实践，以及在个人信息收集和处理方面用户可做的选择。
  - 隐私声明、个人数据说明应描述产品收集的所有个人数据类型、目的、处理方式、时限等信息。
  - 在产品页面应提供隐私声明，方便用户访问。例如用户可以在网页的页脚找到隐私声明链接。

3.8.2.3 SEC08-03 数据主体的选择和同意

数据主体的选择和同意是指在个人数据被收集、处理或使用之前，数据处理者需要获得数据主体（个人）的明确同意，并且数据主体有权选择是否同意其个人数据被处理的过程。

- **风险等级**

  高

- **关键策略**
  - 收集或使用个人数据前，须明确提示用户，并获得用户的同意，并且允许用户随时关闭对个人数据的收集和使用。
  - 从数据主体系统中传出包含个人数据的错误报告之前，必须提供机制告知数据主体，并获得其同意。
  - 若需要将个人数据用于营销、用户画像、市场调查，数据控制者和设备供应者必须提供机制单独获取数据主体明示同意，并提供随时撤销同意的机制。
  - 设置或读取在数据主体系统上的Cookie前（如用于营销或广告），应提供获取数据主体同意及撤销的机制。
  - 修改用户个人空间的行为（如系统或应用配置变更、下载软件、对用户系统或软件升级），须得到用户的同意。
  - 对未成年人提供服务或收集了包含年龄信息的个人信息时，需要实现从未成年人的监护人处获取同意的功能。
  - 数据控制者应提供对用户的同意和撤销同意行为进行记录的机制。

3.8.2.4 SEC08-04 数据收集合规性

数据收集合规性是指数据控制者在收集个人数据时需遵守相关的法律法规和隐私保护准则，确保数据收集活动符合法律规定并尊重数据主体的权利。

- **风险等级**

  高

- **关键策略**
  - 收集个人数据必须获得数据主体授权。
  - 收集敏感个人数据必须获得数据主体明示同意。
  - 个人数据收集范围、使用目的、处理方式不得超出隐私声明，且遵循最小化原则。
  - 收集到同意之后也需要向数据主体提供撤销或修改同意的途径。

3.8.2.5 SEC08-05 数据使用、留存和处置合规性

数据使用、留存和处置的合规性是指数据处理者在处理个人数据的过程中，包括数据的使用、保留和销毁阶段，需遵守相关的法律法规和隐私保护准则，确保数据处理活动符合法律规定并尊重数据主体的权利。

- **风险等级**

  高

- **关键策略**
  - 使用个人数据前必须获取数据主体授权，使用范围及方法不能超出收集目的。
  - 系统应将隐私保护的功能默认设置成保护状态。
  - 使用个人数据过程中，必须保证个人数据的安全，如记录运营运维阶段对个人数据增删改、批量导出等操作。
  - 用于问题定位的日志中记录个人数据遵循最小化原则。

  - 对于数据控制者，数据主体撤销同意之后，产品必须禁止继续收集和处理其相应个人数据。
  - 对于提供用户画像的系统应为用户提供退出用户画像分析的机制。
- **相关云服务和工具**
  - 数据安全中心DSC：用户可以通过DSC的预置脱敏规则，或自定义脱敏规则来对指定数据库表进行脱敏，DSC支持RDS，ECS自建数据库等云上各类场景。另外，DSC可基于扫描结果自动提供脱敏合规建议，支持一键配置脱敏规则。
  - 数据库安全服务 DBSS：使用数据库安全服务DBSS进行数据脱敏。当需要对输入的SQL语句的敏感信息进行脱敏时，客户可以通过开启DBSS的隐私数据脱敏功能，以及配置隐私数据脱敏规则来对指定数据库表以及来自特定源IP、用户和应用的查询进行脱敏。

3.8.2.6 SEC08-06 向第三方披露个人数据合规性

在将个人数据分享、转移或提供给第三方时，数据控制者必须遵守相关的法律法规和隐私保护准则，以确保数据转移活动符合法律规定并尊重数据主体的权利。

- **风险等级**

  高
- **关键策略**
  - 产品需评估是否存在将个人数据推送给第三方应用。评估是否存在高度敏感的用户数据在未获得用户明示同意便推送。同时应该对齐第三方应用，是否对共享的数据设置了合理的保护机制。
  - 用户个人数据转移给第三方前须经过用户同意，符合合法性原则。
  - 转移的目的和范围不能超出收集时所声明的目的和范围。
  - 必须保证个人数据的准确性、完整性和最新状态，保证在任何阶段和环节不能随意篡改、删除、滥用个人数据。
  - 输出者必须获得接收者的明确承诺，保证个人数据的完整性、准确性和安全性，防止滥用及不正当披露。
  - 高影响个人数据（包括口令，银行帐号，批量个人数据等）的传输须采用安全传输通道或者加密后传输。
  - 如涉及数据的跨境传输，需遵从当地的法律法规。

3.8.2.7 SEC08-07 数据主体有权访问其个人隐私数据

数据主体有权访问其个人隐私数据是指根据相关的隐私保护法律和规定，个人拥有权利要求数据处理者提供关于其个人数据的访问权限。

- **风险等级**

  高
- **关键策略**
  - 向用户提供查询、更新个人数据的功能，且必须是实时、无成本，符合主体参与原则。
  - 数据主体访问个人数据之前必须有认证机制。
  - 记录数据的录入或者更新的时间。
  - 建议提供必要的校验措施，比如通过Web页面的输入框录入e-mail地址的时候，校验e-mail地址格式的合法性等。

- 针对用户的注册信息，必须为用户提供修改其注册信息的途径。
- 用户修改隐私偏好的设置和选项要便于用户发现和使用。
- 对于收集、处理、存储个人数据的系统，应提供数据主体限制其个人数据处理机制。
- 对于收集、处理、存储个人数据的系统，应提供数据主体提供的个人数据导出的机制。

3.9 安全运营

3.9.1 SEC09 安全感知及分析

3.9.1.1 SEC09-01 实施标准化管理日志

对身份防线、网络防线、应用防线、主机防线、数据防线和运维防线等日志实施标准化管理，以监测系统和用户活动，实现日志的统一管理，并确保透明可追溯。

- **风险等级**

  高

- **关键策略**
  - 跟踪并监测对网络资源和关键数据的所有访问。通过系统的活动记录机制和用户活动跟踪功能可有效降低恶意活动对于数据的威胁程度。常见的安全日志如主机安全日志、操作系统日志、堡垒机日志、IAM日志、WAF攻击日志、CFW日志、VPC流日志、DNS日志等。当系统出现错误或安全事件时，通过执行彻底地跟踪、告警和分析，可以较快地确定导致威胁的原因。
  - 确保日志存储时长满足需求。主机和云服务的日志数据上报至云日志服务（LTS）后，在默认存储事件过期后会被自动删除。因此，需要用户根据业务需求配置存储时长。对于需要长期存储的日志数据，应在 LTS中配置日志转储。
  - 对于大型企业，涉及多账号统一安全管理和运营。集中收集来自多云环境、多账号和多云服务产品的日志、告警、配置、策略和资产数据等，提高安全运营和运维效率，实现企业的多账号与资源的统一管理。基于统一管理日志，可支持统一存储、统一分析、统一建模、统一威胁分析、统一编排响应、统一态势报告和统一安全策略管理等。
- **相关云服务和工具**
  - 云日志服务 LTS：使用LTS记录日志数据，快速高效地进行实时决策分析、设备运维管理以及业务趋势分析。
  - Web应用防火墙 WAF
  - 安全云脑 SecMaster

3.9.1.2 SEC09-02 安全事件记录及分析

在发生安全事件之前，可以考虑构建取证能力来支持安全事件调查工作。记录攻击和异常行为并对其分析：应在关键网络节点处（例如内外网的交界处、ELB流量转发处等）检测、防止或限制网络攻击行为；应采取技术措施对采集的安全日志进行持续监控和分析，实现对网络攻击特别是新型网络攻击行为和异常行为的识别和分析。

- **风险等级**

高

- **关键策略**
  - 在发生安全事件之前，可以考虑构建取证能力来支持安全事件调查工作。记录攻击和异常行为并对其分析：应在关键网络节点处检测、防止或限制网络攻击行为；应采取技术措施对采集的安全日志进行持续监控和分析，实现对网络攻击特别是新型网络攻击行为和异常行为的识别和分析。
  - 基于安全事件进行攻击链分析和攻击溯源, 包含攻击的各个路径，初始访问、执行、持久化、权限提升、防御绕过、凭证访问、信息收集、横向移动、数据采集、命令控制、数据窃取和影响破坏等。
  - 可基于流批一体化平台，支持在线、近线和离线的各种异常行为分析模型的构建，包含身份防线，网络防线，应用防线，数据防线，运维防线和主机防线等。也可同时基于AD-HOC实时进行安全事件分析，并聚合成各种报表动态化展示分析。
- **相关云服务和工具**
  - 安全云脑 SecMaster

3.9.1.3 SEC09-03 实施安全审计

对云服务的关键操作开启安全审计，审计覆盖到每个用户。对审计日志进行保护并定期备份，避免受到未预期的删除、修改或覆盖。

- **风险等级**

  高
- **关键策略**
  - 云服务的关键操作包含高危操作（如创建IAM用户、删除IAM用户、重启虚拟机、变更安全配置等）、成本敏感操作（创建、删除高价资源等）、业务敏感操作（网络配置变更等）。
  - 启用关键操作通知功能。启用云审计服务CTS的关键操作通知功能后，CTS会对这些关键操作通过消息通知服务（SMN）实时向相关订阅者发送通知。
  - 开启审计日志转储，将CTS的审计日志存储到OBS。依据您的合规性、业务要求设置日志保留时长。
  - 对审计日志进行保护并定期备份，避免受到未预期的删除、修改或覆盖。可以同步开启审计日志的文件校验，保障审计文件的完整性，防止文件被篡改。
  - 集中管控运维账号访问系统和资源的权限，对系统和资源的访问权限进行细粒度设置。
  - 关于数据的安全审计见：**SEC07-03 对数据操作实施监控**
- **相关云服务和工具**
  - 云审计服务 CTS：用户开通CTS后，系统会自动创建一个追踪器，该追踪器会自动识别并关联当前租户所使用的所有云服务，并将当前租户的所有操作记录在该追踪器中。CTS服务具备对各种云资源操作记录的收集、存储和查询功能，可用于支撑安全分析、合规审计、资源跟踪和问题定位等常见应用场景。
  - 云堡垒机 CBH
  - 数据库审计 DBSS
  - 安全云脑 SecMaster

- 消息通知服务 SMN

3.9.1.4 SEC09-04 安全态势感知

跟踪并监控对网络资源和关键数据的所有访问：通过系统的活动记录机制和用户活动跟踪功能可有效降低恶意活动对于数据的威胁程度。当系统出现错误或安全事件时，通过执行彻底地跟踪、告警和分析，可以较快地确定导致威胁的原因。

- **风险等级**

  中

- **关键策略**
  - 采集各类安全服务的告警事件，并进行大数据关联、检索、排序，全面评估安全运营态势。
  - 生成定期的安全状态报告，总结安全态势，包括发现的问题、采取的行动和改进措施。
  - 确保所有安全措施都符合相关的法规和行业标准，如网络安全等级保护、GDPR、HIPAA、PCI DSS等。
  - 定期对员工进行安全培训，提高他们对云安全的意识和理解。
- **相关云服务和工具**
  - 安全云脑 SecMaster
  - 云监控 CES：使用CES获取安全事件的告警通知。CES提供对监控指标的告警功能，当云服务的状态变化触发告警规则设置的阈值时，系统提供邮件和短信通知，用户可以在第一时间知悉业务运行状况，还可以通过HTTP、HTTPS将告警信息发送至告警服务器，便于用户构建智能化的程序处理告警。

3.9.2 SEC10 安全事件响应

3.9.2.1 SEC10-01 建立安全响应团队

建立安全事件响应团队，明确各角色与职责。

- **风险等级**

  高

- **关键策略**

  安全事件响应团队一般包含如下角色及职责：
  - 安全响应专家：主导网络安全事件调查，负责对事件进行定级、通报、攻击溯源以及确定影响范围，制定应急处置措施，推动服务控制风险。
  - 攻击溯源专家：根据攻击的IOC信息进行溯源，追溯攻击者信息，攻击范围（无遗漏），攻击溯源图（攻击路径）和攻击溯源报告，确认攻击事件性质。
  - 高级分析专家：漏洞分析及复现，恶意样本逆向分析，输出病毒查杀脚本。
  - 服务安全响应专家：协助安全响应人员对事件进行调查分析，配合执行各类日志取证，提供业务架构、网络架构、业务日志等协助分析攻击源头，影响范围。实施应急处置措施，并覆盖安全产品。

3.9.2.2 SEC10-02 制定事件响应计划

事件响应计划（Incident Response Plan, IRP）是组织安全策略的重要组成部分，它旨在确保在安全事件发生时，能够迅速、有序地采取行动，最大限度地减少损失，并尽快恢复正常运营。

- **风险等级**

  高

- **关键策略**
  - 建立事件响应计划，包括定义事件级别、响应流程和恢复策略。对服务可用性有影响或者租户可感知的安全事件划分为5个等级，S1/S2/S3/S4/S5。
  - 实施持续的监控，包括云环境的日志、网络流量和异常行为。当检测到潜在事件时，进行初步分析以确定事件的性质和严重性。
  - 实施快速安全响应动作，隔离受影响的系统或账户、断开网络连接、停止服务、清除恶意文件、修复漏洞、替换受损系统并加固系统，确认所有威胁已经被完全清除，避免再次发生。
  - 制定恢复策略，逐步恢复受影响服务，确保数据和系统一致性，进行测试确保所有系统恢复正常运作。
  - 进行事件后分析，总结事件的起因、响应过程和教训。更新事件响应计划，根据经验教训进行改进。
  - 定期审查和更新事件响应计划，以适应新的威胁和业务需求。

事件级别	事件及时响应时间	平均风险控制时间
S1事件	5分钟	1小时
S2事件	5分钟	2小时
S3事件	5分钟	4小时
S4事件	10分钟	24小时
S5事件	10分账	48小时

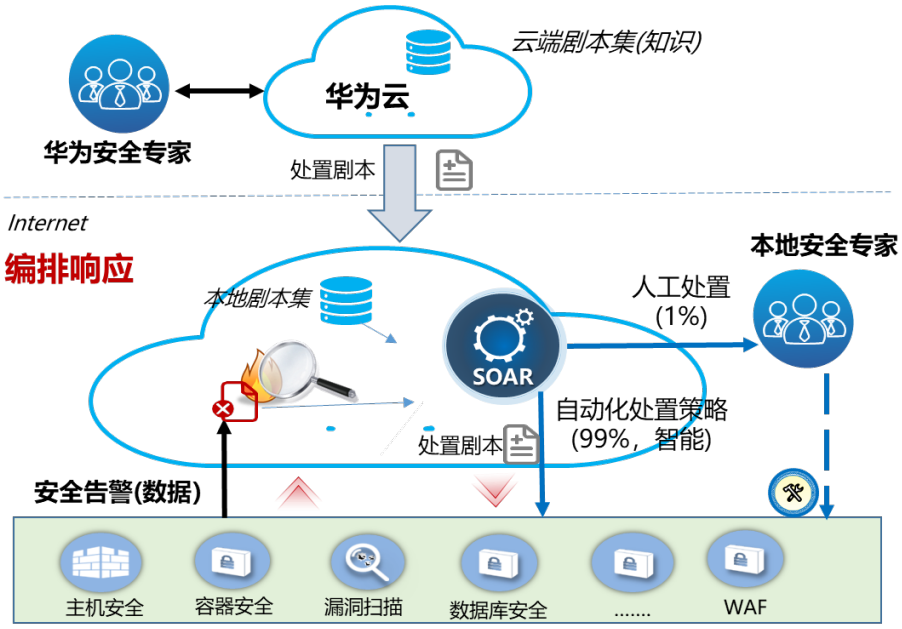
3.9.2.3 SEC10-03 自动化响应安全事件

自动化的响应工作流是安全自动化的核心组成部分，旨在减少安全事件的响应时间，并提高处理效率。

- **风险等级**

  高

- **关键策略**

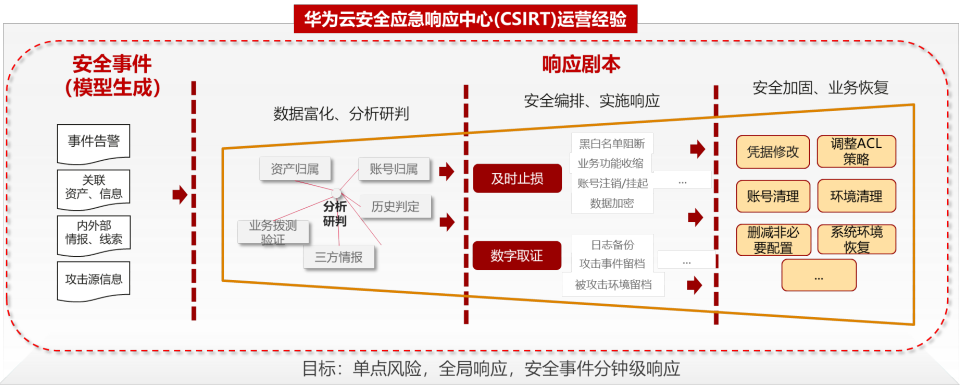


SOAR-Security Orchestration, Automation and Response 安全编排和自动化响应

- 定义响应触发条件：基于威胁情报、异常行为检测和实时监测的结果，确定哪些情况会触发自动化响应。
- 制定响应策略：为每种类型的威胁或事件制定具体的响应动作，例如隔离、修复、通知、调查等。
- 优先级与分级：根据事件的严重性和紧急程度，定义响应的优先级，确保重要事件得到优先处理。
- 持续监控：利用SIEM（安全信息和事件管理）、UEBA（用户和实体行为分析）等工具，对网络、系统、应用程序和用户活动进行实时监控。
- 智能警报：当检测到符合预定义触发条件的事件时，自动生成警报，并根据事件的优先级进行分类。
- 隔离与控制：自动隔离受感染的设备或网络段，防止威胁扩散。
- 自动修复：对于已知的漏洞或问题，自动化执行补丁安装、配置更改或清除恶意软件。
- 取证与记录：自动收集与事件相关的日志、网络包和其他证据，保存为后续分析使用。
- 通知与沟通：向指定的安全团队成员发送警报，同时向IT部门、管理层或其他相关方发送通知。
- 自动化分析：利用机器学习和数据分析工具，自动分析事件的性质、来源和影响范围。
- 人机协作：安全分析师审查自动化分析的结果，必要时进行手动分析，以确认事件的严重性和后续步骤。
- 决策支持：基于分析结果，决定是否需要进一步的人工介入，或是调整自动化响应策略。
- 自动化恢复：对于已解决的事件，自动化执行系统恢复、数据恢复或服务重启。
- 生成报告：自动化生成事件处理报告，包括事件详情、响应行动、处理结果和建议措施。
- 合规性检查：确保整个响应过程符合法律法规和行业标准的要求。

- 事件回顾：定期回顾已处理的事件，评估自动化响应的效果，识别改进点。
- 规则与策略更新：根据回顾结果，更新自动化响应规则和策略，增强系统的自适应能力。
- 培训与演练：定期对安全团队进行自动化响应流程的培训和演练，确保人员熟悉流程并在实际操作中高效执行。

依赖剧本实现威胁处置自动化，让顶级安全专家的经验全面落地，运筹帷幄。事件响应剧本是一套被定义的响应流程，针对明确的安全事件，运用安全编排技术（SOAR）对不同资产、防护组件实时配置动作，以达到单点风险，全局响应的群防群控效果。



- **相关云服务和工具**

  安全云脑 SecMaster：安全编排和自动化响应

3.9.2.4 SEC10-04 安全事件演练

安全事件演练是一种模拟性的活动，旨在让组织成员在一个模拟的安全事件场景下进行实际操作和应对，以测试和提高其应对安全事件的能力。通过安全事件演练，组织可以评估其安全事件响应计划的有效性，发现潜在的问题并进行改进，提高团队的准备性和反应能力。

- **风险等级**

  高

- **关键策略**

按照“三化六防”（实战化、体系化、常态化，动态防御、主动防御、纵深防护、精准防护、整体防护、联防联控）的安全要求，实战化攻防演习将成为常态，以攻促防是实现精确防御、精准防护的有效方法。

事件演练为了高度模拟真实攻击场景，其攻击链与实际的网络战一样，包括信息搜集、边界突破、武器投送和横向扩散等，在实际攻击当中，利用率最高的是弱口令（简单口令、重复口令）爆破、流行漏洞利用和钓鱼。

攻防演习是有规则的，约定开展时间、确定靶标系统、设定战果分数、限制影响面大小等，但实际攻击没有规则，所以常态化的安全建设要求必须高于攻防演习特定要求，才能真正提升整体能力。

- 攻击战法分析：
  - 0day攻击、后门利用、VPN漏洞、邮件钓鱼、社工。随着攻击强度的提升、攻击资源的投入，0day攻击的占比增加、社工手段的多样性增加，大部分攻击都是内网渗透、正面入侵很少。整体攻击战法更贴近于真实的网络入侵，符合“以攻促防”的目标。
- 防守要点变化：
  - 从单点防护开始转变为多点协同防护；从大范围的黑名单拦截转变为有技巧性的联动防护；从边界的纵深拦截延伸到内网的异常监控；从被动的监控防御延伸到主动的诱捕溯源。
- 用户需求变化
  - 产品层面：除传统的入侵防御、WAF和漏扫之外，对资产测绘、APT检测、安全情报和蜜罐的需求在不断增加。
  - 服务层面：除保障期间的安全加固和值守服务外，对日常的安全巡检、安全培训和内部攻防演习需求不断增加。

3.9.2.5 SEC10-05 建立复盘机制

建立安全事件复盘机制可以帮助团队从过去的安全事件中学习经验教训，并改进未来的安全措施。

- **风险等级**

  中

- **关键策略**

  a. 确定复盘的目的：在进行复盘之前，明确目的是非常重要的。确定您希望从这次安全事件中学到什么，以及如何改进未来的安全措施。

  b. 收集事实和数据：收集关于安全事件的所有相关信息和数据，可以用5W2H方法整理该事件，包括事件发生的时间、地点、责任人、事件的过程、原因、影响等。

  c. 组建复盘团队：邀请相关的团队成员和利益相关者参与复盘过程。确保涵盖各个关键领域的代表，如技术人员、安全运营人员等。

  d. 分析根本原因：通过结果追溯分析事件的根本原因，连续问几个为什么，找出导致事件发生的最根本的问题。这有助于避免将来类似事件的发生。

  e. 识别失误和缺陷：识别在安全事件中发生的失误、缺陷或不足之处。这包括技术、流程、人员等方面。

  f. 制定改进措施：基于复盘的结果，制定具体的改进措施和行动计划。这些措施包括人、流程、技术等方面。确保这些措施是可行的、具体的，并且能够有效地解决问题。

  g. 实施改进措施：将制定的改进措施付诸实施，并监控其执行情况。确保所有相关人员都了解并遵守这些改进措施。

  h. 定期检视和更新：定期检视复盘结果和改进措施的执行情况，并根据需要进行更新和调整。持续改进是一个持久的过程。

  i. 文档和分享：将复盘的结果和改进措施进行文档化，并与团队内部分享。这有助于确保所有人都能从中学习，并避免类似的错误再次发生。

j. 培训和意识提升：通过培训和意识提升活动，确保团队成员了解安全事件复盘的重要性，并能够积极参与其中。

3.10 参考架构

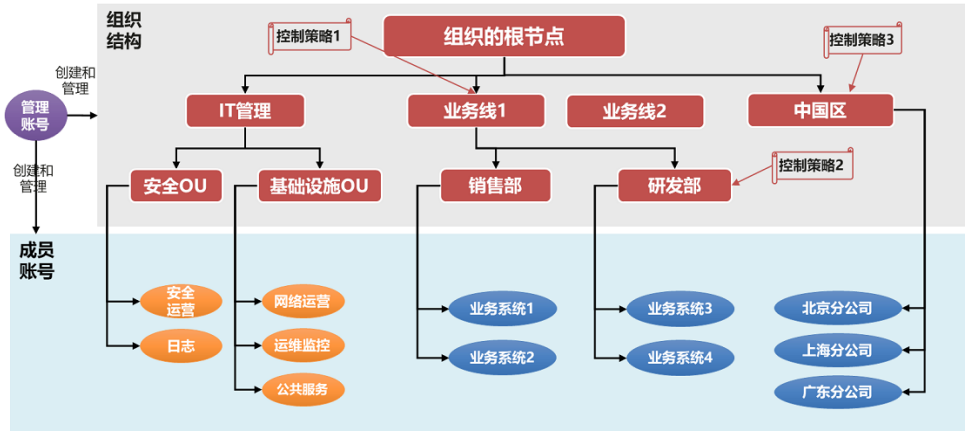
为了构建安全、可信、合规的云上工作负载，华为云提供了大量的与安全相关的云服务。华为云客户基于Well-Architected架构的最佳实践会组合使用到这些云服务。我们的解决方案架构师在与客户进行沟通时，客户通常会提出以下疑问：

- 是否有一个全局性的视图可以表达构建安全工作负载的整体情况?
- 在多账号环境以及单账号环境中应该使用哪些云服务?
- 如何从全局到局部、自顶向下及从不同视角考虑工作负载的安全?

基于以上诉求，我们构建了安全参考架构。安全参考架构旨在帮助客户有效地使用华为云服务构建出安全工作负载的指南。它提供了不同层次视角下的架构图、安全云服务分组的方法和云服务集成参考。需要说明的是，安全参考架构是一个参考范式，而不是绝对的标准。客户需要根据其具体情况和需求进行定制和实施。

3.10.1 组织级参考架构

华为云提供了Landing Zone解决方案帮助企业客户在云上构建架构卓越、安全合规、易扩展的多账号运行环境，首要环节是规划组织和账号架构。按照康威定律，企业在华为云上的组织和账号架构要与企业的组织和业务架构总体保持一致，但也不要完全照搬复制。华为云提供以下参考架构，建议按照业务架构、地理架构、IT职能等维度设计组织层级和账号。



- 按照业务架构在华为云上划分不同的组织层级和OU，每个业务OU下面可以按照业务系统创建独立的成员账号。规模较大的业务系统或安全隔离要求严格（如需要遵守PCI-DSS、HIPPA等合规标准）的业务系统对应一个独立的成员账号，安全隔离要求不高的多个小型业务系统可以共享一个成员账号。以销售部为例，可以为销售管理系统、数字化营销系统等较大的业务系统创建独立的成员账号；以研发部为例，可以将围绕单个产品的设计、研发等系统部署在一个成员账号中。
- 按照地理架构在华为云上划分不同的组织层级和OU，每个地理区域OU下面可以按照国家或地区创建独立的成员账号，在上面可部署本地的客户关系管理系统、客户服务系统等。上述参考架构把中国区等区域组织映射为OU，为其下属的北京、上海等分公司创建独立的成员账号以承载本地化的应用系统。
- 针对企业的IT部门，在华为云上创建对应的组织单元，并按照IT职能创建对应的成员账号，一方面实现IT管理领域的职责和权限隔离，另一方面对企业内多个成员

账号进行统一的IT管理。上述参考架构中创建了两个OU，安全OU下面创建用于安全运营和日志审计的账号，基础设施OU下面创建用于网络运营、运维监控、公共服务和沙箱测试的账号。下表是这些IT职能账号的详细说明。

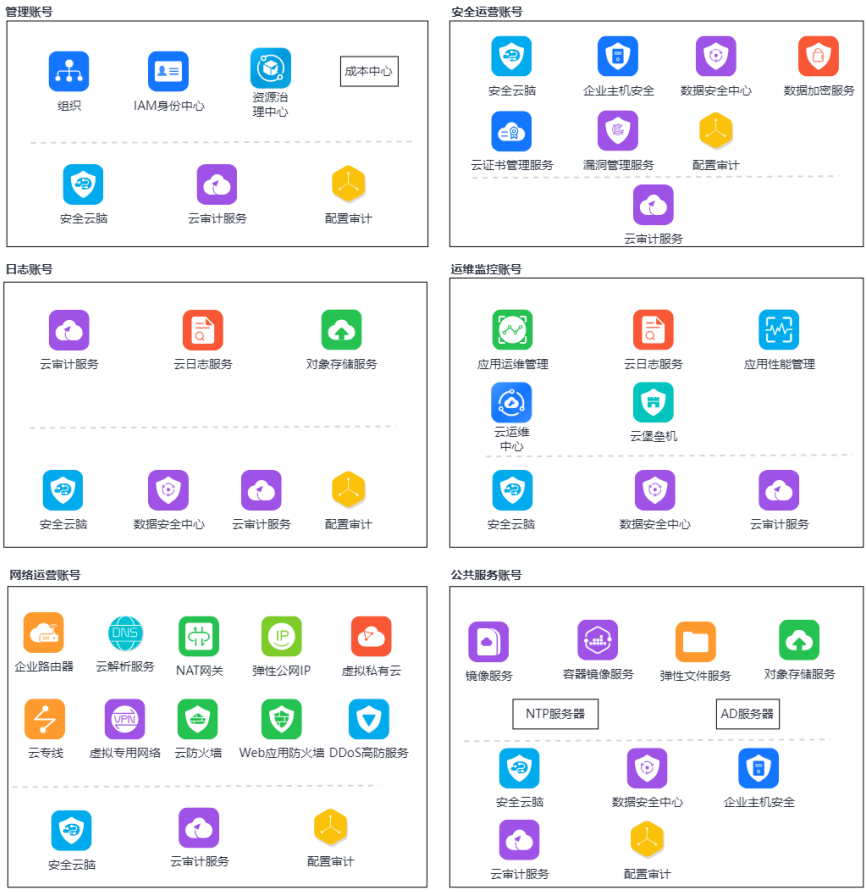
- 除了上述账号之外，每个组织有且仅有一个管理账号，管理账号不建议部署任何云资源，主要是做好以下管理工作：
  - 统一组织和账号管理：创建和管理组织结构和组织单元，在组织单元下面创建成本账号，或者邀请已有账号作为组织单元的成员账号。
  - 统一财务管理：针对整个企业在华为云上的所有账号进行统一财务管理，包括统一预算管理、统一账单管理、统一成本结算、统一成本分析等。
  - 统一控制策略管理：为各个组织单元和成员账号设置服务控制策略，强制限定成员账号下IAM用户（包括成员账号的管理员用户）的权限上限，避免用户权限过大带来安全风险，创建服务控制策略时可以将其应用到某一个组织单元，该服务控制策略可以继承到关联的成员账号和下层组织单元。
  - 统一身份权限管理：针对整个企业在华为云上的所有账号进行集中的用户身份管理、权限设置，统一设置跟外部IdP的身份联邦。

以下是上述账号的详细说明，其中安全运营账号和日志账号针对企业所有账号行使集中的安全管理职责，需要在其他账号下按需开通相应的安全云服务，使得安全运营账号和日志账号能够统一汇聚其他账号的安全态势、安全审计日志等数据，安全运营账号的安全策略和事件响应指令也可以统一下发给其他账号。为了保护该账号下开通的云服务，我们建议在该账号下开通相关的安全云服务，见下表最右边的列：

账号名称	履行的职能	责任团队	建议开通的云服务	建议开通的安全云服务
管理账号	针对整个企业进行统一组织和账号管理、统一财务管理、统一控制策略管理和统一身份权限管理	IT治理团队	组织Organizations、资源治理中心RGC、成本中心、IAM身份中心	安全云脑SecMaster、云审计服务CTS、配置审计Config
安全运营账号	作为企业安全运营中心，统一管控整个企业内所有账号的安全策略、安全规则和安全资源，为成员账号设置安全配置基线，对整个企业的信息安全负责	安全管理团队	统一部署具备跨账号安全管控的服务，如安全云脑SecMaster、企业主机安全HSS、数据安全中心DSC、数据加密服务DEW、云证书服务CCM、漏洞管理服务CodeArts Inspector、配置审计Config等	云审计服务CTS

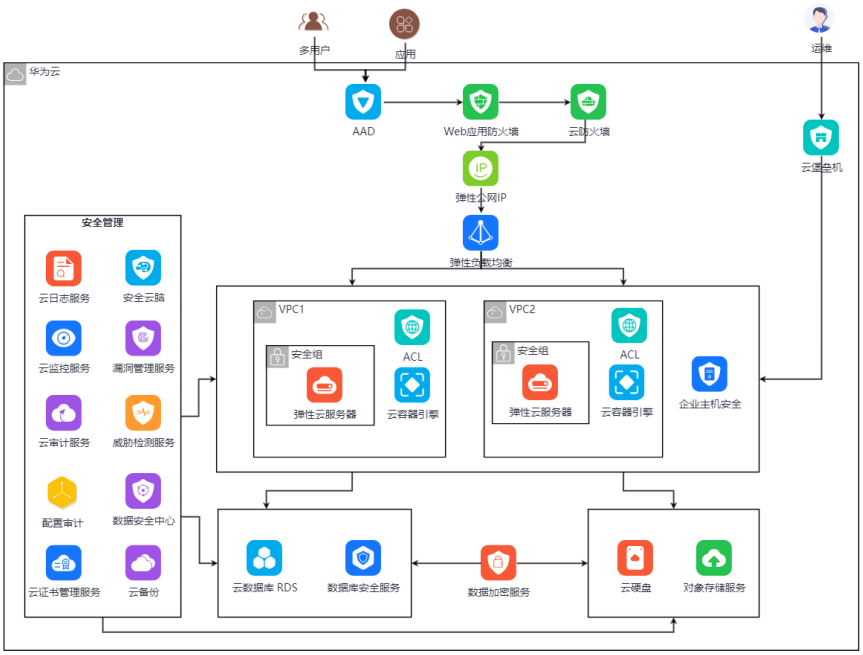
账号名称	履行的职能	责任团队	建议开通的云服务	建议开通的安全云服务
日志账号	集中存储和查看所有账号的审计日志和安全相关的日志（如VPC流日志和OBS访问日志等）	合规审计团队	云审计服务CTS、云日志服务LTS、对象存储服务OBS等	安全云脑SecMaster、数据安全中心DSC、云审计服务CTS、配置审计Config
运维监控账号	统一监控和运维各个成员账号下的资源和应用，统一进行告警管理、事件处理和变更管理，并提供运维安全保障措施	运维团队	应用运维管理AOM、COC、云日志服务LTS、应用性能管理APM、云堡垒机CBH等	安全云脑SecMaster、云审计服务CTS、配置审计Config
网络运营账号	集中部署和管理企业的网络资源，包括网络边界安全防护资源，实现多账号环境下的统一网络资源管理和多账号下VPC网络的互通，尤其需要集中管理面向互联网的出入口和面向线下IDC机房的网络出入口	网络管理团队	ER、DNS、NATG、EIP、VPC、DC、CC、VPN、CFW、WAF、AAD等	安全云脑SecMaster、云审计服务CTS、配置审计Config
公共服务账号	集中部署和管理企业的公共资源、服务和应用系统，并共享给其他所有成员账号使用	公共服务管理团队	镜像服务IMS、容器镜像服务SWR、弹性文件服务SFS、对象存储服务OBS、自建NTP服务器、自建AD服务器等公共资源	安全云脑SecMaster、云审计服务CTS、配置审计Config、企业主机安全HSS、数据安全中心DSC
业务账号	根据业务架构和地理架构创建，用于部署支撑研发、生产、供应、销售、服务各个业务领域的应用系统	应用DevOps团队	按需开通业务系统所需的云服务	安全云脑SecMaster、云审计服务CTS、配置审计Config、企业主机安全HSS、数据安全中心DSC

组织级的参考架构图如下：



3.10.2 工作负载级参考架构

对于一些中小企业，使用华为云单账号即能满足其IT系统管理的诉求。客户会把所有工作负载部署在一个账号内。以下是一个单账号的工作负载级的安全参考架构。



该架构主要的安全设计如下：

- **网络安全**
  - 防DDoS攻击使用AAD服务
  - Web类攻击采用WAF防护
  - 采用SSL证书进行通信加密
  - 互联网边界、VPC之间采用云防火墙
- **运行环境安全**
  - 企业主机安全服务保护主机安全和容器安全
  - VPC内访问控制使用网络ACL+安全组
  - 使用漏洞扫描服务定时扫描云上各资源漏洞
- **数据安全**
  - 数据安全中心实现数据全生命周期安全
  - 存储默认启数据加密
  - 关键数据库部署数据库安全服务
  - 使用云备份归档服务防关键数据丢失
- **安全运营**
  - 使用安全云脑鸟瞰整个云上安全
  - 使用云日志、云审计、配置审计、云监控等服务管理云上资源
  - 使用威胁检测服务检测各类云服务日志中的恶意活动和未经授权行为
  - 使用云堡垒机接入运维

3.11 安全性云服务介绍

- **安全治理**
  - **统一身份认证服务 IAM**：提供权限管理、访问控制和身份认证的基础服务，安全地控制华为云服务和资源的访问权限。
  - **组织 Organizations**：为企业用户提供多账号关系的管理能力。用户可以将多个华为云账号整合到创建的组织中，并可以在组织中设置治理策略。
  - **应用身份管理服务 OneAccess**：为云提供的应用身份管理服务，具备集中式的身份管理、认证和授权能力，保证企业用户根据权限访问受信任的云端和本地应用系统，并对异常访问行为进行有效防范。
  - **资源治理中心 RGC**：提供搭建安全、可扩展的多账号环境并持续治理的能力。
  - **资源访问管理 RAM**：为用户提供安全的跨帐号共享资源的能力。您可以创建一次资源，并使用RAM服务将该资源共享给指定对象（包括组织、组织单元以及帐号）
  - **IAM 身份中心**：为客户提供基于华为云组织（Organizations）的多帐号统一身份管理与访问控制。可以统一管理企业中使用华为云的用户，一次性配置企业的身份管理系统与华为云的单点登录，以及所有用户对组织下帐号的访问权限。
- **网络安全**
  - **云防火墙 CFW**：新一代的云原生防火墙服务，弹性灵活降低部署成本，智能极简助力高效运维。
  - **DDoS防护 AAD**：华为云DDoS防护提供全球化服务，以应对DDoS攻击挑战，可提供毫秒级攻击响应、多维度行为分析及机器学习、防御策略自动调优，精确识别各种复杂DDoS攻击，以保护您的业务连续性。
  - 边缘安全 EdgeSec：华为云基于CDN边缘节点提供的安全防护服务，包括：边缘DDoS 防护、CC 防护、WAF 防护等功能。已使用华为云内容分发网络（CDN）或全站加速（WSA）的用户，购买服务后可为加速域名开启安全防护相关配置，全方位保障业务内容分发。
- **威胁检测**
  - **Web应用防火墙 WAF**：保护网站等Web应用程序免受常见Web攻击，保障业务持续稳定运行，满足合规和监管要求。
  - **配置审计 Config**：为用户提供全局资源配置的检索，配置历史追溯，以及基于资源配置的持续的审计评估能力，确保云上资源配置变更符合客户预期。
  - **企业主机安全 HSS**：帮助客户方便地管理主机、容器的安全风险，实时发现勒索、挖矿、渗透、逃逸等入侵行为，是等保合规、护网、重保必备服务。
  - **漏洞管理服务 CodeArts Inspector**：面向软件研发和服务运维提供的一站式漏洞管理能力，通过实时持续评估系统和应用等资产，内置风险量化管理和在线风险分析处置能力，帮助组织快速感知和响应漏洞，并及时有效地完成漏洞修复工作，更好地应对潜在的安全威胁。
- **数据安全**
  - **数据安全中心 DSC**：新一代的云原生数据安全平台，提供数据分类分级，敏感数据扫描，数据安全体检，数据水印溯源，数据脱敏等基础数据安全能力。通过数据安全资产地图整合数据安全生命周期各阶段状态，对外整体呈现云上数据安全态势。

  - **数据加密服务 DEW**：提供密钥管理、凭据管理、密钥对管理、专属加密功能，安全可靠为用户解决数据安全、密钥安全、密钥管理复杂等问题。
  - **云证书管理服务 CCM**：为云上海量证书颁发和全生命周期管理的服务。目前它可以提供SSL证书管理和私有证书管理服务。
  - **数据库安全服务 DBSS**：基于机器学习机制和大数据分析技术，提供数据库审计，SQL注入攻击检测，风险操作识别等功能，保障云上数据库的安全。
- **合规与隐私保护**
  - **合规中心**：为您提供全方位的合规遵从性指导和资源
  - **云审计服务 CTS**：提供对各种云资源操作记录的收集、存储和查询功能，可用于支撑安全分析、合规审计、资源跟踪和问题定位等常见应用场景
  - **配置审计 Config**：为用户提供全局资源配置的检索，配置历史追溯，以及基于资源配置的持续的审计评估能力，确保云上资源配置变更符合客户预期。
- **安全运营**
  - **安全云脑 SecMaster**：华为云原生的新一代安全运营中心，集成华为云多年安全经验，基于云原生安全，提供云上资产管理、安全态势管理、安全信息和事件管理、安全编排与自动响应等能力，可以鸟瞰整个云上安全，精简云安全配置、云防护策略的设置与维护，提前预防风险，同时，可以让威胁检测和响应更智能、更快速，帮助您实现一体化、自动化安全运营管理，满足您的安全需求。
  - **威胁检测服务 MTD**：威胁检测服务持续发现恶意活动和未经授权的行为，从而保护账户和工作负载。该服务通过集成AI智能引擎、威胁黑白名单、规则基线等检测模型，识别各类云服务日志中的潜在威胁并输出分析结果，从而提升用户告警、事件检测准确性，提升运维运营效率，同时满足等保合规。

3.12 更多参考文档

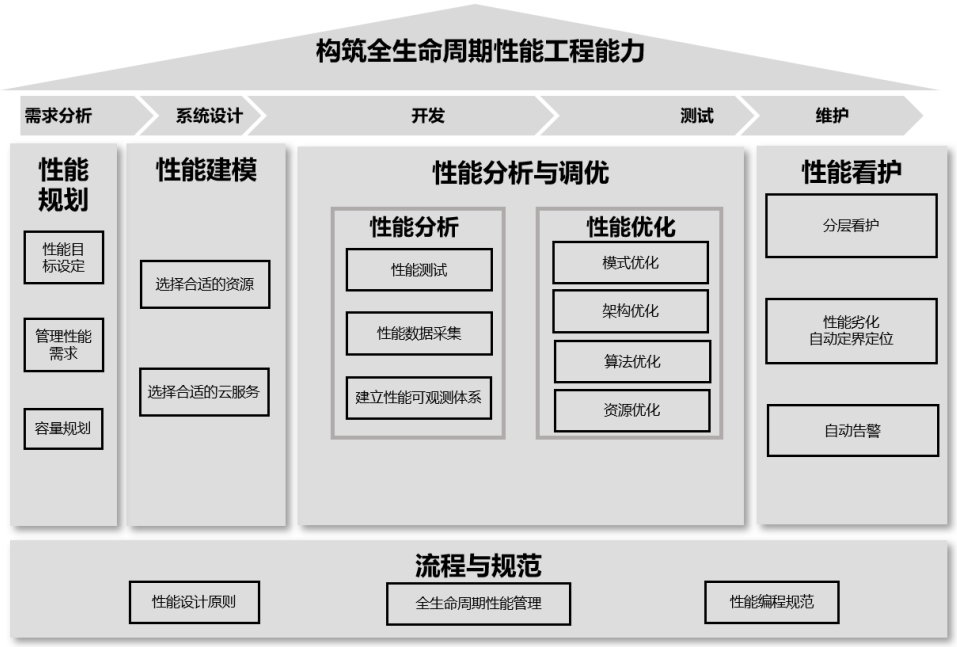
- **华为云零信任能力成熟度模型白皮书**
- **企业上云安全白皮书**
- **华为云安全白皮书**
- **华为云隐私保护白皮书**
- **华为云服务的安全特性**
- **华为云安全配置基线指南**

4 性能效率支柱

4.1 性能效率支柱简介

如何设计出高性能的架构是一个普遍性的问题。作为基本的质量属性，性能的重要性和性能失败后果的严重性是毋庸置疑的，实际上公司内外都有很多性能失败的例子。本文试图为性能设计、性能优化提供一些技术方法和手段，这些方法手段可以用于系统的软件性能工程建设，也可用于指导性能调整和优化。

早期的设计决策会对性能调节能否成功，以及是否有必要进行性能调节产生重要影响。如果开发的软件对性能非常敏感，实际上需要从设计阶段和开发周期的第一天起就考虑性能管理的问题，即采取系统的主动性能管理的办法来解决性能问题。除了管理上的措施外，解决性能问题需要在系统和架构设计、实现方案设计及编码实现上采取有效的技术手段来保证。



性能工程技术全景图

一般认为，性能问题通常由于体系架构或设计问题造成，而不是低效的编码引起的。性能问题在开发过程的早期已经引入，而大部分开发团队直到集成测试，或更晚的时

候才予以考虑。实际情况并非完全如此，编码实现阶段引入的性能问题也很普遍，只是解决体系架构引起的问题代价要高得多。下面给出影响系统性能的几个因素：

- 体系架构设计：影响性能的决定性因素，需要在设计之初考虑
- 实现方案设计：影响性能的主要因素，在不改变整体架构的情况下可以修改
- 编码实现：目前情况下是重要因素，也是可以不断改进的因素

系统或组件的性能问题，对外的表象上反应为：

- 请求响应延迟时间过长
- 资源占有量过大

对常见的性能问题进行分析，可以发现对于一个系统或组件来说，性能问题经常发生在以下方面：

- 实体间通信或者调用处理（包括数据库）
- 频繁调用函数、模块处理过程、数据组织等问题
- 并行处理资源争用引起的延迟
- 串行处理进程/线程间等待延迟

4.2 基础概念

指标	概念解读
性能	性能是指软件系统或软件对应其及时性要求的符合程度。及时性用响应时间或吞吐量来衡量。
响应性	响应性是系统实现其响应时间或吞吐量目标的能力。
响应时间（RT）	用户感受系统为其服务所耗费的时间。不同业务系统的响应时间期望值不同，如互联网业务多为500ms以下、金融业务1s以下等。
可伸缩性	可伸缩性是系统自对齐软件功能的要求增加的情况下，继续实现其响应时间或吞吐量目标的能力。
吞吐量（TPS）	吞吐量反映处理能力，指系统在每单位时间内能处理多少个事务/客户请求/单位数据等。
网络带宽	带宽是指在一定时间内，传输数据的能力或速率。
网络流量	网络流量是指在网络中传输的数据量，它可以是指定时间内通过网络传输的数据总量，也可以是指网络中某个特定节点或连接上的数据传输速率。
网络延迟	网络延迟指的是从发送数据到接收数据所需的时间间隔。

4.3 设计原则

以下是常用的性能优化指导原则：

- 中心化原则：识别支配性工作量负载功能，并使其处理过程最小化，把注意力集中在对性能影响最大的部分进行提升。
- 本地化原则：选择靠近的活动、功能和结果的资源；避免通过间接的方式去达到目的，导致通信量或者处理量大辐增加，性能大辐下降。
- 共享资源： 采取共享资源的设计，通过协作减少争用延时从而改善整体性能；如多个进程可以从一个数据库的同一部分读取。
- 并行处理：当并行处理过程的增速能抵消通信开销和资源争用延迟时，执行并行处理。
- 分散负载原则：通过在不同时间或者不同位置处理冲突负载，从而分散负载：将资源划分为成一些相对独立的小资源组，不同进程/线程可以独立访问，是“资源”分散的常见方案；将同一时间点的多个请求分散到一个时间区段，是“时间”分散的方案。

4.4 问题和检查项

问题	检查项/最佳实践
PERF01 如何确立流程与规范?	1. 全生命周期性能管理 2. 应用性能编程规范
PERF02 如何进行性能规划?	1. 定义性能目标 2. 容量规划
PERF03 如何进行性能建模?	1. 选择合适类型的计算云服务 2. 选择合适规格的虚拟机和容器节点 3. 使用弹性伸缩 4. 选择合适类型的网络云服务 5. 选择合适类型的存储云服务 6. 选择合适的消息队列 7. 选择合适规格的Kafka 8. 选择合适规格的RocketMQ 9. 选择合适规格的RabbitMQ 10.选择合适的关系型数据库 11.选择合适的非关系型数据库
PERF04 如何进行性能分析?	1. 定义性能验收标准 2. 选择合适的测试方式 3. 性能测试步骤 4. 资源性能数据采集 5. 应用性能数据采集 6. 建立性能可观测性体系

PERF05 如何进行性能优化?	1. 设计优化 2. 通用算法优化 3. WEB场景资源优化 4. 大数据场景资源优化
PERF06 如何进行性能看护?	1. 分层看护 2. 性能劣化自动定界定位 3. 自动告警

4.5 PERF01 流程与规范

4.5.1 全生命周期性能管理

全生命周期性能管理围绕需求、设计、开发、测试与编护完整的软件生周期展开，将性能活动内化到生命周期流程中，实现性能工作的常态化。

4.5.1.1 PERF01-01 全生命周期性能管理

- **风险等级**

  高

- **关键策略**

  指定性能目标

  从性能角度来看，最好为性能场景定义具体的、量化的、可测量的性能目标。若要设置这些目标，需要充分了解业务要求以及预期将提供的服务质量。 需要与业务利益干系人共同关键功能的体验要求，而不是只关注技术指标。通过明确地说明性能需求来控制性能，说明要足够明确，以便可以定量地确定软件系统是否满足该目标。具体要求：

  - 定义明确的性能需求目标
  - 避免使用定性的、模糊的性能目标
  - 为每个性能场景定义一个或多个目标
  - 性能指标项的粒度要合适

  功能够用

  在业务初期的设计阶段，要考虑简化组件/模块/方法/类的功能设计，避免设计面面俱到的多功能组件/模块/方法/类；调用功能时，避免功能过剩、并对性能影响较大的调用；选择云服务的时候，选择合适的云服务，结合业务的特征选择合适的云服务类型和规格，利用好云弹性的特性的优势。设计功能过于复杂的组件，有时候是为了通用，有时候则是一种不好的软件设计习惯。够用原则适用于自己设计或者调用已有的功能，使用时注意避免过度设计。

  性能可观测

  在业务系统开发维护阶段，采取措施（例如在关键点插入代码，探测器）使测试和分析负载场景、资源需求、性能目标达成一致。使用监控工具来分析历史趋势，并识别支配性占比的数据流和代码实现路径。本原则强调采取措施使性能指标可测试，可以利用商用工具测试质量指标，也可以在设计时考虑相关性能指标的可测试性措施。需要测试的数据包括响应时间，处理容量，也包括功能/模块被执行频度等指标。

通过优化提高效率

在初始阶段设置的目标考虑到各种约束和业务目标，随着业务的增长应不断进行调整。为了进一步优化性能效率，需要清楚地了解系统的使用方式、演变过程，以及平台或技术是如何随时间变化的。需要预留足够的时间来进行持续的性能优化，可以构建性能驱动的优化文化，让团队成员主动监视性能数据；通过指标数据驱动改进，使用新的设计模式和新的技术来优化体系结构。

性能优化成熟度模型

4.5.2 应用性能编程规范

4.5.2.1 PERF01-02 应用性能编程规范

- **风险等级**

  高

- **关键策略**

  性能效率是一个系统性的工程，需要综合考虑从架构、设计、编码，到编译、运行的全过程，特别是在编码实现层面，有很多编码技巧，在不影响可读性、可维护性的前提下，提升软件性能。结合编程语言，将高性能编码最佳实践内建的规范中，将会充份发挥性能优势，提升软件的执行效率，最终提升产品的竞争力。

  高性能编码规范构建策略：

  - JAVA语言：结合语言基础能力的使用、并发模型、部署调优、工具链辅助等维度展开。

- C/C++语言：结合语言基础能力、编译技术、并发技术、高效数据结构与算法、高性能库及工具链辅助展开。

4.6 PERF02 性能规划

4.6.1 性能规划

4.6.1.1 PERF02-01 定义性能目标

- **风险等级**

  中

- **关键策略**

  建立性能目标是实现工作负载性能效率的重要步骤。性能目标定义了工作负载所需的性能级别，并帮助衡量实现这些目标的有效性。性能目标提供了衡量和比较工作负载效率的基准。此基准可帮助你突出显示改进领域。这些目标还使任务与组织的目标保持一致，并增强业务成果。此外，性能目标还提供资源分配方面的指导，帮助确保工作负载能够适应不同的需求，同时保持最佳性能。

  - 尽早设计性能目标

  性能目标是定义性能的指标，清晰明确的性能目标是关键，通过性能目标，团队可以针对特定目标持续改进。为了确保系统能够满足预期的可靠性和性能要求，避免系统性能瓶颈，性能目标设计需要在部署业务之前开展，重点的是明确系统的需求和预期目标，以生成性能目标范围。

  - 结合业务明确性能要求

    通过性能目标可以确定系统能够承载的最大用户量、并发请求量等，要保持性能目标与业务目标的一致性，需要在设计性能目标时考虑到业务目标的需求。

    - 明确业务相关的性能指标：性能指标应该与业务目标有关，例如响应时间、吞吐量、并发用户数等，这些指标需要能够反映出业务的需求。
    - 确定业务优先级：不同的业务需求有不同的优先级，因此需要根据业务的重要性和紧急程度确定业务的优先级，以便在性能测试和优化时重点关注。
    - 定期回顾和更新性能目标：业务需求会随着时间的推移而发生变化，因此需要定期回顾和更新性能目标，以确保其与业务目标保持一致。

  - 确定关键性能指标

  关键性能指标有助于衡量与业务目标相关工作负载的运行状况和性能，通过监测系统的性能指标，确定系统的性能瓶颈，如响应时间、吞吐量、资源使用率等。

  - 设置特定指标

  关键指标只是一个参考，在确定关键指标后，需要根据实际情况设定具体的性能目标或阈值。设定这些目标和阈值可以帮助我们更好地监控和管理性能，并采取优化措施。这不仅可以提高系统的性能，还可以提高用户满意度。

  比如购物网站，我们可以设定页面加载时间不能超过5秒，如果页面加载时间超过设定的阈值，我们就可以采取优化措施，例如优化图片大小、减少HTTP请求等，以提高页面加载速度。

  - 记录并公开性能目标

满足性能目标是一个持续的过程，需要开发和运营团队的共同努力。开发团队需要在开发阶段考虑性能因素并进行优化；运营团队则需要在运营阶段持续监控和优化性能。记录并公开性能目标，以便开发和运营团队便捷查看性能目标，可以帮助开发和运营团队更好地理解和达成共同的目标。

- 评估客户反馈

客户反馈对于任何组织都至关重要，我们非常重视客户声音及客户满意度，并将其视为我们持续改进的指南。我们会定期收集和分析客户的反馈，将客户反馈纳入技术基准和持续优化流程，了解客户的需求和期望的变化，以便我们可以根据客户的需求和期望进行优化，并相应地调整性能目标。

4.6.1.2 PERF02-02 容量规划

- **风险等级**

  中

- **关键策略**

  容量规划指根据业务需求和系统性能，包括用户数量、并发请求量、响应时间要求等，以此规划和配置系统所需的资源。容量规划对于任何组织来说都非常重要，有效的容量规划可以确保有足够的资源来满足预期的需求，同时避免浪费资源。

  - 收集容量数据

    收集容量数据有助于将业务目标转化为技术要求，并且对于预测容量至关重要。为了满足工作负载需求，收集容量数据需要包括系统资源消耗数据以及业务关键数据。

    - 资源消耗数据：包括CPU、内存、磁盘空间、网络带宽等，以便确定系统的瓶颈所在。
    - 业务关键数据：包括用户数量、用户行为模式、业务类型、业务时段等，以便确定业务需求对工作负载的影响。

  - 预测需求

  有效的容量规划需要为未来的业务需求做好准备，通常使用工作负载的数据来预测未来需求。预测需求是一个复杂的过程，涉及到多种因素，包括市场趋势、消费者行为、竞争环境等。通过多种方法的组合，如历史数据分析、资源分析、趋势分析等，以此作为预测需求的基础，并结合人工智能机器学习算法，以便更准确地预测未来的需求，评估工作负载的资源需求。

  - 使预测与工作负载目标保持一致

  为了确保预测与工作负载目标保持一致，需要定期对预测进行评估，比较实际结果与预测结果，根据需要对容量预测模型进行调整。例如新的应用或服务添加到系统中，那么容量预测模型就需要考虑这些新的容量需求。预测与工作负载目标的一致性，可确保充分预配资源，防止资源浪费或工作负载过载。

  - 确定资源需求

    根据需求和预测分析的结果，进行容量评估和规划。确定系统所需的计算资源、存储资源和网络带宽等资源，以满足系统的性能要求。

    - 计算资源：根据预测的需求，计算所需的CPU、GPU、内存等计算资源，并根据实际情况进行选择和配置。
    - 存储资源：根据预测的需求，计算所需的存储空间，例如需要存储大量的数据，可能需要选择分布式存储系统。

    - 网络带宽：根据预测的需求，计算所需的网络带宽，例如需要进行大规模的数据传输或者实时的网络通信，可能需要选择高速网络

- 了解资源限制

容量规划时了解和合理使用资源限制非常重要，常见的资源限制包括进程、线程、CPU使用率、内存使用量、磁盘空间等。资源限制的主要目的是保证系统的稳定性，防止某些进程或应用程序占用过多的系统资源，导致其他进程或应用程序无法正常运行，甚至导致系统崩溃。

4.7 PERF03 性能建模

4.7.1 选择合适的计算资源

评估计算要求涉及评估工作负载的特定计算需求，包括实例类型、可伸缩性和容器化等因素。不同的计算服务具有不同的功能和特征，可能会影响工作负载的性能。选择最佳计算服务以确保工作负载高效运行。请考虑以下策略：

- 了解实例类型

不同的实例类型针对不同的工作负载进行优化，例如CPU优化、内存优化和GPU优化，选择符合需求的实例类型。

- 考虑自动缩放

如果工作负载的需求不定，请考虑具有自动缩放功能的计算服务，该功能可根据需求自动调整计算容量。自动缩放有助于确保在高峰期拥有足够的资源，并防止在低需求时段过度预配。

- 考虑容器化

与非容器化工作负载相比，容器具有性能优势。如果适合体系结构需求，请考虑使用容器化。容器可以通过隔离、资源效率、快速启动时间和可移植性来提高计算性能。

使用容器时，请考虑设计因素，例如将所有应用程序组件容器化。将基于Linux的容器运行时用于轻型映像。为容器提供较短的生命周期，使其不可变且可替换。从容器、容器主机和基础群集收集相关日志和指标。使用此数据监视和分析性能。容器只是整体体系结构的一个组件。选择适当的容器业务流程协调程序（如Kubernetes），以进一步增强性能和可伸缩性。

4.7.1.1 PERF03-01 选择合适类型的计算云服务

- **风险等级**

  中

- **关键策略**

  根据应用的特征选择合适的计算云服务。选择计算云服务主要考虑以下两个因素：

  - 应用本身的部署形态
  - 上云时，业务的迁移方式（例如：业务是简单的迁移上云，还是本身要做改造）

  如果业务本身在IDC部署模式是虚拟机部署，应用系统比较老旧，业务本身也没有改造的计划，建议按照原来IDC的部署模式，采用ECS或者BMS的形式进行应用部署，以满足应用和业务本身的性能诉求。

如果借助上云的机会业务侧也会做容器化和微服务的改造，上云时选择容器化（CCE/CCI）进行部署。

如果本身业务已经是微服务化的形式，建议上云时考虑容器服务（CCE/CCI）进行部署。

- **相关服务和工具**
  - 弹性云服务器 ECS
  - 裸金属服务器 BMS
  - 云容器引擎 CCE
  - 云容器实例 CCI

4.7.1.2 PERF03-02 选择合适规格的虚拟机和容器节点

- **风险等级**

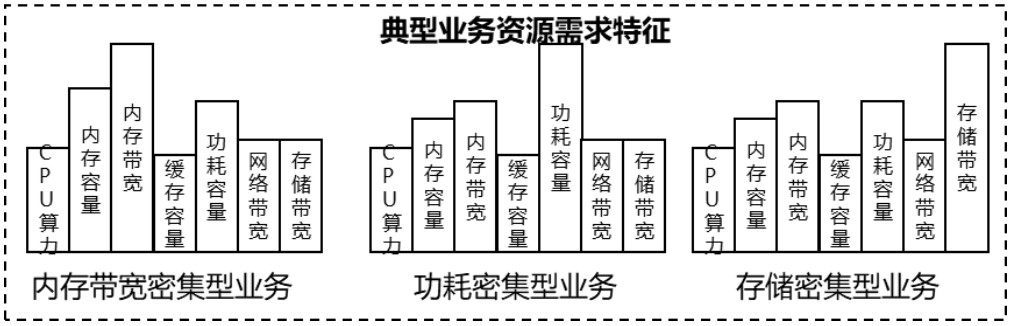
  中

- **关键策略**

  服务器资源就类似一块块资源拼成的木桶，其最多能承载的业务需求取决于哪一块资源最先达到瓶颈。

  不同应用对资源需求不同，例如：

  - 功耗密集型业务（如高性能计算、人工智能、深度学习等场景）主要就是消耗计算维度的容量。
  - 内存密集型业务（如大数据处理、图像/视频处理、游戏开发、数据库等场景）主要消耗内存和存储维度的容量。
  - 存储密集型业务（如大型数据库、大数据分析、大规模文件存储、编译构建等场景）可能会比较消耗存储的带宽。



根据业务的特征选择合适的虚拟机类型和规格。具体的虚拟机类型规格请参考**官方文档**。

- **相关云服务和工具**
  - 弹性云服务器 ECS
  - 裸金属服务器 BMS

4.7.1.3 PERF03-03 使用弹性伸缩

- **风险等级**

  中

- **关键策略**

  如果工作负载能够支持弹性（例如：应用无状态化），请考虑具有自动缩放功能的计算服务，该功能可根据需求自动调整计算容量。自动缩放有助于确保在高峰

期拥有足够的资源，并防止在低需求时段过度预配。虚拟机弹性伸缩和容器弹性伸缩都是实现应用自动化扩容和缩容的方式，但虚拟机弹性伸缩需要更多的资源和时间来启动和部署，而容器弹性伸缩可以更快速地响应变化，同时具有更高的资源利用率。虚拟机场景可以使用AS，容器场景充分考虑CA和HPA的弹性策略。

使用容器时弹性策略可参考下面内容：

CCE的弹性伸缩能力分为如下两个维度：

- **工作负载弹性伸缩**：即调度层弹性，主要是负责修改负载的调度容量变化。例如，HPA是典型的调度层弹性组件，通过HPA可以调整应用的副本数，调整的副本数会改变当前负载占用的调度容量，从而实现调度层的伸缩。
- **节点弹性伸缩**：即资源层弹性，主要是集群的容量规划不能满足集群调度容量时，会通过弹出ECS资源的方式进行调度容量的补充。

两个维度的弹性组件与能力可以分开使用，也可以结合在一起使用，并且两者之间可以通过调度层面的容量状态进行解耦，详情请参见**使用HPA+CA实现工作负载和节点联动弹性伸缩**。

工作负载弹性组件介绍

类型	组件名称	组件介绍	参考文档
HPA	**Kubernetes Metrics Server**	Kubernetes内置组件，实现Pod水平自动伸缩的功能，即Horizontal Pod Autoscaling。在kubernetes社区HPA功能的基础上，增加了应用级别的冷却时间窗和扩缩容阈值等功能。	**HPA策略**
CustomedHPA	**CCE容器弹性引擎**	自研的弹性伸缩增强能力，主要面向无状态工作负载进行弹性扩缩容。能够基于指标（CPU利用率、内存利用率）或周期（每天、每周、每月或每年的具体时间点）。	**CustomedHPA策略**
Prometheus	**Prometheus（停止维护）** **云原生监控插件**	一套开源的系统监控报警框架，负责采集kubernetes集群中kubelet的公开指标项（CPU利用率、内存利用率）。	NA

类型	组件名称	组件介绍	参考文档
CronHPA	CCE容器弹性引擎	CronHPA可以实现在固定时间段对集群进行扩缩容，并且可以和HPA策略共同作用，定时调整HPA伸缩范围，实现复杂场景下的工作负载伸缩。	CronHPA定时策略

节点弹性伸缩组件介绍

组件名称	组件介绍	适用场景	参考文档
CCE集群弹性引擎	Kubernetes社区开源组件，用于节点水平伸缩，CCE在其基础上提供了独有的调度、弹性优化、成本优化的功能。	全场景支持，适合在线业务、深度学习、大规模成本算力交付等。	节点自动伸缩
CCE突发弹性引擎（对接CCI）	将Kubernetes API扩展到无服务器的容器平台（如CCI），无需关心节点资源。	适合在线突增流量、CI/CD、大数据作业等场景。	CCE容器实例弹性伸缩到CCI服务

- **相关云服务和工具**
  - 弹性伸缩 AS
  - 云容器引擎 CCE
  - 云容器实例 CCI

4.7.2 选择合适网络服务资源

选择合适的网络服务资源是一个复杂的过程，需要考虑许多因素。以下提供了一些主要因素：

评估合适网络云服务，主要考虑如下性能指标：

- 网络流量：评估工作负载的预期网络流量，了解数据传输需求和网络请求的频率。
- 带宽要求：确定工作负载的带宽要求，考虑通过网络传输和接收的数据量。
- 网络延迟：评估工作负载所需的延迟，使用专用虚拟网络和主干网络，而不是遍历公共Internet。此方法可降低工作负载的延迟。

- 吞吐量：请考虑工作负载所需的吞吐量。配置网络路由选项以利用网络吞吐量优势。

4.7.2.1 PERF03-04 选择合适类型的网络云服务

- **风险等级**

  中

- **关键策略**

  根据网络特征，选择合适类型的网络云服务。

场景	华为云服务	选择策略
云上组网（云内、云间）	VPC	在逻辑隔离的虚拟网络中定义和启动华为资源，方便管理、配置内部网络。
	ER	将VPC和本地网络连接到一个网关中，支持路由学习、动态选路以及链路切换，极大的提升网络的可扩展性及运维效率，从而保证业务的连续性。
	ESW	将VPC和本地网络连接到一个网关中（二层互联），助力企业客户灵活构建大规模、高性能、高可靠的云上/云下网络。
	NAT gateway	通过地址转换的方式，使多个云主机可以共享私网IP访问用户本地数据中心或其他VPC，并支持云主机面向私网提供服务。
应用组网（用户<->云）	ELB	针对HTTP/HTTPS的流量做负载分发，扩展应用系统对外的服务能力，提高应用程序的容错能力。
	VPC Endpoint	在 VPC 与华为云服务之间建立连接，而无需将数据暴露于互联网；提供性能更加强大、更加灵活的网络。
接入网络（用户<->PoP）	DNS	提供高可用，高扩展的权威DNS服务和DNS管理服务，将最终用户路由到互联网应用程序的可靠且经济高效的方法。
	EIP	提供独立的公网IP资源，连接公共互联网和VPC虚拟网络，可以与弹性云服务器、裸金属服务器、虚拟IP、弹性负载均衡、NAT网关等资源灵活地绑定及解绑。
	Global Accelerator	使用华为云全球网络提升应用程序的可用性、性能和安全性，使终端用户在全球能快速访问云上应用，获得优质体验。
	Direct Connect	用于搭建用户本地数据中心与华为云VPC之间高速、低时延、稳定安全的专属连接通道，实现灵活一体，可伸缩的混合云计算环境。
	VPN	安全的远程连接到华为云或者本地资源，将已有数据中心无缝扩展到华为云上。

场景	华为云服务	选择策略
混合组网	CC	提供构建、管理和监控全球广域网能力，帮助用户打造一张具有企业规模和通信能力的全球云上网络。

4.7.3 选择合适的存储云服务

了解数据特征（如可共享、大小、访问模式、延迟、吞吐量和数据持久性），以便为您的工作负载选择合适的专用数据存储。

4.7.3.1 PERF03-05 选择合适类型的存储云服务

- **风险等级**

  中

- **关键策略**

  在架构设计过程中，根据业务场景、数据特征等因素，选择相应的存储服务。目前可供您选择的有三种数据存储服务，分别是云硬盘、弹性文件服务（Scalable File Service, SFS）以及对象存储服务（Object Storage Service, OBS），这三种数据存储的主要区别如下：

对比维度	弹性文件服务	对象存储服务	云硬盘
概念	提供按需扩展的高性能文件存储，可为云上多个云服务器提供共享访问。弹性文件服务就类似Windows或Linux中的远程目录。	提供海量、安全、高可靠、低成本的数据存储能力，可供用户存储任意类型和大小的数据。	可以为云服务器提供高可靠、高性能、规格丰富并且可弹性扩展的块存储服务，可满足不同场景的业务需求。云硬盘就类似PC中的硬盘。
存储数据的逻辑	存放的是文件，会以文件和文件夹的层次结构来整理和呈现数据。	存放的是对象，可以直接存放文件，文件会自动产生对应的系统元数据，用户也可以自定义文件的元数据。	存放的是二进制数据，无法直接存放文件，如果需要存放文件，需要先格式化文件系统后使用。
访问方式	在ECS/BMS中通过网络协议挂载使用，支持NFS和CIFS的网络协议。需要指定网络地址进行访问，也可以将网络地址映射为本地目录后进行访问。	可以通过互联网或专线访问。需要指定桶地址进行访问，使用的是HTTP和HTTPS等传输协议。	只能在ECS/BMS中挂载使用，不能被操作系统应用直接访问，需要格式化成文件系统进行访问。

对比维度	弹性文件服务	对象存储服务	云硬盘
使用场景	如高性能计算、媒体处理、文件共享和内容管理和Web服务等。 说明： 高性能计算：主要是高带宽的需求，用于共享文件存储，比如基因测序、图片渲染这些。	如大数据分析、静态网站托管、在线视频点播、基因测序和智能视频监控等。	如高性能计算、企业核心集群应用、企业应用系统和开发测试等。 说明： 高性能计算：主要是高速率、高IOPS的需求，用于作为高性能存储，比如工业设计、能源勘探这些。
容量	PiB级别	EiB级别	TiB级别
时延	3~10ms	10ms	亚毫秒级
IOPS/TPS	单文件系统 10K	千万级	单盘 128K
带宽	GiB/s级别	TiB/s级别	MiB/s级别
是否支持数据共享	是	是	是
是否支持远程访问	是	是	否
是否能单独使用	是	是	否
云服务链接	SFS官网	OBS官网	EVS官网

4.7.4 选择合适的应用中间件云服务资源

华为云提供Kafka、RocketMQ、RabbitMQ三种不同版分布式消息服务，您可根据业务需求和不同版本优势来选择合适的消息队列。

4.7.4.1 PERF03-06 选择合适的消息队列

- **风险等级**
  中
- **关键策略**

三种不同版分布式消息服务的适用场景如下：

Kafka：兼容开源Kafka，适用构建实时数据管道、流式数据处理、第三方解耦、流量削峰去谷等场景，有大规模、高可靠、高并发访问、可扩展且完全托管的特点。

RocketMQ：兼容开源RocketMQ，提供顺序、延迟、定时、重投、死信、事务与会话消息等功能，适用电商、金融场景。

RabbitMQ：兼容开源RabbitMQ，支持广播、事务消息、消息路由、死信队列、优先级队列等，适用于秒杀、流控、系统解耦等场景。

详细版本对比可参考**官方文档**。

- **相关云服务和工具：**
  - 分布式消息服务Kafka版
  - 分布式消息服务RocketMQ版
  - 分布式消息服务RabbitMQ版

4.7.4.2 PERF03-07 选择合适的 Kafka

- **风险等级**

  中

- **关键策略**

  根据生产流量、消费流量、老化时间、副本数等指标，计算业务所需的规格，选择合适的Kafka规格。

  规格测算：

  性能容量维度所需最小节点数 = max（（存储带宽需求 / 单节点存储带宽），（网络带宽需求 / 单节点网络基准带宽））

  磁盘容量维度所需最小节点数 = max（总磁盘容量需求 / 单节点磁盘容量上限）

  详细规格选择参考**官方文档**。

4.7.4.3 PERF03-08 选择合适的 RocketMQ

- **风险等级**

  中

- **关键策略**

  RocketMQ服务提供了多个维度定义规格，如资源规格、代理个数、存储容量、单个代理TPS、单个代理Topic数上限、单个代理消费组数上限等，建议根据不同版本涉及的具体规格情况选择合适的RocketMQ服务。

  详细版本与对应支持规格参数请参考**官方文档**。

4.7.4.4 PERF03-09 选择合适的 RabbitMQ

- **风险等级**

  中

- **关键策略**
  - 版本选择：RabbitMQ服务版本随时间更迭，选择版本时需注意查看不同版本状态与区分，详情可参考**官方公告**。
  - 规格选择：RabbitMQ服务提供了不同规格实例可供选择，建议按照业务需求对比，选择合适的规格型号，具体实例规格请参考**官方文档**。

4.7.5 选择合适的数据库资源

华为云提供了多款数据库服务，不同服务的优化方式和注意事项均有差异，可以通过以下四个不同考虑因素入手，选择合适的数据库资源：

- 兼容性：一般原则是平替迁移，选择云上数据库，是为了利用云上服务使得生产工作更聚焦到应用层，上云前系统中数据库的选型已经过业务实践的检验，基于对兼容性的考量（避免迁移上云后，数据库层与应用层不兼容），上云过程中采用云上同样生态的数据库进行平替，是首要的决策依据。
- 可迁移性：针对数据库上云迁移，解决方案要具备平滑迁移的能力；结合数据库迁移服务所提供的能力，评估迁移上云过程中，数据库的切换对业务系统中其他组件的影响（如服务中断的影响、数据转移效率），这是具体实施业务上云过程中的重点关切。
- 业务应用场景的评估：如果是在云上新建业务系统，则要通过业务的实际需要进行云数据库的选型，它的评估与数据库是否建立在云上无关，而是根据实际业务系统的特点来决定的。如电商系统，考虑选型MySQL满足用户信息管理、买家信息管理、交易处理的业务需求，选型MongoDB满足商品信息管理的业务需求。
- 架构设计（性能、可靠性、多区域部署、安全）约束：设计的约束，本质是业务系统在各个维度的具体需求，决定了对数据库能力规格以及资源量的具体要求；对业务性能和可靠性诉求的把握，以及对云数据库的性能负载能力和负载稳定性的评估，能够更好地帮助您选择符合业务需求的数据库服务；在做所需资源量的评估时，应基于数据库连接数、事务处理性能等关键指标的要求以及部署设计的约束（如容灾要求）来分析； 安全方面，则要针对设计约束进行逐条评估（如访问控制、数据加密），以判断数据库云服务满足度。

4.7.5.1 PERF03-10 选择合适的关系型数据库

- **风险等级**

  中

- **关键策略**

  华为云数据库提供了多款关系型数据库服务，包含GaussDB、GaussDB（for MySQL)、RDS（MySQL、PostgreSQL、SQL Server、MariaDB）；其中GaussDB、GaussDB（for MySQL）是华为推出的基于openGauss生态和MySQL生态的企业级关系型数据库，性能表现更好；RDS数据库服务可支持四种开源数据库引擎，RDS依托云计算平台，为用户提供了更稳定、更弹性、运维更便捷的在线云服务。

  适用场景：

  - **GaussDB**：基于华为主导的openGauss生态推出的企业级分布式关系型数据库，具有云上高可用、高可靠，高安全、弹性伸缩、一键部署、快速备份恢复等能力，适用于高并发、大数据量、以联机事务处理为主的交易型应用，如政务、金融、电商、O2O、电信CRM/计费等。
  - **GaussDB（for MySQL)**：华为自研的最新一代企业级高扩展高性能云原生关系型数据库，完全兼容MySQL，具有高性能、高扩展性、高可靠性、高兼容性、低成本、非中间件式架构等特点，适用金融行业、游戏行业等需要高安全或高性能的行业。
  - **RDS**：具有低成本、高性能、高安全性、高可靠性等特点。
  - **RDS for MySQL**：MySQL是当前应用最广泛的开源关系型数据库。RDS for MySQL适用于网站业务、应用程序、中小型企业等场景。

- **RDS for PostgreSQL**：PostgreSQL是一个开源对象云数据库管理系统，并侧重于可扩展性和标准的符合性。RDS for PostgreSQL适用于网站业务、位置应用系统、复杂数据对象处理等场景。
- **RDS for SQL Server**：Microsoft SQL Server是老牌商用级数据库，成熟的企业级架构，轻松应对各种复杂环境；RDS for SQL Server适用于政府、金融、医疗、教育、游戏等领域。
- **RDS for MariaDB**：MariaDB是当前比较主流开源社区数据库之一，对MySQL有较好的兼容性；RDS for MariaDB适用于各种规模的应用程序。

关系型数据库的选择建议：

场景一：基于兼容性原则

考虑平滑上云，上云前系统中数据库的选型已经过业务实践的检验，建议选取生态相同的关系型数据库服务进行平替，避免出现数据库层与应用层不兼容或数据库切换对业务架构中其他组件产生负面影响。

场景二：基于场景评估

如果是在云上新建业务系统或基于同数据库不同服务中选取时，建议结合业务的实际需要选取合适的数据库服务；例如RDS for MySQL和GaussDB（for MySQL），在选取时应考虑更多因素，如性能等因素，这些因素可以从官方资料查看。

4.7.5.2 PERF03-11 选择合适的非关系型数据库

- **风险等级**

  中

- **关键策略**

  华为云数据库提供了DDS、GeminiDB两种非关系型数据库服务。

  - **DDS**：文档数据库服务（Document Database Service）完全兼容MongoDB协议，提供安全、高可用、高可靠、弹性伸缩和易用的数据库服务，同时提供一键部署、弹性扩容、容灾、备份、恢复、监控和告警等功能，适用于游戏、物联网业务、互联网应用等多个场景。
  - **GeminiDB Redis接口**：GeminiDB Redis接口是一款基于华为自研的计算存储分离架构，兼容Redis生态的云原生NoSQL数据库，基于共享存储池的多副本强一致机制，支持久化存储，保证数据的安全可靠。具有高兼容、高性价比、高可靠、弹性伸缩、高可用、无损扩容等特点，适用于游戏、推荐、GIAmgGAP、推送等场景。
  - **GeminiDB Influx接口**：GeminiDB Influx 接口是一款基于华为自研的计算存储分离架构，兼容InfluxDB生态的云原生NoSQL时序数据库。提供大并发的时序数据读写，压缩存储和类SQL查询，并且支持多维聚合计算和数据可视化分析能力。具有高写入、灵活弹性、高压缩率和高查询等特点，适用于IoT、金融、软硬件设备实时监控、数据采集等场景。
  - **GeminiDB Cassandra接口**：GeminiDB Cassandra 接口是一款基于华为自研的计算存储分离架构，兼容Cassandra生态的云原生NoSQL数据库，支持类SQL语法CQL。具有安全可靠、超强读写、弹性扩展、便捷管理等特点，适用于互联网应用、工业数据采集等场景。
  - **GeminiDB Mongo接口**：GeminiDB Mongo 接口是一款基于华为自研的计算存储分离架构，兼容MongoDB生态的云原生NoSQL数据库。具有企业级性能、灵活弹性、高可靠、可视化管理等特点，广泛应用于游戏应用等场景。
  - **GeminiDB HBase接口**：GeminiDB HBase接口是一款兼容HBase生态的分布式NoSQL数据库，在Apache HBase的基础上进行扩展和优化，具有高性能、

高可靠性、强大的扩展性和灵活的伸缩性等特点，适用于金融、电信、物流、游戏等场景。

同关系型数据库一样，非关系型数据的选择同样主要基于兼容性与场景评估两个原则：

场景一：基于兼容性原则

考虑平滑上云，上云前系统中数据库的选型已经过业务实践的检验，建议选取生态相同的关系型数据库服务进行平替，避免出现数据库层与应用层不兼容或数据库切换对业务架构中其他组件产生负面影响。

场景二：基于场景评估

如果是在云上新建业务系统或基于同数据库不同服务中选取时，建议结合业务的实际需要选取合适的数据库服务，如考虑性能、安全性等因素，产品详细介绍与规格信息可参考官方文档。

4.8 PERF04 性能分析

4.8.1 性能测试

性能测试是一种软件测试形式，通过性能测试工具模拟正常、峰值及异常负载等状态下对系统的各项性能指标进行测试的活动，它关注运行系统在特定负载下的性能，可帮助你评估系统负载在各种方案中的功能，涉及系统在负载下的响应时间、吞吐量、资源利用率和稳定性，以帮助确保系统性能满足基线要求，有助于提早发现性能问题，防止随着系统运行可能出现的性能裂化小于基线的情况。以下内容将带领大家了解性能测试。

4.8.1.1 PERF04-01 定义验收标准

- **风险等级**

  高

- **关键策略**

  验收标准是用于评估指定工作负载是否满足性能要求的指标，需要在性能测试前期定义合理的验收标准。

  – 查看性能目标

  性能目标定义了工作负载所需的性能级别。查看为工作负载建立的性能目标。性能目标是可能涉及响应时间、吞吐量、资源利用率或任何其他相关绩效指标的指标。例如响应时间的目标可能低于特定阈值，如小于2秒。

  – 定义验收标准

  将性能目标转换为可用于评估工作负载性能的特定验收标准。例如，假设响应时间的性能目标是2秒或更短。接受条件可以是工作负载的平均响应时间应小于2秒。使用这些验收标准来确定工作负载是否满足所需的性能级别。

4.8.1.2 PERF04-02 选择合适的测试方式

- **风险等级**

  高

- **关键策略**

  性能测试的常见方式如下，需要注意的是，各种测试方式并不是正交的，而是有耦合关系的：

- 性能验收：性能验收测试的运行环境必须是确定的，验证系统在确定的场景条件下是否达到了其宣称的能力规格。
- 负载测试：是在被测系统上进行负载阶梯加载，直至摸到系统性能极限，一般用来测试系统性能容量或调优。
- 压力测试：是检查系统处于超负载压力下的性能表现，可以考察系统的流控机制和极限场景下的性能。
- 长时间稳定性测试：该测试需要在负载压力下进行，是考察性能表现稳定性的重要手段，经常结合压力测试开展。
- 配置测试：通过对被测系统软硬件配置的调整以及业务模型调整，了解不同配置对系统性能的影响，从而找到系统资源的最优分配原则、不同业务模型的性能趋势。
- 并发测试：通常通过构造多用户或多任务并发的手段来暴露可能隐藏的进程死锁、资源泄露或其他性能问题。

- **相关云服务和工具**
  - 性能测试 CodeArts PerfTest

4.8.1.3 PERF04-03 性能测试步骤

- **风险等级**

  高

- **关键策略**

  1.确定验收性能指标

  对被测系统从用户角色、开发角色、维护管理员等角色出发分析，结合生产环境系统当前情况，识别并定义业务指标、数据指标、资源指标三种维度指标需要达到的目标基线，指导系统能达到以最小的资源占用管理最大的数据并给用户提供最优的体验目标，输出系统各个场景所要达到的SLA。

  2.创建测试方案

  创建测试方案是指设计适合性能测试系统负载的特定场景或条件的过程，性能测试方案设计要求全面、无遗漏，使用测试设计模板把所有的组网场景要求全覆盖，根据性能测试需求、测试模型、测试组网图，罗列出业务测试要点，逐一去分解测试场景和步骤。创建测试方案以模拟真实的用户行为和系统负载， 这些方案为性能测试人员提供了一种方法来评估服务负载在不同条件下的性能，包括测试环境、测试工具、测试监控项等。

  通过测试方案可以复制各种系统负载档位，例如并发用户访问、峰值负载时段或特定场景。 通过测试不同的负载档位，可以识别性能瓶颈并优化部署资源，输出件是可执行性能测试方案。

  - 用户原子行为：识别测试场景，通过识别用户在与服务系统交互时大量执行的步骤和操作，模拟真实的用户行为和系统负载模式。 例如登录、执行搜索、批操场景、导入导出、提交表单或访问特定功能等活动。 将每个方案分解为表示用户与服务系统交互的特定场景步骤和操作。 可以包括页面、执行事务或与系统负载的各种混合场景。
  - 确定数据模型： 确定运行测试方案所需的测试背景数据。 可以创建或生成各种场景、用户配置文件或数据量的实际数据集。 确保测试数据多样化并涵盖不同的场景数据，以提供全面的性能评估。
  - 设计测试脚本： 创建执行定义的测试方案的测试脚本。 测试脚本通常包含一系列操作、HTTP 请求或与服务系统相关的 API 或用户界面的交互。 使用性能测试工具编写脚本，考虑参数化、动态数据处理等因素。 调试脚本，例如脚本错误、缺少或不正确的操作或与数据相关的问题，测试脚本的准确对于帮助确保准确可靠的性能测试执行至关重要。

- 配置测试变量和参数： 在测试脚本中配置变量和参数，以模拟真实场景请求。 包括多用户登录、查询数据或随机化等参数，以模拟不同的用户行为和工作负载响应。
- 脚本自动化： 脚本自动化，根据反馈、测试结果或更改要求不断优化和更改测试脚本。 考虑优化脚本逻辑、参数化和错误处理，或添加额外的验证和检查点，配置自动化脚本，迭代测试无需更改脚本。

3.配置测试环境

性能测试环境是指用于开展性能测试活动的环境，为性能测试服务，性能测试环境原则上与生产环境进行对标，在测试过程中需要保持其独立性，尽量避免其它因素对性能测试结果的影响，软件版本、系统组网、硬件规格等要保持与生产环境基本一致。

性能测试环境配置通常要考虑以下因素：

- 系统组网与架构：系统组网方式如主备、集群、分布式等组网，系统架构分析服务间依赖关系，确定周边依赖服务。
- 硬件规格：所需服务器的数量、规格以及硬件配置，包括 CPU 主频/核数、内存容量、磁盘类型与容量、存储池类型与容量，网卡带宽等。
- 软件环境：软件版本与配置，如操作系统版本、服务版本、数据库版本、以及影响性能的相关配置。

4.完成测试设计

在本步骤完成前文确认的系统负载、背景数据量与需要请求的用户数据模型等测试设计。

5.执行测试

使用所选的测试工具进行性能测试，测试涉及查看和记录性能指标、监控运行情况以及查看出现的任何性能问题，同时监控和收集性能指标，例如响应时间、吞吐量、CPU和内存利用率以及其他相关指标。

使用定义的测试方案将工作负载置于预期负载之下。在这些不同的负载条件下进行测试。例如，使用正常、峰值和压力级别等级别来分析各种方案中工作负载的行为。

6.分析测试结果

分析测试结果是指检查从性能测试结果收集的测试结果和记录的监控指标，由此分析服务的瓶颈点，分析确认是性能问题的，需要提单优化。从以下几个方面展开分析：

- 查看性能指标：查看性能测试期间收集的性能指标，例如响应时间、吞吐量、错误率、CPU 和内存利用率以及网络使用等。分析这些指标以了解服务的整体性能。
- 确定性能瓶颈：评估性能指标，以确定哪些性能指标是该场景测试的瓶颈点。评估包括高响应时间、资源使用、数据库问题、网络延迟和扩容限制等。确定服务场景的瓶颈点，有助于服务的优化改进与扩缩容处理。
- 性能关联指标：评估各种性能指标之间的关系和相关性。例如，背景数据量和资源利用率影响服务的响应时间，清楚这些关联关系可以为不同环境下的场景提供有价值的性能指导，在需要扩容的时候知道扩容哪部分节点给服务以数据支撑。
- 评估验收条件：将测试结果与预定义的验收条件SLA进行比较。评估服务当前性能是否满足生产环境所需的性能要求。如果不满足生产环境性能调用，需要优化服务或者扩容等手段进行处理

7.确定基线

性能基线是产品在系统形成时，所建立的一个基线库，性能基线按照分层原则，建立后通常在版本设计阶段就会归档与发布。基线提供了一个参考点，用于比较一段时间内的性能结果，基线应该是工作负载性能的参考

考虑工作负载目标，并记录性能，以便随着版本迭代比较基线和优化性能。使用这些基线度量作为未来性能测试的基准，并用它们来识别服务系统有无性能裂化。

若要为性能测试建立基线并将其用作未来性能测试的基准，请执行以下步骤：

- 确定性能指标：确定要度量和约定的性能指标。示例包括：

  响应时间，或服务响应请求的速度。

  吞吐量，或按单位时间处理的请求数。

  资源利用率，例如CPU、内存和磁盘使用率。
- 记录性能相关的度量值：将测试期间获得的性能指标记录为基线度量值。这些度量与测试前约定的SLA比较值。
- 比较将来的测试：在后续性能测试中，将性能指标与已建立的基线和阈值进行比较。通过比较，可以识别有可能出现的性能裂化等。

性能基线管理的几个原则与要求：

- 性能基线不允许随意更改：性能基线已经发布后，则在同一个版本内不允许进行更改，如果要更改则需要与产品相关SE、产品经理对齐，重新进行发布。
- 性能基线则需要持续继承：性能基线的数值则在每个版本设计阶段进行确认完成，不能出现之前的基线内容全部推翻重来。
- 性能基线的维护主体：性能基线一经发布，则测试严格按照性能基线要求进行测试与评估。
- 非性能基线测试则有权进行拒绝验证：对于不合理的非产品基线内容测试则可以进行拒绝验证。

基线不合理内容测试则有权利进行质疑，并且能够根据测试数据、现网应用、客户规范、标准等要求与SE、产品经理进行评审，基线变更等。

4.8.2 性能数据采集

收集性能数据是收集指标和日志的过程，这些指标和日志提供有关工作负载性能的信息。 此数据包括数值，称为指标。 指标描述系统在特定时间点的状态。 它还包括包含组织成记录的不同类型的数据的日志。

通过收集性能数据，可以监视和分析工作负载的性能。 可以使用此信息来识别性能瓶颈、解决问题、优化资源分配，以及做出数据驱动的决策，以提高工作负载的整体性能效率。

影响：如果没有数据驱动的见解，你可能不知道潜在的性能问题或优化机会。 潜在结果包括响应时间变慢、吞吐量降低、资源使用率增加，最终用户体验欠佳。 此外，由于缺少性能数据，因此难以及时诊断和排查问题，从而导致停机时间延长并降低工作效率。

4.8.2.1 PERF04-04 资源性能数据收集

- **风险等级**

  中
- **关键策略**

每个华为云提供的云服务都有一组特定于资源功能的指标，用于呈现有关资源的使用情况。通过收集资源性能数据，可以深入了解工作负载的运行状况和行为。

指标作用：

- 帮助你了解资源的运行状况和性能，
- 在云监控平台上配置对应的告警策略和配置指标看板。
- 通过跟踪分析网络路径上的流量来优化网络性能。

- **相关云服务和工具**
  - 云监控服务 CES

4.8.2.2 PERF04-05 应用性能数据采集

- **风险等级**

  中

- **关键策略**

  应用程序的性能数据（吞吐量、延迟和完成时间），通常需要通过代码采集，例如嵌入代码片段或将工具集成到应用程序代码中。通过应用的性能数据，可以识别性能瓶颈、评估系统行为、识别可用性风险、规划容量等指标。

  常用应用性能监控策略有：

  - APM 工具：可用使用云上APM 工具或者开源的APM工具和分析性能数据（指标、日志、调研链）
  - 使用基于日志调用链框架：这些框架具备日志生成、日志格式化、日志上下文关联分析登能力。 通过框架引入到代码库中，可以在运行时采集相关的性能数据。
  - 自定义检测：仅当平台指标不足时，才建议开发人员可以添加自定义代码采集独有的性能指标。
  - 使用业界可观测的标准。请考虑使用围绕业界标准构建的工具，例如OpenTelemetry。

  建议：使用分布式的调用链技术，可以识别多个服务和组件之间请求链路；通过收集调用链数据实现数据流端到端的分析，产品阻塞瓶颈点或者效率低下的请求片段，从而进行针对性的优化。

- **相关云服务和工具**
  - 应用运维管理 AOM
  - 应用性能管理 APM
  - 云日志服务LTS

4.8.3 建立性能可观测性体系

4.8.3.1 PERF04-06 建立性能可观测性体系

- **风险等级**

  中

- **关键策略**

  可观测性体系是指在云原生架构中通过使用各种工具和技术来实现对应用程序和基础设施的监控告警、日志、故障排除等功能的一套完整的解决方案。性能可观测体系在此基础上突出了性能指标，通过收集和分析性能数据，可以识别系统瓶颈、优化资源分配等，找到性能优化方向。

性能监控对象：服务器、操作系统、数据库、应用程序、网络设备、云服务。

常见性能指标：包括资源CPU、内存，硬盘等，及程序的响应时间、吞吐量、并发数等。

4.9 PERF05 性能优化

性能优化工作中，需警惕“过早优化”的问题。我们的基本指导策略还是首先让系统运行起来，再考虑怎么让它变得更快。一般只有在我们证实某部分代码的确存在一个性能瓶颈的时候，才应进行优化。除非用专门的工具分析瓶颈，否则很有可能是在浪费自己的时间。另外，性能优化的隐含代价会使我们的代码变得难于理解和维护，这一点也是需要权衡和关注的。

4.9.1 设计优化

4.9.1.1 PERF05-01 设计优化

- **风险等级**

  中

- **关键策略**

  – 快速通道模式

  通过减少支配性工作量负载的处理量，只剩下必要的部分，来改进响应的时间。一个软件可以有多项功能，只有几个是被经常使用的，经常使用的功能构成支配性工作量负载。快速通道模式减少这些功能的处理量，或简化其处理过程。快速通道通过简化执行路径的方式来实现，简化路径即是快速通道。

  快速通道的前提是识别出支配性工作量，可以根据某项功能的使用频率来选择。常见的快速通道如，页面快速导航键、DB的索引等。

  – 重要事情优先

  把资源优先用于或者集中在重要的任务处理上，确保重要任务的完成；如果不能在可用的时间内完成所有事情，被忽略的是最不重要的任务。主要用于处理瞬时突发负载导致超出系统处理的容量的情况，一般给重要任务赋予高优先级，最重要的行为优先得到处理。只适用于暂时超载的情况，如果超载不是暂时的，需要减少处理量，或者升级系统。如在性能过载场景下，按照功能优先级进行熔断间接，保证主要功能可用。

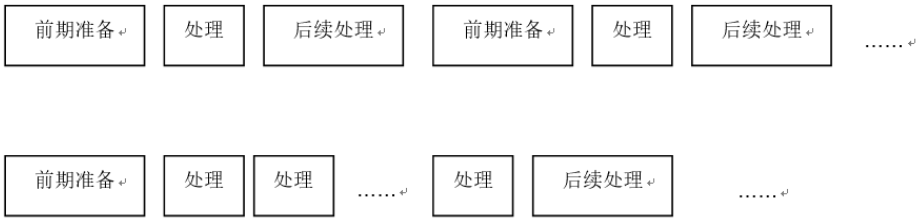
  – 聚合

  将大多数场合在一起使用的功能组合在一起，以减少调用的交互次数。

  本模式要求将组合调用居多的一些子功能，合并起来使用。聚合这个模式要求尽量将相关或紧耦合的功能放到一个对象中，使用本地接口，避免在外部接口或重开销的接口（如CORBA接口），呈现小粒度对象。聚合模式使用更粗粒度的对象，经常被访问的数据应当组合成一个聚合物，以消除对少量信息的频繁请求。如，帐户类CustAcct可以提供访问函数getName()，getAddress()，getZip()，如果经常用到该类的任务是创建邮件标签，可以使用一个新函数genMailLableInfo()，以便调用一次取得所有信息，减少交互次数。

  – 批处理

  把经常性的服务请求合并到一起，节省请求的初始化、传输、终止的处理开销。当请求的任务初始化、传输、终止的开销较大时，系统的额外开销可能超过真正的处理时间。通过将请求合并为批处理，开销处理为一批请求所分摊，不再是单独分别执行一次，从而提高处理效率。如DB的批量保存、Redis的popeline方式都是常用的批处理操作



批处理过程模型

– 替代路由（Bypass）

从空间上分散对高使用率对象的请求，将请求分散到其他对象或者对象的其他位置，以降低争用延时.类似于通过一条替代路线，绕开交通瓶颈，到达目的地。具体方案一是对目标对象进行空间划分，划分成小粒度对象，操作分散到不同物理位置；二是增加单独线程，每个线程更新自身的数据区域。

比如进程必须与下游单个进程协作时，只有一个下游进程，在高负载的情况下跟不上，造成交通堵塞。可以使用多个下游进程，每个进程只更新其自身的数据区域。

– 弹性时间

在时间上分散对高使用率对象的要求，分散到不同的时间段上，减少对这些对象的争用延时。

很多请求在某一时刻同时发出，导致要求返回的结果激增，响应缓慢。而在其他时间，很少或者没有请求。当处理请求以特定频率出现，或者在一天中的某个特定时间出现，就会产生这种情况。识别那些定期的，以特定间隔反复处理的功能，然后修改他们的处理时间，在给定时间范围内，随机分散到不同时间，以解决这个问题。

– 空间换时间

通过使用更多的存储空间，以节省执行时间。

空间换时间包括简单地预先存储结果，或者存储经常被访问的数据以方便计算；另一种空间换时间则包括选择特定的算法，如HASH算法就是一种典型的空间换时间的算法。另一种是OLAP技术，在此技术中，数据被按照一定的层级关系预先汇总，这样会大辐降低后续查询的耗时。

比如在慢SQL优化的时候，常用收段是识别频繁访问的字段并且设置索引，通过索引来缩短访问时延。

– 处理有效负载

识别出必须要处理的数据，排除对其他数据的重复处理。在一项处理数据的操作中，并非所有的处理数据都是必须处理的，可以通过分析，识别出必须处理的数据。可以有多种方式，来减轻负载的方法，如增量处理、变化通知等。

- 增量处理
- 变化通知

有效减负的反模式是“负载过重”，是在处理过程中，处理了大量不必要的处理数据，大辐增加了处理负载。出现这种情况的原因，有时为了简化处理过程，有时候是没有做必要性分析。可以通过对待处理数据进行分析，筛选出必须处理的数据，重新设计处理方案。

– 串行同步衔接

给线程/进程间的串行衔接设计紧密的衔接和同步措施，减少同步等待延时。

将一个串行处理通过划分多个线程并行处理，但各线程的处理又有一定的顺序和同步关系，需要设计合适的衔接和同步措施。同步方式可以是定时查询，也可以选择消息同步，或Event关联，选择的原则是衔接时间缝隙满足性能目标要求。

这个方法最佳的效果只是让整体性能略等于处理环节中最差的一个环境的性能（忽略线程切换及信号量等切换时间）。

4.9.2 算法优化

4.9.2.1 PERF05-02 通用算法优化

- **风险等级**

  中

- **关键策略**

  算法优化是提高程序性能的关键，可以通过改进算法的设计和实现方式来提高其效率和性能。以下是一些最佳实践：

  - 使用正确的数据结构：选择合适的数据结构可以大辐提高算法的效率。例如，使用哈希表可以快速查找元素，使用数组可以快速访问元素。
  - 减少内存分配：内存分配是一个耗时的操作。可以通过预先分配内存或者重复使用已分配的内存来减少内存分配。
  - 减少循环次数：循环是一个常见的算法结构，但是循环次数过多会导致程序性能下降。可以通过使用更高效的算法来减少循环次数。
  - 使用并行计算：对于一些计算密集型的算法，可以使用并行计算来提高程序性能。可以使用多线程或者分布式计算来实现并行计算。

4.9.3 资源优化

4.9.3.1 PERF05-03 WEB 场景资源优化

- **风险等级**

  中

- **关键策略**

  对于已经配置好的资源，可以通过优化来提高性能。例如，优化操作系统的设置、调整网络带宽、优化数据库查询等。

  云服务资源性能优化步骤包括:

  - 识别性能瓶颈: 通过监控和分析云服务资源使用情况，找出性能瓶颈。
  - 优化资源配置: 根据性能瓶颈，调整云服务资源的配置，如 CPU 、内存、网络等。
  - 使用缓存: 使用缓存技术，如 CDN 、 Redis 等，提高数据访问速度。
  - 代码优化: 对云服务资源使用的代码进行优化，提高代码执行效率。
  - 数据库优化: 对云服务资源使用的数据库进行优化，如索引优化、查询优化等。
  - 负载均衡: 使用负载均衡技术，将请求分发到多个云服务资源，提高系统的处理能力。
  - 监控和调整: 持续监控云服务资源的性能，根据实际情况进行调整，以保持最佳性能。

4.9.3.2 PERF05-04 大数据场景资源优化

- **风险等级**

  中

- **关键策略**

  在大数据场景下，可以通过优化资源的使用和分配，提高系统的性能和效率。以下是一些常见的大数据场景资源优化方法：

  - 分布式存储：使用分布式存储系统，如Hadoop HDFS、Apache Cassandra等，将数据分散存储在多个节点上，以提高数据的可靠性和可扩展性。
  - 数据压缩：对于大量的数据，可以采用压缩算法进行压缩，以减少数据的存储空间和传输带宽。
  - 并行计算：使用并行计算框架，如Apache Spark、Apache Flink等，将计算任务分配到多个节点上并行执行，以提高计算速度和效率。
  - 内存优化：通过调整内存分配和使用策略，如使用内存缓存、内存映射等技术，以提高数据处理和计算的速度和效率。
  - 负载均衡：通过负载均衡技术，将数据和计算任务均匀地分配到多个节点上，以避免单个节点过载，提高系统的可用性和性能。
  - 数据分区：将数据按照一定的规则分成多个分区，以便更好地进行数据处理和计算。
  - 网络优化：通过优化网络带宽、延迟等参数，以提高数据传输的速度和效率。
  - 数据清洗和预处理：在进行大数据处理之前，对数据进行清洗和预处理，以提高数据的质量和准确性，减少后续计算的错误率和计算量。

4.10 PERF06 性能看护

4.10.1 性能看护

4.10.1.1 PERF06-01 分层看护

- **风险等级**

  高

- **关键策略**

  基于业务的部署架构，一般可以从最底层的硬件基础设施到最上层的应用分成5层资源，云上服务可以只需要关注虚拟网络、实例、应用三层。结合每一层资源的特征指标进行分层建模，分别设置不同梯度的性能看护指标。通常按照指标劣化程度可以设计成一般、紧急、重要三个梯度，对应每个梯度的指标配套对应的处理措施。对于敏感度或业务重要度的应用架构，可以新增一个提示级别的梯度。

- **相关云服务和工具：**
  - 云监控服务 CES
  - 应用运维管理 AOM
  - 应用性能管理APM

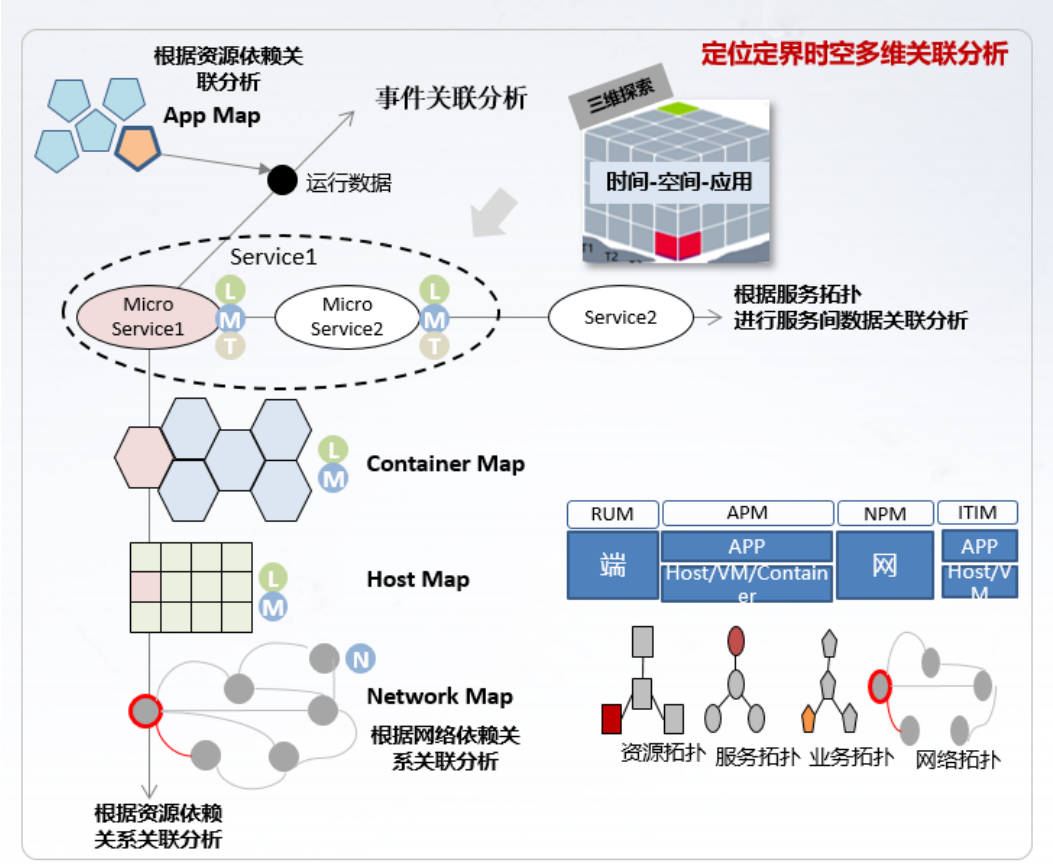
4.10.1.2 PERF06-02 性能劣化自动定界定位

- **风险等级**

  中

- **关键策略**

  通过建立的分层性能模型，判断系统是否会出现性能劣化的情况。当出现劣化事件时，需要通过自动化手段快速定位定界发现根因。可以通过应用模型建设三维的拓扑，把架构-空间-时间数据关联起来。这里面的关键是架构模型的建立及分层指标的聚合可视化能力，需要依赖持续的资源治理和数据治理。



- **相关云服务和工具：**
  - 优化顾问 OA
  - 云监控服务 CES
  - 应用运维管理 AOM

4.10.1.3 PERF06-03 自动告警

- **风险等级**

  中
- **关键策略**

  通过在云监控平台配置对应的告警策略，可以及时了解资源风险，以便做出对应调整和策略。
- **相关云服务和工具：**
  - 优化顾问 OA
  - 云监控服务 CES

4.11 云服务性能优化介绍

4.11.1 缓存性能优化

以下章节我们结合一些具体建议和案例来说明如何针对缓存的使用进行性能优化。

Redis使用规范

如下的规范可以帮助我们在系统运行过程中，尽可能减少遇到redis不稳定或异常的概率， 保证系统的长稳运行。

业务使用规范

原则	原则说明	级别	备注
就近部署业务，避免时延过大	如果部署位置过远（非同一个region）或者时延较大（例如业务服务器与Redis实例通过公网连接），网络延迟将极大影响读写性能。	强制	如果对于时延较为敏感，请避免创建跨AZ Redis实例。
冷热数据区分	建议将热数据加载到 Redis中。低频数据可存储在 Mysql或者ElasticSearch中。	建议	Redis将低频数据存入内存中，并不会加速访问，且占用Redis空间。
业务数据分离	避免多个业务共用一个Redis。	强制	一方面避免业务相互影响，另一方面避免单实例膨胀，并能在故障时降低影响面，快速恢复。
	禁止使用select功能在单Redis实例做多db区分。	强制	Redis单实例内多DB隔离性较差，Redis开源社区已经不再发展多DB特性，后续不建议依赖该特性。
设置合理的内存淘汰（逐出）策略	合理设置淘汰策略，可以在Redis内存意外写满的时候，仍然正常提供服务。	强制	DCS默认的逐出策略为volatile-lru，请根据业务需求选择。 **Redis支持的数据逐出策略**
以缓存方式使用Redis	Redis事务功能较弱，不建议过多使用。	建议	事务执行完后，不可回滚。
	数据异常的情况下，支持清空缓存进行数据恢复。	强制	Redis本身没有保障数据强一致的机制和协议，业务不能强依赖Redis数据的准确性。
	以缓存方式使用Redis时，所有的key需设置过期时间，不可把Redis作为数据库使用。	强制	失效时间并非越长越好，需要根据业务性质进行设置。
防止缓存击穿	推荐搭配本地缓存使用Redis，对于热点数据建立本地缓存。本地缓存数据使用异步方式进行刷新。	建议	-
防止缓存穿透	非关键路径透传数据库，建议对访问数据库进行限流。	建议	-
	从Redis获取数据未命中时，访问只读数据库实例。可通过域名等方式对接多个只读实例。	建议	核心是未命中的缓存数据不会打到主库上。 用域名对接多个只读数据库实例，一旦出现问题，可以增加只读实例应急。

原则	原则说明	级别	备注
不用作消息队列	发布订阅场景下，不建议作为消息队列使用。	强制	• 如没有非常特殊的需求，不建议将 Redis 当作消息队列使用。 • Redis 当作消息队列使用，会有容量、网络、效率、功能方面的多种问题。 • 如需要消息队列，可使用高吞吐的Kafka或者高可靠的RocketMQ。
合理选择规格	如果业务增长会带来Redis请求增长，请选择集群实例（Proxy集群和Cluster集群）	强制	单机和主备扩容只能实现内存、带宽的扩容，无法实现计算性能扩容。
	生产实例需要选择主备或者集群实例，不能选用单机实例	强制	-
	主备实例，不建议使用过大的规格。	建议	Redis在执行RewriteAOF和BGSAVE的时候，会fork一个进程，过大的内存会导致卡顿
具备降级或容灾措施	缓存访问失败时，具备降级措施，从DB获取数据；或者具备容灾措施，自动切换到另一个Redis使用。	建议	-

数据设计规范

分类	原则	原则说明	级别	备注
Key相关规范	使用统一的命名规范。	一般使用业务名（或数据库名）为前缀，用冒号分隔。Key的名称保证语义清晰。	建议	例如，业务名:子业务名:id
	控制Key名称的长度。	在保证语义清晰的情况下，尽量减少Key的长度。有些常用单词可使用缩写，例如，user缩写为u，messages缩写为msg。	建议	建议不要超过128字节（越短越好）。

分类	原则	原则说明	级别	备注
	禁止包含特殊字符（大括号“{}”除外）。	禁止包含特殊字符，如空格、换行、单双引号以及其他转义字符。	建议	由于大括号“{}”为Redis的hash tag语义，如果使用的是集群实例，Key名称需要正确地使用大括号避免分片不均的情况。
Value相关规范	设计合理的Value大小。	设计合理的Key中Value的大小，推荐小于10 KB。	建议	过大的Value会引发分片不均、热点Key、实例流量或CPU使用率冲高等问题，还可能导致变更规格和迁移失败。应从设计源头上避免此类问题带来的影响。
	设计合理的Key中元素的数量。	对于集合和列表类的数据结构（例如Hash，Set，List等），避免其中包含过多元素，建议单Key中的元素不要超过5000个。	建议	由于某些命令（例如HGETALL）的时间复杂度直接与Key中的元素数量相关。如果频繁执行时间复杂度为O(N)及以上的命令，且Key中的子Key数量过多容易引发慢请求、分片流量不均或热点Key问题。
	选择合适的数据类型。	合理地选择数据结构能够节省内存和带宽。	建议	例如存储用户的信息，可用使用多个key，使用set u:1:name "X"、set u:1:age 20存储，也可以使用hash数据结构，存储成1个key，设置用户属性时使用hmset一次设置多个，同时这样存储也能节省内存。
	设置合理的过期时间。	合理设置Key的过期时间，将过期时间打散，避免大量Key在同一时间点过期。	建议	设置过期时间时，可以在基础值上增减一个随机偏移值，避免在同一个时间点大量Key过期。大量Key过期会导致CPU使用率冲高。

命令使用规范

原则	原则说明	级别	备注
谨慎使用O(N)复杂度的命令	时间复杂度为O(N)的命令，需要特别注意N的值。避免N过大，造成Redis阻塞以及CPU使用率冲高。	强制	例如：hgetall、lrange、smembers、zrange、sinter这些命令都是做全集操作，如果元素很多，会消耗大量CPU资源。可使用hscan、sscan、zscan这些分批扫描的命令替代。

禁用高危命令	禁止使用flushall、keys、hgetall等命令，或对命令进行重命名限制使用。	强制	请参考**命令重命名**的内容。
慎重使用select	Redis多数据库支持较弱，多业务用多数据库实际还是单线程处理，会有干扰。最好是拆分使用多个Redis。	建议	-
使用批量操作提高效率	如果有批量操作，可使用mget、mset或pipeline，提高效率，但要注意控制一次批量操作的元素个数。	建议	mget、mset和pipeline的区别如下： ● mget和mset是原子操作，pipeline是非原子操作。 ● pipeline可以打包不同的命令，mget和mset做不到。 ● 使用pipeline，需要客户端和服务端同时支持。
避免在lua脚本中使用耗时代码	lua脚本的执行超时时间为5秒钟，建议不要在lua脚本中使用比较耗时的代码。	强制	比如长时间的sleep、大的循环等语句。
避免在lua脚本中使用随机函数	调用lua脚本时，建议不要使用随机函数去指定key，否则在主备节点上执行结果不一致，从而导致主备节点数据不一致。	强制	-
遵循集群实例使用lua的限制	遵循集群实例使用lua的限制。	强制	● 使用EVAL和EVALSHA命令时，命令参数中必须带有至少1个key，否则客户端会提示“ERR eval/evalsha numkeys must be bigger than zero in redis cluster mode”的错误。 ● 使用EVAL和EVALSHA命令时，DCS Redis集群实例使用第一个key来计算slot，用户代码需要保证操作的key是在同一个slot。
对mget，hmget等批量命令做并行和异步IO优化	某些客户端对于MGET，HMGET这些命令没有做特殊处理，串行执行再合并返回，效率较低，建议做并行优化。	建议	例如Jedis对于MGET命令在集群中执行的场景就没有特殊优化，串行执行，比起lettuce中并行pipeline，异步IO的实现，性能差距可达到数十倍，该场景建议使用Jedis的客户端自行实现slot分组和pipeline的功能。

禁止使用del命令直接删除大Key	使用del命令直接删除大Key（主要是集合类型）会导致节点阻塞，影响后续请求。	强制	Redis 4.0后的版本可以通过UNLINK命令安全地删除大Key，该命令是异步非阻塞的。 对于Redis 4.0之前的版本： • 如果是Hash类型的大Key，推荐使用hscan + hdel • 如果是List类型的大Key，推荐使用ltrim • 如果是Set类型的大Key，推荐使用sscan + srem • 如果是SortedSet类型的大Key，推荐使用zscan + zrem

4.11.2 消息队列性能优化

以下章节我们结合一些具体建议和指标来说明如何针对消息队列的使用进行性能优化。

4.11.2.1 Kafka 性能优化

Kafka性能优化

优化客户端配置

生产者配置建议

可参考**配置建议**。

消费者配置建议

参数	推荐值	说明
max.poll.records	500	消费者一次能消费到的最大消息数量，默认为500，如果每条消息处理时间较长，建议调小该值，确保在max.poll.interval.ms时间内能完成这一批消息的处理。
max.poll.interval.ms	300000	两次消费拉取请求允许的最大时间间隔，默认为300秒，超过这个时间会认为消费者异常。

fetch.min.bytes	根据业务调整	默认为1，每次FETCH请求最少返回数据量。增加该值可以提高吞吐量，同时也会产生一定延迟。

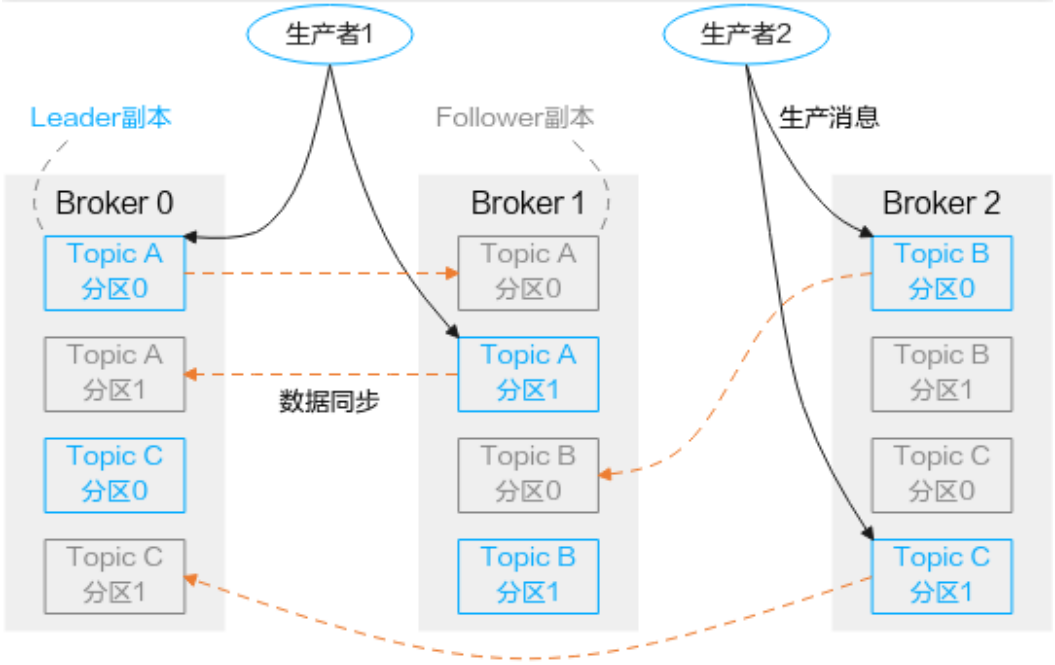
观测性能指标

Kafka提供了以下性能相关监控指标，从这些指标可以帮助分析消息堆积、分区数据倾斜、流量倾斜等问题。

指标ID	指标名称	指标说明
broker_disk_usage	磁盘容量使用率	该指标为从Kafka节点虚拟机层面采集的磁盘容量使用率。
broker_cpu_core_load	CPU核均负载	该指标为从Kafka节点虚拟机层面采集的CPU每个核的平均负载。
broker_memory_usage	内存使用率	该指标为Kafka节点虚拟机层面采集的内存使用率。
broker_cpu_usage	CPU使用率	统计Kafka节点虚拟机的CPU使用率。
group_msgs	堆积消息数	该指标用于统计Kafka实例中所有消费组中总堆积消息数。
topic_messages_remained	队列可消费消息数	该指标用于统计消费组指定队列可以消费的消息个数。
broker_messages_in_rate	每秒消息生产速率	统计Kafka节点每秒生产速率。
broker_connections	连接数	统计Kafka节点连接数。

优化数据分区

Kafka将Topic划分为多个分区，所有消息分布式存储在各个分区上。每个分区有一个或多个副本，分布在不同的Broker节点上，每个副本存储一份全量数据，副本之间的消息数据保持同步。Kafka的Topic、分区、副本和代理的关系如下图所示：



在实际业务过程中可能会遇到各节点间或分区之间业务数据不均衡的情况，业务数据不均衡会降低Kafka集群的性能，降低资源使用率。

业务数据不均衡原因

- 业务中部分Topic的流量远大于其他Topic，会导致节点间的数据不均衡。
- 生产者发送消息时指定了分区，未指定的分区没有消息，会导致分区间的数据不均衡。
- 生产者发送消息时指定了消息Key，按照对应的Key发送消息至对应的分区，会导致分区间的数据不均衡。
- 系统重新实现了分区分配策略，但策略逻辑有问题，会导致分区间的数据不均衡。
- Kafka扩容了Broker节点，新增的节点没有分配分区，会导致节点间的数据不均衡。
- 业务使用过程中随着集群状态的变化，多少会发生一些Leader副本的切换或迁移，会导致个别Broker节点上的数据更多，从而导致节点间的数据不均衡

使用数据压缩

在客户端CPU资源情况可控的情况下，使用压缩算法对数据进行压缩。

常用的压缩算法包括：ZIP，GZIP，SNAPPY，LZ4等。选择压缩算法时，需考虑数据的压缩率和压缩耗时。通常压缩率越高的算法，压缩耗时也越高。

压缩方式	压缩比	客户端CPU占用	服务端CPU占用	磁盘占用	broker带宽占用
gzip	中	中	低	中	低
lz4	中	中	中	中	中

zstd	高	中	低	低	低
snappy	低	高	高	高	高

如果追求高TPS，建议采用lz4压缩算法；如果追求较低的网络I/O或希望较低的客户端/服务端CPU占用，建议采用zstd压缩算法。这里通常推荐使用lz4压缩算法，同时不建议使用gzip算法，因为它会是一种计算敏感的压缩算法。同时针对一批数据（batch)消息压缩，更好的运用批处理可以获得更高的TPS。

4.11.2.2 RabbitMQ 性能优化

保持尽可能短的队列长度

太多的消息堆积在队列中会造成内存负载过高，为了释放内存，RabbitMQ 会把消息转存到磁盘，转存过程会耗费大量时间，造成消息处理速度下降或直接阻塞生产流程。因此队列中堆积过多的消息容易对 broker 产生负面效应。除此之外，如果节点崩溃后重启，过多的数据会使得重建索引需要消耗大量时间，集群模式下的节点间同步数据也会非常耗时。

使用惰性队列提升稳定性

惰性队列（lazy queues）是 RabbitMQ 3.6 之后新增的特性。惰性队列的消息会自动存储到磁盘，因此减少了内存的使用率，但是会增加I/O开销，影响吞吐量。使用惰性队列能够更好的把控性能，并且使得集群更加的稳定。和非惰性队列不同，消息不会积累在内存中然后等到内存不足再一次性刷到磁盘，造成队列性能不稳定。如果你需要一次发送大量消息，或者消费速度长时间赶不上生产速度，那么我们推荐使用惰性队列。 请注意，以下情况不建议使用惰性队列： a. 追求高性能 b. 队列无明显积压 c. 队列设置了 max-length 策略

通过TTL或者max-length限制队列长度

可以通过设置消息 TTL、队列 TTL 和 max-length 来限制队列长度。如果队列长度达到 max-length 值，队列头部的消息会被丢弃或进入死信队列。消息的生存时间到期也会被丢弃或者进入死信队列。

关注队列个数

在 RabbitMQ 中，一条队列是由一个线程处理的。利用服务器的多核特性和分布式特性建立多条队列，将不同队列分布到不同 CPU 或不同节点，以此来获取高吞吐量。同时需要注意，过多的队列可能会对 CPU 和内存造成较高的负担，RabbitMQ management 接口的响应速度也会受到影响。

自动为临时队列分配队列名

如果使用临时队列（包括排他队列、自动删除队列、非持久化队列），可以调用不带参数的接口queueDeclare()让 RabbitMQ 自动为你分配一个队列名。

根据需要使用自动删除队列

如果不再使用的队列资源长期保存在服务端，可能对 RabbitMQ 性能造成影响，可以通过三种方法自动地删除队列：为队列设置 TTL 属性、为队列设置 auto-delete 属性、为队列设置 exclusive 属性。

控制优先级队列的使用

每一个优先级会在Erlang VM中使用一个内部队列，这会消耗一定的资源。大多数场景下，使用最多5个优先级就够了。

如何确定消息大小

如何选择发往RabbitMQ的消息长度是一个常见问题。记住，每秒钟发送的消息数比消息大小更容易达到瓶颈。虽然发送大消息不是一个好的做法，但是发送多条小的消息也可能不是一个好的选择。更好的方法是生产者把多条小消息封装成一条大消息，然后由消费者来拆开处理。然而，如果一条大消息封装了太多的子消息，处理速度将会受到影响。如果一条子消息处理失败，整个大消息都需要重传。因此，当选择消息大小时，需要考虑带宽和业务架构。

连接和通道

每条连接（connection）大约占用 100KB 内存（启动 TLS 则会占用更多）。数以千计的连接会对 RabbitMQ 服务端造成较大的负担，极端情况下会因为 OOM 而崩溃。AMQP 协议可以在单条 TCP 连接上进行多路复用。建议一个进程只创建一条 TCP 连接，每个线程复用这条连接创建各自的通道（channel）。连接应该保持长生命周期，因为 AMQP 建立连接需要一定的开销（至少 7 个 TCP 包）。相反，通道则允许较为频繁的开启和关闭，但在发布消息时复用通道也是一个好的习惯，不要每发送一条消息都开启一个新的通道。同时需要注意如下几点：

- 多线程不要共享通道，因为你很难实现线程安全。
- 不要频繁的开启或关闭连接和通道，否则会造成更高的延迟。
- 生产者和消费者使用独立的连接，来提高吞吐量。
- 大量的连接和通道可能会影响管理接口的性能，造成请求超时。

消息确认

消费者使用确认（Acknowledgment）机制避免消息因为连接问题而丢失，客户端可以在收到消息或者处理完消息后回给服务端一个 ack 消息。消费者确认机制会对性能造成影响，如果单纯追求高吞吐量，应该关闭手动确认功能。如果消息对于业务来说比较重要，那么应该在消息被处理完之后才确认该条消息。生产者使用 Confirm 机制来实现类似的功能，当服务端收到生产者发来的消息，它会返回一个 ack 消息，以此来实现最少一次（at-least-once）的投递语义。注意，Confirm 机制同样会影响性能。

控制未确认消息个数

所有未确认的消息都会暂存在内存中，太多的未确认消息可能造成服务 OOM。为了限制未确认消息的规模，你可以在消费者端开启**prefetch**功能来限制消息拉取上限。

持久化资源

为了防止因为服务宕机、重启、硬件问题等原因造成的消息丢失，请使用持久化队列和消息。持久化消息涉及到写磁盘操作，如果使用了惰性队列也会将所有消息写入磁盘，包括非持久化消息。如果单纯追求高性能，可以使用非持久化消息。

开启TLS连接

你可以在 AMQP 之上使用 TLS 连接 RabbitMQ。因为数据需要加解密，所以 TLS 对性能有一定的影响。

如何正确选择 QoS 值

- 如果只有单个或少量消费者，并且消费速度很快，那么建议 QoS 设置的大一点，使得客户端保持忙碌状态。

- 如果客户端的消息处理速度和带宽保持不变，简单的用公式RTT / 单条消息处理时间就可以估算出应该设置多大的 QoS 值。
- 如果消费者数量较多并且消息处理速度较快，那么建议 QoS 设置的小一点。
- 如果消费者数量较多并且消息处理速度较慢，那么建议 QoS 设置为 1，让消息平均的分发到所有消费者。

请注意，如果客户端使用拉模式或者开启了自动确认（auto-ack），预拉取功能将会失效。一个常见的错误是不限制预拉取消息个数，所有消息全部发送给一个消费者，造成消费者 OOM 崩溃和消息重复投递。更多关于 RabbitMQ 预拉取功能的信息可以参考**这里**。

观测性能指标

指标ID	指标名称	指标说明
channels	通道数	该指标用于统计RabbitMQ实例中的总通道数。
queues	队列数	该指标用于统计RabbitMQ实例中的总队列数。
connections	连接数	该指标用于统计RabbitMQ实例中的总连接数。
connections_usage	连接数使用率	当前节点实际连接数占最大连接数比率。
rabbitmq_disk_usage	磁盘容量使用率	统计Rabbitmq节点虚拟机的磁盘容量使用率。
rabbitmq_cpu_usage	CPU使用率	统计Rabbitmq节点虚拟机的CPU使用率。
rabbitmq_memory_usage	内存使用率	统计Rabbitmq节点虚拟机的内存使用率。
rabbitmq_cpu_core_load	CPU核均负载	统计Rabbitmq节点虚拟机CPU每个核的平均负载。

全量指标可参考**RabbitMQ支持的监控指标**。

4.11.3 Serverless 性能优化

Serverless函数配置最佳实践

- 运行时语言

当选择编译型语言（如Java，C#等），冷启动时延一般由于首次初始化消耗比较大会导致冷启动时延偏高，但是初始化完成后每次执行的时延相较其他解释型语言（NodeJs，Python等）会有一定优势。如果流量不均衡，且对冷启动时延或者最大时延有一定要求的业务使用NodeJs, Python等运行时语言，如果流量比较均衡或者对最大时延不敏感，但是对平均时延敏感的业务选择Java，Go等编译型语言。

- 内存规格

函数Pod中分配的CPU资源与内存规格成正比，所以更大的内存规格可以获得更高的CPU资源从而提升执行性能。如果业务场景为CPU密集型或者需要大量使用内存的，建议配置更大的内存规格来获取更低的执行时延，可以通过配置不同大小内存进行性能测试，观察时延监控结合业务实际预算选择合适的内存规格。

同时如果同一个函数在不同场景下对内存和CPU资源的要求不一样，可以使用动态内存能力，参考**配置动态内存**。

- 最大实例数

如果函数执行并发量较大，那么默认的最大实例数可能不足以支撑业务调用，如果超出最大实例数对应的处理能力，可能造成执行排队时延升高，甚至出现报错。建议最大实例数和业务实际最大并发数保持一致。

- 预留实例数

预留实例是将函数实例的创建和释放交由用户管理，当您为某一函数创建了预留实例，函数工作流收到此函数的调用请求时，会优先将请求转发给您的预留实例，当请求的峰值超过预留实例处理能力时，剩余部分的请求将会转发给按量实例，由函数工作流自动为您分配执行环境。如果业务流量不均衡，存在波峰波谷情况，且对冷启动时延有要求的可以结合实际预算配置一定数量预留实例，同时，如果业务波峰波谷存在周期规律的，也可以配置预留实例的定时伸缩和智能预测弹性策略，提升资源利用率，减少资源浪费。参考**预留实例管理**。

网络配置

公网访问：函数提供了公网访问能力，但是函数提供的公网出口带宽有限，且所有租户共享，只适合对带宽、可靠性要求较低的测试业务使用。

VPC访问：函数提供了指定VPC访问的能力，但在冷启动时会初始化到该VPC网络的网络链路造成额外的冷启动时延。

如果需要访问公网，且对带宽有要求的生产业务可以通过配置绑定了NAT网关的VPC来访问公网；如果函数没有网络访问场景的，不建议配置VPC。

参考**配置网络**。

- 超时时间

如果函数配置的超时时间比较长的话，且函数代码中发生异常导致阻塞，函数同步调用会等待直到超出超时时间才返回超时异常，造成业务卡顿，长时间不退出等问题，无法实现failfast，影响业务体验。建议结合业务实际场景配置超时时间，避免超时时间配置过大。

Serverless函数代码最佳实践

如果业务可以异步实现，那么不需要关心函数的性能（除了优化成本之外）。FunctionGraph函数的性能很大程度上取决于需要FunctionGraph函数执行何种逻辑。

策略：

- 正确的使用连接池，保持连接存活并重用在上一次调用中建立的连接（HTTP，数据库、redis等）。
- 通过接口调用FunctionGraph函数时，建议客户端维护http连接池，减少http连接初始化时间。
- 避免在每次调用时重新初始化变量对象（使用全局静态变量、单例等）。

- 选择解释性语言（nodejs、python）而不是编译型语言（java、go）。
- 精简函数代码包，满足其运行时需要即可。 这将大辐减少在调用前从华为云 OBS 下载代码包所花费的时间。
- FunctionGraph函数调用华为云其他云服务资源时（例：dis、obs），如果选择对应云服务sdk，需要对sdk参数进行调优（例：超时时间、连接数、重试次数等）。
- FunctionGraph函数中访问第三方服务或华为云服务时，性能上限受第三方服务或华为云服务的性能上限制，需要确保访问的服务无性能瓶颈。

冷启动优化实践

Serverless按用付费、自动弹性伸缩、屏蔽复杂性等特征使其逐渐成为下一代云计算新范式。但是在Serverless架构带来极大便利的同时，在实时性要求较高的应用场景下，冷启动将是面临的一个挑战。当使用serverless构建 Web 服务时，冷启动和Web服务初始化时间一共超过了5秒钟，那么无疑将会使用户体验大打折扣，因此设法减少冷启动时间，提高终端用户的使用体验，是构建无服务器架构时亟待解决的问题。

Serverless实例的生命周期可以分为三个阶段：初始化阶段、执行阶段、关闭阶段。

当触发FunctionGraph时，若当前没有处于激活阶段的函数实例可供调用，则会下载函数的代码并创建一个函数的执行环境。从事件触发到新的FunctionGraph环境创建完成这个周期通常称为 冷启动时间。在serverless架构中，冷启动问题是无法避免的。

目前FunctionGraph已经对系统侧的冷启动做了大量优化，针对用户侧参考以下方案。：

- 选择合适的编程语言

目前FunctionGraph支持的编程语言（Runtime）有 .NET Core、Go、Java、Node.js、Python。在默认 512MB 内存的情况下，我们创建不同语言的函数，做一个冷启动时间横向的测试和比较。各个语言的冷启动时间为JAVA>GO>.NET>Node.js>Python。

- 选择合适的内存

在请求并发量一定的情况下，函数内存越大，分配的CPU资源相应越多，一般冷启动表现越优。

- 快照冷启动

Java应用冷启动速度慢的问题尤为突出。华为云FunctionGraph创新提出的基于进程级快照的冷启动加速解决方案，致力于在用户无感知（无需/少量进行代码适配）的前提下，帮助用户突破冷启动的性能瓶颈。本优化方案直接从应用初始化后的快照进行运行环境恢复，跳过复杂的框架、业务初始化阶段，从而显著降低 Java 应用的启动时延，实测性能提升达90%+。

观测性能指标

为了更好地支持用户使用函数，FunctionGraph提供了一系列指标供用户参考。一旦函数被调用，您可以在函数的监控页面和日志页面查看相应指标的变化，通过FunctionGraph函数的观测能力来进一步优化函数配置和函数代码。

目前，FunctionGraph提供的指标主要分为总览指标和函数指标。详细指标可参考**官方指标文档**。

4.11.4 数据库性能优化

以下章节我们结合一些具体建议和案例来说明如何针对数据库的使用进行性能优化：

1.优化数据库配置实践

数据库的配置参数应从具体业务诉求着手，根据实际需要进行设计；华为云在各个数据库云服务中均提供了默认的配置参数，以满足最普遍的业务需要。

华为云提供了多款数据库服务，不同服务的优化方式和注意事项均有差异，此方面需求，建议使用华为云提供的专业服务。

2.观测性能指标实践

性能监控有助于实时了解业务和系统的负载情况以及资源使用情况，结合告警规则的设置，云服务可自动对负载异常部分进行告警，以便更好地使用和维护云数据库系统。以GeminiDB 为例，您可以通过管理控制台，直观地查看GeminiDB Redis的各项监控指标。

3.设置数据分区实践

GaussDB数据库支持的分区表为范围分区表，列表分区表，哈希分区表。分区表和普通表相比具有如改善查询性能、增强可用性、便于维护、均衡I/O等优势。

普通表若要转成分区表，需要新建分区表，然后把普通表中的数据导入到新建的分区表中。因此在初始设计表时，请根据业务提前规划是否使用分区表。

4.GaussDB SQL语句调优实践

根据数据库的SQL执行机制以及大量的实践，总结发现：通过一定的规则调整SQL语句，在保证结果正确的基础上，能够提高SQL执行效率。如果遵守这些规则，能够大幅度提升业务查询效率，如使用union all代替union、join列增加非空过滤条件、not in转not exists等都可以提升查询速度。

5.GaussDB语句下推调优实践

目前，GaussDB优化器在分布式框架下制定语句的执行策略时，有三种执行计划方式：生成下推语句计划、生成分布式执行计划、生成发送语句的分布式执行计划。在第3种策略中，要将大量中间结果从DN发送到CN，并且要在CN运行不能下推的部分语句，会导致CN成为性能瓶颈（带宽、存储、计算等）。在进行性能调优的时候，应尽量避免只能选择第3种策略的查询语句。

执行语句不能下推是因为语句中含有**不支持下推的函数**或者**不支持下推的语法**。一般都可以通过等价改写规避执行计划不能下推的问题。

语句下推典型场景包含单表查询语句下推与多表查询语句下推，一些特殊场景如语句中带有with recursive子句，列存表等不支持下推。

6.GaussDB子查询调优实践

应用程序通过SQL语句来操作数据库时会使用大量的子查询，这种写法比直接对两个表做连接操作在结构上和思路上更清晰，尤其是在一些比较复杂的查询语句中，子查询有更完整、更独立的语义，会使SQL对业务逻辑的表达更清晰更容易理解。
GaussDB根据子查询在SQL语句中的位置把子查询分成了子查询SubQuery、子链接SubLink两种形式。

7..GaussDB算子级调优实践

一个查询语句要经过多个算子步骤才会输出最终的结果。由于各别算子耗时过长导致整体查询性能下降的情况比较常见。这些算子是整个查询的瓶颈算子。通用的优化手段是EXPLAIN ANALYZE/PERFORMANCE命令查看执行过程的瓶颈算子，然后进行针对性优化。

8.GaussDB(for MySQL)读写分离最佳实践

读写分离是指通过一个读写分离的连接地址实现读写请求的自动转发。创建实例后，您可以**开通读写分离功能**，通过GaussDB(for MySQL)的代理地址，写请求自动访问主节点，读请求按照读权重配比或者活跃连接数情况分发到各个节点。

开通读写分离时，需选择加入代理的节点（包括主节点和只读节点）。

9.GaussDB(for MySQL)持锁长事务导致后续业务报等锁超时的解决实践

由于持锁长事务长时间未提交或回滚导致后续操作阻塞，如果持锁长事务已经阻塞了后续的业务，需要将长事务KILL，后续业务侧尽量避免持锁长事务。

10.GaussDB(for MySQL)长事务产生大量临时表导致内存超限的解决实践

考虑升级实例规格，将内存利用率维持在合理范围，防止业务突增导致实例OOM，或根据业务实际情况优化慢查询。

11.GaussDB(for MySQL)联合索引设置不当导致慢SQL的解决实践

查询变慢首先确认是否由于CPU利用率达到性能瓶颈导致执行慢，考虑升级资源规格；或是库表结构设计不合理，索引缺失或索引设置不恰当，应进行语句调优。

12.GaussDB(for MySQL)使用INSTANT方式快速添加列

云数据库 GaussDB(for MySQL)兼容开源MySQL 8.0.22，支持使用ALGORITHM=INSTANT快速添加列，避免造成锁等待影响业务或者SQL执行超时无法新增成功。

13.DAS数据诊断优化性能问题实践

DBA智能运维功能基于运行数据结合算法对实例进行诊断，并对异常项提供具体的诊断结果以及优化建议。

14.观测性能指标（GaussDB为例）

指标ID	指标名称	指标说明
rds001_cpu_util	CPU使用率	该指标用于统计测量对象的CPU使用率。
rds002_mem_util	内存使用率	该指标用于统计测量对象的内存使用率。
rds003_bytes_in	数据写入量	该指标用于统计测量对象对应VM的网络发送字节数，取时间段的平均值。
rds004_bytes_out	数据传出量	该指标用于统计测量对象对应VM的网络接受字节数，取时间段的平均值。
iops_usage	IOPS使用率	当前IOPS与磁盘最大IOPS比值。

rds007_instance_disk_usage	实例数据磁盘已使用百分比	该指标用于统计测量对象的实例数据磁盘使用率，该值为实时值。
rds010_disk_usage	磁盘已使用百分比	该指标用于统计测量对象的节点数据磁盘使用率，该值为实时值。

更多指标与其他数据库指标信息可参考官方文档。

4.11.5 人工智能性能优化

1.训练优化模型性能提升实践

参数调优策略：调整模型flash attention、并行切分策略、micro batch size、重计算策略等参数。

尽可能充分利用显存和算力，通过参数调优，初步优化性能。

性能拆解

参数调优后性能仍然与转商目标有较大的差距，需要考虑进行profiling，采集性能数据后从更底层的算子、通信、调度和内存等维度将性能进行拆解分析，训练脚本中加入profiling代码。具体步骤：生成profiling数据目录结构；利用att工具，将NPU与竞品之间的数据进行端到端耗时对比分析；Tracing分析。

算子分析

通过生成profiling中的summary文件对具体的算子进行分析，考虑算子层面向FA与MM算子方向优化。

2.路由规划加速最佳实践

ranktable路由规划是一种用于分布式并行训练中的通信优化能力，在使用NPU的场景下，支持对节点之间的通信路径根据交换机实际topo做网络路由亲和规划，进而提升节点之间的通信速度。本案例介绍如何在ModelArts Lite场景下使用ranktable路由规划完成Pytorch NPU分布式训练任务，训练任务默认使用Volcano job形式下发到Lite资源池集群。详细步骤可参考**最佳实践文档**。

训练显存优化实践

pytorch的内存池基本管理策略

- pytorch的内存池以block为粒度来进行管理，block池分为小内存池与大内存池，block是pytorch向device驱动申请内存的粒度，整存整取。用户/Pytorch代码向内存池申请内存的接口归一为tensor的申请释放（这点最开始也不是很好理解，也就是说：任何一个pytorch代码申请内存的地方，均表现为一个tensor的申请释放）。
- tensor的生命周期使用类似智能指针的引用计数方式来管理，且打通了Python与C++的通道，即：一个Python的tensor对象关联一个C++的tensor对象，Python的tensor对象的消失会触发C++的tensor对象析构释放内存。一个在C++环境里创建的tensor对象可以返回成一个Python的tensor对象。
- C++的tensor对象分为两部分：一部分是viewTensor，包含tensor的各种meta信息：shape, stride, dataType等，一部分是storageTensor，包含具体的内存

addr，offset，对外呈现的是viewTensor，这是pytorch做view类操作后，多个tensor对应同一块内存的基本支撑。在storageTensor申请时，向pytorch的block池申请一块内存，找到空闲块之后，视实际要求对block进行切分使用并返回address指针。

pytorch的内存的跨流复用策略

- 如果一个stream上的内存池里申请的tensor需要给另一个stream使用，那么则需要进行recordStream操作，将这个tensor的所属block标识上新stream的信息，在这个tensor的生命周期消失触发address释放时，发现其所属block有其他stream信息，此时会给对应stream下发一个event_record task然后返回。
- 在之后本stream在每一次新申请内存时，均做一下event_query操作，如果发现event已经被record则其他stream上的task已经执行完成，此时可以放心地回收这个block。

Pytorch的内存统计信息说明

- pytorch的内存一般看三个峰值信息：allocated / active / reserved。allocated对应host上的tensor实际申请了但是没释放的内存（注意：是在host上申请释放，不代表device状态）。active对应host上还未释放的内存+还在被别的流占用的内存。
- 举例，一个tensor在streamA上申请了，让供streamB做allreduce的集合通信操作，然后tensor进过一次add后被释放，此时：释放会减去allocated值，但是不会减去active 值，直至这个allreduce真实执行之后，通过query_event查询到结果之后触发释放，才会减去active值。
- reserved对应pytorch向device申请了的内存，比如申请了100M，然后释放了，然后又申请了20M，还未释放，此时的allocated 为20M，reserved为100M。真实网络里，reserved里存在大量block被切小了的可使用内存但是当申请一个大块内存时又无法复用，这种会导致reserved 与 allocated 的较大差值，通常称为内存碎片。

pytorch的内存碎片影响因素

- 一个step里更多的内存申请释放内存次数理论上一定会导致更多的内存碎片，为什么说是一个step呢？因为pytorch内存池只取决于host上的训练脚本逻辑，而每个step的训练脚本逻辑是相同的，所以一般第一个step之后内存状态可以稳定下来。
- 不同生命周期的tensor交替地申请释放，因为pytorch向驱动申请是整存整取，所以：一个常规的优秀做法是把长生命周期放在最开始申请，这样不易形成碎片。而workspace内存由于可以绝对意义上地串行复用，因此对此单独做一个定制的内存池策略可以减少对内存碎片影响，在NPU上常见的非连续转连续操作，就是一个相对GPU来说较多的内存申请。

显存优化策略

由于大模型的参数成倍数的增长，远超出了单GPU物理显存所能承载的范围，大模型训练必然需要进行显存优化。显存优化要么是优化算法本身，降低模型算法的显存消耗；要么是去扩大显存，通过一些置换方式获得“额外“空间，由于显存物理大小一定，我们获得额外空间的方式不外乎两种：时间换空间和空间转移。其中，时间换空间通常会消耗算力、带宽；空间转移主要是消耗I/O带宽，有一定的时延，可能会降低吞吐。

观测性能指标

指标ID	指标名称	指标说明
cpu_usage	CPU使用率	该指标用于统计ModelArts用户服务的CPU使用率。
mem_usage	内存使用率	该指标用于统计ModelArts用户服务的内存使用率。
gpu_util	GPU使用率	该指标用于统计ModelArts用户服务的GPU使用情况。
gpu_mem_usage	GPU显存使用率	该指标用于统计ModelArts用户服务的GPU显存使用情况。
npu_util	NPU使用率	该指标用于统计ModelArts用户服务的NPU使用情况。
npu_mem_usage	NPU显存使用率	该指标用于统计ModelArts用户服务的NPU显存使用情况。
disk_read_rate	磁盘读取速率	统计ModelArts用户服务的磁盘读取速率。
disk_write_rate	磁盘写入速率	统计ModelArts用户服务的磁盘写入速率。

全量指标可参考**ModelArts支持的监控指标**文档。

4.11.6 大数据性能优化

4.11.6.1 HIVE 优化

概述

Hive架构

Hive提供了Hadoop的SQL能力，主要参考标准的SQL，Hive进行了部分的修改，形成了自己的特有的SQL语法HQL（Hive SQL），更加适合于Hadoop的分布式体系，该SQL目前是Hadoop体系的事实标准。

Hive调优

用户输入HQL，Hive将HQL进行词法解析，语法解析，之后生成执行计划，并对执行计划进行优化，最后提交任务给YARN去执行。所以Hive的调优分为以下几个部分：

- 接入层：主要包括用户的连接性能，如网络速度、认证、连接并发数。
- HiveServer：以SQL的优化为主，执行计划是SQL优化的主要手段，通过接口查看Hive对整个SQL语句是如何进行任务的分解和编排，并结合MapReduce/Spark的执行情况针对性的进行任务的优化。
- HiveMetaStore：因为Hive的MetaStore可能是外部的独立数据库，所以它的性能也会影响到整个HiveServer的性能，主要包括HiveMetaStore访问时间，访问次数，连接并发数。
- MapReduce/Spark：以该组件进行执行时，MapReduce/Spark执行的情况直接引影响到Hive的性能，如每个任务的大小，任务与资源分配均匀度，任务拆分合理度等。
- HDFS：最底层的IO读也是性能的关键，主要考虑的指标是读取和写入的性能，还包括块大小合理设置等。

其中MapReduce/Spark/HDFS组件有自己独立的调优手册及文档，请参考对应组件的调优。本文档重点讨论上述的1，2，3部分的性能调优的内容，并结合MapReduce/Spark的进行调优说明。

批处理业务

批处理主要特点是耗时时间长，消耗的资源比较多，主要的调优和设计推荐如下：

- 尽量使用ORC File， 配上合适的压缩算法， 主要可选的压缩算法为Zlib和Snappy。其中Zlib压缩比高，但压缩解压时间比Snappy长，消耗资源比如Snappy多。Snappy平衡了的压缩比和压缩解压的性能。推荐使用Snappy。
- 尽量使用Map Join减少Shuffle的次数，大幅提升性能
- 不同SQL语句，完成同一个功能，生成Map Reduce的数量越少越好
- Hive系统默认是典型的配置场景，结合业务实际情况，可以做一些参数的调整，如文件块的大小，Map个数与Reduce的个数，压缩算法等。
- 合理的使用分区，分区数量不要太多，查询的SQL尽量指定具体的分区值；

具体请参考第5章节**11.5 性能调优常用方法**。

衡量指标

衡量指标主要用于查看相应的指标来发现Hive服务或执行过程中的一些问题，尽快能定位Hive的性能问题。通常我们查看指标的顺序应该是通用指标，接入层指标，HiveMetaStore，HiveServer相关指标，其它相关组件的指标（如MapReduce/Spark/HDFS）。下面列举目前可查看到的相关指标信息：

- 通用指标

主要是指通用的服务器的相关性能指标：CPU使用率，内存占用量，磁盘IO读写速度，使用Core数量等，通过这些指标可以衡量任务在该类型机器或该机器上的执行情况，观察集群各机器的通用指标，可以看到集群的负载是否均衡。

- 接入层指标

Hive连接数，并行SQL数量，输入缓存值（或每批大小）。单HiveServer实例可以处理的最大并发数可以通过参数控制，默认是500，该参数主要受JVM内存和CPU的处理能力的限制。

- HiveMetaStore

HiveMetaStore连接数，并行SQL数量，语句执行计划。

- HiveServer

主要查看SQL语句的执行计划，进行SQL相关的调优，并结合MapReduce/Spark引擎的相关参数，主要是Job数量，Map数量，Reduce数量。

- 通用测试标准

Hive性能上业界主要是拿TPC-DS来跟同类型的产品或者自己的老版本进行对比。标准测试仅做为性能测试的一些参考。

指标观测方法

- 通用指标的观测

集群机器的CPU，内存，IO的使用情况可以通过Manager的主机管理界面查看到所有Host的资源使用情况。

- 接入层指标的观测

Manger的服务->Hive服务状态页面可以查看到相关的HiveServer的连接数，HQL的执行成功的统计信息。

- HiveMetaStore指标的观测

在Manager的服务->Hive服务状态页面，查看HiveMetaStore当前的请求连接数量以及关键API性能。

- HiveServer相关指标的观测

这里主要以SQL调优为主，参考**11.4.2 Hive的HQL调优**。

4.11.6.2 Spark 性能优化

概述

Spark是基于内存的分布式计算框架。在迭代计算的场景下，数据处理过程中的数据可以存储在内存中，提供了比MapReduce高10到100倍的计算能力。Spark可以使用HDFS作为底层存储，使用户能够快速地从MapReduce切换到Spark计算平台上去。Spark提供一站式数据分析能力，包括小批量流式处理、离线批处理、SQL查询、数据挖掘等，用户可以在同一个应用中无缝结合使用这些能力。

Spark的特点如下：

- 通过分布式内存计算和DAG（无回路有向图）执行引擎提升数据处理能力，比MapReduce性能高10倍到100倍。
- 提供多种语言开发接口（Scala/Java/Python），并且提供几十种高度抽象算子，可以很方便构建分布式的数据处理应用。
- 结合SQL、Streaming、MLlib、GraphX等形成数据处理栈，提供一站式数据处理能力。
- 完美契合Hadoop生态环境，Spark应用可以运行在Standalone、Mesos或者YARN上，能够接入HDFS、HBase、Hive等多种数据源，支持MapReduce程序平滑转接。

集群服务部署规划

服务规模与业务容量参数配置对照表

- Spark作为内存计算引擎，需要更多的内存和CPU。用户在规划规格时，应根据当前的业务容量和增长速度，规划合理的内存和CPU资源，特别需要关注以下几点：
- 当程序运行在yarn-client模式下时，需要关注在driver端汇聚的数据量大小，根据自己的业务场景，为driver设置合理的内存。
- 根据自己的业务目标，规划CPU资源和内存资源。规划时，需要结合当前的数据分布情况，业务复杂度，设置“executor-memory”，“executor-cores”，“Executor-num”，并在此基础上，规划需要的CPU核数和内存大小。
- 在规划内存时，要预留一定量的内存空间作为操作系统的buffer cache，一般预留20%。
- 从HDFS中读入数据时，要考虑block解压缩后的数据膨胀。
- 规划一定的磁盘作为缓存空间，包括缓存数据、日志、Shuffle数据。

调优原则

- 提高cpu使用率同时减少额外性能开销。
- 提高内存使用率。
- 优化业务逻辑，减少计算量和IO操作。

典型业务的调优

- 优化代码逻辑：在进行Spark参数调优之前，要进行相应的规划设计，优化代码逻辑。
- Spark任务跑的比较慢，cpu利用率低：检测室executor线程不能全部吃满，此时应减少每个executor的core数量，增加executor个数，同事增加partition个数。
- 任务容易出现内存溢出：部分数据分片较大，单个task处理数据过大，或者executor中并行度不足，单个task内存不足导致。此时应减少executor数量，增大数据分片。
- 数据量少，但小文件数量多：减少数据分片，在reduce算子后执行coalesce算子，以减少task数量，减少cpu负载。
- 使用spark sql查找一个大表，表列数较多，但是查找的列较少：尽量使用rcfile或parquet格式，减少文件读取成本，同时选择合适的压缩格式，减少内存负载。

指标观测方法

性能衡量指标包括吞吐量、资源利用率、伸缩性。

- 吞吐量：在相同资源环境下，执行相同计算任务，查看任务的完成速度
- 资源利用率：执行计算任务，查看在不同负载情况下，cpu、内存、网络的使用率。
- 伸缩性：

– 横向扩容带来的性能提升曲线：增加资源，执行相同计算任务，查看性能提升比率。

– 增加系统负担带来的性能下降曲线：在相同资源环境下，增加计算负载，查看性能下降比率

4.11.6.3 Flink 性能优化

概述

Flink是一个批处理和流处理结合的统一计算框架，其核心是一个提供了数据分发以及并行化计算的流数据处理引擎。它的最大亮点是流处理，是业界最顶级的开源流处理引擎。Flink最适合的应用场景是低时延的数据处理（Data Processing）场景：高并发pipeline处理数据，时延毫秒级，且兼具可靠性。

集群服务部署架构

服务规模与业务容量参数配置

Flink作为流数据处理引擎，依赖内存和CPU。用户在规划规格时，应根据当前的业务容量和增长速度，规划合理的内存和CPU资源，特别需要关注以下几点：

- 根据自己的业务目标，规划CPU资源和内存资源。规划时，需要结合当前的数据分布情况，业务复杂度，设置JobManager的内存，TaskManager的数量，TaskManager的内存，每个TaskManager的slot数量，规划适当的CPU核数和内存大小。
- 在规划内存时，要预留一定量的内存空间作为操作系统的buffer cache，一般预留20%。
- 从HDFS中读入数据时，要考虑block解压缩后的数据膨胀。
- 规划一定的磁盘作为缓存空间，包括缓存数据与日志。

调优目标

Flink调优的目标是在不影响其他业务正常运行的前提下，高效的完成业务目标，通常为了达成该目标，一般需要最大限度利用集群的物理资源，如CPU、内存、磁盘IO，使其某一项达到瓶颈。

调优原则

- 提高CPU使用率同时减少额外性能开销。
- 提高内存使用率。
- 优化业务逻辑，减少计算量和IO操作。

性能调优常用方法-DataStream调优

- 配置内存：调整老年代和新生代的比值；开发Flink应用程序时，优化datastream的数据分区活分组操作。
- 设置并行度：用户可以根据实际的内存，CPU，数据以及应用程序逻辑的情况调整并行度参数。任务的并行度可以按优先级从高到低排列，由算子层次、执行环境层次、客户端层次、系统层次这四种层次指定。
- 配置进程参数：配置JobManager内存、TaskManager个数、TaskManager Slot数、TaskManager内存。
- 设计分区方法：可设置随机分区、rebalancing（round-robin partitioning，基于round-rebin对元素进行分区，使得每个分区负责均衡）、rescaling（以round-robin的形式将元素分区到下游操作的子集中）、广播分区（广播每个元素到所有分区）、自定义分区。
- 配置netty网络通信：可在客户端的“conf/flink-conf.yaml”配置文件中进行修改适配。

指标观测方法

性能衡量指标包含吞吐量、资源利用率、伸缩性。

- 吞吐量：在相同资源环境下，执行相同计算任务，查看任务的完成速度。
- 资源利用率：执行计算任务，查看在不同负载情况下，CPU、内存、网络的使用率。
- 伸缩性：

– 横向扩容带来的性能提升曲线：增加资源，执行相同计算任务，查看性能提升比率。

– 增加系统负担带来的性能下降曲线：在相同资源环境下，增加计算负载，查看性能下降比率。

5 成本优化支柱

5.1 成本优化支柱简介

成本优化支柱专注于帮助企业高效地使用云服务来构建工作负载，面向工作负载的整个生命周期不断完善和改进，减少不必要的开支并提升运营效率，让云上应用始终最具成本效益。

成本优化实践不意味着只有降本，它是安全合规、韧性等维度的平衡，也是达成业务目标的最优投入。

华为公司结合云业务成本运营经验和业界最佳实践总结并提炼出体系化实践与建议，包括：提升成本管理效率、合理选择与分配云资源、建立预算管理机制、持续成本治理、提升资源效率及架构优化等。

5.2 基础概念

基本概念

名称	名词解释
FinOps	FinOps 是 Finance 和 DevOps 的合成词，强调 IT、财务和业务团队必须协作，将财务责任引入云，并在速度、成本和性能之间做权衡时做出数据驱动的明智决策。
CFM	华为云云财务管理（Cloud Financial Management），参考FinOps流程实践，E2E构建云财务管理能力，旨在帮助客户提高云支出的透明度和可预测性，以更加准确、高效的方式分配、控制和优化云成本。

5.3 设计原则

组织，流程和成本管理相匹配

在成本优化过程中，一个很重要的原则是需要将组织结构，流程和成本管理相匹配。需要建立“责权分明”的体系，否则即使用再好的成本优化工具，也无法将成本优化落到实处。

流程上，需要把成本管理作为各个上云流程中必备的一环； 组织上，需要投入适当的时间，资源和人力用于建立云财务管理的能力。 例如，在云账号申请和云资源申请的时候，就需要建立完善的流程，以便于将组织，项目和其所使用的云账号，云资源进行关联，并确保每个部门或团队都对其使用的资源负责。而在最终的企业财务层面，引入云财务管理的能力，也有助于理解企业的每个子部门的产出和成本的关系，甚至于清晰地了解企业每挣一块钱，最终的云成本是多少，这样企业不至于为了完成本优化，而牺牲自身真正的产出。

在最终实际的优化实施阶段，由于成本优化往往需要运营，运维，研发等多个部门的参与，也涉及到平台部门和业务部门的合作，故而需要确定一个各个组织成员都参与的完善流程，如释放空闲资源的流程。

企业也可以定期生成报告，并同步给干系人；同时联席例会，如组织多角色参与的例会(如月度例会)，审视预算执行情况、讨论风险应对策略、总结优化经验和计划下一步重点工作等；

事前规划，做好成本模型，预算规划和成本预测

理解每个组织，项目的成本并非易事，尤其是很多云资源是事实上的跨组织和项目的公共资源，故而，在一开始的时候，就需要建立一个基础的，得到管理团队认可的成本模型，用于在以后的成本优化执行过程中确保整体的成本可追溯性。

在确定成本模型之后，则需要设立一个预算规划，同时在各种基于云的项目一开始的时候，就引入该预算规划和成本管理，同时设定不同层次的预算开销的阈值，用于成本的预警 。

最后，对于使用了一段时间华为云的账号，华为云也提供了成本预测工具，基于用户的实际上云成本，提供了未来的预测，善用这些工具，云用户可以根据预算情况，设立基于预测的告警。

持续监控，了解账单和资源使用情况

成本优化不是一个“一锤子买卖”，随着业务的发展，云资源的申请和释放是不停变化的，使用云的软件/服务的架构也是随时在演进。 企业应建立定期的云成本监控和审查机制，并根据实际情况调整和完善云成本优化策略。比如一个快速增长的业务组织更多地可能会偏向于提升业务的速度，而设计一个比较宽松的云成本优化策略，而稳定的业务组织则可以以成本效率为主要考量，设计比较严格的云成本优化策略。 企业还可以借助华为云成本中心提供的云成本管理工具和平台来实现自动化的成本监控和优化。

节省和优化，使用不同的计费模式，资源优化和架构优化

云支出的主要影响因素是费率和用量，结合云化业务模型和成本数据分析，可以使用不同的优化措施。从费率上，云服务存在按需、包年包月、资源包、竞价实例等多种计费模式，不同的计费模式有着不同的适用场景。企业可以根据自己的需要，合理选

择各种计费模式来适配不同的业务形态和降低费率，实现成本节省。从用量上，企业可以通过资源优化，释放闲置资源，降配等降低资源用量，或者通过离在线混部，弹性缩扩容等架构方案实现资源复用和闲时释放，也达到了资源用量的节省。费用优化、资源优化和业务架构优化代价逐步增高，对业务的影响也依次增大。企业可以根据业务目标、对业务影响、优化代价和收益的评估，确定优化目标和优化措施优先级。确定优化措施。

5.4 问题和检查项

在企业进行成本优化的过程中，推荐使用如下问题寻找自身可以改进的点，并参考检查项/最佳实践进行改进，以下所有的检查项，也是最佳实践建议，将在下一章节进行详细描述。

问题	检查项/最佳实践
COST01 您是否按照成本优化的需求，规划了相应的组织机构和流程?	1. 规划企业组织，将组织结构，流程和成本管理相匹配 2. 规划IT治理体系，提高管理效率 3. 明确团队责任，建立和维护成本意识文化 4. 指定云资源管理策略和相应的权限管理机制
COST02 您是否有预算规划管理机制?	1.建立云预算与预测流程 2.精细化预算管理和跟踪
COST03 您是否将成本分配到组织单元?	1. 制定成本分摊原则 2. 可视化成本分摊结果 3. 公共成本分配
COST04 您是否有持续的成本治理?	1. 建立规范，持续提升成本分配比例 2. 主动监控成本
COST05 您是否为优化指定了策略和目标?	1. 分析业务趋势和优化收益 2. 建立可以量化的优化目标 3. 定期回顾和审核
COST06 您是否使用考虑了不同的计费模式优化成本?	1. 了解云上不同计费模式的特点 2. 为工作负载选择合适的计费模式 3. 跟踪并监控权益商品的使用情况
COST07 您是否管理了和优化了资源使用情况?	1. 持续监控资源利用率指标 2. 释放闲置资源 3. 考虑不同的云资源技术选型 4. 降配低负载资源或升配高负载资源
COST08 您是否考虑了架构优化?	1. 按地域规划应用架构 2. 云原生架构改造 3. 存算分离 4. Serverless探索

5.5 COST01 规划成本优化相应的组织机构和流程

5.5.1 COST01-01 规划企业组织，将组织结构，流程和成本管理相匹配

- **风险等级**

  高

- **关键策略**

  在成本优化过程中，一个很重要的原则是需要将组织结构，流程和成本管理相匹配。需要建立“责权分明”的体系，否则即使用再好的成本优化工具，也无法将成本优化落到实处。一个比较好的实践是在初始的时候，创建一个团队（云业务办公室、云卓越中心或 FinOps 团队），负责在整个组织内建立并维护成本意识。成本优化的负责人可以是了解整个组织和云财务的个人或团队。而整个团队的成员需要包含相关决策部门和实施部门的人员，典型的团队成员通常包括来自企业的核心决策者（CXO），财务、开发，运维/运营，数据分析团队的人员。

  这个团队可以是一个虚拟团队，该团队在企业开始实施成本优化的时候，建立相应的成本管理流程，例如，将成本支出纳入应用、业务全生命周期的关键评估指标，推动各个分歧（如成本归属）的解决。 最终定义企业中成本相关完善的问责机制，落实机制。

5.5.2 COST01-02 规划 IT 治理体系，提高管理效率

- **风险等级**

  高

- **关键策略**

  实施与您的组织对应的IT治理结构。这有助于在整个组织内分摊和管理成本。随着经营范围和规模的不断扩张，不断建立子公司、分公司，大部门也逐步拆分成多个小部门，组织结构的层级也就越来越多。企业的IT治理架构也会受到组织结构的影响，需要匹配企业管理模型，帮助企业以多层级组织的方式管理人、财、物，所有资源都可以找到责任团队。企业根据组织结构合理规划IT治理架构后，可将成本分配到业务团队，让各业务团队为使用的云服务成本负责。

- **相关服务和工具**

  对于大型企业或集团公司，推荐优先使用**企业组织+多账号**的方式，通过账号隔离资源和成本，方便业务快速拓展。

  中小型企业以及单账号客户，可以使用**企业项目**来映射组织。如果存在更多维度、更细粒度规划的诉求，可以使用**标签**作为组织规划的补充。比如用标签来区分资源归属的产品团队、应用和负责人。

  企业组织采用企业主子多账号形式进行成本管理时，企业主子关系可以分为财务托管和财务独立两种。

  - 财务托管模式下，企业主账号可以统一管理企业主+企业子的成本，包括统一成本分析、统一预算跟踪、统一异常监控、统一成本建议等，使用该模式可大幅提升管理效率。
  - 财务独立模式下，企业主子各自管理各自账号下的成本，包括成本分析、预算跟踪、异常监控、成本建议等。如果企业子向企业主授权了查看消费信息，则企业主也可以统一分析企业主+企业子的成本。

5.5.3 COST01-03 明确团队责任，建立和维护成本意识文化

- **风险等级**

  中

- **关键策略**

  成本优化的流程中落实成本意识、都需要明确团队责任。一种比较好的实践是使用一组明确定义的 KPI 指标，提供团队级别的报告，实现成本透明度和成本问责制，这些指标可以包含收益/成本比率，单位商品成本，核心资源利用率等等。

  值得注意的是，成本优化不是一锤子买卖，团队对责任的接受，实施包括指标自身的完善都需要一个过程。 KPI指标应该随着阶段的演变而演变，以建立不断成功，不断进步的心态，而不是一次性推动成熟。在实施过程中，更多应该将错误视为学习和改进流程的机会，这将减少不成熟的团队和管理团队对成本优化的恐惧心理。指标自身也需要不断优化，一开始可以从比较保守的目标开始推动，避免IT成本迅速下降造成的业务风险。而后不断迭代预算分配、IT 支出和预测，不断优化这些指标。

  最后，需要在流程中提高成本意识，在员工培训中贯彻成本意识。

5.5.4 COST01-04 指定云资源管理策略和相应的权限管理机制

- **风险等级**

  高

- **关键策略**

  由于成本优化是跨组织多个业务部门的事项，而云资源是云上成本的主要开销，故而应该制定策略，确定您的组织应该如何管理资源。如上文所说的，可以使用账号隔离不同组织/部门的资源，甚至于在同一个组织/部门内部，开发，测试，核心业务，非核心业务，也使用不同的账号和环境。

  然而即使账号/环境是分散的，云资源管理策略和权限管理机制应该是集中的。 企业的中心团队，如上文所提的云业务办公室、云卓越中心或 FinOps 团队需要为各个账号环境实施与策略一致的组和角色，控制每个组中谁可以创建、修改或停用实例和资源。同时依据企业的业务环境，创建统一的资源/成本视图，统一管理企业的账单和成本。

- **相关服务和工具**

  客户可通过**统一身份认证服务IAM**的细粒度权限管理，精细化控制账号下用户的资源访问权限，实施最小授权。

  对于多账号场景，客户可通过**Organization**的服务控制策略（ Service Control Policy ），集中控制每个账号可执行的操作。

5.6 COST02 实施预算规划管理机制

5.6.1 COST02-01 建立云预算与预测流程

- **风险等级**

  高

- **关键策略**

  由于云资源天然的易申请，易缩扩容的特性，使用云可以提高效率、创新速度和灵活性，与此同时，也导致了云成本和使用模式的高度可变，客户应调整现有的组织预算和预测流程，以适应云的变化。

客户应密切关注历史消费趋势和不断变化的业务趋势，力求尽可能准确的预算规划。同时结合基于趋势（以历史支出作为输入）的预测和基于业务驱动因素（例如新业务上云或区域扩张）的预测，可以有效改进并提升企业的财务预测准确率。

- **相关服务和工具**

  使用成本中心的成本分析，可以根据客户的历史支出预测未来时间范围的成本。成本分析的**成本和使用量预测**，会参考不同的计费模式特征，结合机器学习和基于规则的模型来分别预测所有消费模式的成本和使用量。

  使用成本分析确定基于趋势的预测之后，您还可以利用华为云的**价格计算器**，根据新业务上云或区域扩展所需的产品和使用量，自主搭配产品进行未来成本的估算。

5.6.2 COST02-02 精细化预算管理和跟踪

- **风险等级**

  高

- **关键策略**

  针对企业不同项目/业务/应用，应该建立预算管理机制，精细化管理每个项目/业务/应用全生命周期的云开销。

  企业的项目/业务是随时间变化而变化的，一般而言，新兴业务/项目常有更多云资源扩容的需求，而稳定的业务/项目则可以更多考虑单位收益的云成本是否可以持续优化，而处于生命周期末尾的项目/业务则需要考虑逐步释放不再需要的资源。

  企业制定预算时，应该伴随项目生命周期的有效跟踪，建立定期审查机制，同时根据业务情况动态调整，不断改进预算规划来更好控制成本，以确保其符合企业的实际情况。

  企业应该建立紧急响应和应对计划，设计应对成本超支情况的紧急响应计划。包括明确的流程和责任人，确定接收警报并负责采取行动的责任人或团队。这些人员可能是财务团队、运维团队、部门负责人或特定的成本管理团队。以便在警报触发时能够快速采取必要的措施，如优化资源、停止不必要的服务，或者针对某个部门，项目进行新购买云资源的限制等。

- **相关服务和工具**

  华为云提供了通用的**预算管理**工具，您可以根据企业实际规划的预算，用预算管理工具跟踪起来，并可以设置细粒度的过滤条件，精细化跟踪具体产品、团队、项目的成本。

  除了在成本中心查看预算进展外，您还可以为指定预算设置预算提醒，当实际使用或预测使用达到提醒阈值时，及时接收系统发出的短信或邮件预警，从而及时采取下一步措施。

  您还可以设置**预算报告**，定期将指定预算的执行情况以日报、周报、月报的形式周知给企业内部相关角色，比如业务部门、财务、CTO，达到成本透明的目的。

5.7 COST03 对成本进行分配

5.7.1 COST03-01 制定成本分摊原则

- **风险等级**

  高

- **关键策略**

  成本分配支撑企业将成本分配到各业务团队中，使得各业务团队的成本清晰可见。这也是上文中明确的团队责任的基础。 根据清晰的成本，业务部门可准确定价，并平衡成本、稳定性和性能，经济高效的提供领先方案。企业管理者基于数据决策各业务的云开支，保障核心业务和战略业务方向的支出，不超支，不浪费。

  成本分配需匹配业务实质，具体有以下几个原则：

  - 按实际使用者进行分配。即谁使用产生的成本分配给谁，而不是谁购买分配给谁。
  - 基于实际消耗进行分配。比如客户1月份购买了一个包年资源，365元，按照实际支出这笔成本分配在1月份；如果按照实际消耗，那么就会在整个订购周期进行分配，每天分配1元。这种成本分配机制，更体现了成本责任制。
  - 公共成本也应该在组织内进行分摊。比如企业内的共享资源、平台服务、支持计划、未及时标记的成本。只有将公共成本也分配下去，才能让业务团队关注这部分消费，从而合理化使用，减少不必要的浪费。

- **相关服务和工具**

  华为云成本中心提供包年包月、资源包成本按实际使用者和实际消耗的**成本分摊**（即摊销成本）。

5.7.2 COST03-02 可视化成本分摊结果

- **风险等级**

  中

- **关键策略**

  将成本分配到责任团队，使各责任团队及时准确清晰的了解自己业务成本，加强团队成本意识，在构建方案时追求性能、可靠性和成本的平衡。

- **相关服务和工具**

  华为云支持您按照组织规划的方式分配成本。随着云服务的使用，规划的组织方式也会随着云服务的费用生成，体现在**账单管理**、**成本分析**等可视化工具的数据中。

  您还可以使用**成本单元**，综合多种条件（产品类型、账单类型、关联账号、企业项目、成本标签），自定义规则，将成本按照业务语义分配到有意义的分组。基于成本单元的成本分配结果，支持在“成本分析”页面可视。

  如果需要获取明细数据与自身云管平台进行集成，实现定制化的成本和使用分析，您还可以**订阅账单明细数据**、**OBS转储成本明细**或调用**客户运营能力API**。比如将成本和使用明细与企业的业务运营数据结合，生成业务单位成本KPI。

5.7.3 COST03-03 公共成本分配

- **风险等级**

  中

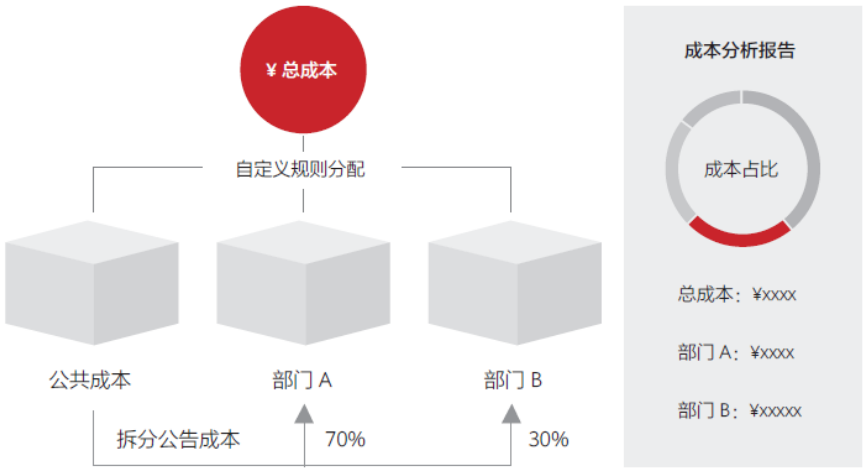
- **关键策略**

  跨团队共享使用的CDN、直播带宽应按照各业务团队的实际带宽占比，将带宽费用拆分到不同的业务团队。

  跨团队共享使用的CCE集群服务，应按照各团队分配和使用的CPU/内存等比例，将容器集群成本（包含CCE、ECS、EVS等服务成本）拆分到各个业务团队。

  以上公共成本，以及其他共享资源&平台服务&服务支持&未及时标记产生的未分配成本，也可以按照一定的比例规则，比如平均分配、按消费比例分配、按约定

比例分配等规则，拆分到各个业务部门，从而满足各团队或业务部门公平分配公共成本的需求



- **相关服务和工具**

  华为云成本中心的**成本单元**提供按比例的公共成本分拆方式。

  华为云成本中心提供**共同成本分拆**，支持CDN、Live按照域名流量进行成本分拆。

  华为云**CCE服务**提供细化的按照Pod Level的成本分拆，并可以卷积到Workload，Service等各种标准K8S模型层级。

5.8 COST04 持续进行成本治理

5.8.1 COST04-01 建立规范，持续提升成本分配比例

- **风险等级**

  中

- **关键策略**

  成本是否准确有效的分配，是后续进行成本监控和优化的基础。客户应关注并提升成本分配比例，奠定成本治理的基础。

  标签作为一种常见的成本分配方式，可以灵活匹配组织内多种分配场景（比如产品、应用、责任人），但在实施标签过程中，企业会发现有各种不利因素导致标签的标记覆盖率下降，例如：

  实施标签工作量大：云上创建的资源不断增加，资源数量巨大，且每个资源需打多个标签

  标签实施不一致：业务部门执行进展不一、添加的标签key&value错误、部分资源无人认领未打标签

  环境和诉求变化：部门&应用调整、管理层&财务等利益相关人诉求变化

  针对这些不利因素，需要引入运营，定期审视标签覆盖率，并通过治理持续提升。一些可选的方法有：

  a. **制定KPI指标支撑改进**: 成本标签核心目的是成本分配，建议将KPI定为成本可分配比例，用于衡量标签的覆盖率。可分配成本比例越高，成本分配和报告

效率越高，成本数据越可信任。在标签治理过程中，通过可分配成本比例趋势的上升和下降，检查组织内标签的标记覆盖率是在提升还是在下降

b. **识别标签缺失和错误**：在确定需要进行标签治理后，需要首先识别所有未打标签的资源和标签key&value错误的资源，然后从费用最高的资源开始逐步治理。建议利用云厂商提供的工具或者自建工具，通过自动化规则的方式，在资源创建的时候，就判断标签是否规范。另外一个更好的方式通过权限管理，识别资源创建人和组织，自动为资源打上标签。

c. **定期审查和优化规范：**变化不可避免，良好的标签管理不是一个一劳永逸的过程。通过定期审查和优化规范，确保成本标签适应环境和诉求变化。管理层&财务等利益相关人诉求变化，他们可能会对更细粒度的提出请求，定期和利益相关人确定并更新规范。

- **相关服务和工具**

  企业可在**成本中心**查看可分配成本比例，并通过该指标诊断标签覆盖率和牵引企业内部治理标签。

  企业可通过**成本中心**、**TMS**、云服务控制台来识别和治理未打标签资源，标签Key&value错误

  客户可通过Config服务预设的**资源合规策略**，识别资源标签为空等不合规场景。

  客户可通过Organization服务，设置**标签策略**，帮助您在组织账号中对资源添加的标签进行标准化管理。

5.8.2 COST04-02 主动监控成本

- **风险等级**

  中

- **关键策略**

  不要只在出账后或收到异常通知时再查看成本和用量，应使用工具定期检查成本。定期监控和主动分析成本，有助于您及时识别成本趋势，避免异常发生。

- **相关服务和工具**

  创建**预算提醒**，将预算设置为提醒阈值，在预测或实际成本超出预算时，及时获取超预算通知，防止潜在成本超支。

  创建**成本监控**，华为云成本中心的**成本监控**引入机器学习，对客户历史消费数据进行建模，对于不符合历史数据模型的成本增长，识别为异常成本记录，同时提供异常增长的Top潜在原因。客户可设置监控提醒，定期获取影响成本高的异常记录提醒，进而快速做出反应，维持预期的成本支出。

  在费用中心设置**可用额度监控**，在可用额度余额低于阈值时预警，避免客户额度耗尽，业务中断。

  使用**资源包监控**，在资源包剩余不足预警，避免资源包用尽自动转为按需计费。

  使用成本分析**预置报告**或创建常用的成本分析报告，定期快速了解成本分布和趋势。

5.9 COST05 优化指定策略和目标

5.9.1 COST05-01 分析业务趋势和优化收益

- **风险等级**

  高

- **关键策略**

  云成本是一个综合工程，也是一个定期审核、回顾和执行的流程，除了考虑优化带来的收益以外，还需要考虑相关成本，例如，因为优化带来的人员和时间成本。

  为了降低整体成本，优化的工作量必须与潜在的节省额成比例。优化可以从应用占成本的比例考虑。 例如，与占总成本 5% 的应用相比，应更经常、更彻底地审核占总成本 50% 的应用。优化时要考虑的另一个因素是实施更改的工作量。如果测试和验证变更的成本很高，优化的频率应该降低。您应该反方向考虑是否可以通过替身自动化测试和验证能力，从而进一步降低人力成本。

  此外，由于成本优化带来可能带来的资源冗余度的下降，故而也应该综合考虑业务的趋势。 比如一个快速增长的业务组织更多地可能会偏向于提升业务的速度，而设计一个比较宽松的云成本优化策略，而稳定的业务组织则可以以成本效率为主要考量，设计比较严格的云成本优化策略。如果应用服务于特定的地理位置或市场领域，并且您已预测出该领域会出现的对云业务使用量的降低，则提高优化频率可能会更有利于节省成本。

5.9.2 COST05-02 建立可以量化的优化目标

- **风险等级**

  高

- **关键策略**

  成本优化是一项投资，而且是一个需要持续进行的流程。为了向公司或者组织的决策者、利益相关方说明投资的价值，就需要对成本优化自身，尤其是其执行的目标进行量化。从而在持续的优化活动中，都可以从决策者或者利益相关者那里得到支持，并获得一个固化的流程框架来衡量成本优化活动的成果。

  简单的成本优化量化目标/成果就是报告成本节省优化的费用， 例如，您可以建立报告，在不牺牲质量或产出的情况下，给公司或者组织带来多少成本的节省。

  此外优化的量化目标也可以包含效率的提升，例如，从传统IT架构向容器化，Serverless迈进的过程中，您不只是提升了资源利用率，同时也可以是提升了业务开发，部署的速度，从而提升了业务对市场的响应时间，以及人员的效率。这部分也应该列入量化的内容。

  最后，优化的目标是使企业或者组织每一块钱的花费都能产生最大的效益。不能只专注于降低成本而忽略业务价值。设定一个明确的可量化的优化目标，有助于成本优化团队（ 上文中提到的云业务办公室、云卓越中心或 FinOps 团队 ）和决策层，利益相关方取得一致。

5.9.3 COST05-03 定期回顾和审核

- **风险等级**

  高

- **关键策略**

  为了让云上应用始终最具成本效益，推荐您定期对其进行回顾和审核，以了解是否有机会实施新的优化措施。

  回顾和审核可以基于成本分配的原则，在应用级别执行，持续审核组织为每个云上应用付出的总体成本。通过综合考虑云资源成本，研发成本，运营管理成本（ 如托管服务 vs 非托管云服务 ）来计算总拥有成本。审核工作量应该体现可能带来的好处（ 例如分析时间与应用成本成正比 ）以及相应的成本是否带来正向的营收。

回顾和审核的频率应该综合考虑多种因素，包括成本优化在企业或者组织中的重要性，测试和验证成本，应用的复杂性和优化变更的难易程度。同时，在每次回顾和审核时，持续改进流程，例如，通过降低测试和变更的成本从而提升整体的优化频率。最后，在云厂商新的服务、资源类型和配置推出后，也可以启动流程，对它们进行评估，以优化您的工作负载成本。

5.10 COST06 使用不同计费模式优化成本

5.10.1 COST06-01 了解云上不同计费模式的特点

- **风险等级**

  高

- **关键策略**

  云服务存在按需、包年包月、资源包、竞价实例等多种计费模式，不同的计费模式有着不同的适用场景。企业或者组织需要根据自己的需要，了解不同计费模式的特点，合理选择各种计费模式来适配不同的业务形态和降低费率，实现成本节省。

  - 按需计费：适用于临时、突发的业务场景；
  - 包年包月：通过预付一定周期的资源使用费用，来获取优惠的计费模式。一般适用于资源长期使用，业务较稳定的场景；
  - 资源包：一种特殊的包年包月，可通过预付一定周期下某种资源使用量的费用，来获取优惠的计费模式。资源包可以抵扣多个资源的用量，适用于长期使用且用量比较稳定的场景；
  - 竞价计费：适应于业务稳定性不高，中断也不影响业务的场景。

5.10.2 COST06-02 为工作负载选择合适的计费模式

- **风险等级**

  中

- **关键策略**

  分析工作负载的每个组件。确定组件和资源是长时间运行（应享受承诺折扣，包年包月或购买资源包），还是短时间动态运行（采用 Spot 或按需定价）。使用成本管理工具中的建议对工作负载执行分析，并对这些建议应用业务规则以实现高回报。

- **相关服务和工具**

  为提高成本效率，华为云根据您过去的使用情况，为您提供多项计费模式的优化建议，帮助您在不改变资源性能的情况下，通过调整计费模式来节省成本。您可以重点关注高节省低风险的节省建议（“预计月度节省”高且“盈亏平衡时间”短）

  按需转包年包月成本优化评估：自动识别客户长期按需使用的资源（比如ECS、EVS、RDS、ELB、SFS Turbo），生成按需转包年包月的优化建议和节省评估。

  资源包购买建议：自动分析客户按需资源消耗和华为云在售资源包商品（比如OBS标准存储单AZ存储包、OBS标准存储多AZ存储包、OBS公网流出流量包、SFS存储资源包）的覆盖情况，生成相应的资源包购买建议和节省评估。

5.10.3 COST06-03 跟踪并监控权益商品的使用情况

- **风险等级**

  低

- **关键策略**

  客户购买资源包等权益商品时，应定时跟踪资源包的使用情况，若资源包到期或用尽应及时续购，资源包覆盖不足应及时增购，资源包使用过少则应在资源包到期后续购合适大小的资源包，避免浪费。

- **相关服务和工具**

  华为云成本中心提供**资源包的使用率/覆盖率分析**，您可以通过该工具了解已购资源包的使用率和覆盖率情况，识别资源包购买过多（使用率低），还是过少（覆盖率低），从而优化下一阶段的购买。

  华为云费用中心提供**资源包剩余使用量预警**功能，您可以根据实际需要，按照剩余使用量百分比、绝对值或自定义方式来设置阈值，及时获取提醒。

5.11 COST07 管理和优化资源

5.11.1 COST07-01 持续监控资源利用率指标

- **风险等级**

  高

- **关键策略**

  持续地在组织中定义资源的核心利用率指标（如CPU利用率，内存，CDN服务的流量，数据库的TPS），按（天、周、月）等时间周期发现规律，对低利用率资源的应用/项目进行审查。

5.11.2 COST07-02 释放闲置资源

- **风险等级**

  中

- **关键策略**

  持续监控资源的闲置情况（如ELB无流量，EVS盘无挂载，EIP没有绑定到虚机），释放资源，或者监控资源使用只是在某个固定的时间(如每天的十二点，每个周末），可以使用自动化的方式定期申请资源，使用后释放

- **相关服务和工具**

  华为云**优化顾问**，提供成本维度的巡检，识别ECS、EIP、EVS、ELB等闲置资源。

  华为云**成本中心**，除识别ECS、EIP、EVS、ELB等闲置资源外，还基于历史消费提供节省评估。您可参考系统给出的利用率信息、预估月度节省，结合业务团队意见，采取资源优化行动。

5.11.3 COST07-03 考虑不同的云资源技术选型

- **风险等级**

  中

- **关键策略**

定期咨询专家或 华为 合作伙伴，以便确定哪些服务和功能的成本更低。查看华为博客和其他信息源。如在非计算密集型场景，使用华为云的云耀系列服务器取代普通ECS服务器

5.11.4 COST07-04 合理降配低负载资源或升配高负载资源

- **风险等级**

  中

- **关键策略**

  根据工作负载和资源特征选择合适的资源大小或类型。您可通过持续监控资源利用率，发现资源的利用率低于/高于阈值，选择降配或者升配资源来优化成本。

5.12 COST08 进行架构优化

5.12.1 COST08-01 按地域规划应用架构

- **风险等级**

  中

- **关键策略**

  国家已启动“东数西算”工程，将东部发达地区的数据，传输到西部算力资源丰富的地区进行运算、存储。西部数据中心综合成本有明显优势，低PUE低能耗，如贵阳资源价格比广州上海等区域低10%左右。企业可将灾备、离线分析、转码、运维等对网络要求低的系统部署在贵阳、乌兰察布，降低资源成本。

  可以关注华为云新推出的云区域以及相关的服务，考虑多Region部署方案。

- **相关服务和工具**

  布局优化可以参考华为云不同Region的算力价格，尤其乌兰察布和贵阳等Region

5.12.2 COST08-02 云原生架构改造

- **风险等级**

  中

- **关键策略**

  基于云原生架构改造，主要是应用容器化和微服务化的改造，从而发挥云原生的优势，如：自动弹性扩缩容等，容器技术可以提高资源利用率，避免闲置资源，从而降低计算成本，应用微服务化可以降低运维复杂度，从而降低运维成本。

  广告电商等在线作业服务SLA要求较高，高峰时段明显，使用资源存在潮汐现象；大数据/转码等离线作业容错性高，计算需求大，可容忍较高的时延。为了保证在线业务的性能和稳定性，通常按波峰时需要申请资源，这样在非波峰时段就有资源浪费。将在线离线业务混合部署，可有效提升整体利用率。

5.12.3 COST08-03 存算分离

- **风险等级**

  中

- **关键策略**

  传统大数据方案计算和存储融合部署，扩容磁盘时必须扩容计算节点，在实际使用时产生浪费。存算分离是一种数据处理技术，它将数据存储和数据处理（计

算）分开，使得存储和计算可以独立地进行优化和扩展，这种技术提高数据处理的效率、降低成本并满足大规模数据存储和分析的需求。

如某导购网站日志分析业务，存储经常扩容，计算需求没有明显增长，计算资源浪费；某互联网客户推荐业务，存储容量缓慢线性增加，计算突发需求大，峰值计算资源消耗是低谷时几十倍，无法弹性使用计算资源。使用对象存储代替HDFS/本地盘，计算存储分离，多种计算组件独立部署，计算和存储各自按需使用，避免绑定性浪费，结构化降本30%。

5.12.4 COST08-04 Serverless 探索

- **风险等级**

  低

- **关键策略**

  Serverless是下一代云原生范式，无服务计算带来简化的开发运维、更少的资源成本，Serverless架构最大限度计算、存储、网络等资源，提升整体资源利用率、缩短需求发布周期，提高应用的研发效率。

  用户可以通过云监控服务监控Serverless实例的CPU使用率、内存使用率，当满足一定条件，自动触发Serverless算力扩容和缩容，从而提供资源使用率，降低成本。

- **相关服务和工具**

  华为云以Serverless形态存在的产品，存储类的对象存储服务（Object Storage Service，OBS），应用类的云应用引擎（Cloud Application Engine， CAE），容器类的云容器实例（Cloud Container Instance，CCI）以及计算类的函数工作流（FunctionGraph）

5.13 成本优化云服务介绍

- **成本中心**是华为云免费向用户提供的云财务管理服务，可帮助您收集华为云成本和使用量的相关信息、探索和分析华为云成本使用情况、监控和跟踪华为云成本，及时了解云支出的趋势和动因，减少异常支出，持续成本优化。
- **费用中心**为您提供财务信息、发票、合同、续费、退订和变更等服务，有助于更好的了解您的消费信息。同时费用中心还提供余额预警、资源包预警等服务，可以帮助您及时了解支出情况， 管控支出。
- **企业中心**面向大型企业，提供多账号环境下的财务管理服务，帮助企业以多层级组织的方式管理人、财、物，满足企业IT治理诉求。支持财务托管和财务独立两种多账号财务关联模式。理
- **企业项目管理 EPS**为客户提供单账号下的人财物权管理，用户可以根据组织架构规划企业项目，将企业分布在不同区域的资源按照企业项目进行统一管理，同时可以为每个企业项目设置拥有不同权限的用户组和用户。
- **标签管理TMS**是一种快速便捷将标签集中管理的可视化服务，提供跨区域、跨服务的集中标签管理和资源分类功能。
- **优化顾问**结合华为云最佳实践与用户的配置和使用情况进行分析，为客户提供包括可靠性、安全、性能、成本等维度的自助检查与优化建议，从而帮助客户实现高效运营与成本节约。

6 卓越运营支柱

6.1 卓越运营支柱简介

在华为公司，卓越运营代表着质量、效率和可持续的卓越客户体验。它帮助改进设计、开发、测试、部署、发布和运维活动，持续实现高质量的交付结果，推动了持续集成和持续交付（CI/CD）落地；同时助力打造确定性运维体系，让研发团队将更多时间用在构建让客户受益的新功能上，减少用于维护和处理突发事件的时间，从而带来运行良好的系统和平衡的工作负载，尤其是卓越的客户体验。卓越运营支柱融合了这些优秀实践，聚焦如何正确地构建软件，高效地运维软件，持续提供卓越的客户体验，包含：组织团队、设计工作负载、大规模运营工作负载和随时间变化改进工作负载的最佳实践。

6.2 基础概念

名称	名词解释
确定性运维	确定性运维旨在构建可防、可控、可治的运维管理体系。首先是通过高质量的产品开发，严谨的运维流程和制度来降低故障的概率，要挑战零故障，同时也要有技术手段对可能发生的故障，将间隔、影响范围及故障恢复时间做到可防、可控、可治，要把数字化带来的“不确定性”通过运维变成“确定性”。
IaC 基础设施即代码	基础设施即代码（IaC）是指使用代码而不是手动流程和设置来配置和支持基础设施的能力。任何应用程序环境都需要许多基础设施组件，例如操作系统、数据库连接和存储。 开发人员必须定期设置、更新和维护基础设施，以开发、测试和部署应用程序。 手动管理基础设施既耗时又容易出错，尤其是在大规模管理应用程序时。

名称	名词解释
CI/CD 持续集成/持续交付	持续集成是一种编码理念和一套实践，它促使开发团队频繁地实施小的代码更改并将其签入版本控制存储库。大多数现代应用程序都需要使用各种平台和工具来开发代码，因此团队需要一种一致的机制来集成和验证更改。持续集成建立了一种自动化的方式来构建、打包和测试他们的应用程序。拥有一致的集成流程可以鼓励开发人员更频繁地提交代码更改，从而实现更好的协作和代码质量。 持续交付从持续集成结束的地方开始，并自动将应用程序交付到选定的环境，包括生产、开发和测试环境。持续交付是一种将代码更改推送到这些环境的自动化方式。
Telemetering 遥测	遥测是对被测量对象的参数进行远距离测量的一种技术。是将对象参数的近距离测量值传输至远距离的测量站来实现远距离测量的技术，并把测得结果传送到接收地点进行记录、显示和处理的活动。
CMDB	配置管理数据库（configuration management database）简称CMDB，是信息技术基础架构库（ITIL）用语，是组织用来储存软体硬体资产（常称为形态项目，CI）资讯的数据库。用CMDB来追踪资产（例如产品、系统、软体、设备、人员）的状态，例如这些资产在特定的时间点是否存在，以及各资产之间的关系，并通过公开的接口支持IT管理各种业务数据消费。
MTTR	MTTR（Mean Time to Repair）平均恢复时长，平均修复时间指从故障发生到验证确认故障恢复的耗时。MTTR 分为三个维度：MTTI（Mean Time To Identify）平均发现时长、MTTK（Mean Time to Know）平均诊断时长、MTTF（Mean Time to Fix）平均修复时长
变更风险控制	在变更作业过程中，建立事前检查、事中拦截和事后验证的能力，防止异常行为。
安全生产	安全生产目的是为了持续保障现网“安全、稳定、高质量”，从人员、工具、产品能力、流程规范等方面在安全预防、过程监控、结果稽查等维度进行端到端管理，减少或防止现网故障的发生，其中如何防止异常行为导致的事件是安全生产的重要目标。
故障快速恢复	故障快恢是以故障模式库为基础，建立应急预案，提升故障恢复效率、降低故障恢复时长，结合混沌工程演练把不确定的恢复时长做到确定的。
资源生命周期管理	指的资源的申请、创建、交付、运维以及最终的销毁释放过程。
故障演练	故障演练指通过沉淀通用的故障场景和可控成本在线上故障重放，以持续性的演练和回归方式的运营来暴露问题，不断验证和推动系统、工具、流程、人员能力的提升，从而提前发现并修复可避免的重大问题，或通过验证故障发现手段、故障修复能力来达到缩短故障修复时长的作用。
运维托管	运维托管服务是一种针对企业或组织的IT基础设施进行全面管理和维护的专业服务，旨在提高IT系统的可用性、可靠性和安全性。该服务涵盖了多个方面，包括系统监控、故障排除、系统优化、安全防护等。

6.3 设计原则

建立持续改进的团队文化和标准化运维体系

在卓越运营中，团队文化建设至关重要。运营是一门不断改进的艺术。只有不断从已有事故中学习经验，持续学习和改进，才能最终达到卓越运营。故而，团队应该培养持续学习和改进的文化，此外，在事故发生时，应该以对事不对人的态度，思考系统的改进，而不是惩罚或者指责个人。片面指责个人或者直接处罚的做法很容易引起一系列后果，例如后续的运维团队成员由于担心处罚，会隐瞒事故或者隐藏真正的事故原因，导致后续无法切实改进系统的运维能力和流程。一线的工程师也因为担心处罚，不愿意担责，不愿意做任何系统的改变。部门、组织之间由于担心事故影响部门、组织内部成员，导致了消极应对系统中隐藏的问题或者将问题推给了其他组织，部门。最终，这种文化上的高压导致整个组织和运维流程的僵化，以及系统不能持续迭代更新之后的代码、架构腐化，最终导致无法运维的系统。 故而，文化上，惩前毖后，应重在总结经验，明确改进责任主体组织，不责怪个人。

在总结经验上，应该将相关经验进行标准化的沉淀，即将经验总结成自动化工具，流程以及建立相应的组织体系，我们称之为标准化运维体系。非标是大规模运维的头号天敌，主要表现是运维无序，团队成员依靠自身技术各自为战，处于被动响应和疲于应付的工作状态，效率低下，人为失误多，故障处理难度大。标准化运维体系是对有效经验总结后，运维活动例行化的高效管理。通过对运维活动的标准化、流程化和工具化管理，实现从无序向有序演进，达到运维操作团队运作“最佳秩序”，简化运维交付工作，降低技能依赖，提高运维效率，降低运作成本。

通过 CI/CD 实现高效的频繁可逆的小规模变更

在软件开发过程中，应该尽量使需求分析，设计，开发，测试，部署的开发周期较小，使用频繁的小型迭代进行。一个典型的实践是使用微服务和CI/CD实践，微服务架构是一种更为灵活、可扩展和易于维护的架构风格，已经逐渐成为现代应用开发的主流选择。它通过将应用程序拆分为小的、自治的服务，每个服务都负责执行特定的业务功能，可以使用不同的技术栈，由独立的团队开发，测试，部署和扩展，并通过轻量级通信机制相互交互。而在CI/CD下，同一团队以流水线的方式集成整个微服务的开发，测试和进行不同地域的部署、发布和运维。

对于已经采用DevOps模式的组织，应该更进一步，不仅在软件项目的管理，而是从运维角度来看，小型频发的迭代有助于快速发现问题，一旦发现问题，也易于回滚到软件的上一版本，并降低部署失败时发生大规模问题的风险。

X 即代码，尽量自动化所有流程

云上应用和传统应用的一大区别是，您可以将整个云上应用，包含应用程序自身、运行应用的云基础设施、安全策略、以及相应的运维操作视为代码。这意味着整个卓越运营的各种实践，都可以极大地使用代码自动化，例如定义应用的基础设施，部署应用自身，修改应用的配置，设置安全策略，甚至日常的运维操作，故障修复，都可以通过代码实现并执行。

自动化是沉淀运维经验，建立标准运维最重要的一环，通过自动化，可以避免人为错误，标准化流程并提高效率。 即使在部分自动化流程中依然需要人工干预，例如决策点。在决策点前的自动化流程依然可以确认人员权限，向人员提供必要的上下文和信息，以便做出明智的决策，比之纯手工流程，最大程度避免了错误。

通过可观测性进行持续改进

可观测性是指通过观察系统的外部输出，推断其内部状态的能力。一般来说，云上应用的可观测性通常使用三种核心手段，一：指标，指标是系统状态的定量度量，例如TPS、请求延迟、调用量等。二: 日志：日志是系统事件的人可读记录，例如应用程序的运行信息，运行错误、安全事件等。三：跟踪（Trace)，跟踪可以追踪单个请求或事务在系统中的路径，帮助我们了解系统的执行情况。

对于构建在云上的应用，通过可观测性，可以快速发现和解决系统故障，从而提高系统从故障中的恢复速度。进一步地，可以提前发现系统的问题，例如性能，容量瓶颈，提前解决问题。更进一步地，您可以通过联动可观测性带来的告警和上文中的自动化流程，通过主动式响应，包括动态缩扩容，流控，主动切流，节点的迁移等，消灭问题于无形之间。

6.4 问题和检查项

在迈向卓越运营的过程中，推荐使用如下问题寻找自身可以改进的点，并参考检查项/最佳实践进行改进，以下所有的检查项，也是最佳实践建议，将在下一章节进行详细描述。

问题	检查项/最佳实践
OPS01 您是否已经建立持续改进的团队文化和标准化运维体系?	1. 建立持续学习和改进的文化 2. 规划标准化的运维组织 3. 规划标准化的运维流程与运维工具
OPS02 您是否通过CI/CD实现高效的频繁可逆的小规模变更?	1. 进行需求管理与迭代开发 2. 关联源代码版本和部署的应用版本，使用代码质量最佳实践
OPS03 你是否有完备的测试验证体系?	1. 推行开发者测试 2. 使用多个环境进行集成测试，构建和生产环境相同的预生产环境 3. 性能压测 4. 生产环境拨测 5. 混沌测试和演练
OPS04 自动化构建和部署流程是否完备?	1. 有效落地持续集成 2. 采用持续部署模型 3. 基础设施即代码 4. 自动化工程运维任务
OPS05 是否有运维准备和变更管理体系?	1. 进行生产准备度评审 2. 进行变更风控 3. 定义变更流程

OPS06 是否建立了完备的可观测体系?	1.建立可观测体系 2.定义可观测对象 3.制定和实施可观测性指标 4. 规范化应用日志 5. 实施依赖项遥测 6. 实施分布式跟踪 7. 通过可观测性指标引入自动化措施
OPS07 是否进行故障分析与管理?	1. 创建可操作的告警 2. 创新监控看板 3. 支持事件管理 4. 支持故障恢复流程
OPS08 是否有运营状态度量和持续改进机制?	1. 使用度量指标衡量运营目标 2. 进行事故复盘和改进 3. 知识管理

6.5 OPS01 建立持续改进的团队文化和标准化的运维体系

6.5.1 OPS01-01 建立持续学习和改进的文化

- **风险等级**

  高

- **关键策略**

  由于系统的独特性和复杂性，没有放之四海皆准的方案，为了达到卓越运营，需要不断改进这些最佳实践，并建立自己的最佳实践。所以，在所有最佳实践的第一条，就是在您的团队中培养持续学习和改进的文化。

  而持续学习和改进需要鼓励团队沟通和共享，例如，在您公司/组织中总结的最佳实践应该得到广泛地传播，对已有事故的分析，应该得到记录，确保相关根因都得到充分理解，尤其重要的是制定有效的标准化流程/自动化工具来降低事故再次发生的可能性和影响，这些流程和自动化工具，也需要广而告之，以向团队解释清楚缘由。

6.5.2 OPS01-02 规划标准化的运维组织

- **风险等级**

  高

- **关键策略**

  承载卓越运营，应该建立适应您实际的运维组织。运维组织的团队之间具有明确的流程，规定了团队之间的协作方式，例如规定不同团队的响应时间、服务级别目标（ SLO) 或服务等级协议(SLA ），同时应该记录团队间沟通信息，确保有足够的数据用于后续的改进。

  例如一种运维组织设计是：将运维组织分为一线、二线和三线阶梯型运维支持团队，一线受理客户的服务请求，第一时间将大部分的服务请求闭环。二线处理一

线升级的服务请求和监控发现的客户的问题，按照SLA完成闭环，涉及到软件版本缺陷类问题升级到三线进行解决，大部分时间处理告警、事件和故障的恢复，其余时间开展转维验收、应急预案与演练等主动运维活动，对现网的稳定性和可用性负责。三线聚焦解决软件版本缺陷问题。

此外也可以使用DevOps模式，由开发工程师直接运维系统，而保留一个小而精干的卓越运营使能团队，用于负责组织整体的卓越运营流程改进和相应的流程工具落地。

无论如何设立组织，应该确保具有一个整体的流程，在流程中的每个团队和成员都有自己明确的责任。 同时可以使用明确的方式（如收集运营/运维数据）分析团队工作对业务成果的影响，从而可以在实际工作中确定不同任务的优先级，并适时改进。

6.5.3 OPS01-03 规划标准化的运维流程和运维工具

- **风险等级**

  高

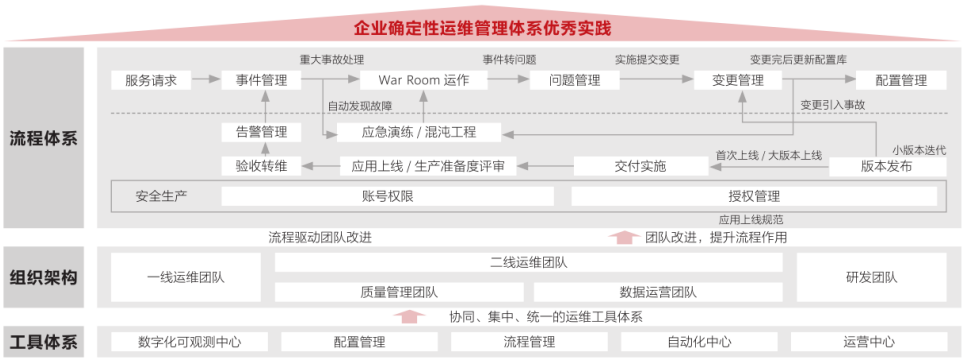
- **关键策略**

  流程和工具是经验的承载，通过标准化的流程，可以大幅降低在运维过程中因为个人的因素受到的影响和无序化。 通过标准化的、统一的运维工具，向运维人员提供集中、统一维护界面及清晰易上手的操作手册，方便运维人员的集中维护，提高运维效率。常见的运维流程有：

  - **变更管理流程**：适用于生产环境软、硬件的变更活动管理，减少变更导致服务意外中断或服务质量下降，确保企业的环境安全、稳定地运行，并最大化的提升系统的可用性，满足所承诺的服务水平。
  - **告警和事件管理流程**：适用于开发，生产环境故障等事件的受理、处理、升级流程，确保用户的业务及时得到响应和处理，支撑SLA的达成，需要明确定义企业各类事件的等级，以及处理的职责，规范各类事件响应和处理时限及通报机制，保障业务的安全性和稳定性。
  - **问题和回溯流程**：适用于事件复盘分析，识别故障的根因、管理规避方案和已知错误，来降低故障再次发生的可能性和影响。通过有效的问题管理运作，促进产品质量的不断完善，提升产品的质量稳定性，降低产品现网故障数量。
  - **产品可用度评审流程（Product Readiness Review）**：对于您云上业务是否在产品环境有问题的审查，以确定产品/应用已做好产品发布准备，在运维阶段是否有问题。 值得注意的是，由于云上应用迭代更新的特性，产品可用度评审不应该只是在产品刚上线时进行审查，而以后则置之不理。由于您的云上应用不断更新，这个流程应该定期/或者由重大事件（比如电商企业的促销）触发。

  此外还有类似于企业IT服务的管理，账号的管理等流程，围绕这些流程，您的企业可以使用并标准化一系列云上工具，如流水线，监控报警，日志处理，运维中心。从而将您企业的运维标准化，进而迈向卓越。上文中的一些关键流程的最佳实践（变更管理，告警和事件处理，问题和回溯流程，运维可用度评审流程）也会在本白皮书的其他最近章节详述。

- **设计建议：**



- **相关云服务和工具**
  云运维中心 COC
  华为云AOM服务
  云监控服务 CES
  华为云LTS服务
  应用性能管理 APM

6.6 OPS02 通过 CI/CD 实现高效的频繁可逆的小规模变更

6.6.1 OPS02-01 进行需求管理和迭代开发

- **风险等级**
  高
- **关键策略**
  您的云上应用要达到卓越运营，从设计和开发阶段就需要保证可用性，可恢复性，同时也需要保证代码的质量。您需要评估和了解软件DFX相关要求，包括可靠性、性能、可服务性、可运维性、可交付性等要求 将监管、行业和内部合规性要求纳入需求范围中，同时在需求排序的时候，给予这些需求足够的时间和重视。
  同时从可维护性来看，较之于一次性颠覆性的大范围应用/软件更新，小步快跑，持续迭代地进行云上软件的更新更有利于运维，因为一则小范围的云上软件更新和部署更不容易引起大范围事故，其次，不停地迭代更新也有效地保证了开发，运维团队成员能够时刻处于练兵状态，不至于对运维的流程，最佳实践比较陌生。要保证云上应用进行迭代更新，那么从需求阶段，就要进行迭代规划和跟踪，通过迭代的方式进行开发管理，根据需求划分迭代计划。
- **相关云服务和工具**
  华为云CodeArts Req服务

6.6.2 OPS02-02 关联源代码版本和部署的应用版本，使用代码质量最佳实践

- **风险等级**
  高
- **关键策略**

在代码开发阶段，需要开展代码协作设计和管理。使用现代化的代码仓管理代码，确保代码合并后，代码将保持一致，并且不会丢失任何更改。通过正确的版本控制，同时，现代化的代码仓可以方便设置代码版本，关联源代码版本和部署的应用版本，在运维阶段，一旦部署在云上的应用发生任何问题，可以方便回溯到源代码，而且方便使用上一版本的源代码回滚到上一版本的应用。

其次，在软件开发生命周期内，推动开发人员采用代码质量最佳实践，例如，使用代码审查或结对编程等最佳实践来提高代码质量，确保每行代码在合入代码仓时，都有两个以上的工程师审查过，同时，通过设置代码合入策略进行代码控制，确保代码审查规范的执行。最后，建议通过自动化代码检查策略进行代码问题检查。

- **相关云服务和工具**

  华为云CodeArts Check服务

  华为云CodeArts Artifact服务

  华为云CodeArts Repo服务

6.7 OPS03 完备的测试验证体系

6.7.1 OPS03-01 推行开发者测试

- **风险等级**

  高

- **关键策略**

  开发者测试是现代软件工程中非常重要的一环，一般而言，开发者的测试代码可以在本地，或者构建阶段反复多次执行，依赖低，也是在软件系统运维之前成本最低的发现软件问题的方式，尤其是各种异常场景或者用户输入，开发者测试的过程实际上“强制”了开发者去思考线上业务可能出现的场景，从而有利于减轻后续运维阶段系统的负担。

  此外，云上的软件是不断演进和重构的，很多时候我们不敢修改已有系统代码的原因，就是不知道它的影响范围，担心产生某种程度上的蝴蝶效应，影响了其它模块而造成线上系统的问题，有了开发者测试之后，只要在改完代码后运行一下测试就知道改动对整个系统的影响了，从而可以让我们放心的重构和演进代码。

  同时，应该有一个适用于您软件的开发者测试标准，如代码覆盖率和分支覆盖率。

6.7.2 OPS03-02 使用多个环境进行集成测试，构建和生产环境相同的预生产环境

- **风险等级**

  高

- **关键策略**

  开发者测试虽然成本低，但是缺乏对生产环境配置以及不同服务和应用之间实际交互的验证。为此，您的组织可以在云上提供多个环境，典型的环境包含测试环境，预生产环境和生产环境。在生产环境部署之前，可以通过测试环境进行联调测试，验证不同团队代码之间的业务交互流程是否正确。但是测试环境和生产环境的配置不尽相同。 而预生产环境使用与生产环境相同的部署配置、安全控制、步骤和程序，在预生产环境中测试发布过程。验证所有部署步骤是否按预期完

成，如检查依数据、配置和服务。通过集成功能测试，和各种非功能测试以及运行状况检查等各种监控方法，进一步测试所有更改。

6.7.3 OPS03-03 进行性能压测

- **风险等级**

  高

- **关键策略**

  性能压测主要通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。通常把性能测试、负载测试、压力测试等统称为性能压测。广义而言，是为保证系统运行后的性能可以满足用户需求，而开展的一系列测试组织工作。

  在应用系统上线发布之前，通过性能压测，测试应用系统能承受的最大并发、响应速度、以及稳定性是否满足设计要求。同时通过压测合理配置基础设施资源，提高资源利用率。性能压测主要目的是检查各项指标是否符合系统的业务要求，主要的测试分类包括：

  负载测试：是指在一定的软件、硬件及网络环境下，运行一种或多种业务，在不同虚拟用户数量的情况下，测试环境的性能指标是否在用户的要求范围内，以此确定系统所能承载的最大用户数、最大有效用户数以及不同用户数下系统响应时间及硬件设备或云服务的资源利用率，负载测试强调的是在一定的环境下系统能够达到的峰值指标。

  压力测试：指在一定的软件、硬件及网络环境下，模拟大量的虚拟用户向测试环境产生负载，使测试环境处于极限状态下并长时间连续运行，以测试硬件设备或云服务在高负载情况下是否能够稳定工作。压力测试强调在极端情况下系统的稳定性。

  容量测试：指在一定的软件、硬件及网络环境下，构造不同数量级别的测试数据及记录，运行一种或多种业务，在一定虚拟用户数量的情况下，获取不同数量级别的硬件设备或云服务性能指标，以确定业务系统的最佳容量和最大容量。

  并发测试：测试多个用户同时访问同一个应用、同一个模块或者数据记录时是否存在死锁或者其他性能问题，所以几乎所有的性能测试都会涉及一些并发测试。因为并发测试对时间的要求比较苛刻，通常并发用户的模拟都是借助于工具，采用多线程或多进程方式来模拟多个虚拟用户的并发性操作。

  配置测试：指在一定的软件、硬件及网络环境下，模拟一定数量的虚拟用户运行一种或多种业务，将测试结果作为基线数据，在系统调优或系统评测的过程中，通过运行相同的业务场景比较测试结果，确定调优的结果是否达到预期效果或者为系统的选择提供决策数据。

  在性能压测过程中，需要模拟或者还原现实业务场景进行测试，这就必须借助特定测试工具达到相应的要求。不同类别的性能压测工具适用场景与测试能力各不相同，有的基于静态与动态资源测试能力，有的具备加压与负载测试能力，有的针对端到端业务请求与响应具备计时与计量能力，有的针对平台网站具有Web应用、移动应用和API测试能力，所以性能压测工具是要依据业务测试场景来选择。

- **相关云服务和工具**

  参考华为云CodeArts PerfTest工具

6.7.4 OPS03-04 对生产环境进行拨测

- **风险等级**

  高

- **关键策略**

  拨测是利用软件系统以外，甚至现有账号或云Region外的系统，以系统用户使用场景为视角，模拟用户使用场景的测试。和普通的云拨测可实现对网络质量、页面性能、端口性能、文件传输、音视频体验等场景进行周期性监控，支持多维度分析性能指标。利用可视化性能数据及时对业务质量作出反应，保证业务稳定正常运行。

6.7.5 OPS03-05 进行混沌测试和演练

混沌工程（Chaos Engineering）是通过故障注入，验证故障快速恢复能力及系统可靠性的实践活动。

- **风险等级**

  高

- **关键策略**

  通过混沌工程的方法模拟可能出现的故障，进而综合验证系统在不同故障场景下的容错能力、监控能力、应急响应能力、定界定位、快速恢复等确定性恢复能力。

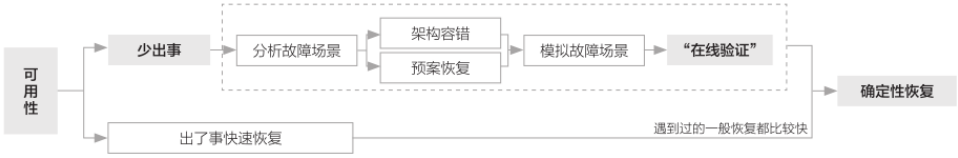
  验证高可用设计：业务系统在规划设计阶段进行架构高可用设计、监控设计，在上线前进行生产准备度评审 (PRR)、性能压测，确保系统能够持续提供稳定、可靠的服务。混沌工程从应用部署架构、服务容量、监控告警、应用高可用等多维度设计演练场景，先测试、后攻防、再突袭逐步递进式的开展演练。通过持续演练，对架构高可用、监控、PRR 等能力做“在线验证”，实现持续性的动态风险治理。混沌演练和高可用设计共同成为系统稳定性的“双引擎”。

  系统风险消减、业务快速恢复：分析系统潜在风险（故障场景），制定应急预案，验证故障场景的覆盖率和命中率，验证应急预案的质量和执

  行效率，做到“少出事”和“出了事快速恢复”，实现确定性恢复的目的。

  少出事：尽量挖掘潜在风险，区分等级和危害，通过执行演练检验业务风险消减能力。

  出了事快速恢复：通过主动制造故障，让运维和研发熟悉故障场景，验证应急恢复预案，从而加快恢复速度。



  混沌工程度量指标

  - 故障场景的覆盖率：分析故障场景的覆盖率，例如容灾场景覆盖 80%，过载场景覆盖 60%。
  - 故障场景的命中率：分析故障场景中，真实发生的比率。
  - 应急预案的质量：用于度量应急预案有效性和执行效率。
  - 风险发现个数与等级：定期评估分析（季度或年度）主动发现的风险数量和级别。
  - 风险消减个数、等级与类型：风险降级的数量，风险消减的数量，增加预案的数量，改进监控项的数量。
  - 故障恢复时长提升率：对应故障场景经过混沌工程演练，平均恢复速度提升的比率。

- 故障数量相比上年减少数量：本年度故障数量相比上年度减少多少。

- **相关云服务和工具**

  MAS 混沌工程

  COC 故障演练

6.8 OPS04 自动化构建和部署流程

6.8.1 OPS04-01 有效落地持续集成

- **风险等级**

  高

- **关键策略**

  持续集成是一种软件开发实践，开发人员使用它定期将软件更新集成到源代码控制系统中。当工程师向代码仓提交代码时，持续集成过程就开始了。理想情况下，集成过程会根据多个基线和测试来验证代码。然后，它向提交者提供有关这些测试状态的反馈。如果基线检查和测试进展顺利，集成过程将生成并暂存部署更新软件的资产。这些资产包括编译的代码和容器映像。

  持续集成可以通过执行以下操作更快地交付高质量的软件：

  - 针对代码运行自动化测试，以便尽早检测到重大更改。
  - 运行代码分析以确保代码标准、质量和配置。
  - 运行合规性和安全检查以确保软件不存在已知漏洞。
  - 运行验收或功能测试以确保软件按预期运行。
  - 对检测到的问题提供快速反馈。
  - 在适用的情况下，生成包含更新代码的可部署资产或包。

- **相关云服务和工具**

  CodeArts Pipeline

6.8.2 OPS04-02 采用持续部署模型

当部署出问题时，通过使用持续部署模型来实现尽早发现问题，减少对最终用户的影响。

金丝雀部署是持续部署的常见模型，通过一小群内部或外部用户首先部署新功能，当新版本没有问题后，陆续部署到更大的组，直到所有用户群体都运行新版本。

另一种常见的部署模型是蓝绿部署，通过部署了两组相同的工作负载实例，分别处理完整的生产负载。第一个（蓝色）实例处理所有工作负载。第二个（绿色）实例已使用新功能进行更新并进行了内部测试。经过内部测试后，生产流量的子集从蓝色实例路由到绿色实例。与金丝雀部署一样，当您引流更多流量转移到绿色实例时，引流是渐进的。完成转出后，更新实例将变为蓝色实例，绿色实例已准备好进行下一次部署。这两个实例在逻辑上彼此分离，以防止发生故障。

- **风险等级**

  高

- **关键策略**

  选择这两种模型时，部署的每个阶段之间的时间应该足够长，以便能够监控工作负载的运行状况指标。应该提供充足的部署间隔时间（即部署组之间的时间），

以确保来自不同区域的用户或执行不同任务的用户有时间使用工作负载。间隔时间应以小时和天而不是分钟来衡量。每个部署组的间隔时间也应该增加，以便考虑不同的时区和使用模式。

- **相关云服务和工具**

  CodeArts Deploy

6.8.3 OPS04-03 基础设施即代码

基础设施即代码（IaC）是指使用代码而不是手动流程来管控基础设施的能力。 应用程序环境都需要许多基础设施组件，例如操作系统、数据库连接和存储。 开发人员必须定期设置、更新和维护基础设施，以开发、测试和部署应用程序。 手动管理基础设施既耗时又容易出错，尤其是在大规模管理应用程序时。

- **风险等级**

  高

- **关键策略**

  使用声明式工具：与命令式工具相比，声明式工具是部署和管理 IaC 的更好的整体选择。声明性工具对其定义文件使用更简单的语法，仅定义部署完成后所需的环境状态。命令式工具需定义达到所需最终状态所需的步骤，因此文件可能比声明性文件复杂得多。声明性定义文件还有助于减少维护命令式代码（例如部署脚本）的技术债务，这些技术债务会随着时间的推移而增加。

  使用云平台工具和其他经过行业验证、集成到平台中的工具：云平台提供的工具可以使 IaC 的部署变得简单直接。利用这些工具而不是开发自己的解决方案。云平台包含满足您大多数需求的内置功能，并且由平台提供商不断更新，随着平台的发展而变得更加有用。

  标准化模块化方案：模块可以使基础设施部署可重复，标准化有助于确保模块的构建能够满足特定目标。建议使用模块来封装复杂的配置或资源组合。此外，在开发新模块时非敏感场景可以适当使用开源模块。

  标准化人工步骤：如果存在与部署和维护相关的人工步骤，要尽可能减少这类活动。在运维指南和标准操作程序中，清楚地记录人工步骤，并实现标准化，以确保安全、一致地执行任务。

  回收闲置资源：由于配置管理工具及其限制等原因，有时 IaC 工具无法自动删除资源。例如，假设需要从虚拟机迁移到 PaaS 服务，而 IaC 工具没有删除闲置资源的逻辑。如果忘记手动删除这些资源，这些资源可能会成为孤立资源。为了处理这些场景，需要标准化扫描闲置资源并明确删除策略。

- **相关云服务和工具**

  资源编排服务 RFS

6.8.4 OPS04-04 自动化工程运维任务

在日常开发工作中，尽可能自动化一切，以减轻管理负担并最大限度地减少人为错误。为了最大限度地提高自动化投资的价值，优先考虑简单、程序化且长期的任务。应用自动化并不是一种全有或全无的策略。即使需要人工干预的工作流(例: 决策点)，也可以从自动化中受益。

- **风险等级**

  高

- **关键策略**

  优先考虑从自动化中受益最多的任务：

  - **专注于高度程序化且容易出现人为错误的任务：**这些任务被明确定义，高度自动化，没有增加复杂性的变量，并且作为正常路径的一部分执行。示例包括：重新启动服务器、创建帐户以及将日志传输到数据存储。这些任务可能会按计划发生，作为对事件或监视警报的响应，或者根据外部因素的需要而发生。
  - **可以解放运维工程师的任务：为应用的DevOps团队提供自动服务，通过运行的脚本自动执行运维操作步骤。**例如，客户引入多租户解决方案时，数据库管理员经常收到创建新数据库的请求。如果为运营人员构建自助服务门户，则可以让他们自己安全地创建空数据库。
  - **通过自动化显著提升效率的任务：**高价值的自动化需要最少的管理开销，并显着提高效率。例如，如果可以通过自动化数据库条目每天为运营团队节省一个小时，那么就可以有更多时间实现自动化做持续改进。

- **设计建议**
  - **管道定义、执行和管理**：使用持续集成和持续交付 (CI/CD) 工具（例如 华为云**CodeArts Pipeline**）自动定义管道及其运行方式.
  - **部署**：使用华为云**资源编排服务 RFS**、Terraform 和 Ansible 等工具来自动化工作负载开发和发布流程。通过使用基础架构即代码 (IaC) 方法，可以使用相同的自动化平台部署并优化基础架构。
  - **测试**：许多工具可用于自动化测试过程。这些工具可以减轻质量保证团队的重大负担，并确保测试标准化且可靠。
  - **扩展**：使用平台提供的功能和其他工具（例如: **资源编排服务 RFS**），在负载增加或减少时自动扩展基础架构。
  - **监控和警报**：使用**云运维中心 COC**和**云监控服务 CES**提供的工具自动注册新部署的资源并配置警报触发的操作，以帮助在出现问题时加快修复速度。
  - **自我修复**：使用**云监控服务 CES**生成的警报来自动执行操作并恢复出现故障的组件或作业。
  - **配置管理**：使用编排和策略工具确保所有资源运行相同的配置，并在整个工作负载中强制执行合规性要求。
  - **其他管理任务**：使用脚本自动执行重复性任务，例如更新数据库记录或 DNS 记录。
  - **审批**：使系统能够根据预定义规则自动做出审批决策，以提高具有审批关口的工作流程的效率。这种方法鼓励使用标准化表格和模板，从而提高流程的效率。在高环境下自动批准可能存在风险。密切关注并测试您的自动批准，以确保定义特定标准来授予批准。
  - **新用户和新员工入职**：您可以自动执行与新应用程序用户或新员工入职相关的许多任务，例如数据库更新和凭据创建。

- **相关云服务和工具**

  资源编排服务 RFS

  CodeArts Pipeline

  CodeArts Deploy

  云运维中心 COC

  云监控服务 CES

  华为云命令行工具服务 KooCLI

6.9 OPS05 运维准备和变更管理

6.9.1 OPS05-01 进行生产准备度评审（Product Readiness Review）

- **风险等级**

  高

- **关键策略**

  Production Readiness Review 生产准备度评估基线：从SLI/SLO、可冗余、可容灾、可过载控制、可故障管理、可变更能力、可运维、安全生产等维度，对服务可用性及运维能力提出基线要求。在服务产品开发前端构筑能力，进行相关需求规划、设计和开发工作，并在服务上线前进行生产准入审视。

  具备以下核心价值：

  1）准确评价产品可用性、维护能力并明确相关上线标准；不满足上云标准的服务，原则上不允许上线。

  2）持续导入服务可服务性、运维需求基线，实现标准化、减少例外操作，帮助服务快速上云。

  3）持续提升自动化验证能力，减少手工评估，提升产品的交付与运维效率。

- **相关云服务和工具**

  COC PRR评审

6.9.2 OPS05-02 进行变更风控

- **风险等级**

  高

- **关键策略**

  根据不同变更场景构建风险控制能力，通过风险数字化度量分析和评估风险影响程度，并采取风险控制措施削减或规避风险，保障变更成功。变更风险指现网各要素增、删、改及状态改变（如版本迭代、配置改变、节点扩缩容等）时引发的业务中断风险及变更失败可能导致的业务受损风险。

  设计建议

  变更风控衡量指标：变更风控衡量指标为变更导致事件密度和变更引入重大事件数。

  变更导致事件密度定义：每月变更导致对客户造成影响的事件数与总变更数的比值。

  计算公式：变更导致事件密度=变更导致对客户造成影响的事件数/总变更数。

  变更引入重大事件数定义：每月变更引入对客户产生重大影响的事件次数。

- **相关云服务和工具**

  COC 变更管理

6.9.3 OPS05-03 定义变更流程

- **风险等级**

  高

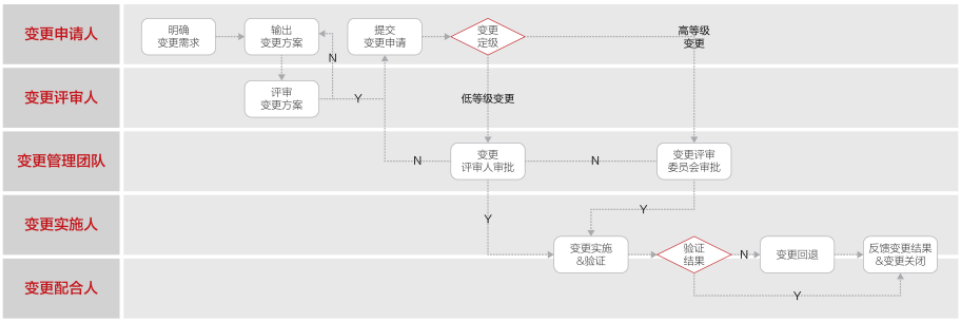
- **关键策略**

  在建立标准的变更管理流程前，清晰明白变更各个流程的定义：

  - 变更发起：在变更发起前，需明确变更内容与变更原因等信息。信息的明确可减轻变更评估人的工作量，同时明确变更的意义。变更信息包括：

  - 基本信息：标题、时间、变更人、原因等。
  - 变更信息：变更系统、变更场景、变更类型等。
  - 变更方案：变更实施方案、回滚方案、验证方案等。
- 变更审批：由于变更系统相关的负责人进行审核，确保变更风险级别，若无法控制或无法预测，则建议明确变更方案或禁止变更。变更审批流程可由多人进行组合，包括：业务负责人、团队TL、技术TL等，变更涉及的人员可根据变更的影响程度以及影响范围等因素确定。
- 变更执行：通过发起时确认的执行人来进行执行工作的分派，以确保执行变更的是与变更内容相关的技术人员，从而确保变更的准确执行。
- 变更验证：在变更完成后，对变更对象与变更内容进行检查，确保变更并未影响实际业务，检查完成后，发布变更结果。
- 变更关闭：在变更完成后，关闭变更任务。对变更记录进行留存，便于后续变更数据的运营与分析。

- **设计建议**



6.10 OPS06 可观测性体系

6.10.1 OPS06-01 建立可观测性体系

可观测性（observability）最初是系统理论中的一个概念，指系统的状态能否被外部观察到和重现。随着云原生、微服务架构的发展，IT系统对可观测性的需求日益增强。业界对可观测性的定义：通常是指基于对复杂系统外部输出的了解，能够了解其内部状态或状况的程度。系统越可观测，定位问题根本原因的过程就越快速越准确，而无需进行额外的测试或编码。

- **风险等级**

  高

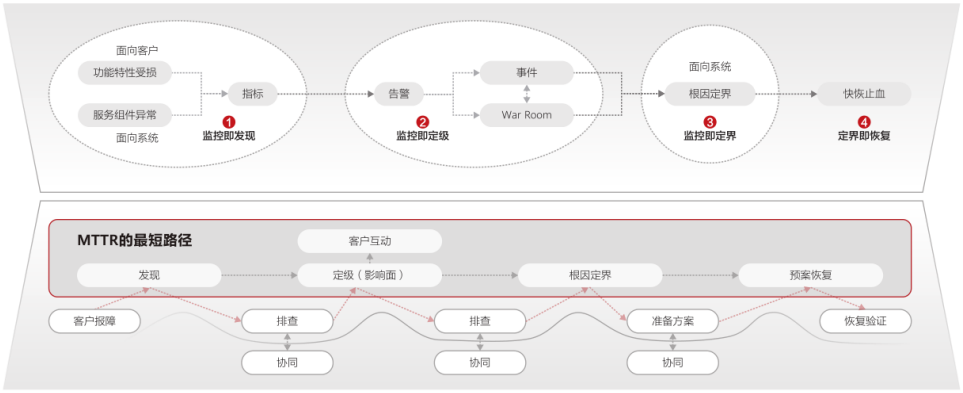
- **关键策略**

  可观测体系是围绕确定性恢复命题展开的，决定了确定性恢复能力构建与 SLO 达成。可观测体系能够直接决定一些故障的恢复时长，如下图所示，MTTR 平均恢复时长由平均发现时长、平均定界时长和平均处置时长三部分构成，而可观测能决定的是发现时长和定界时长（经验值占比 1/2 左右）。在一个事件里，MTTR 的恢复时长越短，那么它的整体 SLO 达成可能性就越高。

  MTTR平均恢复时长=平均发现时长+平均定界时长+平均处置时长

- **设计建议**

面向 MTTR的可观测体系设计的核心逻辑就是寻找最短恢复路径。如下图所示案例，在故障恢复 MTTR 的逻辑中，当业务发生故障，从故障发现、到故障定级和影响面分析、再到故障定界定位和故障恢复，几乎全部依赖人工处理。要想缩短时间，本质上是监控即发现、监控即定级、监控系统定界、定界即恢复——如果能达成这样的设计就能够形成 MTTR 的最短路径。



6.10.2 OPS06-02 定义可观测对象

- **风险等级**

  高

- **关键策略**

  客户可感知的观测对象分类如下：

可观测分层	功能 / 主要指标
IT 资源监控	IT 资源监控对 IT 资源的性能和容量进行监视和报告，确保您的业务稳定可靠运行
应用监控	应用监控基于应用资源管理对资源实行从应用、业务组件、到环境的分层监控，每一层对应的观测指标均不同。在应用层，主要监控业务层、应用层、中间件层以及基础设施层告警信息，同时通过绑定当前应用的仪表盘，以图表的形式展示指标源、日志源以及系统图表信息。主要关注：WAITING 状态线程数、TIMED_WAITING 状态线程数、可使用内存等指标
进程监控	进程监控是针对主机内活跃进程进行的监控，默认采集活跃进程消耗的 CPU、内存，以及打开的文件数量等信息。当您配置了自定义进程监控，还会监控包含关键字的进程个数。主要关注：运行中进程数、空闲进程数、僵死进程数等指标
日志监控	配置日志服务从日志中提取指定的关键词，便于您使用监控服务对日志中的关键指标进行监控及告警。主要关注：访问日志数量、错误日志数量、日志大小等指标
自定义监控	自定义监控展示用户所有自主定义上报的监控指标。用户可以针对自己关心的业务指标进行监控，将采集的监控数据通过使用简单的 API 请求上报至监控服务进行处理和展示

中间件监控	提供快捷安装配置各类型中间件插件的功能，并提供开箱即用的专属监控大盘，目前支持的中间件插件有以下几种： MYSQL、REDIS、MONGODB、NGINX、NODE、HAPROXY、COMP_EXPORTER、COMP_REDIS_EXPORTER、COMP_MYSQL_EXPORTER
主机监控	主机监控提供了包括基础监控和操作系统监控两种不同监控粒度层次的监控。基础监控为 ECS自动上报的监控指标，操作系统监控通过在ECS中安装Agent插件，为用户提供服务器的系统级、主动式、细颗粒度监控服务。主要关注：CPU_UTIL、DISK_READ_BYTES_RATE、带外网络流入速率等指标
用户线下 组件监控	对用户的线下组件统一监控，主要支持：线下 Grafana 对接、线下自建 Prometheus 对接、业务监控、应用监控、线下 IDC 监控和线下中间件监控
网络性能 管理监控	功能：对客户端 - 网 - 边 - 云全链路网络进行监控，帮助用户及时发现网络故障，全面掌握网络的实时状况。主要关注：应用响应时间、DNS 解析时间、TCP 建连时间、访问流量等指标

6.10.3 OPS06-03 制定和实施可观测性指标

- **风险等级**

  高

- **关键策略**

  指标是对时间周期内的测量数据的数值表示。可观测性指标是围绕发现率、定级准确率、定界时长、覆盖率、有效率、 一致率打造可观测能力，将可观测设计规范统一发布，统一设计要求与运维管理要求。

- **设计建议**

  整体技术方案会变成标准并进行发布，各个业务系统架构师在设计时遵循这套标准，这样可以保证能力能够从设计态开始，包括运行态、高可用架构等场景中得到应用。

可观测指标可以通过监控工具来实现，并允许在发生异常时发送警报。有很多监控工具可以使用，例如Prometheus、Grafana、Zabbix等，以及华为云提供的云监控服务CES。这些工具可以定期收集指标，提供可视化的指标报告，并且可以发送警报，以帮助组织及时发现问题。

可参考CES的最佳实践，**https://support.huaweicloud.com/bestpractice-ces/ces_14_0002.html**。

6.10.4 OPS06-04 规范化应用日志

日志是随时间推移发生的不可变、记录时间戳的离散事件。系统需要记录关键事件和故障，以帮助诊断问题和解决故障。

- **风险等级**

  高

- **关键策略**

  对于一个系统来说，日志是非常重要的。它可以记录在系统中发生的一切，包括成功的操作、错误的操作、警告信息等等。因此，日志记录是可观测性设计中最基本的需求之一。通过将事件和错误信息记录到日志文件或数据库中，可以方便地进行故障排除和问题诊断。但是，仅仅记录日志并不足够，还需要对日志进行有效的管理和分析。如果日志太多，将会成为一个负担，因为它们需要占用存储空间，并且需要花费很长时间来查找有用的信息。因此，需要对日志进行过滤和归档，以便更好地管理它们。

- **设计建议**

  可参考**LTS最佳实践**

6.10.5 OPS06-05 实施依赖项遥测

- **风险等级**

  高

- **关键策略**

  依赖项遥测可以监控工作负载所依赖的外部服务和组件的运行状况及性能。提供有关与 DNS、数据库或第三方 API 等依赖项相关的可访问性、超时及其他关键事件的高价值指标采集。当对应用程序进行检测，以发布有关这些依赖项的指标、日志和跟踪时，就能更清楚地了解可能影响工作负载的潜在瓶颈、性能问题或故障。

6.10.6 OPS06-06 实施分布式跟踪

Trace是一系列因果相关的分布式事件的表示，这些事件编码了流经分布式系统的端到端请求流。

- **风险等级**

  高

- **关键策略**

  当系统出现问题时，需要能够追踪系统中每个组件的行为和交互情况。通过在系统中实现分布式跟踪，可以快速定位问题并进行有效的故障排除。

- **设计建议**

  链路跟踪可以通过在系统中添加跟踪标识符来实现。当请求进入系统时，标识符将被添加到请求中，并在整个系统中传递。每个组件都可以将标识符添加到它们的日志中，以便在出现问题时进行故障排除。分布式跟踪可以使用开源工具Jaeger、Zipkin、skywalking或CAT等,华为云APM提供了调用链观测能力。

  可参考**APM最佳实践**

6.10.7 OPS06-07 通过可观测性指标引入自动化措施

- **风险等级**

  高

- **关键策略**

  可观测与自动化运维工具联动，实现自动化的故障检测、恢复及弹性伸缩等功能，进一步提升运维响应速度和准确性，降低人为干预带来的延误，甚至错误。

6.11 OPS07 进行故障分析和管理

6.11.1 OPS07-01 创建可操作的告警

- **风险等级**

  高

- **关键策略**

  收到告警时，一般需要做出响应，消除无须响应的告警。 比如磁盘IO 量瞬间飙升，CPU 使用率瞬间飙高，这类告警无需做出响应，对业务而言，意义就不大了。 遵循可操作性原则能避免很多误报。并且要定期统计和分析告警频率，识别高频告警，解决告警问题，清除明确的告警误报。

- **设计建议**

  – 优化告警阈值：适当提高 内存 / CPU / 网络 IO 告警阈值。

- 优化日志级别：优化不合理的日志级别，把部分 ERROR 级别的日志调整为WARNING。
- 屏蔽某些日志：对难以调整日志级别的应用，根据关键字屏蔽某些频繁的日志告警。
- 预警增强：对于某些影响业务方的操作，提供预警。
- 增强紧急预警：有些硬件故障会出现反应在 /var/log/messages 中，根据关键字匹配硬件类告警，以便及时处理。

- **相关云服务和工具**

  应用运维管理 AOM

  云运维中心 COC

  云监控服务 CES

6.11.2 OPS07-02 创建监控看板

- **风险等级**

  高

- **关键策略**

  监控看板为您提供自定义查看监控数据的功能，将您关注的核心服务监控指标集中呈现在一张监控看板里，为您定制一个立体化的监控平台。同时监控看板还支持在一个监控项内对不同服务、不同维度的数据进行对比查看，实现不同云服务间性能数据对比查看。

- **华为云相关云服务和工具**

  云监控服务 CES

  云运维中心 COC

6.11.3 OPS07-03 支持事件管理

- **风险等级**

  高

- **关键策略**

  事件(incidents)是需要干预的事情。当发生事故(incidents)时，通过流程来处理。如何与团队沟通活动的状态？谁负责响应处置？使用哪些工具来缓解该事件？这些都是流程中需要回答的问题，并需要获得可靠的响应过程。流程必须中心化，并且可供参与工作负载的任何人使用。如果没有wiki 或文档存储，可以使用源代码版本控制机制。

  优先通过自动化响应事件，避免占用业务交付和创新的时间。首先构建一个可重复的流程来缓解问题，然后关注自动缓解或解决根本问题以提升效率。

- **华为云相关云服务和工具**

  云监控服务 CES

  云运维中心 COC

6.11.4 OPS07-04 支持故障恢复流程

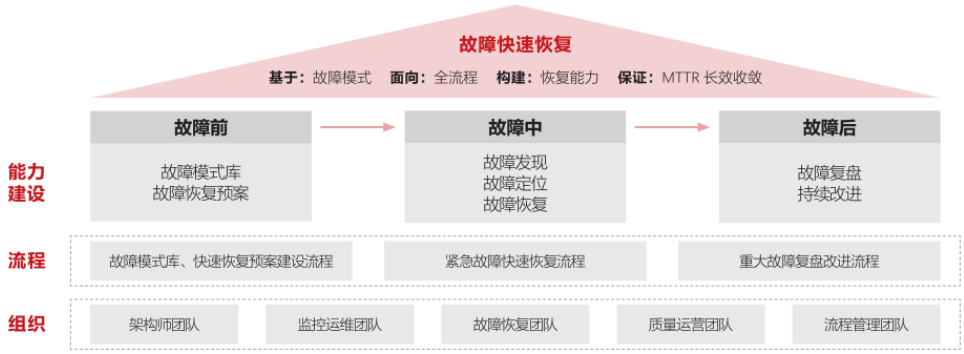
- **风险等级**

  高

- **关键策略**

  当现网发生故障时，既要快速恢复业务，又要降低影响，首先需要围绕故障全生命周期采取一系列控制流程，包含故障预防、故障发现、故障定位、故障恢复、故障复盘及持续改进（含故障演练），基于故障模式库，面向全流程、构建恢复能力、保证平均恢复时长（MTTR）的长效收敛，实现故障的快速恢复。

- **设计建议**



6.12 OPS08 度量运营状态和持续改进

6.12.1 OPS08-01 使用度量指标衡量运营目标

- **风险等级**

  高

- **关键策略**

  定义清晰的运营成功的目标和 KPI，设置基线作为参考点并定期重新评估。与业务领导者和利益相关者确定服务的总体目标。确定各个运营团队的任务以及可能面临的挑战。并明确运营目标的关键绩效指标 (KPI)，可能是客户满意度、TTM、平均问题解决时间等等。根据 KPI，识别关键指标和数据源。客户满意度可能是各种指标的组合，例如呼叫等待或响应时间、满意度评分以及提出的问题类型。

6.12.2 OPS08-02 进行事故复盘和改进

事故分析的目的是：规范和指导重大事故发生后，优化事故的输入、输出，确保事故回溯工作有效开展，回溯报告中发现的问题有效整改，总结的经验有效推广。

- **风险等级**

  高

- **关键策略**

  故障发生后，通过对现网重大故障处理过程 Review 及根因进行分析和改进总结，规范整个恢复过程，实现对可用性和技术能力的提升。故障复盘的技术过程按照 RASA 法、Review（回顾）、Analyze（分析）、Summary（总结）、Action（行动）。

  - Review（回顾）：完整记录故障的发生、发现、根因定位、决策、处理、预案执行、回滚、故障解决等的关键人与关键时间点，保证信息尽可能的客观、准确。
  - Analyze（分析）：分析故障的根本原因及故障处理过程中优化点。

- Summary（总结）：总结本次故障及处理故障的过程。进行故障定性、故障定责及总结本次故障带来的经验教训并举一反三。
- Action（行动）：确定上面分析总结的结论，进行改进、优化及落地实施。

6.12.3 OPS08-03 知识管理

- **风险等级**

  高

- **关键策略**

  日益庞大的数据量和复杂的业务系统，对运维人员的要求越来越高。为了方便运维人员获取知识，学习和解决问题，运维知识管理能力变得必要。运维知识管理应集成丰富的运维知识，可以帮助运维人员快速解决问题，提高工作效率。一般通过运维知识库系统承载，运维人员可以轻松地查找和获取各种运维知识，包括网络配置、服务器管理、数据库维护等方面的知识。下面将介绍运维知识库系统的五个主要功能和优势。

  - 丰富的知识资源：运维知识库系统收集整理了大量的运维知识和经验，涵盖了各个领域和层次的内容。用户可以通过系统进行检索，查找到相关的知识和解决方案。不仅可以解决一些常见的问题，还可以提供高级的技术支持，帮助用户解决复杂的问题。
  - 快速定位问题：运维知识库系统配备了强大的搜索功能，用户可以根据关键词进行搜索，系统会自动匹配相关的知识，并提供相应的解决方案。用户只需通过简单的操作，就能快速定位问题，并找到解决方案，节省了大量的时间和精力。
  - 知识分享和交流：运维知识库系统还支持用户之间的知识分享和交流。用户可以将自己的经验和知识上传到系统中，与其他用户进行交流和讨论。这不仅可以扩展自己的知识面，还可以通过与其他人的交流，获得更多的解决方案。
  - 随时更新和维护：运维知识库系统会不断更新和维护知识库中的内容，保证系统中的知识始终是最新的和准确的。用户可以通过订阅功能，及时获取到最新的知识和解决方案。这样可以保证用户能够始终处于技术领先的状态，应对各种复杂的问题。
  - 提供多种形式的知识展示：运维知识库系统支持多种形式的知识展示，包括文字、图片、视频等。这样可以满足用户的不同需求，让用户可以更直观地理解和掌握知识。用户可以根据自己的喜好选择适合自己的知识展示形式。、

6.13 参考案例

6.13.1 通过 AOM 助力系统运维能力提升，降低运维成本与难度

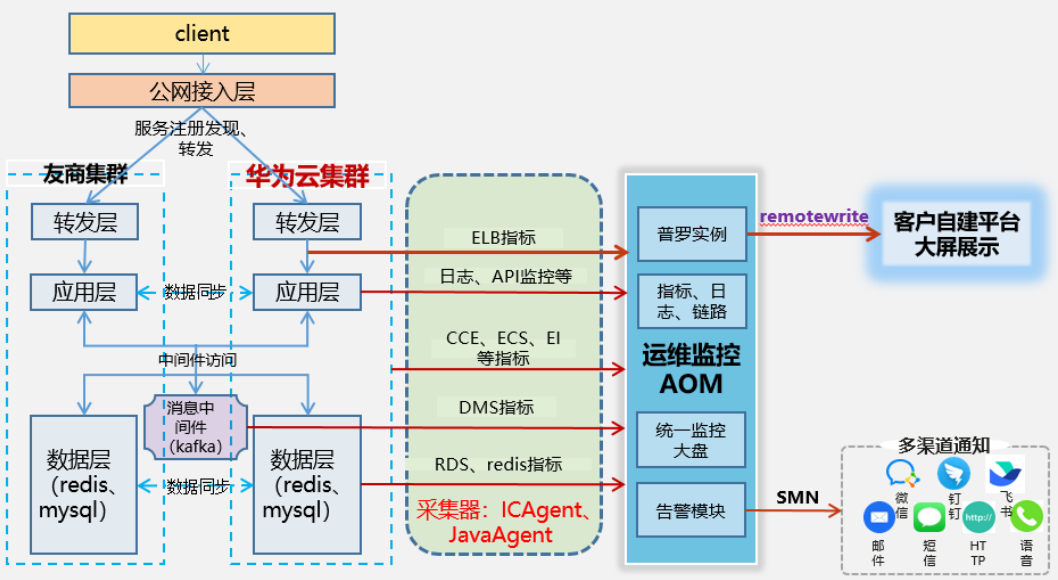
某平台服务的认证驾驶员用户1000万人，货主用户500万人，集团业务覆盖全国339个主要城市，覆盖线路数量超过11万条，实现了全国多中心运营的架构。

客户痛点：

- 多云双活场景运维难保障：大规模集群场景，单个云厂商灾备不足以保障业务，需引入双活并行，故障零切换，过程中，客户自建运维平台能力较为欠缺，不足以满足运维需要

- 无法采集云服务指标信息：客户自建运维体系无法采集到云服务等场景的指标信息，不能满足大屏展示需要
- 告警通知能力不足：自建运维平台告警通知能力不能完全满足多场景通知的需要，且没有告警降噪能力

解决方案：



业务价值：

- 降低了运维成本与难度：降低了运维多套系统的难度，减少了客户运维起步的资源投入，降低了运维成本
- 运营分析能力提升：基于可视化图表和开箱即用的仪表盘等强大功能，快速实现对业务的运营分析
- 排障能力提升：云端多维度监控实现对业务立体运维，结合自动告警规则达到对故障的快速感知定位处理

6.13.2 基于 LTS 采集多类端侧日志，问题全链路追踪分析和业务运营分析

某公司核心业务专注于IT信息传播、技术交流、教育培训和专业技术人才服务。拥有超过3200万注册会员、超过1000家企业客户及合作伙伴。

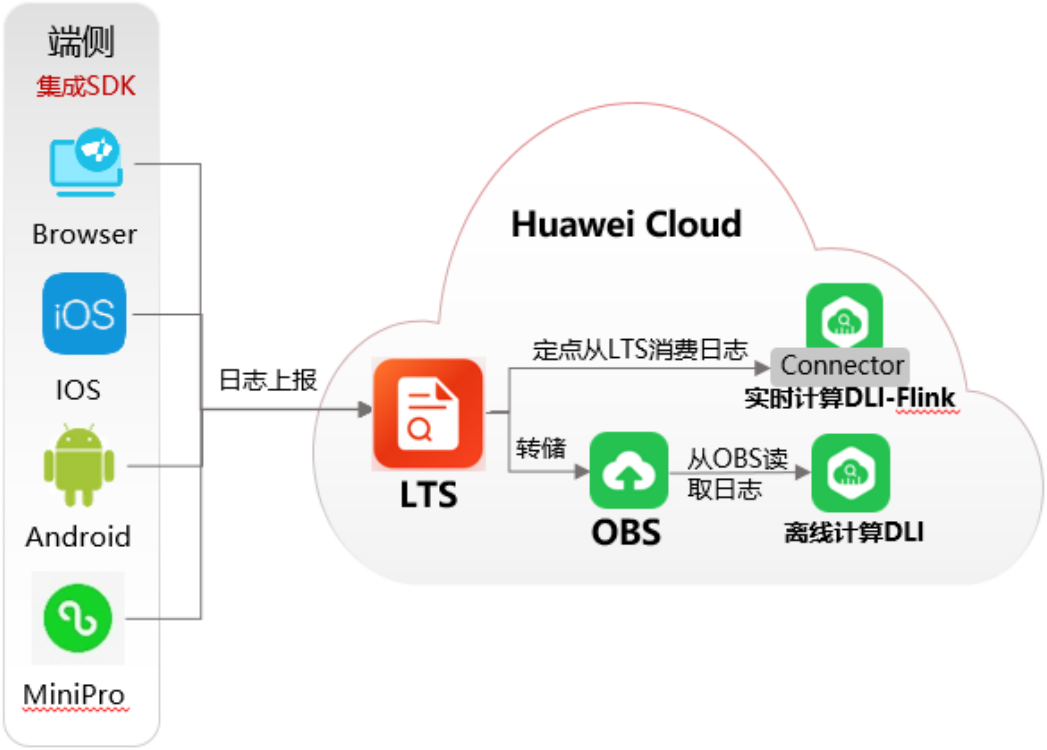
客户痛点：

- 端侧采集工具不统一，不支持自定义域名上报，问题定位复杂

Web、IOS、安卓、百度小程序、微信小程序等多类端侧日志无法使用同一家厂商工具采集，问题定位分析时，需在多个工具间需来回切换，增加了定位复杂度，且无法自定义日志上报的服务端域名，合规性受到部分用户质疑

- 端侧日志上报慢且易丢失：上报速度小时级，也极易出现丢失，对问题端到端定位分析、业务完整性分析均造成一定影响
- 业务挖掘分析难：日志数据无法直接写入DLI，需投递到Kafka后，再被DLI消费，链路长，且成本高

解决方案：



业务价值：

- 端侧日志全面采集接入，自定义域名上报：集成LTS提供的多端SDK，全面采集端侧日志，接入LTS，且支持上报服务端域名自定义，在用户面保持了业务一致性与合规性，降低了问题定位复杂度，提升了运维效率
- 端侧日志数据毫秒级上报，数据0丢失：端侧采集日志后，毫秒级完成上报，且无数据丢失，支撑客户快速完成从前端到后端对问题做全链路追踪分析，同时，也支持对业务做完整性分析
- 便捷低成本获取日志，助力业务挖掘分析：DLI-Flink简易集成Connector，定点从LTS实时消费日志，做实时业务计算分析；简易化配置LTS日志转储到OBS，供DLI快速从OBS读取日志，做离线业务计算分析

6.13.3 LTS 助力某公司高效完成日常业务运维与等保合规

某公司是一家拥有IT，汽车及新能源三大产业群的新技术民营企业。2022年8月，公司入选2022年《财富》世界500强排行榜。

客户痛点：

- 业务部门较多，日志量较大，项目管理较为困难
- 云服务资源种类数量较多，监控指标和运维日志不熟悉，运维难度大
- 等保合规要求日志长时间存储，运维部门较多，人员不足，自建ELK成本高

解决方案：

业务价值：

全量日志接入：汽车APP、软件开发、流量平台等170个业务系统接入云日志服务，全面覆盖业务、应用、中间件和基础设施。

分钟级问题定界：秒级日志查询和分钟级日志监控，可配置告警和多渠道通知，90%问题感知与定位分析控制在30分钟。

存储时长满足等保要求：支持存储时长最大为365天，满足等保合规要求，智能冷热存储可降低存储成本，且提供便捷检索能力。

6.14 卓越运营云服务介绍

6.14.1 软件开发生产线(CodeArts）

软件开发生产线（CodeArts）是一站式、全流程、安全可信的DevSecOps平台，开箱即用，内置华为多年研发最佳实践，助力效能倍增和数字化转型。

CodeArts由以下几个主要服务构成：

- 需求管理：提供需求管理与团队协作服务，内置多种开箱即用的场景化需求模型和对象类型（需求/缺陷/任务等），可支撑IPD、DevOps、精益看板等多种研发模式，还包含跨项目协同、基线与变更管理、自定义报表、Wiki在线协作、文档管理等功能。
- 代码托管：基于Git提供分布式代码管理和协同开发能力，包括成员管理、权限控制、代码托管、代码检查、代码审核、代码追溯、持续集成等功能，助力不同规模企业的研发质量和效率提升。
- 流水线：提供可视化、可定制的持续交付流水线服务，实现缩短交付周期和提升交付质量的效果。
- 代码检查：为用户提供代码风格、通用质量与网络安全风险等丰富的检查能力，提供全面质量报告、便捷的问题闭环处理帮助企业有效管控代码质量，助力企业成功。
- 编译构建：基于云端大规模分布式加速，为客户提供高速、低成本、配置简单的混合语言构建能力，帮助客户缩短构建时间，提升构建效率。

- 部署：支持主机、容器等多种部署形态，部署能力覆盖Tomcat、Springboot等多种语言和技术栈。基于其对部署功能的插件化封装和编排能力，帮助您实现软件的快速、高效发布。
- 测试计划：覆盖测试计划、测试设计、测试用例、测试执行和测试评估等全流程，旨在帮助企业协同、高效、可信的开展测试活动，保障产品高质量上市。
- 制品仓库：用于管理源代码编译后的构建产物，支持Maven、Npm等常见制品包类型。可以与本地构建工具和云上的持续集成、持续部署无缝对接，同时支持制品包版本管理、细粒度权限控制、安全扫描等重要功能，实现软件包生命周期管理，提升发布质量和效率。
- CodeArts IDE Online：基于云计算的轻量级WebIDE，通过浏览器即可实现环境快速获取和环境访问，完成编码、构建、调试、运行、访问代码仓库和命令执行等工作，支持第三方业务集成，支持插件扩展并提供独立插件市场。
- 开源镜像站：由华为云提供的开源组件、开源操作系统及开源DevOps工具镜像站，目前已提供Maven、NPM、NuGet、CentOS、Ubuntu、Debian等镜像下载服务。

6.14.2 资源编排服务(RFS)

资源编排服务是完全支持业界事实标准Terraform（HCL + Provider）的新一代云服务资源终态编排引擎，在应用编排服务(AOS)基础上实现了生态、体验、特性的全新升级；资源编排服务基于业界开放生态HCL语法模板，实现云服务资源的自动化批量构建，帮助用户高效、安全、一致创建、管理和升级云服务资源，能有效提升资源管理效率，并降低资源管理变更带来的安全风险。

6.14.3 云运维中心(COC)

云运维中心（Cloud Operations Center，简称COC）为用户提供安全、高效的一站式智能运维平台，满足客户集中运维诉求。承载华为云确定性运维业务场景，提供变更管理、批量运维等核心特性，实现在安全合规的前提下，提升用户运维能力成熟度和云上运维效率。COC产品介绍：

统一资源管理

- 应用管理：提供应用和资源关联关系建模能力，满足用户云上资源的集中式管理要求，降低管理成本。
- 资源管理：同步并纳管用户在云平台上使用的资源实例，构筑资源运维能力底座。
- 配置管理：提供应用和资源视角的管理能力，以及参数配置集中式看护、全生命周期管理的能力。
- 合规性管理：资源运维提供批量的补丁扫描修复能力，安全合规先行，兼顾高效。

全方位变更管理

- 方案评审：支持变更方案标准化（Standard Operating Procedure，简称SOP），将变更方案明确并电子化，经评审后归档。支持规则和流程解耦，保证变更执行过程不走样，同时将变更方案沉淀。
- 变更审批：按照预设审批流程审批变更单，保障变更方案可靠性、时间合理性、流程合规性。
- 风险评估：基于场景规则、流程规则、业务规则对变更进行管控，提前识别和拦截变更风险；通过变更日历实现变更冲突检测，降低服务间变更依赖导致的变更风险。

- 实施保障：按预定方案执行变更，变更步骤标准化、可观测，变更异常及时介入处理，实现变更实施全过程可控、可视、可管。

确定性故障管理

- 统一事件中心：提供事件发现、事件处理、恢复验证及持续改进的全流程标准化机制。
- 承载Warroom和故障回溯能力：现网事件智能启动Warroom，缩短故障处理非必要耗时，指挥中心实时观测故障处理进展。故障回溯实现问题总结和经验沉淀，客户问题不重犯，缩短故障恢复MTTR。
- 支持响应预案：支持客户对已知故障制定响应预案，通过预案自动化帮助客户处理确定性问题，实现已知问题快速恢复。
- 故障模式：融合专业风险分析方法和专家知识库，积累故障模式库，帮助客户分析云应用存在的潜在风险、传承运维经验。

韧性中心优化

- 全生命周期风险管理：覆盖部署态和运行态两部分的风险治理，贯穿应用和资源全生命周期，将华为云多年沉淀的动态清零风险管理经验使能用户。
- 使能主动运维：通过性能压测、应急演练/混沌工程、韧性评估等主动运维手段提升客户关键业务的质量和韧性。
- 丰富的故障演练武器：沉淀华为云实践经验，内置50个+演练攻击武器，赋能客户模拟复杂多样的业务受损场景并制定应对策略。
- 降本增效：在不影响用户云上业务的同时，从专业的角度提供优化建议，帮助客户降本、增效和业务调优。

6.14.4 云监控中心(CES)

云监控服务为用户提供一个针对弹性云服务器、带宽等资源的立体化监控平台。使您全面了解云上的资源使用情况、业务的运行状况，并及时收到异常告警做出反应，保证业务顺畅运行。

云监控服务主要具有以下功能：

- 自动监控：云监控服务不需要开通，在创建弹性云服务器等资源后监控服务会自动启动，您可以直接到云监控服务查看该资源运行状态并设置告警规则。
- 主机监控：通过在弹性云服务或裸金属服务器中安装云监控服务Agent插件，用户可以实时采集ECS或BMS 1分钟级粒度的监控数据。已上线CPU、内存和磁盘等40余种监控指标。有关主机监控的更多信息，请参阅主机监控简介。
- 灵活配置告警规则：对监控指标设置告警规则时，支持对多个云服务资源同时添加告警规则。告警规则创建完成后，可随时修改告警规则，支持对告警规则进行启用、停止、删除等灵活操作。
- 实时通知：通过在告警规则中开启消息通知服务，当云服务的状态变化触发告警规则设置的阈值时，系统通过短信、邮件通知或发送消息至服务器地址等多种方式实时通知用户，让用户能够实时掌握云资源运行状态变化。
- 监控面板：为用户提供在一个监控面板跨服务、跨维度查看监控数据，将用户关注的重点服务监控指标集中呈现，既能满足您总览云服务的运行概况，又能满足排查故障时查看监控详情的需求。
- OBS转储：云监控服务各监控指标的原始数据的保留周期为两天，超过保留周期后原始数据将不再保存。您可以在对象存储服务（Object Storage Service，以下简称OBS）创建存储桶，然后将原始数据同步保存至OBS，以保存更长时间。

- 资源分组：资源分组支持用户从业务角度集中管理其业务涉及到的弹性云服务器、云硬盘、弹性IP、带宽、数据库等资源。从而按业务来管理不同类型的资源、告警规则、告警记录，可以迅速提升运维效率。
- 站点监控：站点监控用于模拟真实用户对远端服务器的访问，从而探测远端服务器的可用性、连通性等问题。
- 日志监控：日志监控提供了针对日志内容的实时监控能力。通过云监控服务和云日志服务的结合，用户可以针对日志内容进行监控统计、设置告警规则等操作，降低用户监控日志的运维成本，简化用户使用监控日志的流程。
- 事件监控：事件监控提供了事件类型数据上报、查询和告警的功能。方便您将业务中的各类重要事件或对云资源的操作事件收集到云监控服务，并在事件发生时进行告警。

6.14.5 云日志服务(LTS)

云日志服务（Log Tank Service，简称LTS）是高性能、低成本、功能丰富、高可靠的日志平台，提供全栈日志采集、百亿日志秒搜、PB级存储、日志加工、可视化图表、告警和转储等功能，满足应用运维、等保合规和运营分析等应用场景需求。

云日志服务提供多种接入方式实现海量日志接入LTS，支持日志搜索引擎、SQL分析引擎、日志加工引擎，详细请参考下图。

- 端云全场景日志接入：40+云服务、主机/容器、移动端、跨云、多语言SDK、多账号汇聚，满足全场景客户丰富的日志接入需求。
- 海量日志存储搜索：百亿日志秒级搜索，千亿日志迭代搜索，PB级智能冷存储。
- SQL统计和可视化图表：100+SQL函数、多种可视化图表、10多种开箱即用仪表盘。
- 实时日志告警：自定义告警内容，短信/邮件/微信/钉钉/HTTP多渠道通知。
- 一站式日志加工：200＋函数、一站式日志规整、富化、脱敏、过滤、分裂加工平台。
- 日志数据服务间集成：日志转储OBS/DWS/DIS/DLI/DMS，助力用户快速构建水平解决方案。

6.14.6 应用运维管理(AOM2.0)

应用运维管理（Application Operations Management，简称AOM）是云上应用的一站式立体化运维管理平台，融合云监控、云日志、应用性能、真实用户体验、后台链接数据等多维度可观测性数据源，提供应用资源统一管理、一站式可观测性分析和自动化运维方案，帮助用户及时发现故障，全面掌握应用、资源及业务的实时运行状况，提升企业海量运维的自动化能力和效率。

6.14.7 应用性能管理(APM)

华为云应用性能管理服务（Application Performance Management，简称APM）帮助运维人员快速发现应用的性能瓶颈，以及故障根源的快速定位，为用户体验保驾护航。

您无需修改代码，只需为应用安装一个APM Agent，就能够对该应用进行全方位监控，帮助您快速定位出错接口和慢接口、重现调用参数、发现系统瓶颈，从而大幅提升线上问题诊断的效率。目前支持JAVA、Python、Node.js、Go、Php和.Net应用，具体的应用监控能力概览如下表。

6.14.8 云堡垒机(CBH)

云堡垒机（Cloud Bastion Host，CBH）是华为云的一款统一安全管控平台，为企业提供集中的账号（Account）、授权（Authorization）、认证（Authentication）和审计（Audit）管理服务。

云堡垒机提供云计算安全管控的系统和组件，包含部门、用户、资源、策略、运维、审计等功能模块，集单点登录、统一资产管理、多终端访问协议、文件传输、会话协同等功能于一体。通过统一运维登录入口，基于协议正向代理技术和远程访问隔离技术，实现对服务器、云主机、数据库、应用系统等云上资源的集中管理和运维审计。

6.14.9 应用管理与运维平台(ServiceStage)

应用管理与运维平台（ServiceStage）是面向企业的应用管理与运维平台，提供应用发布、部署、监控与运维等一站式解决方案。支持Java、Php、Python、Node.js、Docker、Tomcat技术栈。支持Apache ServiceComb Java Chassis（Java Chassis）、Spring Cloud等微服务应用，让企业应用上云更简单。

ServiceStage主要包含如下能力：

1. 应用管理：支持应用生命周期管理、环境管理。
2. 微服务应用接入：支持Java Chassis、Spring Cloud微服务框架。配合微服务引擎可实现服务注册发现、配置管理和服务治理，请参考**微服务开发指南**。
3. 应用运维：通过日志、监控、告警支持应用运维管理。

6.14.10 多活高可用(MAS)

多活高可用(MAS)的混沌工程（ChaosEngineering）是一种通过主动注入故障识别并修复系统未知隐患的工程实践。MAS-CAST混沌工程服务提供丰富的故障模式库，通过混沌实验编排攻击目标、攻击策略进行故障注入，支持添加背景流量和资源监控，同时在故障注入能力的基础上，通过体系化的流程和规范来创建故障演练，从而验证和提升系统可靠性和技术团队应急响应能力。

6.15 更多参考文档

确定性运维白皮书

