AWS 架构完善的框架

2020 年 7 月

This paper has been archived.

The latest version is now available at:

https://docs.aws.amazon.com/zh_cn/wellarchitected/latest/framework/welcome.html

本文档介绍了 AWS 良好架构框架，借助它，您可以审查并改进您的基于云的架构，并更好地了解设计决策对业务产生的影响。我们讲解了五个概念领域（我们将其定义为良好架构框架的五大支柱）中的一般设计原则以及具体最佳实践与指南。

声明

客户负责对本文档中的信息进行独立评估。本文档：(a) 仅供参考，(b) 代表 AWS 当前的产品服务和实践，如有变更，恕不另行通知，以及 (c) 不构成 AWS 及其附属公司、供应商或授权商的任何承诺或保证。AWS 产品或服务均“按原样”提供，没有任何明示或暗示的担保、声明或条件。AWS 对其客户的责任和义务由 AWS 协议决定，本文档与 AWS 和客户之间签订的任何协议无关，亦不影响任何此类协议。

简介

AWS 良好架构框架能够帮助您认识到您在 AWS 上构建系统时所做决策的优缺点。通过使用此框架，您将了解在云中设计和运行可靠、安全、高效且经济实惠的系统的架构最佳实践。它提供了一种方法，使您能够根据最佳实践持续衡量架构，并确定需要改进的方面。审查架构的流程是关于架构决策的建设性对话，不是一种审核机制。我们相信，拥有架构良好的系统能够大大提高实现业务成功的可能性。

AWS 解决方案架构师拥有多年为各种垂直行业和用例设计解决方案的经验。我们也已经帮助成千上万客户对其 AWS 之上的架构进行设计与审查。从这些经验中，我们得以总结出在云中设计系统的最佳实践和核心策略。

AWS 良好架构框架囊括了一系列基础性问题，来帮助您了解某种架构是否符合云最佳实践。该框架为您提供了一种一致的方法，来对标您所期望的现代云端系统能力，建立一整套质量评估体系，以及评估实现这样的质量需要采取的具体措施。随着 AWS 不断发展，我们将继续与客户协作并增进了解，同时将现实经验融入到良好架构定义的持续完善当中。

此框架面向各类技术性角色，例如首席技术官 (CTO)、架构师、开发人员和运维团队成员。它介绍了可在设计和运行云工作负载时使用的 AWS 最佳实践和策略，提供了进一步实现细节和架构模式的链接。有关更多信息，请参阅 AWS Well-Architected 主页。

AWS 还提供用于审查您的工作负载的免费服务。AWS Well-Architected Tool（AWS WA 工具）是一种云服务，它提供统一的流程，可帮助您使用 AWS 架构完善的框架对您的架构进行审核和衡量。AWS WA 工具提供的建议可让您的工作负载变得更加可靠、安全、高效和经济实惠。

为了帮助您应用最佳实践，我们创建了 AWS Well-Architected 实验室，它可以为您提供代码和文档的存储库，让您亲自体验最佳实践的实施。我们还与精选 AWS 合作伙伴网络 (APN) 合作伙伴合作，他们是 AWS Well-Architected 合作伙伴计划的成员。这些 APN 合作伙伴拥有丰富的 AWS 知识，可以帮助您审查并改进工作负载。

定义

AWS 的专家每天都在帮助客户设计系统，以利用云中的最佳实践。在设计过程中，我们与您一起对架构做出权衡。当您在真实环境中部署这些系统时，我们将关注这些系统的运作状况，同时衡量上述调整的效果。

基于实践经验，我们构建了 AWS 良好架构框架，它为客户和合作伙伴评估架构提供了一系列最佳实践，并提供了相应的可用于评估架构是否符合 AWS 最佳实践的问题。

AWS 架构完善的框架建立在五个支柱的基础上，它们是卓越运营、安全性、可靠性、性能效率和成本优化。

表 1. AWS 架构完善的框架的支柱

名称	描述
卓越运营	能够有效地支持发展和运行工作负载，获取对运营的洞察，以及不断改进支持流程和程序以实现业务价值。
安全性	安全性支柱包括保护数据、系统和资产以利用云技术来改善安全性的能力。
可靠性	工作负载按照预期正确并且一致地执行其预期功能的能力，包括在其整个生命周期内运营和测试工作负载的能力。
性能效率	高效利用计算资源来满足系统要求，并随着需求变化和技术演进保持这种效率的能力。
成本优化	以最低价格运行系统来交付业务价值的能力

在 AWS 架构完善的框架中，我们使用这些术语

- 组件是针对需求提供的代码、配置和 AWS 资源的组合。组件通常是技术处理单元，与其他组件分离。
- 我们使用术语工作负载来表示共同提供业务价值的组件集合。工作负载通常是业务和技术领导者沟通的细节层次。
- 随着架构在整个产品生命周期内（设计、测试、上线和生产）的演进，里程碑将记录架构中的关键变更。
- 我们将架构定义为组件在工作负载中协同工作的方式。架构图的重点通常是组件如何通信和交互。
- 组织内的技术组合是业务运营所需的工作负载集合。

在设计工作负载时，您会基于您的业务环境在各个支柱之间做出权衡。这些业务决策可以确定设计优先事项。您可以优化您的架构，例如在开发环境中以牺牲一部分可靠性来降低成本；而对于关键业务生产环境，您可以通过增加成本来提高可靠性。在电子商务解决方案中，性能可能会影响收入和客户的购买偏好。对于安全性和卓越运营，一般不会在它们和其他支柱之间进行权衡。

关于架构

在本地环境中，客户通常有一个技术架构中心团队，来监督其他产品或功能团队，从而确保他们遵循最佳实践。技术架构团队通常是由一组角色组成，比如技术架构师（基础设施）、解决方案架构师（软件）、数据架构师、网络架构师和安全架构师。这些团队通常将 TOGAF 或 Zachman 框架用作企业架构能力的一部分。

在 AWS，我们倾向于将能力分配到多个团队，而不是只让一个核心团队具有这种能力。当您选择分配决策权限时，会存在一定的风险，例如，确保团队达到内部标准。我们通过

[1]

两种方法降低这些风险。首先，我们有专注于让每个团队都具有这种能力的实践，并且我们通过设置专家组来确保团队不断提高他们需要满足的标准。第二，我们实施了各种机制[2]，自动执行检查，以确保满足各项标准[3]。这种分布式方法由Amazon 领导力原则支持，在所有角色中建立一种逆向工作的文化，只有以客户为中心的团队才能开发出真正满足客户需求的产品。

对于架构，这意味着我们希望每个团队都有能力创建架构并遵循最佳实践。为了帮助新团队获得这些能力或帮助现有团队提高其标准，我们创建了一个由首席工程师组成的虚拟社群，这些工程师可以检查现有团队的设计，帮助他们了解 AWS 最佳实践。首席工程师社群旨在让您能够接触和了解最佳实践。例如，通过午间谈话交流如何将最佳实践应用到实例中。这些谈话会被记录下来，用作新团队成员入门材料的一部分。

AWS 最佳实践源于我们在 Internet 规模上运行成千上万个系统的经验。我们倾向于使用数据定义最佳实践，同时我们还通过首席工程师等主题专家来设定最佳实践。当首席工程师发现新的最佳实践时，他们将以社群的形式确保所有团队遵循这些最佳实践。同时，这些最佳实践还会被正式纳入我们的内部审查流程以及强制性合规机制中。良好架构（Well-Architected) 是面向客户实施我们的内部审查流程，其中将我们在不同领域角色（例如解决方案架构和内部工程团队）中的主要设计思维编制成文。良好架构是一种可扩展的机制，使您能够有效利用现有的经验。

通过首席工程师在社群内分散架构责任的方法，我们相信设计良好的企业架构是由客户的需求驱动的，并且可以付诸实现。通过让技术主管（例如首席信息官或开发经理）针对所有工作负载执行良好架构审查，您能够更好地了解技术栈存在的风险。以此方法，您可以确定不同团队间可以使用的主题，通过机制、培训或午间谈话等方式，让首席工程师可以与多个团队分享他们在特定领域的想法。

[1]

[2] 行为方式、流程、标准和公认的规范。

[3] “徒有良好的心愿没有用，需要良好的机制来实现它们” Jeff Bezos。这意味着用机制（通常是自动的）来替代人为工作，检查是否遵守了规则或流程。

逆向工作是我们创新过程的基本组成部分。我们从客户和客户需求出发，定义和指导我们的工作。

一般设计原则

良好架构框架（Well-Architected Framework）定义了一系列一般性设计原则，以促进良好的云端设计：

- 停止猜测您的容量需求: 您不必再猜测基础设施容量需求。如果您在部署系统前作出容量决策，结果常常造成昂贵的资源闲置或因容量不足而影响性能。利用云计算，这些问题都不复存在。您可以按需使用容量，并自动对容量规模进行扩缩。
- 以生产规模进行系统测试: 在云中，您可以根据需要创建一套生产规模等级的测试环境，完成测试，然后停用资源。由于测试环境只需在运行时付费，您模拟真实环境的成本仅为本地测试成本的一小部分。
- 实现自动化，使架构试验变得更容易: 通过自动化操作，您可以低成本创建和复制系统，避免人力支出。您可以跟踪自动化变更，审核所产生的影响，并在必要时恢复到以前的参数。
- 支持实现架构演进: 支持不断演进的架构。在传统环境中，架构决策通常为静态的一次性事件，在其生命周期内包含几个重要的系统版本。随着业务及其环境继续发生变化，这些初始决策可能无法适应不断变化的业务能力需求。在云中，自动化和按需测试能力将显著降低设计变更带来的影响与风险。这使系统能够随时间推移不断演进，以便企业能够不断地发展创新。
- 利用数据驱动架构: 在云中，您可以收集有关您的架构选择如何影响工作负载表现的数据。这使您能够基于事实做出如何改进工作负载的决策。您的云基础设施以代码形式存在，因此您可以随着时间的推移，基于这些数据做出明智的架构选择和改进。
- 通过实际演练不断改进: 通过定期安排实际演练来模拟生产中的各种事件，测试架构和流程的性能。这将帮助您了解可以从哪些方面作出改进，并有助于培养组织处理各种事件的经验。

框架的五个支柱

构建软件系统与建楼很像。如果基础不牢固，结构问题将会破坏整栋大楼的完整性和功能。在设计技术解决方案时，如果忽视卓越运营、安全性、可靠性、性能效率和成本优化这五大支柱，就很难构建一个能够满足您的期望和需求的系统。通过把这些支柱整合到架构中，您将能构建稳定而高效的系统，这将使您能够专注于设计的其他方面，例如功能性需求。

卓越运营

卓越运营 支柱包括 能够有效地支持发展和运行工作负载，获取对运营的洞察，以及不断改进支持流程和程序以实现业务价值。

卓越运营支柱概述了各种设计原则、最佳实践和问题。如需有关具体实施的说明性指导，请参阅卓越运营支柱白皮书。

设计原则

云中的 卓越运营 有 五 项设计原则：

- 执行运营即代码: 在云中，您可以将用于应用程序代码的工程规范应用于整个环境。您可以将整个工作负载（应用程序、基础设施）定义为代码，并使用该代码进行更新。您可以将运营流程写成代码（脚本），并通过事件触发来自动执行这些脚本。通过以代码形式执行操作，您可以减少人为错误并实现对事件的一致响应。
- 频繁进行可逆的小规模更改: 将工作负载设计为支持组件定期更新。以较小增量进行失败时可逆的更改（尽可能不影响客户）。
- 经常改进运营流程: 在使用运营程序时，要寻找机会改进它们。在改进工作负载的同时，您也要适当改进一下流程。设置定期的实际演练，以检查并验证所有流程是否有效，以及团队是否熟悉这些流程。
- 预测故障: 执行“故障演练”，找出潜在的问题，以便消除和缓解问题。测试您的故障场景，并确认您了解相应影响。测试您的响应流程以确保它们有效，并确保团队能够熟练执行。设置定期的实际演练，以测试工作负载和团队对模拟事件的响应。
- 从所有运营故障中吸取经验教训: 从所有运营事件和故障中吸取的经验教训，推动改进。在多个团队乃至组织范围分享经验教训。

定义

云中的 卓越运营 有 四 个最佳实践领域：

- 组织

- 准备
- 运营
- 演进

您的组织领导层负责定义业务目标。您的组织必须了解各种要求和重点，并利用它们来组织和开展工作，从而为获得业务成果提供支持。您的工作负载必须发出所需信息以提供支持。采用多种服务来支持工作负载的集成、部署和交付，这将通过自动化重复流程，增加对生产的有益更改。

工作负载的运营可能存在固有风险。您必须了解这些风险并做出明智的生产决策。您的团队必须能够支持您的工作负载。从预期业务成果中得出的业务和运营指标将使您能够了解工作负载的运行状况、运营活动以及对事件的响应。您的重点将随着您的业务需求和业务环境的变化而变化。将这些作为反馈循环，持续推动组织和工作负载运营的改进。

最佳实践

组织

您的团队需要对整个工作负载、他们在其中的角色以及共同的业务目标有一致的理解，以便设置运营重点以实现业务成功。明确运营重点可以让您的工作效益最大化。评估内部和外部客户需求，让包括业务、开发和运营团队在内的主要利益相关方参与进来，以便确定工作重心。评估客户需求将确保您充分了解实现业务成果所需的支持。确保了解组织监管规定的指导原则或义务，以及监管合规性要求和行业标准等可能需要遵循或重视的外部因素。验证您是否具有确定内部监管和外部合规性要求更改的机制。如果未确定要求，请确保您已对此决定进行尽职调查。定期审查您的运营重点，以便在需求发生变化时对其进行更新。

评估业务面临的威胁（例如业务风险和负债以及信息安全威胁），并在风险注册表中维护这些信息。评估风险的影响，在有冲突的利益或替代方法之间做出权衡。例如，新功能的加速上市可能会比成本优化更重要，或者您可以为非关系数据选择关系数据库来简化系统迁移工作，而无需重构。管理收益和风险，以便在确定工作重心时做出明智的决策。有些风险或选择可能在一段时间内可以接受，这可能会降低相关风险，或者允许风险继续存在可能会令人无法接受，在这种情况下，您将采取措施来化解风险。

您的团队必须了解他们在实现业务成果方面所发挥的作用。团队需要了解自己在其他团队获得成功过程中所扮演的角色、其他团队在他们获得成功的过程中所扮演的角色，并设定共同的目标。了解责任分配、所有权归属、决策制定方式以及决策者将有助于集中精力，最大限度地发挥团队的优势。团队的需求将由其所支持的客户、所在组织、团队的组成以及工作负载的特征决定。期望单个运营模式能够支持组织中的所有团队及其工作负载是不合理的。

确保每个应用程序、工作负载、平台和基础设施组件都有确定的负责人，并且每个流程和程序都有确定的负责人负责其定义，有负责人负责其性能。了解每个组件、流程和程序

的业务价值，了解为什么要配置这些资源或为什么要执行这些活动，以及为什么要拥有该所有权，这些都有助于确定团队成员的行动。清晰定义团队成员的责任以便他们可以适当地采取行动，并制定相关机制，确定责任和所有权。制定用于请求添加、更改和例外的机制，以免限制创新。在团队之间定义协议，描述团队之间如何开展合作以相互支持以及您的业务成果。

为您的团队成员提供支持，以便他们可以更有效地采取行动并为您的业务成果提供支持。参与其中的高层领导应设定期望并衡量是否成功。他们应是采用最佳实践和组织发展的发起人、倡导者和推动者。授权团队成员在成果面临风险时采取行动以尽可能减少影响，并鼓励他们在认为存在风险时向决策者和利益相关者上报，以便解决问题并避免事故。及时、清晰、可行地传达已知风险和计划内事件，以便团队成员可以及时采取适当行动。

鼓励进行试验，以加快学习速度，并使团队成员保持兴趣和参与热情。团队必须增强自己的技能组合，以采用新技术，并随需求和责任的变化继续提供支持。专门安排学习时间，以提供支持并鼓励参与其中。确保您的团队成员拥有取得成功所需的资源（包括工具和团队成员），并具有支持您的业务成果的规模。利用跨组织的多样性来寻求多种独特的见解。利用这种见解提高创新能力、对您的假设提出质疑，并降低确认偏差的风险。在团队内部提升包容性、多样性和可达性有助于获取有益的见解。

如果存在适用于您组织的外部法规或合规性要求，则应使用 AWS 云合规性提供的资源来帮助培训您的团队，以便他们能够确定运营重点会受到的影响。架构完善的框架强调学习、衡量和改进。它为您提供了一种一致的方法来评估架构，并实施将随着时间推移而扩展的设计。AWS 提供了 AWS 架构完善的工具，可帮助您在开发之前查看方法、生产前的工作负载状态以及生产中的工作负载状态。您可以将其与最新的 AWS 架构最佳实践进行比较，监控工作负载的整体状态，并深入了解潜在风险。AWS Trusted Advisor 是一种工具，让您可以访问一组核心检查，这些检查会提出优化建议，帮助确定您的运营重点。商业支持和企业支持客户可以访问其他检查，这些检查重点关注安全性、可靠性、性能和成本优化，可进一步帮助他们帮助确定运营重点。

AWS 可以帮您向团队介绍 AWS 及其服务，让他们深入了解自己的选择会如何影响工作负载。您应该使用由 AWS Support（AWS 知识中心、AWS 开发论坛和 AWS Support 中心）和 AWS 文档提供的资源来培训您的团队。请通过 AWS Support 中心联系 AWS Support，获取与 AWS 问题有关的帮助。AWS 还分享了我们通过在 Amazon Builders' Library 中的 AWS 运营学到的最佳实践和模式。您可以通过 AWS 博客和 AWS 官方播客，获得各种其他有用信息。AWS Training and Certification 提供了一些免费培训，可以通过自定进度的数字课程，学习 AWS 的基础知识。您还可以注册讲师指导培训，进一步帮助培养您团队的 AWS 技能。

您应使用能够跨 AWS Organizations 等账户集中监管环境的工具或服务帮助管理运营模式。AWS Control Tower 等服务扩展了这一管理功能，使您能够定义账户设置的蓝图（支持您的运营模式），使用 AWS Organizations 进行持续监管以及自动预置新账户。托管服务提供商（如 AWS Managed Services）、AWS Managed Services 合作伙伴）或 AWS 合作伙伴网络中的托管服务提供商会提供实施云环境的专业知识，并为您的安全性和合规性要求以及业务目标提供支持。将托管服务添加到您的运营模式可以节省您的时间和资源，并使您的内部团队保持精干，专注于凸显业务优势的战略成果，而不是开发新的技能和功能。

以下问题主要针对 卓越运营 的准备阶段。 (有关 卓越运营 问题的列表和最佳实践，请参阅“附录”。).

OPS 1: 您如何确定自己的重点？
每个人都需要了解自己在业务成功中扮演的角色。设置共同的目标，以便为资源设定重点。这可以让您的工作效益最大化。
OPS 2: 如何构建组织结构来为业务成果提供支持？
您的团队必须了解他们在实现业务成果方面所发挥的作用。团队需要了解自己在其他团队获得成功过程中所扮演的角色、其他团队在他们获得成功的过程中所扮演的角色，并设定共同的目标。了解责任分配、所有权归属、决策制定方式以及决策者将有助于集中精力，最大限度地发挥团队的优势。
OPS 3: 组织文化如何为业务成果提供支持？
为您的团队成员提供支持，以便他们可以更有效地采取行动并为您的业务成果提供支持。

您可能会发现，您需要在某个时间点侧重于一小部分运营重点。长期使用平衡的方法来确保所需能力的发展和风险管理。定期回顾运营重点，并根据需求变化更新运营重点。当责任和所有权不确定或未知时，您将面临以下风险：没有及时执行必要的活动，以及在处理这些需求时可能出现工作冗余和潜在冲突。组织文化会直接影响团队成员的工作满意度和保留率。增强团队成员的参与度和能力，助力业务成功。创新必须进行试验，才能将创意转化为成果。应认识到，取得非预期结果也算试验成功，因为这种试验发现了无法实现成功的途径。

准备

要为卓越运营做好准备，您必须了解您的工作负载及其预期行为。然后，您需要能够针对它们进行设计，以提供对其状态的洞察并构建程序以提供支持。

将工作负载设计成能够提供必要的信息，以便您了解其所有组件的内部状态（例如指标、日志、事件和跟踪信息），为可观测性和调查问题提供支持。迭代开发必要的遥测技术，以监控工作负载的运行状况，确定结果何时面临风险并做出有效响应。在检测工作负载时，请捕获一组广泛的信息以启用情景感知（例如，状态变化、用户活动、特权访问和利用率计数器等的变更），因为您可以随时间变化筛选最有用的信息。

采用改进生产调整流程并支持重构、快速质量反馈和错误修复的方法。这些方法可以加快有益更改进入生产环境的速度、减少产生的问题，并能够快速识别和修复通过部署活动引入的问题或在环境中发现的问题。

采用提供快速质量反馈，并且若更改没有达到目标成效，则支持快速恢复的方法。使用这些实践可以减轻因部署更改而产生的问题的影响。制定计划以防更改不成功，这样在必要时能够更快速的响应，并测试和验证所做的更改。了解环境中的计划活动，以便可以管理更改风险，避免影响计划活动。强调频繁、小规模、可逆更改，以限制更改范围。这样可以简化故障排除工作、加快修复速度，并支持回滚更改。此外，还意味着能够更频繁地从有价值的更改中获益。

评估工作负载、流程和程序以及工作人员的运营准备就绪情况，以了解与工作负载相关的运营风险。您应该使用一致的流程（包括手动或自动化检查清单）来了解何时可运营工作负载或进行更改。这也使您能够发现需要制定计划予以解决的任何问题。准备好记录日常活动的运行手册和指导问题解决流程的行动手册。了解收益和风险，以便做出明智的决策，从而使更改应用到生产环境。

AWS 使您能够将整个工作负载（应用程序、基础设施、策略、监管和运营）视为代码。这些全部可以使用代码来定义和更新。这意味着，您可以将用于应用程序代码的工程规范应用于堆栈的每个元素，并在团队或组织之间共享，提高开发工作的效益。使用云中的运营即代码功能和安全测试功能开发工作负载、运营流程和故障演练。使用 AWS CloudFormation，您可以实现一致的模板化沙箱开发、测试和生产环境，提高运营管理水平。

以下问题主要针对 卓越运营 的准备阶段。

OPS 4: 如何设计工作负载以便自己了解其状态？
将工作负载设计成能够提供所有组件（例如指标、日志和跟踪信息）的必要信息，以便您了解其内部状态。这让您能够在适当的时候提供有效的响应。
OPS 5: 如何减少缺陷、简化修复和改进生产流程？
支持在生产时调整改进流程并支持重构、快速质量反馈和错误修复方法。这些方法可以加快有益更改进入生产环境的速度、减少产生的问题，并能够快速识别和修复通过部署活动引入的问题。
OPS 6: 您如何缓解部署风险？
采用提供快速质量反馈，并且若更改没有达到目标成效，则支持快速恢复的方法。使用这些实践可以减轻因部署更改而产生的问题的影响。
OPS 7: 如何知道您已经准备好支持某种工作负载？
评估工作负载、流程及程序和工作人员的操作准备就绪情况，以便了解与工作负载相关的操作风险。

对代码化运营进行投资，以最大限度地提高运营人员的工作效率，最大限度地降低错误率，并实现自动响应。使用“预先检验”来预测故障，并在适当的时候创建程序。使用资源标签和 AWS Resource Groups，按照一致的标记策略应用元数据，以标识您的资源。标记您的资源，以便进行整理、成本核算、访问控制并有针对性地自动执行操作活动。利用云的弹性特点结合相应部署实践，来推动开发活动和系统的预部署，以加快部署速度。当您对用于评估工作负载的检查清单进行更改时，请计划要对不再符合条件的活动系统执行哪些操作。

运营

工作负载运营是否成功通过业务成果和客户结果的实现情况加以衡量。定义预期结果、确定成功的衡量方式，并确定将在这些计算中使用的指标，以确定工作负载和运营是否成功。运营状况包括工作负载的运行状况，以及为支持工作负载而执行的操作的运行状况和成败（例如，部署和事件响应）。设立改进、调查和介入的指标基线，收集和分析您的指标，然后验证您对运营成功的理解及其随时间变化的规律。使用收集的指标确定您是否可以满足客户需求和业务需求，并确定需要改进的领域。

要实现卓越运营，您需要进行高效且有效的运营事件管理。这适用于计划内和计划外的运营事件。使用已确定的运行手册解决易于理解的事件，并使用行动手册来帮助调查和解决问题。您需要根据事件对业务和客户的影响排定其优先级。务必确保在出现事件警报时，会有指定负责人启动相关流程。事先定义解决事件所需的人员，并配备一个上报触发器，以便根据紧急程度和影响在必要时引入额外人员。确定并引入有权决定行动方案的人员，这些行动方案将对之前未解决的事件响应产生业务影响。

通过为目标受众（例如，客户、业务人员、开发人员、运营人员）定制的控制面板和通知来发布工作负载的运行状态，以便他们可以采取相应措施、管理预期，并在恢复正常运营时收到通知。

在 AWS 中，您可以为收集的工作负载指标和 AWS 自带指标生成控制面板视图。您可以利用 CloudWatch 或第三方应用程序来聚合和呈现运营活动的业务、工作负载和运营级别视图。AWS 通过日志记录功能（包括 AWS X-Ray、CloudWatch、CloudTrail 和 VPC 流日志）提供工作负载洞察，从而帮助识别工作负载问题，以支持根本原因分析和修复。

以下问题主要针对 卓越运营 的准备阶段。

OPS 8: 您如何了解工作负载的运行状况？
定义、记录和分析指标以便了解工作负载事件，从而采取适当的措施。
OPS 9: 您如何了解自己的运营状况？
定义、记录和分析运营指标以便了解运营事件，从而采取适当的措施。
OPS 10: 您如何应对工作负载事件和运营事件？
制定和验证用于响应事件的程序，以便尽可能减少其对工作负载的干扰。

您收集的所有指标都应该与业务需求及其支持的结果相符。为充分理解的事件开发脚本式响应，并自动执行响应以识别事件。

演进

必须学习、分享和不断改进，以保持卓越运营。专注于工作周期，以持续进行渐进式改进。对影响客户的所有事件执行事件后分析。确定导致这些事件的因素和预防措施，以限制或防止再次发生。根据需要与受影响的团体沟通导致这些事件的因素。定期评估并优先处理改进机会（例如，功能请求、问题修复和合规性要求），包括工作负载和运营程序。将反馈周期纳入您的流程，以快速确定需要改进的领域，并从运营执行中获取经验教训。

在团队中分享得到的经验教训，并从中受益。分析经验教训中的趋势，并对运营指标进行跨团队回顾性分析，以确定改进的机会和方法。实施改进措施，并评估结果以确定是否成功。

在 AWS 上，您可以将日志数据导出到 Amazon S3 或将日志直接发送到 Amazon S3 以便长期存储。使用 AWS Glue，您可以在 Amazon S3 中发现并准备您的日志数据以供分析，并将相关元数据存储在 AWS Glue 数据目录中。然后，Amazon Athena 通过与 Glue 的原生集成，可用于分析您的日志数据，并使用标准 SQL 进行查询。使用像 Amazon QuickSight 这样的商业智能工具，您可以直观显示、浏览和分析您的数据。发现可能推动改进的相关趋势和活动。

以下问题主要针对 卓越运营 的准备阶段。

OPS 11: 如何改进运营？ 分配专用的时间和资源用于持续增量改进，以便提高运营的有效性和效率。

运营的成功演进建立在以下基础上：频繁的小规模改进；提供安全的环境和时间来试验、开发和测试改进；以及鼓励人们从失败中获取经验教训的整体氛围。随着运营控制水平的提高，对于沙箱、开发、测试和生产环境的运营支持促进了开发，并提高了对生产环境中部署的变更结果成功与否的可预测性。

资源

请参阅以下资源，详细了解有关 卓越运营 的最佳实践。

文档

- DevOps and AWS

白皮书

- Operational Excellence Pillar

视频

- DevOps at Amazon

安全性

安全性 支柱包括 安全性支柱包括保护数据、系统和资产以利用云技术来改善安全性的能力。

安全性支柱概述了设计原则、最佳实践和问题。如需有关具体实施的说明性指导，请参阅安全性支柱白皮书。

设计原则

云中的 安全性 有 七 项设计原则：

- 健壮的身份验证体系: 实施最小权限原则，并通过对每一次与 AWS 资源之间的交互进行适当授权来强制执行职责分离。集中进行身份管理，并努力消除对长期静态凭证的依赖。
- 实现可追溯性: 实时监控和审计对环境执行的操作和更改并发送警报。为系统集成日志和指标收集功能，以自动调查并采取措施。
- 在所有层面应用安全措施: 利用多种安全控制措施实现深度防御。应用到所有层面（例如网络边缘、VPC、负载均衡、每个实例和计算服务、操作系统、应用程序和代码）。

- 自动实施安全最佳实践: 借助基于软件的自动化安全机制，您能够以更为快速且更具成本效益的方式实现安全扩展。创建安全架构，包括实施可在版本控制模板中以代码形式定义和管理的控制措施。
- 保护动态数据和静态数据: 将您的数据按敏感程度进行分类，并采用加密、令牌和访问控制等机制（如适用）。
- 限制对数据的访问: 使用相关机制和工具来减少和消除直接访问或人工处理数据的需求。这样可以降低处理敏感数据时数据处理不当、被修改以及人为错误的风险。
- 做好应对安全性事件的准备: 制定符合您组织要求的事件管理和调查策略和流程，做好应对事件的准备工作。开展事件响应模拟演练并使用具有自动化功能的工具来提高检测、调查和恢复的速度。

定义

云中的 安全性 有 六 个最佳实践领域：

- 安全性
- 身份识别与访问管理
- 检测
- 基础设施保护
- 数据保护
- 事件响应

在为任何工作负载设计架构之前，您需要确定可能影响安全性的实践。您需要控制谁可以执行什么操作。另外，您希望能够识别安全事件、保护您的系统和服务，并通过数据保护机制来保持数据的机密性和完整性。您应该具备一个定义明确且经过实践的流程来响应安全事件。这些工具和方法非常重要，因为它们有助于实现诸如避免财务损失或遵循法律与合规性要求等一系列目标。

借助 AWS 责任共担模式，组织能够利用云服务实现其安全性和合规性目标。AWS 负责保护用于支持云服务的基础设施，作为 AWS 的客户，您能够专注于使用云上的各种服务来实现您的目标。还可通过 AWS 云更好地访问安全数据，并以自动化方式响应安全性事件。

最佳实践

安全性

为了安全地操作您的工作负载，您必须对安全性的各个方面应用总体最佳实践。采用您在组织和工作负载层面的卓越运营中定义的要求和流程，并将它们应用到各个方面。

及时了解最新的 AWS、行业建议以及威胁情报信息可帮助您改进您的威胁模型和控制目标。实现安全流程、测试和验证的自动化可扩展您的安全运营。

以下问题主要针对 安全性 的准备阶段。 (有关 安全性 问题的列表和最佳实践，请参阅“附录”。).

SEC 1: 如何安全地操作您的工作负载？

为了安全地操作您的工作负载，您必须对安全性的各个方面应用总体最佳实践。采用您在组织和工作负载层面的卓越运营中定义的要求和流程，并将它们应用到各个方面。及时了解最新的 AWS、行业建议以及威胁情报信息可帮助您改进您的威胁模型和控制目标。实现安全流程、测试和验证的自动化可扩展您的安全运营。

在 AWS 中，建议根据账户的功能和合规性或数据敏感性要求分离不同的工作负载。

身份识别与访问管理

身份识别与访问管理是信息安全计划的关键部分，可以确保只有经过授权和通过身份验证的用户和组件才能访问您的资源，并且只能以您要求的方式进行访问。例如，您需要定义一些主体（即可以在您的账户中执行操作的账户、用户、角色和服务）、创建与这些主体相匹配的策略，并实施严格的凭证管理。这些权限管理元素构成了身份验证和授权的核心。

在 AWS 中，权限管理主要通过 AWS Identity and Access Management (IAM) 服务来实现，您可以使用该服务控制对 AWS 服务和资源的用户和编程访问。您需要应用细粒度的策略向用户、组、角色或资源分配权限。您还可以应用强密码原则（例如复杂程度）来避免重复使用并强制执行多重验证 (MFA)。您可以将联合身份验证与现有的目录服务配合使用。对于需要系统接入 AWS 的工作负载，IAM 可以通过角色、实例配置文件、联合身份和临时凭证进行安全访问。

以下问题主要针对 安全性 的准备阶段。

SEC 2: 如何管理人员和机器的身份？

在访问和运行安全的 AWS 工作负载时，您需要管理两种类型的身份。了解管理和授予访问权限所需的身份类型，这有助于确保正确的身份能够在正确的条件下访问正确的资源。人员身份：您的管理员、开发人员、操作员和最终用户需要确定身份才能访问您的 AWS 环境和应用程序。这些是您的组织成员或您与之协作的外部用户，以及通过 Web 浏览器、客户端应用程序或交互式命令行工具与您的 AWS 资源交互的用户。机器身份：您的服务应用程序、操作工具和工作负载需要一个身份来向 AWS 服务发出请求 - 例如，读取数据。这些身份包括在 AWS 环境中运行的机器，如 Amazon EC2 实例或 AWS Lambda 函数。您还可以管理需要访问权限的外部各方的机器身份。此外，您可能还有需要访问您 AWS 环境的 AWS 之外的机器。

SEC 3: 如何管理人员和机器的权限？

管理权限以控制对需要访问 AWS 和您的工作负载的人员和机器身份的访问。权限用于控制哪些人可以在什么条件下访问哪些内容。

凭证不得与任何用户或系统共享。应使用最小权限原则授予用户访问权限，并采用密码规则和强制执行 MFA 等最佳实践。应使用临时凭证和有限权限凭证（例如 AWS Security Token Service 发放的凭证）来执行程序访问（包括对 AWS 服务的 API 调用）。

AWS 提供了能够帮助您使用 Identity and Access Management 的资源。为了帮助学习最佳实践，请探索我们的管理凭证和身份验证、控制人员访问和控制程序访问的相关动手实验。

检测

您可以使用检测控制来识别潜在的安全威胁或事件。检测控制是管理框架的重要组成部分，并且可以用于支持质量流程、法律或合规，还可以用于威胁识别和响应工作。检测控制分为多种不同类型。例如，编制资产清单及其详细属性有助于更有效地做出决策（以及进行生命周期管理），从而有助于建立运营基准。您可以通过内部审计（是指对信息系统相关的控制措施进行的检查）来确保实践符合策略和要求，并确保您已根据定义的条件设置了正确的自动告警通知。这些控制措施都是重要的响应手段，可以帮助您的组织识别和了解异常活动的范围。

在 AWS 中，您可以通过处理可用于审计、自动化分析和触发警报的日志、事件以及监控来实施检测控制。CloudTrail 日志、AWS API 调用和 CloudWatch 可以提供对指标进行监控以及报警的功能，AWS Config 可以提供配置历史记录。Amazon GuardDuty 是一种托管的威胁检测服务，可以持续监控恶意或未经授权的行为，从而帮助您保护您的 AWS 账户和工作负载。您还可以使用服务级别日志，例如，您可以使用 Amazon Simple Storage Service (Amazon S3) 来记录访问请求。

以下问题主要针对 安全性 的准备阶段。

SEC 4: 您如何检测和调查安全事件？
通过日志和指标来记录和分析事件，以便了解信息。针对安全事件和潜在的威胁采取措施，以便保护您的工作负载。

日志管理对于架构完善的工作负载至关重要，这其中原因众多，包括安全性或取证、法律或法规要求。分析日志并相应地做出响应至关重要，以便您能够识别潜在的安全事件。借助 AWS 提供的功能，您能够定义数据保留生命周期或定义数据保存、存档或最终删除的位置，从而更轻松地管理日志。这样，您就能够以更为简单且更具成本效益的方式进行可预测且可靠的数据处理。

基础设施保护

基础设施保护包括满足最佳实践和组织、法律及监管义务所必需的控制方法（例如深度防御）。使用这些方法对于在云中或本地持续成功运营是至关重要的。

在 AWS 中，您可以通过使用 AWS 原生技术或使用 AWS Marketplace 提供的合作伙伴产品和服务来进行有状态和无状态数据包检查。您还可使用 Amazon Virtual Private Cloud (Amazon VPC) 创建一个安全且可扩展的私有环境，您可以在其中定义拓扑结构，包括网关、路由表以及公有子网和私有子网。

以下问题主要针对 安全性 的准备阶段。

SEC 5: 如何保护您的网络资源？
任何以某种形式连接至网络的工作负载（互联网或私有网络）都需要多层防御，以帮助防御基于外部和内部网络的威胁。
SEC 6: 如何保护计算资源？
工作负载内的计算资源需要采用多层防御，才有助于免受内部和外部威胁。计算资源包括 EC2 实例、容器、AWS Lambda 函数、数据库服务、IoT 设备等。

在任何类型的环境，我们都建议使用多层防御。在基础设施保护方面，许多概念和方法在跨云和本地模型中都有效。实施边界保护、监控入站点和出站点以及建立全面的日志记录、监控和告警机制对于制定有效的信息安全计划至关重要。

AWS 客户能够定制或加强 Amazon Elastic Compute Cloud (Amazon EC2)，Amazon EC2 Container Service (Amazon ECS) 容器或 AWS Elastic Beanstalk 实例的配置，并将配置捆绑到不可变的 Amazon 系统映像 (AMI)。之后，无论是由 Auto Scaling 触发还是手动启动，使用此 AMI 启动的所有新虚拟服务器（实例）都会收到上述加强的配置。

数据保护

在为任何系统设计架构之前，您应确定可能影响安全性的基本实践。例如，数据分级提供了一种基于敏感程度对组织数据进行分类的方法，加密通过让未经授权的用户无法获知数据的真正内容来保护数据。这些工具和方法非常重要，因为它们有助于实现诸如避免财务损失或遵循法律与合规性要求等一系列目标。

在 AWS 中，以下实践有助于保护数据：

- 作为 AWS 客户，您拥有对自己的数据的完全控制权。
- AWS 可帮助您更轻松地加密数据和管理密钥（包括定期密钥轮换），这些操作可以由 AWS 轻松自动执行，也可由您执行。
- 我们提供包含文件访问和更改等重要内容的详细日志记录。
- AWS 设计的存储系统具有优异的弹性。例如，Amazon S3 标准、S3 标准 – IA、S3 单区 – IA 和 Amazon Glacier 都设计为可以在一年内实现 99.999999999% 的对象持久性。这一持久性级别相当于平均每年仅有 0.000000001% 的数据对象可能会发生丢失。
- 作为较大规模数据生命周期管理流程中的一部分，版本控制可以防止意外的数据覆盖、删除和类似的数据损害。
- AWS 永远不会主动在区域之间移动数据。除非您明确启用相关功能或利用提供该功能的服务移动数据，否则放置在某个区域中的内容将保留在该区域中。

以下问题主要针对 安全性 的准备阶段。

SEC 7: 如何对数据进行分类？
分类提供了一种基于关键性和敏感度对数据进行分类的方法，以帮助您确定适当的保护和保留控制措施。
SEC 8: 如何保护静态数据？
通过实施多个控制措施来保护静态数据，以降低未经授权的访问或处理不当带来的风险。
SEC 9: 如何保护传输中的数据？
通过实施多个控制措施来保护传输中的数据，以降低未经授权的访问或数据丢失所带来的风险。

AWS 提供了多种加密静态数据和传输中的数据的方法。我们将这些功能内置在我们的服务中，这样您可以更轻松地加密数据。例如，我们为 Amazon S3 实施了服务器端加密 (SSE)，这样您可以更轻松地以加密的方式存储数据。您还可以将整个 HTTPS 加密和解密过程（通常称为 SSL 终端）交给 Elastic Load Balancing (ELB) 来完成。

事件响应

即使采用极为成熟的预防和检测控制机制，您的组织仍应制定相关流程来响应安全事件并缓解安全事件可能带来的影响。工作负载的架构会极大地影响团队在事件发生期间采取行动、隔离或约束系统并将运行状态恢复到已知的良好状态的能力。在安全事件发生之前确保相关工具部署到位，而后定期进行响应演练，将有助于确保您的架构有能力及时进行调查和恢复。

在 AWS 中，以下实践有助于提升安全事件响应效率：

- 采用详尽的日志记录方案，其中应包含文件访问与变更等重要内容。
- 事件可以自动处理，并且会触发通过使用 AWS API 自动做出响应的工具。
- 您可以使用 AWS CloudFormation 预先配置工具和一个“清洁屋”。这样您就可以在安全且隔离的环境中进行取证。

以下问题主要针对 安全性 的准备阶段。

SEC 10: 如何预测、响应事件以及从事件中恢复？
准备工作对于及时有效地调查、响应安全事件以及从安全事件中恢复至关重要，可以尽可能减少对组织的破坏。

确保您能够快速授予安全团队访问权限，而且系统可以自动隔离实例并自动捕捉数据与状态信息用于取证。

资源

请参阅以下资源，详细了解有关 安全性 的最佳实践。

文档

- AWS Cloud Security
- AWS Compliance
- AWS Security Blog

白皮书

- Security Pillar
- AWS Security Overview
- AWS Security Best Practices
- AWS Risk and Compliance

视频

- AWS Security State of the Union
- Shared Responsibility Overview

可靠性

可靠性 支柱包括 工作负载按照预期正确并且一致地执行其预期功能的能力，包括在其整个生命周期内运营和测试工作负载的能力。

可靠性支柱概述了设计原则、最佳实践和问题。如需有关具体实施的说明性指导，请参阅可靠性支柱白皮书。

设计原则

云中的 可靠性 有 五 项设计原则：

- 自动从故障中恢复: 通过监控工作负载的关键性能指标 (KPI)，您可以在指标超过阈值时触发自动化功能。这些 KPI 应该是对业务价值（而不是服务运营的技术方面）的一种度量。这包括自动发送故障通知和跟踪故障，以及启动解决或修复故障的自动恢复流程。借助更高级的自动化功能，您可以在故障发生之前预测和修复故障。
- 测试恢复过程: 在本地环境中，经常会通过执行测试来证明工作负载能够在特定场景中正常运作。通常不会利用测试来验证恢复策略。在云中，您可以测试工作负载的故障情况，并验证您的恢复程序。您可以采用自动化方式来模拟不同的故障，也可以重新建立之前导致故障的场景。此方式可以在实际的故障发生以前揭示您可以测试与修复的故障路径，从而降低风险。
- 横向扩展以提高聚合工作负载的可用性: 使用多个小型资源替换一个大型资源，以降低单个故障对整个工作负载的影响。跨多个较小的资源分配请求，以确保它们不共用常见故障点。

- 无需预估容量: 本地工作负载出现故障的常见原因是资源饱和，即对工作负载的需求超过该工作负载的容量（这通常是拒绝服务攻击的目标）。在云中，您可以监控需求和工作负载利用率，并自动添加或删除资源，以保持最佳水平来满足需求，而不会出现超额预置或预置不足的问题。虽然还有很多限制，但有些配额是可控的，其他配额也可以管理（请参阅“管理服务配额与限制”）。
- 变更管理自动化: 应利用自动化功能对基础设施进行更改。需要管理的变更包括，对自动化的变更，可对其进行跟踪与审查。

定义

云中的 可靠性 有 四 个最佳实践领域：

- 基础
- 工作负载架构
- 变更管理
- 故障管理

要实现可靠性，您必须从基础入手，而基础是服务配额和网络拓扑适应工作负载的环境。在设计时，分布式系统的工作负载架构必须能够预防与减少故障。工作负载必须处理需求或要求的变化，而且它的设计必须能够检测故障，并自动加以修复。

最佳实践

基础

基础要求是指其范围超出单个工作负载或项目的因素。在为任何系统设计架构之前，您应确定影响可靠性的基本要求。例如，您必须为数据中心提供足够的网络带宽。

在您使用 AWS 时，这些基础要求中的大部分已经包含在内，并且可以根据需要进行处理。云环境在设计层面拥有几乎无限的资源，因此 AWS 要负责满足对联网和计算容量的需求，让您可以根据需求随意更改资源大小和分配。

以下问题主要针对 可靠性 的准备阶段。 (有关 可靠性 问题的列表和最佳实践，请参阅“附录”。).

REL 1: 如何管理服务配额和限制？

基于云的工作负载架构存在服务配额（也被称作服务限制）。存在这些配额是为了防止意外预置超过您所需的资源，并对 API 操作的请求速率进行限制，以保护服务不会遭到滥用。还存在资源限制，例如，将比特推入光缆的速率，或物理磁盘上的存储量。

REL 2: 如何规划网络拓扑？

工作负载通常存在于多个环境中。其中包括多个云环境（可公开访问云和私有云），可能还包括现有数据中心基础设施。相关计划必须涵盖网络注意事项，如系统内部和系统间连接、公有 IP 地址管理、私有 IP 地址管理，以及域名解析。

基于云的工作负载架构存在服务配额（也被称作服务限制）。这些配额的存在目的在于，防止您意外预置超出必要量的资源，限制 API 操作的请求速率，从而避免服务遭到滥用。工作负载通常存在于多个环境中。您必须为所有工作负载环境监控和管理这些配额。其中包括多个云环境（可公开访问的云和私有云），可能还包括您的现有数据中心基础设施。相关计划必须涵盖网络注意事项，如系统内部和系统间连接、公有 IP 地址管理、私有 IP 地址管理以及域名解析。

工作负载架构

可靠的工作负载始于前期的软件和基础设施设计决策。您的架构选择将影响所有五个架构完善支柱的工作负载行为。针对可靠性，您必须遵循特定的模式。

使用 AWS 时，工作负载开发人员可以选择要使用的语言和技术。AWS 开发工具包通过为 AWS 服务提供特定于语言的 API，省去了复杂的编码过程。通过这些开发工具包，以及语言选择，开发人员可以实现此处列出的可靠性最佳实践。开发人员还可以访问 Amazon Builders' Library，阅读并了解 Amazon 构建和运营软件的方法。

以下问题主要针对 可靠性 的准备阶段。

REL 3: 如何设计工作负载服务架构？

使用面向服务的架构 (SOA) 或微服务架构构建高度可扩展的可靠工作负载。面向服务的架构 (SOA) 可通过服务接口使软件组件可重复使用。微服务架构则进一步让组件变得更小、更简单。

REL 4: 您如何在分布式系统中设计交互以预防发生故障？

分布式系统依赖于通信网络实现组件（例如服务器或服务）的互联。尽管这些网络中存在数据丢失或延迟，但是您的工作负载必须可靠运行。分布式系统组件的运行方式不得对其他组件或工作负载产生负面影响。这些最佳实践能够预防故障，并改善平均故障间隔时间 (MTBF)。

REL 5: 您如何在分布式系统中进行交互设计，从而缓解或经受住故障影响？

分布式系统依赖于通信网络以便使组件互相连接（如服务器或服务等）。尽管这些网络中存在数据丢失或延迟，但是您的工作负载必须可靠运行。分布式系统组件的运行方式不得对其他组件或工作负载产生负面影响。这些最佳实践使工作负载能够承受压力或故障，从中更快地恢复，并且降低此类伤害的影响。其结果是缩短平均恢复时间 (MTTR)。

分布式系统依赖于通信网络实现组件（例如服务器或服务）的互联。尽管这些网络中存在数据丢失或延迟，但是您的工作负载必须可靠运行。分布式系统组件的运行方式不得对其他组件或工作负载产生负面影响。

变更管理

必须提前为您的工作负载或其环境的更改做好准备，使其可以实现工作负载的可靠操作。此类更改包括，从外部施加到工作负载上的更改（如，需求高峰），以及内部更改（如功能部署和安全补丁）。

使用 AWS，您可以监控工作负载的行为并自动对 KPI 做出响应。例如，您的工作负载可以在某项工作负载用户增加时，添加更多服务器。您可以控制谁有权进行工作负载变更并审核这些变更的历史记录。

以下问题主要针对 可靠性 的准备阶段。

REL 6: 如何监控工作负载资源？

日志和指标是用于了解工作负载运行状况的强大工具。您可以配置工作负载以监控日志和指标，并在超出阈值或发生重大事件时发送通知。监控让您的工作负载可以发现超出低性能阈值和发生故障的情形，从而在响应中自动恢复。

REL 7: 您如何设计工作负载，以适应不断变化的需求？

可扩展工作负载具有自动添加或移除资源的弹性，因此确保在任何时间点都能准确满足当前的需求。

REL 8: 如何实施更改？

要部署新功能，必须对更改加以控制，以确保工作负载和操作环境正在运行已知的软件，并以可预测的方式进行修补和替换。如果此类更改不受控制，您将难以预测这些更改的影响，或难以处理由它们引发的问题。

当您构建工作负载来根据需求变化自动添加和删除资源时，这不仅可以提高可靠性，还可以确保业务成功不至于带来额外负担。借助既有的监控功能，当 KPI 偏离预期标准时，系统会自动向您的团队发送提醒。通过自动记录环境变更，您可以审核并快速识别可能影响可靠性的操作。对变更管理的控制确保您可以实施可提供所需的可靠性的规则。

故障管理

在任何具备一定复杂度的系统中，发生故障在意料之中。可靠性要求您的工作负载知晓故障的发生，并采取相应措施以避免对可用性产生影响。工作负载必须既能承受故障，又能自动解决问题。

使用 AWS，您可以发挥自动化优势对监控数据做出响应。例如，当特定指标超过阈值时，您可以触发自动操作来解决问题。此外，与其尝试诊断并修复作为生产环境一部分的失败资源，您可以将其替换为新的资源，并对被替换的旧有资源进行故障排查。由于云使您能够以低成本构建整个系统的临时版本，您可以使用自动化测试来验证完整的恢复流程。

以下问题主要针对 可靠性 的准备阶段。

REL 9: 如何备份数据？
备份数据、应用程序和配置，以满足恢复时间目标 (RTO) 和恢复点目标 (RPO) 的要求。
REL 10: 如何使用故障隔离来保护您的工作负载？
故障隔离边界可将一个工作负载内的故障影响限制于有限数量的组件。边界以外的组件不会受到故障的影响。使用多个故障隔离边界，您可以限制作用于您的工作负载的影响。
REL 11: 如何将您的工作负载设计为可承受组件故障的影响？
在设计具有高可用性和较短平均恢复时间 (MTTR) 要求的工作负载时必须考虑到弹性。
REL 12: 如何测试可靠性？
在为您的工作负载采用弹性设计以应对生产压力以后，测试是确保其按设计预期运行，并且提供您所预期弹性的唯一方式。
REL 13: 如何规划灾难恢复 (DR)？
拥有适当的备份和冗余工作负载组件是您的 DR 策略的开始。RTO 和 RPO 是您恢复可用性的目标。根据业务需求设置这些目标。通过实施策略来实现这些目标，同时考虑工作负载资源和数据的位置和功能。

请定期备份数据并测试备份文件，以确保您可以从逻辑和物理错误中恢复。管理故障的关键在于自动且频繁地测试工作负载以致其出现故障，然后观察它们如何恢复。请定期执行此操作，并确保在工作负载发生重大变更后也会触发此测试。主动跟踪 KPI（如恢复时间目标 (RTO) 和恢复点目标 (RPO)）以评估工作负载的弹性（特别是在故障测试场景中）。跟踪 KPI 将有助于您识别和减少单点故障。充分测试您的工作负载恢复流程，确保可以恢复所有数据并继续为您的客户提供服务，即使面对持续存在的问题也是如此。您的恢复流程应该与您的标准生产流程一样完备而有效。

资源

请参阅以下资源，详细了解有关 可靠性 的最佳实践。

文档

- AWS Documentation
- AWS Global Infrastructure
- AWS Auto Scaling: How Scaling Plans Work
- What Is AWS Backup?

白皮书

- Reliability Pillar: AWS Well-Architected
- Implementing Microservices on AWS

性能效率

性能效率 支柱包括 高效利用计算资源来满足系统要求，并随着需求变化和技术演进保持这种效率的能力。

性能效率支柱概述了设计原则、最佳实践和问题。如需有关具体实施的说明性指导，请参阅性能效率支柱白皮书。

设计原则

云中的 性能效率 有 五 项设计原则：

- 普及先进技术: 通过将复杂的任务委派给云供应商，降低您的团队实施高级技术的难度。与要求您的 IT 团队学习有关托管和运行新技术的知识相比，考虑将新技术作为服务使用是一种更好的选择。例如，NoSQL 数据库、媒体转码和机器学习都是需要专业知识才能使用的技术。在云中，这些技术会转变为团队可以使用的服务，让团队能够专注于产品开发，而不是资源预置和管理。
- 数分钟内实现全球化部署: 您可以在全球多个 AWS 区域中部署工作负载，从而以最低的成本为客户提供更低的延迟和更好的体验。
- 使用无服务器架构: 借助无服务器架构，您无需运行和维护物理服务器即可执行传统计算活动。例如，无服务器存储服务可以充当静态网站（从而无需再使用 Web 服务器），事件服务则可以实现代码托管。这不仅能够消除管理物理服务器产生的运行负担，还可以借由以云规模运行的托管服务来降低业务成本。
- 提升试验频率: 利用虚拟和可自动化的资源，您可以快速利用各种类型的实例、存储或配置执行各种测试。
- 考虑软硬件协同编程: 了解如何使用云服务，并始终使用最适合您工作负载目标的技术方法。例如，在选择数据库或存储方法时考虑数据访问模式。

定义

云中的 性能效率 有 四 个最佳实践领域：

- 选择
- 审核
- 监控
- 权衡

采用数据驱动型方法来构建高性能架构。收集架构各方面的数据，从总体设计到资源类型的选择与配置都包括在内。

定期审核您的选择，确保充分利用不断发展的 AWS 云的优势。监控可以确保您随时发现与预期性能的偏差。您可以对您的架构作出权衡以便提高性能，例如使用压缩或缓存，或放宽一致性要求。

最佳实践

选择

针对特定工作负载的最佳解决方案各不相同，而且解决方案通常会结合多种方法。架构完善的工作负载会使用多种解决方案，并且启用各种不同的功能来提高性能。

我们提供多种类型和配置的 AWS 资源，可让您更轻松地找到最能满足您工作负载需求的方法。此外，我们还提供了无法使用本地基础设施轻松实现的选项。例如，Amazon DynamoDB 之类的托管服务可以提供完全托管的 NoSQL 数据库，确保在任何规模下都只会有几毫秒的延迟。

以下问题主要针对 性能效率 的准备阶段。 (有关 性能效率 问题的列表和最佳实践，请参阅“附录”。).

PERF 1: 如何选择性能最好的架构？
一个工作负载通常需要采用多种方法才能实现最佳性能。架构完善的系统会使用多种解决方案和功能来提高性能。

使用数据驱动型方法来为您的架构选择模式和实施方式，获得经济高效的解决方案。AWS 解决方案架构师、AWS 参考架构和 AWS 合作伙伴网络 (APN) 合作伙伴可以根据自身的专业知识帮助您选择合适的架构，不过将需要通过基准测试或负载测试提取的数据来优化您的架构。

您的架构可能会结合多种不同的架构方法（例如事件驱动、ETL 或管道）。架构的实现将使用各种专门用于优化架构性能的 AWS 服务。在接下来的章节中，我们会介绍您应该考虑的四种主要资源类型：计算、存储、数据库和网络。

计算

选择满足您的要求和性能需求并具有出色成本效益的计算资源，将使您能够利用同等数量的资源获取更多收益。在评估计算选项时，请注意您的工作负载性能需求和成本要求，并以此做出明智的决策。

在 AWS 中，计算有三种形式：实例、容器和函数：

- 实例是虚拟化的服务器，因此您只需单击一个按钮或发起一次 API 调用即可更改其功能。因为云中的资源决策不是固定不变的，所以您可以尝试使用不同的服务器类型。在 AWS 中，这些虚拟服务器实例具有不同的系列和尺寸，并且可以提供各种功能，包括固态硬盘 (SSD) 和图形处理单元 (GPU)。

- 容器是实现操作系统虚拟化的一种方法，让您能够在资源隔离的进程中运行应用程序及其依赖项。AWS Fargate 是适用于容器的无服务器计算引擎。如果您需要控制计算环境的安装、配置和管理，则可以使用 Amazon EC2。此外，您还可以从多个容器编排平台中进行选择：Amazon Elastic Container Service (ECS) 或 Amazon Elastic Kubernetes Service (EKS)。
- 函数从您要执行的代码中抽象出执行环境。例如，AWS Lambda 让您可以在不运行实例的情况下执行代码。

以下问题主要针对 性能效率 的准备阶段。

PERF 2: 如何选择计算解决方案？

适合工作负载的最佳计算解决方案会根据应用程序设计、使用模式和配置设置而有所不同。架构可以使用不同的计算解决方案来支持各种组件，并且可以实现各种不同的功能来提高性能。为架构选择错误的计算解决方案可能会降低性能效率。

在设计如何使用计算资源时，您应该利用弹性机制来确保自己具有充足的容量，以便在需求发生变化时保持性能水平。

存储

云存储是云计算的关键组成部分，它存储着工作负载所使用的信息。云存储通常比传统的本地存储系统更加安全可靠且可扩展。从对象、数据块和文件存储服务以及您工作负载的云数据迁移选项中进行选择。

在 AWS 中，存储有三种形式：对象、数据块和文件：

- 对象存储提供了一个可扩展的耐用平台，允许从任何互联网位置访问数据，适用于用户生成的内容、活跃的存档、无服务器计算、大数据存储或备份，以及恢复。Amazon Simple Storage Service (Amazon S3) 是一种对象存储服务，提供行业领先的可扩展性、数据可用性、安全性和性能。Amazon S3 的耐用性可达到 99.999999999%（11 个 9），为全球各地的公司存储数百万个应用程序的数据。
- 数据块存储可为每个虚拟主机提供具有高可用性、一致性且低延迟的数据块存储，类似于直连式存储 (DAS) 或存储区域网络 (SAN)。Amazon Elastic Block Store (Amazon EBS) 旨在满足需要持久性存储的工作负载的需求，此类持久性存储可通过 EC2 实例访问，可帮助您根据适合的存储容量、性能和成本对应用程序进行微调。
- 文件存储可以跨多个系统提供对共享文件系统的访问。Amazon Elastic File System (EFS) 等文件存储解决方案非常适合大型内容存储库、开发环境、媒体存储或用户主目录等使用案例。Amazon FSx 让您可以轻松且经济高效地启动和运行热门文件系统，因此您可以利用应用广泛的开源和商业许可文件系统的丰富功能集和快速性能。

以下问题主要针对 性能效率 的准备阶段。

PERF 3: 如何选择存储解决方案？

针对特定系统的最佳存储解决方案往往取决于访问类型（块、文件或者对象存储）、访问模式（随机或者连续）、数据吞吐量要求、访问频率（在线、离线、归档）、更新频度（WORM、动态）以及可用性与持久性限制等因素。架构良好的系统使用多种解决方案，并且可以实现各种不同的功能来提高性能。

选择存储解决方案时，确保它与您的访问模式保持一致对于实现预期性能至关重要。

数据库

云可以提供专用数据库服务，解决您的工作负载所带来的各种问题。您可以从许多专用数据库引擎（包括关系、键值、文档、内存、图形、时间序列和分类账数据库）中进行选择。通过选择最佳数据库来解决特定问题或一组问题，您可以摆脱限制性的“一刀切”整体式数据库，并专注于构建应用程序以满足客户的性能需求。

在 AWS 中，您可以从多个专用数据库引擎（包括关系、键值、文档、内存、图形、时间序列和分类账数据库）中进行选择。有了 AWS 数据库，您再也无需担心数据库管理任务，例如数据库预置、修补、设置、配置、备份或恢复。AWS 会通过自修复存储和自动扩展功能持续监控您的集群，以使您的工作负载保持正常运行，这样您就可以专注于更高价值的应用程序开发。

以下问题主要针对 性能效率 的准备阶段。

PERF 4: 如何选择数据库解决方案？

针对特定系统的最优数据库解决方案取决于您的具体需求，包括可用性、一致性、分区容错性、延迟、持久性、可扩展性以及查询能力等等。许多系统会使用多种不同的数据库解决方案来满足其子系统的实际需要，并启用不同的功能来提高性能。为系统选择错误的数据库解决方案和功能可能会导致性能效率降低。

工作负载的数据库方法对性能效率具有重大影响。它通常是根据组织默认设置（而不是通过数据驱动型方法）选择的区域。如同考虑存储问题时一样，请务必要考虑工作负载的访问模式，另外还要考虑其他非数据库解决方案是否可以更高效地解决问题（例如图形、时间序列或内存存储数据库）。

网络

由于网络位于所有工作负载组件之间，因此可能会对工作负载性能和行为产生巨大的正面和负面影响。还有一些严重依赖网络性能的工作负载，例如，对于高性能计算 (HPC)，深入了解网络对于提高群集性能很重要。您必须确定带宽、延迟、抖动和吞吐量方面的工作负载要求。

在 AWS 中，网络资源以虚拟化形式存在，而且支持多种类型和配置。这让您可以更轻松地找到贴合您需求的网络方案。AWS 提供多种产品功能（例如增强联网、Amazon EBS

优化实例、Amazon S3 Transfer Acceleration 和动态 Amazon CloudFront）来优化网络流量。AWS 还提供多种联网功能（例如 Amazon Route 53 的基于延迟的路由、Amazon VPC 终端节点、AWS Direct Connect 和 AWS Global Accelerator）来减少网络距离或抖动。

以下问题主要针对 性能效率 的准备阶段。

PERF 5: 如何配置联网解决方案？

适合某个工作负载的最佳网络解决方案会因延迟、吞吐量要求、抖动和带宽而有所不同。物理限制（例如用户资源或本地资源）决定位置选项。这些限制可以通过边缘站点或资源置放来抵消。

您必须考虑部署网络的位置，您可以选择将资源放置在靠近使用地点的位置，以缩短距离。使用网络指标来随着工作负载的发展对网络配置进行更改。利用区域、置放群组和边缘服务，您可以显著提高性能。基于云的网络可以快速重建或修改，因此有必要随着时间的推移改进网络架构，以保持性能效率。

审核

云技术的发展日新月异，因此您必须确保工作负载组件使用的是最新的技术和方法，以持续提高性能。您必须不断评估工作负载组件并考虑对其进行更改，以确保您能够满足其性能和成本目标。机器学习和人工智能 (AI) 等新技术可以让您重塑客户体验，并对所有业务工作负载进行创新。

利用由客户需求驱动的 AWS 持续创新。我们会定期发布新的区域、边缘站点、服务和功能。这些发布内容都可以明显提高架构的性能效率。

以下问题主要针对 性能效率 的准备阶段。

PERF 6: 如何改进工作负载以便利用新的版本？

在最初构建解决方案时，您可能会从有限的方案选项中进行选择。但是随着时间的推移，可提升工作负载性能的新技术和方法会不断涌现。

通常，不存在或不完整的性能审核流程会导致架构性能不佳。如果您的架构性能不佳，请实施性能审核流程，以便应用戴明的计划-执行-检查-处理 (PDCA) 循环来驱动迭代改进。

监控

实施工作负载后，必须监控其性能，以便在问题对客户造成影响之前进行补救。您应该使用监控指标，确保系统在指标超出阈值时发出告警。

Amazon CloudWatch 是一项监控和可观测性服务，可为您提供相关数据和切实见解，以监控工作负载、响应系统范围的性能变化、优化资源利用率，并在统一视图中查看运行状况。CloudWatch 以日志、指标和事件的形式从在 AWS 和本地服务器上运行的工作负载中收集监控和运营数据。AWS X-Ray 可以帮助开发人员分析和调试分布式生产应用程序。借助 AWS X-Ray，您可以了解应用程序的执行情况，发现性能问题，并找到根本原因。使用这些分析结果，您可以快速做出反应，保证工作负载顺畅运行。

以下问题主要针对 性能效率 的准备阶段。

PERF 7: 如何监控资源以确保其性能？

系统性能会随着时间的推移而降低。监控系统性能，以发现性能降低的情况，并针对内部或外部因素（例如操作系统或应用程序负载）采取修复措施。

有效监控解决方案的关键是确保不会看到误报。自动触发器可以避免人为错误，并且可以缩短解决问题的用时。请安排时间在生产环境中执行模拟以测试告警解决方案，确保它可以正确识别各种问题。

权衡

在架构解决方案时，需要权衡各种因素才能确保获得最佳方案。根据具体情况，您可以在一致性、持久性和空间与时间或延迟之间进行权衡，以便实现更高的性能。

使用 AWS，您可以在几分钟内实现全球化部署，并可在世界范围内的多个位置部署资源，从而缩短与最终用户的距离。您还可以将只读副本动态添加到信息存储系统（例如数据库系统），以减少主数据库的负载。

以下问题主要针对 性能效率 的准备阶段。

PERF 8: 如何使用权衡机制来提高性能？

在构建解决方案时，确定权衡机制可以帮助您选出最佳方法。通常，您可以牺牲一致性、持久性和空间来换取缩短时间和延迟，从而提高性能。

对工作负载进行更改时，需要收集并评估各项指标，以确定更改产生的影响。衡量对系统和最终用户的影响，以便了解权衡机制如何影响工作负载。使用负载测试等系统的方法来确定权衡机制是否可以提高性能。

资源

请参阅以下资源，详细了解有关 性能效率 的最佳实践。

文档

- Amazon S3 Performance Optimization
- Amazon EBS Volume Performance

白皮书

- Performance Efficiency Pillar

视频

- AWS re:Invent 2019: Amazon EC2 foundations (CMP211-R2)

- AWS re:Invent 2019: Leadership session: Storage state of the union (STG201-L)
- AWS re:Invent 2019: Leadership session: AWS purpose-built databases (DAT209-L)
- AWS re:Invent 2019: Connectivity to AWS and hybrid AWS network architectures (NET317-R1)
- AWS re:Invent 2019: Powering next-gen Amazon EC2: Deep dive into the Nitro system (CMP303-R2)
- AWS re:Invent 2019: Scaling up to your first 10 million users (ARC211-R)

成本优化

成本优化 支柱包括 以最低价格运行系统来交付业务价值的能力

成本优化支柱概述了设计原则、最佳实践和问题。如需有关具体实施的说明性指导，请参阅成本优化支柱白皮书。

设计原则

云中的 成本优化 有 五 项设计原则：

- 践行云财务管理: 为获得财务上的成功并加速在云中实现业务价值，需要投资云财务管理/成本优化。您的组织需要投入时间和资源增强自身在这个新的技术和使用情况管理领域中的能力。与安全性或运营能力类似，您的组织需要通过知识构建、计划、资源和流程来培养能力，从而成为一家具有成本效益的组织。
- 采用消费模式: 仅为所需计算资源付费，并可根据业务需求而非复杂的预测增加或减少使用量。例如，开发和测试环境通常只需要在每个工作日运行八个小时。您可以在不需要时停用这些资源，从而实现 75% 的潜在成本节约（40 小时对比 168 小时）。
- 衡量整体效率: 衡量工作负载的业务产出及这些产出的实现成本。使用这种衡量方式了解您通过提高产出和降低成本获得的收益。
- 不再将资金投入到无差别的繁重任务上: AWS 会负责繁重的数据中心运营任务，例如安装、堆叠和驱动服务器。它还会消除使用托管服务管理操作系统和应用程序的运营负担。因此，您可以集中精力处理客户和业务项目而非 IT 基础设施。
- 对支出进行分析和归因: 使用云，您可以更轻松地确定系统的准确使用量和成本，从而将 IT 成本透明地分摊到各个工作负载拥有者。这有助于衡量投资回报率 (ROI)，并让工作负载拥有者能够据此优化资源和降低成本。

定义

云中的 成本优化 有 五 个最佳实践领域：

- 践行云财务管理
- 支出和使用情况意识
- 具有成本效益的资源
- 管理需求和供应资源
- 随着时间的推移不断优化

与架构完善的框架中的其他支柱一样，成本优化支柱也需要权衡各种因素，例如是优化上市速度还是优化成本。在某些情况下，最好优化上市速度以便快速上市、交付新功能或只是为了按时完成任务，而不是优化预付成本。设计决策有时是在仓促中而非是由数据决定的，并且人们总是倾向于过度补偿“以防万一”，而不是花时间进行基准测试以获得成本最优的部署。这可能会导致过度预置和优化不足的部署。但是，当您需要将资源从本地环境“直接迁移”到云，然后再进行优化时，这是一个合理的选择。通过预先在成本优化策略中投入适量的精力，您可以确保始终如一地遵守最佳实践，避免不必要的过度预置，从而更轻松地实现云的经济优势。以下部分介绍了一些技巧和最佳实践，可帮助您开始并持续实施工作负载的云财务管理和成本优化。

最佳实践

践行云财务管理

采用云后，由于缩短了审批、采购和基础设施部署周期，技术团队的创新速度会更快。要实现业务价值和财务成功，需要实施一种在云中管理财务的新方法。这种方法便是云财务管理，通过实施组织范围的知识构建、计划、资源和流程，在整个组织内培养能力。

许多组织由许多不同的单位构成，而这些单位又具有不同的要务。若能让组织遵循一组商定的财务目标并为组织提供实现这些目标的机制，将会打造一个更高效的组织。一个有能力的组织的创新和构建速度更快，更敏捷，并能够适应任何内部或外部因素。

在 AWS 中，您可以使用 Cost Explorer，也可以选择使用 Amazon Athena 和 Amazon QuickSight 查看成本和使用情况报告 (CUR)，从而了解整个组织的成本和使用情况。AWS 预算可主动发出成本和使用情况通知。AWS 博客提供有关新服务和新功能的信息，以确保您及时了解新发布的服务。

以下问题主要针对 成本优化 的准备阶段。 (有关 成本优化 问题的列表和最佳实践，请参阅“附录”。).

COST 1: 如何实施云财务管理？
实施云财务管理后，组织可以在 AWS 上优化成本和使用情况并进行扩展，从而实现业务价值和财务成功。

在组建成本优化部门时，可以考虑包括成员并为团队配备 CFM 和 CO 方面的专家。现有的团队成员将了解组织的当前运作方式以及如何快速实施改进。此外，还可以考虑配备拥有辅助或专业技能组合的人员，例如具备分析和项目管理能力的人员。

在组织中树立成本意识时，可以考虑改进现有计划和流程或基于现有计划和流程进行构建。与构建新流程和计划相比，向现有流程和计划增添内容要快得多。这样将更快地取得成果。

支出和使用情况意识

通过云，您可以获得更大的灵活性和敏捷性，从而支持创新以及快速的开发和部署。这样便节省了自建本地基础设施所需的人工环节和时间，包括确定硬件规格、协商报价、管理购买订单、安排发货和部署资源。然而，要实现这种易用性并利用近乎无限的按需容量，我们需要以新方式考虑支出。

很多企业有多个由不同团队运行的系统。将资源成本分摊到各个组织或产品拥有者可以推动更高效的资源使用模式，减少浪费。准确的成本分摊能够帮助您了解哪些产品是真正盈利的，让您能够做出更明智的预算分配决策。

在 AWS 中，您可以使用 AWS Organizations 或 AWS Control Tower 创建账户结构，这种方式不仅实现了分离，而且有助于分配成本和使用。此外，也可以通过资源标记操作在使用情况和成本中标注业务和组织信息。使用 AWS Cost Explorer 查看您的成本和使用情况，或者使用 Amazon Athena 和 Amazon QuickSight 创建自定义控制面板和分析。成本和使用情况控制通过 AWS 预算的通知来实现，并使用 AWS Identity and Access Management (IAM) 和服务配额进行控制。

以下问题主要针对 成本优化 的准备阶段。

COST 2: 您如何管理使用情况？
制定各种策略和机制，确保花费适当的成本来达到目标。采用制约与平衡方法，您可以在不超支的情况下进行创新。
COST 3: 如何监控使用情况和成本？
建立策略和程序以便监控并适当分配您的成本。这让您能够衡量和改进工作负载的成本效益。
COST 4: 您如何停用资源？
在从项目开始到结束的过程中实施变更控制和资源管理。这可以确保您关闭或终止未使用的资源，以便减少浪费。

您可以使用成本分配标签对 AWS 使用情况和成本进行分类并跟踪。当您对 AWS 资源（例如 EC2 实例或 S3 存储桶）应用标签后，AWS 将通过使用量和成本标签生成成本和使用量报告。您可以使用代表组织类别的标签（例如成本中心、工作负载名称或拥有者）整理您的多个服务的成本。

确保在成本和使用情况报告和监控中使用正确的详细级别和粒度。要获得大概见解和趋势，请在 AWS Cost Explorer 中使用每日粒度。要更深入地进行分析和检查，请在 AWS Cost Explorer 中使用每小时粒度，或者在 Amazon Athena 和 Amazon QuickSight 中以每小时为粒度查看成本和使用情况报告 (CUR)。

结合标记资源和实体生命周期跟踪（员工、项目），您可以确定无法再为组织创造价值而应停用的孤立资源或项目。您可以设置账单提醒，以在预计超支时通知您。

具有成本效益的资源

为工作负载使用合适的实例和资源是节约成本的关键。例如，在小型服务器上运行某个报告需要五个小时，而在另一个两倍成本的大型服务器上运行只需要一个小时。虽然两个服务器提供同样的结果，但小型服务器随着时间推移会产生更多成本。

良好架构的工作负载会使用最具有成本效益的资源，这样可以产生巨大而积极的经济效益。您还可以使用托管服务降低成本。例如，您可以使用按电子邮件收费的服务，而无需自己维护电子邮件服务器。

AWS 提供各种灵活且具有成本效益的定价选项，您可以从 EC2 和其他服务获取最符合您需求的实例。按需实例允许按小时支付计算容量的费用，且无需承诺最低用量。Savings Plans 和预留实例与按需定价相比最高可节约 75% 的成本。使用 Spot 实例，您可以利用未使用的 Amazon EC2 容量，并且与按需定价相比最高可节约 90% 的成本。Spot 实例适用于以下情况：系统可以容忍使用服务器队列，其中单个服务器可以动态装卸（例如无状态 Web 服务器）、批处理、高性能计算及大数据场景。

选择合适的服务还可以减少使用量和降低成本；例如，使用 CloudFront 可以最大限度地减少数据传输成本；例如，使用 Amazon Aurora on RDS 可以消除昂贵的数据库许可成本。

以下问题主要针对 成本优化 的准备阶段。

COST 5: 您在选择服务时如何评估成本？
Amazon EC2、Amazon EBS 和 Amazon S3 属于基础服务。托管服务（如 Amazon RDS 和 Amazon DynamoDB）属于更高级别或应用程序级别的 AWS 服务。通过选择适当的基础服务和托管服务，您可以优化工作负载，从而降低成本。例如，使用托管服务，您可以节省或消除大部分管理和运营开销，从而使您有精力从事应用程序和业务相关活动。
COST 6: 在选择资源类型、规模和数量时，如何实现成本目标？
确保选择适合当前任务的资源规模和资源数量。选择最经济实惠的资源类型、规模和数量可以尽可能减少浪费。
COST 7: 您如何使用定价模式来降低成本？
使用最适合的资源定价模式可以尽可能减少支出。
COST 8: 您如何规划数据传输费用？
务必要监控和规划您的数据传输费用，以便制定架构决策，尽可能降低成本。持续以小步迭代的方式进行架构优化可以实现运营成本的大幅降低。

通过在选择服务时考虑成本因素，并使用 Cost Explorer 和 AWS Trusted Advisor 等工具定期检查 AWS 使用情况，您可以主动监控利用率并相应地调整部署。

管理需求和供应资源

在您迁移到云时，您仅为所需内容付费。您可以在需要时供应与工作负载需求匹配的资源，从而消除昂贵且浪费的过度预置需求。还可以通过限流、缓冲区或队列来修改需求，

以满足需求并以更少的资源达成目标，从而降低成本，或者在以后使用批处理服务处理需求。

在 AWS 中，您可以自动预置资源来满足工作负载需求。通过使用基于需求或时间的方法进行自动扩展，您可根据需要添加和删除资源。如果您可以预测需求变化，便可以节省更多资金并确保资源与工作负载需求匹配。您可以使用 Amazon API Gateway 实施限流，也可以使用 Amazon SQS 在工作负载中实施队列。这两种方法都允许您修改工作负载组件的需求。

以下问题主要针对 成本优化 的准备阶段。

COST 9: 如何管理需求和供应资源？

为了工作负载的性能与支出实现平衡，请确保您支付过费用的所有资源都得到利用，并避免出现资源利用率过低的情况。无论是从运营成本（由于过度使用导致性能下降）还是从浪费 AWS 支出（由于超额配置）的角度衡量，利用率指标过高或过低都会对您的组织产生负面影响。

当进行修改需求和供应资源的设计时，请主动考虑资源使用模式、预置新资源所需要耗费的时间，以及需求模式的可预测性。当管理需求时，确保您具有大小正确的队列或缓冲区，并在所需的时间内响应工作负载需求。

随着时间的推移不断优化

AWS 不断发布新服务和功能，因此您最好不断审视现有架构决策，以便确保其始终最具成本效益。当您的需求发生变化时，请主动停用不再需要的资源、整体服务和系统。

实施新功能或资源类型可以逐步优化您的工作负载，同时最大程度地减少实施变更所需的工作量。这样可不断提高效率，并确保您始终使用最新的技术，从而降低运营成本。您还可以使用新服务替换或向工作负载中添加新组件。这可以显著提高效率，因此必须定期审查您的工作负载，并实施新服务和新功能。

以下问题主要针对 成本优化 的准备阶段。

COST 10: 如何评估新服务?

AWS 不断发布新服务和功能，因此您最好不断审视现有架构决策，以便确保其始终最具成本效益。

定期审查部署时，评估更新的服务如何帮助您节省成本。例如，Amazon Aurora on RDS 可以降低关系数据库的成本。使用无服务器（例如 Lambda）服务，无需操作和管理实例即可运行代码。

资源

请参阅以下资源，详细了解有关 成本优化 的最佳实践。

文档

- AWS Documentation

白皮书

- Cost Optimization Pillar

审查流程

需要持续不断对架构进行审查，同时要允许试错，建立良好的研究探索氛围。架构审查本身应该是一个简单流程（数小时，而不是几天），是一种对话，而不是审核。审查架构的目的是找出任何需要解决的关键问题或可以改进之处。审查后应采取一些措施，以改善客户体验。

正如在“关于架构”部分讨论的那样，团队中的每位成员都应该对架构质量负责。我们建议负责架构的团队利用良好架构框架持续检视架构，而不仅仅只是召开一个正式的审查会议。持续的审查使团队成员能够随着架构的演进不断获得知识体系与对架构认识的更新，并在您推出新功能时改进架构。

AWS 良好架构高度借鉴了 AWS 在内部审查系统和服务的方式，并与之保持一致。它基于一套可以影响架构方法的设计原则，并确保不会忽略常见于根因分析中的那些因素。当内部系统、AWS 产品或客户遇到严重问题时，我们会查看 RCA，寻求改进当前审查流程的可能性。

[1]
应在产品生命周期内的关键里程碑阶段，设计阶段早期，进行多次审查，避免单向决策很难进行更改，并在正式上线之前再次审查。上线后，随着时间变化，伴随着新功能和新技术的出现与发展，架构也会随之演进。您需要遵循良好的架构实践，避免出现架构退化。当架构面临重大变更时，您应遵循一套系统健康规范与流程，包括执行 Well-Architected 审查。

如果您想把审查用作一次性快照或独立的衡量方法，则需要确保让所有相关人员参与对话。我们经常发现，通过审查，团队才第一次真正了解他们实施了什么。在审查其他团队的工作负载时，一种有效的方法是围绕架构展开一系列非正式的对话，在此过程中您可以收集到大多数问题的答案。然后通过一两次会议进行跟进，来帮助您理清思路，或深入了解不明确的方面或已感知的风险。

下面建议了一些召开会议需要准备的事项：

- 配有白板的会议室
- 任何图表或设计说明的打印件
- 需要带外研究来获取答案的问题（例如，“是否已启用加密？”）

完成审查后，您应有一个问题清单，并基于您的业务环境来确定这些问题的优先级。您还需要考虑这些问题对团队日常工作的影响。如能及早解决这些问题，您就可以腾出时间开展创造业务价值的工作，而不是解决重复出现的问题。在解决问题时，您可以反复进行审查，来确认架构的改进效果。

虽然在完成审查后，审查的价值显而易见，但您可能发现新团队在开始时可能会对审查抱有抵触情绪。可以通过与团队沟通审查的益处来解决下列异议：

[1] 许多决策都是可逆的，是双向的。这些决策可以采用简单流程。单向决策很难或者不可逆，所以在做出决策之前需要更加全面的检查。

- “我们实在太忙了！”（一般会在团队准备重大发布时这么说。）
  - 如果您正在为重大发布做准备，您会希望一切进展顺利。审查能够帮助您发现您可能错过的任何问题。
  - 我们建议您在产品生命周期早期执行审查，以及时发现风险，并制定与功能交付路线图一致的规避计划。
- “我们已经没有时间了，结果已成定局！”（一般会在他们面临无法改变的事件 [比如超级碗] 时这么说。）
  - 这些事件是无法改变的。您真的想在不了解架构风险的情况下将它投入使用吗？ 即使不能解决所有问题，您仍然可以编制一个潜在问题处理手册。
- “We don’t want others to know the secrets of our solution implementation!”
  - 如果您向团队指出良好架构框架的问题，他们不会在这些问题中发现任何商业或技术信息。

在您与团队进行多次审核后，您可能会发现一些问题。例如，您可能发现一些团队在某个支柱或主题方面出现较多问题。建议您以全局眼光看待所有审查，找出能够帮助解决这些问题的任何机制、培训或首席工程师会谈方案。

总结

AWS 良好架构框架涵盖了五大支柱，强调了在云中设计和运行可靠、安全、高效且经济实惠的系统的架构最佳实践。该框架提供了一系列问题清单，来帮助您审查现有或将要实现的架构。它还为每个支柱提供了一组 AWS 最佳实践。在架构中应用该框架将能帮助您打造稳定且高效的系统，从而使您能够将主要精力集中在功能需求上。

贡献者

以下是对此文档做出贡献的个人和组织：

- Rodney Lester: Amazon Web ServicesWell-Architected 高级经理
- Brian Carlson: Amazon Web Services Well-Architected 操作主管
- Ben Potter: Amazon Web Services Well-Architected 安全主管
- Eric Pullen: Amazon Web Services Well-Architected 性能主管
- Seth Eliot: Amazon Web Services Well-Architected 可靠性主管
- Nathan Besh: Amazon Web Services Well-Architected 成本主管
- Jon Steele: Amazon Web Services 技术客户经理
- Ryan King: Amazon Web Services 技术项目经理
- Erin Rifkin: Amazon Web Services 高级产品经理
- Max Ramsay: Amazon Web Services 首席安全解决方案架构师
- Scott Paddock: Amazon Web Services 安全解决方案架构师
- Callum Hughes: Amazon Web Services 解决方案架构师

延伸阅读

AWS Cloud Compliance

AWS Well-Architected Partner program

AWS Well-Architected Tool

AWS Well-Architected homepage

Cost Optimization Pillar whitepaper

Operational Excellence Pillar whitepaper

Performance Efficiency Pillar whitepaper

Reliability Pillar whitepaper

Security Pillar whitepaper

The Amazon Builders' Library

文档修订

表 2.

重大修订：

日期	描述
2020 年 7 月	审核并重写大多数问题和答案。
2019 年 7 月	添加了 AWS Well-Architected Tool 和指向 AWS Well-Architected 实验室与 AWS Well-Architected 合作伙伴的链接，进行了小的修复，实现框架的多语言版本。
2018 年 11 月	审核并重写大多数问题和答案，确保问题一次集中在一个主题上。这导致某些之前的问题被拆分成多个问题。向定义中添加了常见术语（工作负载、组件等）。更改了正文中问题的表达以包含描述性文本。
2018 年 6 月	进行了更新，以简化问题文本、实现答案的标准化和提高可读性。
2017 年 11 月	将卓越运营移至第一个支柱，并进行重新编写，以此引出其他支柱。更新了其他支柱以反映 AWS 的发展。
2016 年 11 月	更新了框架，添加了卓越运营支柱，修订并更新了其他支柱，以减少重复并整合从成千上万的客户审查实践中吸取的经验。
2015 年 11 月	为附录更新了当前的 Amazon CloudWatch Logs 信息。
2015 年 10 月	原始版本：

附录：问题和最佳实践

卓越运营

组织

OPS 1 您如何确定自己的重点？

每个人都需要了解自己在业务成功中扮演的角色。设置共同的目标，以便为资源设定重点。这可以让您的工作效益最大化。

最佳实践:

- 评估外部客户需求: 让包括业务、开发和运营团队在内的主要利益相关方参与进来，以便确定怎样将工作重心放在外部客户的需求上。这可以确保您充分了解实现您期望的业务成果所需的运营支持。
- 评估内部客户需求: 让包括业务、开发和运营团队在内的主要利益相关方参与进来，以便确定怎样将工作重心放在内部客户的需求上。这可以确保您充分了解实现业务成果所需的运营支持。
- 评估监管要求: 确保您了解组织确定的指导方针或义务，它们可能会要求或强调特定的重点。评估内部因素，例如组织策略、标准和要求。验证您是否制定了相应的机制，来识别监管变化。如果未确定监管要求，请确保您已对此决定进行尽职调查。
- 评估合规性要求: 评估监管合规性要求和行业标准等外部因素，确保您了解自己可能需要遵循或重视的指导原则或义务。如果未确定合规性要求，请确保您已对此决定进行尽职调查。
- 评估威胁形势: 评估对业务的威胁（例如竞争、业务风险和负债、运营风险和信息安全威胁），并在风险注册表中维护当前信息。在确定工作重心时，将风险的影响考虑在内。
- 评估权衡: 在有冲突的利益或替代方法之间做出权衡并评估其影响，以便在确定工作重心或选择行动方案时做出明智的决策。例如，加快新功能上市的速度可能会比成本优化更重要，或者您可以为非关系数据选择关系数据库，以简化迁移系统的工作，而不是迁移到针对您的数据类型优化的数据库和更新您的应用程序。
- 管理收益和风险: 管理收益和风险，以便在确定工作重心时做出明智的决策。例如，为了向客户提供重要的新功能，部署仍存在未决问题的工作负载是可以接受的。这可能会降低相关风险，或者允许风险继续存在可能会令人无法接受，在这种情况下，您将采取措施来化解风险。

OPS 2 如何构建组织结构来为业务成果提供支持？

您的团队必须了解他们在实现业务成果方面所发挥的作用。团队需要了解自己在其他团队获得成功过程中所扮演的角色、其他团队在他们获得成功的过程中所扮演的角色，并设定共同的目标。了解责任分配、所有权归属、决策制定方式以及决策者将有助于集中精力，最大限度地发挥团队的优势。

最佳实践:

- 确定资源所有者: 了解对每个应用程序、工作负载、平台和基础设施组件拥有所有权的人员，各组件提供了哪些业务价值，以及为什么具有这种所有权。了解这些独立组件的业务价值以及它们如何支持业务成果将为它们应用的流程和程序提供信息。
- 确定流程和程序所有者: 了解谁对各个流程和程序的定义拥有所有权、为何使用这些特定的流程和程序，以及为什么存在这种所有权。了解使用特定流程和程序的原因将有助于发现改进机会。
- 确定对运营活动绩效负责的所有者: 了解谁负责针对定义的工作负载执行特定活动，以及为什么负责。了解谁负责执行活动可让我们知晓谁来开展活动、验证结果并向活动所有者提供反馈。
- 团队成员知道自己的责任: 了解您的角色具有哪些责任以及如何为业务成果做出贡献可帮助您确定任务的优先级以及自身角色的重要性。这使团队成员能够了解需求并做出适当响应。
- 制定用于确定责任和所有权的机制: 在未确定个人或团队时，要为有权分配所有权或计划满足该需求的人定义升级路径。
- 制定用于请求添加、更改和例外的机制: 您可以向流程、程序和资源的所有者提出请求。对收益和风险进行评估之后，做出明智的决定，批准可行的和确认合适的请求。
- 预先定义或协商团队间的职责: 团队之间具有定义或协商的协议，以说明团队之间的合作和相互支持方式（例如响应时间、服务级别目标或服务级别协议）。了解团队工作对业务成果的影响以及其他团队和组织的成果可以确定其任务的优先级，并帮助他们做出适当的响应。

OPS 3 组织文化如何为业务成果提供支持？

为您的团队成员提供支持，以便他们可以更有效地采取行动并为您的业务成果提供支持。

最佳实践:

- 高管支持: 高层领导明确为组织设定期望并评估是否成功。高层领导是采用最佳实践和组织发展的发起人、倡导者和推动者
- 赋能团队成员在结果有风险时采取行动: 工作负载所有者定义了指南和范围，赋能团队成员在结果有风险时做出响应。当事件超出定义的范围时，使用上报机制获取指示。
- 鼓励上报: 团队成员具有相应机制，如果他们认为结果存在风险，鼓励他们向决策者和利益相关者上报问题。应经常尽早上报，以便能够确定风险，并防止造成意外事件。
- 沟通及时、清晰、可行: 制定相应机制，用于将已知风险和计划内事件及时通知给团队成员。提供必要的相关信息、详细信息和时间（如果可能），为确定是否需要采取措施、需要采取什么措施以及及时采取措施提供支持。例如，提供软件漏洞通知可以加快修补过程；或者，提供计划内促销活动的通知可以实施变更冻结以避免发生服务中断的风险。
- 鼓励试验: 试验可加快学习速度，并使团队成员保持兴趣和参与热情。取得非预期结果也算试验成功，因为这种试验确定了无法实现成功的途径。团队成员不会因为取得非预期结果的成功试验而受到惩罚。创新必须进行试验，才能将创意转化为成果。
- 支持和鼓励团队成员保持和增强他们的技能组合: 团队必须增强自己的技能组合，以采用新技术；并随需求和职责的变化继续提供支持，以支持工作负载。新技术技能的增强通常能提升团队成员满意度并支持创新。支持您的团队成员获取和维护行业认证，以验证和认可他们不断增强的技能。进行交叉培训，以促进知识转移并降低在您失去熟练掌握机构知识、经验丰富的团队成员时产生重大影响的风险。专门安排时间进行学习。
- 为团队配置适当的资源: 培养团队成员的能力，并提供工具和资源来支持工作负载需求。团队成员超负荷工作会增加人为错误导致事故发生的风险。购买工具和资源（例如对频繁执行的活动实现自动化）可以提高团队的效率，让他们为其他活动提供支持。
- 鼓励在团队内部和团队之间提出不同的观点: 利用跨组织的多样性来寻求多种独特的见解。利用这种见解提高创新能力、对您的假设提出质疑，并降低确认偏差的风险。在团队内部提升包容性、多样性和可达性有助于获取有益的见解。

准备

OPS 4 如何设计工作负载以便自己了解其状态？

将工作负载设计成能够提供所有组件（例如指标、日志和跟踪信息）的必要信息，以便您了解其内部状态。这让您能够在适当的时候提供有效的响应。

最佳实践:

- 实施应用程序监控: 构建应用程序代码，使其能够提供其内部状态和业务成果实现情况的信息。例如队列深度、错误消息和响应时间。使用这些信息来确定需要在什么时候响应。
- 实施和配置工作负载遥测: 设计和配置工作负载，使其能够提供其内部状态和当前状态的信息。例如 API 调用量、HTTP 状态代码和扩展事件。使用这些信息帮助确定需要在什么时候响应。
- 实施用户活动遥测: 构建应用程序代码，使其能够发出关于用户活动的信息，例如点击流或者开始、放弃和完成的事务。使用这些信息来帮助了解应用程序的使用方式和使用量模式，并确定需要在什么时候响应。
- 实施依赖项遥测: 设计和配置工作负载，使其能够提供关于其依赖的资源状态（例如可访问性或响应时间）的信息。外部依赖项的示例可以包括外部数据库、DNS 和网络连接。使用这些信息来确定需要在什么时候响应。
- 实施事务跟踪: 实施应用程序代码并配置工作负载组件，提供关于工作负载之间的事务流的信息。使用这些信息来确定需要在什么时候做出响应，并帮助您确定导致问题的因素。

OPS 5 如何减少缺陷、简化修复和改进生产流程？

支持在生产时调整改进流程并支持重构、快速质量反馈和错误修复方法。这些方法可以加快有益更改进入生产环境的速度、减少产生的问题，并能够快速识别和修复通过部署活动引入的问题。

最佳实践:

- 使用版本控制: 使用版本控制来跟踪更改和发布。
- 测试并验证变更: 测试并验证变更以便发现并减少错误。实现自动测试以便减少手动过程引起的错误，并减少测试工作量。
- 使用配置管理系统: 使用配置管理系统来实现和跟踪配置更改。这些系统可以减少手动过程引起的错误，并减少部署更改的工作量。
- 使用构建和部署管理系统: 使用构建和部署管理系统。这些系统可以减少手动过程引起的错误，并减少部署更改的工作量。
- 执行补丁管理: 执行补丁管理以便实现功能、解决问题并保持监管合规性。实现自动补丁管理以便减少手动过程引起的错误，并减少修补工作量。
- 共享设计标准: 在不同团队间共享最佳实践，以便提高认识并最大程度地实现开发工作的效益。
- 实施提高代码质量的实践: 实施能够提高代码质量并尽可能减少缺陷的最佳实践。例如，测试驱动型开发、代码审查和标准采用。
- 使用多个环境: 使用多个环境来试验、开发和测试您的工作负载。当环境接近于生产环境时，逐步加强控制，以确保工作负载在部署后能够按预期运行。
- 频繁进行可逆的小规模更改: 频繁进行可逆的小规模变更可以减少变更的范围和影响。这可以简化故障排除、支持更快的修复，并提供回滚更改的选项。
- 完全自动化集成和部署: 实现自动构建、部署和测试工作负载。这可以减少手动过程引起的错误，并减少部署更改的工作量。

OPS 6 您如何缓解部署风险？

采用提供快速质量反馈，并且若更改没有达到目标成效，则支持快速恢复的方法。使用这些实践可以减轻因部署更改而产生的问题的影响。

最佳实践:

- 针对不成功的更改制定计划: 制定计划，以便在变更没有达到目标成效时在生产环境中恢复到已知良好状态，或者进行修复。做好充分的准备，以备快速响应，最大限度缩短回滚时间。
- 测试并验证变更: 在所有生命周期阶段测试更改并验证结果，以便确认新功能并尽可能减少部署失败的风险和影响。
- 使用部署管理系统: 使用部署管理系统来跟踪并实施更改。这可以减少手动过程引起的错误，并减少部署更改的工作量。
- 使用有限部署进行测试: 在全面部署之前使用现有系统和灰度发布进行测试，以便确认所需成果。例如使用 Canary 部署测试或一体化部署。
- 使用并行环境进行部署: 在并行环境中实施变更，然后过渡到新环境。保留之前的环境，直到确认部署成功为止。这样可以支持回滚到以前的环境，从而尽可能缩短恢复时间。
- 部署频繁、小规模、可逆的更改: 频繁进行可逆的小规模更改可以缩小变更的范围。这样可以简化故障排除工作、加快修复速度，并支持回滚更改。
- 完全自动化集成和部署: 实现自动构建、部署和测试工作负载。这可以减少手动过程引起的错误，并减少部署更改的工作量。
- 自动测试和回滚: 自动测试部署的环境以便确认目标效果。在没有达到预期时自动回滚到之前的已知良好状态，尽可能缩短恢复时间，并减少手动过程引起的错误。

OPS 7 如何知道您已经准备好支持某种工作负载？

评估工作负载、流程及程序和工作人员的操作准备就绪情况，以便了解与工作负载相关的操作风险。

最佳实践:

- 确保员工能力: 通过一种机制来验证您是否有适当数量技术娴熟的员工来提供对运营需求的支持。根据需要进行员工培训并调整人员产能，以便保持有效的支持。
- 确保以一致的方式对运营准备就绪情况进行审核: 确保您以一致的方式对运营工作负载的准备就绪情况进行审核。审核内容必须至少包括团队和工作负载的运营就绪情况，以及是否符合安全要求。以代码方式开展审核，针对事件触发自动审核，以便确保一致性、执行速度，并减少由手动流程引起的错误。
- 使用运维手册来执行程序: 运行手册是用来实现特定结果的书面程序。通过在运行手册中记录程序，实现对为人熟知的事件的一致且及时的响应。通过代码实施运行手册，并在适当情况下针对事件触发运行手册的执行，以便确保一致性、响应速度并减少由手动流程引起的错误。
- 使用行动手册调查问题: 通过在行动手册中记录调查流程，对不熟悉的问题做出一致而及时的响应。行动手册是在确定哪些因素导致故障场景时要执行的预定义步骤。所有流程步骤的结果都将用于确定要采取的后续步骤，直到问题得到确定或上报。
- 做出明智的决策来部署系统和更改: 评估团队支持工作负载的能力以及工作负载的监管合规性。在决定是否将系统或更改投入生产环境时，将这些与部署的收益进行比较。了解收益和风险，以便做出明智的决策。

运营

OPS 8 您如何了解工作负载的运行状况？

定义、记录和分析指标以便了解工作负载事件，从而采取适当的措施。

最佳实践:

- 识别关键性能指标: 根据期望的业务成果（例如，订单率、客户保留率和利润与运营开支）和客户成果（例如，客户满意度）识别识别关键性能指标 (KPI)。评估 KPI 以便确定工作负载是否成功。
- 定义工作负载指标: 定义工作负载指标来衡量 KPI（例如，放弃的购物车、下达的订单、成本、价格和分配的工作负载费用）的完成情况。定义工作负载指标以衡量工作负载的运行状况（例如，接口响应时间、错误率、提出的请求数、完成的请求数和利用率）。评估指标以便确定工作负载是否实现所需成果，并了解工作负载的运行状况。
- 收集和分析工作负载指标: 定期主动检查各种指标，以便发现趋势并确定哪里需要做出适当响应。
- 建立工作负载指标基准: 建立指标基准以便提供预期值，作为比较和识别性能不足和性能过剩组件的依据。确定改进、调查和干预的阈值。
- 了解工作负载的预期活动模式: 通过建立工作负载活动的模式来识别异常行为，以便您可以在需要时做出适当的响应。
- 在工作负载成果面临风险时发出提醒: 在工作负载成果面临风险时发出提醒，从而在必要时做出适当响应。
- 在检测到工作负载异常时发出提醒: 在检测到工作负载异常时发出提醒，从而在必要时做出适当响应。
- 验证实现的成果以及 KPI 和指标的有效性 : 在业务层面查看工作负载的运行情况，以便确定自己是否满足需求，并确定需要改进哪些方面才能实现业务目标。验证 KPI 和指标的有效性并在需要时进行修改。

OPS 9 您如何了解自己的运营状况？

定义、记录和分析运营指标以便了解运营事件，从而采取适当的措施。

最佳实践:

- 识别关键性能指标: 根据期望的业务成果（如交付新功能）和客户成果（如客户支持案例）识别识别关键性能指标 (KPI)。评估 KPI 以便确定运营是否成功。
- 定义运营指标: 定义运营指标以衡量 KPI 的实现情况（例如，成功的部署和失败的部署）。定义运营指标以衡量运营活动的运行状况（例如，事件的平均检测时间 (MTTD) 和事件的平均恢复时间 (MTTR)）。评估指标以便确定运营是否已实现期望的成果，并了解运营活动的运行状况。
- 收集和分析运营指标: 定期主动审核各种指标，以便发现趋势并确定哪里需要做出适当响应。
- 建立运营指标基准: 建立指标基准以便提供预期值，作为比较和识别运营活动执行不足和运营活动执行过度的依据。
- 了解运营的预期活动模式: 建立运营活动的模式来识别异常行为，以便您在必要时做出适当响应。
- 在运营成果面临风险时发出提醒: 在运营成果面临风险时发出提醒，从而在必要时做出适当响应。
- 在检测到运营异常时发出提醒: 在检测到运营异常时发出提醒，从而在必要时做出适当响应。
- 验证实现的成果以及 KPI 和指标的有效性 : 在业务层面查看运营活动，以便帮助您确定自己是否满足需求，并确定需要改进哪些方面才能实现业务目标。验证 KPI 和指标的有效性并在需要时进行修改。

OPS 10 您如何应对工作负载事件和运营事件？

制定和验证用于响应事件的程序，以便尽可能减少其对工作负载的干扰。

最佳实践:

- 使用流程来管理事件、意外事件和问题: 设置流程，用于处理发现的事件、需要干预的事件（意外事件）和需要干预并且要么会重复发生要么当前无法解决的事件（问题）。借助这些流程确保及时恰当的响应，以便减轻这些事件对业务和客户的影响。
- 针对每个提醒设置一个流程: 针对引发提醒的任何事件制定明确的响应措施（运维手册或管理手册），并明确指定负责人。这样可以确保您及时有效地响应运营事件，并防止可以针对其采取措施的事件被不重要的通知所掩盖。
- 根据业务影响确定运营事件的优先顺序: 确保在多个事件需要干预时，优先处理对业务最为重要的事件。举例来说，人身伤亡、经济损失、名誉或信任损害都是一种影响。
- 定义上报路径: 在运维手册和管理手册中定义上报路径，包括触发上报的事件和上报程序。明确指定每项措施的负责人，以便确保有效而及时地响应运营事件。
- 启用推送通知: 在用户使用的服务受到影响以及这些服务的运行状况再次恢复正常时，直接与用户联系（例如通过电子邮件或 SMS），确保用户采取适当的措施。
- 通过控制面板展现状况信息: 提供为目标受众（例如内部技术团队、领导和客户）专门设计的控制面板，以传达业务当前的运营状况并提供值得关注的指标。
- 自动响应事件: 自动响应事件以便减少由手动流程引起的错误，并确保响应及时并且一致。

演进

OPS 11 如何改进运营？

分配专用的时间和资源用于持续增量改进，以便提高运营的有效性和效率。

最佳实践:

- 设置持续改进流程: 定期评估各种改进机会并确定其优先顺序，以便集中精力处理可以实现最大收益的工作。
- 在意外事件发生后执行分析: 审核影响客户的事件，确定导致这些事件的因素和预防措施。利用这些信息来制定缓解措施，以限制或防止再次发生同类事件。制定程序以迅速有效地做出响应。根据目标受众，适当传达事件成因和纠正措施。
- 设置反馈环路: 在程序和工作负载中设置反馈环路，有助于发现问题和需要改进的方面。
- 执行知识管理: 执行机制，以方便您的团队成员及时发现和访问他们正在寻找的信息，并确定信息是最新且完整的。制定适当的机制，以确定所需的内容、需要更新的内容以及应存档的内容（以便不再引用它们）。
- 确定推动改进的因素: 确定推动改进的因素，以便评估各种机会并确定其优先顺序。
- 验证分析结果: 与跨职能团队和业务负责人共同查看分析结果和响应措施。通过这些工作来建立共识、发现其他影响并确定行动方案。适当调整响应措施。
- 审核运营指标: 定期与来自不同业务领域的跨团队参与者对运营指标进行回顾性分析。通过这些分析来确定改进机会和可能的行动方案，并分享经验教训。
- 记录和分享经验教训: 记录和分享在运营活动执行过程中获得的经验教训，以便在内部和不同团队中利用。
- 分配时间进行改进: 流程中专用的时间和资源可以实现持续增量改进。

安全性

安全性

SEC 1 如何安全地操作您的工作负载？

为了安全地操作您的工作负载，您必须对安全性的各个方面应用总体最佳实践。采用您在组织和工作负载层面的卓越运营中定义的要求和流程，并将它们应用到各个方面。及时了解最新的 AWS、行业建议以及威胁情报信息可帮助您改进您的威胁模型和控制目标。实现安全流程、测试和验证的自动化可扩展您的安全运营。

最佳实践:

- 使用账户分隔工作负载: 根据函数或一组通用控件，在单独的账户和组账户中整理工作负载，而不是镜像您公司的报告结构。从安全性和基础设施入手，随着工作负载的增长，使您的组织能够设置通用防护。
- 保护 AWS 账户: 安全访问您的账户，例如，启用 MFA 并限制根用户的使用，以及配置账户联系人。
- 识别并验证控制目标: 根据您的合规性要求以及从威胁模型中发现的风险，获得并验证您需要应用于工作负载的控制目标和控制措施。持续验证控制目标和控制措施可帮助您衡量风险缓解措施的有效性。
- 及时了解最新的安全威胁: 通过及时了解最新的安全威胁来识别攻击媒介，以帮助您定义并实施适当的控制措施。
- 及时了解最新的安全建议: 及时了解最新的 AWS 和行业安全建议，以改善您的工作负载安全状况。
- 在管道中自动执行测试和验证安全控制措施: 为安全机制建立可靠的基准和模板，并将其作为构建、管道和流程的一部分进行测试和验证。利用工具和自动化功能，持续测试并验证所有的安全控制措施。例如，将机器映像和基础设施等项目作为代码模板进行扫描，以发现安全漏洞、异常以及与每个阶段的既定基准的偏差。
- 使用威胁模型识别风险并确定其优先级: 使用威胁模型识别并维护一个最新的潜在威胁登记表。确定您的威胁优先级并调整您的安全控制措施，以进行防范、检测和响应。在不断变化的安全环境中，重新审视和维护此登记表。
- 定期评估和实施新的安全服务和功能: AWS 和 APN 合作伙伴不断推出各种新功能和新服务，以改善您的工作负载安全状况。

身份识别与访问管理

SEC 2 如何管理人员和机器的身份？

在访问和运行安全的 AWS 工作负载时，您需要管理两种类型的身份。了解管理和授予访问权限所需的身份类型，这有助于确保正确的身份能够在正确的条件下访问正确的资源。人员身份：您的管理员、开发人员、操作员和最终用户需要确定身份才能访问您的 AWS 环境和应用程序。这些是您的组织成员或您与之协作的外部用户，以及通过 Web 浏览器、客户端应用程序或交互式命令行工具与您的 AWS 资源交互的用户。机器身份：您的服务应用程序、操作工具和工作负载需要一个身份来向 AWS 服务发出请求 - 例如，读取数据。这些身份包括在 AWS 环境中运行的机器，如 Amazon EC2 实例或 AWS Lambda 函数。您还可以管理需要访问权限的外部各方的机器身份。此外，您可能还有需要访问您 AWS 环境的 AWS 之外的机器。

最佳实践:

- 使用强大的登录机制: 强制执行最小密码长度策略，并指导用户避免使用常见或重复使用过的密码。使用软件或硬件机制强制实施 Multi-Factor Authentication (MFA)，以提供额外保护。
- 使用临时凭证: 需要身份以动态获取临时凭证。对于员工身份，使用 AWS Single Sign-On 或与 IAM 角色联合访问 AWS 账户。对于机器身份，需要使用 IAM 角色，而不是长期访问密钥。
- 安全存储和使用密钥: 对于需要密钥（例如用于第三方应用程序的密码）的员工和机器身份，请使用最新的行业标准在专业服务中自动轮换存储它们。
- 依赖集中式身份提供商: 对于员工身份，依赖身份提供商，使您能够在集中位置管理身份。这使您能够从单个位置创建、管理和撤消访问，从而更轻松地管理访问。这降低了对多个凭证的需求，并提供了与 HR 流程集成的机会。
- 定期审核和轮换凭证: 当您无法依赖临时凭证并需要长期凭证时，请审核凭证，以确保实施了定义的控制措施（例如 MFA）、凭证定期轮换且处于适当的访问级别。
- 利用用户组和属性: 将具有常见安全要求的用户置于由您的身份提供商定义的组中，并建立机制以确保用于访问控制（例如，部门或位置）的用户属性正确无误且已更新。使用这些组和属性（而不是单个用户）来控制访问。这使您可以通过一次性更改用户的组成员身份或属性来集中管理访问，而不是在用户的访问需要更改时更新多个单独策略。

SEC 3 如何管理人员和机器的权限？

管理权限以控制对需要访问 AWS 和您的工作负载的人员和机器身份的访问。权限用于控制哪些人可以在什么条件下访问哪些内容。

最佳实践:

- 定义访问要求: 管理员、最终用户或其他组件都需要访问您工作负载的每个组件或资源。明确定义哪些人员或事物应当有权访问每个组件，选择用于进行身份验证和授权的适当身份类型和方法。
- 授予最低访问权限: 通过允许在特定条件下访问特定 AWS 资源上的特定操作，仅授予身份所需的访问权限。依靠组和身份属性来大规模动态设置权限，而不是为单个用户定义权限。例如，您可以允许一组开发人员访问，以便仅管理其项目的资源。这样，当开发人员从组中删除时，开发人员的访问权限将在使用该组进行访问控制的任何位置被撤消，而无需对访问策略进行任何更改。
- 建立紧急访问流程: 此流程允许在遇到不太可能发生的自动化流程或管道问题时紧急访问您的工作负载。这将帮助您依赖最低权限访问，但确保用户可以在需要时获得相应的访问级别。例如，为管理员建立一个验证和批准其请求的流程。
- 持续减少权限: 当团队和工作负载确定他们需要哪些访问权限时，删除他们不再使用的权限，并建立审核流程以实现最低权限。持续监控和减少未使用的身份和权限。
- 为您的组织定义权限防护: 建立通用控件以限制对组织中所有身份的访问权限。例如，您可以限制对特定 AWS 区域的访问，或防止操作员删除常见资源，例如用于您的核心安全团队的 IAM 角色。
- 基于生命周期管理访问权限: 将访问控制措施与操作员和应用程序生命周期以及您的集中式联合提供工具集成。例如，在用户离开组织或角色发生变化时删除用户的访问权限。
- 分析公共和跨账户访问: 持续监控突出公共和跨账户访问的发现。减少对仅需要此类访问的资源的公共访问和跨账户访问。
- 安全地共享资源: 管理跨账户或您的 AWS 组织内共享资源的占用。监控共享资源并查看共享资源访问。

检测

SEC 4 您如何检测和调查安全事件？

通过日志和指标来记录和分析事件，以便了解信息。针对安全事件和潜在的威胁采取措施，以便保护您的工作负载。

最佳实践:

- 配置服务和应用程序日志记录: 在整个工作负载中配置日志记录，包括应用程序日志、资源日志和 AWS 服务日志。例如，确保对组织中的所有账户启用 AWS CloudTrail、Amazon CloudWatch Logs、Amazon GuardDuty 和 AWS Security Hub。
- 集中分析日志、结果和指标: 应集中收集所有日志、指标和遥测数据，并自动对其进行分析，以检测异常和发现未经授权的活动。通过控制面板，您可以轻松访问实时运行状况见解。例如，确保 Amazon GuardDuty 和 Security Hub 日志已发送至一个集中的位置，以用于警报和分析。
- 自动响应事件: 使用自动化流程调查和修复事件可减少人工处理工作量和人为错误，从而扩展调查功能。定期审核将帮助您优化自动化工具，并实现持续迭代。例如，通过自动执行第一个调查步骤来自动响应 Amazon GuardDuty 事件，然后进行迭代以逐步消除人工。
- 实施可操作的安全事件: 创建发送给团队并将由团队处理的警报。确保警报包含团队采取措施所需的相关信息。例如，确保已将 Amazon GuardDuty 和 AWS Security Hub 警报发送至团队以便他们采取措施，或确保已将其发送至响应自动化工具，并通过从自动化框架发送消息来通知团队。

基础设施保护

SEC 5 如何保护您的网络资源？

任何以某种形式连接至网络的工作负载（互联网或私有网络）都需要多层防御，以帮助防御基于外部和内部网络的威胁。

最佳实践:

- 创建网络层: 将具有相同可访问性需求的组件分组为若干层。例如，应将 VPC 中无需进行互联网访问的数据库集群放在无法向/从互联网路由的子网中。在不使用 VPC 操作的无服务器工作负载中，使用微服务进行类似的分层和隔离可实现相同目的。
- 控制所有层的流量: 为入站和出站流量应用具有深度防御方法的控制措施。例如，对于 Amazon Virtual Private Cloud (VPC)，这通过安全组、网络 ACL 和子网来实现。对于 AWS Lambda，请考虑在私有 VPC 中运行时使用基于 VPC 的控制措施。
- 自动执行网络防护: 自动运行保护机制，以提供基于威胁情报和异常检测的自我防御网络。例如，可主动应对最新威胁并减轻其影响的入侵检测和预防工具。
- 实施检查和保护: 检查和筛选每层的流量。例如，使用 Web 应用程序防火墙可帮助防止应用程序网络层遭到意外访问。对于 Lambda 函数，第三方工具可将应用程序层防火墙添加到运行时环境中。

SEC 6 如何保护计算资源？

工作负载内的计算资源需要采用多层防御，才有助于免受内部和外部威胁。计算资源包括 EC2 实例、容器、AWS Lambda 函数、数据库服务、IoT 设备等。

最佳实践:

- 执行漏洞管理: 频繁扫描和修补您的代码、依赖项和基础设施中的漏洞，以帮助防御新的威胁。
- 缩小攻击面: 通过强化操作系统、尽量减少所使用的组件、库和外部可用的服务，以缩小您的攻击面。
- 采用托管服务: 实施用于托管资源的服务，例如 Amazon RDS、AWS Lambda 和 Amazon ECS，以便在责任共担模式中减少安全维护任务。
- 自动保护计算: 自动执行计算保护机制，包括管理漏洞、缩小攻击面和管理资源。
- 帮助人员远程执行操作: 移除交互式访问功能可降低人为错误的风险以及手动配置或管理的可能性。例如，利用变更管理工作流，借助基础设施即代码来部署 EC2 实例，然后使用工具来管理 EC2 实例，而不是允许直接访问或堡垒主机。
- 验证软件完整性: 实施一些机制（例如代码签名），以确保工作负载中使用的软件、代码和库来自可信的来源且未被篡改。

数据保护

SEC 7 如何对数据进行分类？

分类提供了一种基于关键性和敏感度对数据进行分类的方法，以帮助您确定适当的保护和保留控制措施。

最佳实践:

- 识别工作负载内的数据: 这包括数据的类型和分类、相关的业务流程、数据所有者、适用的法律和合规性要求、数据的存储位置以及为此需要实施的控制措施。这可能包括用于指明数据是可公开访问、仅供内部使用（例如客户的个人可识别信息 (PII)）还是受到更加严格的访问限制（例如知识产权、法律特权数据或敏感数据等等）的分类。
- 定义数据保护控制措施: 根据数据分类级别保护数据。例如，使用相关建议保护分类为公共的数据，同时使用其他控制措施保护敏感数据。
- 自动识别和分类: 自动识别和分类数据，以降低手动交互导致的人为错误的风险。
- 定义数据生命周期管理: 您定义的生命周期策略应基于敏感度级别以及法律和组织要求。应考虑您的数据保留期限、数据销毁、数据访问管理、数据转换和数据共享等方面。

SEC 8 如何保护静态数据？

通过实施多个控制措施来保护静态数据，以降低未经授权的访问或处理不当带来的风险。

最佳实践:

- 实施安全密钥管理: 必须安全地存储加密密钥，并实施严格的访问控制，例如使用 AWS KMS 等密钥管理服务。考虑使用不同的密钥和密钥访问控制，并结合 AWS IAM 和资源策略，以符合数据分类级别和隔离要求。
- 强制实施静态加密: 根据最新的标准和推荐强制实施加密要求，以保护您的静态数据。
- 自动执行静态数据保护: 利用自动化工具持续验证和实施静态数据保护，例如，确保只存在经过加密的存储资源。
- 强制实施访问控制: 强制实施包含最低权限和机制的访问控制，包括备份、隔离和版本控制，以帮助保护您的静态数据。防止操作员授予对您的数据的公共访问权限。
- 使用机制限制对数据的访问: 禁止所有用户直接访问正常运行环境中的敏感数据和系统。例如，提供一个控制面板而不是通过直接访问数据存储来执行查询。当未使用 CI/CD 管道时，确定需要利用哪些控制措施和流程来充分提供通常禁用的 Break Glass 访问机制。

SEC 9 如何保护传输中的数据？

通过实施多个控制措施来保护传输中的数据，以降低未经授权的访问或数据丢失所带来的风险。

最佳实践:

- 实施安全密钥和证书管理: 按照适当的时间间隔安全地存储和轮换加密密钥和证书，并应用严格的访问控制。例如使用 AWS Certificate Manager (ACM) 等证书管理服务。
- 执行传输中加密: 实施您根据适当的标准和建议定义的加密要求，以帮助您满足组织、法律和合规性要求。
- 自动检测意外数据访问: 使用 GuardDuty 等工具，自动根据数据分类级别检测尝试将数据移出所定义的边界的行为，例如，检测利用 DNS 协议将数据复制到未知或不可信的网络中的特洛伊木马程序。
- 对网络通信进行身份验证: 使用传输层安全性 (TLS) 或 IPsec 等支持身份验证的协议来验证通信的身份。

事件响应

SEC 10 如何预测、响应事件以及从事件中恢复？

准备工作对于及时有效地调查、响应安全事件以及从安全事件中恢复至关重要，可以尽可能减少对组织的破坏。

最佳实践:

- 确定关键人员和外部资源: 确定能够帮助您的组织响应事件的内部和外部人员、资源、以及法律义务。
- 制定事件管理计划: 制定计划，以帮助您响应事件、在事件期间进行沟通以及从事件中恢复。例如，您可以根据您的工作负载和组织中最可能遇到的情况开始制定事件响应计划。在计划中说明如何在内部和外部进行沟通和上报。
- 准备取证能力: 确定并准备适当的取证调查能力，包括外部专家、工具和自动化。
- 自动控制功能: 自动控制意外事件并从意外事件中恢复，以缩短响应时间和减少对组织的影响。
- 预置访问权限: 确保事件响应者将正确的访问权限预置到 AWS 中，以缩短调查到恢复的时间。
- 预先部署工具: 确保安全人员将适当的工具预先部署到 AWS 中，以缩短调查到恢复的时间。
- 执行实际演练: 定期执行事件响应实际演练（模拟）、将经验教训纳入事件管理计划，并持续改进。

可靠性

基础

REL 1 如何管理服务配额和限制？

基于云的工作负载架构存在服务配额（也被称作服务限制）。存在这些配额是为了防止意外预置超过您所需的资源，并对 API 操作的请求速率进行限制，以保护服务不会遭到滥用。还存在资源限制，例如，将比特推入光缆的速率，或物理磁盘上的存储量。

最佳实践:

- 了解服务配额和限制: 您要知道您的工作负载架构的默认配额和配额提高请求。您还要了解哪些资源限制（如磁盘或网络）可能会对您产生影响。
- 跨多个账户和区域管理服务配额: 如果您目前使用多个 AWS 账户或 AWS 区域，请确保在运行生产工作负载的所有环境中都请求适当的配额。
- 通过架构适应固定服务配额和限制: 请注意不可更改的服务配额和物理资源，并且在设计架构时要防止这些因素影响可靠性。
- 监控和管理配额: 评估您的可能使用情况，并适当提高您的配额，支持使用量按计划增长。
- 自动管理配额: 实施工具以便在接近阈值时向您发送提醒。通过使用 AWS Service Quotas API，您可以自动发出配额提高请求。
- 确保在当前配额与最大使用量之间存在足够的差距，以便应对故障转移: 当资源出现故障时，它可能仍会被计入配额，直到被成功终止。在出现故障的资源被终止之前，请确保您的配额涵盖所有出现故障的资源与其替换资源的叠加。在计算此差距时，应将可用区故障考虑在内。

REL 2 如何规划网络拓扑？

工作负载通常存在于多个环境中。其中包括多个云环境（可公开访问云和私有云），可能还包括现有数据中心基础设施。相关计划必须涵盖网络注意事项，如系统内部和系统间连接、公有 IP 地址管理、私有 IP 地址管理，以及域名解析。

最佳实践:

- 为工作负载公有终端节点使用高度可用的网络连接: 这些终端节点及其路由必须高度可用。为此，需使用高度可用的 DNS、内容分发网络 (CDN)、API Gateway、负载均衡或反向代理。
- 为云环境和本地环境之间的私有网络预置冗余连接: 在单独部署的私有网络之间使用多个 AWS Direct Connect (DX) 连接或 VPN 隧道。使用多个 DX 位置以实现高可用性。如果使用多个 AWS 区域，请确保其中至少有两个区域存在冗余。您可能想要评估终止 VPN 的 AWS Marketplace 设备。如果您使用 AWS Marketplace 设备，请在不同的可用区中部署冗余实例以实现高可用性。
- 确保 IP 子网分配考虑扩展和可用性: Amazon VPC IP 地址范围必须足够大，以满足工作负载的要求，包括考虑未来的扩展以及跨可用区为子网分配 IP 地址。这包括负载均衡器、EC2 实例和基于容器的应用程序。
- 轴辐式拓扑优先于多对多网格: 如果通过 VPC 对等连接、AWS Direct Connect 或 VPN 连接的网络地址空间超过两个（例如，VPC 和本地网络），则使用与 AWS Transit Gateway 所提供的模型类似的轴辐式模型。
- 在互相连接的所有私有地址空间中必须采用非重叠的私有 IP 地址范围: 多个 VPC 通过对等连接或 VPN 连接时，各个 VPC 的 IP 地址范围不得重叠。与之类似，您必须避免 VPC 和本地环境或与其他您使用的云提供商之间出现 IP 地址冲突。您还必须能够在需要时分配私有 IP 地址范围。

工作负载架构

REL 3 如何设计工作负载服务架构？

使用面向服务的架构 (SOA) 或微服务架构构建高度可扩展的可靠工作负载。面向服务的架构 (SOA) 可通过服务接口使软件组件可重复使用。微服务架构则进一步让组件变得更小、更简单。

最佳实践:

- 选择如何划分工作负载: 应避免使用整体式架构。您应该在 SOA 和微服务之中做选择。在做选择时，权衡优点和复杂性 – 适用于即将首次发布的新产品的功能有别于从一开始就构建用于扩展的工作负载的需求。使用较小分段的好处包括提高敏捷性、组织灵活性和可扩展性。复杂性则包括可能会增加延迟，调试变得更复杂，而且加重运营负担
- 构建专注于特定业务领域和功能的服务: SOA 采用由业务需求定义的划分明确的功能来构建服务。微服务则使用领域模型和有界上下文对此进行进一步限制，使每项服务都只用于一种用途。专注于特定功能让您可以区分不同服务的可靠性要求，并且更有针对性地锁定投资目标。简洁的业务问题和与每项服务关联的小型团队也简化了组织扩展。
- 根据 API 提供服务合同: 服务合同是团队之间关于服务集成的成文协议，它包括机器可读的 API 定义、速率限制和性能预期。版本控制策略让客户能够继续使用现有的 API，并在更新的 API 准备就绪时将他们的应用程序迁移到此类 API。只要遵守合同，即可随时进行部署。服务提供商团队可以使用自己选择的技术堆栈来满足 API 合同要求。同样，服务使用者可以使用自己的技术。

REL 4 您如何在分布式系统中设计交互以预防发生故障？

分布式系统依赖于通信网络实现组件（例如服务器或服务）的互联。尽管这些网络中存在数据丢失或延迟，但是您的工作负载必须可靠运行。分布式系统组件的运行方式不得对其他组件或工作负载产生负面影响。这些最佳实践能够预防故障，并改善平均故障间隔时间 (MTBF)。

最佳实践:

- 确定需要哪种类型的分布式系统: 硬性实时分布式系统需要同步并快速地做出响应，而软性实时系统有更宽松的以分钟计的时间窗口，或更多响应。离线系统会对响应进行批处理或异步处理。硬性实时分布式系统具有最严格的可靠性要求。
- 实施松散耦合的依赖关系: 队列系统、流系统、工作流和负载均衡器等依赖关系是松散耦合的。松散耦合有助于隔离来自其他依赖于它的组件的行为，从而提升弹性和敏捷性
- 使所有响应幂等: 幂等服务承诺每个请求只完成一次，因此发起多个相同请求与进行单个请求的效果相同。幂等服务使客户端可以轻松进行重试，而不必担心错误地处理多次。要执行此操作，客户端可以发出具有幂等性令牌的 API 请求，每当重复请求时都会使用同一令牌。幂等服务 API 使用令牌返回响应，该响应与首次完成请求时返回的响应相同。
- 持续工作: 系统会在负载中存在剧烈快速更改时失败。例如，监控着数千个服务器的运行状况检查系统每次都应发送相同大小的有效负载（当前状态的完整快照）。无论是否有服务器或有多少服务器发生故障，运行状况检查系统都会持续工作，而不会有剧烈、快速的变动。

REL 5 您如何在分布式系统中进行交互设计，从而缓解或经受住故障影响？

分布式系统依赖于通信网络以便使组件互相连接（如服务器或服务等）。尽管这些网络中存在数据丢失或延迟，但是您的工作负载必须可靠运行。分布式系统组件的运行方式不得对其他组件或工作负载产生负面影响。这些最佳实践使工作负载能够承受压力或故障，从中更快地恢复，并且降低此类伤害的影响。其结果是缩短平均恢复时间 (MTTR)。

最佳实践:

- 实现轻松降级以将适用的硬依赖关系转换为软依赖关系: 某个组件的依赖关系运行不正常时，该组件仍可在性能降低的条件下运行。例如，当依赖关系调用失败时，进行故障转移，使用预先确定的静态响应。
- 限制请求: 这是对按需求意外增加做出响应的缓解模式。部分请求会得到执行，但超出定义限制的请求会被拒绝，并返回说明它们已被限制的消息。客户端预期将会回退，并且放弃请求或以较低速率进行重试。
- 控制与限制重试调用: 在逐渐延长的间隔以后使用指数回退进行重试。引入抖动使此类重试间隔随机化，并限制重试的最大次数。
- 快速试错和限制队列: 如果工作负载无法成功响应请求，则快速试错。这样可释放与请求关联的资源，并允许该服务在资源不足的情况下恢复。如果工作负载能够成功响应，但请求速率过高，则使用队列来对请求进行缓冲。不过，不要允许使用长队列，它可能导致处理已被客户端放弃的过时请求。
- 设置客户端超时: 适当设置超时，对它们进行系统性验证，而且不要依赖默认值，因为它们通常设置得过高
- 尽可能使服务为无状态: 服务应该不需要状态或在不同的客户端请求之间卸载状态，在磁盘上或内存中本地存储的数据不存在依赖关系。从而支持任意替换服务器，而且不会对可用性产生影响。Amazon ElastiCache 或 Amazon DynamoDB 是卸载状态的理想目标位置。
- 实施紧急杠杆: 这些是可帮助您的工作负载减轻可用性影响的快速流程。即使未找到根本原因，它们也可以运行。理想的紧急杠杆可通过提供完全确定的激活与停用标准，将解析器的认知负担降低到零。杠杆示例包括，阻止所有的机器人流量或静态响应处理。杠杆通常需要手动操作，但也可实现自动化。

变更管理

REL 6 如何监控工作负载资源？

日志和指标是用于了解工作负载运行状况的强大工具。您可以配置工作负载以监控日志和指标，并在超出阈值或发生重大事件时发送通知。监控让您的工作负载可以发现超出低性能阈值和发生故障的情形，从而在响应中自动恢复。

最佳实践:

- 为工作负载监控全部组件（生成）: 使用 Amazon CloudWatch 或第三方工具监控工作负载组件。使用 Personal Health Dashboard 监控 AWS 服务
- 定义与计算指标（聚合）: 存储日志数据并在必要时应用筛选条件以计算指标，例如，特定日志事件的数量，或从日志事件时间戳计算得到的延迟
- 发送通知（实时处理和警报）: 发生重大事件时，需要知晓的组织会收到通知
- 自动响应（实时处理和警报）: 检测到事件后，利用自动化功能执行操作；例如，更换故障组件
- 存储与分析: 收集日志文件和指标历史，并对其进行分析以获得更广泛的趋势和工作负载见解
- 定期进行审核: 经常审核工作负载监控的实施情况，并根据重大事件和变更加以更新
- 对通过系统的请求的端到端跟踪进行监控: 利用 AWS X-Ray 或第三方工具，使开发人员可以更轻松地分析与调试分布式系统，理解他们的应用程序及其底层服务的表现

REL 7 您如何设计工作负载，以适应不断变化的需求？

可扩展工作负载具有自动添加或移除资源的弹性，因此确保在任何时间点都能准确满足当前的需求。

最佳实践:

- 在获取或扩展资源时利用自动化: 在替换被破坏的资源或扩展您的工作负载时，通过采用托管 AWS 服务（如 Amazon S3 和 AWS Auto Scaling）对流程进行自动化。您还可以使用第三方工具和 AWS 开发工具包自动扩展。
- 在检测到对工作负载的破坏时获取资源: 如果可用性受到影响，在必要时被动扩展资源，从而还原工作负载的可用性。
- 当检测到某个工作负载需要更多资源时，就会获取资源: 主动扩展资源以满足需求并避免影响可用性。
- 对工作负载进行负载测试: 采用负载测试方法来衡量扩展活动能否满足工作负载要求。

REL 8 如何实施更改？

要部署新功能，必须对更改加以控制，以确保工作负载和操作环境正在运行已知的软件，并以可预测的方式进行修补和替换。如果此类更改不受控制，您将难以预测这些更改的影响，或难以处理由它们引发的问题。

最佳实践:

- 对部署等标准活动使用运行手册: 运行手册是用于实现特定结果的预定义步骤。使用运行手册执行标准活动，无论这些活动是手动还是自动执行。例如部署工作负载，对其进行修补，或修改 DNS。
- 将功能测试作为部署的一部分进行集成: 功能测试作为自动化部署的一部分运行。若未满足成功条件，则相关管道会中止或回滚。
- 将弹性测试作为部署的一部分进行集成: 将弹性测试（混沌工程的一部分）作为预生产环境中自动化部署管道的一部分执行。
- 使用不可变基础设施部署: 这种模式要求在生产系统上不会就地出现更新、安全补丁或配置更改。需要更改时，会在新的基础设施上构建架构，并将其部署到生产环境中。
- 使用自动化功能部署更改: 自动部署与修补以消除负面影响。

故障管理

REL 9 如何备份数据？

备份数据、应用程序和配置，以满足恢复时间目标 (RTO) 和恢复点目标 (RPO) 的要求。

最佳实践:

- 识别和备份需要备份所有数据，或从源复制数据: Amazon S3 可用作多个数据源的备份目标位置。Amazon EBS、Amazon RDS 和 Amazon DynamoDB 等 AWS 服务具有创建备份的内置功能。此外，也可使用第三方备份软件。另外，如果可以从其他源中复制数据以满足 RPO 要求，您可能不需要进行备份
- 保护并加密备份: 通过身份验证和授权（例如 AWS IAM）检测访问，并使用加密检测数据完整性是否受损。
- 自动执行数据备份: 将备份配置为根据定期计划自动备份，或在数据集发生更改时自动备份。RDS 实例、EBS 卷、DynamoDB 表和 S3 对象均可配置为自动备份。您还可以使用 AWS Marketplace 解决方案或第三方解决方案。
- 定期执行数据恢复以验证备份完整性和流程: 通过执行恢复测试，验证您的备份流程实施是否满足恢复时间目标 (RTO) 和恢复点目标 (RPO) 要求。

REL 10 如何使用故障隔离来保护您的工作负载？

故障隔离边界可将一个工作负载内的故障影响限制于有限数量的组件。边界以外的组件不会受到故障的影响。使用多个故障隔离边界，您可以限制作用于您的工作负载的影响。

最佳实践:

- 将工作负载部署到多个位置: 将工作负载数据和资源分发到多个可用区，或在必要时分发到多个 AWS 区域。可通过选择不同位置满足各种需求。
- 组件的自动恢复受限于单个位置: 如果工作负载的组件只能在单个可用区或本地数据中心内运行，您必须利用相关功能在定义的恢复目标内彻底重建工作负载。
- 采用隔板架构: 类似于船上的隔板，此模式确保将故障限制在较小的请求/用户子集，受损的请求数量有限，因此大部分可以继续执行而不会受错误影响。数据的隔板经常被称作分区或分片，而服务的隔板称为单元格。

REL 11 如何将您的工作负载设计为可承受组件故障的影响？

在设计具有高可用性和较短平均恢复时间 (MTTR) 要求的工作负载时必须考虑到弹性。

最佳实践:

- 监控工作负载的所有组件以检测故障: 持续监控您的工作负载的运行状况，以便您和您的自动化系统在性能下降或发生全面故障时立即察觉。监控基于商业价值的关键性能指标 (KPI)。
- 故障转移至未受损位置内正常运行的资源: 确保在某位置发生故障时，来自正常运行位置的数据和资源可以继续用于处理请求。这对于多可用区工作负载而言更简单，因为 Elastic Load Balancing 和 AWS Auto Scaling 等 AWS 服务可帮助跨可用区分配负载。对于多区域工作负载就比较复杂。例如，跨区域只读副本让您可以将数据部署到多个 AWS 区域，但您仍必须提升只读副本至主节点，并在主要位置发生故障时将您的流量指向该节点。Amazon Route 53 和 AWS Global Accelerator 还可以帮助跨 AWS 区域路由流量。
- 自动修复所有层: 在检测到故障时，使用自动化功能执行修复操作。
- 使用静态稳定性来防止双模态行为: 双模态行为是指您的工作负载在正常和故障模式下展现出不同的行为，例如，若可用区发生故障时依赖于启动新的实例。您应该构建静态稳定的工作负载，并且仅在一个模式下运行。在这种情况下，如果删除了一个可用区，要在每个可用区内预置足够的实例来处理工作负载，然后再使用 Elastic Load Balancing 或 Amazon Route 53 运行状况检查将负载从受损实例中转出。
- 当事件影响可用性时发出通知: 在检测到重大事件时发送通知，即使由事件引发的问题已经自动解决。

REL 12 如何测试可靠性？

在为您的工作负载采用弹性设计以应对生产压力以后，测试是确保其按设计预期运行，并且提供您所预期弹性的唯一方式。

最佳实践:

- 根据行动手册调查故障: 通过在行动手册中记录调查流程，实现对并不十分了解的故障场景做出一致且及时的响应。行动手册是在确定哪些因素导致故障场景时要执行的预定义步骤。所有流程步骤的结果都将用于确定要采取的后续步骤，直到问题得到确定或上报。
- 在意外事件发生后执行分析: 审核影响客户的事件，确定这些事件的成因和预防措施。利用这些信息来制定缓解措施，以限制或防止再次发生同类事件。制定程序以迅速有效地做出响应。根据目标受众，适当传达事件成因和纠正措施。如果需要，可将这些原因告知他人。
- 测试功能要求: 这包括用于验证所需功能的单元测试和集成测试。
- 测试扩展和性能要求: 这包括负载测试以验证工作负载是否满足扩展和性能要求。
- 使用混沌工程测试弹性: 运行定期将故障注入到预生产和生产环境的测试。假设您的工作负载将会如何对故障做出反应，然后比较您的假设和测试结果，若不相符，重复执行。确保生产测试不会影响用户。
- 定期进行实际测试: 通过定期进行实际测试，与将参与实际故障情景的人员一起在尽可能接近生产环境的环境中（包括在生产环境中）练习您的故障程序。实际测试会强制执行相关措施，以确保生产测试不会影响用户。

REL 13 如何规划灾难恢复 (DR)？

拥有适当的备份和冗余工作负载组件是您的 DR 策略的开始。RTO 和 RPO 是您恢复可用性的目标。根据业务需求设置这些目标。通过实施策略来实现这些目标，同时考虑工作负载资源和数据的位置和功能。

最佳实践:

- 定义停机和数据丢失的恢复目标: 工作负载具有恢复时间目标 (RTO) 和恢复点目标 (RPO)。
- 使用定义的恢复策略来实现恢复目标: 定义了灾难恢复 (DR) 策略以满足目标。
- 测试灾难恢复实现以验证实现效果: 定期测试 DR 故障转移以确保满足 RTO 和 RPO 目标。
- 管理 DR 站点或区域的配置漂移: 确保 DR 站点或区域的基础设施、数据和配置满足需求。例如，检查 AMI 和服务配额是否为最新。
- 自动执行恢复: 利用 AWS 或第三方工具自动进行系统恢复，并将流量路由至 DR 站点或区域。

性能效率

选择

PERF 1 如何选择性能最好的架构？

一个工作负载通常需要采用多种方法才能实现最佳性能。架构完善的系统会使用多种解决方案和功能来提高性能。

最佳实践:

- 了解可用的服务和资源: 了解云中提供的各种服务和资源。识别与您的工作负载相关的服务和配置选项，并了解如何实现最佳的性能。
- 制定架构选择流程: 使用关于云的内部经验和知识，或外部资源（例如，已发布的使用案例、相关文档或白皮书），制定资源和服务选择流程。您应该制定一个流程，以鼓励对可能会用于工作负载的不同服务进行试验和基准测试。
- 在制定决策时考虑成本要求 : 工作负载通常具有运营成本要求。根据预测的资源需求，使用内部成本控制机制来选择资源类型和规模。
- 使用策略或参考架构: 通过评估内部策略和现有参考架构，以及使用分析为工作负载选择服务和配置，来最大程度提高性能和效率。
- 使用云提供商或相关合作伙伴提供的指南: 使用云公司提供的资源，例如，解决方案架构师、专业服务或适当的合作伙伴来指导您的决策。这些资源可帮助进行审核，并改进您的架构，从而实现最佳性能。
- 对现有工作负载进行基准测试: 对现有工作负载的性能进行基准测试，以了解工作负载在云上的运行情况。使用从基准测试中收集的数据来推动架构决策。
- 对工作负载进行负载测试: 使用不同的资源类型和大小在云上部署最新的工作负载架构。监控部署情况，捕获用于识别性能瓶颈或容量过剩的性能指标。使用此性能信息来设计或改进您的架构和资源选择。

PERF 2 如何选择计算解决方案？

适合工作负载的最佳计算解决方案会根据应用程序设计、使用模式和配置设置而有所不同。架构可以使用不同的计算解决方案来支持各种组件，并且可以实现各种不同的功能来提高性能。为架构选择错误的计算解决方案可能会降低性能效率。

最佳实践:

- 评估可用的计算方案: 了解您可以使用的与计算相关的方案的性能特性。了解实例、容器和函数的工作原理，以及它们对您的工作负载的有利影响和不利影响。
- 了解可用的计算配置选项: 了解各种选项如何补充您的工作负载，以及哪些配置选项最适合您的系统。这些选项的示例包括实例系列、规模、功能（GPU、I/O）、函数大小、容器实例、单租户和多租户。
- 收集与计算相关的指标: 了解计算系统性能的最佳方法之一是，记录和跟踪各种资源的真实利用率。此数据可用于更准确地确定资源需求。
- 通过合理调整大小来确定需要的配置: 分析您的工作负载的各种性能特性，以及这些特性与内存、网络和 CUP 使用率之间的关系。根据这些数据选择最适合您的工作负载配置文件的资源。例如，实例的 r 系列可以最好地处理内存密集型工作负载（例如数据库）。但是，弹性容器系统可为突增的工作负载提供更多优势。
- 利用资源的可用弹性: 云让您能够通过各种机制灵活地动态扩展或缩减资源，以便满足不断变化的需求。结合与计算相关的指标，工作负载可以自动响应这些变化并利用一系列最优的资源来实现其目标。
- 根据指标重新评估计算需求: 使用系统级指标来确定工作负载的行为和要求。通过比较可用资源和这些要求来评估工作负载的需求，并对计算环境进行更改以实现与您的工作负载配置文件的最佳匹配。例如，随着时间的推移，系统可能比最初认为的要更频繁地使用内存，所以转为使用其他实例系列或调整实例大小可能会提高性能和效率。

PERF 3 如何选择存储解决方案？

针对特定系统的最佳存储解决方案往往取决于访问类型（块、文件或者对象存储）、访问模式（随机或者连续）、数据吞吐量要求、访问频率（在线、离线、归档）、更新频度（WORM、动态）以及可用性与持久性限制等因素。架构良好的系统使用多种解决方案，并且可以实现各种不同的功能来提高性能。

最佳实践:

- 了解存储的特性和要求: 了解在选择最适合您的工作负载的各种服务（例如对象存储、数据块存储、文件存储或实例存储）时所需的不同特性（例如可共享能力、文件大小、缓存大小、访问模式、延迟、吞吐量和数据持久性）。
- 评估可用的配置选项: 评估各种特性和配置选项以及它们与存储的关系。了解在何处以及如何使用预置 IOPS、SSD、磁性存储、对象存储、存档存储或短暂存储来针对工作负载优化存储空间和性能。
- 根据访问模式和指标做出决策: 根据工作负载的访问模式选择存储系统，并通过确定工作负载访问数据的方式对其进行配置。通过选择对象存储而不是数据块存储来提高存储效率。按照您的数据访问模式，配置您选择的存储选项。

PERF 4 如何选择数据库解决方案？

针对特定系统的最优数据库解决方案取决于您的具体需求，包括可用性、一致性、分区容错性、延迟、持久性、可扩展性以及查询能力等等。许多系统会使用多种不同的数据库解决方案来满足其子系统的实际需要，并启用不同的功能来提高性能。为系统选择错误的数据库解决方案和功能可能会导致性能效率降低。

最佳实践:

- 了解数据特性: 了解工作负载中数据的不同特性。确定工作负载是否需要事务、工作负载如何与数据交互，以及工作负载的性能需求有哪些。使用这些数据来选择适用于工作负载的最佳数据库方法（例如关系数据库、NoSQL 键值、文档、宽列、图形、时间序列或内存中存储）。
- 评估可用的选项: 在选择工作负载存储机制的过程中，评估可用的服务和存储选项。了解如何以及何时使用给定的服务或系统进行数据存储。了解可以优化数据库性能或效率的可用配置选项，例如预置 IOPS、内存和计算资源以及缓存。
- 收集和记录数据库性能指标: 使用各种工具、库和系统来记录与数据库性能相关的性能测量值。例如，测量每秒事务数、慢速查询或访问数据库时引入的系统延迟。根据这些数据来了解您数据库系统的性能。
- 根据访问模式选择数据存储: 根据工作负载的访问模式来确定要使用的服务和技术。例如，对于需要事务的工作负载，使用关系数据库，或者使用能够提供更高吞吐量但能保持最终保持一致性的键值存储（如适用）。
- 根据访问模式和指标优化数据存储: 使用性能特性和访问模式来优化数据的存储和查询方式，以便实现最佳性能。衡量索引、键分配、数据仓库设计或缓存策略等优化对系统性能或整体效率的影响。

PERF 5 如何配置联网解决方案？

适合某个工作负载的最佳网络解决方案会因延迟、吞吐量要求、抖动和带宽而有所不同。物理限制（例如用户资源或本地资源）决定位置选项。这些限制可以通过边缘站点或资源置放来抵消。

最佳实践:

- 了解联网对性能的影响: 分析并了解与网络相关的决策对工作负载性能的影响。例如，网络延迟通常会影响用户体验，而使用错误的协议会产生过多的开销，进而导致网络容量不足。
- 评估可用的联网功能: 评估云中可能提高性能的联网功能。借助测试、指标和分析来衡量这些功能的影响。例如，利用可用的网络级功能来减少延迟、网络距离或抖动。
- 为混合工作负载选择适当大小的专用连接或 VPN: 需要进行本地通信时，请确保您有足够的带宽来满足工作负载性能要求。根据带宽要求，单个专用连接或单个 VPN 可能不够，您必须启用多个连接之间的流量负载均衡。
- 利用负载均衡和加密卸载: 跨多个资源或服务分配流量，以便让工作负载能够利用云提供的弹性。您也可以使用负载均衡机制来卸载加密终端，以便提高性能并有效管理和路由流量。
- 选择网络协议以提高性能: 根据对工作负载性能的影响，做出有关系统与网络之间的通信协议的决策。
- 根据网络要求选择工作负载的位置: 使用可用的云位置选项来降低网络延迟或提高吞吐量。利用 AWS 区域、可用区、置放群组和边缘站点（例如 Outposts、本地区域和 Wavelength），来降低网络延迟或提高吞吐量。
- 根据各项指标优化网络配置: 使用收集和分析的数据做出有关优化网络配置的明智决策。衡量更改带来的影响，并根据衡量结果来做出进一步决策。

审核

PERF 6 如何改进工作负载以便利用新的版本？

在最初构建解决方案时，您可能会从有限的方案选项中进行选择。但是随着时间的推移，可提升工作负载性能的新技术和方法会不断涌现。

最佳实践:

- 及时了解最新资源和服务: 当新的服务、设计模式或产品问世时，评估可以提高性能的方法。通过临时评估、内部讨论或外部分析来确定哪些方法可以提高工作负载的性能或效率。
- 制定流程来提高工作负载性能: 制定相应流程，以在新的服务、设计模式、资源类型和配置推出后，对它们进行评估。例如，对新实例产品运行现有性能测试，以确定它们改进工作负载的潜力。
- 随着时间的推移改进工作负载: 组织需要使用在评估流程中收集的信息，积极推动对新推出的服务或资源的采用。

监控

PERF 7 如何监控资源以确保其性能？

系统性能会随着时间的推移而降低。监控系统性能，以发现性能降低的情况，并针对内部或外部因素（例如操作系统或应用程序负载）采取修复措施。

最佳实践:

- 记录与性能相关的指标: 使用监控和可观察性服务来记录性能相关的指标。例如，记录数据库事务、慢查询、I/O 延迟、HTTP 请求吞吐量、服务延迟或其他关键数据。
- 在发生事件或意外事件时分析各项指标: 在某个事件或意外事件发生后（或发生过程中），使用监控控制面板或报告来了解和诊断影响。这些视图可让您了解工作负载哪些部分的性能没有达到预期。
- 建立关键性能指标 (KPI) 来衡量工作负载性能: 确定用于指示工作负载性能是否达到预期的 KPI。例如，基于 API 的工作负载可以使用整体响应延迟来指示整体性能，电子商务网站可以使用购买量作为其 KPI。
- 借助监控来生成基于告警的通知: 根据您定义的与性能相关的关键性能指标 (KPI)，使用当测量值超出预期范围时能够自动生成警报的监控系统。
- 定期检查指标: 在例行维护时，或者事件或意外事件发生后，检查收集到了哪些指标。通过这些检查，找出哪些指标对于解决问题至关重要，以及跟踪哪些其他指标会有助于发现、解决问题或预防问题发生。
- 主动监控和警报: 使用关键性能指标 (KPI) 并结合监控和警报系统，主动解决与性能相关的问题。使用警报触发自动操作，以便在可能的情况下修复问题。如果无法实现自动响应，则将告警上报给能够响应的人员。例如，您的系统在关键性能指标 (KPI) 超出特定阈值时，能够预测预期 KPI 值并发出警报；或者您的工具在 KPI 超出预期值时，能够自动停止或回滚部署。

权衡

PERF 8 如何使用权衡机制来提高性能？

在构建解决方案时，确定权衡机制可以帮助您选出最佳方法。通常，您可以牺牲一致性、持久性和空间来换取缩短时间和延迟，从而提高性能。

最佳实践:

- 了解对性能最至关重要的因素: 了解并确定在哪些方面提高工作负载性能，会对效率或客户体验产生积极的影响。例如，拥有大量客户交互的网站会因为使用边缘服务在距离客户更近的位置向客户分发内容而受益。
- 了解设计模式和服务: 研究和理解有助于提高工作负载性能的各种设计模式和服务。在分析的过程中，确定您需要牺牲哪些方面来获得更高的性能。例如，使用缓存服务有助于减少数据库系统上的负载；不过，这需要您完成一些设计工作，以实现安全的缓存，或者可能需要在某些方面实现最终一致性。
- 确定权衡机制对客户和效率的影响: 在评估与性能相关的改进时，确定哪些选择会对客户和工作负载效率产生影响。例如，如果使用键值数据存储可以提高系统性能，那么评估它的最终一致性将对客户的影响就非常重要。
- 衡量性能提高产生的影响: 在进行更改以提高性能时，对收集的指标和数据进行评估。使用这些信息来确定性能提高对工作负载、工作负载组件和客户的影响。这种衡量可让您了解采用权衡机制后实现的性能提高，还可以帮助确定性能提高是否产生了任何不利的副作用。
- 使用各种与性能相关的策略: 如果合适，利用多种策略来提高性能。例如，可以使用缓存数据等策略来防止出现过多的网络或数据库调用；使用数据库引擎的只读副本来提高读取速度；尽可能对数据进行分片或压缩以减少数据卷；在数据可用时进行缓冲和流式处理，避免拥堵。

成本优化

践行云财务管理

COST 1 如何实施云财务管理？

实施云财务管理后，组织可以在 AWS 上优化成本和使用情况并进行扩展，从而实现业务价值和财务成功。

最佳实践:

- 建立成本优化部门: 创建一个团队，负责在整个组织内建立并维护成本意识。该团队需要整个组织内担任财务、技术和业务角色的人员加入。
- 在财务和技术之间建立合作关系: 在云之旅的所有阶段，都让财务和技术团队参与成本和使用情况的讨论。团队定期开会，讨论组织目标、成本和使用情况的当前状态以及财务和会计实务等主题。
- 制定云预算和预测流程: 调整现有的组织预算和预测流程，使之适应云成本和使用情况的易变特性。流程必须是动态的，可以使用基于趋势或基于业务驱动因素的算法，也可以将两者结合使用。
- 在组织流程中落实成本意识: 将成本意识落实到影响使用的新流程或现有流程中，并利用现有流程提高成本意识。在员工培训中贯彻成本意识。
- 报告和通知成本优化: 配置 AWS 预算，以便对照目标提供成本和使用情况通知。定期举行会议来分析此工作负载的成本效益并推行成本意识文化。
- 主动监控成本: 利用工具和控制面板主动监控工作负载的成本。收到通知时，不要只查看费用和类别。这有助于发现正面趋势并在整个组织中推广。
- 及时了解新发布的服务: 定期咨询专家或 APN 合作伙伴，以便确定哪些服务和功能的成本更低。查看 AWS 博客和其他信息源。

支出和使用情况意识

COST 2 您如何管理使用情况？

制定各种策略和机制，确保花费适当的成本来达到目标。采用制约与平衡方法，您可以在不超支的情况下进行创新。

最佳实践:

- 根据组织的要求制定各种策略: 制定策略，规定您的组织应该如何管理资源。策略应该涵盖资源和工作负载的成本，包括在资源生命周期内创建、修改和停用。
- 制定方向性目标和执行性目标: 制定工作负载的成本和使用量目标。方向性目标为组织在成本和使用情况方面指明了方向，执行性目标则为工作负载提供了可衡量的结果。
- 实施账户结构: 实施与您的组织对应的账户结构。这有助于在整个组织内分摊和管理成本。
- 实施组和角色: 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。例如，实施开发组、测试组和生产组。这适用于 AWS 服务和第三方解决方案。
- 实施成本控制: 根据组织策略以及定义的组和角色来实施控制。这样可以确保成本只根据组织要求的规定产生，例如，使用 IAM 策略控制用户对区域或资源类型的访问。
- 跟踪项目生命周期: 跟踪、衡量并审计项目、团队和环境的生命周期，以避免使用不必要的资源并为此付费。

COST 3 如何监控使用情况和成本？

建立策略和程序以便监控并适当分配您的成本。这让您能够衡量和改进工作负载的成本效益。

最佳实践:

- 配置详细信息源: 将 AWS 成本和使用情况报告以及 Cost Explorer 配置为以每小时为粒度，以便提供详细的成本和使用情况信息。配置工作负载，使交付的每个业务成果都有日志条目。
- 确定成本归属类别: 确定可以用于在组织内分摊成本的组织类别。
- 建立组织指标: 建立此工作负载需要的组织指标。生成的客户报告或提供给客户的 Web 页面都属于工作负载指标。
- 配置账单和成本管理工具: 配置符合组织策略的 AWS Cost Explorer 和 AWS 预算。
- 在成本和使用情况中添加组织信息: 根据组织、工作负载属性和成本分摊类别来定义标记方案。在所有资源上应用标记。使用 Cost Categories，根据组织属性对成本和使用情况进行分组。
- 根据工作负载指标分配成本: 根据指标或业务成果分配工作负载的成本，以便衡量工作负载的成本效益。实施一个流程，使用 Amazon Athena 来分析 AWS 成本和使用情况报告，以便深入了解成本因素。

COST 4 您如何停用资源？

在从项目开始到结束的过程中实施变更控制和资源管理。这可以确保您关闭或终止未使用的资源，以便减少浪费。

最佳实践:

- 在资源生命周期内跟踪资源: 制定和实施一种方法，在资源生命周期内跟踪资源及其与系统的关联。您可以使用标记来标识资源的工作负载或功能。
- 实施停用流程: 实施一个流程来确定和停用孤立的资源。
- 停用资源: 停用由定期审计或使用情况发生变化等事件触发的资源。停用通常定期执行，可以手动停用，也可以自动停用。
- 自动停用资源: 设计您的工作负载，使其在您发现并停用非关键资源、不需要的资源或使用率低的资源时妥善处理资源的终止。

具有成本效益的资源

COST 5 您在选择服务时如何评估成本？

Amazon EC2、Amazon EBS 和 Amazon S3 属于基础服务。托管服务（如 Amazon RDS 和 Amazon DynamoDB）属于更高级别或应用程序级别的 AWS 服务。通过选择适当的基础服务和托管服务，您可以优化工作负载，从而降低成本。例如，使用托管服务，您可以节省或消除大部分管理和运营开销，从而使您有精力从事应用程序和业务相关活动。

最佳实践:

- 确定组织对成本的要求: 与团队成员合作，为此工作负载确定成本优化与其他支柱（例如性能和可靠性）之间的平衡。
- 分析此工作负载的所有组件: 确保分析工作负载的每个组件，无论当前大小或当前成本如何。审核工作应该体现出可能带来的好处，例如当前成本和预期成本。
- 对每个组件进行彻底分析: 分析组织为每个组件付出的总体成本。通过考虑运营和管理成本（尤其是使用托管服务时）来分析总拥有成本。审核工作应该体现出可能带来的好处，例如用于分析的时间与组件成本成正比。
- 选择具有经济实惠的许可的软件: 开源软件无需软件许可成本，从而大大节省了工作负载的成本。如果需要许可软件，应避免使用绑定到任意属性（如 CPU）的许可证，而应使用绑定到输出或结果的许可证。这些许可证的成本与所提供的效益更为相当。
- 选择此工作负载的组件，以便根据组织的优先事项优化成本: 在选择所有组件时考虑成本因素。这包括使用 Amazon RDS、Amazon DynamoDB、Amazon SNS 和 Amazon SES 等应用程序级别的托管服务降低组织的总体成本。使用无服务器服务和容器进行计算，例如 AWS Lambda、用于静态网站的 Amazon S3，以及 Amazon ECS。使用开源软件或不收取许可证费用的软件，尽可能减少许可证成本：例如，对计算工作负载使用 Amazon Linux 或将数据库迁移到 Amazon Aurora。
- 对不同时间的不同使用情况执行成本分析: 工作负载可能会随时间而变化。某些服务或功能在不同的使用水平下更具成本效益。随着时间的变化根据每个组件的预期使用情况执行分析，您可以确保工作负载在其生命周期内具有成本效益。

COST 6 在选择资源类型、规模和数量时，如何实现成本目标？

确保选择适合当前任务的资源规模和资源数量。选择最经济实惠的资源类型、规模和数量可以尽可能减少浪费。

最佳实践:

- 执行成本建模: 确定组织要求，并对工作负载及其每个组件执行成本建模。对不同预计负载下的工作负载执行基准测试活动，并比较成本。建模工作应该反映出可能带来的好处，例如花费的时间与组件成本成正比。
- 根据数据选择资源类型和规模: 根据工作负载和资源特征相关数据选择资源规模或类型，例如计算、内存、吞吐量或写入密集型资源。通常使用工作负载的上一个版本（例如本地版本）、文档或关于工作负载的其他信息源进行选择。
- 根据指标自动选择资源类型和规模: 使用当前运行的工作负载的指标选择正确的规模和类型，从而优化成本。针对 Amazon EC2、Amazon DynamoDB、Amazon EBS (PIOPS)、Amazon RDS、Amazon EMR 和联网等服务适当预置吞吐量、规模和存储。这可以通过自动扩展等反馈环路进行，也可以在工作负载中使用自定义代码来实现。

COST 7 您如何使用定价模式来降低成本？

使用最适合的资源定价模式可以尽可能减少支出。

最佳实践:

- 执行定价模式分析: 分析工作负载的每个组件。确定组件和资源是长时间运行（享受承诺折扣），还是短时间动态运行（采用 Spot 或按需定价）。利用 AWS Cost Explorer 中的建议功能对工作负载执行分析。
- 根据成本选择区域: 资源定价在每个区域中可能各不相同。考虑区域成本可以确保您为此工作负载支付最低的总体费用
- 选择具有经济实惠的条款的第三方协议: 经济实惠的协议和条款可确保这些服务的成本与所提供的效益相称。选择与可为组织带来额外效益相称的协议和定价。
- 针对此工作负载的所有组件实施定价模式: 永久运行的资源应利用预留容量，如 Savings Plans 或预留实例。短期容量配置为使用 Spot 实例或 Spot 队列。按需实例仅用于无法中断并且运行时间没有长到可以使用预留容量的短期工作负载，时间为使用时期的 25% 到 75%，具体取决于资源类型。
- 在主账户级别执行定价模式分析: 使用 Cost Explorer Savings Plans 和预留实例建议在主账户级别执行承诺折扣定期分析。

COST 8 您如何规划数据传输费用？

务必要监控和规划您的数据传输费用，以便制定架构决策，尽可能降低成本。持续以小步迭代的方式进行架构优化可以实现运营成本的大幅降低。

最佳实践:

- 执行数据传输建模: 收集组织要求，并对工作负载及其每个组件执行数据传输建模。这样可以确定满足当前数据传输要求的最低成本点。
- 选择组件以便优化数据传输成本: 选择所有组件然后设计架构，以便降低数据传输成本。其中包括使用 WAN 优化和多可用区配置等组件
- 实施服务以便降低数据传输成本: 实施服务以便减少数据传输，例如使用 Amazon CloudFront 等 CDN 向最终用户传输内容、使用 Amazon ElastiCache 建立缓存层，或者使用 AWS Direct Connect 而不是 VPN 来连接 AWS。

管理需求和供应资源

COST 9 如何管理需求和供应资源？

为了工作负载的性能与支出实现平衡，请确保您支付过费用的所有资源都得到利用，并避免出现资源利用率过低的情况。无论是从运营成本（由于过度使用导致性能下降）还是从浪费 AWS 支出（由于超额配置）的角度衡量，利用率指标过高或过低都会对您的组织产生负面影响。

最佳实践:

- 对工作负载需求执行分析: 分析工作负载需求随时间的变化。确保分析涵盖季节性趋势，并准确反映整个工作负载生命周期内的运行条件。分析工作应该体现出可能带来的好处，例如花费的时间与工作负载成本成正比。
- 实施缓冲区或限流来管理需求: 缓冲和限流可修改工作负载需求，从而避免出现任何峰值情形。在客户端执行重试时实施限流。实施缓冲以存储请求并将处理任务往后推迟一段时间。确保设计限流和缓冲区时客户端能够在所需的时间内收到响应。
- 动态供应资源: 资源按计划预置。这种预置可以基于需求（例如通过自动扩展来实现），也可以基于时间（需求可以预测，基于时间提供资源）。这些方法可以尽可能减少超额预置或预置不足的情况。

随着时间的推移不断优化

COST 10 如何评估新服务?

AWS 不断发布新服务和功能，因此您最好不断审视现有架构决策，以便确保其始终最具成本效益。

最佳实践:

- 制定工作负载审核流程: 制定一个流程，定义工作负载的审核标准和流程。审核工作应该体现出潜在优势，例如，核心工作负载或费用占比超过 10% 的工作负载每季度审核一次，而费用占比低于 10% 的工作负载每年审核一次。
- 定期审核和分析此工作负载: 按照定义的流程定期审核现有工作负载。