卓越运营支柱

AWS 架构完善的框架

2020 年 7 月

声明

客户有责任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。

目录

简介 1

卓越运营 1

- 设计原则 1
- 定义 1

组织 2

- 组织重点 2
- 运营模式 5
- 组织文化 13

准备 16

- 设计遥测 16
- 运营设计 18
- 降低部署风险 21
- 运营准备 23

运营 27

- 了解工作负载的运行状况 27
- 了解运营状况 30
- 响应事件 32

发展 35

- 学习、分享和改进 35

总结 37

贡献者 38

延伸阅读 38

文档修订 38

摘要

本白皮书主要介绍 AWS 架构完善的框架的卓越运营支柱。它提供了指导，以帮助您在 AWS 工作负载的设计、交付和维护过程中应用最佳实践。

简介

AWS 架构完善的框架能够帮助您认识到您在 AWS 上构建工作负载时所做决策的收益和风险。通过使用此框架，您将了解在云中设计和运行可靠、安全、高效且经济实惠的工作负载的运营和架构最佳实践。它提供了一种统一的方法，使您能够根据最佳实践衡量运营和架构，并确定需要改进的方面。我们相信，拥有在设计时充分考虑了运营因素的 Well-Architected 工作负载，可大大提高实现业务成功的可能性。

该框架基于五大支柱：

- 卓越运营
- 安全性
- 可靠性
- 性能效率
- 成本优化

本白皮书重点介绍了卓越运营支柱，以及如何将其用作架构完善的解决方案的基础。卓越运营在环境中很难实现，因为在环境中，运营被视为一种独立的职能，与它支持的业务团队和开发团队是区分开的。通过采用本白皮书中的实践，您可以构建这样一种架构：提供状态洞察、支持有效且高效的运营和事件响应，并可以持续改进和支持您的业务目标。

本白皮书的目标读者是技术岗位的人员，如首席技术官 (CTO)、架构师、开发人员和运维团队成员。阅读本白皮书后，您将了解在设计云架构以实现卓越运营时可以采用的 AWS 最佳实践和策略。本白皮书不提供实施细节或架构模式，但会针对此类信息提供适当资源。

卓越运营

卓越运营支柱包括您的组织如何支持您的业务目标，您有效运行工作负载的能力，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。

设计原则

在云中实现卓越运营有五个设计原则：

- **执行运营即代码：**在云中，您可以将用于应用程序代码的相同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、基础设施等）定义为代码，并使用该代码进行更新。您可以为运营流程编写脚本，并通过触发来自动执行这些脚本，以响应事件。通过执行运营即代码，您可以减少人为错误并针对事件启用一致响应。
- **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，从而增加对工作负载的有益更改。以较小的增量进行更改，如果更改不能帮助识别和解决向您的环境引入的问题（在可能的情况下不会影响客户），则可以撤消更改。
- **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，以便消除或缓解问题。测试您的故障场景，并确认您了解相应影响。测试您的响应程序，以确保它们有效且团队能够熟练执行。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经验教训，推动改进。在整个团队乃至组织范围*分享经验教训*。

定义

云中的卓越运营包括四个方面：

- 组织

- 准备
- 运营
- 发展

您的组织领导能力定义了业务目标。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。实施服务以实现工作负载的集成、部署和交付，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。

组织

您需要了解您组织的重点、组织结构以及您的组织如何支持您的团队成员，以便为您的业务成果提供支持。

要实现卓越运营，您必须了解以下内容：

- 组织重点
- 运营模式
- 组织文化

组织重点

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。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，以便确定怎样把工作重心放在外部客户的需求上。

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，以便确定怎样把工作重心放在内部客户的需求上。

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。

使用这些已明确的重点，将改进工作集中部署在能发挥最大影响（例如，开发团队技能、提高工作负载性能、降低成本、自动化运行手册或增强监控）的方面。要随着需求的变化更新重点。

评估监管要求：确保您了解组织定义的指导方针或义务，它们可能会要求或强调特定的重点。评估内部因素，如组织策略、标准和要求。验证您是否有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。

评估外部合规性要求：确保您了解指导方针或义务，它们可能会要求或强调特定的重点。评估外部因素，如法规合规性要求和行业标准。验证您是否有确定合规性要求变更的机制。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。

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。

评估威胁情况：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。在确定工作重点时，包括风险的影响。

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 Well-Architected Tool，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。

企业支持客户可以使用针对关键任务型工作负载的指导式 Well-Architected Review，以根据 AWS 最佳实践来衡量其架构。

他们还可以使用运营审核，该审核旨在帮助他们找出云中的运营方法所存在的漏洞。

这些审核需要跨团队参与，可帮助各团队在工作负载以及他们在实现成功中的角色方面达成一致的理解。通过审查所确定的需求可以帮助确定您的运营重点。

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。

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，以便在确定运营重心或选择行动方案时做出明智的决策。例如，对于新功能的加速上市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。

AWS 可以帮您向团队介绍 AWS 及其服务，让他们深入了解他们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。

管理收益和风险：管理收益和风险，以便在确定运营重心时做出明智的决策。例如，部署存在未解决问题的工作负载可能会有所帮助，因此可以向客户提供这一重要的新功能。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，您将采取措施来解决风险。

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。要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。

资源

请参阅以下资源，详细了解有关组织重点的 AWS 最佳实践。

文档

- AWS Trusted Advisor
- AWS 云合规性
- AWS 架构完善的框架
- AWS 商业支持

- AWS 企业支持
- AWS 企业支持权益
- AWS Support 云运营审核
- AWS 云采用框架

运营模式

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中所扮演的角色、其他团队在他们获得成功中所扮演的角色，并设定共同的目标。了解应负的责任、拥有的所有权和决策制定方式以及有权进行决策的人员，将有助于确定运营重点，最大限度地从您的团队中获益。

团队的需求将由其所在行业、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所有团队及其工作负载是不合理的。

随着开发团队数量的增加，组织中存在的运营模式数量也可能会增加。您可能需要使用运营模式组合。

采用标准和消费服务可以简化运营，并限制运营模式中的支持负担。通过确定采用标准和采用新功能的团队的数量，可以放大在共享标准方面的开发工作的益处。

一定要建立相应的请求增加、更改标准和标准例外的机制，以支持团队的活动。如果没有这样的机制，标准将成为创新的约束。对收益和风险进行评估之后，批准可行的和确认适当的请求。

明确定义职责范围将减少发生冲突和导致产生冗余工作的频率。如果业务团队、开发团队和运营团队之间存在紧密的协作关系，则更容易实现业务成果。

运营模式 2:2 展示图

这些运营模式 2:2 展示图可帮助您了解您环境中的团队之间的关系。这些图表着重说明成员的职责以及团队之间的关系，但我们也将通过这些示例讨论监管和做出决策。

我们的团队可能需要在多个模式的多个部分承担责任，具体取决于他们支持的工作负载。您可能希望打破比所描述的高级规范更专业的规范。当您分离或汇总活动，或叠加团队并提供更具体的细节时，这些模式可能会出现无穷无尽的变化。

您可能发现团队中存在重叠或未被认可的能力，这些能力可以提供额外优势或提高效率。您可能还发现您可以计划解决的组织中未满足的需求。

在评估组织变革时，请检查模式之间的权衡，您的各个团队采用的模式（现在和变革之后），您的团队的关系和职责将如何变化，以及所获得的益处是否抵得过对您的组织产生的影响。

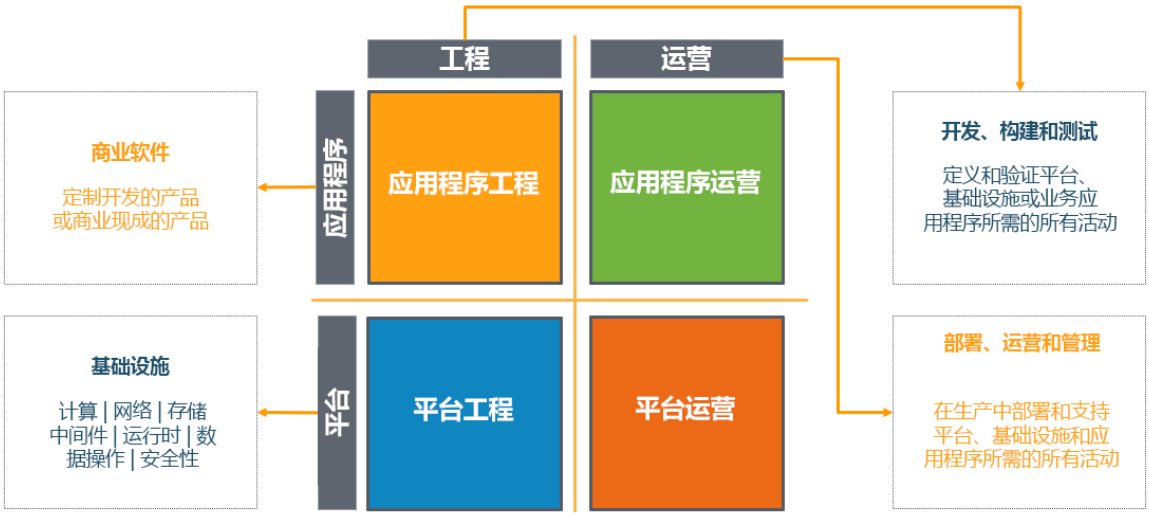
您可以成功使用以下四种运营模式。某些模式更适合于特定使用案例或您的开发中的特定点。其中一些模式可能比在您的环境中使用的模式更具优势。

- 完全分离运营模式
- 独立的应用程序工程和运营 (AEO) 以及具有集中监管的基础设施工程和运营 (IEO)
- 采用集中监管并具有服务提供商的分离的 AEO 和 IEO
- 采用分散监管的分离的 AEO 和 IEO

完全分离运营模式

在下图中，纵轴上为应用程序和基础设施。应用程序是指促进取得业务成果的工作负载，可以是自定义开发或购买的软件。基础设施是指支持该工作负载的物理和虚拟基础设施以及其他软件。

横轴上为我们的工程和运营。工程是指应用程序和基础设施的开发、构建和测试。运营是应用程序和基础设施的部署、更新和持续支持。



在许多组织中，存在这种“完全分离”模式。每个象限中的活动由单独的小组执行。通过工作请求、工作队列、票证等机制或使用 IT 服务管理 (ITSM) 系统在团队之间分配工作。

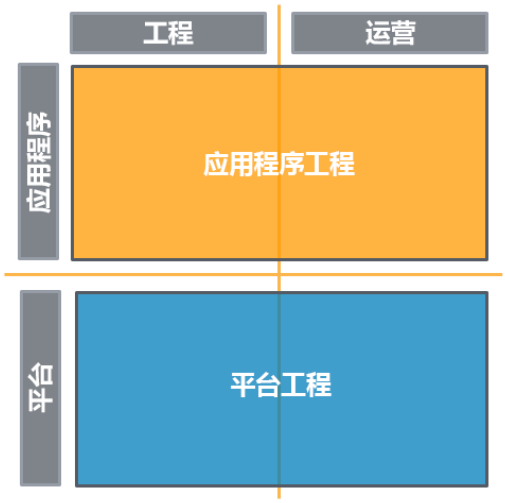
任务向团队或在团队之间的转移会增加复杂性，并造成瓶颈和延迟问题。请求可能会被延迟，直至它们成为重点事项。较迟发现缺陷可能需要大量返工，可能需要再次经历相同的团队及其职能部门。如果存在需要工程团队采取行动的事件，则将因转移活动而延迟响应。

当围绕正在执行的活动或职能组织业务团队、开发团队和运营团队时，出现工作重点偏失的风险较高。这可能导致团队专注于其特定职责，而不是专注于实现业务成果。团队可能专业化水平受限、被物理隔离或逻辑隔离，阻碍了沟通和协作。

采用集中监管的分离的 AEO 和 IEO

这种“分离的 AEO 和 IEO”模式采用“你构建，你运行”的方法。

应用程序工程师和开发人员同时执行工作负载工程设计和运营。同样，您的基础设施工程师可以对他们用以支持应用程序团队的平台同时进行工程设计和运营。



在本示例中，我们采用集中监管。标准方式是将分发、提供或共享给应用程序团队。

您应使用能够跨 AWS Organizations 等账户集中监管环境的工具或服务。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。

“你构建，你运行”并不意味着应用程序团队负责完全堆栈、工具链和平台。

平台工程设计团队为应用程序团队提供一套标准化的服务（例如，开发工具、监控工具、备份和恢复工具以及网络）。平台团队还可以为应用程序团队提供对经批准的云提供商服务、相同或两个团队的特定配置的访问权限。

提供部署已批准的服务和配置的自助服务功能的机制，如 AWS Service Catalog，可以在实施监管的同时帮助限制与执行请求相关的延迟。

平台团队实现了完全堆栈可见性，因此应用程序团队可以区分应用程序组件的问题以及应用程序所使用的服务和基础设施组件。平台团队还可以提供配置这些服务的辅助措施，以及有关如何改进应用程序团队运营的指导。

如前所述，应用程序团队一定要建立相应的请求增加、更改标准和标准例外的机制，以支持团队的活动及其应用程序的创新。

分离的 AEO 和 IEO 模式为应用程序团队提供了强大的反馈环路。工作负载的日常运营通过直接交互或通过支持和功能请求间接增加与客户的联系。这种更高的可见性使应用程序团队能够更快地解决问题。更深入的互动和更密切的关系可提供对客户需求的洞察，并实现更快速的创新。

这也完全适用于为应用程序团队提供支持的平台团队。

采用的标准可以预先批准以供使用，从而减少投产所需的审核量。采用由平台团队提供的受支持的、业经测试的标准可以减少这些服务出现问题的频率。标准的采用可帮助应用程序团队专注于差异化工作负载。

采用集中监管并具有服务提供商的分离的 AEO 和 IEO

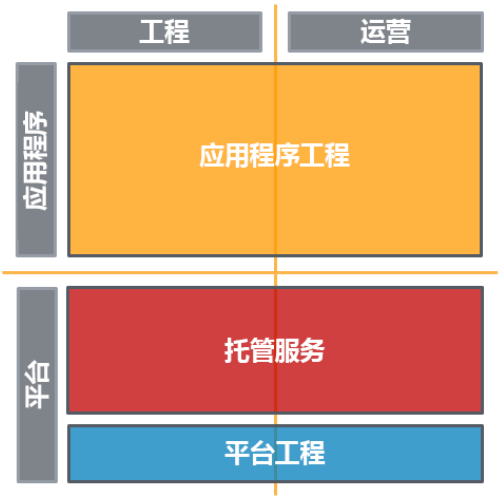
这种“分离的 AEO 和 IEO”模式采用“你构建，你运行”的方法。

应用程序工程师和开发人员同时执行工作负载工程设计和运营。

您的组织现在可能无法为专门的平台工程和运营团队提供相应的技能或团队人员支持，或者您可能不想为此花费时间和精力。

或者，您可能希望有一个平台团队能够专注于打造凸显业务优势的能力，不过您希望将千篇一律的日常运营工作交给外包商。

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。



对于这一变体，我们视为监管由平台团队集中管理，并使用 AWS Organizations 和 AWS Control Tower 管理账户创建和策略。

此模式需要您修改自身机制，以便使用服务提供商的机制。它不能解决由于团队（包括您的服务提供者）之间的任务转换所造成的瓶颈和延迟，也无法解决由于发现缺陷较晚而存在的潜在返工。

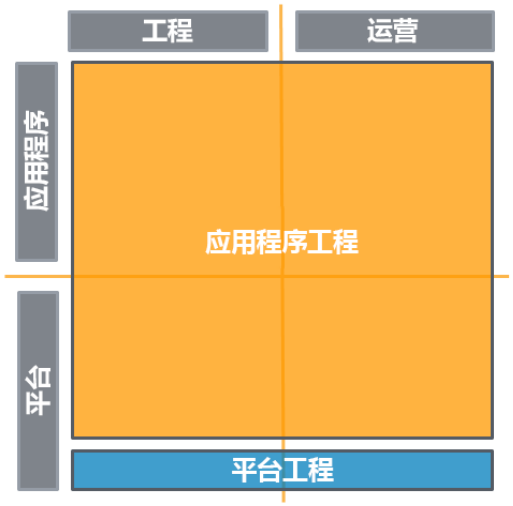
提供商的标准、最佳实践、流程和专业知识将让您受益良多。此外，他们还会不断开发服务产品，您也会从中获益。

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。

采用分散监管的分离的 AEO 和 IEO

这种“分离的 AEO 和 IEO”模式采用“你构建，你运行”的方法。

应用程序工程师和开发人员同时执行工作负载工程设计和运营。同样，您的基础设施工程师可以对他们用以支持应用程序团队的平台同时进行工程设计和运营。



在本示例中，我们采用分散监管。

标准仍由平台团队分发、提供或共享给应用程序团队，但是应用程序团队可以自由设计和操作新的平台功能来支持其工作负载。

在此模式中，对应用程序团队的约束较少，但是随之而来的是责任的显著增加。必须具备更多技能以及潜在的团队成员，才能支持其他平台功能。如果缺乏相应技能且不能及早发现缺陷，则会增加大量返工的风险。

您应该执行那些没有专门委托给应用程序团队的策略。使用能够跨 AWS Organizations 等账户集中管理环境的工具或服务。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。

为应用程序团队设定可请求添加和变更标准的机制，这作用很大。他们也许能够提出新标准，让其他应用程序团队也因此受益。平台团队可以决定，为这些附加功能提供直接支持是否是对业务成果的有效支持。

由于该模式具有重要技能和团队成员要求，因此限制了创新。它解决了团队之间由于任务转换所造成的诸多瓶颈和延迟，同时还促进了团队与客户之间有效关系的发展。

关系和所有权

您的运营模式定义了团队之间的关系，并为可识别的所有权和责任提供支持。

资源已确定拥有者：了解谁对每个应用程序、工作负载、平台和基础设施组件拥有所有权，该组件提供了哪些业务价值以及为什么具有这种所有权。了解这些单个组件的业务价值以及它们如何支持业务成果，为它们应用的流程和程序提供信息。

流程和程序已确定拥有者：了解谁对单个流程和过程的定义拥有所有权，为何使用这些特定的流程和过程以及为什么具有这种所有权。了解使用特定流程和程序的原因，可以发现改进机会。

运营活动已确定负责执行的拥有者：了解谁负责在定义的工作负载上执行特定活动，以及为什么要负责。了解运营活动的执行责任可以了解谁将执行操作、验证结果并向活动拥有者提供反馈。

团队成员了解他们要负责什么：了解您的角色可以确定任务的优先级。这使团队成员能够了解需求并作出相应的响应。

具有识别责任和所有权的机制：在未识别任何个人或团队的情况下，有权分配所有权或计划解决此问题的人具有定义的上报路径。

具有请求添加、变更和异常的机制：您可以向流程、过程和资源的拥有者提出请求。对收益和风险进行评估之后，做出明智的决定，批准可行的和确认适当的请求。

团队之间的职责是预先定义或协商的：团队之间会定义或协商协议，以描述团队之间的合作和相互支持（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。

当责任和所有权不确定或未知时，您将面临以下风险：没有及时处理必要的活动，以及在处理这些需求时可能出现工作冗余和潜在冲突。

资源

请参阅以下资源，详细了解有关运营设计的 AWS 最佳实践。

视频

- AWS re:Invent 2019: [REPEAT 1] How to ensure configuration compliance (MGT303-R1)
- AWS re:Invent 2019: Automate everything: Options and best practices (MGT304)

文档

- AWS Managed Services

- AWS Organizations 特征
- AWS Control Tower 特征

组织文化

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。

高管支持：高级领导层明确为组织设定期望并评估是否成功。高级领导层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：工作负载拥有者定义了指导和范围，授权团队成员在成果面临风险时做出响应。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：团队成员具有相应的机制，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。应尽早和经常地进行上报，以便能够确定风险，并防止造成事故。

沟通及时、清晰、可行：具有相应机制并用于及时将已知风险和计划内事件通知给团队成员。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、需要采取什么措施并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。

可以将计划内事件记录在变更日历或维护时间表中，以便团队成员可以确定哪些活动待处理。

在 AWS 上，可以使用 AWS Systems Manager 变更日历来记录这些详细信息。它支持对日历状态进行程序检查，以确定日历在特定时间点对活动是打开还是关闭。运营活动可以根据特定的“批准”时间窗进行规划，这些时间窗是为潜在的破坏性活动预留的。AWS Systems Manager 维护时段允许您根据实例和其他支持资源安排活动，从而自动执行活动并发现这些活动。

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度。取得非预期结果也算试验成功，因为这种试验发现了不会取得成功的途径。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。

支持并鼓励团队成员保持和增强他们的技能：团队必须增强他们的技能，以采用新技术；并为需求和职责的变化提供支持，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。支持您的团队成员实施和维护行业认证，以验证和认可他们不断增强的技能。进行交叉培训，可以促进知识传播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。专门安排系统时间进行学习。

AWS 提供了许多资源，包括 AWS 入门资源中心、AWS 博客、AWS 在线技术讲座、AWS 事件和网络研讨会以及 AWS Well-Architected 实验室，它们提供了指导、示例和详细演练，用以培训您的团队。

AWS 还在 Amazon Builders' Library 中分享了通过 AWS 运营学到的最佳实践和模式；并通过 AWS 博客和 AWS 官方播客分享了各种实用的教材。

您应该利用 AWS 提供的教育资源，例如 Well-Architected 实验室、AWS Support（AWS 知识中心、AWS 开发论坛和 AWS Support 中心）和 AWS 文档来培训您的团队。请通过 AWS Support 中心联系 AWS Support，获取与 AWS 问题有关的帮助。

AWS 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。

适当地配置资源团队：培养团队成员的能力，并提供工具和资源来支持您的工作负载需求。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寻求不同的观点：利用跨组织的多样性来寻求多种独特的见解。利用见解提高创新能力，对您的假设提出质疑以及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。

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。

资源

请参阅以下资源，详细了解有关运营设计的 AWS 最佳实践。

视频

- AWS re:Invent 2019: [REPEAT 1] How to ensure configuration compliance (MGT303-R1)

- AWS re:Invent 2019: Automate everything: Options and best practices (MGT304)

文档

- AWS Managed Services
- AWS Managed Services 服务描述
- AWS Organizations 特征
- AWS Control Tower 特征

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。

要为卓越运营做好准备，您需要执行以下操作：

- 设计遥测
- 改进流程
- 降低部署风险
- 了解运营准备

设计遥测

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，确定何时面临风险并做出有效响应。

在 AWS 中，您可以从应用程序和工作负载组件中发出并收集日志、指标和事件，以了解其内部状态和运行状况。您可以集成分布式跟踪，在请求通过工作负载时跟踪它们。使用此数据来了解您的应用程序与基础组件之间的交互方式，并分析问题和性能。

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。

实施应用程序遥测：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）。使用这些信息并确定需要在什么时候响应。

您应安装和配置统一的 Amazon CloudWatch Logs 代理，将系统级应用程序日志和高级指标从 EC2 实例和物理服务器发送到 Amazon CloudWatch。

使用 AWS CLI 或 CloudWatch API 生成和发布自定义指标。确保发布富有洞察力的业务指标和技术指标，以帮助您了解客户的行为。

您可以使用 CloudWatch Logs API 将日志直接从应用程序发送到 CloudWatch，或者使用 AWS 开发工具包和 Amazon EventBridge 发送事件。将日志记录语句插入到 AWS Lambda 代码中，以将其自动存储在 CloudWatch Logs 中。

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。使用这些信息并确定需要在什么时候响应。

使用 Amazon CloudWatch 等服务聚合工作负载组件中的日志和指标（例如，AWS CloudTrail 的 API 日志、AWS Lambda 指标、Amazon VPC 流日志和 其他服务）。

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）。使用这一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。

实施依存项遥测：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。外部依存项的示例可以包括外部数据库、DNS 和网络连接。使用这些信息并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响应，并帮助您确定导致问题的因素。

在 AWS 中，您可以使用分布式跟踪服务（例如 AWS X-Ray）来收集和记录事务通过工作负载时的跟踪记录，生成地图以查看事务如何在工作负载和服务之间流动，深入了解组件之间的关系，并实时识别和分析问题。

随着工作负载的发展，迭代并开发遥测技术，以确保您继续接收必要的信息，从而深入了解工作负载的运行状况。

资源

请参阅以下资源，详细了解有关运营设计的 AWS 最佳实践。

视频

- AWS re:Invent 2016: Infrastructure Continuous Delivery Using AWS CloudFormation (DEV313)
- AWS re:Invent 2016: DevOps on AWS: Accelerating Software Delivery with AWS

Developer Tools (DEV201)

- AWS CodeStar: The Central Experience to Quickly Start Developing Applications on AWS

文档

- 访问 Amazon CloudWatch Logs 了解 AWS Lambda
- 使用 Amazon CloudWatch Logs 监控 CloudTrail 日志文件
- 将流日志发布到 CloudWatch Logs

文档

- Enhancing workload observability using Amazon CloudWatch Embedded Metric Format
- Amazon CloudWatch 入门
- Store and Monitor OS & Application Log Files with Amazon CloudWatch
- High-Resolution Custom Metrics and Alarms for Amazon CloudWatch
- 通过 Amazon CloudWatch Events 监控 AWS 运行状况事件
- AWS CloudFormation 文档
- AWS 开发人员工具
- 在 AWS 上设置 CI/CD 管道
- AWS X-Ray
- AWS 标记策略
- Enhancing workload observability using Amazon CloudWatch Embedded Metric Format

运营设计

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。这些方法可以加快将有益更改落实到生产环境的速度、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。

在 AWS 中，您可以将整个工作负载（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。

使用版本控制：使用版本控制来跟踪更改和发布。

许多 AWS 服务提供版本控制功能。使用修订或源代码控制系统（如 AWS CodeCommit）管理代码和其他构件，如基础设施的版本控制的 AWS CloudFormation 模板。

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。

在 AWS 上，您可以创建临时并行环境，以降低试验和测试的风险、工作量及成本。使用 AWS CloudFormation 自动部署这些环境，以确保临时环境实施的一致性。

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。

在 AWS 中，您可以使用像 AWS 开发人员工具（例如，AWS CodeCommit、AWS CodeBuild、AWS CodePipeline、AWS CodeDeploy 和 AWS CodeStar）这样的服务来构建持续集成/持续部署 (CI/CD) 管道。

执行补丁管理：执行补丁管理以实现功能、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。

补丁和漏洞管理是优势和风险管理活动的一部分。最好是具有不可变的基础设施和在已验证的已知良好状态下部署工作负载。如果该方法都不可行，那就只能进行修补。

更新系统镜像、容器镜像或 Lambda 自定义运行时和其他库以消除漏洞，是补丁管理的一部分。您应该使用 EC2 Image Builder 来管理适用于 Linux 或 Windows Server 镜像的 Amazon 系统映像 (AMI) 更新。您可以使用 Amazon Elastic Container Registry 与您的现有管道来管理 Amazon ECS 镜像以及管理 Amazon EKS 镜像。AWS Lambda 中包含版本管理功能。

在未事先在安全环境中测试的情况下，不应对生产系统执行修补操作。仅当补丁支持操作或业务结果时，才应该应用补丁。在 AWS 上，您可以使用 AWS Systems Manager 补丁管理器和 AWS Systems Manager 维护时段来自动执行修补托管系统的过程和安排修补活动。

共享设计标准：在不同团队间共享最佳实践，以提高认识并最大限度地提高开发工作的效率。

在 AWS 上，可以使用代码方法定义和管理应用程序、计算、基础设施和操作。这让您可以轻松地发布、分享和采用内容。

许多 AWS 服务和资源都可以设计为跨账户共享，从而使您能够跨团队分享所创建的资产和所学到的知识。例如，您可以与特定账户共享 CodeCommit 存储库、Lambda 函数、Amazon S3 存储桶和 AMI。

发布新资源或更新时，请使用 Amazon SNS 提供跨账户通知。订阅者可以使用 Lambda 获取新版本。

如果在组织中强制实施了共享标准，则必须存在相应的机制来以请求增加、更改标准和标准例外，以为团队的活动提供支持。如果没有这样的机制，标准将成为创新的约束。

实施实践以提高代码质量：实施实践以提高代码质量并最大程度地减少缺陷。例如，测试驱动型开发、代码审查和实施标准化。

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。当环境接近于生产时，使用更高级别的控制，让您能够确信您的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更改可以减小更改的范围和影响。这样可以简化故障排除，实现更快地修复，并提供回滚更改的选项。

完全自动化集成和部署：实现工作负载的自动构建、部署和测试。这样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。

资源

请参阅以下资源，详细了解有关运营设计的 AWS 最佳实践。

视频

- AWS re:Invent 2016: Infrastructure Continuous Delivery Using AWS CloudFormation (DEV313)
- AWS re:Invent 2016: DevOps on AWS: Accelerating Software Delivery with AWS Developer Tools (DEV201)
- AWS CodeStar: The Central Experience to Quickly Start Developing Applications on AWS

文档

- 什么是 AWS Resource Groups
- Amazon CloudWatch 入门
- Store and Monitor OS & Application Log Files with Amazon CloudWatch
- High-Resolution Custom Metrics and Alarms for Amazon CloudWatch
- 通过 Amazon CloudWatch Events 监控 AWS 运行状况事件
- AWS CloudFormation 文档
- AWS 开发人员工具
- 在 AWS 上设置 CI/CD 管道
- AWS X-Ray
- AWS 标记策略

降低部署风险

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缓因部署更改而产生的问题的影响。

工作负载的设计应包括其部署、更新和运营方式。您需要实施以减少缺陷并快速安全地修复为目标的工程实践。

针对不成功的更改制定计划：制定计划，以便在更改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，缩短回滚时间。

测试并验证更改：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。

在 AWS 上，您可以创建临时并行环境，以降低试验和测试的风险、工作量及成本。使用 AWS CloudFormation 自动部署这些环境，以确保临时环境实施的一致性。

使用部署管理系统：使用部署管理系统来跟踪并实施更改。这样可以减少手动过程引起的错误，并减少部署更改的工作量。

在 AWS 中，您可以使用像 AWS 开发人员工具（例如，AWS CodeCommit、AWS CodeBuild、AWS CodePipeline、AWS CodeDeploy 和 AWS CodeStar）这样的服务来构建持续集成/持续部署 (CI/CD) 管道。

当计划的重要业务、运营活动或事件受到更改实施的影响时，建立更改日历并进行跟踪。围绕这些计划来调整活动以管理风险。AWS Systems Manager Change Calendar 提供了一种机制，可以记录更改开始或结束的时间块及更改原因，并与其他 AWS 账户共享该信息。可以将 AWS Systems Manager Automation 脚本配置为符合更改日历状态。

AWS Systems Manager 维护时段可用于安排在指定的时间执行 AWS SSM Run Command 或 Automation 脚本、AWS Lambda 调用或 AWS Step Function 活动。在更改日历中标记这些活动，以便将其包含在您的评估中。

使用有限部署进行测试：在全面扩展部署之前使用有限的部署和现有系统进行测试，以确认目标成效。例如，使用部署 Canary 测试或单盒部署。

使用并行环境进行部署：对并行环境实施更改，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更改影响的范围。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：实现工作负载的自动构建、部署和测试。这样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成效。在没有达到预期结果时自动回滚到之前的已知良好状态，尽可能地缩短恢复时间，并减少手动过程引起的错误。

资源

请参阅以下资源，详细了解有关运营设计的 AWS 最佳实践。

视频

- AWS re:Invent 2016: Infrastructure Continuous Delivery Using AWS CloudFormation (DEV313)
- AWS re:Invent 2016: DevOps on AWS: Accelerating Software Delivery with AWS Developer Tools (DEV201)
- AWS CodeStar: The Central Experience to Quickly Start Developing Applications on AWS

文档

- Amazon CloudWatch 入门
- Store and Monitor OS & Application Log Files with Amazon CloudWatch
- High-Resolution Custom Metrics and Alarms for Amazon CloudWatch
- 通过 Amazon CloudWatch Events 监控 AWS 运行状况事件
- AWS CloudFormation 文档
- AWS 开发人员工具
- 在 AWS 上设置 CI/CD 管道
- AWS X-Ray
- AWS 标记策略

运营准备

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）来了解何时已准备就绪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。您需要有记录日常活动的运行手册和指导问题解决过程的行动手册。

确保员工产能：制定一种机制来验证您是否具有适当数量的训练有素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。

您需要拥有足够的团队成员来完成所有活动（包括待命的团队成员）。确保您的团队拥有必要的技能，成功地接受过关于您的工作负载、运营工具和 AWS 的培训。

AWS 提供了许多资源，包括 AWS 资源中心入门、AWS 博客、AWS 在线技术讲座、AWS 事件和网络研讨会，以及 AWS Well-Architected 实验室，这些资源提供了指导、示例和详细演练，用以培训您的团队。此外，AWS Training and Certification 还提供了一些免费培训，包括关于 AWS 的基础知识的自主进度数字课程。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。

确保以一致的方式审核运行准备就绪情况：确保以一致的方式对运行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行就绪情况，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过程引起的错误。

您应使用 AWS Config 建立基准，并使用 AWS Config 规则检查配置，从而自动执行工作负载配置测试。您可以使用 AWS Security Hub 的服务和功能评估安全要求和合规性。这些服务将有助于确定您的工作负载是否与最佳实践和标准保持一致。

使用运行手册来执行程序：运行手册是用来实现特定成果的书面程序。在运行手册中记录程序，确保对常见事件做出一致且及时的响应。通过代码实施运行手册，并在适当情况下针对事件触发自动执行运行手册，以便确保一致性、响应速度并减少由手动过程引起的错误。

使用行动手册来发现问题：行动手册是用于调查问题的书面程序。在行动手册中记录调查流程，实现对故障场景做出一致且及时的响应。通过代码实施行动手册，并在适当情况下针对事件触发自动执行行动手册，以便确保一致性、响应速度并减少由手动过程引起的错误。

AWS 支持您将运营视为代码，为运行手册和行动手册活动编写脚本，以降低出现人为错误的风险。您可以将资源标记或资源组与您的脚本一起使用，以根据您定义的条件（例如，环境、所有者、角色或版本）有选择地执行。

您可以使用脚本程序通过触发脚本来启用自动化，以响应事件。通过将运营和工作负载视为代码，您还可以编写脚本并自动评估您的环境。

您应使用 AWS Systems Manager (SSM) Run Command 在您的实例上编写程序脚本，使用 AWS Systems Manager Automation 在实例和其他资源上编写操作脚本并创建工作流程，或使用 AWS Lambda 无服务器计算函数编写响应脚本，以响应 AWS 服务 API 和您自己的自定义接口中的事件。您还可以使用 AWS Step Functions 协调多个以脚本方式写入到无服务器工作流程的 AWS 服务。通过使用 CloudWatch Events 触发这些脚本以自动做出响应，并使用 Amazon EventBridge 将所需的事件路由到其他运营支持系统。

您应该测试您的程序、故障场景以及您的响应是否成功（例如，通过举办游戏日和在上线前测试），以确定需要制定计划予以解决的问题。

在 AWS 上，您可以创建临时并行环境，以降低试验和测试的风险、工作量及成本。使用 AWS CloudFormation 自动部署这些环境，以确保以一致的方式实施您的临时环境。在客户影响可以接受或没有客户影响的安全环境中执行故障注入测试，并制定或修改相应的响应措施。

做出部署系统和更改的明智决策：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，根据部署的优势对这些指标进行评估。了解优势和风险，以便做出明智的决策。

使用“预先检验”来预测故障，并在适当的时候创建程序。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。

资源

请参阅以下资源，详细了解有关运营准备就绪情况的 AWS 最佳实践。

文档

- AWS Lambda
- AWS Systems Manager
- AWS Config Rules – Dynamic Compliance Checking for Cloud Resources
- How to track configuration changes to CloudFormation stacks using AWS Config
- Amazon Inspector 更新博客文章
- AWS 活动和网络研讨会
- AWS 培训

- AWS Well-Architected 实验室
- AWS 启动标签策略
- Using AWS Systems Manager Change Calendar to prevent changes during critical events

运营

成功是指按照您定义的指标衡量，实现了业务成果。通过了解工作负载和运营的运行状况，您可以确定何时组织和业务成果可能陷入风险或已遇到风险，并采取适当的响应措施。

要想取得成功，您必须能够：

- 了解工作负载的运行状况
- 了解运营状况
- 响应事件

了解工作负载的运行状况

定义、记录和分析工作负载指标以便了解工作负载事件，从而采取适当的措施。

您的团队应该能够轻松了解工作负载的运行状况。您需要根据工作负载结果使用指标进行判断，以获得有用的见解。您应该使用这些指标来实施提供业务和技术观点的控制面板，以帮助团队成员做出明智决策。

AWS 使您能够轻松汇总和分析您的工作负载日志，以便您可以生成指标，了解工作负载的运行状况，并深入了解一段时间内的运营情况。

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）识别关键性能指标 (KPI)。评估 KPI 以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效，并了解工作负载的运行状况。

您应将日志数据发送到像 CloudWatch Logs 这样的服务，并根据对必要日志内容的观察生成指标。

CloudWatch 具有一些专业功能，例如，适用于 .NET 和 SQL Server 的 Amazon CloudWatch Insights 及 Container Insights，这些功能可通过识别和设置专门支持的应用程序资源和技术堆栈的关键指标、日志和警报来为您提供帮助。

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需要做出哪些适当响应。

您应汇总应用程序、工作负载组件、服务和对服务（如 CloudWatch Logs）的 API 调用的日志数据。通过对必要的日志内容进行观察生成指标，从而深入了解运营活动的表现。

在 AWS 责任共担模式中，部分监控会通过 AWS Personal Health Dashboard 提供给您。AWS Personal Health Dashboard 会在 AWS 遇到可能会影响您的事件时提供提醒和修正指导。拥有商业支持和企业支持订阅的客户还可以获取 AWS 运行状况 API，从而实现与其事件管理系统的集成。

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。

另一种解决方案是，使用 Amazon Elasticsearch Service 和 Kibana 来收集、分析和显示跨多个账户和 AWS 区域的 AWS 日志。

建立工作负载指标基准：建立指标基准以便提供预期值，作为比较和标识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。

了解工作负载活动的预期模式：建立工作负载活动的模式以识别异常行为，从而使您能够根据需要作出相应的响应。

CloudWatch 通过 CloudWatch 异常检测功能来应用统计和机器学习算法，以生成代表正常指标行为的预期值范围。

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，以便您能够根据需要做出适当响应。

理想情况下，您之前已经确定能够作为发出提醒依据的指标阈值，或可以用于触发自动响应的事件。

您也可以使用专门构建的查询语言，使用 CloudWatch Logs Insights 以交互方式搜索和分析您的日志数据。CloudWatch Logs Insights 会自动发现 AWS 服务日志中的字段，以及 JSON 格式的自定义日志事件。它会随您的日志量和查询复杂性而扩展，并在数秒内为您提供答案，从而帮助您搜索引发事件的因素。

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，以便您能够根据需要做出适当响应。

您对一段时间内工作负载指标的分析可能会建立行为模式，您可以对这些模式进行充分量化，以定义事件或发出警报作为响应。

经过训练后，CloudWatch 异常检测功能可用于对检测到的异常发出警报，或将期望值叠加到指标数据图表上，以进行持续的比较。

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运营情况，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。

AWS 还通过 AWS 服务 API 和软件开发工具包（例如，Grafana、Kibana 和 Logstash）支持第三方日志分析系统和商业智能工具。

资源

请参阅以下资源，详细了解有关工作负载运行状况的 AWS 最佳实践。

视频

- AWS re:Invent 2015: Log, Monitor, and Analyze your IT with Amazon CloudWatch (DVO315)
- AWS re:Invent 2016: Amazon CloudWatch Logs and AWS Lambda: A Match Made in Heaven (DEV301)

文档

- 什么是 Amazon CloudWatch Application Insights for .NET and SQL Server?

- Store and Monitor OS & Application Log Files with Amazon CloudWatch
- API & CloudFormation Support for Amazon CloudWatch Dashboards
- AWS Answers：集中式日志记录

了解运营状况

定义、记录和分析运营指标以便了解工作负载事件，从而采取适当的措施。

您的团队应该能够轻松了解自己的运营状况。您需要根据运营结果使用指标进行判断，以获得有用的见解。您应该使用这些指标来实施提供业务和技术观点的控制面板，以帮助团队成员做出明智决策。

AWS 使您能够轻松汇总和分析您的运营日志，以便您可以生成指标，了解您的运营状况，并深入了解运营状况在一段时间内的变化情况。

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实现所需成果，并了解运营活动的运行状况。

收集和分析运营指标：定期主动检查各种指标，以便发现趋势并确定需要做出哪些适当响应。

您应该将来自执行运营活动和操作 API 调用的日志数据聚合到像 CloudWatch Logs 这样的服务中。根据对必要日志内容的观察生成指标，从而深入了解运营活动的性能。

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。

建立运营指标基准：建立指标基准以提供预期值，作为比较和标识运营活动性能不足和性能过高的依据。

了解运营活动的预期模式：建立运营活动的模式以识别异常活动，从而根据需要作出适当响应。

在工作负载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，以便您能够根据需要作出适当响应。

理想情况下，您之前已经确定能够作为发出提醒依据的指标，或可以用于触发自动响应的事件。

您也可以使用专门构建的查询语言，使用 CloudWatch Logs Insights 以交互方式搜索和分析您的日志数据。CloudWatch Logs Insights 会自动发现 AWS 服务日志中的字段，以及 JSON 格式的自定义日志事件。它会随您的日志量和查询复杂性而扩展，并在数秒内为您提供答案，从而帮助您搜索引发事件的因素。

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，以便您能够根据需要作出适当响应。

您对一段时间内运营指标的分析可能会建立行为模式，您可以对这些模式进行充分量化，以定义事件或发出警报作为响应。

经过训练后，CloudWatch 异常检测功能可用于对检测到的异常发出警报，或将期望值叠加到指标数据图表上，以进行持续的比较。

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。

AWS 还通过 AWS 服务 API 和软件开发工具包（例如，Grafana、Kibana 和 Logstash）支持第三方日志分析系统和商业智能工具。

资源

请参阅以下资源，详细了解有关运营状况的 AWS 最佳实践。

视频

- AWS re:Invent 2015: Log, Monitor, and Analyze your IT with Amazon CloudWatch (DVO315)
- AWS re:Invent 2016: Amazon CloudWatch Logs and AWS Lambda: A Match Made in Heaven (DEV301)

文档

- Store and Monitor OS & Application Log Files with Amazon CloudWatch
- API & CloudFormation Support for Amazon CloudWatch Dashboards
- AWS Answers：集中式日志记录

响应事件

您应该预测运营事件，包括计划内（例如，促销、部署和故障测试）和计划外（例如，利用率激增和组件故障）事件。在响应提醒时，您应该使用现有的运行手册和行动手册来交付一致的结果。定义的警报应由负责响应和升级的角色或团队所有。您还需要了解系统组件的业务影响，并在需要时使用它来设定工作目标。您应该在事件发生后执行根本原因分析 (RCA)，然后防止故障再次发生或记录解决方法。

AWS 可以提供工具，为工作负载和运营即代码的方方面面提供支持，从而简化您的事件响应过程。借助这些工具，您可以编写对运营事件的响应脚本，并触发其执行以响应监控数据。

在 AWS 中，您可以将故障组件替换为现有的优秀版本，而不是尝试修复它们，以此来缩短恢复时间。然后，您可以在带外对故障资源进行分析。

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。

在 AWS 上，您可以将 AWS Systems Manager OpsCenter 用作中心位置，以查看、调查并解决与任何 AWS 资源相关的运营问题。它可以聚合运营问题并提供上下文相关的数据，以帮助响应事件。

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（运行手册或行动手册），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、声誉或信任损害都是一种影响。

定义升级路径：在运行手册和行动手册中定义升级路径，包括触发升级的事件和升级程序。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。

在采取措施之前，确定何时需要人为决定。与决策者合作，提前做出决策，这样 MTTR 便不会因为等待响应而延长。

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。

您可以使用 Amazon CloudWatch 控制面板，在 CloudWatch 控制台中可自定义的主页上创建控制面板。借助像 Amazon QuickSight 这样的商业智能服务，您可以创建和发布工作负载和运营状况的交互式控制面板（例如，订单达成率、连接的用户和交易时间）。您可以创建控制面板，用来显示指标的系统级和业务级视图。

自动响应事件：自动响应事件以便减少由手动过程引起的错误，并确保响应及时并且一致。

有多种方法可以在 AWS 上自动执行运行手册和行动手册操作。要响应 AWS 资源状态更改事件或您自己的自定义事件，您应创建 CloudWatch Events 规则，以通过 CloudWatch 目标（例如，Lambda 函数、Amazon Simple Notification Service [Amazon SNS] 主题、Amazon ECS 任务和 AWS Systems Manager Automation）触发响应。

要响应超过资源阈值的指标（例如，等待时间），您应创建 CloudWatch 警报，以使用 Amazon EC2 操作或 Auto Scaling 操作执行一个或多个操作，或向 Amazon SNS 主题发送通知。如果您需要执行自定义操作以响应警报，请通过 Amazon SNS 通知调用 Lambda。使用 Amazon SNS 发布事件通知和升级消息，以便让用户了解情况。

AWS 还通过 AWS 服务 API 和软件开发工具包支持第三方系统。APN 合作伙伴和第三方提供了许多用于监控、通知和响应的监控工具。其中一些工具包括 New Relic、Splunk、Loggly、SumoLogic 和 Datadog。

您应该保留关键的手动程序，以备在自动程序出故障时使用。

资源

请参阅以下资源，详细了解有关响应事件的 AWS 最佳实践。

视频

- AWS re:Invent 2016: Automating Security Event Response, from Idea to Code to Execution (SEC313)

文档

- 什么是 Amazon CloudWatch Events?
- How to Automatically Tag Amazon EC2 Resources in Response to API Events
- Amazon EC2 Systems Manager Automation is now an Amazon CloudWatch Events Target
- EC2 Run Command is Now a CloudWatch Events Target
- Automate remediation actions for Amazon EC2 notifications and beyond using EC2 Systems Manager Automation and AWS Health
- High-Resolution Custom Metrics and Alarms for Amazon CloudWatch

发展

发展是在一段时间内持续的改进周期。根据从您的运营活动中吸取的经验教训，实施频繁的小幅度增量变更，并评估其在带来改进方面的成效。

要持续改进您的运营，您必须能够：

- 学习、分享和改进

学习、分享和改进

要定期拿出时间进行运营活动分析、故障分析、试验和改进，这一点很重要。如果事情失败，您需要确保团队和大型工程社区能从这些失败中学习。您应该对失败进行分析，以吸取经验教训并计划改进。您需要定期与其他团队一起查看学习到的经验教训，以验证您的见解。

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，以便集中经理处理可以实现最大效益的工作。

执行事件后分析：审查影响客户的事件，确定导致这些事件的原因和预防措施。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。针对目标受众适当地传达导致因素和纠正措施。

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：设定机制，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。设定适当的机制，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。

在 AWS 上，您可以聚合所有运营活动、工作负载和基础设施的日志，以创建详细的活动历史记录。然后，您可以根据推动因素，使用 AWS 工具分析您在一段时间内的运营状况和工作负载运行状况（例如，确定趋势、将事件和活动与结果相关联，并在环境之间/跨系统进行比较和对比），以发现改进机会。

您应该使用 CloudTrail 跟踪 API 活动（通过 AWS 管理控制台、CLI、开发工具包和 API），以了解您账户中发生的情况。您可以使用 CloudTrail 和 CloudWatch 跟踪您的 AWS 开发人员工具部署活动。这将向您的 CloudWatch Logs 日志数据添加部署的详细活动历史记录及其结果。

将您的日志数据导出到 Amazon S3 进行长期存储。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。使用 Amazon Athena，借助其与 Glue 的原生集成来分析您的日志数据。使用像 Amazon QuickSight 这样的商业智能工具来直观显示、浏览并分析您的数据。

验证见解：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（包括领导）一起，对事件和运营指标执行回顾性分析。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。

寻找在所有环境（例如，开发、测试和生产环境）中改进的机会。

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，在内部和不同团队中加以利用。

您应该分享团队学到的经验教训，以增加整个组织的效益。您需要分享信息和资源，以防止出现可避免的错误并简化开发工作。这让您可以专注于交付所需的功能。

使用 AWS Identity and Access Management (IAM) 定义权限，以允许对您要在账户内和账户之间共享的资源进行受控访问。然后，您应该使用版本受控的 AWS CodeCommit 存储库来分享应用程序库、脚本编写程序、程序文档和其他系统文档。您可以共享对 AMI 的访问权限并授权跨账户使用 Lambda 函数，以此来分享您的计算标准。您还应将您的基础设施标准共享为 CloudFormation 模板。

通过 AWS API 和开发工具包，您可以集成外部和第三方工具和存储库（例如，GitHub、BitBucket 和 SourceForge）。

在分享您学到的和开发的内容时，请注意设定权限以确保共享存储库的完整性。

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。

在 AWS 上，您可以创建临时的环境副本，从而降低试验和测试的风险、工作量及成本。这些重复的环境可用于测试分析、试验、开发和测试计划改进时所得出的结论。

资源

请参阅以下资源，详细了解从经验中学习的 AWS 最佳实践。

文档

- 查询 Amazon VPC 流日志
- 使用 Amazon CloudWatch 工具监控部署
- Analyzing VPC Flow Logs with Amazon Kinesis Data Firehose, Amazon Athena, and Amazon QuickSight
- 共享 AWS CodeCommit 存储库
- 使用基于资源的策略来授予其他账户和 AWS 服务使用您的 Lambda 资源的权限
- 将 AMI 与特定 AWS 账户共享
- 将 AWS Lambda 与 Amazon SNS 结合使用

总结

卓越运营是一项持续性和迭代性的工作。

拥有共同的目标可帮助您的组织迈向成功。确保每个人都了解自己在实现业务成果方面发挥的作用，以及他们如何影响他人取得成功的能力。为您的团队成员提供支持，以便他们可以支持您的业务成果。

我们应将每个运营事件和每次故障视为改进架构运营的机会。通过了解工作负载的需求，预定义记录日常活动的运行手册以及指导解决问题的行动手册，运用 AWS 中的运营即代码功能，并保持情景感知，让您的运营做好更充分的准备，并在事件发生时能更有效地做出响应。

通过专注于随着优先级的变化进行的增量改进，以及从事件响应和回顾性分析中汲取的经验教训，您将提高活动的效率和有效性，从而实现业务的成功。

AWS 致力于帮助您构建和运行架构，以便在构建响应迅速的自适应部署的同时最大限度地提高效率。为了提升工作负载的卓越运营，您应该使用本白皮书中讨论的最佳实践。

贡献者

- Brian Carlson，Amazon Web Services Well-Architected 运营主管
- Jon Steele，Amazon Web Services 高级技术客户经理
- Ryan King，Amazon Web Services 技术项目经理
- Philip Fitzsimons，Amazon Web Services Well-Architected 高级经理

延伸阅读

如需更多帮助，请查阅以下资源：

- AWS 架构完善的框架

文档修订

日期	描述
2020 年 4 月	更新以反映新的 AWS 服务和功能以及最新的最佳实践。
2018 年 7 月	更新以反映新的 AWS 服务和功能以及最新参考。
2017 年 11 月	首次发布