试 CI/CD 金石：你的管道是否完全是 CI/CD？

AWS 规范性指导

AWS 规范性指导: 试 CI/CD 金石：你的管道是否完全是 CI/CD？

Table of Contents

试 CI/CD 金石：你的管道是否完全是 CI/CD？

Steven Guggenheimer 和 Ananya Koduri，Amazon Web Services（AWS）

2023 年 8 月（文档历史记录）

您的管线是否已实现自动化？ 这是一个简单的问题，但许多组织给出的答案却过于片面。答案蕴含的复杂性远非是或否所能概括。

技术创新日新月异，有时组织可能很难跟上步伐。这项新技术是昙花一现，还是会成为下一个主流？我应该彻底改革现有的做法，还是应该静观其变？ 通常，当明确某项技术是大势所趋时，企业往往已陷入被动追赶的境地。持续集成和持续交付（CI/CD）已成为长期趋势，但并非一直如此。许多人花了很长时间才被说服，有些人仍持观望态度。

CI/CD is the process of automating the source, build, test, staging, and production stages of the software release process, and it is commonly described as a pipeline. Today, the cost savings and speed of CI/CD自动化使大多数组织相信了它的价值。但过渡到这种新方法并非易事。您需要确保员工接受过正确的培训，需要升级一些资源，然后需要不断测试、测试、再测试。工作量巨大。大多数情况下，您需要逐步进行这些变更，以帮助组织适应。

本文档的目的是定义拥有完整 CI/CD 流程意味着什么。它提供了一种工具来评估您自身的流程，并为尚未实现的流程提供前进方向。这条前进的道路很少能一蹴而就。相关流程很复杂，取决于许多因素，包括当前员工的技能集以及当前基础设施需求。我们建议您分清轻重缓急，小幅、循序渐进地进行改进。

目标

实施本指南中的建议可能带来以下潜在优势：

- 效率 — 全面 CI/CD 部署过程可以减少复杂性、减少工作量，并减少花费在调试、执行手动流程和维护上的无数时间。有关更多信息，请参阅持续交付的优势。根据一篇TechAhead 博客文章，该 CI/CD 流程的实施估计可以节省20%的时间、精力和资源。
- 成本削减：根据 Forbes Insight 报告，“四分之三的高管认为，相较于新项目开发或新举措，花费在持续维护和管理方面的时间、金钱和资源，正在影响其组织的整体竞争力。”开发周期越短，您的组织就越有可能在正确的时间 time-to-market实现雄心勃勃的目标并抓住合适的机会。
- 速度 — 通常，完整的CI/CD pipeline is able to release software changes to customers within a few hours. Especially in cases with quick fault isolations and small patch pushes, the CI/CD管道有助于缩短平均恢复时间 (MTTR)。有关更多信息，请参阅 Reducing MTTR。

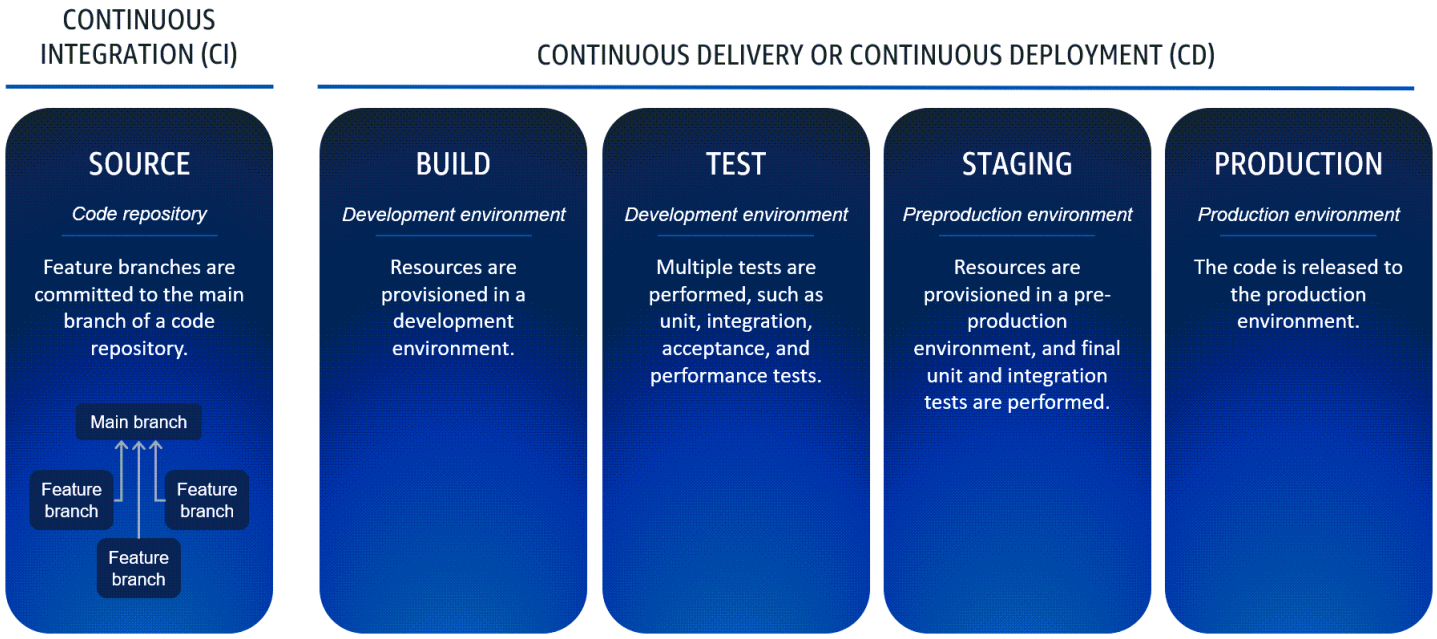
- 安全 — 完全 CI/CD 管道还可以减少攻击的可能入口点并降低人为错误的风险，从而保护发布过程。全自动 CI/CD 管道带来的安全优势有助于避免数据泄露、服务中断等造成的代价高昂的后果。
- 降低流失率：当开发人员可以花更多的时间用来创建出色的功能，而不是陷入无休止的维护和调试循环时，他们会更加满意。对于组织而言，这意味着能够更长时间地吸引和留住顶尖人才。
- 卓越的代码质量 — 开发人员以小批量方式将代码发布到共享存储库中，这使他们能够进行并行测试（BrowserStack 博客文章）。他们并非各自为战，而是经常与团队共享构建成果，并协作识别关键错误。这为开发人员提供了支持，有助于防止不良代码进入生产环境。来自开发人员同行的支持有助于发布高质量版本，并推动组织发展。
- 维护：维护和更新是制作一款出色产品的关键部分。但是，不要在流量高峰时段关闭系统。您可以使用 CI/CD 管道在低使用率时段执行维护，从而最大限度地减少停机时间和对性能的影响。

了解 CI/CD

持续集成和持续交付（CI/CD）是自动执行软件发布生命周期的过程。在某些情况下，D CI/CD 也可以表示部署。持续交付和持续部署之间的区别体现在发布对生产环境的变更时。对于持续交付，在推动对生产环境的变更之前需要手动批准。持续部署的特点是可以不间断地贯穿整个管线，不需要显式批准。由于此策略讨论的是一般 CI/CD 概念，因此所提供的建议和信息适用于持续交付和持续部署方法。

CI/CD automates much or all of the manual processes traditionally required to get new code from a commit into production. A CI/CD pipeline encompasses the source, build, test, staging, and production stages. In each stage, the CI/CD pipelines provisions any infrastructure that is needed to deploy or test the code. By using a CI/CDpipeline 中，开发团队可以对代码进行更改，然后对其进行自动测试并推送到部署中。

让我们回顾一下基本 CI/CD 过程，然后再讨论一些有意或无意地偏离每个阶段的完整CI/CD. The following diagram shows the CI/CD阶段和活动的方式。



关于持续集成

持续集成在代码存储库中进行，例如 GitHub 中的 Git 存储库。您将一个主分支视为代码库的真实来源，为功能开发创建短期分支。当您准备好将功能部署到上层环境时，可以将该功能分支集成到主分支中。功能分支永远不会直接部署到上层环境。有关更多信息，请参阅本指南中的基于主干的方法。

持续集成流程

1. 开发人员从主分支创建一个新分支。
2. 开发人员在本地进行更改、构建和测试。
3. 更改准备就绪后，开发人员会创建一个以主分支为目标的拉取请求（GitHub 文档）。
4. 代码将进行审查。
5. 当代码获得批准后，其会合并到主分支中。

关于持续交付

持续交付在开发环境和生产环境等隔离环境中进行。每种环境中进行的操作可能有所不同。通常，第一个阶段之一用于对管线本身进行更新，然后再继续。部署的最终结果是，每个环境均更新为最新的更改。用于构建和测试的开发环境的数量也各不相同，但我们建议您使用至少两个。在管线中，每个环境都按其重要性顺序进行更新，最后更新最重要的环境，即生产环境。

持续交付流程

管线的持续交付部分通过以下方式启动：从源存储库的主分支提取代码并将其传递到构建阶段。存储库的基础设施即代码（IaC）文档概述了在每个阶段执行的任务。尽管使用 IaC 文档并非强制要求，但强烈建议使用 IaC 服务或工具，例如 AWS CloudFormation 或 AWS Cloud Development Kit (AWS CDK)。最常见的步骤包括：

1. 单元测试
2. 代码构建
3. 资源预调配
4. 集成测试

如果在管线的任何阶段出现任何错误或任何测试失败，则当前阶段将回滚到其之前的状态，并且管线将终止。后续的更改必须从代码存储库中开始并完成整个 CI/CD 过程。

CI/CD 管道测试

部署管线中经常提到的两种类型的自动化测试是单元测试和集成测试。但是，有许多类型的测试可以在代码库和开发环境中运行。AWS Deployment Pipeline Reference Architecture 定义了以下类型的测试：

- 单元测试：这些测试构建并运行应用程序代码，以验证其性能是否符合预期。它们模拟代码库中使用的所有外部依赖关系。单元测试工具的示例包括 JUnitJest 和 py test。

- 集成测试：这些测试通过对预调配的测试环境进行测试来验证应用程序是否满足技术要求。集成测试工具的示例包括 Cucumber、vRest NG 和 integ-tests（适用于 AWS CDK）。
- 验收测试：这些测试通过对预调配的测试环境进行测试来验证应用程序是否满足用户要求。验收测试工具的示例包括 Cypress 和 Selenium。
- 综合测试：这些测试在后台持续运行，以生成流量并验证系统是否正常运行。合成测试工具的示例包括 Amazon Synth CloudWatch etics 和 Dynatr ace 合成监控。
- 性能测试：这些测试模拟生产环境的容量。其确定应用程序是否满足性能要求，并将指标与过去的性能进行比较。性能测试工具的示例包括 Apache JMeter、Locust 和 Gatling。
- 韧性测试：也称为混沌测试，这些测试将故障注入环境中，以识别风险区域。然后，将注入故障的时间段与没有故障的时间段进行比较。韧性测试工具的示例包括 AWS Fault Injection Service 和 Gremlin。
- 静态应用程序安全测试（SAST）：这些测试分析代码中是否存在安全违规行为，例如 SQL 注入或跨站脚本攻击（XSS）。SAST 工具的示例包括亚马逊 CodeGuruSonarQube、和 Check marx。
- 动态应用程序安全测试（DAST）：这些测试也称为渗透测试或 pen 测试。其可以识别漏洞，例如预调配测试环境中的 SQL 注入或 XSS。DAST 工具的示例包括 Zed 攻击代理 (ZAP) 和 HCL。AppScan有关更多信息，请参阅 Penetration Testing。

并非所有完全运行的 CI/CD 管道都能运行所有这些测试。但是，管线至少应对代码库运行单元测试和 SAST 测试，并对测试环境运行集成测试和验收测试。

CI/CD 管道指标

根据AWS 部署管道参考架构，您至少应跟踪 CI/CD 管道的以下四个指标：

- 前置时间：单次提交到完全进入生产环境所需的平均时间。我们建议根据您的使用案例，将前置时间设定在 1 小时到 1 天之间。
- 部署频率：给定时间段内的生产部署次数。我们建议根据您的使用案例，将部署频率设定为每天多次到每周两次之间。
- 平均故障间隔时间（MTBF）：从成功管线启动到故障管线启动之间的平均时间。我们建议设定尽可能高的 MTBF。有关更多信息，请参阅 Increasing MTBF。
- 平均恢复时间（MTTR）：从故障管线启动到下一个成功管线启动之间的平均时间。我们建议设定尽可能低的 MTTR。有关更多信息，请参阅 Reducing MTTR。

这些指标可以帮助团队跟踪其实现完全 CI/CD 的进度。团队应与组织的利益相关者就最佳目标应该是什么进行公开讨论。不同组织，甚至不同团队的情况和需求差异很大。

重要的是要记住，快速、剧烈的变化通常会增加出现问题的风险。设定目标，力求实现小幅、渐进式的改进。整个 CI/CD 管道的常见最佳交货时间少于 3 小时。如果一个团队开始时的前置时间为 5.2 天，则目标应为每隔几周缩短一天。在该团队的前置时间达到一天或更短时间后，其可以保持在该水平数月，仅当团队和组织利益相关者认为有必要时，才转向更为激进的前置时间目标。

CI/CD 流程的不同之处有多大

CI/CD 管道使用基于主干的现代工作流程，在这种工作流程中，开发人员将频繁的小更新合并到主分支（或主干）中，该分支或主干是通过管道的 CD 部分构建和测试的。 CI/CD 该工作流程取代了 Gitflow 工作流程，其中的开发和发布分支通过发布计划分开。在许多组织中，Gitflow 仍然是一种流行的版本控制和部署方法。但是，它现在被认为是遗留的，要集成到 CI/CD 管道中可能具有挑战性。

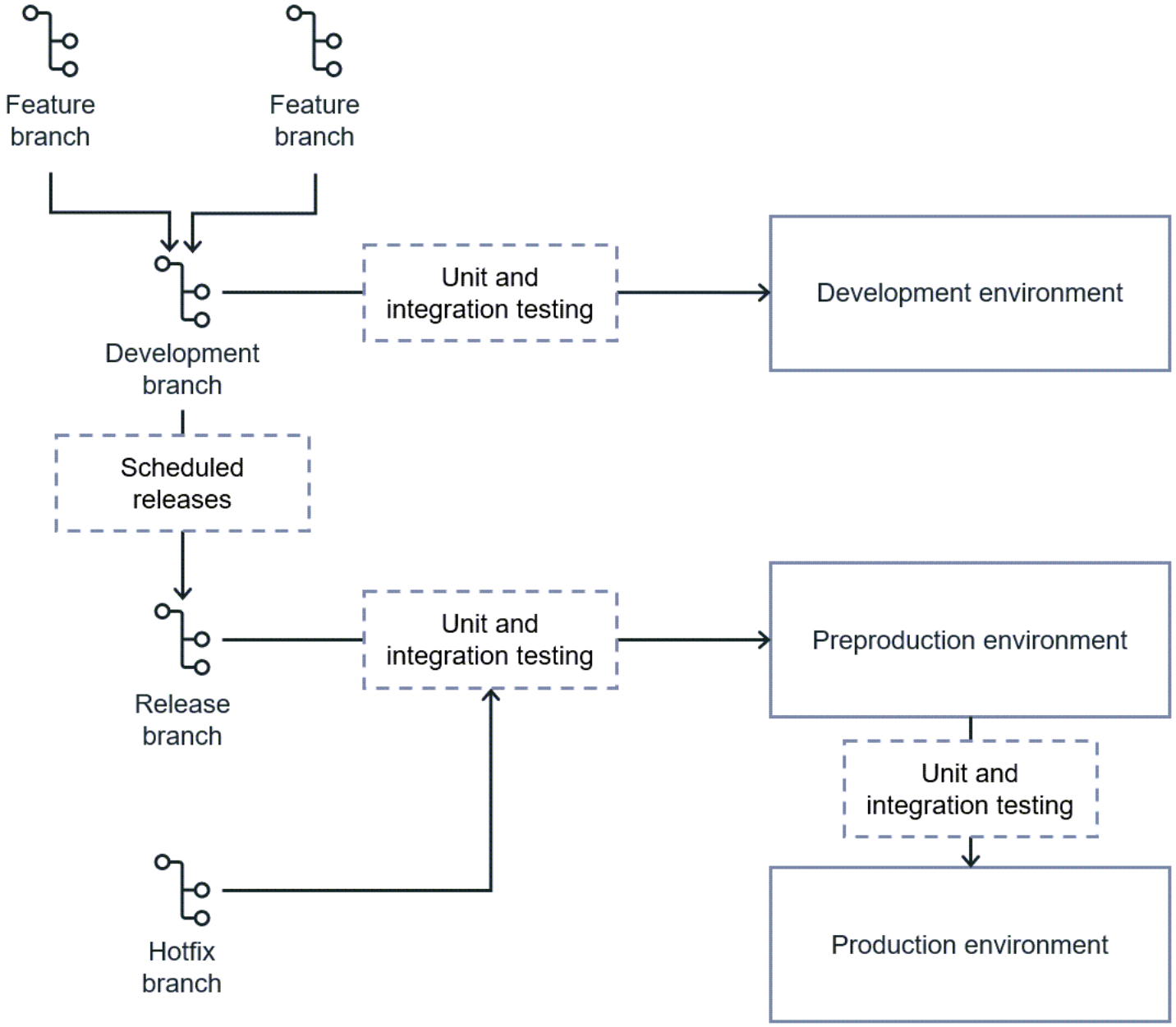
对于许多组织而言，从 Gitflow 工作流程到基于主干的工作流程的转换尚未完成，导致它们在迁移过程中停滞不前，从未完全迁移到 CI/CD。不知何故，它们的管线仍然保留旧版工作流程的某些残余，卡在过去与现在之间的过渡状态。查看 Git 工作流程中的差异，然后了解使用旧版工作流程对以下方面有何影响：

- 环境完整性
- 发行版
- 安全性

为了更轻松地识别现代配置中的旧版 Git 工作流程残余，让我们比较一下 Gitflow 与基于主干的现代方法。

Gitflow 方法

下图显示 Gitflow 工作流程。Gitflow 方法使用多个分支同时跟踪多个不同版本的代码。您可以计划在未来某个时间点发布应用程序更新，而开发人员仍对当前版本的代码继续开发。基于主干的存储库可以使用功能标志来实现此目的，但 Gitflow 默认内置此功能。

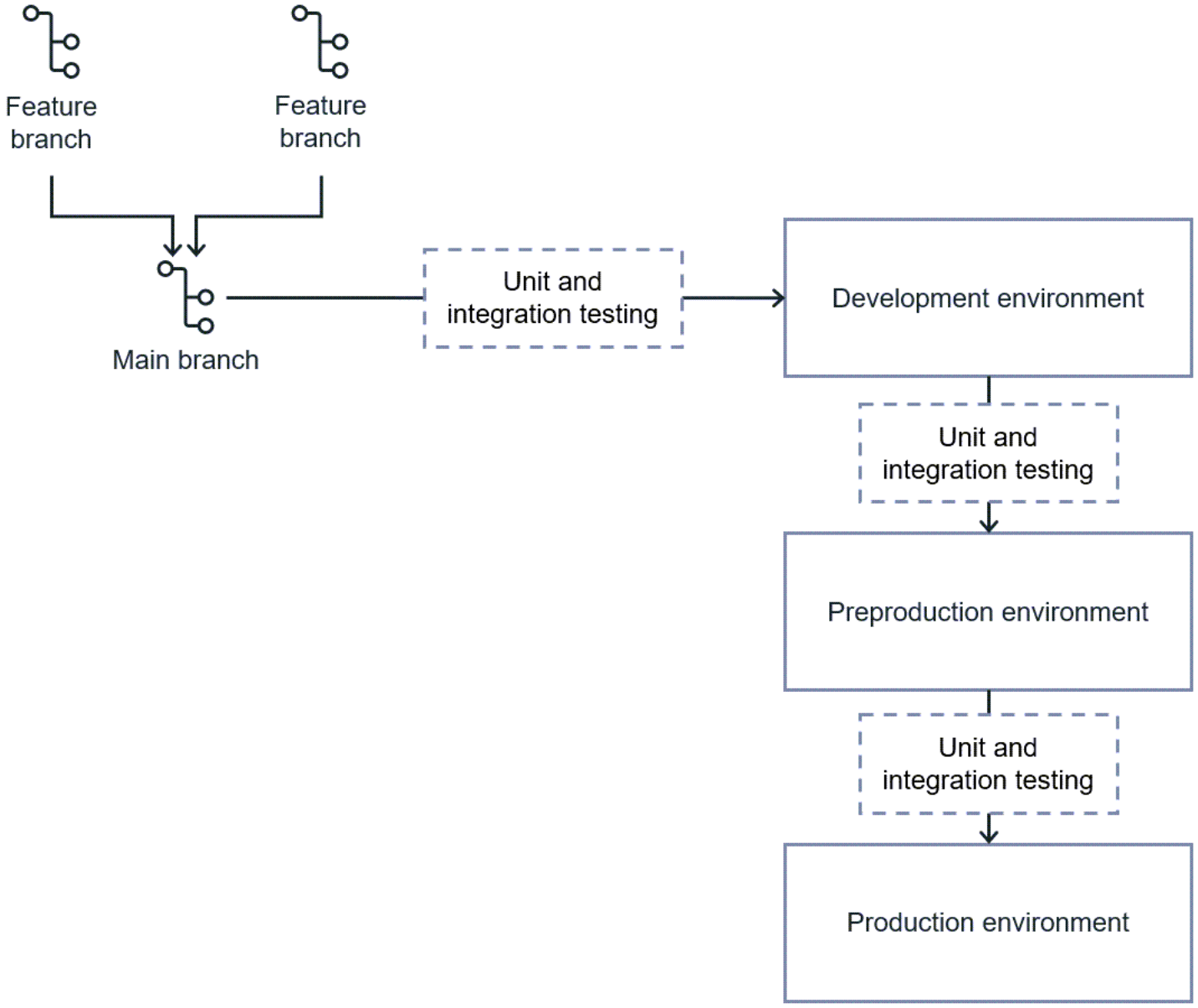


Gitflow 方法的结果之一是应用程序环境通常不同步。在标准 Gitflow 实施中，开发环境反映了代码的当前状态，而预生产和生产环境则冻结在最新版本的代码库状态。

当生产环境中出现缺陷时，情况会变得复杂，因为如果不暴露未发布的功能，开发人员使用的代码库就无法合并到生产环境中。Gitflow 处理这种情况的方式是使用修补程序。从发布分支创建修补程序分支，然后直接部署到上层环境。修补程序分支随后合并到开发分支，以使代码保持最新。

基于主干的方法

下图显示基于主干的工作流程。在基于主干的工作流程中，开发人员在功能分支中本地构建和测试功能，然后将这些更改合并到主分支中。然后，按顺序将主分支构建到开发、预生产和生产环境。每个环境之间都会进行单元测试和集成测试。



使用此工作流程，所有环境都运行相同的代码库。上层环境不需要修补程序分支，因为您可以在主分支中实施更改，而无需暴露未发布的功能。始终假定主分支是稳定、没有缺陷且随时可以发布。这可以帮助您将其集成为 CI/CD 管道的来源，该管道可以在管道中的所有环境中自动测试和部署您的代码库。

基于主干的方法对环境完整性的优势

正如许多开发人员所知道的那样，一次代码更改有时可能会产生蝴蝶效应（American Scientist 文章），即一个看似无关的微小偏差会引发连锁反应，导致意想不到的结果。开发人员随后必须进行全面调查以找出根本原因。

科学家进行实验时，会将受试者分为两组：实验组和对照组。目的是使实验组和对照组除了实验测试内容之外完全相同。如果实验组中出现了对照组中没有出现的情况，那么唯一的原因就是测试的内容。

将部署中的变更视为实验组，将每个环境视为独立的对照组。仅当下层环境的对照组与上层环境的对照组完全相同时，下层环境的测试结果才是可靠的。环境偏差越大，在上层环境中发现缺陷的可能性就越大。换句话说，如果代码变更将在生产环境中失败，我们宁愿其先在测试版中失败，这样它们就永远无法投入生产。这就是为什么应尽一切努力使每个环境（从最低测试环境到生产环境本身）保持同步的原因。这称为环境完整性。

任何完整 CI/CD 流程的目标都是尽早发现问题。使用基于主干的方法保留环境完整性，几乎可以消除对修补程序的需求。在基于主干的工作流程中，问题很少会首先出现在生产环境中。

在 Gitflow 方法中，将修补程序直接部署到上层环境后，再将其添加到开发分支中。这会保留该修复程序以备未来版本使用。但是，该修补程序是直接根据应用程序当前状态开发和测试的。即使该修补程序在生产环境中完美运行，当其与开发分支中的较新的功能交互时，也有可能出现问题。由于通常不希望为修补程序部署修补程序，因此这会导致开发人员花费额外的时间尝试使该修补程序重新适配开发环境。在许多情况下，这可能导致巨额技术债务，并降低开发环境的总体稳定性。

环境中发生故障时，所有更改都会回滚，以使环境恢复到其以前的状态。对代码库的任何更改都应从第一阶段重新启动管线。当生产环境中确实出现问题时，修复也应该通过整个管线进行。与使用此方法可避免的问题相比，经过下层环境所花的额外时间通常可以忽略不计。由于下层环境的全部目的是在错误进入生产环境之前将其捕获，因此通过 Gitflow 方法绕过这些环境是一种效率低下且不必要的风险。

基于主干方法的发布优势

在旧版工作流程中，开发人员正在开发的应用程序状态可能包含多个尚未投入生产的未发布功能，而这往往是导致需要修补程序的原因之一。生产环境和开发环境仅当计划发布时才会同步，然后它们会立即再次开始出现差异，直到下一次计划发布。

在完整的 CI/CD 流程中可以安排发布时间。您可以使用功能标志来延迟将代码发布到生产环境中。但是，完整的 CI/CD 流程使计划发布变得不必要，从而提高了灵活性。毕竟，持续是 CI/CD 中的一个关键词，这表明变更准备就绪后就会发布。避免维护几乎总是与下层测试环境不同步的独立发布环境。

如果管线并非完全 CI/CD，则上层和下层环境之间的差异通常发生在分支级别。开发人员在开发分支中工作，并维护一个独立的发布分支，该分支仅在计划发布时才会更新。随着发布分支与开发分支出现差异，可能会引起其他复杂情况。

除了环境不同步之外，由于开发人员在开发分支上工作并习惯于远超生产环境的应用程序状态，因此每当生产环境出现问题时，他们都必须重新调整以适应生产状态。开发分支的状态可能有许多领先于生产环境的功能。当开发人员每天在该分支中工作时，很难记住哪些功能已经发布到生产环境，哪些功能尚未发布到生产环境。这增加了在修复其他错误的过程中引入新错误的风险。其结果是陷入看似无休止的修复循环，导致延长时间线，使功能发布延迟数周、数月甚至数年。

基于主干方法的安全优势

完整的 CI/CD 流程提供了一种完全自动化的单一事实来源部署方法。管线只有一个入口点。软件更新从一开始就进入管线，然后按原样从一个环境传递到另一个环境。如果在管线的任何阶段发现问题，则修复该问题的代码更改必须经过相同的流程并从第一阶段开始。减少管线的入口点同时减少了将漏洞引入管线的可能途径。

此外，由于入口点是距离生产环境最远的点，因此这显著降低了漏洞到达生产环境的可能性。如果您在完全 CI/CD 管线中实施手动批准流程，则仍可对更改是否升级到下一个环境作出允许执行或不执行的决策。决策者不一定是部署变更的同一个人。这将代码更改部署者与这些更改批准者的责任分开。这也使得技术水平较低的组织领导者更容易胜任批准者的角色。

最后，单一入口点可帮助您将对生产环境用户界面（UI）控制台的写入访问权限限制为少数用户甚至零用户。通过减少可在控制台中进行手动更改的用户数量，可以降低发生安全事件的风险。在传统工作流程中，在生产环境中手动管理控制台的能力比使用 CI/CD 自动化方法更为必要。这些手动更改更难跟踪、审查和测试。进行这些更改通常是为了节省时间，但从长远来看，其会给项目增加大量的技术债务。

控制台安全问题不一定是由恶意行为者造成的。控制台中出现的许多问题都是意外。意外安全漏洞非常普遍，而这导致了零信任安全模式的兴起。该模式的部分理念是，当即使内部员工也拥有尽可能少的访问权限（也称为最低权限）时，发生安全事故的可能性就会降低。通过将所有流程都限制在自动化管线内来保持生产环境的完整性，从而几乎消除与控制台相关的安全问题风险。

CI/CD 管线的试金石

在化学中，石蕊试纸是一种涂有特殊红色或蓝色染料的薄纸条，用于测定物质的酸碱度。酸性物质会使蓝色的石蕊试纸变为红色，碱性物质会使红色的石蕊试纸变为蓝色，而中性物质则不会改变试纸的颜色。

石蕊试纸测定酸碱度的方法是通过测量物质的 pH 值。如果 pH 值高于 8，则为酸性；如果低于 5，则为碱性；如果在 5 到 8 之间，则为中性。同样，CI/CD 试金石可以帮助您衡量管线的 CI/CD 水平。

测试您的管线是否为完全 CI/CD

1. 从 0 分开始。
2. 回答以下每个问题，每次回答为是则分数加 1：
    - 我们的存储库是否都只有一个用于部署到环境的主分支？
    - 我们是否经常将代码提交到主分支并避免长时间运行功能分支？
    - 我们的管线是否只有单一入口点？ 换句话说，我们的管线是否只从每个存储库中提取一次代码？
    - 我们是否有多个部署环境？
    - 当管线未运行时，我们的上层和下层环境通常是否保持同步？
    - 我们是否会在部署前对代码进行测试？
    - 在升级到下一个环境之前，我们是否会在环境中运行测试？
    - 发生故障后，我们的管线是否会进行完全回滚并退出？
    - 从故障中恢复时，我们的管线是否会从第一步重新开始？
    - 我们修复生产环境中的错误所遵循的流程是否与将功能发布到生产环境时的流程相同？
    - 我们是否使用某种形式的基础设施即代码（IaC）模板来部署代码？
3. 回答以下每个问题，每次回答为否则分数加 1：
    - 我们是否曾经从主分支以外的分支直接部署到部署环境中？
    - 我们是否曾经直接从任何分支部署到上层环境或生产环境？
    - 我们是否经常在上层环境中发现下层环境中不存在的错误？
    - 我们是否曾经在部署过程中绕过下层环境？
    - 我们是否会等到计划发布时间才部署到生产环境？
    - 我们是否定期在生产环境的控制台中进行更新？

- 是否必须在生产环境的控制台中完成任何手动部署步骤才能完成部署？
- 是否有多人拥有对生产环境的写入访问权限？
- 是否有超过五人拥有对生产环境的写入访问权限？

4. 将您的分数除以 2，这就是您管线的 CI/CD 分数。
5. 将管线的 CI/CD 分数与下表进行比较，以确定管线的 CI/CD 级别。

CI/CD 分数	CI/CD 级别
9.5 或以上	完全 CI/CD
8-9	大部分 CI/CD
5–7	Neutral
低于 5	非 CI/CD

如果您的分数低于 8，我们建议您设定一个目标，逐步提升到下一个级别。该目标实现后，产品利益相关者应评测是否以及何时应设定新目标。本练习的目的并非一定要更改您的管线，而是让人们了解完全 CI/CD 部署流程是什么样的，以及您的管线目前在该谱系中的位置。

CI/CD 管线的最佳实践

以下是完全 CI/CD 管线的最佳实践：

- 保护生产环境：由于使用 IaC 几乎可以完成账户和环境维护所需的一切，因此重要的是通过限制控制台和编程访问尽一切努力来保护生产环境。我们建议仅允许少数用户（甚至零用户）访问。通过 AWS CloudFormation 部署 IaC 时，用户需要有限的权限。大多数权限都通过服务角色分配给 CloudFormation 服务。有关更多信息，请参阅 CloudFormation 文档中的服务角色和 Implementing policies for least-privilege permissions for AWS CloudFormation。
- 为每个环境创建单独的账户：通过为每个环境指定一个单独的账户，您可以简化部署过程并在账户级别创建精细的访问控制。当多个环境共享资源时，会降低环境作为独立单元的完整性。最好保持环境同步且彼此独立。这对于生产环境来说尤为重要，因为该账户中的所有内容都应被视为生产资源。
- 将个人身份信息（PII）限制在生产环境：无论是为了安全还是避免责任风险，都要尽可能保护 PII。在下层环境中，尽可能使用匿名化或样本数据，而不是复制生产环境中潜在的敏感数据。
- 查看存储库中的代码：完全 CI/CD 流程将管线的入口点减少为一个点，并且该单点应该得到保护。出于此原因，建议您在将功能分支合并到主分支之前，需要进行多次代码审查。这些代码审查可由任何合格的团队成员进行，但至少应有一名高级成员进行审查。该代码应由审查者严格进行测试。毕竟，修复管线中问题的最佳方法是避免引入问题。此外，在合并之前，请务必解决所有审查者提出的所有意见。解决方法可以是简单地解释为什么不需要更改，但是解决所有意见是一项重要的额外检查，有助于防止将问题引入管线。
- 进行小规模且频繁的合并：为了充分利用持续集成，最好将本地变更也持续推送到管线中。毕竟，如果本地环境也能跟上开发环境的步伐，那么保持同步对开发环境更有利。

有关 CI/CD 管线的更多最佳实践，请参阅《在 AWS 上练习持续集成和持续交付》中的最佳实践汇总。

常见问题解答

有哪些关键指标表明我的部署流程并非完全 CI/CD？

最常见的指标是管线中有多个存储库分支代表不同的环境。完整 CI/CD 流程中的存储库使用基于主干的工作流程，其中一个分支充当该存储库部署的单一事实来源。有关更多信息，请参阅 基于主干的方法。其他指标包括除简单的执行或不执行决策以外的手动部署步骤、使用修补程序和计划发布。

如果我想使用完整的 CI/CD 流程，但仍想在特定时间点安排某些功能的发布，该怎么办？

这通常需要使用功能标志来完成。在此流程中，部署仍会持续进行，但会在代码中使用条件闭包来隐藏某些功能，直到需要发布时才显示。

如果我的部署流程中的某些步骤无法自动化，该怎么办？

完整 CI/CD 管道的目标之一是最大限度地减少对手动流程的需求，但肯定存在可能需要手动流程的潜在用例。实际上，查阅应用程序日志等只读流程通常可在生产环境中完成，风险极低。但是，强烈建议您将生产时的手动写入操作视为绝对最后使用的手段。

如果我的技术人员对传统工作流程比对完整 CI/CD 流程更满意，该怎么办？

技术人员通常会抵制重大变更，尤其是当曾经是最佳实践的做法被更新的方法取代时。技术发展日新月异，改进方案层出不穷。尽管保持一定的怀疑精神对技术人员来说是一种优良品质，但对变更持开放态度也同样重要。面对持怀疑态度的员工时，不要操之过急，因为他们需要在系统变更实施之前对其进行管理。关键在于防止持怀疑态度的人员永远固步自封。

如果我的环境有多个账户怎么办？ 我还能使用完整的 CI/CD 流程吗？

是的，实际上，建议为每个环境使用单独的帐户。 有关在不同账户中激活阶段的管道的更多信息，请参阅在中 CodePipeline 创建使用其他 AWS 账户账户资源的管道。

后续步骤

使用CI/CD 管线的试金石章节来评估组织中的 DevOps 流程。确定您的流程是否为完全 CI/CD。如果不是，请决定是否需要改进流程，以充分利用 CI/CD 部署的优势。

如何判断何时完成？ 答案是，许多组织实际上从未完成。他们会在中途停下来，留在适合自身使用案例的某个阶段。尽管完全 CI/CD 管线是最理想的情况，但它在很大程度上取决于组织状况和决策背后的利益相关者。利益相关者必须决定 CI/CD 实施的哪个阶段最适合其使用案例，以及如何最理想地将进展对应后续阶段。

有关设计和构建 CI/CD 管线的更多信息，请参阅资源。

资源

AWS 文档和参考资料

- AWS Deployment Pipeline Reference Architecture
- 什么是持续交付？
- 在 AWS（AWS 白皮书）上练习持续集成和持续交付
- 在（上AWS 手教程 AWS）上设置 CI/CD 管道
- 在其中创建 AWS 区域 不支持的管道 AWS CodePipeline（AWS 规范性指导）
- 部署管道参考架构和参考实现（AWS 博客文章）

服务和工具

- CI/CD 试金石
- AWS Cloud Development Kit (AWS CDK)
- AWS CloudFormation
- AWS CodePipeline

文档历史记录

下表介绍了本指南的一些重要更改。如果您希望收到有关未来更新的通知，可以订阅 RSS 源。

变更	说明	日期
初次发布	—	2023 年 8 月 25 日

AWS 规范性指导词汇表

以下是 AWS 规范性指导提供的策略、指南和模式中的常用术语。若要推荐词条，请使用术语表末尾的提供反馈链接。

数字

7 R

将应用程序迁移到云中的 7 种常见迁移策略。这些策略以 Gartner 于 2011 年确定的 5 R 为基础，包括以下内容：

- 重构/重新架构：充分利用云原生功能来提高敏捷性、性能和可扩展性，以迁移应用程序并修改其架构。这通常涉及到移植操作系统和数据库。示例：将本地 Oracle 数据库迁移到 Amazon Aurora PostgreSQL 兼容版。
- 更换平台：将应用程序迁移到云中，并进行一定程度的优化，以利用云功能。示例：将本地 Oracle 数据库迁移到 AWS 云中的 Amazon Relational Database Service（Amazon RDS）for Oracle。
- 重新购买：转换到其他产品，通常是从传统许可转向 SaaS 模式。示例：将客户关系管理（CRM）系统迁移到 Salesforce.com。
- 重新托管（直接迁移）：将应用程序迁移到云中，无需进行任何更改即可利用云功能。示例：将本地 Oracle 数据库迁移到 AWS 云中 EC2 实例上的 Oracle。
- 重新放置（虚拟机监控器级直接迁移）：将基础设施迁移到云中，无需购买新硬件、重写应用程序或修改现有操作。您将服务器从本地平台迁移到同一平台的云服务中。示例：将Microsoft Hyper-V应用程序迁移到 AWS。
- 保留（重访）：将应用程序保留在源环境中。其中可能包括需要进行重大重构的应用程序，并且您希望将工作推迟到以后，以及您希望保留的遗留应用程序，因为迁移它们没有商业上的理由。
- 停用：停用或删除源环境中不再需要的应用程序。

A

ABAC

请参阅基于属性的访问控制。

抽象服务

请参阅托管服务。

ACID

请参阅原子性、一致性、隔离性、持久性。

主动-主动迁移

一种数据库迁移方法，在这种方法中，源数据库和目标数据库保持同步（通过使用双向复制工具或双写操作），两个数据库都在迁移期间处理来自连接应用程序的事务。这种方法支持小批量、可控的迁移，而不需要一次性割接。它比主动-被动迁移更灵活，但工作量更大。

主动-被动迁移

一种数据库迁移方法，在这种方法中，源数据库和目标数据库保持同步，但在将数据复制到目标数据库时，只有源数据库处理来自连接应用程序的事务。目标数据库在迁移期间不接受任何事务。

聚合函数

一种 SQL 函数，它对一组行进行操作并计算该组的单个返回值。聚合函数的示例包括 SUM 和 MAX。

AI

请参阅人工智能。

AIOps

请参阅人工智能运营。

匿名化

永久删除数据集中个人信息的过程。匿名化可以帮助保护个人隐私。匿名化数据不再被视为个人数据。

反模式

一种用于解决反复出现的问题的常用解决方案，而在这类问题中，此解决方案适得其反、无效或不如替代方案有效。

应用程序控制

一种安全方法，仅允许使用经批准的应用程序，以帮助保护系统免受恶意软件的侵害。

应用程序组合

有关组织使用的每个应用程序的详细信息的集合，包括构建和维护该应用程序的成本及其业务价值。这些信息是产品组合发现和分析过程的关键，有助于识别需要进行迁移、现代化和优化的应用程序并确定其优先级。

人工智能（AI）

计算机科学领域致力于使用计算技术执行通常与人类相关的认知功能，例如学习、解决问题和识别模式。有关更多信息，请参阅什么是人工智能？

人工智能操作 (AIOps)

使用机器学习技术解决运营问题、减少运营事故和人为干预以及提高服务质量的过程。有关如何在 AIOps AWS 迁移策略中使用的更多信息，请参阅操作集成指南。

非对称加密

一种加密算法，使用一对密钥，一个公钥用于加密，一个私钥用于解密。您可以共享公钥，因为它不用于解密，但对私钥的访问应受到严格限制。

原子性、一致性、隔离性、持久性（ACID）

一组软件属性，即使在出现错误、电源故障或其他问题的情况下，也能保证数据库的数据有效性和操作可靠性。

基于属性的访问权限控制（ABAC）

根据用户属性（如部门、工作角色和团队名称）创建精细访问权限的做法。有关更多信息，请参阅 AWS Identity and Access Management (I AM) 文档 AWS中的 AB AC。

权威数据来源

存储主要数据版本的位置，被认为是最可靠的信息源。您可以将数据从权威数据来源复制到其他位置，以便处理或修改数据，例如对数据进行匿名化、编辑或假名化。

可用区

中的一个不同位置 AWS 区域 ，不受其他可用区域故障的影响，并向同一区域中的其他可用区提供低成本、低延迟的网络连接。

AWS 云采用框架 (AWS CAF)

该框架包含指导方针和最佳实践 AWS ，可帮助组织制定高效且有效的计划，以成功迁移到云端。AWS CAF将指导分为六个重点领域，称为视角：业务、人员、治理、平台、安全和运营。业务、人员和治理角度侧重于业务技能和流程；平台、安全和运营角度侧重于技术技能和流程。例如，人

员角度针对的是负责人力资源（HR）、人员配置职能和人员管理的利益相关者。从这个角度来看，AWS CAF 为人员发展、培训和沟通提供了指导，以帮助组织为成功采用云做好准备。有关更多信息，请参阅 AWS CAF 网站和 AWS CAF 白皮书。

AWS 工作负载资格框架 (AWS WQF)

一种评估数据库迁移工作负载、推荐迁移策略和提供工作估算的工具。 AWS WQF 包含在 AWS Schema Conversion Tool (AWS SCT) 中。它用来分析数据库架构和代码对象、应用程序代码、依赖关系和性能特征，并提供评测报告。

B

恶意机器人

一种旨在扰乱或伤害个人或组织的机器人。

BCP

请参阅业务连续性计划。

行为图

一段时间内资源行为和交互的统一交互式视图。您可以使用 Amazon Detective 的行为图来检查失败的登录尝试、可疑的 API 调用和类似的操作。有关更多信息，请参阅 Detective 文档中的行为图中的数据。

大端序系统

一个先存储最高有效字节的系统。另请参阅字节顺序。

二进制分类

一种预测二进制结果（两个可能的类别之一）的过程。例如，您的 ML 模型可能需要预测诸如“该电子邮件是否为垃圾邮件？”或“这个产品是书还是汽车？”之类的问题

bloom 筛选条件

一种概率性、内存高效的数据结构，用于测试元素是否为集合的成员。

蓝/绿部署

一种部署策略，您可以创建两个独立但完全相同的环境。在一个环境中运行当前应用程序版本（蓝色），在另一个环境中运行新应用程序版本（绿色）。此策略可帮助您在影响最小的情况下快速回滚。

自动程序

一种通过互联网运行自动任务并模拟人类活动或交互的软件应用程序。有些机器人是有用或有益的，例如在互联网上索引信息的 Web 爬网程序。还有一些被称为恶意机器人的机器人，其目的是扰乱或伤害个人或组织。

僵尸网络

被恶意软件感染并受单方（称为僵尸网络控制者或僵尸网络操作者）控制的僵尸网络。僵尸网络是最著名的扩展机器人及其影响力的机制。

分支

代码存储库的一个包含区域。在存储库中创建的第一个分支是主分支。您可以从现有分支创建新分支，然后在新分支中开发功能或修复错误。为构建功能而创建的分支通常称为功能分支。当功能可以发布时，将功能分支合并回主分支。有关更多信息，请参阅关于分支（GitHub 文档）。

紧急（break-glass）访问

在特殊情况下，通过批准的流程，用户 AWS 账户 可以快速访问他们通常没有访问权限的内容。有关更多信息，请参阅 AWS Well-Architected Guidance 中的 Implement break-glass procedures 指示器。

棕地策略

您环境中的现有基础设施。在为系统架构采用棕地策略时，您需要围绕当前系统和基础设施的限制来设计架构。如果您正在扩展现有基础设施，则可以将棕地策略和全新策略混合。

缓冲区缓存

存储最常访问的数据的内存区域。

业务能力

企业如何创造价值（例如，销售、客户服务或营销）。微服务架构和开发决策可以由业务能力驱动。有关更多信息，请参阅在 AWS上运行容器化微服务白皮书中的围绕业务能力进行组织部分。

业务连续性计划（BCP）

一项计划，旨在应对大规模迁移等破坏性事件对运营的潜在影响，并使企业能够快速恢复运营。

C

CAF

请参阅 AWS 云采用框架。

金丝雀部署

缓慢而渐进地向最终用户发布版本。当您确信无误后，即可部署新版本，并完全替换当前版本。

CCoE

请参阅云卓越中心。

CDC

请参阅更改数据捕获。

更改数据捕获（CDC）

跟踪数据来源（如数据库表）的更改并记录有关更改的元数据的过程。您可以将 CDC 用于各种目的，例如审计或复制目标系统中的更改以保持同步。

混沌工程

故意引入故障或破坏性事件来测试系统的韧性。您可以使用 AWS Fault Injection Service (AWS FIS) 来执行实验，对您的 AWS 工作负载施加压力并评估其响应。

CI/CD

请参阅持续集成和持续交付。

分类

一种有助于生成预测的分类流程。分类问题的 ML 模型预测离散值。离散值始终彼此不同。例如，一个模型可能需要评估图像中是否有汽车。

客户端加密

在目标 AWS 服务 收到数据之前，对数据进行本地加密。

云卓越中心 (CCoE)

一个多学科团队，负责推动整个组织的云采用工作，包括开发云最佳实践、调动资源、制定迁移时间表、领导组织完成大规模转型。有关更多信息，请参阅 AWS 云 企业战略博客上的 CCoE 帖子。

云计算

通常用于远程数据存储和 IoT 设备管理的云技术。云计算通常连接到边缘计算技术。

云运营模型

在 IT 组织中，一种用于构建、完善和优化一个或多个云环境的运营模型。有关更多信息，请参阅构建您的云运营模型。

云采用阶段

组织迁移到 AWS 云中时通常会经历四个阶段：

- 项目 - 出于概念验证和学习目的，开展一些与云相关的项目
- 基础 — 进行基础投资以扩大云采用率（例如，创建着陆区、定义 CCo E、建立运营模型）
- 迁移 - 迁移单个应用程序
- 重塑 - 优化产品和服务，在云中创新

Stephen Orban在 AWS 云 企业战略博客的博客文章《云优先之旅和采用阶段》中定义了这些阶段。有关它们与 AWS 迁移策略的关系的信息，请参阅迁移准备指南。

CMDB

请参阅配置管理数据库。

代码存储库

通过版本控制过程存储和更新源代码和其他资产（如文档、示例和脚本）的位置。常见的云存储库包括 GitHub 或 Bitbucket Cloud。每个版本的代码都称为一个分支。在微服务结构中，每个存储库都专门用于一个功能。单个 CI/CD 管线可以使用多个存储库。

冷缓存

一种空的、填充不足或包含过时或不相关数据的缓冲区缓存。这会影响性能，因为数据库实例必须从主内存或磁盘读取，这比从缓冲区缓存读取要慢。

冷数据

很少访问的数据，且通常是历史数据。查询此类数据时，通常可以接受慢速查询。将这些数据转移到性能较低且成本更低的存储层或类别可以降低成本。

计算机视觉（CV）

一种 AI 领域，它使用机器学习来分析和提取数字图像和视频等视觉格式中的信息。例如，Amazon SageMaker AI 为 CV 提供了图像处理算法。

配置偏移

对于工作负载而言，一种偏离预期状态的配置更改。这可能会导致工作负载变得不合规，且通常是渐进的，不是故意的。

配置管理数据库（CMDB）

一种存储库，用于存储和管理有关数据库及其 IT 环境的信息，包括硬件和软件组件及其配置。您通常在迁移的产品组合发现和分析阶段使用来自 CMDB 的数据。

合规性包

一系列 AWS Config 规则和补救措施，您可以汇编这些规则和补救措施，以自定义您的合规性和安全性检查。您可以使用 YAML 模板将一致性包作为单个实体部署在 AWS 账户 和区域或整个组织中。有关更多信息，请参阅 AWS Config 文档中的一致性包。

持续集成和持续交付（CI/CD）

自动执行软件发布过程的源代码、构建、测试、暂存和生产阶段的过程。 CI/CD 通常被描述为管道。 CI/CD 可以帮助您实现流程自动化、提高生产力、提高代码质量和更快地交付。有关更多信息，请参阅持续交付的优势。CD 也可以表示持续部署。有关更多信息，请参阅持续交付与持续部署。

CV

请参阅计算机视觉。

D

静态数据

网络中静止的数据，例如存储中的数据。

数据分类

根据网络中数据的关键性和敏感性对其进行识别和分类的过程。它是任何网络安全风险管理策略的关键组成部分，因为它可以帮助您确定对数据的适当保护和保留控制。数据分类是 Well-Architecte AWS d Framework 中安全支柱的一个组成部分。有关详细信息，请参阅数据分类。

数据漂移

生产数据与用来训练机器学习模型的数据之间的有意义差异，或者输入数据随时间推移的有意义变化。数据漂移可能降低机器学习模型预测的整体质量、准确性和公平性。

传输中数据

在网络中主动移动的数据，例如在网络资源之间移动的数据。

数据网格

一种架构框架，可提供分布式、去中心化的数据所有权以及集中式管理和治理。

数据最少化

仅收集并处理绝对必要数据的原则。在中进行数据最小化 AWS 云 可以降低隐私风险、成本和分析碳足迹。

数据边界

AWS 环境中的一组预防性防护措施，可帮助确保只有可信身份才能访问来自预期网络的可信资源。有关更多信息，请参阅在上构建数据边界。 AWS

数据预处理

将原始数据转换为 ML 模型易于解析的格式。预处理数据可能意味着删除某些列或行，并处理缺失、不一致或重复的值。

数据溯源

在数据的整个生命周期跟踪其来源和历史的过程，例如数据如何生成、传输和存储。

数据主体

正在收集和处理其数据的个人。

数据仓库

一种支持商业智能（例如分析）的数据管理系统。数据仓库通常包含大量历史数据，通常用于查询和分析。

数据库定义语言（DDL）

在数据库中创建或修改表和对象结构的语句或命令。

数据库操作语言（DML）

在数据库中修改（插入、更新和删除）信息的语句或命令。

DDL

请参阅数据库定义语言。

深度融合

组合多个深度学习模型进行预测。您可以使用深度融合来获得更准确的预测或估算预测中的不确定性。

深度学习

一个 ML 子字段使用多层人工神经网络来识别输入数据和感兴趣的目标变量之间的映射。

defense-in-depth

一种信息安全方法，经过深思熟虑，在整个计算机网络中分层实施一系列安全机制和控制措施，以保护网络及其中数据的机密性、完整性和可用性。当你采用这种策略时 AWS，你会在 AWS

Organizations 结构的不同层面添加多个控件来帮助保护资源。例如，一种 defense-in-depth方法可以结合多因素身份验证、网络分段和加密。

委派管理员

在中 AWS Organizations，兼容的服务可以注册 AWS 成员帐户来管理组织的帐户并管理该服务的权限。此账户被称为该服务的委托管理员。有关更多信息和兼容服务列表，请参阅 AWS Organizations 文档中使用 AWS Organizations的服务。

部署

使应用程序、新功能或代码修复在目标环境中可用的过程。部署涉及在代码库中实现更改，然后在应用程序的环境中构建和运行该代码库。

开发环境

请参阅环境。

侦测性控制

一种安全控制，在事件发生后进行检测、记录日志和发出提醒。这些控制是第二道防线，提醒您注意绕过现有预防性控制的安全事件。有关更多信息，请参阅在 AWS上实施安全控制中的侦测性控制。

开发价值流映射（DVSM）

用于识别对软件开发生命周期中的速度和质量产生不利影响的限制因素并确定其优先级的流程。DVSM 扩展了最初为精益生产实践设计的价值流映射流程。其重点关注在软件开发过程中创造和转移价值所需的步骤和团队。

数字孪生

真实世界系统的虚拟再现，如建筑物、工厂、工业设备或生产线。数字孪生支持预测性维护、远程监控和生产优化。

维度表

星型架构中的一种较小的表，其中包含事实表中定量数据的数据属性。维度表属性通常是文本字段或行为类似于文本的离散数字。这些属性通常用于查询约束、筛选和结果集标注。

灾难

阻止工作负载或系统在其主要部署位置实现其业务目标的事件。这些事件可能是自然灾害、技术故障或人为操作的结果，例如无意的配置错误或恶意软件攻击。

灾难恢复（DR）

您用来最大程度地减少由灾难造成的停机时间和数据丢失的策略和流程。有关更多信息，请参阅 Well-Architected Fr ame AWS work 中的 “工作负载灾难恢复：云端 AWS 恢复”。

DML

请参阅数据库操作语言。

领域驱动设计

一种开发复杂软件系统的方法，通过将其组件连接到每个组件所服务的不断发展的领域或核心业务目标。Eric Evans 在其著作领域驱动设计：软件核心复杂性应对之道（Boston: Addison-Wesley Professional, 2003）中介绍了这一概念。有关如何将领域驱动设计与 strangler fig 模式结合使用的信息，请参阅使用容器和 Amazon API Gateway 逐步将原有的 Microsoft ASP.NET（ASMX）Web 服务现代化。

DR

请参阅灾难恢复。

偏差检测

跟踪与基准配置的偏差。例如，您可以使用 AWS CloudFormation 来检测系统资源中的偏差，也可以使用 AWS Control Tower 来检测着陆区中可能影响监管要求合规性的变化。

DVSM

请参阅开发价值流映射。

E

EDA

请参阅探索性数据分析。

EDI

请参阅电子数据交换。

边缘计算

该技术可提高位于 IoT 网络边缘的智能设备的计算能力。与云计算比较时，边缘计算可以减少通信延迟并缩短响应时间。

电子数据交换（EDI）

组织之间业务文件的自动交换。有关更多信息，请参阅什么是电子数据交换。

加密

一种将人类可读的纯文本数据转换为加密文字的计算流程。

加密密钥

由加密算法生成的随机位的加密字符串。密钥的长度可能有所不同，而且每个密钥都设计为不可预测且唯一。

字节顺序

字节在计算机内存中的存储顺序。大端序系统先存储最高有效字节。小端序系统先存储最低有效字节。

端点

请参阅服务端点。

端点服务

一种可以在虚拟私有云（VPC）中托管，与其他用户共享的服务。您可以使用其他 AWS 账户 或 AWS Identity and Access Management (IAM) 委托人创建终端节点服务， AWS PrivateLink 并向其授予权限。这些账户或主体可通过创建接口 VPC 端点来私密地连接到您的端点服务。有关更多信息，请参阅 Amazon Virtual Private Cloud（Amazon VPC）文档中的创建端点服务。

企业资源规划（ERP）

一种自动化和管理企业关键业务流程（例如会计、MES 和项目管理）的系统。

信封加密

用另一个加密密钥对加密密钥进行加密的过程。有关更多信息，请参阅 AWS Key Management Service (AWS KMS) 文档中的信封加密。

环境

正在运行的应用程序的实例。以下是云计算中常见的环境类型：

- 开发环境 — 正在运行的应用程序的实例，只有负责维护应用程序的核心团队才能使用。开发环境用于测试更改，然后再将其提升到上层环境。这类环境有时称为测试环境。
- 下层环境 — 应用程序的所有开发环境，比如用于初始构建和测试的环境。

- 生产环境 — 最终用户可以访问的正在运行的应用程序的实例。在 CI/CD 管道中，生产环境是最后一个部署环境。
- 上层环境 — 除核心开发团队以外的用户可以访问的所有环境。这可能包括生产环境、预生产环境和用户验收测试环境。

epic

在敏捷方法学中，有助于组织工作和确定优先级的功能类别。epics 提供了对需求和实施任务的总体描述。例如， AWS CAF 安全史诗包括身份和访问管理、侦探控制、基础设施安全、数据保护和事件响应。有关 AWS 迁移策略中 epics 的更多信息，请参阅计划实施指南。

ERP

请参阅企业资源规划。

探索性数据分析（EDA）

分析数据集以了解其主要特征的过程。您收集或汇总数据，并进行初步调查，以发现模式、检测异常并检查假定情况。EDA 通过计算汇总统计数据和创建数据可视化得以执行。

F

事实表

星型架构中的中心表。它存储有关业务运营的定量数据。通常，事实表包含两种类型的列：包含度量的列和包含维度表外键的列。

快速失效机制

一种使用频繁且增量式的测试来缩短开发生命周期的理念。这是敏捷方法的关键部分。

故障隔离边界

在中 AWS 云，诸如可用区 AWS 区域、控制平面或数据平面之类的边界，它限制了故障的影响并有助于提高工作负载的弹性。有关更多信息，请参阅 AWS 故障隔离边界。

功能分支

请参阅分支。

特征

您用来进行预测的输入数据。例如，在制造环境中，特征可能是定期从生产线捕获的图像。

特征重要性

特征对于模型预测的重要性。这通常表示为数值分数，可以通过各种技术进行计算，例如 Shapley 加法解释（SHAP）和积分梯度。有关更多信息，请参阅使用机器学习模型的可解释性 AWS。

功能转换

为 ML 流程优化数据，包括使用其他来源丰富数据、扩展值或从单个数据字段中提取多组信息。这使得 ML 模型能从数据中获益。例如，如果您将“2021-05-27 00:15:37”日期分解为“2021”、“五月”、“星期四”和“15”，则可以帮助学习与不同数据成分相关的算法学习精细模式。

少样本提示

在要求 LLM 执行类似任务之前，先向其提供少量示例，以演示任务和预期输出。此技术是上下文内学习的一种应用，其中模型可以从提示中嵌入的示例（样本）中学习。对于需要特定格式、推理或领域知识的任务，少样本提示可能非常有效。另请参阅零样本提示。

FGAC

请参阅精细访问控制。

精细访问控制（FGAC）

使用多个条件允许或拒绝访问请求。

快闪迁移

一种数据库迁移方法，通过更改数据捕获使用连续数据复制，在极短的时间内迁移数据，而非使用分阶段方法。目标是将停机时间降至最低。

FM

请参阅基础模型。

基础模型（FM）

一个大型深度学习神经网络，一直在广义和未标记数据的大量数据集上进行训练。 FMs 能够执行各种各样的一般任务，例如理解语言、生成文本和图像以及用自然语言进行对话。有关更多信息，请参阅什么是基础模型。

G

生成式人工智能

AI 模型的一个子集，这些模型已经过大量数据训练，可以使用简单的文本提示来创建新的内容和构件，例如图像、视频、文本和音频。有关更多信息，请参阅什么是生成式人工智能。

地理阻止

请参阅地理限制。

地理限制（地理阻止）

在 Amazon 中 CloudFront，一种阻止特定国家/地区的用户访问内容分发的选项。您可以使用允许列表或阻止列表来指定已批准和已禁止的国家/地区。有关更多信息，请参阅 CloudFront 文档中的限制内容的地理分布。

GitFlow 工作流程

一种方法，在这种方法中，下层和上层环境在源代码存储库中使用不同的分支。Gitflow 工作流程被认为是传统的工作流程，而基于中继的工作流程则是现代的、首选的方法。

黄金映像

系统或软件的快照，用作部署该系统或软件的新实例的模板。例如，在制造业中，黄金映像可用于在多个设备上预调配软件，并有助于提高设备制造操作的速度、可扩展性和生产效率。

全新策略

在新环境中缺少现有基础设施。在对系统架构采用全新策略时，您可以选择所有新技术，而不受对现有基础设施（也称为棕地）兼容性的限制。如果您正在扩展现有基础设施，则可以将棕地策略和全新策略混合。

防护机制

帮助管理各组织单位的资源、策略和合规性的高级规则 (OUs)。预防性防护机制会执行策略以确保符合合规性标准。它们是使用服务控制策略和 IAM 权限边界实现的。侦测性护栏会检测策略违规和合规性问题，并生成提醒以进行修复。它们通过使用 AWS Config、、Amazon、 AWS Security Hub CSPM GuardDuty AWS Trusted Advisor、Amazon Inspector 和自定义 AWS Lambda 支票来实现。

H

HA

请参阅高可用性。

异构数据库迁移

将源数据库迁移到使用不同数据库引擎的目标数据库（例如，从 Oracle 迁移到 Amazon Aurora）。异构迁移通常是重新架构工作的一部分，而转换架构可能是一项复杂的任务。AWS 提供了 AWS SCT 来帮助实现架构转换。

高可用性（HA）

在遇到挑战或灾难时，工作负载无需干预即可连续运行的能力。HA 系统旨在自动进行故障转移、持续提供良好性能，并以最小的性能影响处理不同负载和故障。

历史数据库现代化

一种用于实现运营技术（OT）系统现代化和升级以更好满足制造业需求的方法。历史数据库是一种用于收集和存储工厂中各种来源数据的数据库。

保留数据

从用于训练机器学习模型的数据集中保留的一部分标注的历史数据。通过将模型预测与保留数据进行比较，您可以使用保留数据来评估模型性能。

同构数据库迁移

将源数据库迁移到共享同一数据库引擎的目标数据库（例如，从 Microsoft SQL Server 迁移到 Amazon RDS for SQL Server）。同构迁移通常是更换主机或更换平台工作的一部分。您可以使用本机数据库实用程序来迁移架构。

热数据

经常访问的数据，例如实时数据或近期的转化数据。这些数据通常需要高性能存储层或存储类别才能提供快速的查询响应。

修补程序

针对生产环境中关键问题的紧急修复。由于其紧迫性，修补程序通常是在典型的 DevOps 发布工作流程之外进行的。

hypercare 周期

割接之后，迁移团队立即管理和监控云中迁移的应用程序以解决任何问题的时间段。通常，这个周期持续 1-4 天。在 hypercare 周期结束时，迁移团队通常会将应用程序的责任移交给云运营团队。

我

IaC

请参阅基础设施即代码。

基于身份的策略

附加到一个或多个 IAM 委托人的策略，用于定义他们在 AWS 云 环境中的权限。

空闲应用程序

90 天内平均 CPU 和内存使用率在 5% 到 20% 之间的应用程序。在迁移项目中，通常会停用这些应用程序或将其保留在本地。

IIoT

请参阅工业物联网。

不可变基础设施

一种模型，可为生产工作负载部署新的基础设施，而不是更新、修补或修改现有基础设施。不可变基础设施本质上比可变基础设施更一致、更可靠、更可预测。有关更多信息，请参阅 AWS Well-Architected Framework 中的使用不可变基础设施进行部署最佳实践。

入站（入口）VPC

在 AWS 多账户架构中，一种接受、检查和路由来自应用程序外部的网络连接的 VPC。AWS 安全参考架构建议设置您的网络帐户，包括入站、出站和检查， VPCs 以保护您的应用程序与更广泛的互联网之间的双向接口。

增量迁移

一种割接策略，在这种策略中，您可以将应用程序分成小部分进行迁移，而不是一次性完整割接。例如，您最初可能只将几个微服务或用户迁移到新系统。在确认一切正常后，您可以逐步迁移其他微服务或用户，直到停用遗留系统。这种策略降低了大规模迁移带来的风险。

工业 4.0

该术语由 Klaus Schwab 在 2016 年提出，指的是通过连接、实时数据、自动化、分析和 AI/ML 的进步来实现制造流程的现代化。

基础设施

应用程序环境中包含的所有资源和资产。

基础设施即代码（IaC）

通过一组配置文件预调配和管理应用程序基础设施的过程。IaC 旨在帮助您集中管理基础设施、实现资源标准化和快速扩展，使新环境具有可重复性、可靠性和一致性。

工业物联网 (IIoT)

在工业领域使用联网的传感器和设备，例如制造业、能源、汽车、医疗保健、生命科学和农业。有关更多信息，请参阅制定工业物联网 (IIoT) 数字化转型战略。

检查 VPC

在 AWS 多账户架构中，一种集中式 VPC，用于管理对 VPCs（相同或不同 AWS 区域）、互联网和本地网络之间的网络流量的检查。AWS 安全参考架构建议设置您的网络帐户，包括入站、出站和检查， VPCs 以保护您的应用程序与更广泛的互联网之间的双向接口。

物联网（IoT）

由带有嵌入式传感器或处理器的连接物理对象组成的网络，这些传感器或处理器通过互联网或本地通信网络与其他设备和系统进行通信。有关更多信息，请参阅什么是 IoT？

可解释性

它是机器学习模型的一种特征，描述了人类可以理解模型的预测如何取决于其输入的程度。有关更多信息，请参阅使用机器学习模型的可解释性 AWS。

物联网

请参阅物联网。

IT 信息库（ITIL）

提供 IT 服务并使这些服务符合业务要求的一套最佳实践。ITIL 是 ITSM 的基础。

IT 服务管理（ITSM）

为组织设计、实施、管理和支持 IT 服务的相关活动。有关将云运营与 ITSM 工具集成的信息，请参阅运营集成指南。

ITIL

请参阅 IT 信息库。

ITSM

请参阅 IT 服务管理。

L

基于标签的访问控制（LBAC）

强制访问控制（MAC）的一种实施方式，其中明确为用户和数据本身分配了安全标签值。用户安全标签和数据安全标签之间的交集决定了用户可以看到哪些行和列。

登录区

landing zone 是一个架构精良的多账户 AWS 环境，具有可扩展性和安全性。这是一个起点，您的组织可以从这里放心地在安全和基础设施环境中快速启动和部署工作负载和应用程序。有关登录区的更多信息，请参阅设置安全且可扩展的多账户 AWS 环境。

大语言模型（LLM）

一种基于大量数据进行预训练的深度学习 AI 模型。LLM 可以执行多项任务，例如回答问题、总结文档、将文本翻译成其他语言以及完成句子。有关更多信息，请参阅什么是 LLMs。

大规模迁移

迁移 300 台或更多服务器。

LBAC

请参阅基于标签的访问控制。

最低权限

授予执行任务所需的最低权限的最佳安全实践。有关更多信息，请参阅 IAM 文档中的应用最低权限许可。

直接迁移

请参阅 7 R。

小端序系统

一个先存储最低有效字节的系统。另请参阅字节顺序。

LLM

请参阅大型语言模型。

下层环境

请参阅环境。

M

机器学习（ML）

一种使用算法和技术进行模式识别和学习的人工智能。ML 对记录的数据（例如物联网（IoT）数据）进行分析和学习，以生成基于模式的统计模型。有关更多信息，请参阅机器学习。

主分支

请参阅分支。

恶意软件

旨在危害计算机安全或隐私的软件。恶意软件可能会破坏计算机系统、泄露敏感信息或获得未经授权的访问权限。恶意软件的示例包括病毒、蠕虫、勒索软件、木马、间谍软件和键盘记录器。

托管式服务

AWS 服务 它 AWS 运行基础设施层、操作系统和平台，您可以访问端点来存储和检索数据。Amazon Simple Storage Service（Amazon S3）和 Amazon DynamoDB 就是托管服务的示例。这些服务也称为抽象服务。

制造执行系统（MES）

一种软件系统，用于跟踪、监控、记录和控制在车间将原材料转化为成品的生产过程。

MAP

请参阅迁移加速计划。

机制

一个完整的过程，您可以在其中创建工具，推动工具的采用，然后检查结果以进行调整。机制是一种在运作过程中自我强化和改善的循环。有关更多信息，请参阅在 Well-Architect AWS ed 框架中构建机制。

成员账户

AWS 账户 除属于组织中的管理账户之外的所有账户 AWS Organizations。一个账户一次只能是一个组织的成员。

MES

请参阅制造执行系统。

消息队列遥测传输（MQTT）

一种基于发布/订阅模式的轻量级 machine-to-machine (M2M) 通信协议，适用于资源受限的物联网设备。

微服务

一种小型的独立服务，通过明确的定义进行通信 APIs ，通常由小型的独立团队拥有。例如，保险系统可能包括映射到业务能力（如销售或营销）或子域（如购买、理赔或分析）的微服务。微服务

的好处包括敏捷、灵活扩展、易于部署、可重复使用的代码和恢复能力。有关更多信息，请参阅使用 AWS 无服务器服务集成微服务。

微服务架构

一种使用独立组件构建应用程序的方法，这些组件将每个应用程序进程作为微服务运行。这些微服务使用轻量级通过定义明确的接口进行通信。 APIs该架构中的每个微服务都可以更新、部署和扩展，以满足对应用程序特定功能的需求。有关更多信息，请参阅在上实现微服务。 AWS

迁移加速计划（MAP）

AWS 该计划提供咨询支持、培训和服务，以帮助组织为迁移到云奠定坚实的运营基础，并帮助抵消迁移的初始成本。MAP 提供了一种以系统的方式执行遗留迁移的迁移方法，以及一套用于自动执行和加速常见迁移场景的工具。

大规模迁移

将大部分应用程序组合分波迁移到云中的过程，在每一波中以更快的速度迁移更多应用程序。本阶段使用从早期阶段获得的最佳实践和经验教训，实施由团队、工具和流程组成的迁移工厂，通过自动化和敏捷交付简化工作负载的迁移。这是 AWS 迁移策略的第三阶段。

迁移工厂

跨职能团队，通过自动化、敏捷的方法简化工作负载迁移。迁移工厂团队通常包括运营、业务分析师和所有者、迁移工程师、开发人员和冲刺 DevOps 领域的专业人员。20% 到 50% 的企业应用程序组合由可通过工厂方法优化的重复模式组成。有关更多信息，请参阅本内容集中有关迁移工厂的讨论和云迁移工厂指南。

迁移元数据

有关完成迁移所需的应用程序和服务器的信息。每种迁移模式都需要一套不同的迁移元数据。迁移元数据的示例包括目标子网、安全组和 AWS 账户。

迁移模式

一种可重复的迁移任务，详细列出了迁移策略、迁移目标以及所使用的迁移应用程序或服务。示例：使用 AWS 应用程序迁移服务重新托管向 Amazon EC2 的迁移。

迁移组合评测（MPA）

一种在线工具，提供了用于验证迁移到 AWS 云的业务案例的信息。MPA 提供了详细的组合评测（服务器规模调整、定价、TCO 比较、迁移成本分析）以及迁移计划（应用程序数据分析和数据收集、应用程序分组、迁移优先级排序和波次规划）。所有 AWS 顾问和 APN 合作伙伴顾问均可免费使用 MPA 工具（需要登录）。

迁移准备情况评测（MRA）

使用 AWS CAF 深入了解组织的云就绪状态、确定优势和劣势以及制定行动计划以缩小已发现差距的过程。有关更多信息，请参阅迁移准备指南。MRA 是 AWS 迁移策略的第一阶段。

迁移策略

将工作负载迁移到 AWS 云的方法。有关更多信息，请参见术语表中的 7 R 词条，以及动员您的组织以加快大规模迁移。

ML

请参阅机器学习。

现代化

将过时的（原有的或单体）应用程序及其基础设施转变为云中敏捷、弹性和高度可用的系统，以降低成本、提高效率和利用创新。有关更多信息，请参阅在 AWS 云中实现应用程序现代化的策略。

现代化准备情况评估

一种评估方式，有助于确定组织应用程序的现代化准备情况；确定收益、风险和依赖关系；确定组织能够在多大程度上支持这些应用程序的未来状态。评估结果是目标架构的蓝图、详细说明现代化进程发展阶段和里程碑的路线图以及解决已发现差距的行动计划。有关更多信息，请参阅在 AWS 云中评估应用程序的现代化准备情况。

单体应用程序（单体式）

作为具有紧密耦合进程的单个服务运行的应用程序。单体应用程序有几个缺点。如果某个应用程序功能的需求激增，则必须扩展整个架构。随着代码库的增长，添加或改进单体应用程序的功能也会变得更加复杂。若要解决这些问题，可以使用微服务架构。有关更多信息，请参阅将单体分解为微服务。

MPA

请参阅迁移组合评测。

MQTT

请参阅消息队列遥测传输。

多分类器

一种帮助为多个类别生成预测（预测两个以上结果之一）的过程。例如，ML 模型可能会询问“这个产品是书、汽车还是手机？”或“此客户最感兴趣什么类别的产品？”

可变基础设施

一种用于更新和修改生产工作负载的现有基础设施的模型。为了提高一致性、可靠性和可预测性，Well-Architect AWS ed Framework 建议使用不可变基础设施作为最佳实践。

O

OAC

请参阅来源访问控制。

OAI

请参阅来源访问身份。

OCM

请参阅组织变革管理。

离线迁移

一种迁移方法，在这种方法中，源工作负载会在迁移过程中停止运行。这种方法会延长停机时间，通常用于小型非关键工作负载。

OI

请参阅运营集成。

OLA

请参阅运营级别协议。

在线迁移

一种迁移方法，在这种方法中，源工作负载无需离线即可复制到目标系统。在迁移过程中，连接工作负载的应用程序可以继续运行。这种方法的停机时间为零或最短，通常用于关键生产工作负载。

OPC-UA

请参阅开放流程通信 – 统一架构。

开放流程通信 – 统一架构（OPC-UA）

一种用于工业自动化的 machine-to-machine（M2M）通信协议。OPC-UA 提供了一个包含数据加密、身份验证和授权方案的互操作性标准。

运营级别协议（OLA）

一项协议，阐明了 IT 职能部门承诺相互交付的内容，以支持服务水平协议（SLA）。

运营准备情况审查（ORR）

一份问题核对清单和关联的最佳实践，可帮助您了解、评估、预防或缩小事件和可能的故障的范围。有关更多信息，请参阅 AWS Well-Architected Framework 中的运营准备情况审查（ORR）。

运营技术（OT）

与物理环境配合使用以控制工业运营、设备和基础设施的硬件和软件系统。在制造业中，OT 和信息技术（IT）系统的集成是工业 4.0 转型的关键重点。

运营整合（OI）

在云中实现运营现代化的过程，包括就绪计划、自动化和集成。有关更多信息，请参阅运营整合指南。

组织跟踪

由 AWS CloudTrail 此创建的跟踪记录组织 AWS 账户 中所有人的所有事件 AWS Organizations。该跟踪是在每个 AWS 账户 中创建的，属于组织的一部分，并跟踪每个账户的活动。有关更多信息，请参阅 CloudTrail文档中的为组织创建跟踪。

组织变革管理（OCM）

一个从人员、文化和领导力角度管理重大、颠覆性业务转型的框架。OCM 通过加快变革采用、解决过渡问题以及推动文化和组织变革，帮助组织为新系统和战略做好准备和过渡。在 AWS 迁移策略中，该框架被称为人员加速，因为云采用项目需要变更的速度。有关更多信息，请参阅 OCM 指南。

来源访问控制（OAC）

在中 CloudFront，一个增强的选项，用于限制访问以保护您的亚马逊简单存储服务 (Amazon S3) 内容。OAC 全部支持所有 S3 存储桶 AWS 区域、使用 AWS KMS (SSE-KMS) 进行服务器端加密，以及对 S3 存储桶的动态PUT和DELETE请求。

来源访问身份（OAI）

在中 CloudFront，一个用于限制访问权限以保护您的 Amazon S3 内容的选项。当您使用 OAI 时，CloudFront 会创建一个 Amazon S3 可以对其进行身份验证的委托人。经过身份验证的委托人只能通过特定 CloudFront 分配访问 S3 存储桶中的内容。另请参阅 OAC，其中提供了更精细和增强的访问控制。

ORR

请参阅运营准备情况审查。

OT

请参阅运营技术。

出站（出口）VPC

在 AWS 多账户架构中，一种处理从应用程序内部启动的网络连接的 VPC。AWS 安全参考架构建议设置您的网络帐户，包括入站、出站和检查， VPCs 以保护您的应用程序与更广泛的互联网之间的双向接口。

P

权限边界

附加到 IAM 主体的 IAM 管理策略，用于设置用户或角色可以拥有的最大权限。有关更多信息，请参阅 IAM 文档中的权限边界。

个人身份信息（PII）

直接查看其他相关数据或与之配对时可用于合理推断个人身份的信息。PII 的示例包括姓名、地址和联系信息。

PII

请参阅个人身份信息。

playbook

一套预定义的步骤，用于捕获与迁移相关的工作，例如在云中交付核心运营功能。playbook 可以采用脚本、自动化运行手册的形式，也可以是操作现代化环境所需的流程或步骤的摘要。

PLC

请参阅可编程逻辑控制器。

PLM

请参阅产品生命周期管理。

policy

一个对象，可以定义权限（请参阅基于身份的策略）、指定访问条件（请参阅基于资源的策略）或定义 AWS Organizations 的组织中所有账户的最大权限（请参阅服务控制策略）。

多语言持久性

根据数据访问模式和其他要求，独立选择微服务的数据存储技术。如果您的微服务采用相同的数据存储技术，它们可能会遇到实现难题或性能不佳。如果微服务使用最适合其需求的数据存储，则可以更轻松地实现微服务，并获得更好的性能和可扩展性。

组合评测

一个发现、分析和确定应用程序组合优先级以规划迁移的过程。有关更多信息，请参阅评估迁移准备情况。

谓词

返回 `true` 或 `false` 的查询条件，通常位于 `WHERE` 子句中。

谓词下推

一种数据库查询优化技术，可在传输之前筛选查询中的数据。这将减少从关系数据库检索和处理的数据量，并提高查询性能。

预防性控制

一种安全控制，旨在防止事件发生。这些控制是第一道防线，帮助防止未经授权的访问或对网络的意外更改。有关更多信息，请参阅在 AWS上实施安全控制中的预防性控制。

主体

中 AWS 可以执行操作和访问资源的实体。此实体通常是 IAM 角色的根用户或用户。 AWS 账户有关更多信息，请参阅 IAM 文档中角色术语和概念中的主体。

隐私设计

一种在整个开发过程中都考虑隐私的系统工程方法。

私有托管区

一个容器，其中包含有关您希望 Amazon Route 53 如何响应针对一个或多个 VPCs域名及其子域名的 DNS 查询的信息。有关更多信息，请参阅 Route 53 文档中的私有托管区的使用。

主动控制

一种安全控制，旨在防止部署不合规资源。这些控制会在资源预置之前对其进行扫描。如果资源与控制不兼容，则不会预置它。有关更多信息，请参阅 AWS Control Tower 文档中的控制参考指南，并参见在上实施安全控制中的主动控制 AWS。

产品生命周期管理（PLM）

对产品在其整个生命周期内的数据和流程的管理，从设计、开发和发布，到增长和成熟，再到衰退和淘汰。

生产环境

请参阅环境。

可编程逻辑控制器（PLC）

在制造业中，一种高度可靠、适应性强的计算机，用于监控机器并实现制造过程自动化。

提示串接

使用一个 LLM 提示的输出作为下一个提示的输入，以生成更好的响应。该技术用于将复杂的任务分解为子任务，或者迭代地完善或扩展初步响应。它有助于提高模型响应的准确性和相关性，并允许获得更精细的个性化结果。

假名化

用占位符值替换数据集中个人标识符的过程。假名化可以帮助保护个人隐私。假名化数据仍被视为个人数据。

publish/subscribe (pub/sub)

一种支持微服务间异步通信的模式，可提高可扩展性和响应能力。例如，在基于微服务的 MES 中，微服务可以将事件消息发布到其他微服务可以订阅的频道。系统可以在不更改发布服务的情况下添加新的微服务。

Q

查询计划

一系列用于访问 SQL 关系数据库系统中的数据的步骤，类似于指令。

查询计划回归

当数据库服务优化程序选择的最佳计划不如数据库环境发生特定变化之前时。这可能是由统计数据、约束、环境设置、查询参数绑定更改和数据库引擎更新造成的。

R

RACI 矩阵

请参阅责任、问责、咨询和知情（RACI）。

RAG

请参阅检索增强生成。

勒索软件

一种恶意软件，旨在阻止对计算机系统或数据的访问，直到付款为止。

RASCI 矩阵

请参阅责任、问责、咨询和知情（RACI）。

RCAC

请参阅行列访问控制。

只读副本

用于只读目的的数据库副本。您可以将查询路由到只读副本，以减轻主数据库的负载。

重新架构

请参阅 7 R。

恢复点目标（RPO）

自上一个数据恢复点以来可接受的最长时间。这决定了从上一个恢复点到服务中断之间可接受的数据丢失情况。

恢复时间目标（RTO）

服务中断和服务恢复之间可接受的最大延迟。

重构

请参阅 7 R。

Region

地理区域内的 AWS 资源集合。每一个 AWS 区域 都相互隔离，彼此独立，以提供容错、稳定性和弹性。有关更多信息，请参阅指定您的账户可以使用的 AWS 区域。

回归

一种预测数值的 ML 技术。例如，要解决“这套房子的售价是多少？”的问题 ML 模型可以使用线性回归模型，根据房屋的已知事实（如建筑面积）来预测房屋的销售价格。

重新托管

请参阅 7 R。

版本

在部署过程中，推动生产环境变更的行为。

重新放置

请参阅 7 R。

更换平台

请参阅 7 R。

重新购买

请参阅 7 R。

韧性

应用程序抵御中断或从中断中恢复的能力。在 AWS 云中规划韧性时，高可用性和灾难恢复是常见的考虑因素。有关更多信息，请参阅 AWS 云 韧性。

基于资源的策略

一种附加到资源的策略，例如 AmazonS3 存储桶、端点或加密密钥。此类策略指定了允许哪些主体访问、支持的操作以及必须满足的任何其他条件。

责任、问责、咨询和知情（RACI）矩阵

定义参与迁移活动和云运营的所有各方的角色和责任的矩阵。矩阵名称源自矩阵中定义的责任类型：负责（R）、问责（A）、咨询（C）和知情（I）。支持（S）类型是可选的。如果包括支持，则该矩阵称为 RASCI 矩阵，如果将其排除在外，则称为 RACI 矩阵。

响应性控制

一种安全控制，旨在推动对不良事件或偏离安全基线的情况进行修复。有关更多信息，请参阅在AWS上实施安全控制中的响应性控制。

保留

请参阅 7 R。

停用

请参阅 7 R。

检索增强生成（RAG）

一种生成式人工智能技术，其中 LLM 在生成响应之前引用其训练数据来源之外的权威数据来源。例如，RAG 模型可以对组织的知识库或自定义数据执行语义搜索。有关更多信息，请参阅什么是 RAG。

轮换

定期更新密钥以使攻击者更难访问凭证的过程。

行列访问控制（RCAC）

使用已定义访问规则的基本、灵活的 SQL 表达式。RCAC 由行权限和列掩码组成。

RPO

请参阅恢复点目标。

RTO

请参阅恢复时间目标。

运行手册

执行特定任务所需的一套手动或自动程序。它们通常是为了简化重复性操作或高错误率的程序而设计的。

S

SAML 2.0

许多身份提供商 (IdPs) 使用的开放标准。此功能支持联合单点登录 (SSO)，因此用户无需在 IAM 中为组织中的所有人创建用户即可登录 AWS 管理控制台 或调用 AWS API 操作。有关基于 SAML 2.0 的联合身份验证的更多信息，请参阅 IAM 文档中的关于基于 SAML 2.0 的联合身份验证。

SCADA

请参阅监督控制和数据采集。

SCP

请参阅服务控制策略。

机密密钥

在中 AWS Secrets Manager，您以加密形式存储的机密或受限信息，例如密码或用户凭证。它由密钥值及其元数据组成。密钥值可以是二进制、单个字符串或多个字符串。有关更多信息，请参阅 Secrets Manager 文档中的什么是 Amazon Secrets Manager 密钥？。

安全设计

一种在整个开发过程中都考虑安全的系统工程方法。

安全控制

一种技术或管理防护机制，可防止、检测或降低威胁行为体利用安全漏洞的能力。安全控制有以下四种类型：预防性、检测性、响应性和主动性。

安全固化

缩小攻击面，使其更能抵御攻击的过程。这可能包括删除不再需要的资源、实施授予最低权限的最佳安全实践或停用配置文件中不必要的功能等操作。

安全信息和事件管理（SIEM）系统

结合了安全信息管理（SIM）和安全事件管理（SEM）系统的工具和服务。SIEM 系统会收集、监控和分析来自服务器、网络、设备和其他来源的数据，以检测威胁和安全漏洞，并生成警报。

安全响应自动化

一种预定义的程序化操作，旨在自动响应或修复安全事件。这些自动化可作为侦探或响应式安全控制措施，帮助您实施 AWS 安全最佳实践。自动响应操作的示例包括修改 VPC 安全组、修补 Amazon EC2 实例或轮换凭证。

服务器端加密

由接收数据的人在目的地对数据 AWS 服务 进行加密。

服务控制策略（SCP）

一种策略，用于集中控制组织中所有账户的权限 AWS Organizations。 SCPs 定义防护措施或限制管理员可以委托给用户或角色的操作。您可以使用 SCPs 允许列表或拒绝列表来指定允许或禁止哪些服务或操作。有关更多信息，请参阅 AWS Organizations 文档中的服务控制策略。

服务端点

的入口点的 URL AWS 服务。您可以使用端点，通过编程方式连接到目标服务。有关更多信息，请参阅 AWS 一般参考 中的 AWS 服务 端点。

服务水平协议（SLA）

一份协议，阐明了 IT 团队承诺向客户交付的内容，比如服务正常运行时间和性能。

服务水平指示器（SLI）

对服务性能方面的衡量，例如错误率、可用性或吞吐量。

服务水平目标（SLO）

代表服务运行状况的目标指标，由服务水平指示器衡量。

责任共担模式

描述您在云安全与合规方面共同承担 AWS 的责任的模型。 AWS 负责云的安全，而您则负责云中的安全。有关更多信息，请参阅责任共担模式。

SIEM

请参阅安全信息和事件管理系统。

单点故障（SPOF）

应用程序的单个关键组件出现故障，可能会中断系统。

SLA

请参阅服务水平协议。

SLI

请参阅服务水平指示器。

SLO

请参阅服务水平目标。

split-and-seed 模型

一种扩展和加速现代化项目的模式。随着新功能和产品发布的定义，核心团队会拆分以创建新的产品团队。这有助于扩展组织的能力和服务，提高开发人员的工作效率，支持快速创新。有关更多信息，请参阅在 AWS 云中实现应用程序现代化的分阶段方法。

SPOF

请参阅单点故障。

星型架构

一种数据库组织结构，它使用一个大型事实表来存储事务数据或测量数据，并使用一个或多个较小的维度表来存储数据属性。此结构专为在数据仓库中使用或用于商业智能目的而设计。

strangler fig 模式

一种通过逐步重写和替换系统功能直至可以停用原有的系统来实现单体系统现代化的方法。这种模式用无花果藤作为类比，这种藤蔓成长为一棵树，最终战胜并取代了宿主。该模式是由 Martin Fowler 提出的，作为重写单体系统时管理风险的一种方法。有关如何应用此模式的示例，请参阅使用容器和 Amazon API Gateway 逐步将原有的 Microsoft ASP.NET（ASMX）Web 服务现代化。

子网

您的 VPC 内的一个 IP 地址范围。子网必须位于单个可用区中。

监督控制和数据采集（SCADA）

在制造业中，一种使用硬件和软件来监控实物资产和生产操作的系统。

对称加密

一种加密算法，它使用相同的密钥来加密和解密数据。

综合测试

以模拟用户交互的方式测试系统，以检测潜在问题或监控性能。你可以使用 Amazon S CloudWatch ynthetics 来创建这些测试。

系统提示

一种为 LLM 提供上下文、说明或准则以指导其行为的技术。系统提示有助于设置上下文并制定与用户交互的规则。

T

标签

键值对，用作组织资源的元数据。 AWS 标签有助于您管理、识别、组织、搜索和筛选 资源。有关更多信息，请参阅标记您的 AWS 资源。

目标变量

您在监督式 ML 中尝试预测的值。这也被称为结果变量。例如，在制造环境中，目标变量可能是产品缺陷。

任务列表

一种通过运行手册用于跟踪进度的工具。任务列表包含运行手册的概述和要完成的常规任务列表。对于每项常规任务，它包括预计所需时间、所有者和进度。

测试环境

请参阅环境。

训练

为您的 ML 模型提供学习数据。训练数据必须包含正确答案。学习算法在训练数据中查找将输入数据属性映射到目标（您希望预测的答案）的模式。然后输出捕获这些模式的 ML 模型。然后，您可以使用 ML 模型对不知道目标的新数据进行预测。

中转网关

一个网络传输中心，可用于将您的网络 VPCs 和本地网络互连。有关更多信息，请参阅 AWS Transit Gateway 文档中的什么是公交网关。

基于中继的工作流程

一种方法，开发人员在功能分支中本地构建和测试功能，然后将这些更改合并到主分支中。然后，按顺序将主分支构建到开发、预生产和生产环境。

可信访问权限

向您指定的服务授予权限，该服务可代表您在其账户中执行任务。 AWS Organizations 当需要服务相关的角色时，受信任的服务会在每个账户中创建一个角色，为您执行管理任务。有关更多信息，请参阅 AWS Organizations 文档中的AWS Organizations 与其他 AWS 服务一起使用。

优化

更改训练过程的各个方面，以提高 ML 模型的准确性。例如，您可以通过生成标签集、添加标签，并在不同的设置下多次重复这些步骤来优化模型，从而训练 ML 模型。

双披萨团队

一个小 DevOps 团队，你可以用两个披萨来喂食。双披萨团队的规模可确保在软件开发过程中充分协作。

U

不确定性

这一概念指的是不精确、不完整或未知的信息，这些信息可能会破坏预测式 ML 模型的可靠性。不确定性有两种类型：认知不确定性是由有限的、不完整的数据造成的，而偶然不确定性是由数据中固有的噪声和随机性导致的。

无差别任务

也称为繁重工作，即创建和运行应用程序所必需的工作，但不能为最终用户提供直接价值或竞争优势。无差别任务的示例包括采购、维护和容量规划。

上层环境

请参阅环境。

V

vacuum 操作

一种数据库维护操作，包括在增量更新后进行清理，以回收存储空间并提高性能。

版本控制

跟踪更改的过程和工具，例如存储库中源代码的更改。

VPC 对等连接

两者之间的连接 VPCs ，允许您使用私有 IP 地址路由流量。有关更多信息，请参阅 Amazon VPC 文档中的什么是 VPC 对等连接。

漏洞

损害系统安全的软件缺陷或硬件缺陷。

W

热缓存

一种包含经常访问的当前相关数据的缓冲区缓存。数据库实例可以从缓冲区缓存读取，这比从主内存或磁盘读取要快。

暖数据

不常访问的数据。查询此类数据时，通常可以接受中速查询。

窗口函数

一种对与当前记录有某种关联的一组行执行计算的 SQL 函数。窗口函数对于处理任务很有用，例如计算移动平均值或根据当前行的相对位置访问行的值。

工作负载

一系列资源和代码，它们可以提供商业价值，如面向客户的应用程序或后端过程。

工作流

迁移项目中负责一组特定任务的职能小组。每个工作流都是独立的，但支持项目中的其他工作流。例如，组合工作流负责确定应用程序的优先级、波次规划和收集迁移元数据。组合工作流将这些资产交付给迁移工作流，然后迁移服务器和应用程序。

WORM

请参阅一次写入多次读取。

WQF

请参阅 AWS 工作负载资格鉴定框架。

一次写入多次读取（WORM）

一种存储模型，可一次写入数据并防止数据被删除或修改。授权用户可以根据需要多次读取数据，但无法对其进行更改。此数据存储基础设施被认为不可变。

Z

零日漏洞利用

一种利用零日漏洞的攻击，通常为恶意软件。

零日漏洞

生产系统中不可避免的缺陷或漏洞。威胁主体可能利用这种类型的漏洞攻击系统。开发人员经常因攻击而意识到该漏洞。

零样本提示

为 LLM 提供执行任务的说明，但没有可以帮助指导的示例（样本）。LLM 必须使用预先训练的知识来处理任务。零样本提示的有效性取决于任务的复杂性和提示的质量。另请参阅少样本提示。

僵尸应用程序

平均 CPU 和内存使用率低于 5% 的应用程序。在迁移项目中，通常会停用这些应用程序。

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。