



AWS 云采用框架：平台视角

# AWS 规范性指导



# AWS 规范性指导: AWS 云采用框架：平台视角

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

# Table of Contents

欢迎 .....	1
简介 .....	2
平台架构 .....	5
启动 .....	5
定义多账户策略 .....	5
定义预防性控制 .....	5
定义组织单元结构 .....	5
定义网络连接 .....	5
定义 DNS 策略 .....	6
定义标记标准 .....	6
定义可观测性策略 .....	7
推进 .....	7
定义主动控制和检测控制 .....	7
定义服务入门标准 .....	7
定义模式和原则 .....	7
Excel .....	7
定义修复模式 .....	7
沟通和完善策略 .....	8
了解财务管理能力 .....	8
平台工程 .....	9
晚于 .....	9
构建登录区并部署护栏 .....	9
建立身份验证 .....	9
部署网络 .....	10
收集、汇总和保护事件和日志数据 .....	10
建立控制 .....	10
实施云财务管理 .....	10
推进 .....	10
构建基础设施自动化 .....	10
提供集中式可观测性服务 .....	11
实施系统管理和 AMI 治理 .....	11
管理凭证使用 .....	11
建立安全工具 .....	11
Excel .....	12

通过自动化获取和分发身份结构 .....	12
为跨环境的异常模式添加检测和提醒 .....	12
对威胁进行分析和建模 .....	12
持续收集、审查和完善权限 .....	12
选择、衡量并持续改进您的平台指标 .....	12
数据架构 .....	13
启动 .....	13
定义总体功能 .....	13
整理数据区域 .....	13
规划数据的敏捷性和民主化 .....	14
定义安全数据交付 .....	14
制定成本效益计划 .....	14
推进 .....	14
了解特征工程 .....	14
计划对数据集进行非规范化 .....	14
设计可移植性和可扩展性 .....	15
Excel .....	15
设计可配置的框架 .....	15
计划构建统一的分析引擎 .....	15
定义 DataOps .....	15
数据工程 .....	16
启动 .....	16
部署数据湖 .....	16
开发数据摄取模式 .....	16
加速数据处理 .....	17
提供数据可视化服务 .....	17
推进 .....	18
实现近乎实时的数据处理 .....	18
验证数据质量 .....	18
证明数据转换服务 .....	18
启用数据民主化 .....	19
Excel .....	19
提供基于用户界面的编排 .....	19
集成 DataOps .....	19
预调配和编排 .....	21
晚于 .....	21

部署 hub-and-spoke 目录模型 .....	21
整理模板以供重复使用 .....	21
应用默认参数以便重复使用 .....	21
建立审批流程 .....	22
推进 .....	22
创建自助服务门户 .....	22
启用私有市场 .....	22
管理权限 .....	22
Excel .....	22
与采购系统集成 .....	22
与您的 ITSM 工具集成 .....	23
实施生命周期管理和版本分发系统 .....	23
现代应用程序开发 .....	24
启动 .....	24
探索现代方法 .....	24
采用云原生计算功能 .....	24
使用容器化 .....	25
使用现代数据库 .....	25
推进 .....	25
优化您的现代架构 .....	25
使用服务网格技术 .....	26
确保可见性和可追溯性 .....	26
Excel .....	26
拥抱微服务 .....	26
持续集成和持续交付 .....	28
启动 .....	28
采用软件组件管理 .....	28
创建 CI/CD 管线 .....	28
部署自动化测试 .....	28
创建文档 .....	29
使用基础设施即代码 .....	29
保留和跟踪标准指标 .....	29
推进 .....	30
使用配置管理 .....	30
集成监控和日志记录 .....	30
创建合并节奏 .....	30

捕获部署后的行为 .....	30
Excel .....	31
集成的人工智能/机器学习技术 .....	31
采用混沌工程实践 .....	32
优化性能 .....	32
实施高级可观测性 .....	32
实施 GitOps 实践 .....	33
结论 .....	34
延伸阅读 .....	35
贡献者 .....	36
文档历史记录 .....	37
术语表 .....	38
# .....	38
A .....	38
B .....	41
C .....	43
D .....	45
E .....	49
F .....	50
G .....	52
H .....	53
我 .....	54
L .....	56
M .....	57
O .....	61
P .....	63
Q .....	65
R .....	65
S .....	68
T .....	71
U .....	72
V .....	73
W .....	73
Z .....	74
.....	lxxv

# AWS Cloud Adoption Framework：平台视角

Amazon Web Services ( [贡献者](#) )

2023 年 10 月 ( [文档历史记录](#) )

数字化转型是高管改善客户体验、创新和灵活性的最大推动力。它使用机器学习 ( ML )、人工智能 ( AI )、大数据以及云的速度和规模，来满足不断变化的业务条件和不断变化的客户需求。

[Amazon Web Services \( AWS \)](#) 是世界上最全面、采用最广泛的云平台。它可以帮助您实现组织转型，同时降低业务风险，改善环境、社会和治理 ( ESG ) 绩效，提高收入并提升运营效率。

[AWS Cloud Adoption Framework \( AWS CAF \)](#) 使用 AWS 最佳实践来帮助您加速实现业务成果。使用 AWS CAF 来识别转型机会并确定其优先级，评估和改善您的云准备情况，并迭代发展您的转型路线图。

AWS CAF 将其指引分组为六个视角：业务、人员、治理、平台、安全和运营。每种视角均包含在单独的指南中。本指南涵盖平台视角，其重点是通过企业级、可扩展的混合云环境加快云工作负载的交付。

# 简介

数百万客户（包括增长最快的初创企业、最大的企业和领先的政府组织）都在使用 AWS。（请参阅 AWS 网站上的[客户成功案例](#)。）他们可以[迁移与现代化](#)传统工作负载，变得更加[以数据为导向](#)，[自动化和优化](#)业务流程，以及重塑运营模式。他们能够通过降低业务风险、改善环境、社会和治理（ESG）绩效，增加收入及提高运营效率，来改善[业务成果](#)。

一系列[基础能力](#)增强了组织有效利用云进行[数字化转型](#)（组织云就绪）的能力。能力是指组织利用流程部署资源（人员、技术和任何其他有形或无形资产）以实现特定结果的能力。AWS CAF 确定这些能力并提供规范指引，全球有成千上万的组织已成功使用这些指引来改善云准备情况并加速云转型之旅。

AWS CAF 将其能力分为六个视角：

- [业务](#)
- [人员](#)
- [治理](#)
- [平台](#)
- [安全性](#)。
- [操作](#)

“平台”视角重点是通过企业级、可扩展的混合云环境加快云工作负载的交付。此环境包含如下图所示的七种能力。这些能力由在[云转型之旅](#)中与功能相关的利益相关者管理。典型利益相关者包括首席技术官（CTO）、技术负责人、架构师和工程师。

## AWS CAF Platform Perspective Capabilities

### Platform Architecture

*Establish guidelines, principles, patterns, and guardrails for your cloud environment*

### Data Engineering

*Automate and orchestrate data flows throughout your organization*

### Data Architecture

*Design and evolve a fit-for-purpose analytics and data architecture*

### Provisioning and Orchestration

*Create, manage, and distribute catalogs of approved cloud products to end users*

### Continuous Integration and Delivery

*Rapidly evolve and improve applications and services*

### Platform Engineering

*Build a compliant cloud environment with enhanced security features and packaged, reusable products*

### Modern Application Development

*Build well-architected cloud-native applications*

本指南以下各部分将详细讨论这些能力。各节都提供了有关如何开始、提高和最终在特定能力方面表现出色的准则。

- [平台架构](#)
- [平台工程](#)
- [数据架构](#)

- [数据工程](#)
- [预调配和编排](#)
- [现代应用程序开发](#)
- [持续集成和持续交付 \( CI/CD \)](#)

平台视角是 AWS CAF 的关键部分。它是从所有其他视角做出的决策汇聚在一起的纽带，以提供业务敏捷性和价值。这里做出的决策将从根本上帮助或阻碍您的业务目标。AWS CAF Platform 视角有助于创建企业级、可扩展的云环境，为组织转型提供支持。从这个视角来看，AWS CAF 指导您建立一个稳健的平台，该平台可以实现您的云之旅，最终导致重大的业务转型和增长。

在从平台角度进行工作时，要考虑与企业领导者之间需要培养的跨职能联系，以及他们为团队和组织带来的价值。将更多精力放在运营模式更改和团队拓扑上，以确保满足要求。此外，还要培养团队构建平台所需的技能，并使其能够在应用程序团队中使用。在做出这些决策时，记住组织的人员、业务、治理、安全和运营目标，这些目标对于确保平台的采用和您的努力取得成功至关重要。

AWS 和 [AWS 合作伙伴网络](#) 提供工具和服务（例如讲习会和培训），可以帮助您完成实施和改进安全状况的这一旅程。[AWS 专业服务团队](#) 是一个由专家组成的全球团队，他们可以通过一系列与 AWS CAF 一致的产品来帮助您实现与云转型相关的具体成果。

# 平台架构

为您的云环境制定并维护准则、原则、模式和护栏。

[架构完善的云环境](#)可帮助您加快实施、降低风险并推动云采用。平台架构功能可让您的组织内部就推动云采用的企业标准达成共识。您可以定义最佳实践蓝图和护栏，以促进身份验证、安全、联网及日志和监控。此外，您还要考虑并规划由于延迟、数据处理或数据驻留要求而可能需要在本地保留的工作负载，并评估混合云使用案例，例如云爆发、云备份和灾难恢复、分布式数据处理及边缘计算。

## 启动

### 定义多账户策略

一个好的[多账户策略](#)会考虑规模和运营效率问题。这意味着将您的[工作负载隔离](#)到最能满足您的运营需求的逻辑模式中。我们建议您从一组基础账户开始，以适应企业中的集中式和分散式服务。您可以集中管理安全、财务和运营职能，以有效管理及治理分布式和自主的团队和账户。您需要在整个组织中保持一致，以了解平台和工作负载将如何分段和管理。了解此结构有助于确保身份验证和授权的安全原则到位，同时与平台不断发展的可接受使用策略保持一致。

### 定义预防性控制

使用一组嵌入式默认控制（护栏），规划安全的多账户环境。开始了解并使用诸如[服务控制策略（SCP）](#)之类的机制，来管理整个组织的服务使用情况，包括云平台中可供使用的 AWS 区域。策略提供了一种集中式机制，用于控制所有账户的最大可用权限，并确保它们遵守组织的访问控制准则。

### 定义组织单元结构

组织单元（OU）是根据监管要求和软件开发生命周期（SDLC）环境对账户进行管理和分类的实用方法。通过使用 OU，组织可以简化在云基础设施中申请适当策略和权限的流程。[工作负载 OU](#) 专为支持应用程序基础设施资源的账户而设计，可确保强制执行正确的策略。使用 OU 和 SCP 有助于增强组织云基础设施的安全性和合规性，同时还能确保应用程序和服务的平稳运行。这最终会带来更高效、更稳健的云采用流程。

### 定义网络连接

[网络连接](#)是任何支持创建安全、可扩展且高度可用的网络以支持应用程序和工作负载的云基础设施的关键方面。精心设计的网络可提供始终如一的高性能，并确保在不同的环境中实现无缝运行。

在设计网络架构时，考虑是否由于延迟、数据处理或数据驻留要求而希望保留在**本地**的工作负载。通过评估混合云[使用案例](#)（例如云爆发、云备份和灾难恢复、分布式数据处理及边缘计算），您可以确定以下方面的关键要求：

- 与互联网的连接。这方面涉及在您的应用程序或工作负载与互联网之间提供安全可靠的连接。此连接对于促进访问基于 Web 的资源、实现用户和应用程序之间的通信以及确保公众在需要时可以访问您的服务至关重要。
- 跨云环境的连接。本区域侧重于在云基础设施内的各种组件和服务之间建立稳健的连接。它可确保在不同的云服务之间轻松共享和访问数据和资源，从而促进高效协作和更顺畅的运营。这里的一个关键考虑因素是您对[虚拟私有云 \(VPC\)](#) 的使用。为了简单起见，请考虑制定有关如何创建和跟踪 VPC 的标准。考虑以编程方式创建这些标准，并计划使用 [IP 地址管理 \(IPAM\)](#) 解决方案。分配足够的 IP 空间以允许增长，并设计子网结构，以便在使用多个可用区时轻松排查问题。在设计和实施网络连接时，务必遵循 [VPC 的安全最佳实践](#)。
- 本地网络与云环境之间的连接。这方面涉及本地基础设施与基于云的环境的集成。通过在两者之间建立安全可靠的连接，组织可以从混合架构的优势中受益。例如，您可以同时使用本地资源和云服务，以提高性能、增强可扩展性和优化成本。

通过解决网络连接的这三个关键领域，您可以构建稳健的云基础设施，有效支持您的应用程序和工作负载，从而充分利用云采用的好处。请记住网络要求，并创建一个简单的设计，使您能够根据自己的多账户策略进行扩展。

## 定义 DNS 策略

精心策划的 DNS 策略可帮助您在云环境增长时避免复杂情况。如果您保持本地 DNS 功能，我们建议您设计使用本地 DNS 基础设施和云 DNS 的[混合 DNS 架构](#)，以满足任何基于云的 DNS 要求。使用解析器端点和转发规则，将 DNS 解析与本地 DNS 环境集成。使用私有托管区来保存有关云 DNS 如何响应一个或多个网络内域及其子域的查询的信息。

## 定义标记标准

标记资源是有效管理成本和确定资源所有权的必要做法。请考虑您的组织将如何进一步促进云使用，包括在平台内使用特定服务。定义标记策略，以跟踪哪些团队正在部署哪些资源。获取 [AWS CAF 运营视角](#) 的意见，并使用标签自动执行已部署基础设施的任务。

此外，通过使用相关元数据标记资源，您可以从 [AWS CAF 治理视角](#) 根据云财务管理 (CFM) 功能中规定的组织要求对支出进行分组和跟踪。确定支持您的会计和财务实践的报告机制，包括在违反财务策略时应采取的操作。

## 定义可观测性策略

制定可观测性策略是优化和保护云架构的关键步骤。该策略的核心是将云服务产生的指标和日志转化为可用于战略决策的可操作的洞察。优先监控关键绩效指标并设置提醒，以提前解决潜在问题。为了防止工具泛滥、优化成本并专注于对贵组织最重要的事情，请将此可观测性策略纳入您的平台和应用程序中。如需进一步指引，请参阅我们关于 [Developing an observability strategy](#) ( AWS re:Invent 2022 ) 的演示。

## 推进

### 定义主动控制和检测控制

为了取得进展，您的组织必须确定环境中是否需要主动控制和检测性控制（护栏）。创建策略，以定义角色和用户和组织单元（OU）内的账户中的护栏或限制。查看平台的所有默认检测性护栏，然后选择要应用的护栏。根据需要创建其他预防性和检测性控制，并按 OU 对它们进行分组，使其与您的多账户策略保持一致。考虑需要哪些组织工具和机制，来检查由检测性控制发现的不合规资源。

### 定义服务入门标准

为平台的可接受使用、与服务消费关联的模式以及如何管理服务制定标准。考虑允许使用哪些初始服务。创建一份概述这些标准的文档，并将其发布给平台的用户和操作者。确保这些标准随着时间的推移而不断调整，以满足组织不断变化的目标和不断变化的云计算能力。

### 定义模式和原则

通过采纳应用程序所有者的意见，考虑组织内允许采用哪些架构模式，并开始定义标准化的蓝图。随着云规模的扩展，标准化可以实现更好的治理并减轻管理负担。定义将使用基础设施即代码（IaC）的模式，并使用集成到更改控制流程和 IT 服务管理（ITSM）系统的服务目录来规划简化的部署模式。定义如何使用这些蓝图以及允许例外的情况。规划这些例外情况及其治理，同时考虑身份验证、安全监控和护栏。

## Excel

### 定义修复模式

考虑如何注释您的检测性护栏调查发现并确定其优先顺序，以便根据您的安全与合规框架对其进行修复。计划使用自动化来检测不符合策略的资源预调配，包括违反预算和标签策略的资源预调配。在更新

运行手册和行动手册时，确定设定和衡量服务水平目标所需的能力。定期审查这些实践，并建立反馈机制，以捕获与平台演变相关的数据。定义相应地创建和更新运行手册和行动手册的机制。

## 沟通和完善策略

为所有文档创建集中式内容管理系统，并将其分发给平台的用户和操作者。创建一种机制来收集反馈，以便将来考虑策略的更改。

## 了解财务管理能力

组织若能保持对预算的透明且全面的了解，就能蓬勃发展。这使他们能够做出明智的决策，高效地分配资源，并实现战略目标。清晰的预算视图有助于组织做出明智的决策、有效的资源分配、成本控制、绩效衡量以及保持问责制和合规性，从而帮助组织脱颖而出。最终这将造就一个更高效、财务更稳定、更繁荣的组织。当您拥有成功的标记策略时，您可以使用 [AWS Budgets](#) 中的成本筛选条件，根据资源标签筛选费用。这可帮助您创建针对特定项目、部门、环境或其他标准量身定制的预算，从而进一步增强财务管理能力。您可以将 [成本分配标签](#) 和 [AWS Cost Categories](#) 与标签关联，以便在报告成本时提高财务洞察力和透明度。

# 平台工程

通过打包的、可重复使用的云产品构建安全、合规的多账户云环境。

为了支持开发团队进行创新，该平台需要快速适应，以满足业务需求。（请参阅 [AWS CAF 业务视角](#)。）它必须做到既足够灵活以适应产品管理需求，又足够严格以遵守安全限制，并且足够快速以满足运营需求。此过程需要构建一个合规的多账户云环境，该环境具有增强的安全功能和打包的、可重复使用的云产品。

有效的云环境使您的团队可以轻松预调配新账户，同时确保这些账户符合组织策略。一组策管式云产品可以帮助您规范最佳实践，帮助您进行治理，并帮助提高云部署的速度和一致性。部署最佳实践蓝图，以及检测性和预防性[护栏](#)。将您的云环境与现有环境[集成](#)，以实现所需的混合云使用案例。

自动执行账户预调配工作流程，并使用[多个账户](#)来支持您的安全和治理目标。在本地环境和云环境之间以及不同云账户之间设置连接。在现有身份提供者（IdP）和云环境之间实现[联合身份验证](#)，以便用户可以使用现有登录凭证进行身份验证。集中日志记录，建立跨账户安全审计，创建入站和出站 DNS 解析程序，并获得对账户和护栏的控制面板可见性。

根据企业标准和配置管理，评估和认证可供使用的云服务。将企业标准打包并不断改进为自助部署产品和消耗性服务。利用[基础设施即代码（IaC）](#)以声明式方式定义配置。创建支持团队，向开发者和业务用户宣传该平台，并允许他们构建集成，从而加快整个组织的采用速度。

要完成以下各部分中讨论的任务，您需要培养[能力](#)和团队，使您的组织朝着现代平台工程方向发展。有关技术详细信息，请参阅 [Establishing your Cloud Foundation on AWS](#) 白皮书。

## 晚于

### 构建登录区并部署护栏

当您开始迈向成熟的平台工程之旅时，必须首先按照平台架构功能中定义的方式，部署具有检测性和预防性护栏的[登录区](#)。护栏可确保应用程序所有者在使用云资源时不会违反组织标准。通过此机制，您可以自动执行账户预调配工作流程，以使用[多个账户](#)来支持您的[安全](#)和[治理](#)目标。

### 建立身份验证

根据 [AWS CAF 安全视角](#)中规定的标准，在所有环境、系统、工作负载和流程中实施[身份管理和访问控制](#)。对于员工身份，限制 [AWS Identity and Access Management（IAM）](#) 用户的使用，改为依赖允许您在集中位置管理身份的身份提供者。这样可以更轻松地管理跨多个应用程序和服务的访问权限，

因为您可以从一个位置创建、管理和撤消访问权限。使用现有流程来管理对 AWS 环境的访问权限的创建、更新和删除。

## 部署网络

根据您的[平台架构](#)设计，创建[集中式网络账户](#)，以控制进出您环境的入站和出站流量。我们建议您设计网络，以便在本地网络与 AWS 环境之间、与互联网之间以及跨 AWS 环境之间快速配置连接。集中网络管理使您能够部署网络控制，通过使用预防性和被动控制来隔离整个环境中的网络和连接。

## 收集、汇总和保护事件和日志数据

使用 [Amazon CloudWatch 跨账户可观察性](#)。它提供了一个统一的界面，用于搜索、可视化及分析关联账户中的指标、日志和跟踪记录，并消除账户边界。

如果您的组织对集中日志控制 and 安全性具有特定的合规性要求，请考虑设置一个专用的[日志归档账户](#)。这提供了一个专门用于日志数据的集中式加密存储库。通过定期轮换加密密钥来增强此归档的安全性。

实施稳健的策略来保护敏感的日志数据，必要时使用[掩蔽技术](#)。将日志聚合用于合规、安全和审计日志，并确保使用严格的防护栏和身份结构，以防止未经授权更改日志配置。

## 建立控制

根据 [AWS CAF 安全视角](#)的定义，部署满足您业务需求的基础[安全功能](#)。部署额外的[预防性](#)和[检测性控制](#)，并根据需要在所有账户中以编程方式一致地预调配这些控制。将检测性控制集成到平台架构功能所定义的运营工具中，以便运营机制可以审查不合规资源。

## 实施云财务管理

根据 [AWS CAF 治理的视角](#)，实施成本分配标签和 AWS Cost Categories，使组织的标签策略与云消费的财务责任保持一致。AWS Cost Categories 允许您使用诸如[AWS Cost Explorer](#)中发布的账单数据之类的工具，向内部成本中心收取或显示云费用[AWS 成本和使用情况报告](#)。

## 推进

### 构建基础设施自动化

在继续操作之前，根据您的[平台架构](#)对云服务进行评估和认证，以供使用。然后，将企业标准打包并不断改进为可部署的产品和可消费的服务，并使用基础设施即代码 ( IaC ) 以声明式方式定义配置。基础

设施自动化通过基于角色的访问权限控制 ( RBAC ) 或基于属性的访问权限控制 ( ABAC ) 来允许访问每个账户中的特定服务，从而模拟软件开发周期。通过使用或开发自助服务功能，部署一种方法来快速配置新帐户 APIs，并使其与您的服务和事件管理功能保持一致。创建账户时自动进行网络集成和 IP 分配，以确保合规和网络安全。通过使用配置为与 AWS 协同工作的原生连接器，将新账户与您的 IT 服务管理 ( ITSM ) 解决方案集成。根据需要更新您的行动手册和运行手册。

## 提供集中式可观测性服务

为了实现有效的[云可观测性](#)，您的平台应支持对本地和集中式日志数据的实时搜索和分析。随着运营规模的扩展，平台索引、可视化和解释日志、指标和跟踪的能力是将原始数据转化为可操作洞察的关键。

通过关联日志、指标和跟踪，您可以提取可操作的结论，并制定有针对性、明智的响应。制定规则，允许对日志、指标或跟踪中识别的安全事件或模式做出主动响应。随着 AWS 解决方案的扩展，请确保您的监控策略同步扩展，以保持和增强您的可观察性能力。

## 实施系统管理和 AMI 治理

广泛使用 Amazon Elastic Compute Cloud ( Amazon EC2 ) 实例的组织需要操作工具来大规模管理实例。软件资产管理、端点检测和响应、库存管理、漏洞管理和访问管理是许多组织的基础功能。这些功能通常通过安装在实例上的软件代理来实现。开发将代理和其他自定义配置打包到亚马逊机器映像 ( AMI ) 中的功能，并向云平台的用户提供这些 AMI。使用预防性和检测性控制来治理这些 AMI 的使用。AMI 应包含能够大规模管理长时间运行的 EC2 实例的工具，特别是对于不经常使用新 AMIs 实例的可变 Amazon EC2 工作负载。您可以大规模使用 [AWS Systems Manager](#) 来自动升级代理、收集系统清单、远程访问 EC2 实例以及修补操作系统漏洞。

## 管理凭证使用

根据 [AWS CAF 安全视角](#)，实现角色和临时凭证。使用工具通过使用预安装的代理来管理对实例或本地系统的远程访问，而无需存储机密密钥。减少对长期凭证的依赖，在 IaC 模板中扫描硬编码凭证。如果您无法使用临时凭证，请使用应用程序令牌和数据库密码等编程工具来自动轮换和管理凭证。使用 IaC 的最低权限原则对用户、组和角色进行编码，并通过使用护栏来防止手动创建身份账户。

## 建立安全工具

安全监控工具应支持跨基础设施、应用程序和工作负载的精细安全监控，并为模式分析提供聚合视图。与所有其他安全管理工具一样，您应扩展检测和响应 (XDR) 工具，以提供按照 [AWS CAF Security 视角](#) AWS 中定义的要求评估、检测、响应和修复应用程序、资源和环境安全的功能。

# Excel

## 通过自动化获取和分发身份结构

使用 IaC 工具对角色、策略和模板等身份结构进行编码和版本化。使用策略验证工具检查是否有安全警告、错误、一般警告、建议的 IAM 策略更改和其他调查发现。在适当情况下，部署和删除以自动方式提供对环境的临时访问权限的身份构造，并禁止使用控制台的个人进行部署。

## 为跨环境的异常模式添加检测和提醒

主动评估环境中是否存在已知漏洞，并增加对异常事件和活动模式的检测。审查调查发现并向平台架构团队提出建议，以进行能够进一步提高效率和创新的变革。

## 对威胁进行分析和建模

根据 [AWS CAF 安全视角](#) 的要求，对行业和安全基准进行持续监控和测量。在实施检测方法时，确定哪些类型的事件数据和信息最能为您的安全管理职能提供信息。此监控包括多种攻击向量，包括服务使用量。您的安全基础应包括在多账户环境中进行安全日志记录和分析的全面功能，包括关联来自多个来源的事件的能力。使用特定的控件和护栏防止更改此配置。

## 持续收集、审查和完善权限

记录身份角色和权限的更改，并在检测性护栏检测到与预期配置状态的偏差时发出提醒。使用汇总和模式识别工具来审查您的集中式事件集合，并根据需要完善权限。

## 选择、衡量并持续改进您的平台指标

为了使平台成功运营，应建立并定期审查全面的指标。确保它们与组织目标和利益相关者的需求保持一致。使用团队支持和工具采用指标，跟踪平台性能和改进指标，并结合补丁、备份和合规等操作参数。

使用 [CloudWatch 跨账户可观察性](#) 实现高效的指标管理。此服务简化了数据聚合和可视化，以实现明智的决策和有针对性的增强。使用这些指标作为成功指标和变革驱动力，营造持续改进的环境。

# 数据架构

设计和发展适合用途的数据和分析架构。

[精心设计的数据和分析架构](#)对于获得可操作的洞察至关重要。通过设计和发展适合用途的数据和分析架构，组织可以降低复杂性、成本和技术债务，同时从不断增长的数据量中获得宝贵的洞察。通过与 AWS CAF 原则保持一致，企业可以创建与其现有平台无缝集成的数据架构。此调整使组织能够利用现代数据处理和分析技术提供的优势。

数据和分析架构是组织从数据中获取价值的能力的蓝图。它可帮助组织获得新的业务洞察，也是业务增长的催化剂。为了支持业务需求，现代数据架构应与短期和长期业务目标保持一致，且应符合组织的文化和情境需求。在当今世界，成功实施和采用数据和分析架构是基于在正确的时间为正确的消费者提供正确的数据的原则。

这是通过规划和组织如何对组织的数据资产进行物理或逻辑建模、如何保护数据以及这些数据模型如何相互交互来解决业务问题、推导出未知模式并生成洞察来实现的。

## 启动

### 定义总体功能

在当前业务环境中，对于现代数据分析平台而言，从数据中获取价值以支持组织内的各个领域至关重要。[现代数据架构](#)不应采用单一数据架构方法，而应包括针对特定使用案例专门构建和优化的工具集和模式。该架构应该能够不断发展，并包括基本构建块，例如可扩展的数据湖、专用分析服务、统一的数据访问和统一治理。

### 整理数据区域

如何整理和存储数据以便快速便捷地访问是数据架构的一个关键方面。这可以通过在数据湖中设置自定义数据区域来实现。数据区域分为以下几类：

- 从异构来源收集的原始数据
- 经过整理和转换的数据，用于支持各领域的分析需求。
- 满足报告需求的使用案例或基于产品的数据集市
- 具有安全性和合规性控制的外部公开数据

## 规划数据的敏捷性和民主化

分析平台的有效性取决于数据预调配的速度以及使可供消费的预调配数据民主化。数据预调配敏捷性是通过数据架构能够根据使用案例以多种方式获取和处理数据来实现的，例如实时、近乎实时、批处理、微批处理或混合。数据民主化是通过定义由数据管家监控的数据共享和访问控制工作流程来实现的。实施数据市场是实现数据民主化的推动因素之一。

### 定义安全数据交付

现代数据架构对外部世界来说是一个安全堡垒，但允许员工或数据用户根据其工作职能轻松访问，并遵守合规限制，例如[健康保险流通与责任法案 \(HIPAA\)](#)、个人信息 (PII)、[通用数据保护条例 \(GDPR\)](#) 等。这是通过基于角色的访问控制 (RBAC) 和基于标签的访问控制 (TBAC) 方法来实现的。在 AWS 上，标签用于控制对数据的访问，从而简化访问控制管理。请按照 [AWS CAF 安全视角](#) 中概述的原则执行此操作。

### 制定成本效益计划

传统数据仓库将计算和存储紧密耦合，资源利用成本很高。现代架构将计算和存储解耦，并根据数据生命周期实现分层存储。例如，在 AWS 上，您可以使用 [Amazon Simple Storage Service \(Amazon S3\)](#)，来控制成本并将数据存储与计算解耦。[Amazon S3 存储类别](#) 是专门为不同的访问模式提供成本极低的存储而设计的。此外，AWS 计算工具 (例如 [Amazon Athena](#)、[AWS Glue](#)、[Amazon Redshift](#) 和 [Amazon SageMaker Runtime](#)) 是无服务器的，因此您无需管理基础设施，只需按使用量付费。

## 推进

可以进一步增强现代数据架构，以增加数据使用的广度 (从支持业务和运营职能的标准分析到支持预测和洞察的更复杂的功能)，并有助于支持更快的决策。为实现此目的，该架构支持以下各部分描述的功能。

### 了解特征工程

[特征工程](#) 使用机器学习，涉及设置特征存放区或特征集市。数据科学团队为监督和无监督学习模型创建新特征 (派生属性)，并将其存储在特征集市中，以简化转换并提高数据准确性。企业可以在多个分析模型中重复使用这些功能，从而加快上市速度。

### 计划对数据集进行非规范化

构造非规范化数据集或数据集市可以使所需的数据在单一位置随时可用，并提高分析速度，从而显著简化业务用户的数据集。如果精心设计，一条记录可以支持多种使用模式，并缩短整个开发生命周期。有

效治理非规范化数据集也很重要，原因有二。实施非规范化数据可能会产生大量冗余数据集，这在规模化管理上可能会成为一个挑战。此外，如果这些数据集未正确建模，那么重新利用它们可能会变得越来越难。

## 设计可移植性和可扩展性

大型组织很少将所有应用程序和用户都放在单个数据平台上。其应用程序和数据存储通常分布在传统的本地和云平台上，这使得分析团队难以混合和合并数据。我们建议您根据域、地理位置、业务使用案例等特征对数据进行容器化。这种容器化提高了各种平台和应用程序之间的可移植性，并支持更有效的使用。将数据分割到容器中并通过 API 将其公开，可帮助您更轻松地扩展数据架构。它支持混合的端到端数据流，并帮助本地和基于云的应用程序无缝协作。

## Excel

随着组织内部现代分析架构的发展，通过引入可重用的概念来管理这种变化非常重要。这些概念在提高持久性和采用率的同时，还能控制成本。以下各部分将讨论一些要考虑的概念。

## 设计可配置的框架

各组织通常会创建多个复杂的模型来满足其独特的业务需求。这些模型需要创建多个数据管线和工程特征。随着时间的推移，这会造成大量冗余并增加运营成本。创建包含一组参数驱动的可配置基础模型的框架可缩短开发时间并降低运营成本。分析引擎可以实现这些可配置的模型，来提供所需的输出。

## 计划构建统一的分析引擎

业务问题是独一无二的，通常需要自定义技术来满足需求，从而导致一个组织中有多个分析引擎。设计和开发可支持多种编程范例的基于人工智能的统一分析引擎接口，可简化使用并降低成本。

## 定义 DataOps

大多数数据专业人员花费大量时间执行数据操作，例如查找正确的数据、转换、建模等。拥有敏捷的数据操作 (DataOps) 可以打破数据工程师、数据科学家、数据所有者和分析师的孤岛，从而极大地增强数据架构。DataOps 可以改善团队之间的沟通，缩短周期时间，并确保高数据质量。随着时间的推移，由于业务需求的变化和技术进步，数据和分析架构经历了多次变革。组织必须努力开发、实施和维护一个能够随着时间的推移而演变并支持其业务的数据和分析架构。

# 数据工程

自动化和编排组织中的数据流。

使用元数据自动化用于处理原始数据并生成优化输出的[管线](#)。利用 AWS CAF 平台架构和平台工程能力及运营视角中定义的现有架构护栏和安全控制。与平台工程支持团队合作，为简化管线部署的常见模式开发可重复使用的[蓝图](#)。

## 启动

### 部署数据湖

通过为结构化和非结构化数据使用合适的存储解决方案，建立基础数据存储能力。这使您能够收集和存储来自各种来源的数据，并使这些数据可供进一步处理和分析。数据存储是数据工程策略的关键组成部分。精心设计的数据存储架构使组织能够经济高效地存储、管理和访问数据。AWS 提供各种数据存储服务，来满足特定的业务需求。

例如，您可以使用 [Amazon Simple Storage Service \( Amazon S3 \)](#) 用于对象存储、使用 [Amazon Relational Database Service \( Amazon RDS \)](#) 用于关系数据库及使用 [Amazon Redshift](#) 用于数据仓库，从而建立基础数据存储能力。这些服务可帮助您安全、经济高效地存储数据，并使数据易于访问以供进一步处理和分析。我们建议您同时实施数据存储最佳实践（例如数据分区和压缩），以提高性能并降低成本。

### 开发数据摄取模式

要自动化和编排数据流，请建立数据摄取流程，以从各种来源（包括数据库、文件和 API）收集数据。您的数据摄取流程应支持业务敏捷性，并将治理控制考虑在内。

编排工具应该能够运行基于云的服务，并提供自动调度机制。它应该为任务之间的条件链接和依赖关系提供选项，以及轮询和错误处理功能。此外，它还应与提醒和监控系统无缝集成，以确保管线平稳运行。

一些热门的编排机制包括：

- 基于时间的编排以递归间隔和定义的频率启动工作流程。
- 基于事件的编排会根据发生的事件（例如创建文件或 API 请求）启动工作流程。
- 轮询实现了一种机制，在这种机制中，任务或工作流程调用服务（例如，通过 API）并等待定义的响应，然后再继续下一步。

现代架构设计强调利用托管服务来简化云中的基础设施管理，并减轻开发者和基础设施团队的负担。此方法也适用于数据工程。我们建议您在适用的情况下使用托管服务来构建数据摄取管线，以加快数据工程流程。这些类型的服务的两个示例是 Amazon Managed Workflows for Apache Airflow ( Amazon MWAA ) 和 AWS Step Functions :

- Apache Airflow 是一款热门编排工具，用于以编程方式创作、安排和监控工作流程。AWS 将 [Amazon Managed Workflows for Apache Airflow \( Amazon MWAA \)](#) 作为托管服务提供，使开发者能够专注于构建而不是管理编排工具的基础设施。借助 Amazon MWAA，您可以使用 Python 脚本轻松创作工作流程。有向无环图 ( DAG ) 以显示每个任务的关系和依赖关系的方式将工作流程表示为任务的集合。您可以随心所欲地拥有任意数量的 DAG，Apache Airflow 将根据每个任务的关系和依赖关系来运行它们。
- [AWS Step Functions](#) 可帮助开发者构建低代码的可视化工作流程，用于自动化 IT 和业务流程。您使用 Step Functions 构建的工作流程称为状态机，工作流程的每个步骤都称为状态。您可以使用 Step Functions 为内置错误处理、参数传递、建议的安全设置及状态管理创建工作流程。这减少了需要编写和维护的代码数量。任务通过与您在本地或云环境中托管的其他 AWS 服务或应用程序进行协调来执行工作。

## 加速数据处理

数据处理是理解现代组织收集的大量数据的关键一步。为开始进行数据处理，AWS 提供 [AWS Glue](#) 等托管服务，这些服务提供强大的提取、转换、加载 ( ETL ) 功能。各组织可以使用这些服务开始处理和转换原始数据，包括清理、标准化和聚合数据，为分析做好准备。

数据处理从聚合和筛选等简单技术开始，以执行初始数据转换。随着数据处理需求的发展，您可以实施更高级的 ETL 流程，使您能够从各种来源提取数据，对其进行转换以满足您的特定需求，然后将其加载到集中式数据仓库或数据库中进行统一分析。此方法可确保数据准确、完整，且可以及时进行分析。

通过使用 AWS 托管服务进行数据处理，组织可以从更高级别的自动化、可扩展性和成本效益中受益。这些服务可自动执行许多常规数据处理任务 ( 例如架构发现、数据剖析和数据转换 )，并释放宝贵的资源用于更具战略意义的活动。此外，这些服务会自动扩展以支持不断增长的数据量。

## 提供数据可视化服务

想办法向使用数据可视化来有意义、快速地解释数据的决策者提供数据。通过可视化，您可以解释模式并提高不同利益相关者的参与度，无论他们的技术技能如何。一个好的平台使数据工程团队能够预调配资源，从而快速提供数据可视化，而且开销很少。您还可以使用无需工程专业知识即可轻松查询数据存储的工具，从而提供自助服务功能。考虑使用内置工具，该工具可以通过数据可视化和交互式控制面板提供无服务器商业智能，并且可以使用自然语言查询后端数据。

# 推进

## 实现近乎实时的数据处理

数据处理是任何数据工程管线的重要组成部分，它使组织能够将原始数据转化为有意义的洞察。除了传统批处理之外，在当今快节奏的业务环境中，实时数据处理也变得越来越重要。实时数据处理使组织能够在事件发生时做出响应，并提升决策能力和运营效率。

## 验证数据质量

数据质量直接影响从数据中得出的洞察和决策的准确性和可靠性。实施数据验证和清洗流程对于确保使用高质量和可信的数据进行分析至关重要。

数据验证包括根据预定义规则 and 标准检查数据，从而验证数据的准确性、完整性和一致性。这有助于识别数据中的任何差异或错误，并确保其符合用途。数据清洗涉及识别和更正数据中的任何不准确、不一致或重复之处。

通过实施数据质量流程和工具，组织可以提高从数据中获得的洞察的准确性和可靠性，从而改善决策和运营效率。这不仅可以提高组织的绩效，还可以提高利益相关者对所产生的数据和分析的信心和信任。

## 证明数据转换服务

数据转换为高级分析和机器学习模型准备数据。它涉及使用诸如数据标准化、充实和重复数据删除之类的技术，来确保数据干净、一致且随时可供分析。

- 数据标准化涉及将数据组织成标准格式，消除冗余，并确保不同来源的数据保持一致。这使得分析和比较来自多个来源的数据变得更加容易，并使组织能够更全面地了解运营。
- 数据扩充涉及使用来自外部来源的额外信息（例如人口统计数据或市场趋势），来增强现有数据。这为客户行为或行业趋势提供了宝贵的洞察，这些洞察仅从内部数据来源中可能看不出来。
- 重复数据删除包括识别和删除重复的数据条目，并确保数据准确无误。在处理大型数据集时，这一点尤其重要，因为即使只有很小比例的重复数据也可能导致分析结果出现偏差。

通过使用先进的数据转换技术，组织可以确保其数据质量高、准确，且可以进行更复杂的分析。这有助于做出更好的决策，提高运营效率，并在市场中获得竞争优势。

## 启用数据民主化

通过让所有员工都能访问、理解和使用数据，促进数据民主化文化。数据民主化可帮助员工做出数据驱动型决策，并为组织的数据驱动型文化做出贡献。这意味着打破信息孤岛，营造一种文化，让所有员工共享和使用数据来推动决策。

总体而言，数据民主化就是要创造一种文化，在这种文化中，组织中的每个人都重视数据、访问和理解数据。通过实现数据民主化，组织可以培养一种数据驱动型文化，这种文化可以推动创新，改善决策，并最终导致业务成功。

## Excel

### 提供基于用户界面的编排

要构建敏捷且使用有效方法的组织，重要的是要规划一个现代化的编排平台，供各业务部门的开发和运营资源使用。目标是在不依赖单一团队、技术或支持模式的情况下开发、部署和共享数据管线和 workflows。这是通过基于用户界面的编排等功能实现的。诸如拖放交互之类的功能使几乎没有技术专业知识的用户能够构建 DAG 和状态机数据流。然后，这些组件可以生成编排数据管线的可执行代码。

DataOps 有助于克服数据管理的复杂性，并确保组织间的数据无缝流动。元数据驱动型方法可确保数据质量和合规性符合贵组织的规定。对微服务、容器化和无服务器功能等工具集的投资可提高可扩展性和敏捷性。

依靠数据工程团队从数据中创造价值，将日常基础设施任务交给自动化，这使组织能够在自动化和编排方面实现卓越的目标。对数据流管理任务的近乎实时的监控和日志记录支持即时修复措施，并提高数据流管线的性能和安全性。这些原则有助于实现可扩展性和性能，同时确保数据共享模型的安全性，并为组织未来的成功做好准备。

## 集成 DataOps

DataOps 是一种现代的数据工程方法，它强调开发和运营流程的集成，以简化数据管线的创建、测试和部署。为了实施 DataOps 最佳实践，组织使用基础设施即代码 ( IaC ) 及持续集成和持续交付 ( CI/CD ) 工具。这些工具支持自动化管线创建、测试和部署，可显著提高效率并减少错误。DataOps 团队与平台工程支持团队合作构建这些自动化，因此每个团队都可以专注于自己最擅长的事情。

实施 DataOps 方法有助于为数据工程师、数据科学家和业务用户营造协作环境，并实现数据管线和分析解决方案的快速开发、部署及监控。此方法提供了更顺畅的跨团队沟通和协作，从而加快了创新速度并取得了更好的成果。

要充分利用 DataOps 的优势，简化数据工程流程非常重要。这是通过使用平台工程团队的最佳实践来实现的，包括代码审查、持续集成及自动测试。通过实施这些实践，组织可以确保数据管线可靠、可扩展和安全，并满足业务和技术利益相关者的需求。

# 预调配和编排

创建、管理和分发已批准的云产品目录给用户。

随着组织的发展，以一致、可扩展和可重复的方式预调配基础设施变得更具挑战性。简化的[预调配和编排](#)可帮助您实现一致的监管并满足合规性要求，同时使用户能够仅部署经批准的云产品。

在组织中重复使用预先批准的产品可让您的开发者更快、更一致地构建应用程序，同时满足组织的安全和治理要求。

## 晚于

### 部署 hub-and-spoke 目录模型

在服务目录中作为投资组合管理的软件资产按 hub-and-spoke 模式与一个或多个账户中的用户共享。您可以使用私有市场和专属优惠来策划各种第三方解决方案，并与您的基础设施即代码 ( IaC ) 模板一起分发。

要使您的构建者能够使用预先批准的产品，请定义一个流程来审核、批准这些产品并将其发布给您的用户。首先设计和实施包含这些预先批准产品的集中管理存储库。设计一个系统，当组织中的用户需要使用每种产品时，授予对此存储库中许可证和产品的访问权限。

允许组织中的构建者向发布机制提交产品以供批准，以便这些产品在获得批准后可供组织中的所有用户使用。

### 整理模板以供重复使用

在为解决方案编写 IaC 模板并定义 hub-and-spoke 模型后，应为每个分支账户定义两类模板：预配置/强制执行和可供使用。已预调配/强制执行模板作为基础功能直接从管理账户预调配到每个成员账户。可供使用模板可供构建者以自助方式浏览和预调配。

### 应用默认参数以便重复使用

实现包含构建者可以预先选择的默认参数的 IaC 模板。这使构建者无需评估每个参数的详细信息即可与治理保持一致，并防止他们做出错误的选择。此方法仅公开设置所需的内容。例如，[AWS Service Catalog](#) 使用约束功能来实现此方法，该功能可控制应用于特定产品组合中产品的规则。当构建者团队使用自助式模板预调配时，会预先配置此自定义。

## 建立审批流程

如果用户有合理的业务理由使用某个产品，即使他们没有获得该产品的访问权限，也应该能够提交访问该产品的请求。构建一个通知系统，在用户正在使用的产品有更新时通知用户，这样他们就可以遵守最新的安全更新。

建立工作流程，让构建者通过自助服务门户提交新产品以供审核。构建者可以使用门户来定义产品的受众，并确定应有权访问该产品的用户组。对于每次提交，请使用您定义的流程来审核、批准产品并将其发布到自助服务门户。

## 推进

### 创建自助服务门户

创建自助服务门户，用于分发、浏览和使用经批准的云产品。组织中的用户可以使用此门户搜索构建基础设施和将应用程序部署到其环境所需的产品。为有权访问门户中产品的用户设置权限边界，并对用户可以使用许可产品的次数设置限制。定义一组基本资源，这些资源可以直接预调配或在每个分支账户中作为自助服务模式提供，因为这些账户是通过使用诸如[自定义 AWS Control Tower](#) 之类的解决方案创建的。

### 启用私有市场

私人市场提供已购买产品（软件、数据和专业服务）的精选目录，并以一种 hub-and-spoke 模式（具有一个管理帐户和多个成员帐户）实施，因此分支账户只能订阅经批准的软件。这种产品治理有助于控制软件成本并简化法律和合同审查。在管理账户级别创建私有市场，作为主要中心。

### 管理权限

启用控制措施，仅允许授权用户和工作负载在供应商定义的限制内使用许可证。这有助于降低昂贵的审计和意外许可调整的风险。

## Excel

### 与采购系统集成

通过将现有采购流程整合到 [AWS Marketplace](#) 中，完善现有采购流程。这是通过将您的采购系统（Coupa 或 SAP Ariba）扩展到私有市场来实现的，以便您的用户可以遵循现有的采购和批准流程来

获取软件。创建相应的 IAM 管理权限，AWS Marketplace 用于生成配置采购解决方案所需的信息，并配置您的采购解决方案以完成集成。例如，您可以[设置 punchout](#)，将采购订单附加到 AWS 发票上，然后调整采购流程以使用标准配置解决方案。

让您的构建者能够通过内部 API 访问预先批准的产品，以使用户可以将产品整合到应用程序，或者构建自己的个性化门户供其团队使用产品。整合用于创建新产品的提交和发布流程，并允许用户通过申请新许可证和访问产品 APIs。

## 与您的 ITSM 工具集成

如果适用，请[使用 IT 服务管理 \(ITSM\) 工具进行连接](#)，并自动更新您的配置管理数据库 (CMDB)。建立流程和机制来评估您的组织使用的产品。建立一种机制，告知预先批准产品的用户需要更新以确保合规性。使用 ITSM 工具分析您的环境，并在需要关键更新时将安全性和合规性更新推送到整个组织的产品。

## 实施生命周期管理和版本分发系统

在整个开发生命周期中维护 IaC 模板的版本以及根据模板预调配的服务版本。您可以使用为目录实施的 hub-and-spoke 模型来定义是否需要在分支级别进行强制更新（例如，是否有并发版本可用于自助配置），以及哪些版本需要标记为过时。使用 hub-and-spoke 目录还有助于根据需要管理新版本的审计和分发。

# 现代应用程序开发

构建架构完善的云原生应用程序。

[现代应用程序](#)开发实践对于组织构建架构完善的云原生应用程序并保持竞争力至关重要。企业可以使用云原生技术（例如[容器和无服务器](#)计算）来创建可扩展的敏捷应用程序，以适应不断变化的市场需求。这些技术使组织能够优化资源利用率、降低成本并提高其应用程序的性能。

在设计现代应用程序时，开发用于运营和开发的敏捷解决方案。现代应用程序会自动对客户需求的变化做出反应，并且能够抵御故障。工程师可以快速开发和部署更改并监控应用程序性能。现代应用程序旨在实现自我修复，且能够在需要时扩展到大型或小型流量水平，包括在零流量时实现零成本。

构建架构完善的云原生应用程序需要对底层技术及其最佳实践有深入的了解。组织应采用微服务架构，并将应用程序设计成模块化和松耦合，从而实现独立部署和可扩展性。此方法使组织能够将其应用程序分解为更小、更易于管理的组件，这些组件可以快速、独立地开发、测试和部署。

## 启动

### 探索现代方法

首先研究容器、无服务器技术和其他支持[微服务](#)开发的方法，这些方法可以提高资源效率、帮助提高安全性并最大限度地减少基础设施开支。选择对现有的差异化应用程序和企业应用程序进行[现代化](#)改造，以提高效率并最大限度地提高现有投资的价值。根据以价值为导向的决策，考虑[更换平台](#)（将自我管理容器、数据库或消息代理过渡到托管云服务）和[重构](#)（重新开发应用程序以采用云原生架构）。

当您更新现有的基于云的应用程序时，成功的方法是使用 [strangler fig 模式](#) 将您的架构逐渐分解为微服务。该程序有助于采用当代应用方法，因此您可以意识到其固有的好处，并向更大的组织展示其价值。考虑将您的应用程序构建为不同的微服务，在适用的情况下利用[事件驱动型架构](#)。确保您的架构考虑到不可更改的[服务配额](#)和物理资源，以免影响工作负载性能或可靠性。

### 采用云原生计算功能

云原生计算能力是现代应用程序开发的关键。此方法要求组织考虑他们希望如何托管计算单元，并为每个使用案例或服务确定最佳选项。例如，[AWS Lambda](#) 提供了一种用于运行应用程序代码的无服务器机制，并在事件驱动型架构中起着关键作用。Lambda 函数按需启动，且可并行运行，直至达到定义的最大并发数，因此它们可以扩展以执行各种任务。

## 使用容器化

在现代软件开发中，管理应用程序及其依赖关系已成为一项越来越复杂的任务，尤其是当您考虑到需要在各种环境中保持一致性时。为了应对这些挑战，诸如 Docker 之类的容器化技术已成为打包应用程序及其依赖关系的有效解决方案。无论应用程序的运行时环境如何，容器都能确保部署的一致性和可重复性，因此本地环境中的开发与云环境中的生产开发相同。此方法可以减少因环境或其配置不匹配可能导致的错误。

## 使用现代数据库

当您使用现代数据库时，应用程序内的每项微服务都可以使用符合其要求的正确专用数据库，从而提高敏捷性和性能，同时降低成本。例如，一个微服务可能使用 NoSQL 数据库在存储会话数据时实现高吞吐量，另一个微服务可能使用关系数据库来进行复杂的表联接，而另一个微服务可能使用量子分类账数据库来跟踪区块链的变化。

现代数据库提供了可扩展性和灵活性。与传统数据库相比，它们还有助于提供更好的安全性、合规性和可靠性。它们使组织能够更有效地存储和管理数据，并确保应用程序能够在正确的时间访问正确的数据，从而带来更好的性能和用户体验。

迁移到现代数据库是现代应用程序开发的关键组成部分。通过使用正确的数据存储解决方案，组织可以优化数据管理能力，并提供更高效、更可靠的应用程序。通过使每项微服务独立并为每项微服务选择正确的技术，组织可以进一步优化数据能力，从而在最大限度地降低成本的同时实现最高的效率和可扩展性。

## 推进

### 优化您的现代架构

为了进一步优化，请改进无服务器技术的实现，并开发可独立扩展和部署的架构，利用 AWS 服务（例如 [Amazon API Gateway](#) 和 [AWS Lambda](#)）来实现。通过使用 [Amazon Route 53](#) 和 [AWS Cloud Map](#) 实现服务发现，以确保组件之间的无缝通信。

采用 API 版本控制、缓存和速率限制，以保持不同应用程序版本的兼容性和性能。使用 [AWS Identity and Access Management \(IAM\)](#) 和资源策略增强安全性。这有助于确保您的基础设施受到保护，并且仅向授权实体授予访问权限。

如果可能，请在无需管理底层基础设施的情况下使用无服务器服务来运行容器。这使您能够专注于开发核心应用程序，并实现更好的资源管理和性能。还可以帮助您充分利用可扩展性、灵活性和成本效率的优势。

通过深入研究无服务器架构的复杂性并整合这些高级实践，组织可以发现改进和微调的机会，并最终最大限度地发挥云原生应用程序的潜力。这种追求有助于采用更复杂的应用程序模式，从而进一步提升整体用户体验。还使组织能够在软件开发过程中变得更加敏捷和高效。

## 使用服务网格技术

随着组织越来越多地采用微服务架构来构建和部署应用程序，管理这些服务之间的复杂性、安全性和通信变得至关重要。Istio、Linkerd 或 Consul 等服务网格技术在帮助增强微服务的安全性、可观测性和可靠性方面发挥着关键作用。

## 确保可见性和可追溯性

现代实践在开发过程中提供了更高的可见性和可追溯性，并使遵守行业标准和最佳实践变得更加容易。可见性和监控对于现代应用程序开发至关重要。实施监控和日志记录解决方案以提供有关应用程序性能的宝贵洞察，使组织能够确定需要改进的领域并优化应用程序。我们建议您与平台工程团队合作，确保有工具可用来提供端到端可见性，并监控应用程序错误、性能和合规性，以便您可以快速检测、诊断和解决问题。

## Excel

### 拥抱微服务

对于许多组织来说，现代应用程序开发是业务成功的代名词。微服务是此转型的核心，组织可以从采用这些强大的架构模式中受益。

微服务提供了高度可扩展、韧性强且敏捷的应用程序架构。通过将应用程序分解为可独立部署的小型服务，组织可以选择在不影响应用程序其他部分的情况下快速迭代特定组件。高级韧性模式（例如断路器和隔板）在确保这些应用程序的高可用性方面起着至关重要的作用。

[断路器](#)充当一种安全机制，它通过暂时停止或转移来自运行状况不佳的通信来防止级联故障，从而使其能够恢复。[隔板](#)隔离资源并限制潜在故障的影响范围。这些模式共同构成了一个稳健的架构，可以承受不可预见的中断并保持最佳性能。

实施微服务的另一个关键方面是域驱动设计（DDD）原则的采用。DDD 专注于建立对业务领域的共同理解，并将其转化为结构良好的软件设计。此方法可以构建更具凝聚力和可维护性的微服务，并确保应用程序能够随着组织的需求而不断发展。

在基于微服务的应用程序中，优化服务间通信也至关重要。通过实施 gRPC 或 GraphQL 等高级协议，组织可以显著提高服务之间的通信效率。这些协议提供了类型安全、低延迟和灵活性等功能，有助于提高应用程序的整体性能和可维护性。

采用微服务的组织提供了一个促进创新、敏捷性和协作的环境。开发团队通常围绕业务能力组织，且非常注重持续集成和持续交付（CI/CD）实践。他们有权做出决策、进行实验和快速迭代，且秉持责任担当和问责的文化。

# 持续集成和持续交付

与使用传统软件开发和基础设施管理流程的组织相比，更快地演进和改进应用程序及服务

采用具有[持续集成](#)和[持续交付](#) ( CI/CD ) 的 [DevOps](#) 实践可促进构建、测试和部署应用程序的简化、自动化和高效流程。CI/CD 可实现软件的快速交付，降低部署错误的风险，并确保应用程序始终保持最新功能和错误修复。主要目标是通过从传统的软件开发和基础设施管理流程演变出来，以更快的速度发展和改进应用程序和服务。

## 启动

### 采用软件组件管理

软件组件管理是管理用于构建软件的所有个别组件的实践，包括库、框架、源代码存储库、模块、构件和第三方依赖关系。我们建议您使用版本控制系统（例如 Git 或 Apache Subversion）来管理源代码、启用协作和维护代码更改历史记录。您可以监控存储库中的更改和事件，以实现流程自动化、创建管线、管理代码，并根据需要将工作流程与其他服务集成。

### 创建 CI/CD 管线

CI/CD 管线是由提交到版本控制系统的更改启动的一组自动指令。它们通常包括构建应用程序、运行自动测试以及将代码部署到特定环境的说明。您可以使用 [AWS CodePipeline](#)、Jenkins、GitLab 或 CircleCI 等工具设置自动的 CI/CD 管线。您也可以直接在支持管线生成的版本控制系统中进行设置。

从构建一个最小可行的持续集成管线开始，然后逐步演进到包含更多操作和阶段的[持续交付](#)管线。将您的持续交付配置视为代码。您可以为每个分支和团队使用多个不同的管线，因此请考虑需要设置哪些配置变量，以及如何最好地支持将要使用这些管线的团队。

考虑部署时段，即您要部署代码的日期和时间。考虑您的系统需求低谷时段，这样即使需要回滚，也能将对客户的影响降到最低。其他最佳实践包括避免在星期五进行部署，以及在高峰期或节假日之前实施代码冻结。考虑定义有关在提交作者不在场时（例如，在度假时）部署代码的规则。请记住，部署会失败，您可能需要依赖外部帮助。评估不同的[部署方法](#)，例如就地部署、滚动部署、不可变部署和蓝绿部署。考虑使用完全托管式服务来实现持续交付工作流程，以提高可用性和安全性，同时最大限度地减少复杂性和管理工作。

### 部署自动化测试

现代实践建议左移（将测试移到更靠近开发者和 [IDE](#) 的位置，并在生命周期的早期阶段进行），以在问题提交到存储库并启动管线之前检测和修复问题。此实践涉及与开发者进行快速反馈循环，因为开发

者在编码时会检测到错误。左移成本较低，因为测试不需要运行流水线，这可能会导致异步反馈和更高的运营成本。

自动测试可在开发过程的早期发现错误，包括单元测试、集成测试及功能测试。我们建议您鼓励[开发者尽早使用工具](#)创建单元测试，并在将代码推送到中央存储库之前运行这些工具。此外，请确保您的自动化流程包括[静态代码分析](#)、性能基准测试和安全应用程序测试。

## 创建文档

除了实施 CI/CD 管线以简化开发工作流程外，您还应保留清晰而全面的文档，以确保管线的持续有效性、可维护性和可扩展性。文档是 CI/CD 管线的重要方面，因为它使开发团队能够清楚地了解管线的设计、组件和流程。创建文档时，首先要概述管线，解释架构和设计的权衡取舍，描述正在使用的工具和技术，指定初始配置和设置，概述安全措施和访问控制，及包括问题排查和维护信息。

## 使用基础设施即代码

使用 Terraform、Ansible 或 [AWS CloudFormation](#) 等工具来管理基础设施，确保环境的一致性和可重现性。将您的基础设施视为代码，确保跟踪基础设施中的更改，并避免直接在控制台进行更改。将所有基础设施（包括数据库预调配）定义为代码，并使用管线部署这些更改。考虑在管线中将数据库集成作为代码运行，其中包含一小部分经过清理的生产数据。如果可能，进行更改并在代码中跟踪这些更改。

与软件代码一样，遵循以下基础设施代码的最佳实践：

- 使用版本控制。
- 利用错误跟踪和工单系统。
- 让同行在应用更改之前对其进行审查。
- 建立基础设施代码模式和设计。
- 测试基础设施更改。

## 保留和跟踪标准指标

要保持高水平的绩效，请根据关键指标进行开发和跟踪，以了解管线的运行状况和业务影响，包括：

- 构建频率。构建的数量可以让您深入了解团队的工作效率和更改的复杂性。
- 部署频率。定期部署表明开发过程运行状况良好、敏捷。
- 变更前置时间。衡量更改投入生产所需的平均时间可以帮助您确定部署过程中的瓶颈。

- 通过管线的平均时间。从初始管线阶段到每个后续阶段的平均时间可以帮助您优化工作流程。
- 生产更改量。跟踪投入生产的更改数量可以深入了解生产环境的稳定性。
- 构建时间。平均构建时间可以表明代码库或基础设施中存在潜在问题。

## 推进

### 使用配置管理

配置管理工具在自动化软件和基础设施的部署、配置和管理方面起着至关重要的作用。它们提供了一种系统性方法来处理更改，并维护各种环境中基础设施、软件和配置的预期状态。这些工具使开发者能够使用声明式或命令式语言来定义所需的系统状态。然后，配置管理工具会自动将这些配置应用于目标系统，从而确保一致性和可重复性。

使用配置管理工具自动部署、配置和管理软件和基础设施。[AWS Systems Manager 状态管理器](#)是一项安全且可扩展的配置管理服务，可自动将您的托管式节点和其他 AWS 资源保持在定义的状态。

### 集成监控和日志记录

将监控和日志记录解决方案集成到 CD 管线中，可以为开发团队和整个软件开发过程带来诸多好处。这些解决方案可以提供对应用程序性能的实时洞察，能够更快地识别和解决问题，并促进持续改进，以帮助确保应用程序在整个生命周期中保持可靠、高性能和可扩展性。投资监控和日志记录解决方案是维护稳健而高效的 CD 管线的关键方面，它最终有助于成功交付高质量的软件。

### 创建合并节奏

每天至少一次将代码更改提交或合并到主线（干线或主线）分支；理想情况下，每次任务完成后提交或合并，一天可多次操作。此节奏会导致每天多次调用管线。基于拉取的分支工作流程模型与此方法一致。使用[功能标志](#)、[暗启动](#)和类似技术来自定义客户使用的功能。

### 捕获部署后的行为

部署后，使用自动综合测试捕获生产行为，并将结果与持续交付管线同步，以确保及时采取纠正措施。开发者的首要任务应该是尽快修复管线中发现的错误，将代码更改提交到源代码存储库，并在管线中验证错误是否已解决。

最佳部署后实践包括观察最重要的关键绩效指标（KPI），并验证生产环境中没有错误。自动处理错误并评估部署后 KPI，以量化发布的影响。自动生成速度、安全性和稳定性指标，供开发者进行改进。有关更多信息，请参阅 AWS 上的解决方案 [DevOps 监控控制面板](#)。

# Excel

采用尖端实践和技术以实现最佳性能。不断完善 CI/CD 流程可帮助您提高软件质量、缩短上市时间并提高敏捷性。新的技术和工具不断涌现，这使得您的组织必须随时了解情况，并进行调整以保持竞争优势。

要保持适应性，请考虑以下事项：

- 将所有内容定义为代码，包括您的应用程序、配置、基础设施、数据、AWS 账户和组织、部署管线、网络以及安全性和合规性控制。
- 为计算映像、共享服务和应用程序创建相应的[部署管线](#)。
- 考虑一个 GitOps 模型，在该模型中，基于拉取的请求通过将现有基础设施状态与所需状态进行比较来启动工作流程来部署更改，如代码中所述。
- 考虑使用 CD 管线来部署机器学习 ( ML )、数据、物联网 ( IoT ) 及其他工作负载。
- 对所有构建构件进行数字签名，并将其存储在安全的存储库中。
- 通过自动生成软件物料清单来跟踪软件来源，该清单记录部署给客户的所有版本和数字签名的构件。
- 在软件交付过程中消除所有手动活动后，删除手动审查委员会。

对于已实现整个软件交付流程自动化的应用程序和服务，请考虑持续部署，在此部署中，团队会将通过管线中所有检查的更改部署到生产中的客户。有关可视化，请参阅 AWS 网站上的[What is Continuous Delivery?](#)。

## 集成的人工智能/机器学习技术

将人工智能 ( AI ) 和机器学习 ( ML ) 技术集成到 CI/CD 管线中可带来多项好处，包括：

- 自动测试生成
- 智能测试优先级
- 用于问题检测的预测性分析
- 异常检测和根本原因分析
- 代码审查和质量保证
- 部署优化

有关更多信息，请参阅 AWS 网站上的[将智能添加到开发人员操作中](#)。

## 采用混沌工程实践

混沌工程包括故意向系统注入故障，以测试其承受意外事件并从中恢复的能力。通过识别弱点并主动加以解决，组织可以提高其整体系统可靠性并最大限度地减少潜在问题的影响。

采用混沌工程实践，以使用 Gremlin、Chaos Monkey 或 Litmus 等工具来测试系统的韧性。定期运行受控实验，以识别漏洞，验证容错能力，并确保您的应用程序可以优雅地处理意外故障。此积极主动的方法有助于提高系统的可靠性，并有助于建立更稳健的 CI/CD 管线。

## 优化性能

通过使用剖析工具、实时监控和反馈循环，来持续优化应用程序的性能。应用以下技术来确保您的应用程序可以处理增加的流量和需求：

- 代码优化
- 分析
- 实时监控
- 反馈循环
- 缓存
- 负载均衡
- 可扩展性和性能测试

## 实施高级可观测性

提升云基础设施的可观测性不仅仅是收集、汇总和分析指标、日志及跟踪数据这些基本要素。当使用 [Amazon CloudWatch](#) 和 [AWS X-Ray](#) 等工具增强可观测性时，它就会演变为一种推动持续交付和创新的战略实践。

在稳健的 CI/CD 管线中，高级可观测性使您能够发现洞察，不仅可以了解您的应用程序和基础设施，还可以深入了解整个系统的性能和运行状况，包括管线本身。这些洞察可帮助您：

- 快速识别、了解和解决潜在问题，以提高应用程序稳定性并减少停机时间
- 简化 CI/CD 流程，实现更快、更可靠的交付
- 更深入地了解代码更改和部署的影响，以推动明智的决策
- 优化资源利用，以提高运营效率和成本效益

要提高可观测性，请执行以下操作：

- 将可观测性嵌入应用程序和基础设施的每一层，从而全面了解系统的性能、行为和运行状况。
- 使用 Amazon CloudWatch 等工具集中收集、存储和分析数据，统一您的可观测性数据，便于访问和解释。
- 使用 AWS X-Ray 进行分布式跟踪，以了解您的应用程序及其基础服务的性能。
- 建立反馈循环以实现持续改进，并使用您的可观测性数据来推动系统的迭代增强。

采用高级可观测性不仅仅是为了维护您的系统，更是一项战略举措，旨在实现卓越运营，并推动组织内的持续创新。

## 实施 GitOps 实践

通过使用 Git 存储库作为单一事实来源，实施 GitOps 实践来管理基础设施和应用程序配置。此方法简化了更改管理，增强了可追溯性，并确保跨环境的一致性。

# 结论

本指南可作为成功实施和管理成功采用云的基础的手册。它讨论了如何：

- 直接解决[平台架构](#)中的技术挑战和错综复杂的问题，为您的云环境及驻留在其中的数据制定稳健的准则和原则。
- 通过强大的[预调配和编排](#)来构建[平台工程](#)。
- 支持使用合规的多账户云环境，以可扩展和可重复的方式管理已批准的云产品并将其分发给用户。
- 使用[数据工程](#)所需的工具来支持[数据架构](#)决策，以推动数据驱动型决策。
- 将这些功能与[现代应用程序开发策略](#)和 [CI/CD 流程](#)相结合，以提高组织内部的敏捷性、效率和创新性。
- 构建跨职能关系，并在自己的决策中获取其他 AWS CAF 视角的意见，以确保您的平台及其背后的团队取得成功。

## 延伸阅读

[AWS Cloud Adoption Framework \( AWS CAF \)](#) 资源：

- [电子书](#)
- [有声读物](#)
- [信息图](#)
- [适用于人工智能、机器学习和生成式人工智能的 AWS CAF](#)
- [业务角度](#)
- [人员角度](#)
- [治理角度](#)
- [运营角度](#)
- [安全角度](#)

其他资源

- [AWS 架构中心](#)
- [AWS 案例研究](#)
- [AWS General Reference](#)
- [AWS 术语表](#)
- [AWS Knowledge Center](#)
- [AWS 规范指引](#)
- [AWS 合作伙伴解决方案](#) ( 以前称为“快速入门” )
- [AWS 安全性文档](#)
- [AWS 解决方案库](#)
- [AWS 培训和认证](#)
- [AWS Well-Architected](#)
- [AWS 白皮书和指南](#)
- [开始使用 AWS](#)
- [Overview of Amazon Web Services](#)

# 贡献者

本指南的贡献者包括：

- Tony Santiago , AWS 高级合作伙伴解决方案架构师
- Matias Undurraga , AWS 企业技术专家
- Alex Torres , AWS 高级解决方案架构师
- Michael Rhyndress , AWS 高级 DevSecOps 顾问
- Alex Livingstone , AWS 首席解决方案架构师和 CloudOps 专家
- Bruce Cooper , AWS 首席 SDE
- Ravinder Thota , AWS 高级咨询顾问
- Sausan Yazji , AWS 高级业务经理
- Paul Duvall , AWS DevSecOps 总监
- Jeremy Tennant , AWS 首席云交付经理
- Sneh Shah , AWS 首席基础设施主管
- Sasa Baskarada , AWS AWS Cloud Adoption Framework 全球主管

# 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
<a href="#">初次发布</a>	—	2023 年 10 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：

- Refactor/re-architect — 充分利用云原生功能来提高敏捷性、性能和可扩展性，从而移动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：将您的本地 Oracle 数据库迁移到亚马逊 Aurora PostgreSQL-Compatible 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 Cloud 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：将您的客户关系管理 (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，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 Cloud 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

### A2A () Agent-to-Agent

一种支持任务委托和状态转移的代理到代理协作的状态协议。

## 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。

## 座席

一种能够使用工具自主推理、计划和采取行动来实现目标的人工智能系统。

## 特工行动

在生产环境中大规模构建、测试、部署和运行 AI 代理的操作实践。

## 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？](#)

## 人工智能运营 ( 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 AWS 迁移策略中使用 AIOps 的更多信息，请参阅[运营集成指南](#)。

## 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 [IAM](#)) 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。

## blue/green 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，请参阅指南中的[“实施破碎玻璃程序”](#) AWS Well-Architected 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。

### 公民开发者

使用无code/low代码平台创建 AI 应用程序但没有专业技术技能的企业用户。

### 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 Cloud 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 Cloud 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，创建登录区、定义 CCoE、建立运营模型）
- 迁移 - 迁移单个应用程序
- Re-invention — 优化产品和服务，在云端进行创新

Stephen Orban 在 AWS Cloud 企业战略博客的博客文章 [《走向之旅 Cloud-First 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 管道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《AWS Well-Architected 框架》中安全支柱的组成部分。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 Cloud 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。

## 深度防御

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，深度防御方法可能将多因素身份验证、网络分段和加密结合起来。

## 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。此帐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 [《工作负载灾难恢复 AWS：AWS Well-Architected 框架中的云端恢复》](#)。

## 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。埃里克·埃文斯 (Eric Evans) 在他的《Domain-Driven 设计：解决软件核心的复杂性》(波士顿：Addison-Wesley 专业版，2003年)一书中介绍了这个概念。有关如何使用带有 strangler fig 模式的域驱动设计的信息，请参阅使用容器和 [Amazon API Gateway 逐步实现传统微软 ASP.NET \(ASMX\) 网络服务的现代化](#)。

## 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。Big-endian 系统首先存储最重要的字节。Little-endian 系统首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 Cloud，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 [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。这种技术是情境学习的应用，模型可以从提示中嵌入的示例 ( 镜头 ) 中学习。Few-shot 对于需要特定格式、推理或领域知识的任务，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，它已使用海量的通用和未标注数据集进行训练。FM 能够执行各种常规任务，例如理解语言、生成文本和图像以及使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](#)。

## FM 网关

一种集中式中介，用于控制和规范对[基础模型](#)的访问。也称为 LLM 网关。

# 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

一种高级规则，用于跨组织单位 (OU) 管理资源、策略和合规性。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。

## 护栏 (AI)

用于过滤、验证和限制[代理](#)输入和输出的安全机制，有助于确保负责任和安全的 AI 行为。

# 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。

## 人机在圈 (HitL)

一种工作流程模式，其中[代理](#)执行在关键决策点暂停以供人工审查和批准。

## 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

## laC

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 Cloud 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，请参阅框架中的[使用不可变基础架构部署](#)最佳实践。AWS Well-Architected

## 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](#)建议使用入站、出站和检查 VPC 设置网络账户，保护应用程序与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[克劳斯·施瓦布 \( Klaus Schwab \)](#)在2016年推出，指的是通过连接性、实时数据、自动化、分析和的进步实现制造流程的现代化。AI/ML

## 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 \( IIoT \) 数字化转型策略](#)。

## 检查 VPC

在 AWS 多账户架构中，一种集中式 VPC，用于管理 VPC ( 相同或不同 AWS 区域 )、互联网和本地网络之间的网络流量检查。[AWS 安全参考架构](#)建议使用入站、出站和检查 VPC 设置网络账户，保护应用程序与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[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 LLM](#)。

### 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、记录和控制将原材料转化为成品的生产过程。

## MAP

请参阅[迁移加速计划](#)。

## MCP

参见[模型上下文协议](#)。

### 模型上下文协议 ( MCP )

一种用于[代理](#)与[工具](#)通信的无状态协议。

## MCP 服务器

一种通过[模型上下文协议](#)公开一个或多个[工具](#)的服务。

## 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 AWS Well-Architected 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publish/subscribe模式的轻量级机器对机器 \(M2M\) 通信协议，适用于资源受限的物联网设备。](#)

## 微服务

一种小型独立服务，通过明确定义的 API 进行通信，通常由小型独立团队拥有。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 API 通过明确定义的接口进行通信。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

Cross-functional 通过自动化、敏捷的方法简化工作负载迁移的团队。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 DevOps 人员和冲刺专业人员。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 Cloud 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 Cloud 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 Cloud 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 Cloud 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，该 AWS Well-Architected 框架建议使用[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机器对机器 (M2M) 通信协议。OPC-UA 提供了数据加密、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 框架中的[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](#) 建议使用入站、出站和检查 VPC 设置网络账户，保护应用程序与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

私有托管区就是一个容器，其中包含的信息说明您希望 Amazon Route 53 如何响应一个或多个 VPC 中的某个域及其子域的 DNS 查询。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 Cloud 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 Cloud 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 AWS Organizations 的组织中所有账户的权限。SCP 为管理员可以委托给用户或角色的操作定义了防护机制或设定了限制。您可以将 SCP 用作允许列表或拒绝列表，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](#)。

## 暗影人工智能

在组织内受管控渠道之外构建或使用的未经授权的 [AI](#) 应用程序。

## 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 模式

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 Cloud 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 逐步实现传统微软 ASP.NET \(ASMX\) 网络服务的现代化](#)。

## 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。您可以使用 [Amazon S CloudWatch ynthetic](#) 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

Key-value 对充当用于组织 AWS 资源的元数据。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。

## 工具

[代理](#)可以调用以在外部系统中执行操作的函数或 API。

## 中转网关

中转网关是网络中转中心，您可用它来互连 VPC 和本地网络。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，该服务可以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

两个 VPC 之间的连接，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。