一种分阶段的性能工程方法 AWS 云

AWS 规范性指导

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS 规范性指导: 一种分阶段的性能工程方法 AWS 云

Copyright © 2025 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

简介 1
- 什么是性能工程？ 1
- 为什么要使用性能工程？ 1

性能工程支柱 2
- 生成测试数据 3
  - 测试数据生成工具 4
- 测试可观察性 5
  - 日志记录 6
  - 监控 9
  - 跟踪 12
- 测试自动化 14
  - 测试自动化工具 15
- 测试报告 16
  - 标准化录制 16

绩效支柱示例 17

资源 19

贡献者 21

文档历史记录 22

术语表 23
- # 23
- A 23
- B 26
- C 27
- D 30
- E 33
- F 35
- G 36
- H 37
- 我 38
- L 40
- M 41
- O 45
- P 47
- Q 49

R 50
S 52
T 55
U 56
V 57
W 57
Z 58
..... lix

一种分阶段的性能工程方法 AWS 云

亚马逊 Web Services（贡献者）

2024 年 4 月（文档历史记录）

本指南概述了为在 Amazon Web Services (AWS) 上运行的应用程序工作负载规划、构建和启用性能工程的最佳实践。它列出了性能工程的四个支柱，并提出了满足应用程序性能要求的不同方法。对于每个支柱，本指南都列出了用于设置性能测试和测试环境的工具和解决方案。

什么是性能工程？

性能工程包括在系统的开发生命周期中应用的技术，以确保满足非功能性能要求（例如吞吐量、延迟或内存使用量）。

在开始性能测试之前，您需要设置性能环境。典型的性能环境建立在以下支柱之上：

- 生成测试数据
- 测试可观察性
- 测试自动化
- 测试报告

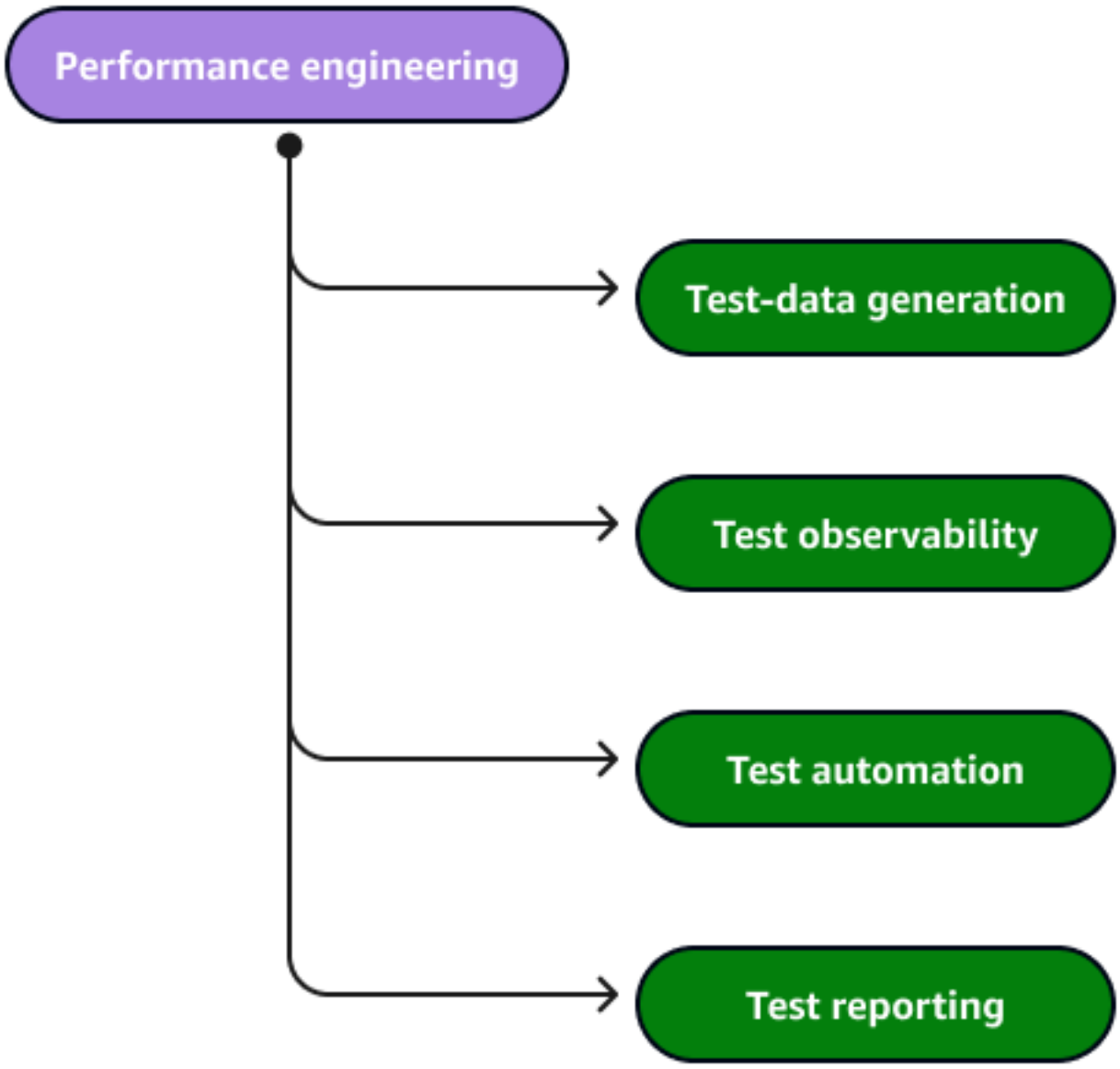
为什么要使用性能工程？

性能工程是指从设计阶段一开始就不断优化应用程序性能的过程。它避免了在开发周期的后期阶段对代码进行返工和重构，从而为业务带来了巨大的价值。在设计阶段开始性能工程会使应用程序性能更好，因为性能可以考虑在设计中。性能工程需要系统架构师 DevOps、开发人员和质量保证部门的积极参与。

性能工程的支柱

要实现性能工程思维，在为应用程序设置性能工程的同时，打下坚实的基础非常重要。性能工程需要建立四个主要支柱：

- 测试数据生成 — 性能工程师设置工具来生成测试数据。
- 测试可观察性 — 性能工程师设置可观测性环境，以确保可以记录和跟踪性能运行，并监控处理负载的资源。
- 测试自动化 — 性能工程师使用诸如 Apache JMet er 或 ghz 之类的工具开发自动化测试，以模拟用户流量和系统负载。
- 测试报告-收集有关每次测试运行的配置以及性能结果的数据。这些数据可以将配置更改与性能关联起来，并提供宝贵的见解。



整合这些支柱将鼓励从设计初始阶段开始的绩效思维。这将有助于避免在开发和测试的后期阶段对应用程序或环境进行更改。

生成测试数据

测试数据的生成涉及生成和维护用于运行性能测试用例的大量数据。生成的这些数据充当测试用例的输入，因此可以在不同的数据集上测试应用程序。

通常，生成测试数据是一个复杂的过程。但是，使用创建不当的数据集可能会导致生产环境中的应用程序行为不可预测。性能测试的测试数据生成不同于传统的测试数据生成方法。它需要真实场景，而且大

多数客户都希望使用与实际生产数据相似的数据来测试其工作负载。每次测试运行后，生成的测试数据通常还需要重置或刷新到其原始状态，这会增加时间和精力。

生成测试数据包括以下主要注意事项：

- 准确性-数据的准确性在测试的各个方面都很重要。不准确的数据会产生不准确的结果。例如，当生成信用卡交易时，它不应该是针对将来的某个日期。
- 有效性-数据应该对用例有效。例如，在测试信用卡交易时，不建议每位用户每天生成 10,000 笔交易，因为这与有效的用例场景大相径庭。
- 自动化 — 自动生成测试数据可以带来时间和精力的好处。它还可以实现有效的测试自动化。手动生成测试数据可能会对质量和时间要求产生影响。

  根据用例，可以采用不同的机制，如下所示：

  - API 驱动 — 在这种情况下，开发人员提供了测试数据生成 API，测试人员可以使用该API来生成数据。使用诸如 jMeter 之类的测试工具，测试人员可以使用商业 API 来扩展数据生成的规模。例如，如果您有用于添加用户的 API，则可以使用相同的 API 来创建数百个具有不同个人资料的用户。同样，您可以通过调用 delete API 操作来删除用户。对于复杂的工作流程应用程序，开发人员可以提供一个可以跨不同组件生成数据集的复合 API。使用这种方法，测试人员可以编写自动化程序，根据自己的要求生成和删除数据集。

    但是，如果系统很复杂或者每次调用的 API 响应时间很长，则设置和删除数据可能需要很长时间。
  - SQL 语句驱动 — 另一种方法是使用后端 SQL 语句生成大量数据。开发人员可以提供基于模板的 SQL 语句来生成测试数据。测试人员可以使用语句来填充数据，也可以在这些语句之上创建包装脚本来自动生成测试数据。使用这种方法，如果测试完成后需要重置数据，测试人员可以非常快速地填充和删除数据。但是，这种方法需要直接访问应用程序的数据库，而这在典型的安全环境中可能无法实现。此外，无效的查询可能会导致数据填充不正确，从而产生偏斜的结果。开发人员还必须不断更新应用程序代码中的 SQL 语句，以反映随着时间的推移对应用程序所做的更改。

测试数据生成工具

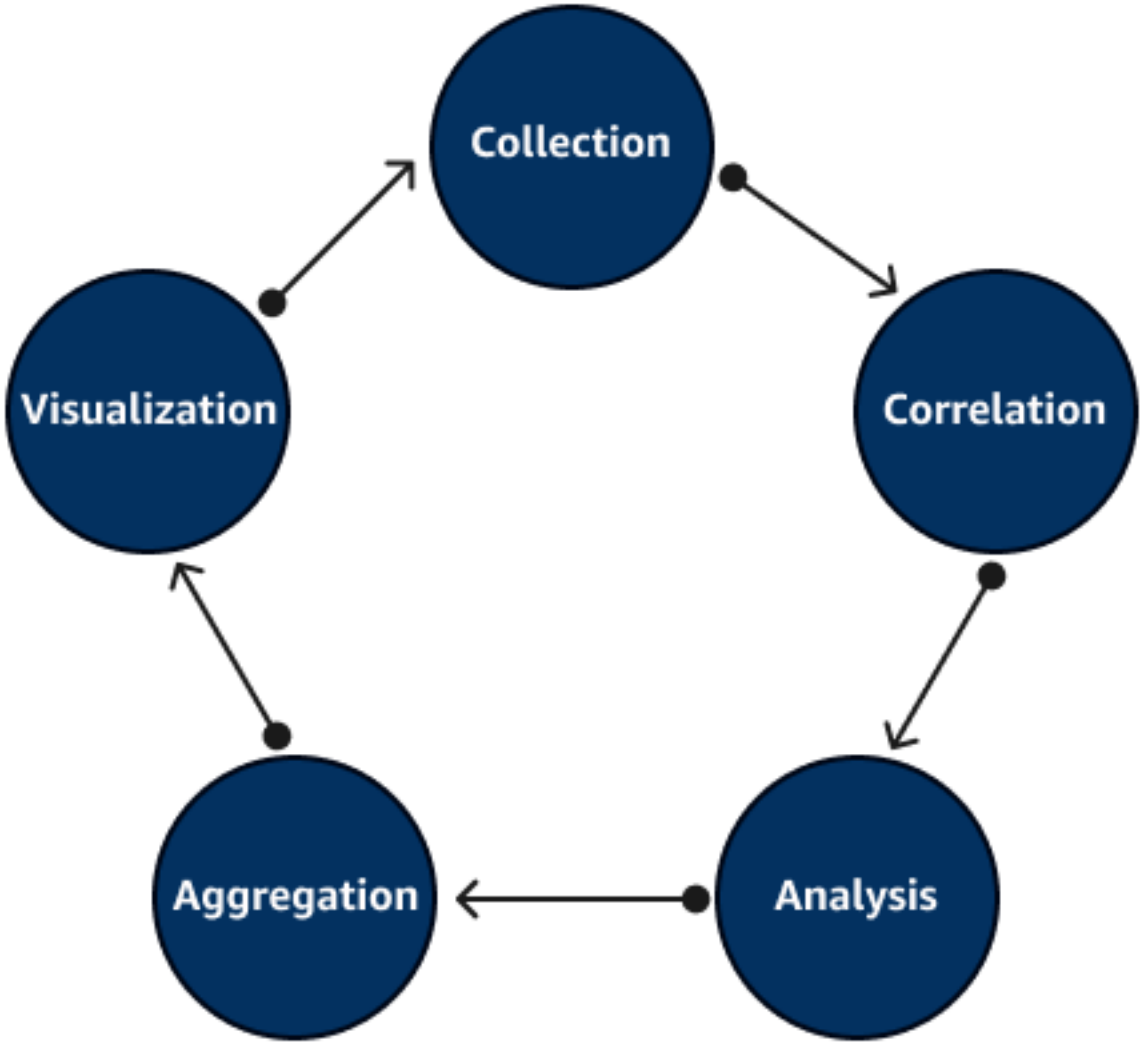
AWS 提供了可用于生成测试数据的原生自定义工具：

- 亚马逊 Kinesis 数据生成器 — 亚马逊 Kinesis 数据生成器 (KDG) 简化了生成数据并将其发送到亚马逊 Kinesis 的任务。该工具提供了一个用户友好的用户界面，可直接在您的浏览器中运行。有关更多信息和参考实现，请参阅 “使用全新 Amazon Kinesis 数据生成器测试您的流数据解决方案” 博客文章。

- AWS Glue 测试数据生成器- AWS Glue 测试数据生成器提供了一个可配置的框架，用于使用 AWS Glue PySpark 无服务器作业生成测试数据。所需的测试数据描述可通过 YAML 配置文件进行完全配置。有关更多信息和参考实现，请参阅AWS Glue 测试数据生成器 GitHub 存储库。

测试可观察性

测试可观测性支持在性能测试运行期间收集、关联、聚合和分析网络、基础架构和应用程序中的遥测数据。您可以全面了解系统的行为、性能和运行状况。这些见解可帮助您更快地检测、调查和修复问题。通过添加人工智能和机器学习，您可以主动做出反应、预测和预防问题。



可观察性依赖于记录、监控和跟踪。成功实施这些活动的责任跨越了应用程序和基础架构团队。

在设计阶段开始时，应用团队应了解其可观测性堆栈的当前状态，包括日志、监控和跟踪。然后，他们可以选择更顺畅地集成到可观测性堆栈中的工具。

同样，基础设施团队负责管理和扩展可观测性基础架构。

在测试可观测性方面，请考虑以下方面：

- 应用程序日志和跟踪的可用性
- 日志和跟踪的关联
- 节点、容器和应用程序指标的可用性
- 按需自动设置和更新可观测性基础架构
- 能够对遥测进行可视化
- 可观测性基础设施的扩展

日志记录

记录是保存有关系统中发生的事件数据的过程。日志可以包括问题、错误或有关当前操作的信息。日志可以分为不同的类型，例如：

- 事件日志
- 服务器日志
- 系统日志
- 授权和访问日志
- 审核日志

开发者可以在日志中搜索特定的错误代码或模式，根据特定字段对其进行筛选，或者将其安全地存档以备将来分析。日志可以帮助开发人员对性能问题进行根本原因分析，并在系统组件之间进行关联。

构建有效的日志解决方案需要应用程序和基础架构团队之间的密切协调。除非有可扩展的日志基础架构来支持日志的解析、筛选、缓冲和关联等用例，否则应用程序日志将毫无用处。可以简化常见用例，例如生成关联 ID、记录关键业务方法的运行时间以及定义日志模式。

应用程序小组

应用程序开发人员必须确保生成的日志遵循日志记录最佳实践。最佳做法包括以下内容：

- 生成关联 ID 以跟踪唯一请求
- 记录关键业务方法所花费的时间

- 在适当的日志级别上进行日志记录
- 共享通用日志库

在设计与不同微服务交互的应用程序时，请使用这些日志设计原则来简化后端的筛选和日志提取。

生成关联 ID 以跟踪唯一请求

当应用程序收到请求时，它可以检查标头中是否已存在关联 ID。如果没有 ID，则应用程序应生成一个 ID。例如，Application Load Balancer 会添加一个名为的标头`X-Amzn-Trace-Id`。应用程序可以使用标头将来自负载均衡器的请求与应用程序关联起来。同样，`traceId`如果调用依赖的微服务，则应用程序应注入，以便请求流中不同组件生成的日志相互关联。

记录关键业务方法所花费的时间

当应用程序收到请求时，它会与其他组件进行交互。应用程序应以定义的模式记录关键业务方法所花费的时间。这样可以更轻松地在后端解析日志。它还可以帮助您从日志中生成有用的见解。您可以使用诸如面向方面的编程 (AOP) 之类的方法来生成此类日志，以便将日志问题与业务逻辑分开。

在适当的日志级别上进行日志记录

应用程序应写入包含大量有用信息的日志。使用日志级别按事件的严重性对事件进行分类。例如，对于需要调查的重要事件，使用`WARNING`和`ERROR`关卡。使用`INFO`和`DEBUG`可进行详细的跟踪和高容量事件。将日志处理程序设置为仅捕获生产中必需的级别。在该`INFO`级别生成过多的日志记录无济于事，它会增加后端基础架构的压力。 `DEBUG`日志可能很有用，但应谨慎使用。使用`DEBUG`日志会生成大量数据，因此不建议在性能测试环境中使用日志。

共享通用日志库

应用程序团队应使用通用的日志库，例如 适用于 Java 的 AWS SDK，带有预定义的通用日志记录模式，开发人员可以在其项目中将其用作依赖项。

基础架构小组

DevOps 在后端筛选和提取日志时，工程师可以使用以下日志设计原则来减少工作量。基础架构团队必须设置和支持以下资源。

日志代理

日志代理（日志采集器）是一个从一个位置读取日志并将其发送到另一个位置的程序。日志代理用于读取存储在计算机上的日志文件，并将日志事件上传到后端进行集中化。

日志是非结构化数据，必须先对其进行结构化处理，然后才能从中获得有意义的见解。日志代理使用解析器来读取日志语句并提取相关字段，例如时间戳、日志级别和服务名称，并将这些数据结构化为 JSON 格式。在边缘安装轻量级日志代理很有用，因为它可以降低资源利用率。日志代理可以直接推送到后端，也可以使用中间日志转发器将数据推送到后端。使用日志转发器可以将工作从源头的日志代理中移开。

日志解析器

日志解析器将非结构化日志转换为结构化日志。日志代理解析器还可以通过添加元数据来丰富日志。数据解析可以在源端（应用程序端）完成，也可以集中完成。存储日志的架构应该是可扩展的，以便您可以添加新字段。我们建议使用标准日志格式，例如 JSON。但是，在某些情况下，必须将日志转换为 JSON 格式才能更好地进行搜索。编写正确的解析器表达式可以实现高效的转换。

日志后端

日志后端服务收集、提取和可视化来自各种来源的日志数据。日志代理可以直接写入后端或使用中间日志转发器。在进行性能测试时，请务必存储日志，以便日后可以对其进行搜索。将每个应用程序的日志分别存储在后端。例如，为应用程序使用专用索引，并使用索引模式搜索分布在不同相关应用程序中的日志。我们建议至少保存 7 天的日志搜索数据。但是，将数据存储更长的时间可能会导致不必要的存储成本。由于在性能测试期间会生成大量日志，因此日志基础架构必须扩展和调整日志后端的大小。

日志可视化

要从应用程序日志中获得有意义且可操作的见解，请使用专用的可视化工具处理原始日志数据并将其转换为图形表示。图表、图表和仪表板等可视化可以帮助发现趋势、模式和异常，这些趋势和异常在查看原始日志时可能不太明显。

使用可视化工具的主要好处包括能够关联多个系统和应用程序中的数据，以识别依赖关系和瓶颈。交互式仪表板支持以不同的粒度深入研究数据，以解决问题或发现使用趋势。专业的数据可视化平台提供分析、警报和数据共享等功能，可以增强监控和分析。

通过在应用程序日志上使用数据可视化的强大功能，开发和运营团队可以深入了解系统和应用程序的性能。得出的见解可用于多种用途，包括优化效率、改善用户体验、增强安全性和容量规划。最终结果是为各种利益相关者量身定制的仪表板，提供将日志数据汇总为可操作且有见地的信息的 at-a-glance 视图。

自动化日志基础架构

由于不同的应用程序有不同的要求，因此自动化日志基础架构的安装和操作非常重要。使用基础设施即代码 (IaC) 工具来配置日志基础架构的后端。然后，您可以将日志基础架构配置为共享服务，也可以作为特定应用程序的独立定制部署。

我们建议开发人员使用持续交付 (CD) 管道来自动执行以下操作：

- 按需部署日志基础架构，并在不需要时将其拆除。
- 在不同的目标上部署日志代理。
- 部署日志解析器和转发器配置。
- 部署应用程序仪表板。

日志工具

AWS 提供原生日志、警报和仪表板服务。以下是常 AWS 服务 用的日志记录资源：

- Amazon S OpenSearch ervice 可帮助组织收集、摄取和可视化来自各种来源的日志数据。有关更多信息，请参阅使用进行集中日志记录 OpenSearch。
- 亚马逊 CloudWatch 代理和 F AWS or Fluent Bit 是上最受欢迎的日志代理 AWS。有关在 Amazon L CloudWatch ogs Insights 中使用 CloudWatch 代理的信息，请参阅博客文章使用 Amazon L ogs Insights 简化 Apache 服务器 CloudWatch 日志。 AWS 有关 Fluent Bit 的参考实现，请参阅博客文章使用 Fl uent Bit 进行集中式容器日志记录。

监控

监控是收集不同指标（例如 CPU 和内存）并将其存储在时间序列数据库（例如适用于 Prometheus 的 Amazon 托管服务）中的过程。监控系统可以是基于推送的，也可以是基于拉动的。在基于推送的系统中，源会定期将指标推送到时间序列数据库。在基于拉取的系统中，抓取器从各种来源抓取指标并将其存储在时间序列数据库中。开发人员可以分析指标，筛选指标，并随时间推移绘制这些指标，以实现性能可视化。成功实施监控可以分为两大领域：应用程序和基础架构。

对于应用程序开发人员来说，以下指标至关重要：

- 延迟-收到响应所花费的时间
- 请求吞吐量-每秒处理的请求总数
- 请求错误率-错误总数

捕获业务交易中涉及的每种资源（例如应用程序容器、数据库）的资源利用率、饱和度和错误计数。例如，在监控 CPU 使用率时，您可以跟踪性能测试运行期间的平均 CPU 利用率、平均负载和峰值负载。当资源在 stress testing 期间达到饱和，但在性能运行期间可能在较短的时间内未达到饱和状态时。

指标

应用程序可以使用不同的执行器（例如弹簧启动执行器）来监控其应用程序。这些生产级库通常会公开 REST 端点，用于监控有关正在运行的应用程序的信息。这些库可以监控底层基础架构、应用程序平台和其他资源。如果任何默认指标不符合要求，则开发者必须实现自定义指标。自定义指标可以帮助跟踪无法通过默认实现的数据进行跟踪的业务关键绩效指标 (KPI)。例如，您可能想要跟踪业务操作，例如第三方 API 集成延迟或已完成的交易总数。

基数

基数是指指标的唯一时间序列的数量。标记指标是为了提供更多信息。例如，跟踪特定 API 请求计数的基于 REST 的应用程序表示基数为 1。如果您添加用户标签来标识每个用户的请求数，则基数会随着用户数量的增加而成比例增加。通过添加创建基数的标签，您可以按不同的组对指标进行切片和切块。为正确的用例使用正确的标签很重要，因为基数会增加后端监控时间序列数据库中指标序列的数量。

解决方案

在典型的监控设置中，监控应用程序被配置为定期从应用程序中获取指标。抓取的周期性定义了监控数据的粒度。由于可用的数据点更多，因此以较短的时间间隔收集的指标往往可以更准确地了解性能。但是，存储的条目越多，时间序列数据库的负载就会增加。通常，60 秒的粒度是标准分辨率，1 秒是高分辨率。

DevOps 球队

应用程序开发人员经常要求 DevOps 工程师设置监控环境，以可视化基础架构和应用程序的指标。DevOps 工程师必须设置一个可扩展的环境，并支持应用程序开发人员使用的数据可视化工具。这包括从不同来源抓取监控数据，然后将数据发送到中央时间序列数据库，例如适用于 Prometheus 的 Amazon 托管服务。

监控后端

监控后端服务支持指标数据的收集、存储、查询和可视化。它通常是一个时间序列数据库，例如适用于 Prometheus 的亚马逊托管服务或 InfluxDB。 InfluxData 使用服务发现机制，监控收集器可以从不同的来源收集指标并将其存储。在进行性能测试时，存储指标数据很重要，以便日后可以对其进行搜索。我们建议至少保存 15 天的指标数据。但是，将指标存储更长的时间并不会带来显著的好处，并且会导致不必要的存储成本。由于性能测试可以生成大量指标，因此在提供快速查询性能的同时扩展指标基础架构非常重要。监控后端服务提供了一种查询语言，可用于查看指标数据。

可视化

提供可视化工具，这些工具可以显示应用程序数据，从而提供有意义的见解。 DevOps 工程师和应用程序开发人员应学习监控后端的查询语言，并密切合作生成可重复使用的仪表板模板。在仪表板上，包括延迟和错误，同时还显示基础架构和应用程序资源的资源利用率和饱和度。

实现监控基础架构的自动化

与日志记录类似，自动安装和运行监控基础架构非常重要，这样您才能适应不同应用程序的不同要求。使用 IaC 工具配置监控基础设施的后端。然后，您可以将监控基础设施配置为共享服务，也可以作为特定应用程序的独立定制部署。

使用 CD 管道自动执行以下操作：

- 按需部署监控基础架构，并在不需要时将其拆除。
- 更新监控配置以筛选或聚合指标。
- 部署应用程序仪表板。

监控工具

适用于 Prometheus 的 Amazon 托管服务是一项与 Prometheus 兼容的监控服务，用于监控容器基础设施和容器的应用程序指标，您可以使用它来安全地大规模监控容器环境。有关更多信息，请参阅博客文章 Prometheus 亚马逊托管服务入门。

Amazon 在上 CloudWatch 提供全栈监控。 AWS CloudWatch 支持 AWS 原生和开源解决方案，因此您可以随时了解技术堆栈中正在发生的事情。

原生 AWS 工具包括以下内容：

- 亚马逊 CloudWatch 控制面板
- CloudWatch Container Insights
- CloudWatch 指标
- CloudWatch 警报

Amazon CloudWatch 提供专门构建的功能，可解决特定用例，例如通过 Container Insights 进行容 CloudWatch 器监控。这些功能是内置的， CloudWatch 因此您可以设置日志、指标收集和监控。

对于您的容器化应用程序和微服务，请使用 Container Insights 收集、汇总和汇总指标和日志。Container Insights 适用于亚马逊弹性容器服务 (Amazon ECS)、亚马逊弹性 Kubernetes Service

(Amazon EKS) 和亚马逊弹性计算云 (Amazon EC2) 上的 Kubernetes 平台。Container Insights 以嵌入式指标格式将数据作为性能日志事件收集。这些性能日志事件条目使用结构化的 JSON 架构，该架构支持大规模的高基数数据摄取和存储。

有关使用 Amazon EKS 实现容器见解的信息，请参阅博客文章使用 AWS Distro 为亚马逊 EKS Fargate 介绍亚马逊 CloudWatch 容器见解。 OpenTelemetry

跟踪

跟踪涉及专门使用有关程序进程的日志信息。从日志中获得的见解可以帮助工程师调试单个交易并识别瓶颈。跟踪可以自动启用，也可以使用手动检测来启用。

由于应用程序与不同的服务集成，因此确定应用程序及其底层服务的性能非常重要。追踪适用于痕迹和跨度。跟踪是完整的请求过程，每条跟踪都由 span 组成。跨度是一个标记的时间间隔，是系统各个组件或服务中的活动。Traces 提供了向应用程序发出请求时会发生什么情况的全局。

应用程序小组

应用程序开发人员通过发送入站和出站请求以及应用程序内其他事件的跟踪数据以及有关每个请求的元数据来检测其应用程序。要生成跟踪，必须对应用程序进行检测以生成跟踪。仪器可以是自动的，也可以是手动的。

自动仪器

您可以使用自动检测从应用程序收集遥测数据，而无需修改源代码。自动检测代理可以生成应用程序或服务的应用程序跟踪。通常，您可以使用配置更改来添加代理或其他机制。

库插桩包括对应用程序代码进行最少的更改，以添加预先构建的工具。这些工具针对特定的库或框架，例如 AWS SDK、Apache HTTP 客户端或 SQL 客户端。

手动检测

在这种方法中，应用程序开发人员在要收集跟踪信息的每个位置向应用程序添加检测代码。例如，使用面向方面的编程 (AOP) 来收集跟踪数据。 AWS X-Ray 开发人员可以使用 SDK 来检测其应用程序。

采样

追踪数据通常会大量生成。重要的是要有一种机制来确定是否应该导出跟踪数据。采样是确定应导出哪些数据的过程。这样做通常是为了节省成本。您可以通过自定义采样规则来控制记录的数据量。您也可以在不更改和重新部署代码的情况下更改采样行为。控制采样速率以生成适量的迹线非常重要。

应用程序开发人员可以通过将元数据添加为键值对来对跟踪进行注释。这些注解丰富了跟踪记录，并有助于完善后端的筛选。

DevOps 球队

DevOps 工程师经常被要求为应用程序开发人员设置跟踪环境，以可视化基础设施和应用程序的跟踪。跟踪环境设置包括从不同来源收集跟踪数据，然后将其发送到中央存储进行可视化。

追踪后端

跟踪后端是诸如此类的服务 AWS X-Ray ，它收集有关您的应用程序所处理的请求的数据。它提供了一些工具，您可以使用这些工具来查看、筛选和深入了解这些数据，从而识别问题和优化机会。对于对您的应用程序的任何跟踪请求，您可以查看有关请求和响应的详细信息，以及您的应用程序对下游 AWS 资源、微服务、数据库和 Web API 发出的其他调用的详细信息。

自动跟踪

由于不同的应用程序有不同的跟踪要求，因此自动化跟踪基础架构的配置和操作非常重要。使用 IaC 工具配置跟踪基础设施的后端。

使用 CD 管道自动执行以下操作：

- 按需部署跟踪基础架构，并在不需要时将其拆除。
- 跨应用程序部署跟踪配置。

追踪工具

AWS 为跟踪及其关联的可视化提供以下服务：

- AWS X-Ray 除了从您的应用程序使用的已与 X-Ray 集成的 AWS 服务中接收跟踪之外，还会接收来自您的应用程序的跟踪。可以使用多种开发工具包、代理和工具来检测应用程序，实现 X-Ray 跟踪。有关更多信息，请参阅 AWS X-Ray 文档。

  开发人员还可以使用 AWS X-Ray 软件开发工具包向 X-Ray 发送跟踪。 AWS X-Ray 提供适用于 Go、、Node.js、JavaPython、.NET 和Ruby的软件开发工具包。每个 X-Ray 开发工具包都提供以下内容：
  - 拦截器，可添加到您的代码中以跟踪传入 HTTP 请求
  - 用于检测您的应用程序用来调用其他 AWS 服务的 AWS SDK 客户端的客户端处理程序
  - HTTP 客户端，用于检测对其他内部和外部 HTTP Web 服务的调用

X-Ray SDK 还支持对 SQL 数据库进行检测调用、自动 AWS SDK 客户端检测以及其他功能。该开发工具包不是直接将跟踪数据发送到 X-Ray;，而是将 JSON 分段文档发送到侦听 UDP 流量的进程守护程序进程。X-Ray 进程守护程序将分段缓冲在队列中，并将分段批量上传到 X-Ray。有关使用 X-Ray SDK 对应用程序进行检测的更多信息，请参阅 X-Ra y 文档。

- Amazon OpenSearch Service 是一项用于运行和扩展 OpenSearch 集群的 AWS 托管服务，可用于集中存储日志、指标和跟踪。可观察性插件为采集和监控来自常见数据源的指标、日志和跟踪信息提供了统一的体验。将数据收集和监控集中在一处，可提供整个基础架构的全栈 end-to-end 可观察性。有关实现信息，请参阅OpenSearch 服务文档。
- AWS Distro for OpenTelemetry (ADOT) 是一个基于云原生计算基金会 (CNCF) 项目的 AWS 发行版。 OpenTelemetry ADOT 目前包括对 Java 和 Python 的自动检测支持。此外，ADOT 支持使用 Node.js 自动检测 AWS Lambda 函数及其下游请求Java，并通过 ADOT 托管 Lambda 层Python运行时自动检测函数及其下游请求。开发人员可以使用 ADOT 收集器将跟踪发送到不同的后端，包括和 AWS X-Ray Ama OpenSearch zon 服务。

  有关如何使用 ADOT SDK 对应用程序进行检测的参考示例，请参阅文档。有关如何使用 ADOT SDK 向亚马逊 OpenSearch 服务发送数据的参考示例，请参阅OpenSearch 服务文档。

  有关如何检测在 Amazon EKS 上运行的应用程序的参考示例，请参阅博客文章 “使用适用于 AWS Distro 的 Amazon EKS 插件收集指标和跟踪”。 OpenTelemetry

测试自动化

使用专门的框架和工具进行自动测试可以减少人为干预并最大限度地提高质量。自动性能测试与自动化测试（例如单元测试和集成测试）没有什么不同。

在不同阶段使用 DevOps 管道进行性能测试。

测试自动化管道的五个阶段是：

1. 设置-在此阶段使用测试数据生成部分中描述的测试数据方法。生成真实的测试数据对于获得有效的测试结果至关重要。您必须谨慎创建涵盖各种用例并与实时制作数据密切匹配的多样化测试数据。在运行全面性能测试之前，您可能需要运行初始试用测试来验证测试脚本、环境和监控工具。
2. 测试工具-要进行性能测试，请选择适当的负载测试工具，例如 jMeter 或 ghz。在模拟真实用户负载方面，请考虑最适合您的业务需求的方案。

3. 测试运行-在建立测试工具和环境后，在一系列预期的用户负载和持续时间内运行 end-to-end 性能测试。在整个测试过程中，密切监视被测系统的运行状况。这通常是一个长期运行的阶段。监控自动测试失效的错误率，如果错误太多，则停止测试。

   负载测试工具可提供对资源利用率、响应时间和潜在瓶颈的见解。
4. 测试报告-收集测试结果以及应用程序和测试配置。自动收集应用程序配置、测试配置和结果，这有助于记录与性能测试相关的数据并将其集中存储。集中维护绩效数据有助于提供良好的见解，并支持以编程方式为您的业务定义成功标准。
5. 清理-完成性能测试运行后，重置测试环境和数据，为后续运行做好准备。首先，恢复运行期间对测试数据所做的任何更改。必须将数据库和其他数据存储恢复到其原始状态，还原测试期间生成的任何新、更新或删除的记录。

您可以重复使用管道多次重复测试，直到结果反映出您想要的性能。您还可以使用管道来验证代码更改不会影响性能。您可以在非工作时间运行代码验证测试，并使用可用的测试和可观察性数据进行故障排除。

最佳做法包括以下内容：

- 记录开始和结束时间，并自动生成用于记录的 URL，这有助于您在相应的时间窗口、监控和跟踪系统中筛选可观察性数据。
- 调用测试时，在标题中注入测试标识符。应用程序开发人员可以通过在后端使用标识符作为过滤器来丰富其日志、监控和跟踪数据。
- 将管道限制为一次只能运行一次。运行并发测试会产生噪音，从而在故障排除期间造成混乱。在专用的性能环境中运行测试也很重要。

测试自动化工具

测试工具在任何测试自动化中都起着重要作用。开源测试工具的热门选择包括以下几种：

- Apache JMeter 是一款强大的工具。多年来，Apache JMeter 变得越来越可靠，并增加了一些功能。利用图形界面，您无需掌握编程语言即可创建复杂的测试。诸如此类的公司 BlazeMeter支持 Apache jMeter。
- K6 是一款免费工具，提供支持、负载源托管以及用于组织、运行和分析负载测试的集成 Web 界面。
- Vegeta 负载测试遵循不同的概念。您无需定义并发性或向系统施加负载，而是定义一定的速率。然后，该工具会创建独立于系统响应时间的负载。

- He@@ y and ab，Apache HTTP 服务器基准测试工具，是基本工具，你可以从命令行使用它们在单个端点上运行指定的负载。如果您有服务器可以运行这些工具，这是生成负载最快的方法。即使是本地笔记本电脑也能运行，尽管它可能不够强大，无法产生高负载。
- g@@ hz 是一个命令行实用程序和 G o 包，用于负载测试和基准测试 gRPC 服务。

AWS 提供 AWS 解决方案的分布式负载测试。该解决方案可以创建和模拟成千上万的连接用户，无需配置服务器即可以恒定的速度生成交易记录。有关更多信息，请参阅AWS 解决方案库。

您可以使用自动 AWS CodePipeline 执行性能测试管道。有关使用自动执行 API 测试的更多信息 CodePipeline，请参阅AWS DevOps 博客和AWS 文档。

测试报告

测试报告是指收集、分析和呈现与系统、应用程序、服务或流程的性能相关的数据。它涉及测量各种指标和指标，以评估特定系统或组件的效率、响应能力、可靠性和整体有效性。

绩效测试报告涉及根据分析的背景和目标选择相关指标。常见的性能指标包括响应时间、吞吐量、错误率、资源利用率（CPU、内存、磁盘）和网络延迟。

收集完与性能相关的数据后，需要将其存储在中央存储库中。这些测试结果可能来自不同的环境、应用程序和测试工具。当您在不同的环境中运行多个工作负载时，很难收集与性能相关的数据并在这些数据点之间进行关联以得出明智的结论。我们建议定义一种标准方法，使用用于数据存储和可视化的中央存储库来收集性能指标数据。

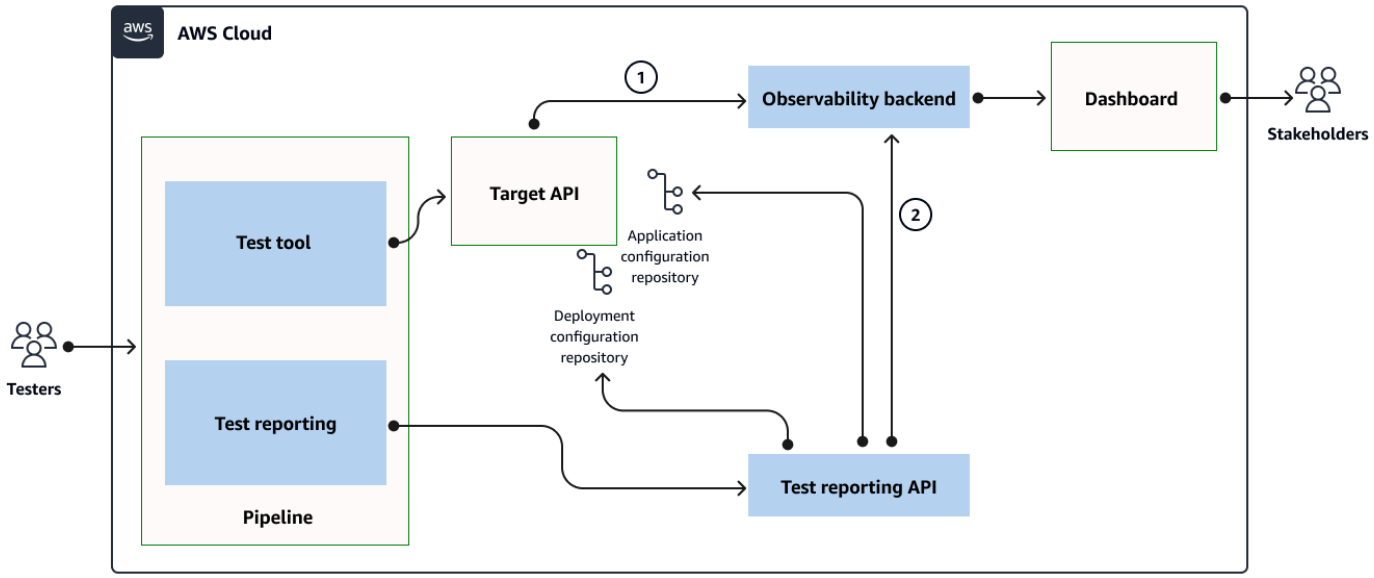
标准化录制

我们建议标准化不同利益相关者执行性能测试并将生成的数据写入中央存储库的方式。例如，这可以采用 API 的形式，接受结果并将其存储到持久存储解决方案中。在需要从诸如 GitOps 适用于 Prometheus 的 Amazon 托管服务等来源获取数据的情况下，API 可以根据描述如何从部署规范和 Kubernetes 规范中提取字段的架构文件直接从指定来源提取这些详细信息。架构文件可以使用JSONPath表达式或 Prometheus 查询语言 (PromQL)。如前所述，收集的指标应与绩效分析的背景和目标相关。

传递给 API 的数据可以包括与应用程序和已执行测试的环境相关的详细信息和标签。这有助于对性能测试数据进行分析。

性能工程支柱在行动

以下参考架构演示了用于测试特定 API 的性能工程支柱。



1. 日志、监控和跟踪数据从目标 API 发送到后端。
2. 调用时，测试报告 API 会将结果和配置信息发送到后端。

核心组件是被测的目标 API 或应用程序。目标 API 以某种 GitOps方式与应用程序配置存储库和部署配置存储库同步，以获取最新的应用程序和基础架构配置。通过这种同步，可以根据 Git 存储库中定义的应用程序及其支持基础设施的当前所需状态运行自动测试。

测试自动化管道可自动生成测试数据、运行测试和报告目标 API 的测试结果。

目标 API 使用可观测性最佳实践生成性能见解（指标、日志和跟踪），并将指标数据流式传输到可观察性后端。

测试报告 API 收集所有与测试相关的报告数据（配置和测试结果），并将其存储在可观察性后端。

性能见解和报告数据（配置、测试结果）的聚合可帮助您查询目标 API 的性能相关数据。例如，你可能会问以下问题：

- 最慢的十大交易有哪些？
- 每项测试的 P99、P90 平均数是多少？
- 两次测试运行的配置如何比较？

将测试用例与一段时间内的结果、配置和指标关联起来有助于确定最佳配置和性能结果。

使用这些测试结果，您可以为 API 做出更精确的、以数据为导向的决策，并在将 API 投入生产时充满信心。

资源

Amazon Web Services

- Amazon CloudWatch
- AWS CodePipeline
- AWS 发行版适用于 OpenTelemetry
- 亚马逊 OpenSearch 服务
- AWS X-Ray

实现

- amazon-kinesis-data-generator
- AWS Glue 测试数据生成器
- 分布式负载测试已开启 AWS

博客文章

- 使用 Fluent Bit 进行集中式容
- 使用全新 Amazon Kinesis 数据生成器测试您的流数据解决方案
- 为亚马逊 EKS Fargate 介绍使用 AWS Distro 的 Amazon Contain CloudWatch er Insights OpenTelemetry
- 使用 Kubernetes 上的应用程序跟踪 AWS X-Ray
- 使用适用于 AWS Distro 的 Amazon EKS 插件收集指标和跟踪 OpenTelemetry
- Prometheus 亚马逊托管服务入门

工作坊

- AWS 可观测性简介

AWS 规范性指导

- 负载测试应用程序（指南）

第三方应用程序

- Apache JMeter
- K6
- Vegeta
- 嘿还有 a b
- 千兆赫

贡献者

本文档的贡献者包括：

- Varun Sharma，高级首席顾问， AWS
- Akash Kumar，高级首席顾问， AWS
- Archana Bhatnagar，业务经理， AWS
- Pratik Sharma，《专业服务 II》， AWS

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	—	2024 年 4 月 24 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：将您的本地 Oracle 数据库迁移到兼容 Amazon Aurora PostgreSQL 的版本。
- 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：在中将您的本地 Oracle 数据库迁移到适用于 Oracle 的亚马逊关系数据库服务 (Amazon RDS) AWS 云。
- 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。
- 更换主机（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：在中的 EC2 实例上将您的本地 Oracle 数据库迁移到 Oracle AWS 云。
- 重新定位（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。您可以将服务器从本地平台迁移到同一平台的云服务。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

参见托管服务。

ACID

参见原子性、一致性、隔离性、耐久性。

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。与主动-被动迁移相比，它更灵活，但需要更多的工作。

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

一个 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

参见人工智能。

AIOps

参见人工智能操作。

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

坏机器人

旨在破坏个人或组织或对其造成伤害的机器人。

BCP

参见业务连续性计划。

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。另请参见字节顺序。

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的应用程序版本（蓝色），在另一个环境中运行新的应用程序版本（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互动的软件应用程序。有些机器人是有用或有益的，例如在互联网上索引信息的网络爬虫。其他一些被称为恶意机器人的机器人旨在破坏个人或组织或对其造成伤害。

僵尸网络

被恶意软件感染并受单方（称为机器人牧民或机器人操作员）控制的机器人网络。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）。

破碎的玻璃通道

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 Well -Architected 指南中的 “实施破碎玻璃程序” 指示 AWS 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

参见AWS 云采用框架。

金丝雀部署

向最终用户缓慢而渐进地发布版本。当你有信心时，你可以部署新版本并全部替换当前版本。

CCoE

参见云卓越中心。

CDC

请参阅变更数据捕获。

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弹性。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

查看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。云计算通常与边缘计算技术相关。

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

组织迁移到以下阶段时通常会经历四个阶段 AWS 云：

- 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

参见配置管理数据库。

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)

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

配置偏差

对于工作负载，配置会从预期状态发生变化。这可能会导致工作负载变得不合规，而且通常是渐进的，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 is commonly described as a pipeline. 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

参见计算机视觉。

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

参见数据库定义语言。

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。

委托管理员

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。

后

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

参见环境。

侦测性控制

一种安全控制，在事件发生后进行检测、记录日志和发出警报。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

在星型架构中，一种较小的表，其中包含事实表中有关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)

您用来最大限度地减少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

参见数据库操作语言。

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

参见灾难恢复。

漂移检测

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

参见开发价值流映射。

E

EDA

参见探索性数据分析。

EDI

参见电子数据交换。

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。与云计算相比，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)

组织之间自动交换业务文档。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密文的计算过程。

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

参见服务端点。

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

参见企业资源规划。

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

星形架构中的中心表。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。

失败得很快

一种使用频繁和增量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

参见分支。

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。

少量提示

在要求法学硕士执行类似任务之前，向其提供少量示例，以演示该任务和所需的输出。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。对于需要特定格式、推理或领域知识的任务，Few-shot 提示可能非常有效。另请参见零镜头提示。

FGAC

请参阅精细的访问控制。

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，它使用连续的数据复制，通过更改数据捕获在尽可能短的时间内迁移数据，而不是使用分阶段的方法。目标是将停机时间降至最低。

FM

参见基础模型。

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

人工智能模型的子集，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工件，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 AI。

地理封锁

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，而基于主干的工作流程是现代的首选方法。

金色影像

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配置软件，并有助于提高设备制造运营的速度、可扩展性和生产力。

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

一项高级规则，可帮助管理各组织单位的资源、策略和合规性 (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。侦测性防护机制会检测策略违规和合规性问题，并生成警报以进行修复。它们通过使用 AWS Config、、Amazon、 AWS Security Hub GuardDuty AWS Trusted Advisor、Amazon Inspector 和自定义 AWS Lambda 支票来实现。

H

HA

参见高可用性。

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。

抵制数据

从用于训练机器学习模型的数据集中扣留的一部分带有标签的历史数据。通过将模型预测与抵制数据进行比较，您可以使用抵制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

参见基础设施即代码。

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

参见工业物联网。

不可变的基础架构

一种为生产工作负载部署新基础架构，而不是更新、修补或修改现有基础架构的模型。不可变基础架构本质上比可变基础架构更一致、更可靠、更可预测。有关更多信息，请参阅 Well-Architected Framework 中的使用不可变基础架构 AWS 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

该术语由克劳斯·施瓦布（Klaus Schwab）于2016年推出，指的是通过连接、实时数据、自动化、分析和人工智能/机器学习的进步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置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。

IoT

参见物联网。

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。

大型语言模型 (LLM)

一种基于大量数据进行预训练的深度学习 AI 模型。法学硕士可以执行多项任务，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

见 7 R。

小端序系统

一个先存储最低有效字节的系统。另请参见字节顺序。

LLM

参见大型语言模型。

下层环境

参见环境。

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

参见分支。

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。恶意软件的示例包括病毒、蠕虫、勒索软件、特洛伊木马、间谍软件和键盘记录器。

托管服务

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。亚马逊简单存储服务 (Amazon S3) Service 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

参见迁移加速计划。

机制

一个完整的过程，在此过程中，您可以创建工具，推动工具的采用，然后检查结果以进行调整。机制是一种在运行过程中自我增强和改进的循环。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

参见制造执行系统。

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 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： EC2 使用 AWS 应用程序迁移服务重新托管向 Amazon 的迁移。

迁移组合评测（MPA）

一种在线工具，可提供信息，用于验证迁移到的业务案例。 AWS 云 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

用于将工作负载迁移到的方法 AWS 云。有关更多信息，请参阅此词汇表中的 7 R 条目和动员组织以加快大规模迁移。

ML

参见机器学习。

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，请参阅中的应用程序现代化策略。 AWS 云

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，请参阅中的评估应用程序的现代化准备情况 AWS 云。

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

参见迁移组合评估。

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？”

可变基础架构

一种用于更新和修改现有生产工作负载基础架构的模型。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

请参阅源站访问控制。

OAI

参见源访问身份。

OCM

参见组织变更管理。

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

参见运营集成。

OLA

参见运营层协议。

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

参见开放流程通信-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)

一份问题清单和相关的最佳实践，可帮助您理解、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 Well-Architecte AWS d Frame work 中的运营准备情况评估 (ORR)。

操作技术 (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

由此创建的跟踪 AWS CloudTrail ，用于记录组织 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

参见运营准备情况审查。

OT

参见运营技术。

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

查看个人身份信息。

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

参见可编程逻辑控制器。

PLM

参见产品生命周期管理。

policy

一个对象，可以在中定义权限（参见基于身份的策略）、指定访问条件（参见基于资源的策略）或定义组织中所有账户的最大权限 AWS Organizations （参见服务控制策略）。

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。有关更多信息，请参阅在微服务中实现数据持久性。

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`或的查询条件`false`，通常位于子`WHERE`句中。

谓词下推

一种数据库查询优化技术，可在传输前筛选查询中的数据。这减少了必须从关系数据库检索和处理的数据量，并提高了查询性能。

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。

通过设计保护隐私

一种在整个开发过程中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

一种安全控制措施，旨在防止部署不合规的资源。这些控件会在资源配置之前对其进行扫描。如果资源与控件不兼容，则不会对其进行配置。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)

在产品的整个生命周期中，从设计、开发和上市，到成长和成熟，再到衰落和移除，对产品进行数据和流程的管理。

生产环境

参见环境。

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。

提示链接

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，以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

一系列步骤，例如指令，用于访问 SQL 关系数据库系统中的数据。

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

参见 “负责任、负责、咨询、知情” (RACI) 。

RAG

请参见检索增强生成。

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

参见 “负责任、负责、咨询、知情” (RACI) 。

RCAC

请参阅行和列访问控制。

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。

重新架构师

见 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

见 7 R。

区域

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 AWS 区域 您的账户可以使用的账户。

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

见 7 R。

版本

在部署过程中，推动生产环境变更的行为。

搬迁

见 7 R。

更换平台

见 7 R。

回购

见 7 R。

故障恢复能力

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

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

见 7 R。

退休

见 7 R。

检索增强生成（RAG）

一种生成式人工智能技术，其中法学硕士在生成响应之前引用其训练数据源之外的权威数据源。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据的过程。

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

参见恢复点目标。

RTO

参见恢复时间目标。

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 Management Console 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

参见监督控制和数据采集。

SCP

参见服务控制政策。

secret

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 Secret s Manager 密钥中有什么？ 在 Secrets Manager 文档中。

安全性源于设计

一种在整个开发过程中考虑安全性的系统工程方法。

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。安全控制主要有四种类型：预防性、侦测、响应式和主动式。

安全加固

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

一种预定义和编程的操作，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 实例或轮换证书。

服务器端加密

在目的地对数据进行加密，由接收方 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。

服务级别指示器 (SLI)

对服务性能方面的衡量，例如其错误率、可用性或吞吐量。

服务级别目标 (SLO)

代表服务运行状况的目标指标，由服务级别指标衡量。

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

参见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

参见服务级别协议。

SLI

参见服务级别指标。

SLO

参见服务级别目标。

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，请参阅中的分阶段实现应用程序现代化的方法。 AWS 云

恶作剧

参见单点故障。

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交易数据或测量数据，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。

监控和数据采集 (SCADA)

在制造业中，一种使用硬件和软件来监控有形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。

系统提示符

一种向法学硕士提供上下文、说明或指导方针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

tags

键值对，充当用于组织资源的元数据。 AWS 标签可帮助您管理、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

参见环境。

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。有关更多信息，请参阅量化深度学习系统中的不确定性指南。

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

参见环境。

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

一个 SQL 函数，用于对一组以某种方式与当前记录相关的行进行计算。窗口函数对于处理任务很有用，例如计算移动平均线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。

蠕虫

参见一次写入，多读。

WQF

参见AWS 工作负载资格框架。

一次写入，多次读取 (WORM)

一种存储模型，它可以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，但他们无法对其进行更改。这种数据存储基础架构被认为是不可变的。

Z

零日漏洞利用

一种利用未修补漏洞的攻击，通常是恶意软件。

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。

零镜头提示

向法学硕士提供执行任务的说明，但没有示例（镜头）可以帮助指导任务。法学硕士必须使用其预先训练的知识来处理任务。零镜头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 few-shot 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。