aws

简化 Amazon EKS 可观察性的最佳实践

AWS 规范性指导

AWS 规范性指导: 简化 Amazon EKS 可观察性的最佳实践

Table of Contents

简化 Amazon EKS 可观察性的最佳实践

Ishwar Chauthaiwale、Naveen Suthar 和 Pratap Kumar Nanda，Amazon Web Services ()AWS

2026 年 3 月（文件历史记录）

Amazon Elastic Kubernetes Service（Amazon EKS）需要全面的可观察性解决方案来有效地监控容器化工作负载并对其进行故障排除。分布式系统和微服务在 Amazon EKS 环境中具有复杂的架构，因此实施适当的可观察性实践对于维持可靠运行至关重要。Amazon EKS 环境中的有效可观察性使团队能够深入了解应用程序性能，高效解决问题并保持最佳集群运行状况。

挑战在于如何驾驭可用于 Amazon EKS 可观察性的庞大工具和技术生态系统，同时遵守符合组织目标和行业标准的最佳实践。有效的可观测性策略必须在全面的数据收集与性能考虑、成本效益和可扩展性之间取得平衡。

本指南旨在帮助组织在以下领域优化其 Amazon EKS 的可观察性：

- 建立高效的日志记录机制
- 实施强大的监控解决方案
- 对复杂架构使用分布式跟踪
- 实施警报和事件响应策略

通过采用这些最佳实践，您的组织可以增强其深入了解 Amazon EKS 环境的能力，从而提高可靠性、性能和运营效率。这种简化的可观察性方法有助于故障排除和维护，并支持数据驱动的决策，以持续改进基于 Kubernetes 的应用程序和基础架构。（有关 Amazon EKS 的详细信息，请参阅服务文档。）

本指南深入探讨了 Amazon EKS 可观察性的各个方面，并探讨了您可以定制的工具和策略，以满足 Amazon EKS 部署的特定需求，从小型应用程序到大型复杂的微服务架构。

在本指南中：

目标

本指南可以帮助您和您的组织实现以下业务目标：

- 增强运营可见性 — 通过有效的可观察性实践，全面了解您的 Amazon EKS 集群和应用程序。

  该目标强调了在整个 Amazon EKS 环境中保持全面可见性的重要性。诸如 AWS X-RayAmazon Contain CloudWatch er Insights 和 AWS Distro 之类的工具可 OpenTelemetry帮助您了解系统行为、快速发现问题并保持最佳性能。
- 提高故障排除效率 — 通过有效的跟踪和监控策略，缩短平均检测时间 (MTTD) 和平均解决时间 (MTTR)。

  该目标侧重于实施可观察性实践，从而能够快速识别和解决问题。分布式跟踪、有效日志记录和全面指标收集等技术是实现这一目标的关键。
- 主动性能管理-在潜在问题影响最终用户之前及早发现这些问题。

  主动监控对于保持高服务可用性和性能至关重要。该目标探讨了实施适当的警报、趋势分析和预测性监控以防止服务中断的重要性。
- 经济实惠的可观测性 — 优化可观测性成本，同时保持全面的系统可见性。

  成本优化包括实施有效的采样策略、适当的数据保留策略和最佳的仪器方法。目标是在确保有效的系统监控的同时，在可观测性需求和成本考虑之间取得平衡。
- 可扩展的监控架构 — 确保您的可观测性解决方案可与您的 Amazon EKS 环境无缝扩展。

  该目标侧重于实施可随您的应用程序增长的监控解决方案。无论您运行的是单集群还是多集群、多区域部署，您的可观测性策略都应相应扩展

登录 Amazon EKS

日志记录是管理和维护在 Amazon EKS 上运行的应用程序的关键方面。Amazon EKS 环境中的有效日志记录可帮助开发人员、运营团队和系统管理员获得有关其容器化应用程序及其底层基础设施的行为、性能和运行状况的宝贵见解。

在 Amazon EKS 中实施强大的日志策略是必不可少的，原因有很多：

- 故障排除：日志有助于快速识别和诊断问题，从而减少停机时间并提高系统的整体可靠性。
- 合规性：许多行业都需要全面的日志记录以进行审计和监管。
- 安全：日志分析可以帮助您检测和调查潜在的安全威胁或漏洞。
- 性能优化：日志提供有关应用程序和系统性能的见解，因此您可以识别瓶颈并优化资源利用率。
- 监控和警报：日志数据可用于设置监控系统并触发针对特定事件或条件的警报。

本节内容：

亚马逊 EKS 中的登录类型

在 Amazon EKS 中，日志包括捕获、存储和分析由 Kubernetes 集群的不同组件生成的各种类型的日志数据，包括：

- 系统日志：有关底层亚马逊弹性计算云 (Amazon EC2) 实例或节点的信息 AWS Fargate
- Kubernetes 组件日志 ：来自 Kubernetes 核心组件的数据，例如 API 服务器、调度程序和控制器管理器
- 容器运行时日志：来自容器运行时的信息，例如 Docker 或 containerd
- 应用程序日志：容器化应用程序的输出

为了有效地管理 Amazon EKS 环境中的日志，您通常会结合使用第三方工具和最佳实践。 AWS 服务这可能包括使用亚马逊 CloudWatch、Fluent Bit、Elasticsearch、Kibana 以及其他日志和分析工具来收集、存储和可视化日志数据。

以下各节探讨了 Amazon EKS 中日志记录的各个方面，包括在 Kubernetes 集群中实施全面日志策略的最佳实践、工具和技术。 AWS

系统日志

在 Amazon EKS 中记录底层 EC2 实例或 Fargate 节点涉及不同的方法，具体取决于节点类型。

要在 Amazon EKS 中实现 EC2 实例的日志记录，您可以使用以下工具：

- CloudWatch 代理：在您的 EC2 实例上安装和配置 CloudWatch 代理。将其配置为收集系统日志，例如`/var/log/messages`和`/var/log/secure`。您可以使用用户数据脚本或配置管理工具来自动执行此过程。
- Fluent Bi t：将 Fluent Bit 部署 DaemonSet 为从所有节点收集日志。将其配置为将日志转发到CloudWatch 日志或其他集中式日志系统。
- 容器见解：在 EKS 集群中启用容器见解，自动收集来自 EC2 实例的指标和日志。
- 自定义脚本：开发自定义脚本以收集特定日志并将其发送到您的首选日志目的地。
- SSM 代理：使用 AWS Systems Manager 代理（SSM 代理）收集日志并将其转发到日志。 CloudWatch

要在 Amazon EKS 中实现 Fargate 节点的日志记录，请使用以下工具：

- Fargate 日志记录：Fargate 会自动从您的容器中收集`stdout`和`stderr`记录日志。配置您的 Fargate 个人资料以将这些日志发送到日志。 CloudWatch
- Fargate 的 Fluent Bit： AWS 提供专门用于 Fargat e 日志的 Fluent Bit 图片。将其作为边车容器部署在 Fargate 吊舱中，以收集和转发日志。
- Fargate 容器见解：启用容器洞察以收集来自 Fargate 节点的指标和日志。

Kubernetes 组件日志

从 Amazon EKS 中的 API 服务器、调度程序和控制器管理器等 Kubernetes 组件收集日志，需要采用与应用程序日志稍微不同的方法。这些组件作为 Amazon EKS 控制平面的一部分运行，该控制平面由管理 AWS。您可以通过以下方式收集和访问这些日志：

- 启用控制平面日志记录：您可以通过 AWS 管理控制台、() 或基础设施即代码 (IaC AWS CLI) 工具 AWS Command Line Interface （例如AWS CloudFormation或 Terraform）为 EKS 集群启用控制平面日志记录。启用控制平面日志记录后，日志将发送到 Amazon CloudWatch Logs。您可以在

CloudWatch 控制台的`/aws/eks/<cluster-name>/cluster`日志组中查看它们。在此日志组中，每个控制平面组件都有自己的日志流，如下所示：

流名称	说明
kube-apiserver	Kubernetes API 服务器日志
kube-scheduler	调度程序决策日志
kube-controller-manager	控制器管理器日志
身份验证器	IAM 身份验证器日志
audit	Kubernetes 审计日志（必须明确启用）

要查看特定组件的日志，请导航到集群日志组并按目标日志流名称进行筛选。

- 使用 CloudWatch 日志见解：您可以使用 CloudWatch Logs Insights 对日志执行复杂查询。
- 将日志导出到 Amazon S3：为了进行长期存储或进一步分析，您可以将日志导出到亚马逊简单存储服务 (Amazon S3)。
- 使用第三方工具：您可以使用诸如 Fluent Bit 之类的工具来收集这些日志，并将其转发到其他日志系统，例如 Elasticsearch 或 Splunk。
- 使用 AWS CloudTrail：该AWS CloudTrail服务可以提供对您的 EKS 集群进行的 API 调用的更多见解。

容器运行日志

在 Amazon EKS 中记录容器运行时日志涉及从容器运行时捕获和管理日志，这通常`containerd`适用于 Amazon EKS。以下是在 Amazon EKS 中记录容器运行时日志的方法：

- 直接访问 Amazon EC2 节点上的日志。对于自行管理的 EC2 节点，您可以从以下位置直接访问主机上的容器运行时日志：
  - `containerd`日志：`/var/log/containers/`
  - Docker 日志（如果你使用的是 Docker 运行时）：`/var/log/docker.log`
- 使用 DaemonSet 进行日志收集。
- 将日志收集代理（例如 Fluent Bit）部署 DaemonSet 为，从所有节点收集日志。

- 将 CloudWatch 代理配置为收集容器运行时日志。
- 启用 “容器见解” 以收集容器运行时指标和日志。
- 使用 Fargate。对于 Fargate 节点，容器运行时日志会自动收集，并且可以通过日志进行访问。CloudWatch
- 使用 Fluent Bit 或 Logstash 等工具实现自定义日志解决方案。设置CloudWatch警报或使用诸如 Prometheus 之类的工具来监控容器运行时日志中的特定模式或问题。考虑使用与 Kubernetes 和 Amazon EKS 很好地集成的第三方日志解决方案，例如 Datadog、Splunk 或 Elastic Stack（ELK Stack）。使用日志聚合工具从多个来源收集日志，并将其转发到集中式日志系统。

应用程序日志

Amazon EKS 中的应用程序日志是维护应用程序和排除应用程序故障的关键部分。要在 Amazon EKS 中实现应用程序日志记录，您可以从以下选项中进行选择：

- 将日志写入`stdout/stderr`：处理应用程序日志的最简单、最基于 Kubernetes 的方法是将它们写入和。`stdout` `stderr`Kubernetes 会自动捕获这些流。
- 实现日志聚合：使用诸如 Fluent Bit 之类的日志聚合器从所有 pod 中收集日志。
- 配置日志路由：配置您的日志聚合器以将日志路由到所需的目的地（例如 CloudWatch 日志或 Elasticsearch）。
- 使用 CloudWatch 容器见解：启用容器见解以进行全面的日志记录和监控。

登录 Amazon EKS 的最佳实践

以下最佳实践有助于为您的 Amazon EKS 环境创建强大、可扩展且高效的日志系统，并为您的 Kubernetes 集群提供更好的故障排除、监控和整体管理。

- 集中日志收集：使用集中式日志解决方案（例如 CloudWatch 日志、Elasticsearch 或第三方服务）来聚合来自所有组件的日志。这为日志分析提供了单一访问点并简化了管理。
- 实现结构化日志：使用 JSON 等结构化日志格式，以便更轻松地解析和搜索日志。包括相关的元数据，例如时间戳、日志级别和源标识符。
- 适当使用日志级别：在应用程序中实现适当的日志级别（例如`DEBUG`、`INFO`、`WARN`、和`ERROR`）。将生产环境配置为在适当级别进行日志，以避免过多的日志记录。
- 启用容器日志记录：将您的容器配置为登录`stdout`和`stderr`。这允许 Kubernetes 捕获这些日志并将其转发到您选择的日志解决方案。

- 启用应用程序日志记录：将应用程序配置为向`stdout`日志写入日志，`stderr`而不是写入日志文件。这遵循 12要素应用程序方法，并符合云原生最佳实践。
- 使用 Kubernetes DaemonSets 进行日志收集：部署日志收集代理（例如 Fluent Bit），DaemonSets 以确保它们在集群中的每个节点上运行。
- 实施保留策略：定义和强制执行日志保留政策，以遵守法规并管理存储成本。
- 安全日志数据：对传输中的日志和静态日志进行加密。实施访问控制以限制谁可以查看和管理日志。
- 监控日志摄取：为日志提取失败或延迟设置警报，以确保持续记录。
- 使用 Kubernetes 注释和标签：使用 Kubernetes 注释和标签向日志添加元数据，以提高可搜索性和筛选性。
- 实现分布式跟踪：使用分布式跟踪工具（例如AWS X-Ray或 Jaeger）在微服务之间关联日志。
- 优化日志量：对记录的内容要有选择性，以避免不必要的成本和性能问题。对大容量、低价值的日志使用采样。
- 实现日志聚合：使用 Logstash 等工具汇总来自多个来源的日志，然后再将其发送到中央日志系统。
- 尽可能 AWS 服务 使用： CloudWatch 日志和容器见解等服务可与其他服务无缝集成 AWS 服务。
- 实现日志分析和可视化：使用 CloudWatch 日志见解、带有 Kibana 的 Elasticsearch 或第三方解决方案等工具进行日志分析和可视化。
- 实现自动日志分析：使用机器学习和 AI 驱动的工具自动检测日志中的异常和模式。
- 记录您的日志策略：为团队保留关于日志架构、实践和工具的清晰文档。

登录 Amazon EKS 的重要注意事项

本节讨论在 Amazon EKS 中实现登录时需要记住的重要注意事项。

- 性能影响：过多的日志记录可能会影响应用程序性能。请注意生成的日志的数量和频率。
- 成本管理：日志存储和处理可能会产生高昂的成本，尤其是在大规模的情况下。实施日志保留策略，并考虑使用日志聚合来降低成本。
- 安全性与合规性：确保日志不包含密码或个人数据等敏感信息。对传输中的日志和静态日志实施加密。在处理日志时，请考虑合规要求，例如《通用数据保护条例》(GDPR) 或《健康保险流通与责任法案》(HIPAA)。
- 可扩展性：确保您的日志解决方案可以根据集群大小和日志量进行扩展。考虑使用缓冲和批处理来传输日志。
- 日志保留：定义和实施适当的日志保留期。在合规性要求和存储成本之间取得平衡。

- 访问控制：为日志访问实施适当的 AWS Identity and Access Management (IAM) 角色和策略。遵循日志管理的最低权限原则。
- 日志一致性：在不同的应用程序和服务中使用一致的日志格式。使用结构化日志可以更轻松地进行解析和分析。
- 时间同步：同步所有节点的时间以在日志中获得一致的时间戳。
- 资源分配：为日志代理分配适当的资源（例如 CPU 和内存）。监控日志组件的资源使用情况。
- Fargate 注意事项：Fargate 具有与基于 EC2 的节点不同的特定日志记录机制。了解 Fargate 日志记录的局限性和功能。
- 多租户集群：在多租户环境中，请确保租户之间的日志正确隔离。
- 日志解析和分析：考虑有效的日志分析所需的工具和技能。实现日志解析以提取结构化数据。
- 监控日志系统：设置对日志基础架构本身的监控。生成有关记录系统故障或积压的警报。
- 网络影响：注意日志传输使用的网络带宽。考虑对日志数据使用压缩。
- Kubernetes 事件：不要忽视作为重要信息来源的 Kubernetes 事件。
- 控制平面日志记录：了解启用控制平面日志记录的含义和成本。
- 调试功能：确保您的日志解决方案便于调试和故障排除。
- 与现有工具集成：考虑一下您的 Amazon EKS 日志解决方案如何与现有监控和警报工具集成。
- 测试：定期测试您的日志设置，尤其是在集群升级之后。
- 文档：清晰地记录您的日志架构和实践。
- 日志聚合延迟：请注意日志聚合中的任何延迟以及它可能如何影响实时监控。

在 Amazon EKS 中监控

在 Amazon EKS 中进行监控可以关键地了解您的 Kubernetes 工作负载的运行状况、性能和安全性。如果没有适当的监控，您将面临服务中断、安全漏洞和资源利用效率低下的风险，这可能会影响业务运营并增加成本。有效的监控使您能够主动识别和解决问题、优化资源使用并维护容器化应用程序的合规性要求。通过实施全面的监控解决方案，您可以确保高可用性，尽早发现异常，并做出以数据为依据的决策，以扩展和改进 Amazon EKS 基础设施。

本节探讨了 Amazon EKS 监控的各个方面，包括不同的监控类型、可用工具和最佳实践，以帮助您为 Kubernetes 环境构建强大的监控策略。

本节内容：

亚马逊 EKS 中的监控类型

Amazon EKS 中的有效可观察性涉及基础设施、应用程序和安全监控活动。

基础设施监控

基础设施监控是 Amazon EKS 可观测性的基本组成部分，它可以深入了解您的 Kubernetes 集群基础元素的运行状况和性能。它的核心是跟踪控制平面组件和工作节点的生命体征，并确保底层平台保持稳定和高效。

- 控制平面监控至关重要，因为它可以监督 API 服务器、etcd 数据库和调度器等关键组件。通过监控 API 服务器延迟，您可以快速识别可能影响应用程序部署或扩展操作的性能瓶颈。Etcd 性能监控可验证集群的状态数据库是否有效运行，并防止可能影响整个集群的数据一致性问题。
- 节点级监控同样重要，因为它侧重于运行容器化工作负载的计算资源。这包括跟踪所有工作节点的 CPU 利用率、内存消耗、磁盘 I/O 和网络性能。了解这些指标有助于防止资源耗尽、优化节点扩展决策并确保适当的容量规划。

- 网络监控在维护 pod、服务和外部资源之间的可靠通信方面起着至关重要的作用。通过监控网络吞吐量、延迟和连接状态，您可以尽早发现连接问题并确保应用程序通信顺畅。存储监控通过跟踪卷性能、容量利用率和 I/O 模式来补充网络监控，以帮助防止与数据相关的瓶颈。

基础设施监控可作为潜在问题的预警系统，实现主动维护，并确保最佳资源分配。如果没有强大的基础架构监控，您将面临意外停机、性能下降和资源使用效率低下的风险，这可能会严重影响业务运营和成本。

应用程序监控

应用程序监控对于在 Amazon EKS 环境中维护健康、高性能和可靠的容器化应用程序至关重要。此级别的监控侧重于集群中运行的实际工作负载，并提供有关应用程序的行为、性能以及与其他服务交互的关键见解。

应用程序监控包括容器级监控、服务级别监控和分布式跟踪。

- 在容器级别，应用程序监控会跟踪关键指标，例如容器运行状况、重启次数和资源消耗模式。这些指标可帮助您识别可能消耗过多资源或频繁重启的有问题的容器，这些容器可能表明存在诸如内存泄漏或配置问题之类的潜在问题。通过监控容器生命周期事件，您可以确保应用程序行为正常，并快速解决部署问题。
- 服务级别监控提供对应用程序性能和可靠性指标的可见性，例如响应时间、错误率和请求吞吐量。这些指标对于维护服务级别目标 (SLOs) 和确保良好的终端用户体验至关重要。您可以跟踪不同服务端点之间的延迟，识别性能瓶颈，监控错误模式以保持应用程序的可靠性。
- 分布式跟踪是应用程序监控的另一个关键方面，尤其是在微服务架构中。通过实现跟踪，您可以跟踪请求流经不同服务的情况，了解依赖关系并识别性能瓶颈。这种 end-to-end可见性可帮助您优化服务交互并解决跨多个组件的复杂问题。

自定义应用程序指标在提供特定于业务的见解方面起着至关重要的作用。这些指标可能包括订单处理率、用户登录频率或交易成功率等指标。您可以将这些自定义指标与基础架构和容器指标关联起来，以更好地了解基础设施性能如何影响业务运营，并做出以数据为依据的扩展和优化决策。

应用程序监控的重要性在于它能够提供应用程序运行状况和性能的全面视图。这种监控使您能够保持高服务质量，快速解决问题，并持续优化应用程序以实现业务目标。

安全监控

Amazon EKS 中的安全监控是一项关键活动，可以帮助组织维护其 Kubernetes 环境的完整性、机密性和合规性。这种全面的安全方法结合了持续监控、威胁检测和合规性监控，可保护容器化工作负载免受

潜在安全风险和未经授权的访问的影响。它包括身份验证和授权监控、网络安全监控以及配置和合规性监控。

- 身份验证和授权监控通过跟踪所有访问集群的尝试来形成第一道防线。这包括监控 API 服务器请求、跟踪成功和失败的登录尝试以及审计基于角色的访问控制 (RBAC) 更改。通过维护详细的审计日志，记录谁访问了哪些资源以及何时访问了哪些资源，您可以快速检测潜在的安全漏洞、未经授权的访问尝试或权限升级活动。在必须保持严格的访问控制的多租户环境中，这一点尤其重要。
- 网络安全监控侧重于检测和防止 Pod 和服务之间未经授权的通信。通过监控网络策略违规行为和异常流量模式，您可以识别潜在的安全威胁，例如容器逃跑尝试或集群内的横向移动。这包括跟踪内部集群通信和外部流量模式，以确保容器仅与授权终端节点通信并遵循定义的安全策略。
- 配置和合规性监控对于维护安全基准和满足监管要求至关重要。它包括持续扫描容器映像中是否存在漏洞、监控运行时安全以及跟踪可能影响安全状况的配置更改。定期的合规性审计可确保遵守行业标准和组织安全政策，配置偏差检测有助于防止可能带来安全风险的未经授权的更改。

Amazon EKS 中的安全监控提供了必要的可见性和控制力，有助于抵御现代安全威胁，同时确保遵守监管要求。通过实施全面的安全监控，您的组织可以保持强大的安全态势，快速响应安全事件，并证明其符合各种监管标准。

适用于 Amazon EKS 的监控工具

本节讨论三类 Amazon EKS AWS 监控工具：监控服务、开源或专有解决方案以及专业工具。

AWS 服务

- Amazon CloudWatch：全面的监控和记录服务

  CloudWatch 构成了 AWS 监控解决方案的支柱，为 Amazon EKS 环境提供了广泛的功能。它为精细的容器和集群指标提供容器见解，因此您可以监控性能、资源利用率和应用程序运行状况。该服务在日志聚合和分析方面表现出色，并支持跨容器和节点的集中日志记录。 CloudWatch 与... 自然融为一体 AWS 服务。它提供自动警报配置，并支持自定义指标和控制面板，这使其成为 Amazon EKS 监控的必备工具。
- AWS X-Ray: 高级分布式追踪平台

  X-Ray 通过提供复杂的分布式跟踪功能来提高可观察性。其服务地图可视化可提供对应用程序架构和依赖关系的清晰见解，详细的请求跟踪有助于识别各服务的性能瓶颈。X-Ray 可以通过复杂的微服务架构跟踪请求，这对于故障排除和优化非常有用，尤其是在跨多个 AWS 服务分布式系统中。
- AWS 发行版 OpenTelemetry：统一可观测性框架

Distro for OpenTelemetry 提供统一的数据收集功能和跨平台支持，因此非常适合混合环境。该服务与其他服务集成 AWS 服务，支持定制仪器，在保持与行业标准的兼容性的同时，为实施全面的监控解决方案提供了灵活性。

- 亚马逊托管 Grafana：企业级可视化

  Amazon Managed Grafana 为数据可视化和分析提供完全托管的服务。它提供了与其他内置安全功能的无缝集成 AWS 服务，并具有企业级可扩展性。该服务简化了仪表板的创建和管理，同时还提供了高级功能，例如跨账户数据源访问和与 AWS IAM Identity Center集成。

- 适用于 Prometheus 的亚马逊托管服务：高度可用、安全、托管的监控

  适用于 Prometheus 的亚马逊托管服务是一项完全托管、与 Prometheus 兼容的监控服务。它提供自动扩展、高可用性以及安全的指标摄取和查询。该服务与 Amazon EKS 无缝集成，消除了管理 Prometheus 服务器的运营开销。

开源或专有解决方案

上一节中描述的 AWS 工具提供无缝集成和托管服务。本节中列出的开源工具 AWS 服务 通过提供灵活性和广泛的自定义选项来补充。了解每种工具的功能和用例有助于您设计最能满足您特定要求的监控策略。

- Prometheus：指标收集工具包

  Prometheus 是一款用于在 Kubernetes 环境中收集指标的开源解决方案。其时间序列数据库和 PromQL 查询语言可实现复杂的指标分析。该平台的服务发现功能可自动适应动态的 Kubernetes 环境，其警报管理系统可让您随时了解关键问题。Prometheus 提供了广泛的集成选项，使其成为全面指标监控的多功能选择。

- Grafana：高级可视化引擎

  Grafana 通过其可视化功能将复杂的监控数据转化为切实可行的见解。该平台可创建自定义仪表板，将来自多个来源的数据组合在一起，并提供基础架构和应用程序指标的统一视图。它支持各种数据源和警报管理功能，可提供全面的监控。Grafana 可以帮助您可视化实时和历史数据，因此您可以识别趋势并做出明智的决策。

- Fluent Bit：统一日志层

  该日志解决方案为 Kubernetes 环境提供日志收集和管理。其原生 Kubernetes 集成可确保从容器和节点无缝收集日志，并且它对多个输出目标的支持为日志存储和分析提供了灵活性。日志解析和筛选

等高级功能使您能够根据特定要求处理和路由日志。Fluent Bit 的轻量级特性使其特别适合容器化环境。

- Datadog：全栈可观察性

  Datadog 提供全面的监控功能，并支持原生 Kubernetes。它提供基础设施监控、应用程序性能监控 (APM)、日志管理和实时分析。您可以使用该平台的自动服务发现和广泛的集成目录进行Amazon EKS监控，并使用其机器学习功能来检测异常和预测潜在问题。

- 全新 Relic：应用程序性能监控

  New Relic 提供了对应用程序性能和基础架构运行状况的可见性。它的 Kubernetes 集成提供了详细的容器见解、分布式跟踪和自定义仪表板。该平台可帮助您将应用程序性能与基础架构指标关联起来，以便您可以快速识别和解决问题。

- EI@@ astic Stack（ELK Stack）：日志分析和搜索

  ELK Stack 结合了 Elasticsearch、Logstash 和 Kibana，提供日志管理和分析功能。它提供高级搜索功能、可视化工具和机器学习功能。您可以使用该堆栈来处理来自您的 Amazon EKS 环境的大量日志数据。

专业工具

您可以根据具体的监控要求、运营规模和组织偏好混合搭配以下工具。关键是要创建一个能够提供全面可见性的监控堆栈，同时保持可管理性和成本效益。

- kube-state-metrics (KSM)：Kubernetes 状态监控

  该附加服务监听 Kubernetes API 服务器并生成有关对象状态的指标。它提供了对部署、Pod 和其他 Kubernetes 资源的运行状况的见解。

- Kubernetes 指标服务器：资源指标

  该指标服务器从 kubelet 收集资源指标，并通过 Kubernetes 指标 API 将其公开。它提供横向 pod 自动缩放以及基本的 CPU 和内存指标。

- Kubecost：Kubernetes 成本监控

  诸如 Kubecost 之类的工具可为 EKS 集群提供详细的成本分析和优化建议。它们可以帮助您了解和优化不同命名空间、部署和服务的云支出。

为 Amazon EKS 监控解决方案实现高可用性

用于 Amazon EKS 监控的强大高可用性 (HA) 策略对于确保持续监控您的 Kubernetes 环境至关重要。本节讨论在监控基础架构的不同方面实施 HA 的全面方法。

架构冗余和可扩展性

要构建高度可用的监控系统，首先要进行适当的架构设计。监控组件应分布在多个 AWS 可用区中，以防出现区域故障。这包括对 Prometheus 服务器、日志收集器和警报管理器等关键监控组件实施横向扩展。您可以使用 AWS 托管服务，例如适用于 Prometheus 的亚马逊托管服务和 Amazon Managed Grafana，以帮助减少运营开销，同时确保高可用性。配置自动故障转移机制，以在组件故障期间保持服务连续性，同时设置运行状况检查和自动恢复程序。

弹性数据存储策略

数据存储弹性是保持监控系统可靠性的基础。实施分布式存储解决方案可确保即使单个存储节点出现故障，指标数据和日志仍可访问。这包括在多个可用区之间配置适当的数据复制，以及使用不同的存储后端实现冗余。为历史数据建立定期备份程序，并记录各种故障情形的恢复流程。对于 Prometheus 等时间序列数据库，实施远程存储解决方案有助于将存储问题与数据收集区分开来，并提高系统的整体可靠性。

冗余警报管理

在 HA 设置中需要特别注意警报管理。部署冗余警报管理器可确保即使在系统故障期间也能将关键通知送达目标收件人。配置多个通知渠道，例如电子邮件、短信、Slack， PagerDuty 并提供备用通信路径。使用警报重复数据删除机制来防止在部分系统故障期间出现警报风暴，并使用后备通知方法来确保不会错过关键警报。实施警报关联有助于在故障转移场景期间维护上下文，并防止来自冗余系统的重复通知。

负载平衡和服务发现

适当的负载平衡对于维持稳定的监控服务至关重要。 AWS 应用程序负载均衡器将传入的监控流量分发到多个终端节点，运行状况检查可确保流量仅路由到运行状况良好的实例。服务发现机制有助于监控组件自动适应环境的变化，例如添加新节点或服务。使用在集群扩展时确保全面覆盖 DaemonSets ，从而在所有节点上一致地部署监控代理。

其他 HA 注意事项

网络弹性：

- 实现冗余网络路径。
- 跨可用区配置正确的子网设计。
- AWS Direct Connect与备用路由一起使用。
- 配置适当的安全组和网络访问控制列表（网络 ACLs）。

监视显示器：

- 部署辅助监控系统。
- 实施跨区域监控。
- 为无响应的系统配置警报。
- 定期测试故障转移程序。

容量规划：

- 监控资源使用趋势。
- 实现预测性扩展。
- 定期测试性能。

数据管理：

- 实施数据保留政策。
- 配置指标聚合。
- 规划数据生命周期管理。
- 定期优化存储。

恢复程序：

- 文档恢复流程。
- 定期测试灾难恢复。
- 尽可能实施自动恢复。
- 确定并实施明确的上报路径。

通过实施这些高可用性实践，您可以确保您的 Amazon EKS 监控基础设施保持可靠性和弹性，并且即使在各种故障情况下，您也可以持续监控 Kubernetes 环境。定期测试和更新这些 HA 配置可确保它们随着环境的演变而保持有效。

在 Amazon EKS 中进行监控的最佳实践

战略实施方法

成功的 Amazon EKS 监控策略始于精心策划的分阶段实施方法。

- 首先确定和监控直接影响业务运营和应用程序可靠性的关键指标。该基础应包括基本的基础架构指标、关键应用程序性能指标和关键安全指标。根据业务需求和经验教训逐步扩大监测范围，并确保每一项增加都能提供有意义的价值。
- 使用 Terraform 等基础设施即代码 (IaC) 工具实施自动化部署流程，或者确保一致性和 CloudFormation 可重复性。
- 测试和验证监控系统，以帮助保持可靠性和准确性。
- 不断完善监控参数，以适应不断变化的业务需求。

有效的数据管理

正确的数据管理对于维护高效且具有成本效益的监控解决方案至关重要。

- 实施明确的数据保留政策，在历史分析需求和存储成本之间取得平衡。
- 为不同的指标类型配置适当的采样率：关键指标的频率更高，不太关键的指标的频率较低。
- 使用指标聚合来减少数据量，同时保持有意义的见解，特别是对于长期趋势分析。
- 为集中式日志系统（例如 CloudWatch 日志）实施系统的日志保留和存档程序，以管理存储成本并保持对重要数据的访问权限。

> **Note**
>
> 在 Amazon EKS 1.21 或更高版本中，kubelet 会自动处理容器级别的日志轮换。

- 考虑实施日志存储 hot-warm-cold架构，以优化访问速度和成本效益。

警报配置和管理

警报配置需要仔细考虑，以便在不造成警报疲劳的情况下保持有效性。

- 根据服务级别目标 (SLOs) 和历史性能模式定义清晰、可操作的阈值。
- 实施分级警报严重度系统，明确区分需要立即关注的关键问题和不太紧急的问题。
- 确保警报提供足够的背景信息和可操作的信息，以便于快速解决问题。
- 针对不同的警报严重程度，制定明确的上报程序，明确所有权和响应时间。
- 定期审查和完善警报配置，以帮助保持其相关性和有效性。

资源优化

持续监控资源利用率对于维持具有成本效益的运营至关重要。

- 对所有集群组件（包括节点、Pod 和永久卷）实施全面的资源监控。
- 根据实际使用模式和性能要求配置自动扩展，以确保高效利用资源，同时保持性能。
- 使用成本分配标签来跟踪不同团队、应用程序或环境的资源消耗。
- 定期分析资源效率指标，以确定优化机会并实施改进。
- 考虑实施成本管理工具来跟踪和优化云支出。

安全性

安全考虑应该是您的监控策略不可或缺的一部分。

- 对所有监控组件实施最低权限访问原则，以确保用户和服务仅拥有他们需要的权限。
- 启用全面的审核日志，以跟踪对监控系统的所有访问和更改。
- 定期对监控配置和访问模式进行安全审查，以识别潜在的漏洞。
- 对传输中和静态的敏感监控数据实施加密。
- 将安全监控与现有的安全信息和事件管理 (SIEM) 系统集成，以实现全面的安全可见性。

Amazon EKS 中的高级监控注意事项

性能优化：

- 优化指标收集间隔。

- 配置高效的查询模式。
- 实现指标预聚合。
- 使用适当的存储解决方案。

合规与治理：

- 维护审计跟踪。
- 实施合规监控。
- 定期提供合规报告。
- 文件监控程序。

灾难恢复：

- 定期备份监控配置。
- 文件恢复程序。
- 测试恢复过程。

持续改进：

- 定期监控复习环节。
- 优化性能周期。
- 根据事件更新监控。
- 纳入用户反馈。

这些最佳实践为实施和维护 Amazon EKS 环境的有效监控解决方案提供了一个框架。定期审查和更新这些做法，使其与您的组织需求和行业标准保持一致。监控不是一次性设置，而是一个持续的过程，需要定期关注和完善。

在 Amazon EKS 中追踪

在 Amazon EKS 中，跟踪是 Amazon EKS 中应用程序可观察性的关键组成部分。当请求通过部署在 EKS 集群上的各种微服务时，Tracing 通过收集、处理和可视化请求的路径，提供对请求流和服务交互的详细可见性。此功能可帮助您了解您的 Amazon EKS 环境中的系统行为、识别瓶颈并有效地解决问题。有效的跟踪提供了对请求流 end-to-end的可见性，从而消除了调试分布式系统的复杂性。它可以跨服务边界跟踪交易，并识别 Amazon EKS 工作负载中的性能问题或故障。

Amazon EKS 中的整体跟踪实现使您能够了解系统行为、优化性能并保持容器化应用程序的可靠性。最终，跟踪功能增强了 Amazon EKS 环境中的操作可见性和系统可维护性。

AWS X-Ray 在跟踪有关您的应用程序的数据方面起着重要作用。跟踪涉及监控服务交互的各个方面，包括：

- 请求路径和依赖关系为分布式系统的行为提供了至关重要的见解。当请求遍历不同的微服务和组件时，它们会跟踪请求的完整旅程。映射服务依赖关系可帮助您了解通信模式并确定应用程序架构中的关键路径。有关实现的详细信息，请参阅 X-Ray 文档中的使用 AWS X-Ray 服务跟踪地图。
- 服务延迟和瓶颈是保持最佳系统性能的基本指标。通过测量和分析服务之间的响应时间，您可以有效地识别性能问题。这些数据使您可以查明导致请求链延迟的特定服务或操作，并进行有针对性的优化工作。要了解有关延迟分析的更多信息，请参阅 X-Ray 文档中的与 Analytics 控制台交互。
- 错误传播模式可帮助您了解系统的可靠性和容错能力。通过跟踪服务间的错误路径来了解故障如何在系统中层叠出现，您可以更好地架构应用程序。这种可见性可以帮助您确定错误的根本原因及其对依赖服务的影响，从而提高系统的弹性。有关实现的详细信息，请参阅 X-Ray 文档中的跟踪。
- 跨服务的资源利用率提供了对系统效率和成本优化的见解。您可以监控与跟踪数据相关的 CPU、内存和网络使用模式，以了解资源需求。这些数据可帮助您分析资源消耗趋势，从而优化整个 EKS 集群的服务性能和成本。有关监控设置，请参阅 Amazon EKS 文档中的监控集群性能和查看日志。
- 最终用户交易流程对于理解和改善用户体验至关重要。通过跟踪从前端到后端服务的完整用户交互，您可以确保最佳的应用程序性能。您可以衡量和优化关键用户旅程的 end-to-end响应时间，这会直接影响客户满意度。要实现终端用户监控，请使用适用于您的编程语言的 AWS X-Ray SDK。
- API 网关交互构成了应用程序性能和安全性的第一线。您可以在 API 入口点监控请求模式和性能，以确保最佳的服务交付。这种可见性可帮助您跟踪身份验证、授权和速率限制对请求流的影响，以满足安全和性能要求。在 Amazon API Gateway with X-Ray 文档中了解有关 API 跟踪的更多信息。

Amazon EKS 中的有效追踪不仅仅是收集跨度和痕迹。它需要一个结构良好的策略，在可观察性需求和系统性能之间取得平衡。该策略应侧重于：

- 实施适当的采样率：根据流量模式和业务优先级配置采样规则，以优化成本，同时保持关键交易的可见性。要了解更多信息，请参阅 X-Ray 文档中的配置采样规则。
- 定义要跟踪的关键路径和服务：确定需要详细跟踪的基本服务和用户旅程并确定其优先级，以确保最佳性能监控。有关更多信息，请参阅 Amazon EKS 文档中的使用 ADOT 操作员发送指标和跟踪数据。
- 制定适当的数据保留政策：设置数据生命周期管理规则，在可观察性需求与存储成本和合规性要求之间取得平衡。 要查看 CloudWatch 保留策略，请参阅日志文档中的使用日志组和 CloudWatch 日志流。
- 设置有效的可视化和分析工具：部署和配置可视化工具，例如 AWS X-Ray 分析控制台或 Amazon Managed Grafana，以有效地分析跟踪数据。 有关更多信息，请参阅 X-Ray 文档中的与分析控制台交互。

本节内容：

适用于 Amazon EKS 的追踪工具

Amazon EKS 支持多种 AWS 和第三方选项来实现分布式跟踪。

AWS 服务

- AWS X-Ray: 高级分布式追踪平台

  X-Ray 是完全托管 AWS 服务 的，可提供 end-to-end跟踪功能。它会自动为您 AWS 服务 在 Amazon EKS 上运行的应用程序提供详细的服务地图和分析。X-Ray 与包括 Amazon AWS 服务在内的其他产品集成 CloudWatch，可自动将跟踪与 AWS 服务 通话进行关联。

- AWS 发行版 OpenTelemetry：统一可观测性框架

  Distro for OpenTelemetry 是适用于云原生应用程序的安全、生产就绪且 AWS受支持的发行版。OpenTelemetry 它提供供应商中立的仪器功能，同时保持原生 AWS 服务 集成，这使其成为混合云环境的理想之选。Distro fo OpenTelemetry r 支持多个可观察性后端，并提供与 AWS 监控服务的无缝集成。

开源解决方案

- OpenTelemetry: 开源可观测性框架

  OpenTelemetry 提供了一个标准化的可观测性框架，其中包含支持多种编程语言的全面仪器库。其灵活的后端选项和供应商中立的方法使其非常适合需要在不同环境中保持一致性的工作负载。该框架广泛的生态系统确保了与各种监控解决方案的广泛兼容性。

- Jaeger：开源分布式追踪平台

  Jaeger 通过实时分布式上下文传播提供全面的跟踪功能。它通过详细的服务依赖关系可视化提供根本原因分析和性能优化。Jaeger 的架构专为高可扩展性而设计，支持各种存储后端，因此适用于大规模 Amazon EKS 部署。查看 Jaeger for EKS 设置

- Grafana Tempo：分布式跟踪

  Tempo 是 Grafana Labs 的解决方案，可提供大规模的跟踪存储以及与 Prometheus 指标的无缝集成。其经济实惠的跟踪保留模型以及与 Grafana 的原生集成使其适合已经使用 Grafana 进行可视化的组织。Tempo 的架构专为云原生环境（例如 Amazon EKS）而设计。

在 Amazon EKS 中进行追踪的最佳实践

本节全面列出了创建有效跟踪系统的最佳实践和技术，该系统可增强在 Amazon EKS 中基于 Kubernetes 的应用程序的可观察性和故障排除。

- 策略抽样：根据应用程序的流量模式和所用服务的重要性配置不同的采样率。对关键路径实施更高的采样率，同时减少对大批量、不太关键的路径的采样以优化成本。有关指导，请参阅 AWS X-Ray 文档中的配置采样规则。
- 仪器设置：使用自动检测工具（例如 X-Ray SDK 或 AWS Distro for C OpenTelemetry ollectory），以最大限度地减少手动检测工作。保持一致的命名约定和跨服务的上下文传播，以实现更好的跟踪关联。 有关更多信息，请参阅 OpenTelemetry 收藏版文档。
- 数据管理：实施适当的保留期和压缩策略，以平衡存储成本和可观测性需求。建立明确的数据隐私控制和备份程序，以保护敏感的跟踪数据。有关更多信息，请参阅“日志” 文档中的 “ CloudWatch 日志” 中的更改 CloudWatch 日志数据保留期。
- 性能优化：监控和优化跟踪开销，以最大限度地减少对应用程序性能的影响。使用高效的缓冲和异步处理来减少延迟影响。有关更多信息，请参阅 X-Ray 文档中的配置 AWS X-Ray 守护程序。
- 安全控制：使用 IAM 角色和策略实施适当的访问控制和数据保护措施。定期的安全审计和合规性审查有助于确保追踪数据的安全。有关更多信息，请参阅 X-Ray 文档AWS X-Ray中的安全。

- 监控和警报：设置对跟踪集合运行状况的全面监控，并为收集问题配置警报。跟踪采样率和系统性能指标，以确保最佳运行。有关更多信息，请参阅 CloudWatch 文档中的容器见解。
- 高可用性：跨可用区部署冗余收集器并配置适当的故障转移机制。定期测试高可用性设置可确保可靠的跟踪采集。有关更多信息，请参阅亚马逊 Prometheus 托管服务文档中的使用 AWS 发行版 OpenTelemetry 作为收集器。

通过遵循这些最佳实践，您可以为您的 Amazon EKS 环境创建强大、高效且有效的跟踪系统。这将有助于确保基于 Kubernetes 的应用程序具有全面的可观察性、高效的故障排除和最佳性能。

在 Amazon EKS 中提醒

警报是管理和维护在 Amazon EKS 上运行的应用程序的关键组成部分。它是一种预警系统，可在潜在问题、异常或性能下降升级为可能影响服务可用性或用户体验的严重问题之前通知运营商和开发人员。警报涉及监控 Kubernetes 集群的各个方面，包括：

- 基础设施运行状况
- 应用程序性能
- 容器指标
- 自定义业务指标

Amazon EKS 中的有效警报不仅仅是设置通知。它需要一种在及时提供信息的需求与警报疲劳的可能性之间取得平衡的 well-thought-out策略。该策略应该：

- 定义有意义的阈值和条件。
- 根据严重性和影响确定警报的优先级。
- 实施适当的路由和上报程序。
- 与事件管理和通信工具集成。

本节内容：

适用于 Amazon EKS 的警报工具

Amazon EKS 支持多种 AWS 用于实施警报的第三方选项。在选择 Amazon EKS 警报工具时，请考虑集成能力、可扩展性、易用性、成本以及与您的监控和警报要求相一致的特定功能等因素。许多组织结合使用这些工具为其 Amazon EKS 环境创建全面的监控和警报解决方案。

- Amazon CloudWatch：AWS 服务 用于监控和可观察性

  CloudWatch 为 EKS 集群提供指标、日志和警报，并与其他集群很好地集成 AWS 服务。
- Prometheus：适用于 Kubernetes 的开源监控和警报工具

  Prometheus 提供了一种用于定义警报条件的强大查询语言 (PromQL)。

- Alertmanager：Prometheus 的伴侣，用于处理警报

  Alertmanager 提供重复数据删除、分组和发送警报功能。它支持各种通知渠道，包括电子邮件、Slack 和。 PagerDuty

- Grafana：用于监控和可观察性的开源平台

  Grafana 提供可视化和警报功能。它可以与各种数据源集成，包括 Prometheus 和。 CloudWatch

- El@@ astic Stack（ELK Stack）：Elasticsearch、Logstash 和 Kibana 的组合

  此工具可用于日志聚合、分析和警报。它可以通过 Elastic 的可观测性功能进行扩展。

- 第三方解决方案

  市场上有许多工具可供选择，包括 Datadog、New Relic、Sysdig、Dynatrace、Zabbix、Nagios、Splunk、IBM Instana 和。 AppDynamics

在 Amazon EKS 中提醒的最佳实践

本节介绍在 Amazon EKS 中创建强大的警报系统以增强基于 Kubernetes 的应用程序的可靠性和性能的最佳实践。

定义明确的警报阈值：

- 根据历史数据和业务需求设置有意义的阈值。
- 在适当时使用动态阈值来应对不同的工作负载。

实现警报优先级：

- 按严重性（例如，严重、高、中、低）对警报进行分类。
- 根据业务影响调整警报优先级。

避免警觉疲劳：

- 通过消除冗余或低值警报来减少噪音。
- 将警报与群组相关问题关联起来。

使用多阶段警报：

- 在达到临界水平之前实施警告阈值。
- 针对不同的警报严重程度使用不同的通知渠道。

实现正确的警报路由：

- 确保将警报发送给正确的团队或个人。
- 使用待命时间表和轮换进行全天、每天的保险。

利用 Kubernetes 原生指标：

- 监控 Kubernetes 的核心组件（节点、容器、服务）。
- 使用 kube-state-metrics (KSM) 获取其他 Kubernetes 对象指标。

监控基础架构和应用程序：

- 设置集群运行状况、节点状态和资源利用率警报。
- 实施特定于应用程序的警报，例如错误率和延迟。

使用 Prometheus 和 Alertmanager：

- 使用 Prometheus 进行指标收集，使用 PromQL 来定义警报条件。
- 使用警报管理器进行警报路由和重复数据删除。

与 Amazon 集成 CloudWatch：

- 使用CloudWatch容器洞察获取 Amazon EKS 特定的指标。
- 为关键 AWS 资源指标设置CloudWatch警报。

实施情境丰富的警报：

- 在警报消息中包含相关信息，例如集群名称、命名空间和 Pod 详细信息。
- 在警报中提供指向相关仪表板或运行手册的链接。

使用异常检测：

- 针对复杂模式实施基于机器学习的异常检测。
- 使用 CloudWatch 异常检测或第三方工具等服务。

实现警报抑制和静音：

- 允许暂时抑制已知问题。
- 实施维护窗口，以减少计划停机期间的噪音。

监控警报性能：

- 跟踪警报频率、解决时间和误报率等指标。
- 根据这些指标定期审查和完善警报规则。

实施上报程序：

- 为未解决的警报定义清晰的上报路径。
- 使用 PagerDuty 或 Opsgenie 等工具进行自动升级。

定期测试警报系统：

- 定期测试您的警报渠道。
- 在灾难恢复演习中包括警报测试。

使用模板确保警报一致性：

- 为常见场景创建标准化警报模板。
- 确保所有警报的格式和信息保持一致。

实施速率限制：

- 通过对频繁触发的警报实施速率限制来防止警报风暴。

使用自定义指标：

- 为特定于应用程序的监控实现自定义指标。

- 使用 Kubernetes 自定义指标 API 根据这些指标进行自动扩展。

实现日志集成：

- 将警报与相关日志关联起来，以便更快地进行故障排除。
- 将 Grafana Loki 或 ELK Stack 等工具与警报系统配合使用。

考虑成本提醒：

- 设置警报，以防资源使用量或成本出现意外高峰。
- AWS Budgets使用我们的第三方成本管理工具。

使用分布式跟踪：

- 集成分布式跟踪工具，例如 Jaeger 或. AWS X-Ray
- 设置异常跟踪模式或延迟的警报。

文档警报操作手册：

- 为每种警报类型创建清晰、可操作的操作手册。
- 在运行手册中包括故障排除步骤和上报程序。

通过遵循这些最佳实践，您可以为您的 Amazon EKS 环境创建强大、高效且有效的警报系统。这将有助于确保基于 Kubernetes 的应用程序具有高可用性、快速解决问题和最佳性能。

后续步骤

本指南为在 Amazon EKS 环境中实现强大的可观察性提供了一个全面的框架，重点是指标收集、日志基础设施、分布式跟踪和成本优化。通过了解和应用这些核心组件，您可以构建一个高度可观察、可维护且经济实惠的容器环境，从而深入了解应用程序和基础设施行为。 AWS 服务 诸如 Amazon Container Insights 之类的集成 AWS X-Ray，再加上 Prometheus 和等开源解决方案，为监控 CloudWatch 容器化 OpenTelemetry应用程序和排除故障奠定了坚实的基础。

成功实施取决于分阶段的方法，从收集核心指标开始，然后逐步扩展到全面的日志记录和分布式跟踪功能。我们建议您首先评估当前的监控能力，找出差距，然后选择符合您的运营要求和团队专业知识的适当工具组合。这种有条不紊的方法可确保可观测性堆栈的每个组成部分都得到正确实施和集成，同时团队开发必要的技能和流程以有效使用这些工具。

Amazon EKS 可观察性的长期可持续性取决于成本、资源和流程的定期优化。您应不断审查和调整您的可观测性基础架构，包括数据保留策略、采样率和资源分配，以便在全面监控和运营效率之间保持适当的平衡。这种迭代改进方法，再加上持续的团队培训和文档更新，使您的组织能够保持有效的可观察性，同时支持业务增长并适应不断变化的应用程序架构。

资源

AWS 文档

- 亚马逊 EKS 最佳实践指南
- Amazon CloudWatch 容器洞察
- Amazon Managed Service for Prometheus
- Amazon Managed Grafana
- AWS 和的 OpenTelemetry 发行版 AWS X-Ray
- 亚马逊 OpenSearch 服务

AWS 博客文章

- Amazon EKS 增强了 Kubernetes 控制平面的可观察性
- 使用适用于 Prometheus 托管抓取器的亚马逊托管服务在亚马逊 EKS 上自动收集指标
- 使用 CloudWatch 容器洞察自动监控您的 Amazon EKS 集群
- 使用适用于 Amazon EKS 的托管监控解决方案增强可观察性

其他资源

- OpenTelemetry 文档
- Prometheus 文档
- 流畅的比特文档
- Kubernetes 文档中的@@ 监控、记录和调试

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
更新	我们更新了 “在 Amazon EKS 中登录” 一章。	2026年3月17日
初次发布	—	2025 年 4 月 10 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 AM) 文档 AWS中的 AB AC。

权威数据来源

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

可用区

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

AWS 云采用框架 (AWS CAF)

该框架包含指导方针和最佳实践 AWS ，可帮助组织制定高效且有效的计划，以成功迁移到云端。AWS CAF将指导分为六个重点领域，称为视角：业务、人员、治理、平台、安全和运营。业务、人员和治理角度侧重于业务技能和流程；平台、安全和运营角度侧重于技术技能和流程。例如，人员角度针对的是负责人力资源（HR）、人员配置职能和人员管理的利益相关者。从这个角度来看，AWS CAF 为人员发展、培训和沟通提供了指导，以帮助组织为成功采用云做好准备。有关更多信息，请参阅 AWS CAF 网站和 AWS CAF 白皮书。

AWS 工作负载资格框架 (AWS WQF)

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

B

恶意机器人

一种旨在扰乱或伤害个人或组织的机器人。

BCP

请参阅业务连续性计划。

行为图

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

大端序系统

一个先存储最高有效字节的系统。另请参阅字节顺序。

二进制分类

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

bloom 筛选条件

一种概率性、内存高效的数据结构，用于测试元素是否为集合的成员。

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。

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。此账户被称为该服务的委托管理员。有关更多信息和兼容服务列表，请参阅 AWS Organizations 文档中使用 AWS Organizations的服务。

部署

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

开发环境

请参阅环境。

侦测性控制

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

开发价值流映射（DVSM）

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

数字孪生

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

维度表

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

灾难

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

灾难恢复（DR）

您用来最大程度地减少由灾难造成的停机时间和数据丢失的策略和流程。有关更多信息，请参阅《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

IaC

请参阅基础设施即代码。

基于身份的策略

附加到一个或多个 IAM 委托人的策略，用于定义他们在 AWS 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、记录和控制在车间将原材料转化为成品的生产过程。

MAP

请参阅迁移加速计划。

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。你可以使用 Amazon S CloudWatch ynthetics 来创建这些测试。

系统提示

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

T

标签

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。