使用 Amazon 设计和实施日志记录和监控 CloudWatch

# AWS 规范性指导

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

# AWS 规范性指导: 使用 Amazon 设计和实施日志记录和监控 CloudWatch

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

简介 ..... 1
目标业务成果 ..... 4
加快运营准备就绪 ..... 4
改善卓越运营 ..... 4
提高运营可见性 ..... 4
扩大运营规模，降低管理成本 ..... 4
规划您的 CloudWatch 部署 ..... 6
CloudWatch 在集中式或分布式账户中使用 ..... 6
管理 CloudWatch 代理配置文件 ..... 9
管理 CloudWatch 配置 ..... 9
示例：将 CloudWatch 配置文件存储在 S3 存储桶中 ..... 11
为 EC2 实例和本地服务器配置 CloudWatch 代理 ..... 13
配置代 CloudWatch 理 ..... 13
为 EC2 实例配置日志捕获 ..... 13
为 EC2 实例配置指标捕获 ..... 15
System-level CloudWatch 配置 ..... 17
配置系统级日志 ..... 17
配置系统级指标 ..... 19
Application-level CloudWatch 配置 ..... 20
配置应用程序级日志 ..... 20
配置应用程序级指标 ..... 21
CloudWatch 适用于 Amazon EC2 和本地服务器的代理安装方法 ..... 23
使用 Systems Manager 分发服务器和状态管理器安装 CloudWatch 代理 ..... 23
为 CloudWatch 代理部署和配置设置状态管理器和分发服务器 ..... 24
使用 Systems Manager 快速设置并手动更新已创建的 Systems Manager 资源 ..... 26
使用 CloudFormation 代替快速设置 ..... 27
在单个账户和带有 CloudFormation 堆栈的区域中进行自定义快速设置 ..... 27
在多个地区和多个账户中自定义快速设置 CloudFormation StackSets ..... 28
配置本地服务器的注意事项 ..... 29
临时 EC2 实例的注意事项 ..... 30
使用自动解决方案部署代 CloudWatch 理 ..... 31
在实例置备期间使用用户数据脚本部署 CloudWatch 代理 ..... 31
将 CloudWatch 代理包括在你的 AMIs ..... 31
在 Amazon ECS 上进行日志记录和监控 ..... 33

使用 EC2 启动类型 CloudWatch 进行配置 ..... 33
适用于 EC2 和 Fargate 启动类型的 Amazon ECS 容器日志 ..... 34
在 Amazon ECS 中使用自定义日志路由 FireLens ..... 35
亚马逊 ECS 的指标 ..... 36
在 Amazon ECS 中创建自定义应用程序指标 ..... 36
Amazon EKS 中的日志记录和监控 ..... 38
登录 Amazon EKS ..... 38
Amazon EKS 控制面板日志记录 ..... 39
Amazon EKS 节点和应用程序日志 ..... 39
在 Fargate 上登录亚马逊 EKS ..... 41
亚马逊 EKS 和 Kubernetes 的指标 ..... 41
Kubernetes 控制平面指标 ..... 41
Kubernetes 的节点和系统指标 ..... 42
应用程序指标 ..... 43
Fargate 上亚马逊 EKS 的指标 ..... 43
Prometheus 在亚马逊 EKS 上进行监控 ..... 44
的日志和指标 AWS Lambda ..... 45
Lambda 函数日志 ..... 45
将日志发送到其他目的地 CloudWatch ..... 46
Lambda 函数指标 ..... 46
系统级指标 ..... 46
应用程序指标 ..... 47
搜索和分析日志 CloudWatch ..... 48
使用 “应用洞察” 对 CloudWatch 应用程序进行集体监控和分析 ..... 48
使用 “日志见解” 执行 CloudWatch 日志分析 ..... 50
使用 Amazon OpenSearch 服务执行日志分析 ..... 52
带有的警报选项 CloudWatch ..... 54
使用 CloudWatch 警报进行监控和报警 ..... 54
使用 CloudWatch 异常检测进行监控和报警 ..... 54
跨多个地区和账户发出警报 ..... 55
使用 EC2 实例标签自动创建警报 ..... 55
监控应用程序和服务可用性 ..... 56
使用跟踪应用程序 AWS X-Ray ..... 57
部署 X-Ray 守护程序以跟踪亚马逊 EC2 上的应用程序和服务 ..... 57
部署 X-Ray 守护程序以跟踪亚马逊 ECS 或 Amazon EKS 上的应用程序和服务 ..... 58
配置 Lambda 以跟踪对 X-Ray 的请求 ..... 58

针对 X-Ray 应用程序进行仪器测试 ..... 58
配置 X-Ray 采样规则 ..... 59
仪表板和可视化效果 CloudWatch ..... 60
创建跨服务仪表板 ..... 60
创建特定于应用程序或工作负载的仪表板 ..... 60
创建跨账户或跨区域控制面板 ..... 60
使用公制数学来微调可观察性和警报 ..... 61
使用适用于亚马逊 ECS、Amazon EKS 和 Lambda 的自动控制面板以及 CloudWatchContainer 洞察和 Lambda Insights CloudWatch ..... 61
CloudWatch 与 AWS 服务集成 ..... 63
用于生成控制面板和可视化的 Amazon Managed Grafana ..... 64
常见问题解答 ..... 66
我在哪里存储我的 CloudWatch 配置文件？ ..... 66
警报发出后，如何在我的服务管理解决方案中创建工单？ ..... 66
如何使用 CloudWatch 来捕获容器中的日志文件？ ..... 66
如何监控 AWS 服务的运行状况问题？ ..... 66
在没有代理支持的情况下，如何创建自定义 CloudWatch 指标？ ..... 66
如何将我现有的日志和监控工具与集成 AWS？ ..... 67
资源 ..... 68
简介 ..... 68
目标业务成果 ..... 68
规划您的 CloudWatch 部署 ..... 68
为 EC2 实例和本地服务器配置 CloudWatch 代理 ..... 68
CloudWatch 适用于 Amazon EC2 和本地服务器的代理安装方法 ..... 69
在 Amazon ECS 上进行日志记录和监控 ..... 69
Amazon EKS 中的日志记录和监控 ..... 70
的日志和指标 AWS Lambda ..... 70
搜索和分析日志 CloudWatch ..... 71
带有的警报选项 CloudWatch ..... 71
监控应用程序和服务可用性 ..... 71
使用跟踪应用程序 AWS X-Ray ..... 72
仪表板和可视化效果 CloudWatch ..... 72
CloudWatch 与 AWS 服务集成 ..... 72
用于生成控制面板和可视化的 Amazon Managed Grafana ..... 72
文档历史记录 ..... 73
..... lxxiv

# 使用 Amazon 设计和实施日志和监控 CloudWatch

Khurram Nizami, Amazon Web Services (AWS)

2023 年 4 月（文档历史记录）

本指南可帮助您使用亚马逊 CloudWatch和相关的亚马逊网络服务 (AWS) 管理和治理服务为使用亚马逊弹性计算云 (Amazon EC2) 实例、亚马逊弹性容器服务 (Amazon ECS)、亚马逊弹性容器服务 (Amazon ECS) 、亚马逊弹性 Kub ernetes 服务 (A mazon EKS) 和本地服务器的工作负载设计和实施日志和监控。AWS Lambda该指南适用于在 AWS 云端管理工作负载的运营团队、 DevOps 工程师和应用工程师。

您的日志和监控方法应基于 Well-Architecte AWS d Framework 的六大支柱。这些支柱是卓越运营、安全性、可靠性、性能效率和成本优化。架构良好的监控和警报解决方案可帮助您主动分析和调整基础架构，从而提高可靠性和性能。

本指南并未广泛讨论安全性或成本优化的日志记录和监控，因为这些都是需要深入评估的主题。有许多 AWS 服务支持安全记录和监控，包括、Amazon Inspector AWS CloudTrailAWS Config、Amazon Detec t ive、A mazon Macie GuardDuty、A mazon 和。AWS Security Hub CSPM您还可以使用AWS Cost ExplorerAWS Budgets、和CloudWatch 账单指标进行成本优化。

下表概述了您的日志和监控解决方案应解决的六个方面。

| | |
|---|---|
| 捕获和摄取日志文件和指标 | 识别、配置来自不同来源的系统和应用程序日志和指标，并将其发送到 AWS 服务。 |
| 搜索和分析日志 | 搜索和分析日志，用于操作管理、问题识别、故障排除和应用程序分析。 |
| 监控指标和警报 | 识别工作负载中的观察结果和趋势并采取行动。 |
| 监控应用程序和服务可用性 | 通过持续监控服务可用性，减少停机时间并提高实现服务级别目标的能力。 |
| 跟踪应用程序 | 跟踪系统中的应用程序请求和外部依赖关系，以微调性能、执行根本原因分析和解决问题。 |

| | |
|---|---|
| 创建仪表板和可视化效果 | 创建仪表板，重点关注系统和工作负载的相关指标和观察结果，这有助于持续改进和主动发现问题。 |

CloudWatch 可以满足大多数日志和监控要求，并提供可靠、可扩展和灵活的解决方案。除了用于监控和分析的 CloudWatch 日志集成外，许多 AWS 服务还会自动提供 CloudWatch 指标。 CloudWatch 还提供代理和日志驱动程序以支持各种计算选项，例如服务器（云端和本地）、容器和无服务器计算。本指南还涵盖以下用于日志记录和监控的 AWS 服务：

- AWS Systems Manager Di st@@ ri butor、Systems Manager 状态管理器和 Systems Manager Autom ation 用于自动化、配置和更新您的 EC2 实例和本地服务器的 CloudWatch 代理
- 用于高级日志聚合、搜索和分析的 Amazon OpenSearch 服务
- Amazon Route 53 运行状况检查和 S CloudWatchynthetics 用于监控应用程序和服务的可用性
- 适用于 Prometheus 的亚马逊托管服务，用于大规模监控容器化应用程序
- AWS X-Ray用于应用程序跟踪和运行时分析
- Amazon Manag ed Grafana 用于可视化和分析来自多个来源（例如亚马逊服务和 OpenSearch 亚马逊 T CloudWatch imestream）的数据

您选择的 AWS 计算服务还会影响日志和监控解决方案的实施和配置。例如，亚马逊 EC2、CloudWatch亚马逊 ECS、Amazon EKS 和 Lambda 的实现和配置是不同的。

应用程序和工作负载所有者通常会忘记日志和监控，或者对其进行配置和实施不一致。这意味着工作负载进入生产时可观察性有限，这会导致识别问题的延迟，并增加故障排除和解决问题所需的时间。除了应用程序日志和指标的应用程序层之外，您的日志和监控解决方案至少还必须针对操作系统 (OS) 级别的日志和指标的系统层。该指南提供了一种推荐的方法，用于跨不同的计算类型（包括下表中概述的三种计算类型）解决这两个层的问题。

| | |
|---|---|
| 长时间运行且不可变的 EC2 实例 | 跨多个 AWS 区域或账户的多个操作系统 (OSs) 的系统和应用程序日志和指标。 |
| 容器 | 您的 Amazon ECS 和 Amazon EKS 集群的系统和应用程序日志和指标，包括不同配置的示例。 |
| 无服务器 | Lambda 函数的系统和应用程序日志和指标以及自定义注意事项。 |

本指南提供了一种日志和监控解决方案，可解决 CloudWatch 以下领域的相关 AWS 服务：

- 规划您的 CloudWatch 部署— 规划 CloudWatch 部署的注意事项和集中 CloudWatch 配置的指导。
- 为 EC2 实例和本地服务器配置 CloudWatch 代理— 系统级和应用程序级日志记录和指标的 CloudWatch 配置详细信息。
- CloudWatch 适用于 Amazon EC2 和本地服务器的代理安装方法— 安装 CloudWatch 代理的方法，包括使用 Systems Manager 在多个区域和账户中自动部署。
- 在 Amazon ECS 上进行日志记录和监控— 在 Amazon CloudWatch ECS 中配置集群级和应用程序级日志记录和指标的指南。
- Amazon EKS 中的日志记录和监控— 在 Amazon CloudWatch EKS 中配置集群级和应用程序级日志记录和指标的指南。
- Prometheus 在亚马逊 EKS 上进行监控— 介绍并比较了适用于 Prometheus 的亚马逊托管服务与 Prometheus 的 Container Insights 监 CloudWatch 控。
- 的日志和指标 AWS Lambda— 为您的 Lambda 函数 CloudWatch 进行配置的指南。
- 搜索和分析日志 CloudWatch— 使用 Amazon A CloudWatch pplication Insights、Lo CloudWatch gs Insights 分析日志以及将日志分析扩展到亚马逊 OpenSearch 服务的方法。
- 带有的警报选项 CloudWatch— 介绍 CloudWatch 警报和 CloudWatch 异常检测，并提供警报创建和设置指南。
- 监控应用程序和服务可用性— 介绍并比较了 S CloudWatch ynthetics 和 Route 53 运行状况检查，以实现自动可用性监控。
- 使用跟踪应用程序 AWS X-Ray— 使用适用于亚马逊 EC2、亚马逊 ECS、亚马逊 EKS 和 Lambda 的 X-Ray 进行应用程序跟踪的介绍和设置
- 仪表板和可视化效果 CloudWatch— CloudWatch 仪表板简介，用于提高跨 AWS 工作负载的可观察性。
- CloudWatch 与 AWS 服务集成— 说明如何与各种 AWS 服务 CloudWatch 集成。
- 用于生成控制面板和可视化的 Amazon Managed Grafana— 介绍并比较了用于仪表板和可视化的 Amazon Managed Gra CloudWatch fana。

本指南中使用了这些领域的实施示例，也可以从AWS 示例 GitHub 存储库中获得。

# 目标业务成果

创建专为云设计的日志和监控解决方案对于实现 AWS 云计算的六大优势不可或缺。您的日志和监控解决方案应帮助您的IT组织实现业务成果，从而使您的业务流程、业务合作伙伴、员工和客户受益。在实施了与 Well-Architecte AWS d Framework 一致的日志和监控解决方案后，您可以期待以下四个结果：

## 加快运营准备就绪

启用日志和监控解决方案是为生产支持和使用准备工作负载的重要组成部分。如果您过于依赖手动流程，运营就绪很快就会成为瓶颈，还会缩短 IT 投资的价值实现时间 (TTV)。无效的方法还会导致工作负载的可观察性受到限制。这可能会增加长时间停机、客户不满和业务流程失败的风险。

您可以使用本指南的方法对 AWS 云端日志记录和监控进行标准化和自动化。然后，新的工作负载需要最少的手动准备和干预来进行生产日志和监控。这还有助于减少为多个账户和地区的不同工作负载大规模创建日志和监控标准所需的时间和步骤。

## 改善卓越运营

本指南提供了多种记录和监控最佳实践，可帮助各种工作负载实现业务目标和卓越运营。本指南还提供了详细的示例和开源、可重复使用的模板，您可以将这些模板与基础设施即代码 (IaC) 方法结合使用，使用服务实现架构良好的日志记录和监控解决方案。 AWS 改善卓越运营是反复的，需要持续改进。该指南提供了有关如何持续改进日志记录和监控实践的建议。

## 提高运营可见性

您的业务流程和应用程序可能由不同的 IT 资源支持，并托管在不同的计算类型上，无论是在本地还是在 AWS 云端。由于日志和监控策略的实施不一致和不完整，您的运营可见性可能会受到限制。采用全面的日志记录和监控方法可以帮助您快速识别、诊断和响应工作负载中的问题。本指南可帮助您设计和实施各种方法，以提高您的完整运营可见性并缩短解决故障的平均时间 (MTTR)。全面的日志记录和监控方法还可以帮助您的组织提高服务质量、增强最终用户体验并满足服务级别协议要求（ SLAs ）。

## 扩大运营规模，降低管理成本

您可以扩展本指南中的日志记录和监控实践，以支持多个区域和账户、短期资源和多个环境。该指南提供了自动执行手动步骤（例如安装和配置代理、监控指标以及在出现问题时通知或采取措施）的方法和

示例。当您的云采用率逐渐成熟和增长，并且您需要在不增加云管理活动或资源的情况下扩展运营能力时，这些方法非常有用。

# 规划您的 CloudWatch 部署

日志和监控解决方案的复杂性和范围取决于多个因素，包括：

- 使用了多少环境、区域和账户，以及这个数字可能如何增加。
- 现有工作负载和架构的种类和类型。
- 必须记录和监控 OSs 的计算类型。
- 是否既有本地位置又有 AWS 基础架构。
- 多个系统和应用程序的聚合和分析需求。
- 防止未经授权泄露日志和指标的安全要求。
- 必须与您的日志和监控解决方案集成以支持操作流程的产品和解决方案。

您必须使用新的或更新的工作负载部署定期检查和更新您的日志记录和监控解决方案。当发现问题时，应识别并应用对日志、监控和警报的更新。然后，可以主动发现这些问题，并在将来加以预防。

您必须确保始终如一地安装和配置用于捕获和摄取日志和指标的软件和服务。成熟的日志和监控方法使用多个 AWS 或独立的软件供应商 (ISV) 服务和解决方案来处理不同的领域（例如安全、性能、网络或分析）。每个域都有自己的部署和配置要求。

我们建议使用 CloudWatch 来捕获和摄取多种计算类型的日志 OSs 和指标。许多 AWS 服务 CloudWatch 用于记录、监控和发布日志和指标，无需进一步配置。 CloudWatch 提供了可以针对不同 OSs 环境进行安装和配置的软件代理。以下各节概述了如何为多个账户、区域和配置部署、安装和配置 CloudWatch 代理：

主题


- CloudWatch 在集中式或分布式账户中使用
- 管理 CloudWatch 代理配置文件


# CloudWatch 在集中式或分布式账户中使用

尽管 CloudWatch 旨在监控一个账户和区域中的 AWS 服务或资源，但您可以使用中央账户来捕获来自多个账户和地区的日志和指标。如果您使用多个账户或区域，则应评估是使用集中账户方法还是使用个人账户来捕获日志和指标。通常，多账户和多区域部署需要采用混合方法，以支持安全、分析、运营和工作负载所有者的需求。

下表提供了选择使用集中式、分布式或混合方法时需要考虑的方面。

| | |
|---|---|
| 账户结构 | 您的组织可能有多个单独的帐户（例如，用于非生产和生产工作负载的帐户），或者在特定环境中为单个应用程序设置数千个帐户。我们建议您在运行工作负载的账户中维护应用程序日志和指标，这样工作负载所有者就可以访问日志和指标。这使他们能够在日志和监控中发挥积极作用。我们还建议您使用单独的日志记录帐户来汇总所有工作负载日志，以进行分析、聚合、趋势和集中操作。单独的日志帐户也可以用于安全、存档和监控以及分析。 |
| 访问要求 | 团队成员（例如工作负载所有者或开发人员）需要访问日志和指标才能进行故障排除和改进。应将日志保存在工作负载的帐户中，以便于访问和故障排除。如果日志和指标保存在与工作负载不同的账户中，则用户可能需要定期在账户之间切换。<br><br>使用集中式帐户可向授权用户提供日志信息，而无需授予工作负载帐户访问权限。这可以简化分析工作负载的访问要求，在这些工作负载中，需要对在多个账户中运行的工作负载进行聚合。集中式日志账户还可以有其他搜索和聚合选项，例如 Amazon S OpenSearch ervice 集群。Amazon S OpenSearch ervice 为您的日志提供精细的访问控制，直至字段级别。当您的敏感或机密数据需要专门的访问权限和权限时，精细的访问控制非常重要。 |
| 操作 | 许多组织都有集中的运营和安全团队或外部组织来提供运营支持，需要访问日志进行监控。集中式日志和监控可以更轻松地识别所有账户和工作负载的趋势、搜索、汇总和执行分析。如果您的组织使用 “你构建，你运行” 的方法 DevOps，那么工作负载所有者需要在其账户中记录和监控信息。除了分布式工作负载所有权外，可能需要采用混合方法来满足中央运营和分析需求。 |
| 环境 | 根据安全要求和账户架构，您可以选择将生产账户的日志和指标托管在中心位置，并将其他环境（例如开发或测试）的日志和指标保存在相同或单独的账户中。这有助于防止生产过程中创建的敏感数据被更广泛的受众访问。 |

CloudWatch 提供了多个选项，可使用 CloudWatch 订阅过滤器实时处理日志。您可以使用订阅过滤器将日志实时流式传输到 AWS 服务，以进行自定义处理、分析和加载到其他系统。如果您采用混合方法，除了集中式账户和区域外，还可以在个人账户和区域中查看日志和指标，这可能特别有用。以下列表提供了可用于此目的的 AWS 服务示例：

- Amazon Data Firehos e — Firehose 提供了一种流媒体解决方案，可根据生成的数据量自动扩展和调整大小。您无需管理 Amazon Kinesis 数据流中的分片数量，无需额外编码即可直接连接到亚马逊简单存储服务 (Amazon S3) OpenSearch 、亚马逊服务或 Amazon Redshift。如果您想将日志集中在这些服务中，Firehose 是一个有效的解决方案。 AWS
- Amazon Kinesis Data Streams — Kinesis Data Streams — 如果你需要与 Firehose 不支持的服务集成并实现额外的处理逻辑，那么Kinesis Data Streams是一个合适的解决方案。您可以在您的账户和区域中创建 Amazon CloudWatch Logs 目标，在中央账户中指定 Kinesis 数据流，并指定一个 AWS Identity and Access Management (IAM) 角色，该角色授予其在流中放置记录的权限。Kinesis Data Streams 为您的日志数据提供了一个灵活的开放式着陆区，然后可以通过不同的选项使用这些数据。您可以将 Kinesis Data Streams 日志数据读入您的账户，执行预处理，然后将数据发送到您选择的目的地。

  但是，您必须为流配置分片，使其大小适合生成的日志数据。Kinesis Data Streams 充当日志数据的临时中介或队列，您可以在 Kinesis 流中将数据存储一到 365 天。Kinesis Data Streams 还支持重播功能，这意味着你可以重播未消耗的数据。
- Amazon S OpenSearch ervic e — CloudWatch 日志可以将日志组中的日志流式传输到个人账户或集中账户中的 OpenSearch 集群。当您将日志组配置为将数据流式传输到 OpenSearch 集群时，将在与您的日志组相同的账户和区域中创建 Lambda 函数。Lambda 函数必须与集群建立网络连接。OpenSearch 您可以自定义 Lambda 函数以执行额外的预处理，此外还可以对 Amazon Service 进行自定义提取。 OpenSearch 使用 Amazon S OpenSearch ervice 进行集中日志记录可以更轻松地分析、搜索和解决云架构中多个组件的问题。
- Lambda — 如果您使用 Kinesis Data Streams，则需要预配置和管理使用流中数据的计算资源。为避免这种情况，您可以将日志数据直接流式传输到 Lambda 进行处理，然后根据您的逻辑将其发送到目的地。这意味着您无需预置和管理计算资源即可处理传入的数据。如果您选择使用 Lambda，请确保您的解决方案与 Lambda 配额兼容。

您可能需要以文件格式处理或共享存储在 CloudWatch 日志中的日志数据。您可以创建导出任务，将特定日期或时间范围内的日志组导出到 Amazon S3。例如，您可以选择每天将日志导出到 Amazon S3 以进行分析和审计。Lambda 可用于自动执行此解决方案。您还可以将此解决方案与 Amazon S3 复制相结合，将您的日志从多个账户和区域传送并集中到一个集中账户和区域。

CloudWatch 代理配置还可以在该部分中指定一个`credentials`字agent段。这指定了向其他账户发送指标和日志时要使用的 IAM 角色。如果指定，则此字段包含`role_arn`参数。当您只需要在特定的集中式账户和地区进行集中记录和监控时，可以使用此字段。

您还可以使用 AWS SDK 以自己选择的语言编写自己的自定义处理应用程序，读取账户中的日志和指标，并将数据发送到中央账户或其他目标以进行进一步处理和监控。

# 管理 CloudWatch 代理配置文件

我们建议您创建标准的亚马逊 CloudWatch 代理配置，其中包括您要在所有亚马逊弹性计算云 (Amazon EC2) 实例和本地服务器上捕获的系统日志和指标。您可以使用 CloudWatch 代理配置文件向导来帮助您创建配置文件。您可以多次运行配置向导，为不同的系统和环境生成唯一的配置。您也可以使用配置文件架构修改配置文件或创建变体。 CloudWatch 代理配置文件可以存储在 AWS Systems Manager Parameter Store 参数中。 如果您有多个 CloudWatch 代理配置文件，则可以创建单独的参数存储参数。如果您使用多个 AWS 账户或 AWS 区域，则必须管理和更新每个账户和区域中的参数存储参数。或者，您可以在 Amazon S3 中将 CloudWatch 配置作为文件或您选择的版本控制工具进行集中管理。

CloudWatch 代理附带的`amazon-cloudwatch-agent-ctl`脚本允许您指定配置文件、参数存储参数或代理的默认配置。默认配置与基本的预定义指标集保持一致，并将代理配置为向其报告内存和磁盘空间指标。 CloudWatch但是，它不包括任何日志文件配置。如果您对 CloudWatch 代理使用 S ystems Manager 快速设置，则也会应用默认配置。

由于默认配置不包括日志记录，也不是根据您的要求进行自定义的，因此我们建议您创建和应用自己的 CloudWatch 配置，并根据您的要求进行自定义。

## 管理 CloudWatch 配置

默认情况下， CloudWatch 配置可以作为参数存储参数或 CloudWatch 配置文件进行存储和应用。最佳选择将取决于您的要求。在本节中，我们将讨论这两个选项的优缺点。还详细介绍了用于管理多个 AWS 账户和 AWS 区域的 CloudWatch 配置文件的代表性解决方案。

Systems Manager 参数存储区参数

如果您想在一小部分 AWS 账户和区域中应用和管理 CloudWatch 单个标准 CloudWatch 代理配置文件，则使用 Parameter Store 参数管理配置效果很好。当您将 CloudWatch 配置存储为 Parameter Store 参数时，您可以使用 CloudWatch 代理配置工具（`amazon-cloudwatch-agent-ctl`在 Linux 上）从 Parameter Store 读取和应用配置，而无需将配置文件复制到您的实例。您可以使用 AmazonCloudWatch- S ManageAgent ystems Manager 命令文档在一次运行中更新多个 EC2 实例的 CloudWatch 配置。由于参数存储参数是区域性的，因此您必须在每个 AWS 区域和 AWS 账户中更新

和维护您的 CloudWatch参数存储参数。如果您要将多个 CloudWatch配置应用于每个实例，则必须自定义 AmazonCloudWatch- C ManageAgent ommand 文档以包含这些参数。

CloudWatch 配置文件

如果您有许多 AWS 账户和区域，并且要管理多个 CloudWatch 配置文件，那么将您的 CloudWatch 配置作为文件进行管理可能效果很好。使用这种方法，您可以在文件夹结构中浏览、组织和管理它们。 您可以对单个文件夹或文件应用安全规则，以限制和授予访问权限，例如更新和读取权限。您可以将它们共享并转移到 AWS 以外进行协作。您可以对文件进行版本控制以跟踪和管理更改。您可以通过将 CloudWatch 配置文件复制到 CloudWatch 代理配置目录来集中应用配置，而不必单独应用每个配置文件。对于 Linux， CloudWatch 配置目录位于中`/opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d`。对于 Windows，配置目录位于以下网址`C:\ProgramData\Amazon\AmazonCloudWatchAgent\Configs`。

启动代理时， CloudWatch 代理会自动附加在这些目录中找到的每个文件以创建 CloudWatch 复合配置文件。配置文件应存储在中央位置（例如，S3 存储桶），您的所需账户和区域可以访问该位置。提供了使用这种方法的示例解决方案。

组织 CloudWatch 配置

无论使用哪种方法来管理您的 CloudWatch 配置，都要整理您的 CloudWatch 配置。您可以使用以下方法将配置组织到文件或参数存储路径中。

| | |
|---|---|
| /config/standard/windows/ec2 | 存储适用于亚马逊 EC2 的 Windows 专用标准 CloudWatch 配置文件。您可以在此文件夹下进一步对不同 Windows 版本、EC2 实例类型和环境的标准操作系统 (OS) 配置进行分类。 |
| /config/standard/windows/onpremises | 为本地服务器存储特定于 Windows 的标准 CloudWatch 配置文件。您还可以在此文件夹下对不同 Windows 版本、服务器类型和环境的标准操作系统配置进行进一步分类。 |
| /config/standard/linux/ec2 | 存储适用于亚马逊 EC2 的标准 Linux 专用 CloudWatch 配置文件。您可以在此文件夹下进一步对不同 Linux 发行版、EC2 实例类型和环境的标准操作系统配置进行分类。 |
| /config/standard/linux/onpremises | 存储本地服务器的标准 Linux 专用 CloudWatch 配置文件。您可以在此文件夹下进一步对不同 |

| | |
|---|---|
| | Linux 发行版、服务器类型和环境的标准操作系统配置进行分类。 |
| /config/ecs | 如果您使用亚马逊 ECS 容器实例，请存储特定于亚马逊弹性容器服务 (Amazon ECS) 的 CloudWatch 配置文件。这些配置可以附加到标准的 Amazon EC2 配置中，用于特定于 Amazon ECS 的系统级日志记录和监控。 |
| /config/ <application_name> | 存储应用程序特定的 CloudWatch 配置文件。您可以使用环境和版本的其他文件夹和前缀对应用程序进行进一步分类。 |

## 示例：将 CloudWatch 配置文件存储在 S3 存储桶中

本节提供了一个使用 Amazon S3 存储 CloudWatch 配置文件的示例，以及使用自定义 Systems Manager 运行手册来检索和应用 CloudWatch 配置文件。这种方法可以解决使用 Systems Manager 参数存储参数进行大规模 CloudWatch 配置的一些挑战：

- 如果您使用多个区域，则必须在每个区域的参数存储中同步 CloudWatch 配置更新。Parameter Store 是一项区域服务，在使用 CloudWatch 代理的每个区域中必须更新相同的参数。
- 如果您有多个 CloudWatch 配置，则必须启动每个参数存储配置的检索和应用。您必须从 Parameter Store 中单独检索每个 CloudWatch 配置，并在添加新配置时更新检索方法。相比之下，CloudWatch 提供了用于存储配置文件的配置目录，并应用该目录中的每个配置，而无需单独指定它们。
- 如果您使用多个账户，则必须确保每个新账户在其 Parameter Store 中都具有所需的 CloudWatch 配置。您还需要确保将来对这些账户及其区域进行任何配置更改。

您可以将 CloudWatch 配置存储在 S3 存储桶中，您的所有账户和区域均可访问该存储桶。然后，您可以使用 Systems Manager Automation 运行手册和 Systems Manager 状态管理器将这些 CloudWatch 配置从 S3 存储桶复制到配置目录。您可以使用 cloudwatch-config-s3 bucket.yaml AW CloudFormation S 模板创建 S3 存储桶，该存储桶可从 AWS Organizations 中一个组织内的多个账户进行访问。该模板包含一个`OrganizationID`参数，该参数授予组织内所有账户的读取权限。

本指南的 “为 CloudWatch 代理部署和配置设置状态管理器和分发服务器” 部分提供的增强示例 Systems Manager 运行手册配置为使用 cloudwatch-config-s3-bucket.y aml AWS 模板创建的 S3 存储桶检索文件。 CloudFormation

或者，您可以使用版本控制系统（例如 GitHub）来存储配置文件。如果要自动检索存储在版本控制系统中的配置文件，则必须管理或集中凭据存储，并更新 Systems Manager Automation 运行手册，该操作手册用于检索您的账户和的凭据。 AWS 区域

# 为 EC2 实例和本地服务器配置 CloudWatch 代理

许多组织在物理服务器和虚拟机 (VM) 上运行工作负载。这些工作负载通常在不同的操作系统上运行，每个操作系统在捕获和摄取指标方面都有独特的安装和配置要求。

如果您选择使用 EC2 实例，则可以高度控制您的实例和操作系统配置。但是，这种更高的控制和责任级别要求您监控和调整配置，以实现更高的使用效率。您可以通过建立日志和监控标准以及应用标准的安装和配置方法来捕获和摄取日志和指标，从而提高运营效率。

将 IT 投资迁移或扩展到 AWS 云端的组织可以利用它 CloudWatch 来实现统一的日志和监控解决方案。 CloudWatch 定价意味着您需要为要捕获的指标和日志按增量付费。您还可以使用与 Amazon EC2 类似的 CloudWatch 代理安装过程来捕获本地服务器的日志和指标。

在开始安装和部署之前 CloudWatch，请务必评估系统和应用程序的日志和指标配置。确保为要使用的操作系统定义需要捕获的标准日志和指标。系统日志和指标是日志和监控解决方案的基础和标准，因为它们是由操作系统生成的，对于 Linux 和 Windows 来说是不同的。除了特定于 Linux 版本或发行版的指标和日志文件外，Linux 发行版中还提供了重要的指标和日志文件。不同的 Windows 版本之间也会出现这种差异。

## 配置代 CloudWatch 理

CloudWatch 使用特定于每个操作系统的代理和CloudWatch 代理配置文件捕获 Amazon EC2 和本地服务器的指标和日志。我们建议您在开始在账户中大规模安装 CloudWatch 代理之前，先定义组织的标准指标和日志捕获配置。

您可以组合多个 CloudWatch 代理配置以形成复合 CloudWatch 代理配置。一种推荐的方法是在系统和应用程序级别定义和划分日志和指标的配置。下图说明了如何将满足不同要求的多种 CloudWatch配置文件类型组合成复合 CloudWatch配置：

还可以根据特定的环境或要求对这些日志和指标进行进一步分类和配置。例如，您可以为不受监管的开发环境定义精度较低的较小日志和指标子集，为受监管的生产环境定义更大、更完整且精度更高的日志和指标集。

## 为 EC2 实例配置日志捕获

默认情况下，Amazon EC2 不监控或捕获日志文件。相反，日志文件由安装在您的 EC2 实例、AWS API 或 AWS Command Line Interface (AWS CLI) 上的 CloudWatch 代理软件捕获并提取到

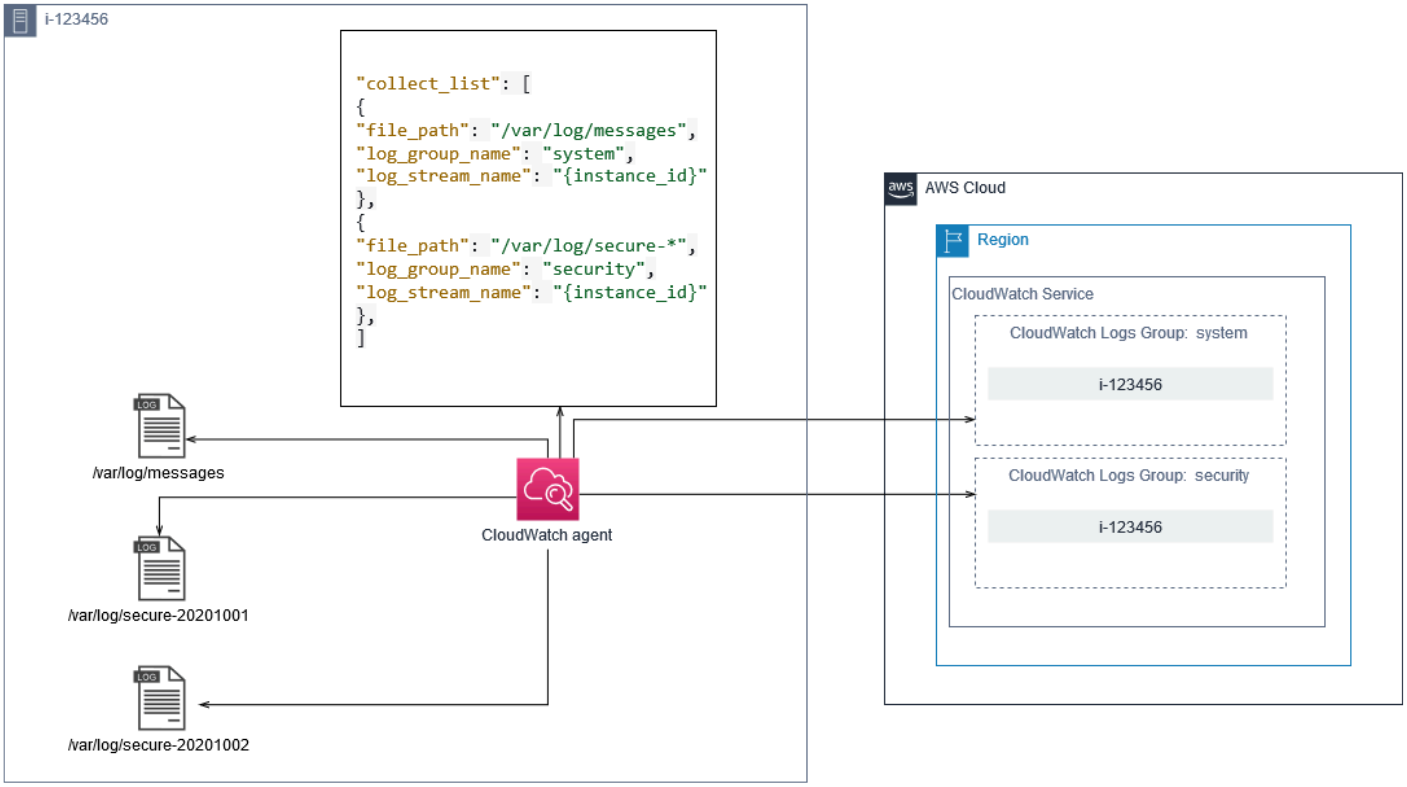
CloudWatch 日志中。我们建议使用 CloudWatch 代理将日志文件提取到 Amazon EC2 和本地服务器的 CloudWatch 日志中。

您可以搜索和筛选日志，也可以提取指标，并根据日志文件中的 CloudWatch模式修补运行自动化。CloudWatch 支持纯文本、空格分隔以及 JSON-formatted 筛选和模式语法选项， JSON-formatted 日志提供了最大的灵活性。要增加筛选和分析选项，应使用格式化的日志输出而不是纯文本。

CloudWatch 代理使用配置文件来定义要发送到的日志和指标 CloudWatch。 CloudWatch 然后将每个日志文件捕获为日志流，并将这些日志流分组到一个日志组中。这可以帮助您对 EC2 实例中的日志执行操作，例如搜索匹配的字符串。

默认日志流名称与 EC2 实例 ID 相同，默认日志组名称与日志文件路径相同。日志流的名称在 CloudWatch 日志组中必须是唯一的。您可以在日志流和日志组名称中使用`instance_id`、`hostname`、`local_hostname`或`ip_address`进行动态替换，这意味着您可以在多个 EC2 实例中使用相同的 CloudWatch 代理配置文件。

下图显示了用于捕获日志的 CloudWatch 代理配置。日志组由捕获的日志文件定义，并且包含每个 EC2 实例的单独日志流，因为`{instance_id}`变量用于日志流名称且 EC2 实例 ID 是唯一的。




日志组定义其所包含的日志流的保留期、标签、安全性、指标筛选器和搜索范围。基于日志文件名的默认分组行为可帮助您搜索和创建指标，并针对账户和地区的 EC2 实例中特定于日志文件的数据发出警

报。您应该评估是否需要进一步细化日志组。例如，您的账户可能由多个业务部门共享，并且拥有不同的技术或运营所有者。这意味着您必须进一步细化日志组名称以反映分离和所有权。这种方法使您可以将分析和故障排除重点放在相关的 EC2 实例上。

如果多个环境使用一个帐户，则可以将每个环境中运行的工作负载的日志记录分开。下表显示了包括业务部门、项目或应用程序以及环境在内的日志组命名惯例。

| | |
|---|---|
| 日志组名称 | `/<Business unit>/<Project or application name>/<Environment>/<Log file name>` |
| 日志流名称 | `<EC2 instance ID>` |

您也可以将 EC2 实例的所有日志文件分组到同一个日志组中。这使得在一组日志文件中搜索和分析单个 EC2 实例变得更加容易。如果您的大多数 EC2 实例都为一个应用程序或工作负载提供服务，并且每个 EC2 实例都用于特定用途，则此功能非常有用。下表显示了如何格式化您的日志组和日志流命名以支持这种方法。

| | |
|---|---|
| 日志组名称 | `/<Business unit>/<Project or application name>/<Environment>/<EC2 instance ID>` |
| 日志流名称 | `<Log file name>` |

## 为 EC2 实例配置指标捕获

默认情况下，您的 EC2 实例启用基本监控，并且 CloudWatch 每五分钟自动向其发送一组标准指标（例如 CPU、网络或存储相关指标）。 CloudWatch 指标可能因实例系列而异，例如，突发性能实例具有 CPU 积分指标。Amazon EC2 标准指标包含在您的实例价格中。如果您为 EC2 实例启用详细监控，则可以在一分钟内接收数据。周期频率会影响您的 CloudWatch成本，因此请务必评估您的所有 EC2 实例还是仅需要对部分 EC2 实例进行详细监控。例如，您可以对生产工作负载启用详细监控，但对非生产工作负载使用基本监控。

On-premises 服务器不包含任何默认指标 CloudWatch ，必须使用 CloudWatch 代理或 AWS SDK 来捕获指标。 AWS CLI这意味着您必须在 CloudWatch 配置文件中定义要捕获的指标（例如 CPU

利用率）。您可以创建包含本地服务器标准 EC2 实例指标的唯一 CloudWatch 配置文件，并在标准 CloudWatch 配置之外应用该文件。

中的@@ CloudWatch 指标由指标名称和零个或多个维度进行唯一定义，并且在指标命名空间中进行唯一分组。 AWS 服务提供的指标的命名空间以AWS（例如AWS/EC2）开头，非AWS 指标被视为自定义指标。您使用 CloudWatch 代理配置和捕获的指标都被视为自定义指标。由于创建的指标数量会影响您的 CloudWatch 成本，因此您应评估所有还是仅部分EC2实例需要每个指标。例如，您可以为生产工作负载定义一整套指标，但对非生产工作负载使用这些指标的较小子集。

CWAgent是 CloudWatch 代理发布的指标的默认命名空间。与日志组类似，指标命名空间组织了一组指标，以便可以在一个地方找到它们。您应该修改命名空间以反映业务部门、项目或应用程序以及环境（例如/<Business unit>/<Project or application name>/<Environment>）。如果多个不相关的工作负载使用同一个帐户，则此方法很有用。您也可以将命名空间命名约定与 CloudWatch 日志组命名约定相关联。

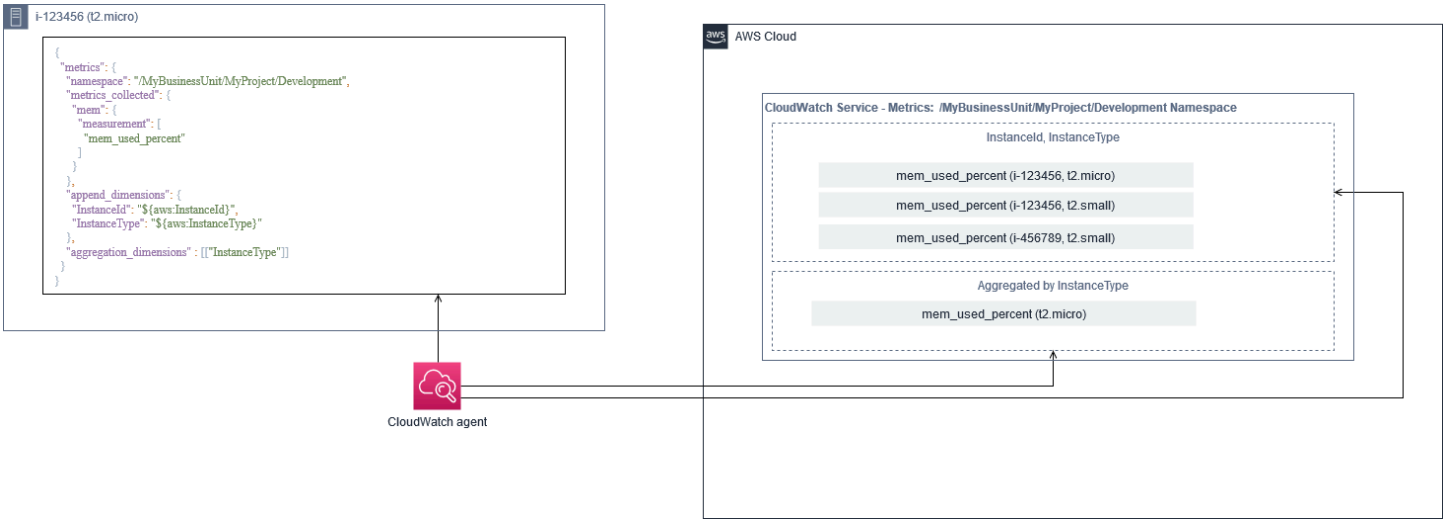
指标还通过其维度进行标识，这些维度可以帮助您根据一组条件对其进行分析，并且是记录观测值所依据的属性。Amazon EC2 包含针对带有InstanceId和AutoScalingGroupName维度的 EC2 实例的单独指标。如果您启用详细监控，则还会收到带有ImageId和InstanceType维度的指标。例如，除了为该维度提供单独的 CPU 利用率指标外，Amazon EC2 还为该InstanceIdInstanceType维度提供了单独的 EC2 实例使用率指标。这可以帮助您分析每个唯一 EC2 实例的 CPU 使用率，以及特定实例类型的所有 EC2 实例。

添加更多维度可以提高您的分析能力，但也会增加总体成本，因为每个指标和唯一的维度值组合都会生成一个新的指标。例如，如果您针对该InstanceId维度创建内存利用率百分比的指标，则这是每个 EC2 实例的新指标。如果您的组织运行数千个 EC2 实例，则会导致成千上万的指标并导致更高的成本。要控制和预测成本，请务必确定指标的基数以及哪些维度增加的价值最大。例如，您可以为生产工作负载指标定义一组完整的维度，但为非生产工作负载定义一小部分维度。

您可以使用该append_dimensions属性向 CloudWatch 配置中定义的一个或所有指标添加维度。您也可以动态地将ImageId、InstanceId、和InstanceType，附加AutoScalingGroupName到 CloudWatch 配置中的所有指标。或者，您可以通过使用特定指标的append_dimensions属性为该指标附加任意维度的名称和值。 CloudWatch 还可以聚合您使用aggregation_dimensions属性定义的指标维度的统计数据。

例如，您可以根据该InstanceType维度汇总所使用的内存，以查看所有 EC2 实例对每种实例类型使用的平均内存。如果您使用在某个区域中运行的t2.micro实例，则可以确定使用该t2.micro类的工作负载是过度利用还是未充分利用所提供的内存。未充分利用可能是工作负载使用具有不需要内存容量的 EC2 类的迹象。相比之下，过度利用可能是工作负载使用内存不足的 Amazon EC2 类的迹象。

下图显示了使用自定义命名空间、添加的维度和聚合的示例 CloudWatch 指标配置InstanceType。




# System-level CloudWatch 配置

Systems-level 指标和日志是监控和日志解决方案的核心组件，该 CloudWatch 代理具有适用于 Windows 和 Linux 的特定配置选项。

我们建议您使用CloudWatch 配置文件向导或配置文件架构为计划支持的每个操作系统定义 CloudWatch 代理配置文件。其他特定于工作负载的 OS-level日志和指标可以在单独的 CloudWatch 配置文件中定义，并附加到标准配置中。这些独特的配置文件应单独存储在 S3 存储桶中，供您的 EC2 实例检索。本指南的管理 CloudWatch 配置部分描述了用于此目的的 S3 存储桶设置示例。您可以使用状态管理器和分发服务器自动检索和应用这些配置。

## 配置系统级日志

System-level 日志对于诊断和解决本地或 AWS 云端问题至关重要。您的日志捕获方法应包括操作系统生成的任何系统和安全日志。根据操作系统版本， OS-generated 日志文件可能会有所不同。

该 CloudWatch 代理支持通过提供事件日志名称来监视 Windows 事件日志。您可以选择要监控的 Windows 事件日志（例如`System`、`Application`、或`Security`）。

Linux 系统的系统、应用程序和安全日志通常存储在`/var/log`目录中。下表定义了您应监视的常见默认日志文件，但您应检查`/etc/rsyslog.conf`或`/etc/syslog.conf`文件以确定系统日志文件的特定设置。

| | |
|---|---|
| Fedora 发行版 | `/var/log/boot.log*` — 启动日志 |
| （亚马逊 Linux、CentOS、红帽企业 Linux） | `/var/log/dmesg` — 内核日志 |

| | | |
|---|---|---|
| | | /var/log/secure — 安全和身份验证日志 |
| | | /var/log/messages — 一般系统日志 |
| | | /var/log/cron* — Cron 日志 |
| | | /var/log/cloud-init-output.log — Userdata 启动脚本的输出 |
| Debian （Ubuntu） | | /var/log/syslog — 启动日志 |
| | | /var/log/cloud-init-output.log — Userdata 启动脚本的输出 |
| | | /var/log/auth.log — 安全和身份验证日志 |
| | | /var/log/kern.log — 内核日志 |

您的组织可能还有其他代理或系统组件来生成您要监控的日志。您应该评估并决定哪些日志文件是由这些代理或应用程序生成的，并通过确定其文件位置将它们包含在配置中。例如，您应该在配置中包括 Systems Manager 和 CloudWatch 代理日志。下表提供了这些适用于 Windows 和 Linux 的代理日志的位置。

| | | |
|---|---|---|
| Windows | CloudWatch 代理人 | $Env:ProgramData\Amazon\AmazonCloudWatchAgent\Logs\amazon-cloudwatch-agent.log |
| | Systems Manager 代理 | %PROGRAMDATA%\Amazon\SSM\Logs\amazon-ssm-agent.log<br><br>%PROGRAMDATA%\Amazon\SSM\Logs\errors.log<br><br>%PROGRAMDATA%\Amazon\SSM\Logs\audits |

| | | |
|---|---|---|
| | | `\amazon-ssm-agent-audit-YYYY-MM-DD` |
| Linux | CloudWatch 代理人 | `/opt/aws/amazon-cloudwatch-agent/logs/amazon-cloudwatch-agent.log` |
| | Systems Manager 代理 | `/var/log/amazon/ssm/amazon-ssm-agent.log`<br>`/var/log/amazon/ssm/errors.log`<br>`/var/log/amazon/ssm/audits/amazon-ssm-agent-audit-YYYY-MM-DD` |

CloudWatch 如果日志文件是在 CloudWatch 代理配置中定义的，但未找到，则忽略该文件。当您想为 Linux 维护单个日志配置，而不是为每个发行版维护单独的配置时，这很有用。当代理或软件应用程序开始运行之前日志文件不存在时，它也很有用。

## 配置系统级指标

Amazon EC2 提供的标准指标中不包括内存和磁盘空间利用率。要包含这些指标，您必须在 EC2 实例上安装和配置 CloudWatch 代理。 CloudWatch 代理配置向导使用预定义的指标创建 CloudWatch 配置，您可以根据需要添加或删除指标。请务必查看预定义的指标集，以确定所需的相应级别。

最终用户和工作负载所有者应根据服务器或 EC2 实例的特定要求发布其他系统指标。这些指标定义应在单独的 CloudWatch 代理配置文件中存储、版本控制和维护，并在中心位置（例如 Amazon S3）共享，以便重复使用和自动化。

本地服务器不会自动捕获标准 Amazon EC2 指标。这些指标必须在本地实例使用的 CloudWatch 代理配置文件中定义。您可以为本地实例创建单独的指标配置文件，其中包含诸如 CPU 利用率之类的指标，并将这些指标附加到标准指标配置文件中。

# Application-level CloudWatch 配置

应用程序日志和指标由运行的应用程序生成，并且是特定于应用程序的。请务必定义必要的日志和指标，以充分监控组织经常使用的应用程序。例如，您的组织可能已将基于 Web 的应用程序的 Microsoft 互联网信息服务器 (IIS) 标准化。您可以为 IIS 创建标准日志和指标 CloudWatch 配置，该配置也可以在整个组织中使用。 Application-specific 配置文件可以存储在集中位置（例如 S3 存储桶），工作负载所有者或通过自动检索进行访问，然后复制到 CloudWatch 配置目录中。 CloudWatch 代理会自动将 CloudWatch每个 EC2 实例或服务器的配置文件目录中的配置文件合并成一个复合 CloudWatch 配置。最终结果是 CloudWatch 包括贵组织的标准系统级配置以及所有相关的应用程序 CloudWatch 级配置的配置。

工作负载所有者应识别和配置所有关键应用程序和组件的日志文件和指标。

## 配置应用程序级日志

Application-level 根据应用程序是商用现成的 (COTS) 应用程序还是定制开发的应用程序，日志记录会有所不同。COTS 应用程序及其组件可能会为日志配置和输出提供多种选项，例如日志详细信息级别、日志文件格式和日志文件位置。但是，大多数 COTS 或第三方应用程序不允许您从根本上更改日志记录（例如，更新应用程序代码以包含其他不可配置的日志语句或格式）。您至少应为 COTS 或第三方应用程序配置日志选项，以记录警告和错误级别的信息，最好采用 JSON 格式。

您可以通过在 CloudWatch 配置中包含应用程序的 CloudWatch 日志文件来将自定义开发的应用程序与日志集成。自定义应用程序可以提供更好的日志质量和控制，因为除了包括任何其他所需的详细信息外，您还可以自定义日志输出格式，对组件输出进行分类和分隔以单独的日志文件。请务必审查日志库以及组织所需的数据和格式，并对其进行标准化，以便更轻松地进行分析和处理。

您也可以使用 Log CloudWatch s `PutLogEvents` API 调用或使用 AWS SDK 写入 CloudWatch 日志流。您可以使用 API 或 SDK 来满足自定义日志要求，例如将日志记录协调到分布式组件和服务器集合中的单个日志流。但是，最容易维护且最广泛适用的解决方案是将应用程序配置为写入日志文件，然后使用 CloudWatch 代理读取日志文件并将其流式传输到 CloudWatch。

您还应该考虑要从应用程序日志文件中衡量的指标类型。您可以使用指标筛选器对 CloudWatch 日志组中的这些数据进行测量、绘制图表并发出警报。例如，您可以使用指标筛选器通过在日志中识别失败的登录尝试来统计失败的登录尝试。

您还可以使用应用程序日志文件中的CloudWatch 嵌入式指标格式，为自定义开发的应用程序创建自定义指标。

# 配置应用程序级指标

自定义指标不是由 AWS 服务直接提供给的指标 CloudWatch ，它们发布在 CloudWatch 指标的自定义命名空间中。所有应用程序指标均被视为自定义 CloudWatch 指标。应用程序指标可能与 EC2 实例、应用程序组件、API 调用甚至业务函数保持一致。您还必须考虑为指标选择的维度的重要性和基数。具有高基数的维度会生成大量自定义指标，并且可能会增加您的 CloudWatch 成本。

CloudWatch 帮助您以多种方式捕获应用程序级指标，包括：

- 通过定义要从 procstat 插件中捕获的各个进程，捕获流程级别的指标。
- 应用程序将指标发布到 Windows 性能监视器，该指标在 CloudWatch 配置中定义。
- 指标筛选器和模式适用于应用程序的登录 CloudWatch。
- 应用程序使用 CloudWatch 嵌入式指标格式写入 CloudWatch 日志。
- 应用程序 CloudWatch 通过 API 或 AWS SDK 向发送指标。
- 应用程序向带有已配置代理的 col lect d 或 StatsD 守护程序发送指标。 CloudWatch

您可以使用 procstat 通过 CloudWatch代理监控和测量关键应用程序进程。这可以帮助您在应用程序的关键进程不再运行时发出警报并采取措施（例如，通知或重启进程）。您还可以测量应用程序进程的性能特征，并在特定进程行为异常时发出警报。

如果您无法使用其他自定义指标更新 COTS 应用程序，Procstat 监控也很有用。例如，您可以创建一个`my_process`衡量`cpu_time`并包含自定义`application_version`维度的指标。如果您对不同的指标有不同的维度，也可以为一个应用程序使用多个 CloudWatch 代理配置文件。

如果你的应用程序在 Windows 上运行，你应该评估它是否已经向 Windows 性能监视器发布了指标。许多 COTS 应用程序都与 Windows 性能监视器集成，可帮助您轻松监控应用程序指标。 CloudWatch 还与 Windows 性能监视器集成，您可以捕获其中已有的指标。

请务必查看应用程序提供的日志格式和日志信息，以确定可以使用指标筛选器提取哪些指标。您可以查看应用程序的历史日志，以确定如何表示错误消息和异常关机。您还应该查看之前报告的问题，以确定是否可以捕获某个指标以防止问题再次出现。您还应该查看应用程序的文档，并要求应用程序开发人员确认如何识别错误消息。

对于自定义开发的应用程序，请与应用程序的开发人员合作，定义可使用 CloudWatch 嵌入式指标格式、 AWS SDK 或 AWS API 实现的重要指标。推荐的方法是使用嵌入式指标格式。您可以使用 AWS 提供的开源嵌入式指标格式库来帮助您按照所需格式编写报表。您还需要更新特定于应用程序的 CloudWatch 配置，以包含嵌入式指标格式代理。这会导致在 EC2 实例上运行的代理充当本地嵌入式指标格式终端节点，向其发送嵌入式指标格式指标 CloudWatch。

如果您的应用程序已经支持将指标发布到 collectd 或 statsd，则可以利用它们将指标提取到。CloudWatch

# CloudWatch 适用于 Amazon EC2 和本地服务器的代理安装方法

自动化 CloudWatch 代理的安装过程有助于您快速一致地部署代理并捕获所需的日志和指标。有多种方法可以自动安装 CloudWatch 代理，包括多账户和多区域支持。讨论了以下自动安装方法：

- 使用 Systems Manager Distributor 和 Systems Manager 状态管理器安装 CloudWatch 代理 — 如果您的 EC2 实例和本地服务器正在运行 Systems Manager 代理，我们建议您使用这种方法。这样可以确保 CloudWatch 代理保持更新，并且您可以报告和修复没有 CloudWatch 代理的服务器。这种方法还可以扩展以支持多个账户和区域。
- 在 EC2 实例预配置期间将 CloudWatch 代理作为用户数据脚本的一部分进行部署 — Amazon EC2 允许您定义在首次启动或重启时运行的启动脚本。您可以定义脚本来自动执行代理的下载和安装过程。这也可以包含在 CloudFormation 脚本和 S AWS ervice Catalog 产品中。如果针对与您的标准不同的特定工作负载采用定制的代理安装和配置方法，则这种方法可能适合根据需要使用。
- 在 Amazon 系统映像中包含 CloudWatch代理 (AMIs) — 您可以将 CloudWatch代理安装到适用于 Amazon EC2 的自定义 AMI 中。使用 AMI 的 EC2 实例将自动安装并启动代理。但是，您必须确保代理及其配置定期更新。

## 使用 Systems Manager 分发服务器和状态管理器安装 CloudWatch 代理

您可以将 Systems Manager 状态管理器与 Systems Manager Distributor 配合使用，在服务器和 EC2 实例上自动安装和更新 CloudWatch 代理。Distributor 包括安装最新 CloudWatch代理版本的`AmazonCloudWatchAgent` AWS 托管软件包。

这种安装方法具有以下先决条件：

- 必须在您的服务器或 EC2 实例上安装并运行 Systems Manager 代理。Systems Manager 代理已预装在亚马逊 Linux、亚马逊 Linux 2 等上。 AMIs还必须在其他映像或本地和服务器上安装 VMs 和配置代理。

> **Note**
>
> 即将终止对 Amazon Linux 2 的支持。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。

- 必须将具有所需权限 CloudWatch 和 Systems Manager 权限的 IAM 角色或证书附加到 EC2 实例，或者在本地服务器的证书文件中定义。例如，您可以创建一个包含 AWS 托管策略的 IAM 角色：`AmazonSSMManagedInstanceCore`适用于 Systems Manager 和`CloudWatchAgentServerPolicy` CloudWatch。您可以使用 ssm-cloudwatch-instance-role.yaml CloudFormation 模板部署包含这两个策略的 IAM 角色和实例配置文件。也可以修改此模板以包含您的 EC2 实例的其他标准 IAM 权限。对于本地服务器或 VMs，应将 CloudWatch 代理配置为使用为本地服务器配置的 S ystems Manager 服务角色。有关这方面的更多信息，请参阅如何将使用 Systems Manager 代理和统一 CloudWatch 代理的本地服务器配置为仅使用临时证书？ 在 AWS 知识中心中。

以下列表提供了使用 Systems Manager 分销商和状态管理器方法安装和维护 CloudWatch 代理的几个优点：

- 自动安装多个操作系统 OSs-您无需为每个操作系统编写和维护脚本即可下载和安装 CloudWatch代理。
- 自动更新检查 — State Manager 会自动定期检查每个 EC2 实例是否具有最新 CloudWatch版本。
- 合规性报告 — Systems Manager 合规控制面板显示哪些 EC2 实例未能成功安装分销商软件包。
- 自动安装新启动的 EC2 实例 — 在您的账户中启动的新 EC2 实例会自动收到 CloudWatch 代理。

但是，在选择此方法之前，还应考虑以下三个方面：

- 与现有关联发生冲突-如果另一个关联已经安装或配置了 CloudWatch 代理，则这两个关联可能会相互干扰并可能导致问题。使用此方法时，应删除安装或更新 CloudWatch代理和配置的所有现有关联。
- 更新自定义代理配置文件-D istributor 使用默认配置文件执行安装。如果您使用一个或多个自定义 CloudWatch 配置文件，则必须在安装后更新配置。
- 多区域或多账户设置-必须在每个账户和区域中设置州经理关联。必须更新多账户环境中的新账户以包含状态经理关联。您需要集中或同步 CloudWatch 配置，以便多个账户和地区可以检索和应用所需的标准。

## 为 CloudWatch 代理部署和配置设置状态管理器和分发服务器

您可以使用 S ystems Manager 快速设置来快速配置 Systems Manager 的功能，包括在 EC2 实例上自动安装和更新 CloudWatch 代理。快速设置部署了一个 CloudFormation 堆栈，该堆栈可根据您的选择部署和配置 Systems Manager 资源。

以下列表提供了 Quick Setup 为自动安装和更新 CloudWatch 代理而执行的两个重要操作：

1. 创建 Systems Manager 自定义文档 — 快速设置创建以下 Systems Manager 文档以供状态管理器使用。文档名称可能有所不同，但内容保持不变：
   - `CreateAndAttachIAMToInstance`— 如果`AmazonSSMRoleForInstancesQuickSetup`角色和实例配置文件不存在，则创建它们并将`AmazonSSMManagedInstanceCore`策略附加到角色。这不包括所需的 `CloudWatchAgentServerPolicy` IAM 策略。您必须更新此政策并更新本 Systems Manager 文档以包含此政策，如下一节所述。
   - `InstallAndManageCloudWatchDocument`— 使用 Distributor 安装 CloudWatch 代理，并使用 S `AWS-ConfigureAWSPackage` ystems Manager 文档为每个 EC2 实例配置一次默认 CloudWatch 代理配置。
   - `UpdateCloudWatchDocument`— 使用 S CloudWatch `AWS-ConfigureAWSPackage` ystems Manager 文档安装最新 CloudWatch 代理来更新代理。更新或卸载代理不会从 EC2 实例中移除现有的 CloudWatch配置文件。
2. 创建状态管理员关联-创建状态管理员关联并将其配置为使用自定义创建的 Systems Manager 文档。状态管理器关联名称可能有所不同，但配置保持不变：
   - `ManageCloudWatchAgent`— 为每个 EC2 实例运行一次 S `InstallAndManageCloudWatchDocument` ystems Manager 文档。
   - `UpdateCloudWatchAgent`— 每 30 天为每个 EC2 实例运行一次 S `UpdateCloudWatchDocument` ystems Manager 文档。
   - 为每个 EC2 实例运行一次 S `CreateAndAttachIAMToInstance` ystems Manager 文档。

您必须补充和自定义已完成的快速设置配置，以包含 CloudWatch权限并支持自定义 CloudWatch 配置。特别
是，`CreateAndAttachIAMToInstance`和`InstallAndManageCloudWatchDocument`文档需要更新。您可以手动更新通过快速设置创建的 Systems Manager 文档。或者，您可以使用自己的 CloudFormation 模板为相同的资源配置必要的更新，也可以配置和部署其他 Systems Manager 资源，而不必使用快速设置。

> **Important**
>
> 快速设置会创建一个 CloudFormation 堆栈，用于根据您的选择部署和配置 Systems Manager 资源。如果您更新了 “快速设置” 选项，则可能需要手动重新更新 Systems Manager 文档。

以下各节介绍如何手动更新由快速设置创建的 Systems Manager 资源，以及如何使用自己的 CloudFormation 模板执行更新的快速设置。我们建议您使用自己的 CloudFormation 模板，以避免手动更新由 Quick Setup 和 CloudFormation.

## 使用 Systems Manager 快速设置并手动更新已创建的 Systems Manager 资源

必须更新使用快速设置方法创建的 Systems Manager 资源，以包含所需的 CloudWatch 代理权限并支持多个 CloudWatch 配置文件。本节介绍如何更新 IAM 角色和 Systems Manager 文档，以使用包含可从多个账户访问的 CloudWatch 配置的集中式 S3 存储桶。本指南的管理 CloudWatch 配置部分讨论了如何创建 S3 存储桶来存储 CloudWatch 配置文件。

### 更新 S **CreateAndAttachIAMToInstance** ystems Manager 文档

快速设置创建的这份 Systems Manager 文档用于检查 EC2 实例是否附加了现有 IAM 实例配置文件。如果是，则会将`AmazonSSMManagedInstanceCore`策略附加到现有角色。这可以保护您的现有 EC2 实例免于丢失可能通过现有实例配置文件分配的 AWS 权限。您需要在本文档中添加一个步骤，将 `CloudWatchAgentServerPolicy` IAM 策略附加到已附加实例配置文件的 EC2 实例。如果 IAM 角色不存在且 EC2 实例没有附加实例配置文件，则 Systems Manager 文档还会创建该角色。您必须更新文档的这一部分，使其也包含 `CloudWatchAgentServerPolicy` IAM 策略。

查看已完成的 CreateAndAttachIAMToInstance.yaml 示例文档，并将其与快速设置创建的文档进行比较。编辑现有文档以包括所需的步骤和更改。根据您的 “快速设置” 选择，“快速设置” 创建的文档可能与提供的示例文档不同，因此请确保进行所需的调整。示例文档包括 “快速设置” 选项，用于每天扫描实例中是否缺少补丁，因此还包含了 Systems Manager Patch Manager 的策略。

### 更新 S **InstallAndManageCloudWatchDocument** ystems Manager 文档

快速安装创建的这份 Systems Manager 文档安装 CloudWatch 代理并使用默认 CloudWatch 代理配置对其进行配置。默认 CloudWatch 配置与基本的预定义指标集一致。您必须替换默认配置步骤并添加从 CloudWatch 配置 S3 存储桶下载 CloudWatch 配置文件的步骤。

查看已完成的 InstallAndManageCloudWatchDocument.yaml 更新文档，并将其与快速设置创建的文档进行比较。您的 “快速设置” 创建的文档可能有所不同，因此请确保已进行所需的调整。编辑现有文档以包括必要的步骤和更改。

## 使用 CloudFormation 代替快速设置

您可以使用来配置 Systems Manager CloudFormation ，而不是使用快速设置。这种方法允许您根据自己的特定要求自定义 Systems Manager 配置。这种方法还可以避免手动更新由快速设置创建的已配置 Systems Manager 资源以支持自定义 CloudWatch 配置。

快速设置功能还使用 CloudFormation 和创建 CloudFormation 堆栈集来根据您的选择部署和配置 Systems Manager 资源。在使用 CloudFormation 堆栈集之前，您必须创建用于支持跨多个账户或区域部署的 CloudFormation StackSets IAM 角色。Quick Setup 创建了支持多区域或多账户部署所需的角色。 CloudFormation StackSets CloudFormation StackSets 如果要在多个区域中配置和部署 Systems Manager 资源，或者通过单个账户和区域在多个账户中配置和部署 Systems Manager 资源，则必须完成的先决条件。有关这方面的更多信息，请参阅 CloudFormation 文档中的堆栈集操作的先决条件。

查看 AWS-QuickSetup-SSMHost mgmt.yaml CloudFormation 模板以获取自定义的快速设置。

您应该查看 CloudFormation 模板中的资源和功能，并根据自己的要求进行调整。您应该对所使用的 CloudFormation 模板进行版本控制，并逐步测试更改以确认所需的结果。此外，您还应进行云安全审查，以确定是否需要根据贵组织的要求进行任何政策调整。

您应该在单个测试账户和区域中部署 CloudFormation 堆栈，并执行任何必要的测试用例以自定义和确认所需的结果。然后，您可以将部署升级到单个账户中的多个区域，然后部署到多个账户和多个区域。

## 在单个账户和带有 CloudFormation 堆栈的区域中进行自定义快速设置

如果您只使用单个账户和区域，则可以将完整的示例部署为 CloudFormation 堆栈而不是 CloudFormation 堆栈集。但是，如果可能，即使只使用单个账户和区域，我们也建议您使用多账户、多区域堆栈集方法。使用将来 CloudFormation StackSets 可以更轻松地扩展到其他账户和区域。

使用以下步骤将 AWS-QuickSetup-SSMHost mgmt.yaml CloudFormation 模板作为 CloudFormation 堆栈部署到单个账户中，然后： AWS 区域

1. 下载模板并将其签入您的首选版本控制系统（例如， GitHub ）。
2. 根据贵组织的要求自定义默认 CloudFormation 参数值。
3. 自定义状态管理器关联时间表。
4. 使用`InstallAndManageCloudWatchDocument`逻辑 ID 自定义 Systems Manager 文档。确认 S3 存储桶前缀与包含您的 CloudWatch配置的 S3 存储桶的前缀一致。
5. 检索并记录包含您的 CloudWatch 配置的 S3 存储桶的 Amazon 资源名称 (ARN)。有关这方面的更多信息，请参阅本指南的管理 CloudWatch 配置部分。提供了一个示例 cloudwatch-config-

s3-bucket.yaml CloudFormation 模板，其中包含用于为账户提供读取权限的存储桶策略。 AWS Organizations

6. 将自定义的快速设置 CloudFormation 模板部署到与 S3 存储桶相同的账户：
   - 在`CloudWatchConfigBucketARN`参数中，输入 S3 存储桶的 ARN。
   - 根据要为 Systems Manager 启用的功能对参数选项进行调整。

7. 部署一个带或不带 IAM 角色的测试 EC2 实例，以确认 EC2 实例是否可以使用 CloudWatch。

- 应用`AttachIAMToInstance`州经理关联。这是一本配置为按计划运行的 Systems Manager 运行手册。使用运行手册的状态管理器关联不会自动应用于新的 EC2 实例，可以将其配置为按计划运行。有关更多信息，请参阅 Systems Manager 文档中的使用状态管理器运行带有触发器的自动化。
- 确认 EC2 实例已关联所需的 IAM 角色。
- 通过确认 EC2 实例在 Systems Manager 中可见，确认 Systems Manager 代理运行正常。
- 根据您的 S3 存储桶中的 CloudWatch 配置查看 CloudWatch 日志和指标，确认 CloudWatch 代理是否正常运行。

## 在多个地区和多个账户中自定义快速设置 CloudFormation StackSets

如果您使用多个账户和区域，则可以将 AWS-QuickSetup-SSMHost mgmt.yaml CloudFormation 模板部署为堆栈集。在使用堆栈集之前，必须完成CloudFormation StackSet先决条件。要求会有所不同，具体取决于您部署的是具有自我管理权限还是服务管理权限的堆栈集。

我们建议您部署具有服务管理权限的堆栈集，以便新账户自动收到自定义的快速设置。您必须使用管理账户或委派管理员帐户部署服务 AWS Organizations 管理堆栈集。您应使用具有委派管理员权限的用于自动化的集中账户而不是 AWS Organizations 管理帐户部署堆栈集。我们还建议您通过将一个区域中只有一个或少量账户的测试组织单位 (OU) 作为目标来测试堆栈集部署。

1. 完成本指南在单个账户和带有 CloudFormation 堆栈的区域中进行自定义快速设置部分中的步骤 1 到 5。
2. 登录 AWS 管理控制台，打开 CloudFormation 控制台并选择创建 StackSet：
   - 选择 “模板已准备就绪”，然后上传模板文件。上传您根据要求自定义的 CloudFormation 模板。
   - 指定堆栈集的详细信息：
     - 输入堆栈集名称，例如`StackSet-SSM-QuickSetup`。
     - 根据要为 Systems Manager 启用的功能对参数选项进行调整。

- 在CloudWatchConfigBucketARN参数中，输入您的 CloudWatch 配置的 S3 存储桶的 ARN。
- 指定堆栈集选项，选择是将服务管理权限与权限一起 AWS Organizations 使用，还是使用自我管理权限。
  - 如果您选择自我管理权限，请输入AWSCloudFormationStackSetAdministrationRole和AWSCloudFormationStackSetExecutionRoleIAM 角色的详细信息。管理员角色必须存在于账户中，执行角色必须存在于每个目标账户中
- 对于的服务管理权限 AWS Organizations，我们建议您先部署到测试 OU，而不是整个组织。
  - 选择是否要启用自动部署。我们建议您选择 “启用”。对于账号移除行为，推荐设置为删除堆栈。
- 要获得自我管理权限，请输入要设置 IDs 的 AWS 账户的账户。如果您使用自我管理权限，则必须对每个新账户重复此过程。
- 输入您将要使用的区域 CloudWatch 和 Systems Manager。
- 在操作和堆栈实例选项卡中查看堆栈集的状态，确认部署成功。
- 按照本指南在单个账户和带有 CloudFormation 堆栈的区域中进行自定义快速设置部分的步骤 7，测试 Systems Manager 和 CloudWatch 是否在已部署的帐户中正常运行。

## 配置本地服务器的注意事项

本地服务器和虚拟机的 CloudWatch 代理的安装和配置方法与 EC2 实例的方法类似。但是，下表提供了在本地服务器和上安装和配置 CloudWatch 代理时必须评估的注意事项 VMs。

| | |
|---|---|
| 将 CloudWatch 代理指向与 Systems Manager 相同的临时凭证。 | 当您在包含本地服务器的混合环境中设置 Systems Manager 时，可以使用 IAM 角色激活 Systems Manager。您应使用为您的 EC2 实例创建的角色，其中包含CloudWatchAgentServerPolicy 和AmazonSSMManagedInstanceCore 策略。<br><br>这会导致 Systems Manager 代理检索临时凭证并将其写入本地凭证文件。您可以将 CloudWatch 代理配置指向同一个文件。您可以使用配置使用 Systems Manager 代理和统一 CloudWatch 代理的本地服务器中的流程来仅使用 AWS 知识中心中的临时证书。 |

| | |
|---|---|
| | 您还可以通过定义单独的 Systems Manager Automation 运行手册和状态管理器关联，并使用标签来定位本地实例，从而自动执行此过程。在为本地实例创建 S ystems Manager 激活时，应包含一个标识该实例为本地实例的标签。 |
| 考虑使用具有 VPN 或 Direct Connect 访问权限的账户和区域，以及 AWS PrivateLink. | 您可以使用 AWS Direct Connect 或 AWS Virtual Private Network (Site-to-Site VPN) 在本地网络和您的虚拟私有云 (VPC) 之间建立私有连接。 AWS PrivateLink使用接口 VPC 终端节点 CloudWatch 建立与 Logs 的私有连接。如果您有限制，禁止通过公共 Internet 将数据发送到公共服务端点，则此方法非常有用。 |
| 所有指标都必须包含在 CloudWatch配置文件中。 | Amazon EC2 包括标准指标（例如 CPU 利用率），但必须为本地实例定义这些指标。您可以使用单独的平台配置文件为本地服务器定义这些指标，然后将该配置附加到平台的标准 CloudWatch 指标配置中。 |

## 临时 EC2 实例的注意事项

如果 EC2 实例由 Amazon EC2 Auto S caling、Amazon EMR、Amazon EC2 竞价型实例或预配置，则它们是临时的或临时的。 AWS Batch临时 EC2 实例可能会在公共日志组下生成大量 CloudWatch 流，而无需提供有关其运行时来源的额外信息。

如果您使用临时 EC2 实例，请考虑在日志组和日志流名称中添加其他动态上下文信息。例如，您可以包括竞价型实例请求编号、Amazon EMR 集群名称或 Auto Scaling 组名称。对于新启动的 EC2 实例，此信息可能有所不同，您可能需要在运行时对其进行检索和配置。为此，您可以在启动时编写 CloudWatch 代理配置文件，然后重新启动代理以包含更新的配置文件。这允许 CloudWatch使用动态运行时信息传送日志和指标。

在您的临时 EC2 实例终止之前，您还应确保 CloudWatch 代理发送您的指标和日志。 CloudWatch 代理包含一个`flush_interval`参数，可以将其配置为定义刷新日志和指标缓冲区的时间间隔。您可以根据自己的工作负载降低此值，在 EC2 实例终止之前停止 CloudWatch 代理并强制缓冲区刷新。

## 使用自动解决方案部署代 CloudWatch 理

如果您使用自动化解决方案（例如 Ansible 或 Chef），则可以利用它来自动安装和更新 CloudWatch 代理。如果您使用这种方法，则必须评估以下注意事项：

- 验证自动化是否涵盖您支持的操作系统版本 OSs 和操作系统版本。如果自动化脚本不支持贵组织的所有脚本 OSs，则应为不支持的 OSs脚本定义替代解决方案。
- 验证自动化解决方案是否定期检查 CloudWatch 代理更新和升级。您的自动化解决方案应定期检查 CloudWatch 代理是否有更新，或者定期卸载并重新安装代理。您可以使用调度程序或自动化解决方案功能来定期检查和更新代理。
- 验证您是否可以确认代理安装和配置合规性。您的自动化解决方案应使您能够确定系统何时未安装代理或代理何时无法运行。您可以在自动化解决方案中实施通知或警报，以便跟踪失败的安装和配置。

## 在实例置备期间使用用户数据脚本部署 CloudWatch 代理

如果您不打算使用 Systems Manager，而是想有选择地用 CloudWatch于 EC2 实例，则可以使用这种方法。通常，这种方法一次性使用，或者在需要特殊配置时使用。 AWS 提供 CloudWatch 代理的直接链接，可以在启动脚本或用户数据脚本中下载这些链接。代理安装包无需用户交互即可静默运行，这意味着您可以在自动部署中使用它们。如果您使用这种方法，则应评估以下注意事项：

- 用户无法安装代理或配置标准指标的风险增加。用户可以在不包括安装 CloudWatch 代理的必要步骤的情况下配置实例。他们还可能错误配置代理，这可能会导致日志记录和监控不一致。
- 安装脚本必须特定于操作系统，并且适用于不同的操作系统版本。如果您打算同时使用 Windows 和 Linux，则需要单独的脚本。根据发行版，Linux 脚本还应有不同的安装步骤。
- 如果 CloudWatch 代理有新版本，则必须定期使用新版本更新代理。如果您将 Systems Manager 与状态管理器一起使用，则可以自动执行此操作，但也可以将用户数据脚本配置为在实例启动时重新运行。然后，每次重新启动时都会更新并重新安装 CloudWatch 代理。
- 您必须自动检索和应用标准 CloudWatch配置。如果您将 Systems Manager 与状态管理器一起使用，则可以自动执行此操作，但也可以配置用户数据脚本以在启动时检索配置文件并重新启动 CloudWatch 代理。

## 将 CloudWatch 代理包括在你的 AMIs

使用这种方法的好处是，您不必等待 CloudWatch 代理的安装和配置，并且可以立即开始记录和监控。这可以帮助您更好地监控实例配置和启动步骤，以防实例启动失败。如果您不打算使用 Systems Manager 代理，这种方法也是合适的。如果您使用这种方法，则应评估以下注意事项：

- 更新过程必须存在，因为 AMIs 可能不包括最新的 CloudWatch 代理版本。AMI 中安装的 CloudWatch 代理仅在上次创建 AMI 时才是最新的。您应该包括另一种方法，用于定期更新代理以及在 EC2 实例配置时进行更新。如果您使用 Systems Manager，则可以使用本指南中提供的使用 Systems Manager 分发服务器和状态管理器安装 CloudWatch 代理解决方案来解决这个问题。如果您不使用 Systems Manager，则可以使用用户数据脚本在实例启动和重启时更新代理。
- 您的 CloudWatch 代理配置文件必须在实例启动时检索。如果您不使用 Systems Manager，则可以配置用户数据脚本以在启动时检索配置文件，然后重新启动 CloudWatch 代理。
- 更新 CloudWatch 配置后，必须重新启动 CloudWatch 代理。
- AWS 证书不得保存在 AMI 中。确保 AMI 中未存储任何本地 AWS 凭证。如果您使用 Amazon EC2，则可以将必要的 IAM 角色应用于您的实例，避免使用本地证书。如果您使用本地实例，则应在启动 CloudWatch 代理之前自动或手动更新实例凭证。

# 在 Amazon ECS 上进行日志记录和监控

亚马逊弹性容器服务 (Amazon ECS) Service 为正在运行的容器提供了两种启动类型，它们决定了托管任务和服务的基础设施类型；这些启动类型 AWS Fargate 是和 Amazon EC2。两种启动类型均可集成， CloudWatch 但配置和支持各不相同。

以下各节可帮助您了解如何使用在 Amazon ECS 上 CloudWatch 进行日志记录和监控。

主题


- 使用 EC2 启动类型 CloudWatch 进行配置
- 适用于 EC2 和 Fargate 启动类型的 Amazon ECS 容器日志
- 在 Amazon ECS 中使用自定义日志路由 FireLens
- 亚马逊 ECS 的指标


## 使用 EC2 启动类型 CloudWatch 进行配置

使用 EC2 启动类型，您可以预置一个由 EC2 实例组成的 Amazon ECS 集群，这些实例使用 CloudWatch代理进行日志记录和监控。亚马逊 ECS 优化的 AMI 预装了亚马逊 ECS 容器代理，可提供亚马逊 ECS 集群的 CloudWatch 指标。

这些默认指标包含在 Amazon ECS 的成本中，但是 Amazon ECS 的默认配置不监控日志文件或其他指标（例如，可用磁盘空间）。您可以使用为 EC2 启动类型配置 Amazon ECS 集群，这将创建一个部署具有启动配置的 Amazon EC2 Auto Scaling 组的 CloudFormation 堆栈。 AWS 管理控制台 但是，这种方法意味着您无法选择自定义 AMI，也无法使用不同的设置或其他启动脚本自定义启动配置。

要监控其他日志和指标，您必须在 Amazon ECS 容器实例上安装 CloudWatch 代理。您可以使用本指南使用 Systems Manager 分发服务器和状态管理器安装 CloudWatch 代理部分中的 EC2 实例安装方法。但是，Amazon ECS AMI 不包括所需的 Systems Manager 代理。在创建 Amazon ECS 集群时，您应该使用带有用户数据脚本的自定义启动配置，该脚本会安装 Systems Manager 代理。这允许您的容器实例向 Systems Manager 注册并应用状态管理器关联来安装、配置和更新 CloudWatch 代理。当 State Manager 运行并更新您的 CloudWatch 代理配置时，它还会应用您的 Amazon EC2 标准系统级 CloudWatch 配置。您还可以将 Amazon ECS 的标准化 CloudWatch 配置存储在 CloudWatch 配置的 S3 存储桶中，并使用状态管理器自动应用这些配置。

您应确保应用于您的 Amazon ECS 容器实例的 IAM 角色或实例配置文件包含必填项`CloudWatchAgentServerPolicy`和`AmazonSSMManagedInstanceCore`策略。你可以使用

ecs_cl uster_with_cloudwatch_linux.yaml 模板来配置基于 Linux 的 Amazon EC CloudFormation S 集群。此模板创建具有自定义启动配置的 Amazon ECS 集群，该配置用于安装 Systems Manager，并部署自定义 CloudWatch 配置来监控特定于 Amazon ECS 的日志文件。

您应该为 Amazon ECS 容器实例捕获以下日志以及标准 EC2 实例日志：

- Amazon ECS 代理启动输出 — `/var/log/ecs/ecs-init.log`
- 亚马逊 ECS 代理输出 — `/var/log/ecs/ecs-agent.log`
- IAM 凭证提供商请求日志 — `/var/log/ecs/audit.log`

有关输出级别、格式和其他配置选项的更多信息，请参阅 Amazon ECS 文档中的 Amazon ECS 日志文件位置。

> **Important**
>
> Fargate 启动类型不需要安装或配置代理，因为您不运行或管理 EC2 容器实例。

Amazon ECS 容器实例应使用经过优化的最新亚马逊 ECS AMIs 和容器代理。 AWS 使用亚马逊 ECS 优化的 AMI 信息（包括 AMI ID）存储公共 Systems Manager 参数存储参数。您可以使用经优化的 Amazon ECS 的参数存储参数格式从参数存储中检索最新优化的 AMI AMIs。您可以在 CloudFormation 模板中引用公共参数存储参数，该参数引用最新的 AMI 或特定 AMI 版本。

AWS 在每个支持的区域中提供相同的参数存储参数。这意味着引用这些参数的 CloudFormation 模板可以跨区域和账户重复使用，而无需更新 AMI。您可以通过参考特定版本来控制将较新 Amazon ECS 部署 AMIs 到您的组织，这有助于您在测试之前防止使用经过优化 Amazon ECS 的新型 AMI。

## 适用于 EC2 和 Fargate 启动类型的 Amazon ECS 容器日志

Amazon ECS 使用任务定义将容器作为任务和服务进行部署和管理。您可以在任务定义中配置要启动到 Amazon ECS 集群中的容器。使用容器级别的日志驱动程序配置日志记录。多个日志驱动程序选项为您的容器提供不同的日志系统（例如、、、、、`awslogs`、`fluentd`、`gelf`、、`json-file`、`journald`、`logentriessplunksyslog`、或`awsfirelens`），具体取决于您使用的是 EC2 还是 Fargate 启动类型。Fargate 启动类型提供了以下日志驱动程序选项的子集：`awslogssplunk`、和。`awsfirelens` AWS 提供`awslogs`日志驱动程序，用于捕获容器输出并将其传输到 CloudWatch Logs。日志驱动程序设置使您可以自定义日志组、区域和日志流前缀以及许多其他选项。

日志组的默认命名和上的 “自动配置 CloudWatch 日志” 选项使用的选项 AWS 管理控制台 是/ecs/<task_name>。Amazon ECS 使用的日志流名称的<awslogs-stream-prefix>/<container_name>/<task_id>格式为。我们建议您使用根据组织要求对日志进行分组的群组名称。在下表中，image_name和包含image_tag在日志流的名称中。

| | |
|---|---|
| 日志组名称 | /<Business unit>/<Project or application name>/<Environment>/<Cluster name>/<Task name> |
| 日志流名称前缀 | /<image_name>/<image_tag> |

此信息也可在任务定义中找到。但是，任务会定期使用新的修订版进行更新，这意味着任务定义可能使用了image_nameimage_tag与任务定义当前使用的不同的。有关更多信息和命名建议，请参阅本指南的规划您的 CloudWatch 部署部分。

如果您使用持续集成和持续交付（CI/CD) pipeline or automated process, you can create a new task definition revision for your application with each new Docker image build. For example, you can include the Docker image name, image tag, GitHub revision, or other important information in your task definition revision and logging configuration as a part of your CI/CD流程。

## 在 Amazon ECS 中使用自定义日志路由 FireLens

FireLens for Amazon ECS 可帮助您将日志路由到 Fluentd 或 Fluen t Bit，这样您就可以直接将容器日志发送到 AWS 服务和 AWS 合作伙伴网络 (APN) 目的地，并支持将日志传送到日志。 CloudWatch

AWS 为 Fl uent Bit 提供 Docker 镜像，其中预装了亚马逊 Kinesis Data Streams、Amazon Data Firehose 和日志的插件。 CloudWatch 您可以使用 FireLens 日志驱动程序代替日志驱动程序，以便对发送awslogs到 Logs 的 CloudWatch 日志进行更多自定义和控制。

例如，您可以使用 FireLens 日志驱动程序来控制日志格式输出。这意味着 Amazon ECS 容器的 CloudWatch 日志会自动格式化为 JSON 对象，并包含ecs_cluster、、、ecs_task_arnecs_task_definitioncontainer_idcontainer_name、和 JSON 格式的属性。ec2_instance_id当您指定awsfirelens驱动程序时，Fluent 主机将通过FLUENT_HOST和FLUENT_PORT环境变量暴露给您的容器。这意味着你可以使用 fluent 的记录器库直接从代码中登录到日志路由器。例如，您的应用程序可能包含使用环境变量中提供的值登录到 Fluent Bit 的fluent-logger-python库。

如果您选择用 FireLens 于 Amazon ECS，则可以配置与awslogs日志驱动程序相同的设置，也可以使用其他设置。例如，您可以使用 ecs-task-nginx-firelense.js on Amazon ECS 任务定义来启动配置为 FireLens 用于登录的 NGINX 服务器。 CloudWatch它还会启动一个 FireLens Fluent Bit 容器作为日志记录的边车。

## 亚马逊 ECS 的指标

Amazon ECS 使用 Amazon ECS 容器代理在集群和服务级别为 EC2 和 Fargate 启动类型提供标准 CloudWatch 指标（例如 CPU 和内存利用率）。您还可以使用 Container Insights 捕获服务、任务和 CloudWatch容器的指标，或者使用嵌入式指标格式捕获自己的自定义容器指标。

Container Insights 是一项 CloudWatch 功能，可提供集群、容器实例、服务和任务级别的 CPU 利用率、内存利用率、网络流量和存储等指标。Container Insights 还会创建自动仪表板，帮助您分析服务和任务，并查看容器级别的平均内存或 CPU 使用率。Container Insights 将ECS/ContainerInsights自定义指标发布到自定义命名空间，可用于绘制图表、警报和仪表板。

您可以通过为每个 Amazon ECS 集群启用容器洞察来开启容器洞察指标。如果您还想查看容器实例级别的指标，可以将 CloudWatch 代理作为守护程序容器启动到您的 Amazon ECS 集群上。您可以使用 cwagent-ecs-instance-metric-cfn.yaml CloudFormation 模板将代理部署 CloudWatch 为 Amazon ECS 服务。重要的是，此示例假设您创建了相应的自定义 CloudWatch代理配置，并将其与密钥一起存储在 Parameter Store 中ecs-cwagent-daemon-service。

作为 Container Insights 的守护程序 CloudWatch 容器部署的CloudWatch代理包括其他磁盘、内存和 CPU 指标，例如instance_cpu_reserved_capacity和instance_memory_reserved_capacityClusterName、C度。容器实例级别的指标由 Container Insights 使用 CloudWatch 嵌入式指标格式实现。您可以使用本指南为 CloudWatch 代理部署和配置设置状态管理器和分发服务器部分中的方法为 Amazon ECS 容器实例配置其他系统级指标。

### 在 Amazon ECS 中创建自定义应用程序指标

您可以使用CloudWatch嵌入式指标格式为应用程序创建自定义指标。awslogs日志驱动程序可以解释 CloudWatch 嵌入式指标格式语句。

以下示例中的CW_CONFIG_CONTENT环境变量设置为 S cwagentconfig ystems Manager 参数存储参数的内容。您可以使用此基本配置运行代理，将其配置为嵌入式指标格式端点。但是，它已不再必要。

```
  {
  "logs": {
    "metrics_collected": {
      "emf": { }
    }
  }
}
```

如果您在多个账户和地区部署了 Amazon ECS，则可以使用 AWS Secrets Manager 密钥来存储您的 CloudWatch 配置，并将密钥策略配置为与您的组织共享。您可以使用任务定义中的 secrets 选项来设置`CW_CONFIG_CONTENT`变量。

您可以在应用程序中使用 AWS 提供的开源嵌入式指标格式库，并指定`AWS_EMF_AGENT_ENDPOINT`环境变量以连接到充当嵌入式指标格式端点的 CloudWatch 代理 sidecar 容器。例如，您可以使用 ecs_cw_emf_example 示例 Python 应用程序将嵌入式指标格式的指标发送到配置为嵌入式指标格式端点的代理 CloudWatch sidecar 容器。

的 Flu ent Bit 插件还 CloudWatch 可用于发送嵌入式公制格式消息。你也可以使用 ecs_firelense_emf_example 示例 Python 应用程序将嵌入式指标格式的指标发送到 F irelens for Amazon ECS 边车容器。

如果您不想使用嵌入式指标格式，则可以通过 AWS API 或 AWS SDK 创建和更新 CloudWatch 指标。除非您有特定的用例，否则我们不建议使用这种方法，因为它会增加代码的维护和管理开销。

# Amazon EKS 中的日志记录和监控

亚马逊 Elastic Kubernetes Service（亚马逊 EKS） CloudWatch 与 Kubernetes 控制平面的日志集成。控制平面由 Amazon EKS 作为托管服务提供，您无需安装 CloudWatch代理即可开启日志记录。也可以部署 CloudWatch 代理来捕获 Amazon EKS 节点和容器日志。还支持 Fluent Bit 和 Fluentd 将你的容器日志发送到 Logs。 CloudWatch

CloudWatch Container Insights 为集群、节点、容器、任务和服务级别的 Amazon EKS 提供了全面的指标监控解决方案。Amazon EKS 还支持使用 Promethe us 捕获指标的多种选项。Amazon EKS 控制平面提供了一个以 Prometheus 格式公开指标的指标终端节点。您可以将 Prometheus 部署到您的 Amazon EKS 集群中以使用这些指标。

除了使用其他 Prometheus 端点外，您还可以将 CloudWatch 代理设置为抓取 Prometheus 指标并 CloudWatch 创建指标。Prometheus 的 Container Insights 监控还可以自动发现和捕获受支持的容器化工作负载和系统中的 Prometheus 指标。

您可以在 Amazon EKS 节点上安装和配置 CloudWatch 代理，方法与使用分销商和状态管理器的 Amazon EC2 所使用的方法类似，以使您的 Amazon EKS 节点与您的标准系统日志和监控配置保持一致。

## 登录 Amazon EKS

Kubernetes 日志可以分为控制平面日志、节点日志和应用程序日志记录。Kubernetes 控制平面是一组组组件，用于管理 Kubernetes 集群并生成用于审计和诊断目的的日志。使用 Amazon EKS，您可以打开不同控制平面组件的日志并将其发送到 CloudWatch。

Kubernetes 还会在运行你的 Pod 的每个 Kubernetes `kube-proxy` 节点上运行系统组件，例如`kubelet`和。这些组件在每个节点内写入日志，您可以配置 CloudWatch 容器见解以捕获每个 Amazon EKS 节点的这些日志。

容器被分组为 Kubernetes 集群中的 Pod，并计划在你的 Kubernetes 节点上运行。大多数容器化应用程序写入标准输出和标准错误，容器引擎将输出重定向到日志驱动程序。在 Kubernetes 中，容器日志位于节点上的`/var/log/pods`目录中。您可以配置 CloudWatch 和容器见解来捕获每个 Amazon EKS 容器的这些日志。

## Amazon EKS 控制面板日志记录

Amazon EKS 集群由用于您的 Kubernetes 集群的高可用性单租户控制平面和运行您的容器的 Amazon EKS 节点组成。控制平面节点在由管理的账户中运行 AWS。Amazon EKS 集群控制平面节点 CloudWatch 与集成，您可以为特定控制平面组件开启日志记录。

为每个 Kubernetes 控制平面组件实例提供了日志。 AWS 管理控制平面节点的运行状况，并为 Kubernetes 终端节点提供服务级别协议 (SLA)。

## Amazon EKS 节点和应用程序日志

我们建议您使用CloudWatch容器见解来捕获 Amazon EKS 的日志和指标。Container Insights 使用 CloudWatch 代理实现集群、节点和 pod 级别的指标，以及用于日志捕获的 Fluent Bit 或 Fluentd。CloudWatch容器见解还提供自动仪表板，其中包含您捕获的 CloudWatch 指标的分层视图。容器见解部署为 CloudWatch DaemonSet 在每个 Amazon EKS 节点上运行 DaemonSet 的 Fluent Bit。Container Insights 不支持 Fargate 节点，因为这些节点由管理 AWS 且不支持。 DaemonSets本指南中单独介绍了 Amazon EKS 的 Fargate 日志记录。

下表显示了 Amazon E CloudWatch KS 的默认 Fluentd 或 Fluent Bit 日志捕获配置捕获的日志组和日志。

| | |
|---|---|
| `/aws/containerinsights/Cluster_Name/application` | 所有日志文件都已输入`/var/log/containers` 。该目录提供了指向目录结构中所有 Kubernetes 容器日志的`/var/log/pods` 符号链接。这将捕获写入`stdout`或的应用程序容器日志`stderr`。它还包括 Kubernetes 系统容器的日志`aws-vpc-cni-init` ，例如、`kube-proxy` 和。`coreDNS` |
| `/aws/containerinsights/Cluster_Name/host` | 来自`/var/log/dmesg` `/var/log/secure` 、和的日志`/var/log/messages` 。 |
| `/aws/containerinsights/Cluster_Name/dataplane` | `kubelet.service` 、`kubeproxy.service` 和 `docker.service` 的 `/var/log/journal` 中的日志。 |

如果您不想使用带有 Fluent Bit 或 Fluentd 的 Container Insights 进行日志记录，则可以使用安装在 Amazon EKS 节点上的 CloudWatch 代理来捕获节点和容器日志。Amazon EKS 节点是 EC2 实例，这意味着您应该将它们包含在适用于 Amazon EC2 的标准系统级日志记录方法中。如果您使用分销商和状态管理器安装代理，则 CloudWatch 代理的安装、配置和更新中还包括 Amazon EKS 节点。CloudWatch

下表显示了特定于 Kubernetes 的日志，如果您没有使用带有 Fluent Bit 或 Fluentd 的容器见解进行日志记录，则必须捕获这些日志。

| | |
|---|---|
| `/var/log/containers` | 该目录提供了指向目录结构下所有 Kubernetes 容器日志的`/var/log/pods` 符号链接。这可以有效地捕获写入`stdout`或的应用程序容器日志`stderr`。这包括 Kubernetes 系统容器的日志`aws-vpc-cni-init` ，例如、`kube-proxy` 和。`coreDNS`重要：如果您使用的是容器见解，则不需要这样做。 |
| `var/log/aws-routed-eni/ipamd.log`<br><br>`/var/log/aws-routed-eni/plugin.log` | L-IPAM 守护程序的日志可以在这里找到 |

您必须确保 Amazon EKS 节点安装并配置 CloudWatch 代理以发送相应的系统级日志和指标。但是，亚马逊 EKS 优化的 AMI 不包括 Systems Manager 代理。通过使用启动模板，您可以自动安装 Systems Manager 代理并使用通过用户数据部分实现的启动脚本来捕获重要的 Amazon EKS 特定日志的默认 CloudWatch 配置。Amazon EKS 节点使用 Auto Scaling 组作为托管节点组或自管理节点进行部署。

对于托管节点组，您可以提供包含用户数据部分的启动模板，以自动安装和 CloudWatch 配置 Systems Manager 代理。您可以自定义并使用 amazon_eks_managed_node_group_launch_config.yaml CloudFormation 模板来创建启动模板，该模板用于安装 Systems Manager 代 CloudWatch 理、代理，还可以将 Amaz on EKS 特定的日志配置添加到配置目录中。 CloudWatch 此模板可用于通过 infrastructure-as-code (IaC) 方法更新您的 Amazon EKS 托管节点组启动模板。 CloudFormation 模板的每次更新都会提供一个新版本的启动模板。然后，您可以更新节点组以使用新的模板版本，并让托管生命周期流程在不停机的情况下更新您的节点。确保应用于您的托管节点组的 IAM 角色和实例配置文件包含`CloudWatchAgentServerPolicy`和`AmazonSSMManagedInstanceCore` AWS 托管策略。

使用自行管理的节点，您可以直接为 Amazon EKS 节点配置和管理生命周期和更新策略。自我管理的节点允许你在 Amazon EKS 集群和 Bottlerock et 上运行 Windows 节点以及其他选项。您可以使用将自我管理的节点部署 CloudFormation 到您的 Amazon EKS 集群中，这意味着您可以对 Amazon EKS 集群使用 IaC 和托管变更方法。 AWS 提供了 amazon-eks-nodegroup.yaml CloudFormation 模板，您可以按原样使用或自定义。该模板为集群中的 Amazon EKS 节点预配置所有必需的资源（例如，单独的 IAM 角色、安全组、Amazon EC2 Auto Scaling 组和启动模板）。amazon-eks-nodegroup.yaml CloudFormation 模板是一个更新版本，用于安装所需的 Systems Manager 代理和 CloudWatch 代理，并将特定于 Amazon EKS 的日志配置添加到 CloudWatch 配置目录中。

## 在 Fargate 上登录亚马逊 EKS

借助 Fargate 上的 Amazon EKS，您无需分配或管理 Kubernetes 节点即可部署 Pod。这样就无需为 Kubernetes 节点捕获系统级日志。要从 Fargate 吊舱中捕获日志，你可以使用 Fluent Bit 将日志直接转发到。 CloudWatch这使您 CloudWatch 无需进一步配置即可自动将日志路由到 Fargate 上的 Amazon EKS 容器，也无需为 Fargate 上的 Amazon EKS 容器设置边车容器。有关这方面的更多信息，请参阅亚马逊 EKS 文档中的 Fargate 登录和博客上的 Fr uent Bit for Amazon EKS。 AWS 此解决方案基于在 Farg STDOUT ate 上为 Amazon EKS 集群建立的 Fluent Bit 配置，捕获容器中的和 STDERR input/output (I/O) 流，并 CloudWatch 通过 Fruent Bit 将其发送到。

# 亚马逊 EKS 和 Kubernetes 的指标

Kubernetes 提供了一个指标 API，允许您访问资源使用率指标（例如，节点和 Pod 的 CPU 和内存使用情况），但该 API 仅提供 point-in-time信息，不提供历史指标。Kubernetes 指标服务器通常用于 Amazon EKS 和 Kubernetes 部署，以汇总指标，提供有关指标的短期历史信息，并支持横向 Pod Autoscaler 等功能。

Amazon EKS 通过 Kubernetes API 服务器以 Prometheus 格式公开控制平面指标，并且可以捕获和摄取这些指标。 CloudWatch CloudWatch 还可以将容器见解配置为为您的 Amazon EKS 节点和 pod 提供全面的指标捕获、分析和警报。

## Kubernetes 控制平面指标

Kubernetes 使用 HTTP API 端点以 Prometheus 格式公开控制平面指标。`/metrics`你应该在你的 Kubernetes 集群中安装 Prometheus，以便使用网络浏览器绘制和查看这些指标。你也可以将 Kubernetes API 服务器公开的指标提取到。 CloudWatch

# Kubernetes 的节点和系统指标

Kubernetes 提供了 Prometheus 指标服务器容器，你可以在 Kubernetes 集群上部署和运行这些容器，以获取集群、节点和 pod 级别的 CPU 和内存统计信息。这些指标用于水平吊舱自动扩缩器和垂直吊舱自动扩缩器。 CloudWatch 也可以提供这些指标。

如果您使用 Kubernetes 控制面板或水平和垂直容器自动扩缩器，则应安装 Kubernetes 指标服务器。Kubernetes 控制面板可帮助您浏览和配置 Kubernetes 集群、节点、容器和相关配置，并从 Kubernetes 指标服务器查看 CPU 和内存指标。

Kubernetes 指标服务器提供的指标不能用于非自动扩展目的（例如，监控）。这些指标用于 point-in-time分析，而不是历史分析。Kubernetes 控制面板部署用于在短时间内存储`dashboard-metrics-scraper`来自 Kubernetes 指标服务器的指标。

Container Insights 使用在 Kubernetes 中运行的 CloudWatch 代理的容器化版本 DaemonSet 来发现集群中所有正在运行的容器并提供节点级指标。它收集性能堆栈每一层的性能数据。您可以使用快速入门中的 AWS 快速入门或单独配置容器见解。Quick Start 使用 CloudWatch 代理设置指标监控和使用 Fluent Bit 进行日志记录，因此您只需要部署一次即可进行日志记录和监控。

由于 Amazon EKS 节点是 EC2 实例，因此除了容器洞察捕获的指标外，您还应使用您为 Amazon EC2 定义的标准来捕获系统级指标。您可以使用本指南为 CloudWatch 代理部署和配置设置状态管理器和分发服务器部分中的相同方法为您的 Amazon EKS 集群安装和配置 CloudWatch 代理。您可以更新您的 Amazon EKS 特定 CloudWatch配置文件以包含指标以及您的 Amazon EKS 特定日志配置。

支持 Prometheus 的 CloudWatch 代理可以自动从支持的容器化工作负载和系统中发现和抓取 Prometheus 指标。它会将它们作为嵌入式指标格式的 CloudWatch 日志摄取，以便使用 L CloudWatch ogs Insights 进行分析，并自动创建 CloudWatch指标。

> **Important**
>
> 您必须部署 CloudWatch 代理的专用版本才能收集 Prometheus 指标。这是与为 Container Insights 部署的 CloudWatch 代理不同的代理。您可以使用 prometheus_jmx 示例 J ava 应用程序（其中包括代理 CloudWatch 和 Amazon EKS 容器部署的部署和配置文件）来演示 Prometheus 指标发现。有关更多信息，请参阅文档中的在 Amazon EKS 和 Kubernetes 上设置 Java/JMX 示例工作负载。 CloudWatch您还可以将 CloudWatch 代理配置为从您的 Amazon EKS 集群中运行的其他 Prometheus 目标捕获指标。

## 应用程序指标

您可以使用CloudWatch嵌入式指标格式创建自己的自定义指标。要提取嵌入式指标格式语句，您需要将嵌入式指标格式条目发送到嵌入式指标格式端点。可以将 CloudWatch 代理配置为您的 Amazon EKS 容器中的边车容器。 CloudWatch 代理配置存储为 Kubernetes， ConfigMap 并由您的 CloudWatch 代理 sidecar 容器读取以启动嵌入式指标格式端点。

您还可以将应用程序设置为 Prometheus 目标，并在支持 Prometheus 的支持下配置 CloudWatch代理，以发现、抓取和提取您的指标。 CloudWatch例如，您可以将开源 JMX 导出器与 J ava 应用程序一起使用，公开 JMX Beans 供代理使用 Prometheus。 CloudWatch

如果您不想使用嵌入式指标格式，也可以使用 AWS API 或 AWS SDK 创建和更新 CloudWatch指标。但是，我们不建议使用这种方法，因为它将监控和应用程序逻辑混为一谈。

## Fargate 上亚马逊 EKS 的指标

Fargate 会自动配置 Amazon EKS 节点来运行你的 Kubernetes pod，因此你无需监控和收集节点级指标。但是，您必须监控在 Fargate 上的 Amazon EKS 节点上运行的 Pod 的指标。Fargate 上的 Amazon EKS 目前不支持容器见解，因为它需要以下目前不支持的功能：

- DaemonSets 目前不支持。Container Insights 是通过 DaemonSet 在每个集群节点上以身份运行 CloudWatch 代理来部署的。
- HostPath 不支持永久卷。 CloudWatch 代理容器使用 HostPath 永久卷作为收集容器指标数据的先决条件。
- Fargate 禁止特权容器和访问主机信息。

您可以使用 Fargate 的内置日志路由器向发送嵌入式指标格式语句。 CloudWatch日志路由器使用 Fluent Bit，它有一个可以配置为支持嵌入式指标格式语句的 CloudWatch 插件。

您可以在 Amazon EKS 集群中部署 Prometheus 服务器，从 Fargate 节点收集指标，从而检索和捕获 Fargate 节点的吊舱级别指标。由于 Prometheus 需要永久存储，因此如果您使用亚马逊弹性文件系统 (Amazon EFS) 进行永久存储，则可以在 Fargate 上部署 Prometheus。您也可以在亚马逊 EC2 支持的节点上部署 Prometheus。有关更多信息，请参阅博客上关于 AWS Fargate 使用 Prometheus 和 Grafana 的监控 Amazon EKS。 AWS

# Prometheus 在亚马逊 EKS 上进行监控

适用于 Prometheus 的亚马逊托管服务为开源 Prometheus 提供可扩展、安全的 AWS 托管服务。您可以使用 Prometheus 查询语言 (PromQL) 监控容器化工作负载的性能，而无需管理用于接收、存储和查询运营指标的底层基础架构。你可以使用 OpenTelemetry Distro for (ADOT) 或 Prometheus 服务器作为收集代理，从亚马逊 EKS 和 Amazon ECS AWS 收集 Prometheus 指标。

CloudWatch Prometheus 的 Container Insights 监控使您能够配置和使用 CloudWatch 代理来发现 Amazon ECS、Amazon EKS 和 Kubernetes 工作负载中的 Prometheus 指标，并将其作为指标摄取。CloudWatch如果这是您的主要可观测性和监控解决方案，则此解决方案 CloudWatch 是合适的。但是，以下列表概述了适用于 Prometheus 的亚马逊托管服务为提取、存储和查询 Prometheus 指标提供了更大的灵活性的用例：

- 亚马逊 Prometheus 托管服务使您能够使用部署在亚马逊 EKS 或自行管理的 Kubernetes 中的现有 Prometheus 服务器，并将其配置为写入适用于 Prometheus 的亚马逊托管服务，而不是本地配置的数据存储。这消除了为 Prometheus 服务器及其基础架构管理高可用性数据存储所带来的无差别繁重的工作。如果您想在云端使用成熟的 Prometheus 部署，那么适用于 Prometheus 的亚马逊托管服务是一个不错的选择。 AWS
- Grafana 直接支持 Prometheus 作为可视化的数据源。如果您想使用带有 Prometheus 的 Grafana 而不是控制面板来监控容器，那么适用于 Prometheus CloudWatch 的亚马逊托管服务可以满足您的要求。适用于 Prometheus 的亚马逊托管服务与 Amazon Managed Grafana 集成，提供托管式开源监控和可视化解决方案。
- Prometheus 使您能够使用 PromQL 查询对运营指标进行分析。相比之下，CloudWatch 代理会将嵌入指标格式 CloudWatch 的 Prometheus 指标提取到日志中，从而生成指标。 CloudWatch 您可以使用 Logs Insights 查询嵌入式指标格式 CloudWatch 日志。
- 如果您不打算 CloudWatch 用于监控和指标捕获，则应将适用于 Prometheus 的亚马逊托管服务与 Prometheus 服务器和 Grafana 等可视化解决方案一起使用。您需要将 Prometheus 服务器配置为从 Prometheus 目标中获取指标，并将服务器配置为远程写入适用于 Prometheus 的亚马逊托管服务工作空间。如果您使用亚马逊托管 Grafana，则可以使用随附的插件将亚马逊托管 Grafana 与适用于 Prometheus 的亚马逊托管服务数据源直接集成。由于指标数据存储在适用于 Prometheus 的 Amazon 托管服务中，因此无需部署 CloudWatch 代理，也不需要将数据采集到其中。 CloudWatchPrometheus 的 Container Insights 监控需要该 CloudWatch 代理。

您还可以使用 ADOT 收集器从装有 Prometheus 工具的应用程序中提取指标，然后将指标发送到适用于 Prometheus 的亚马逊托管服务。有关 ADOT Collector 的更多信息，请参阅AWS 发行版中的文档。 OpenTelemetry

# 的日志和指标 AWS Lambda

Lambda 无需为工作负载管理和监控服务器，无需对应用程序的代码进行进一步配置或检测，即可自动使用 CloudWatch指标和 CloudWatch 日志。本节可帮助您了解 Lambda 所用系统的性能特征以及您的配置选择如何影响性能。它还可以帮助您记录和监控您的 Lambda 函数，以优化性能并诊断应用程序级问题。

## Lambda 函数日志

Lambda 自动将标准输出和标准错误消息从 Lambda 函数流式传输到 CloudWatch 日志，无需记录驱动程序。Lambda 还会自动配置运行您的 Lambda 函数的容器，并将其配置为在单独的日志流中输出日志消息。

后续调用 Lambda 函数可以重复使用相同的容器并输出到相同的日志流。Lambda 还可以预置新的容器并将调用输出到新的日志流。

首次调用 Lambda 函数时，Lambda 会自动创建一个日志组。Lambda 函数可以有多个版本，您可以选择要运行的版本。Lambda 函数调用的所有日志都存储在同一个日志组中。名称无法更改，`/aws/lambda/<YourLambdaFunctionName>`格式为。在日志组中为每个 Lambda 函数实例创建一个单独的日志流。对于使用某种格式的日志流，Lambda 有一个标准的命名约定。`YYYY/MM/DD/[<FunctionVersion>]<InstanceId>`由 AWS 生成`InstanceId`，用于标识 Lambda 函数实例。

我们建议您将日志消息格式化为 JSON 格式，因为您可以使用 Logs Insights 更轻松地 CloudWatch 查询它们。也可以更轻松地对其进行过滤和导出。您可以使用日志库来简化此过程或编写自己的日志处理函数。我们建议您使用日志库来帮助格式化和分类日志消息。例如，如果您的 Lambda 函数是用 Python 编写的，则可以使用 Py thon 日志模块来记录消息并控制输出格式。Lambda 原生使用 Python 日志库来存储用 Python 编写的 Lambda 函数，您可以在 Lambda 函数中检索和自定义记录器。 AWS Labs 创建了 AWS Lambda Powertools for Python 开发者工具包，以便更轻松地使用冷启动等关键数据来丰富日志消息。该工具包可用于 Python、Java、Typescript 和.NET。

另一种最佳做法是使用变量设置日志输出级别，并根据环境和您的要求进行调整。除了使用的库之外，您的 Lambda 函数的代码还可能输出大量日志数据，具体取决于日志输出级别。这可能会影响您的日志记录成本并影响性能。

Lambda 允许您在不更新代码的情况下为 Lambda 函数运行时环境设置环境变量。例如，您可以创建一个`LAMBDA_LOG_LEVEL`环境变量来定义可以从代码中检索的日志输出级别。以下示例尝试检索`LAMBDA_LOG_LEVEL`环境变量并使用该值来定义日志输出。如果未设置环境变量，则默认为`INFO`级别。

```
import logging
from os import getenv

logger = logging.getLogger()
log_level = getenv("LAMBDA_LOG_LEVEL", "INFO")
level = logging.getLevelName(log_level)
logger.setLevel(level)
```

## 将日志发送到其他目的地 CloudWatch

您可以使用订阅筛选条件将日志发送到其他目的地（例如，亚马逊 OpenSearch 服务或 Lambda 函数）。如果您不使用 Amazon OpenSearch 服务，则可以使用 Lambda 函数来处理日志，然后使用将其发送到您选择的 AWS 服务。 AWS SDKs

您还可以在 Lambda 函数中使用 SDKs AWS 云以外的日志目标，将日志语句直接发送到您选择的目标。如果您选择此选项，我们建议您考虑延迟、额外的处理时间、错误和重试处理以及操作逻辑与 Lambda 函数的耦合的影响。

## Lambda 函数指标

Lambda 允许您在不管理或扩展服务器的情况下运行代码，这几乎消除了系统级审计和诊断的负担。但是，了解您的 Lambda 函数的系统级别的性能和调用指标仍然很重要。这可以帮助您优化资源配置并提高代码性能。通过适当调整您的 Lambda 函数规模，有效监控和衡量性能可以改善用户体验并降低成本。通常，作为 Lambda 函数运行的工作负载也有需要捕获和分析的应用程序级指标。Lambda 直接支持嵌入式指标格式，以便更轻松地捕获应用程序级 CloudWatch 指标。

## 系统级指标

Lambda 会自动与 CloudWatch 指标集成，并为您的 Lambda 函数提供一组标准指标。Lambda 还为每个 Lambda 函数提供了一个单独的监控控制面板，其中包含这些指标。您需要监控的两个重要指标是错误和调用错误。了解调用错误与其他错误类型之间的区别有助于您诊断和支持 Lambda 部署。

调用错误会导致您的 Lambda 函数无法运行。这些错误发生在您的代码运行之前，因此您无法在代码中实现错误处理来识别它们。相反，您应该为 Lambda 函数配置警报，以检测这些错误并通知操作和工作负载所有者。这些错误通常与配置或权限错误有关，并且可能是由于您的配置或权限更改而发生的。调用错误可能会启动重试，从而导致多次调用您的函数。

成功调用的 Lambda 函数会返回 HTTP 200 响应，即使该函数抛出了异常也是如此。您的 Lambda 函数应实现错误处理并引发异常，以便该`Errors`指标捕获和识别您的 Lambda 函数的失败运行。您应该从 Lambda 函数调用中返回格式化的响应，其中包含用于确定运行是完全、部分失败还是成功的信息。

CloudWatch 提供了 CloudWatch Lambda 见解，您可以为单个 Lambda 函数启用这些见解。Lambda Insights 收集、汇总和汇总系统级指标（例如 CPU 时间、内存、磁盘和网络使用情况）。Lambda Insights 还会收集、汇总和汇总诊断信息（例如，冷启动和 Lambda 工作程序关闭），以帮助您隔离和快速解决问题。

Lambda Insights 使用嵌入式指标格式自动向日志组发送性能信息，该`/aws/lambda-insights/`日志组带有基于您的 Lambda 函数名称的日志流名称前缀。这些性能日志事件创建的 CloudWatch 指标是自动 CloudWatch 仪表板的基础。我们建议您为性能测试和生产环境启用 Lambda Insights。Lambda Insights 创建的其他指标包括`memory_utilization`有助于正确调整 Lambda 函数的大小，从而避免为不需要的容量付费。

## 应用程序指标

您还可以 CloudWatch 使用嵌入式指标格式创建和捕获自己的应用程序指标。您可以利用AWS 提供的嵌入式指标格式库来创建和发送嵌入式指标格式语句。 CloudWatch集成的 Lambda CloudWatch 日志记录工具配置为处理和提取格式正确的嵌入式指标格式语句。

# 搜索和分析日志 CloudWatch

在以一致的格式和位置捕获日志和指标后，您可以对其进行搜索和分析，以帮助提高运营效率，此外还可以识别和排除问题。我们建议您以格式良好的格式（例如 JSON）捕获日志，以便更轻松地搜索和分析日志。大多数工作负载使用网络、计算、存储和数据库等 AWS 资源的集合。在可能的情况下，您应共同分析来自这些资源的指标和日志，并将它们关联起来，以便有效地监控和管理所有 AWS 工作负载。

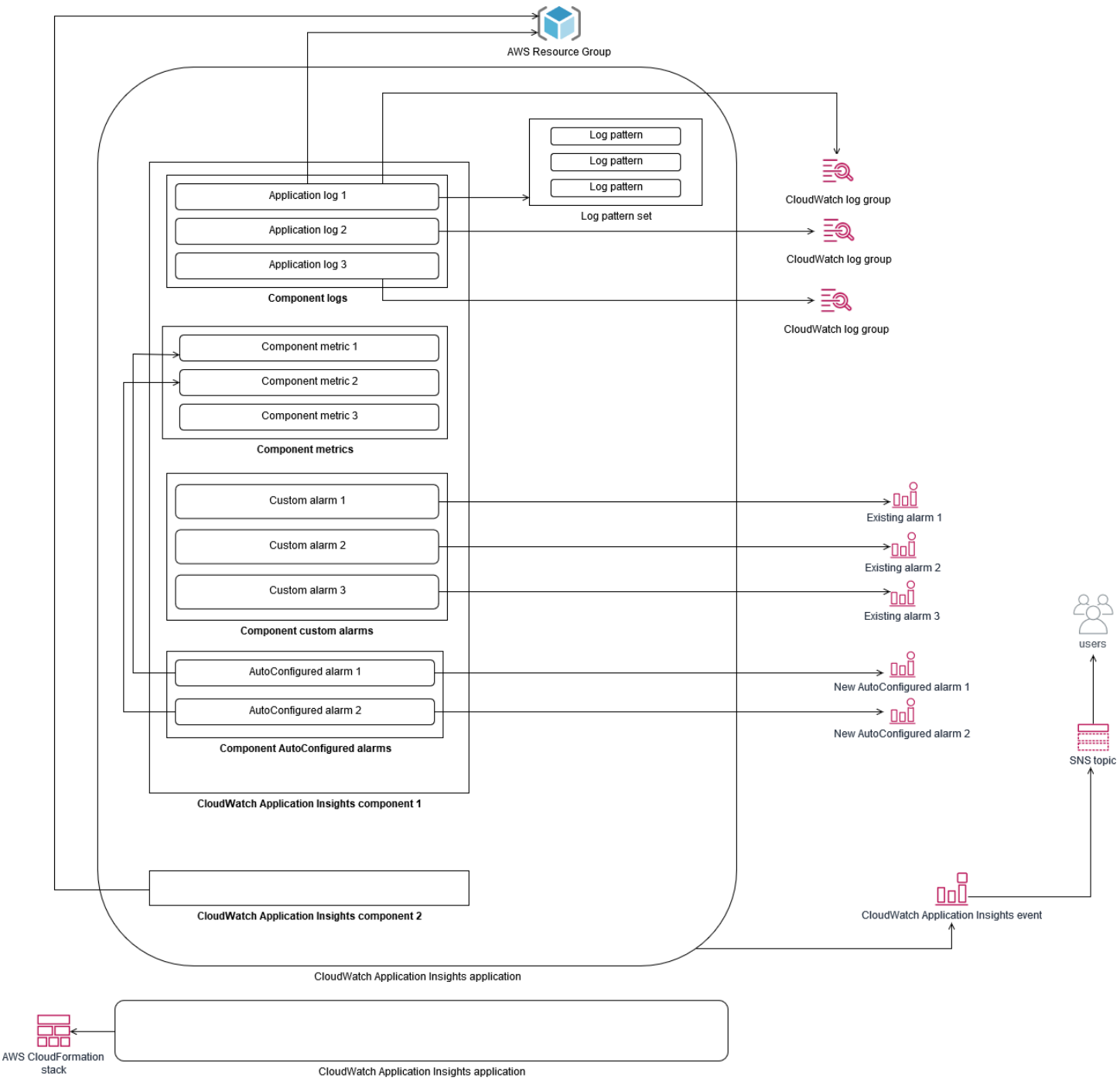
CloudWatch 提供了多种功能来帮助分析日志和指标，例如 CloudWatch Application Insigh ts 用于跨不同 AWS 资源共同定义和监控应用程序的指标和日志，CloudWatch 用于显示指标异常的异常情况，以及用于交互式搜索和分析CloudWatch 日志中日志数据的日志见解。 CloudWatch

## 使用 “应用洞察” 对 CloudWatch 应用程序进行集体监控和分析

应用程序所有者可以使用 Amazon A CloudWatch pplication Insights 来设置工作负载的自动监控和分析。除了为账户中的所有工作负载配置的标准系统级监控之外，还可以进行此配置。通过 App CloudWatch lication Insights 设置监控还可以帮助应用程序团队主动调整运营并缩短平均恢复时间 (MTTR)。 CloudWatch Application Insights 可以帮助减少建立应用程序级日志记录和监控所需的工作量。它还提供了一个基于组件的框架，可帮助团队划分日志和监控职责。

CloudWatch Application Insights 使用资源组来识别应作为应用程序进行集体监控的资源。资源组中支持的资源将成为 Application Insights CloudWatch 应用程序中单独定义的组件。Application Insights CloudWatch 应用程序的每个组件都有自己的日志、指标和警报。

对于日志，您可以定义应在组件和 Application Insights CloudWatch 应用程序中使用的日志模式集。日志模式集是基于正则表达式搜索的日志模式的集合，以及检测到该模式时的低、中或高严重性。对于指标，您可以从服务特定指标和支持的指标列表中选择要监控的每个组件的指标。对于警报，App CloudWatch lication Insights 会自动为所监控的指标创建和配置标准或异常检测警报。 CloudWatch Application Insights 可以自动配置 CloudWatch 文档中 CloudWatch 应用程序见解支持的日志和指标中概述的技术的指标和日志捕获。下图显示了 App CloudWatch lication Insights 组件与其日志和监控配置之间的关系。每个组件都定义了自己的日志和指标，以便使用 CloudWatch 日志和指标进行监控。




App CloudWatch lication Insights 监控的 EC2 实例需要 Systems Manager、 CloudWatch 代理和权限。有关这方面的更多信息，请参阅 CloudWatch 文档中的使用 App lication Insights 配置 CloudWatch 应用程序的先决条件。 CloudWatch Application Insights 使用 Systems Manager 安装和更新 CloudWatch 代理。在 Application Insights 中 CloudWatch 配置的指标和日志会创建一个 CloudWatch 代理配置文件，该文件存储在 Systems Manager 参数中，每个 CloudWatch 应用程序见解组件都有`AmazonCloudWatch-ApplicationInsights-SSMParameter`前缀。这会导致将单独的 CloudWatch 代理配置文件添加到 EC2 实例上的 CloudWatch 代理配置目录中。运行 Systems

Manager 命令将此配置附加到 EC2 实例上的活动配置中。使用 A CloudWatch pplication Insights 不会影响现有的 CloudWatch 代理配置设置。除了自己的系统和 CloudWatch 应用程序级 CloudWatch 代理配置外，您还可以使用 Application Insights。但是，您应确保配置不会重叠。

## 使用 “日志见解” 执行 CloudWatch 日志分析

CloudWatch Logs Insights 使用简单的查询语言可以轻松搜索多个日志组。如果您的应用程序日志采用 JSON 格式结构，则 CloudWatch Logs Insights 会自动发现多个日志组中日志流中的 JSON 字段。您可以使用 CloudWatch Logs Insights 来分析您的应用程序和系统日志，这样可以保存您的查询以备将来使用。L CloudWatch ogs Insights 的查询语法支持诸如使用函数聚合之类的函数，例如 sum ()、avg ()、count ()、min () 和 max ()，这些函数有助于对应用程序进行故障排除或性能分析。

如果您使用嵌入式指标格式创建 CloudWatch 指标，则可以使用支持的聚合函数查询嵌入式指标格式日志，以生成一次性指标。这可以根据需要捕获生成特定指标所需的数据点，而不是主动捕获它们作为自定义指标，从而帮助您降低 CloudWatch 监控成本。这对于基数较高、会产生大量指标的维度特别有效。 CloudWatch Container Insights 也采用这种方法来捕获详细的性能数据，但只为这些数据的子集生成 CloudWatch 指标。

例如，以下嵌入式指标条目仅从嵌入式 CloudWatch指标格式语句中捕获的指标数据生成一组有限的指标：

```
{
 "AutoScalingGroupName": "eks-e0bab7f4-fa6c-64ba-dbd9-094aee6cf9ba",
 "CloudWatchMetrics": [
 {
 "Metrics": [
 {
 "Unit": "Count",
 "Name": "pod_number_of_container_restarts"
 }
 ],
 "Dimensions": [
 [
 "PodName",
 "Namespace",
 "ClusterName"
 ]
 ],
 "Namespace": "ContainerInsights"
 }
 ],
```

```
"ClusterName": "eksdemo",
"InstanceId": "i-03b21a16b854aa4ca",
"InstanceType": "t3.medium",
"Namespace": "amazon-cloudwatch",
"NodeName": "ip-172-31-10-211.ec2.internal",
"PodName": "cloudwatch-agent",
"Sources": [
"cadvisor",
"pod",
"calculated"
],
"Timestamp": "1605111338968",
"Type": "Pod",
"Version": "0",
"pod_cpu_limit": 200,
"pod_cpu_request": 200,
"pod_cpu_reserved_capacity": 10,
"pod_cpu_usage_system": 3.268605094109382,
"pod_cpu_usage_total": 8.899539221131045,
"pod_cpu_usage_user": 4.160042847048305,
"pod_cpu_utilization": 0.44497696105655227,
"pod_cpu_utilization_over_pod_limit": 4.4497696105655224,
"pod_memory_cache": 4096,
"pod_memory_failcnt": 0,
"pod_memory_hierarchical_pgfault": 0,
"pod_memory_hierarchical_pgmajfault": 0,
"pod_memory_limit": 209715200,
"pod_memory_mapped_file": 0,
"pod_memory_max_usage": 43024384,
"pod_memory_pgfault": 0,
"pod_memory_pgmajfault": 0,
"pod_memory_request": 209715200,
"pod_memory_reserved_capacity": 5.148439982463127,
"pod_memory_rss": 38481920,
"pod_memory_swap": 0,
"pod_memory_usage": 42803200,
"pod_memory_utilization": 0.6172094650851303,
"pod_memory_utilization_over_pod_limit": 11.98828125,
"pod_memory_working_set": 25141248,
"pod_network_rx_bytes": 3566.4174629544723,
"pod_network_rx_dropped": 0,
"pod_network_rx_errors": 0,
"pod_network_rx_packets": 3.3495665260575094,
"pod_network_total_bytes": 4283.442421354973,
```

```
 "pod_network_tx_bytes": 717.0249584005006,
 "pod_network_tx_dropped": 0,
 "pod_network_tx_errors": 0,
 "pod_network_tx_packets": 2.6964010534762948,
 "pod_number_of_container_restarts": 0,
 "pod_number_of_containers": 1,
 "pod_number_of_running_containers": 1,
 "pod_status": "Running"
}
```

但是，您可以查询捕获的指标以获得进一步的见解。例如，你可以运行以下查询来查看最新 20 个出现内存页面错误的 pod：

```
fields @timestamp, @message
| filter (pod_memory_pgfault > 0)
| sort @timestamp desc
| limit 20
```

# 使用 Amazon OpenSearch 服务执行日志分析

CloudWatch 通过允许您使用订阅筛选器将日志数据从 CloudWatch 日志组流式传输到您选择的亚马逊 OpenSearch 服务集群，从而与 Amazon Service 集成。 OpenSearch 您可以使用 CloudWatch 主要日志和指标捕获和分析，然后使用 Amazon S OpenSearch ervice 对其进行扩展，以用于以下用例：

- Fine-grained 数据访问控制 — Amazon Ser OpenSearch vice 使您可以将对数据的访问限制到字段级别，并根据用户权限帮助匿名化字段中的数据。如果您想在不暴露敏感数据的情况下支持故障排除，这很有用。
- 跨多个账户、区域和基础设施聚合和搜索日志 — 您可以将来自多个账户和地区的日志流式传输到通用的 Amazon S OpenSearch ervice 集群中。您的集中式运营团队可以分析趋势、问题，并跨账户和地区进行分析。将 CloudWatch 日志流式传输到 Amazon S OpenSearch ervice 还有助于您在中心位置搜索和分析多区域应用程序。
- 使用 ElasticSearch代理直接向 Amazon S OpenSearch ervice 发送和丰富日志 — 您的应用程序和技术堆栈组件可以使用 CloudWatch 代理不支持的操作系统。在将日志数据传送到您的日志解决方案之前，您可能还需要对其进行丰富和转换。亚马逊 OpenSearch 服务支持标准 Elasticsearch 客户端，例如 Elasti c Beats 系列数据传送器和 Logstash，它们支持在将日志数据发送到亚马逊服务之前进行日志充实和转换。 OpenSearch

- 现有的运营管理解决方案使用 Logstash ElasticSearch、Kibana (ELK) 堆栈进行日志记录和监控 — 您可能已经在亚马逊 OpenSearch 服务或开源 Elasticsearch 上进行了大量投资，并且已经配置了许多工作负载。您可能还想继续使用在 Kibana 中创建的操作仪表板。

如果您不打算使用 CloudWatch 日志，则可以使用亚马逊 OpenSearch 服务支持的代理、日志驱动程序和库（例如 Fluent Bit、Fluentd、l ogstash 和 Ope n Distro for ElasticSearch API）将您的日志直接发送到亚马逊服务并绕过。 OpenSearch CloudWatch但是，您还应该实施一种解决方案来捕获 AWS 服务生成的日志。 CloudWatch 日志是许多 AWS 服务的主要日志捕获解决方案，多个服务会自动在中创建新的日志组 CloudWatch。例如，Lambda 会为每个 Lambda 函数创建一个新的日志组。您可以为日志组设置订阅筛选条件，以将其日志流式传输到 Amazon S OpenSearch ervice。您可以为要流式传输到 Amazon S OpenSearch ervice 的每个日志组手动配置订阅筛选条件。或者，您可以部署一个解决方案，自动将新的日志组订阅到 ElasticSearch 集群。您可以使用同一账户或集中账户将日志流式传输到集 ElasticSearch 群。使用同一账户将日志流式传输到 ElasticSearch 集群可以帮助工作负载所有者更好地分析和支持其工作负载。

您应该考虑在集中式或共享账户中设置 ElasticSearch 集群，以便汇总账户、区域和应用程序中的日志。例如， AWS Control Tower 设置用于集中日志记录的日志存档帐户。在中创建新账户后 AWS Control Tower，其 AWS CloudTrail 和 AWS Config 日志将传送到此集中式账户中的 S3 存储桶。所检测的日志 AWS Control Tower 用于配置、更改和审核日志。

要使用 Amazon S OpenSearch ervice 建立集中式应用程序日志分析解决方案，您可以将一个或多个集中式 Amazon S OpenSearch ervice 集群部署到您的集中式日志账户，并将其他账户中的日志组配置为将日志流式传输到集中式 Amazon S OpenSearch ervice 集群。

您可以创建单独的 Amazon S OpenSearch ervice 集群来处理可能分布在您的账户中的不同应用程序或云架构层。使用单独的 Amazon S OpenSearch ervice 集群可以帮助您降低安全和可用性风险，而拥有通用 Amazon S OpenSearch ervice 集群可以更轻松地在同一集群中搜索和关联数据。

# 带有的警报选项 CloudWatch

对重要指标进行一次性自动分析可帮助您在问题影响工作负载之前检测和解决问题。 CloudWatch 通过使用特定时间段内的多个统计数据，可以轻松绘制和比较多个指标。您可以使用 CloudWatch 搜索具有所需维度值的所有指标，以找到分析所需的指标。

我们建议您在开始使用指标捕获方法时加入一组初始指标和维度，用作监控工作负载的基准。随着时间的推移，工作负载会逐渐成熟，您可以添加其他指标和维度来帮助您进一步分析和支持它。您的应用程序或工作负载可能使用多个 AWS 资源并有自己的自定义指标，您应将这些资源分组到一个命名空间下，以便于识别。

您还应考虑如何将日志和监控数据关联起来，以便可以快速识别相关的日志和监控数据来诊断特定问题。您可以使用AWS X-Ray 跟踪映射关联跟踪、指标、日志和警报，以诊断问题。您还应考虑在工作负载日志中的指标和标识符中加入其他维度，以帮助您快速搜索和识别系统和服务中的问题。

## 使用 CloudWatch 警报进行监控和报警

您可以使用CloudWatch 警报来减少工作负载或应用程序中的手动监控。首先，您应查看为每个工作负载组件捕获的指标，并确定每个指标的相应阈值。请务必确定在突破阈值时必须通知哪些团队成员。您应该建立和定位通讯组，而不是单个团队成员。

CloudWatch 警报可以与您的服务管理解决方案集成，以自动创建新工单并运行操作工作流程。例如，AWS 为ServiceNow和提供了 AWS 服务管理连接器AWS 服务管理连接器来帮助您快速设置集成。这种方法对于确保已发出的警报得到确认并与这些产品中可能已经定义的现有操作工作流程保持一致至关重要。

您还可以为同一指标创建多个具有不同阈值和评估周期的警报，这有助于建立上报流程。例如，如果您有一个跟踪客户订单的`OrderQueueDepth`指标，则可以在短短的一分钟内定义一个较低的阈值，通过电子邮件或 Slack 通知应用程序团队成员。您还可以在相同阈值下为同一指标定义另一个警报，持续15分钟，并向应用程序团队和应用程序团队的负责人发送页面、电子邮件和通知。最后，您可以为30分钟内的硬平均阈值定义第三个警报，该警报通知上级管理层并通知之前通知的所有团队成员。创建多个警报可帮助您针对不同的条件采取不同的操作。您可以从简单的通知流程开始，然后根据需要对其进行调整和改进。

## 使用 CloudWatch 异常检测进行监控和报警

如果您不确定要应用于特定指标的阈值，或者希望警报根据观察到的历史值自动调整阈值，则可以使用CloudWatch 异常检测。 CloudWatch 异常检测对于可能有定期、可预测的活动变化的指标特别有

用，例如，当日送达的每日采购订单在截止时间之前增加。异常检测可实现自动调整的阈值，并有助于减少误报。您可以为每个指标和统计数据启用异常检测，并配置 CloudWatch 为根据异常值发出警报。

例如，您可以为 EC2 实例上的`CPUUtilization`指标和`AVG`统计数据启用异常检测。然后，异常检测使用最多 14 天的历史数据来创建机器学习 (ML) 模型。您可以创建具有不同异常检测波段的多个警报，以建立警报升级流程，类似于创建具有不同阈值的多个标准警报。

有关本节的更多信息，请参阅 CloudWatch 文档中的基于异常检测创建 CloudWatch警报。

## 跨多个地区和账户发出警报

应用程序和工作负载所有者应为跨多个区域的工作负载创建应用程序级警报。我们建议在部署您的工作负载的每个账户和区域内创建单独的警报。您可以使用与账户和区域无关以及模板来部署带有所需警报的应用程序资源，从而简化和自动化此过程。模板您可以将警报操作配置为针对常见的亚马逊简单通知服务 (Amazon SNS) Simple Not CloudFormation StackSets ification Service 主题，这意味着无论账户或区域如何，都将使用相同的通知或补救操作。

在多账户和多区域环境中，我们建议您为您的账户和区域创建汇总警报，以便通过使用和汇总指标（例如`CPUUtilization`所有 EC2 实例的平均值）来监控账户 CloudFormation StackSets 和区域问题。

您还应考虑为为捕获的标准 CloudWatch 指标和日志配置的每个工作负载创建标准警报。例如，您可以为每个 EC2 实例创建单独的警报，用于监控 CPU 利用率指标，并在每天平均 CPU 利用率超过 80% 时通知中央运营团队。您还可以创建标准警报，每天监控低于 10% 的平均 CPU 使用率。这些警报可帮助中央运营团队与特定的工作负载所有者合作，在需要时更改 EC2 实例的大小。

## 使用 EC2 实例标签自动创建警报

为您的 EC2 实例创建一组标准警报可能非常耗时、不一致且容易出错。您可以使用该 amazon-cloudwatch-auto-alarms解决方案自动为您的 EC2 实例创建一组标准 CloudWatch警报并基于 EC2 实例标签创建自定义警报，从而加快警报创建过程。该解决方案无需手动创建标准警报，并且在使用诸如之类的工具大规模迁移EC2实例时非常有用 CloudEndure。您也可以使用部署此解决方案 CloudFormation StackSets 以支持多个区域和账户。有关更多信息，请参阅 AWS 博客上的使用标签为 Amazon EC2 实例创建和维护 Amazon CloudWatch 警报。

# 监控应用程序和服务可用性

CloudWatch 帮助您监控和分析应用程序和工作负载的性能和运行时方面。您还应该监控应用程序和工作负载的可用性和可访问性方面。你可以通过使用主动监控方法和 Amazon Route 53 运行状况检查和 S CloudWatch ynthetics 来实现这一目标。

如果要监控通过 HTTP 或 HTTPS 与网页的连接，或者通过 TCP 监控与公共域名系统 (DNS) 名称或 IP 地址的网络连接，则可以使用 Route 53 运行状况检查。Route 53 运行状况检查以十秒或 30 秒为间隔从您指定的区域启动连接。您可以选择运行状况检查的多个区域，每项运行状况检查独立运行，并且必须至少选择三个区域。如果 HTTP 或 HTTPS 请求的响应正文出现在为运行状况检查评估返回的前 5,120 字节数据中，则可以搜索该子字符串以查找该子字符串。如果 HTTP 或 HTTPS 请求返回 2xx 或 3xx 响应，则该请求被视为健康请求。Route 53 运行状况检查可用于通过检查其他运行状况检查的运行状况来创建复合运行状况检查。如果您有多个服务终端节点，并且想要在其中一个服务终端节点变得不健康时执行相同的通知，则可以执行此操作。如果您将 Route 53 用于 DNS，则可以将 Route 53 配置为在运行状况检查变得不健康时故障转移到另一个 DNS 条目。对于每个关键工作负载，您都应考虑为对正常操作至关重要的外部终端节点设置 Route 53 运行状况检查。Route 53 运行状况检查可以帮助您避免在应用程序中写入故障转移逻辑。

CloudWatch synthetics 允许您将金丝雀定义为脚本，以评估工作负载的运行状况和可用性。加那利群岛是用 Node.js 或 Python 编写的脚本，通过 HTTP 或 HTTPS 协议运行。它们以 Node.js 或 Python 为框架在您的账户中创建 Lambda 函数。您定义的每个金丝雀都可以对不同的端点执行多个 HTTP 或 HTTPS 调用。这意味着您可以监控一系列步骤的运行状况，例如用例或具有下游依赖关系的端点。加那利群岛创建的 CloudWatch 指标包括已运行的每个步骤，因此您可以单独发出警报和衡量不同的步骤。尽管与 Route 53 运行状况检查相比，加那利群岛需要更多的规划和精力来开发，但它们为您提供了一种高度可定制的监控和评估方法。加那利群岛还支持在您的虚拟私有云 (VPC) 中运行的私有资源，这使得它们非常适合在终端节点没有公有 IP 地址时进行可用性监控。您也可以使用加那利群岛监控本地工作负载，前提是您可以从 VPC 内部连接到终端节点。当您的工作负载包含本地端点时，这一点尤其重要。

# 使用跟踪应用程序 AWS X-Ray

通过您的应用程序发出的请求可能包括对本地服务器、Amazon EC2、容器或 Lambda 中运行的数据库、应用程序和 Web 服务的调用。通过实现应用程序跟踪，您可以快速确定使用分布式组件和服务的应用程序中出现问题的根本原因。您可以使用AWS X-Ray跨多个组件跟踪您的应用程序请求。当请求流经您的应用程序组件并且每个组件都表示为一个区段时，X-Ray 会在服务图上对请求进行采样和可视化。X-Ray 会生成跟踪标识符，以便您可以在请求流经多个组件时关联请求，从而帮助您端到端查看请求。您可以通过添加注释和元数据来进一步增强这一点，以帮助唯一搜索和识别请求的特征。

我们建议您使用 X-Ray 对应用程序中的每台服务器或端点进行配置和检测。X-Ray 是通过调用 X-Ray 服务在应用程序代码中实现的。X-Ray 还 AWS SDKs 提供多种语言，包括自动向 X-Ray 发送数据的仪器化客户端。X-Ray SDKs 为用于调用其他服务（例如 HTTP、MySQL、PostgreSQL 或 MongoDB）的常用库提供补丁。

X-Ray 提供了一个 X-Ray 守护程序，你可以在 Amazon EC2 和 Amazon ECS 上安装和运行该守护程序，将数据中继到 X-Ray。X-Ray 为您的应用程序创建跟踪，这些跟踪从运行为请求提供服务的 X-Ray 守护程序的服务器和容器中捕获性能数据。通过修补软件开发工具包，X-Ray 会自动将您对 AWS 服务（例如亚马逊 DynamoDB）的调用计量为子分段。 AWS X-Ray 还可以自动与 Lambda 函数集成。

如果您的应用程序组件调用无法配置和安装 X-Ray 守护程序或检测代码的外部服务，则可以创建子分段来封装对外部服务的调用。如果您使用的是，X-Ray 会将 CloudWatch 日志和指标与您的应用程序跟踪关联起来 AWS X-Ray SDK for Java，这意味着您可以快速分析请求的相关指标和日志。

## 部署 X-Ray 守护程序以跟踪亚马逊 EC2 上的应用程序和服务

您需要在运行应用程序组件或微服务的 EC2 实例上安装和运行 X-Ray 守护程序。在配置 EC2 实例时，您可以使用用户数据脚本部署 X-Ray 守护程序，或者如果您创建自己的 AMI，则可以将其包含在 AMI 构建流程中。当 EC2 实例为临时实例时，这可能特别有用。

您应该使用状态管理器来确保 X-Ray 守护程序始终安装在您的 EC2 实例上。对于 Amazon EC2 Windows 实例，你可以使用 Systems Manager AWS-RunPowerShellScript 文档来运行下载和安装 X-Ray 代理的 Windows 脚本。对于 Linux 上的 EC2 实例，您可以使用 AWS-RunShellScript 文档来运行 Linux 脚本，该脚本将代理下载并安装为服务。

您可以使用 Systems Manager AWS-RunRemoteScript 文档在多账户环境中运行脚本。您必须创建可从所有账户访问的 S3 存储桶，如果您使用，我们建议您使用基于组织的存储桶策略创建一个 S3 存储

桶。 AWS Organizations然后，您将脚本上传到 S3 存储桶，但要确保您的 EC2 实例的 IAM 角色有权访问存储桶和脚本。

您也可以将状态管理器配置为将脚本关联到安装了 X-Ray 代理的 EC2 实例。由于您的所有 EC2 实例可能不需要或不使用 X-Ray，因此您可以使用实例标签来定位关联。例如，您可以根据`InstallAWSXRayDaemonWindows`或`InstallAWSXRayDaemonLinux`标签的存在来创建状态管理器关联。

## 部署 X-Ray 守护程序以跟踪亚马逊 ECS 或 Amazon EKS 上的应用程序和服务

您可以将 X-Ray 守护程序部署为基于容器的工作负载（例如 Amazon ECS 或 Amazon EKS）的边车容器。然后，如果您使用 Amazon ECS，则您的应用程序容器可以通过容器链接连接到您的边车容器；如果您使用 aw svpc 网络模式，则该容器可以直接连接到本地主机上的边车容器。

对于 Amazon EKS，您可以在应用程序的 pod 定义中定义 X-Ray 守护程序，然后您的应用程序可以在您指定的容器端口上通过本地主机连接到该守护程序。

## 配置 Lambda 以跟踪对 X-Ray 的请求

您的应用程序可能包括对 Lambda 函数的调用。您无需为 Lambda 安装 X-Ray 守护程序，因为守护程序进程完全由 Lambda 管理，无法由用户进行配置。您可以在 X-Ray 控制台中使用 AWS 管理控制台并选中 “活动跟踪” 选项，为 Lambda 函数启用该功能。

如需进一步检测，您可以将 X-Ray SDK 与 Lambda 函数捆绑在一起，以记录传出呼叫并添加注释或元数据。

## 针对 X-Ray 应用程序进行仪器测试

您应该评估与应用程序编程语言一致的 X-Ray SDK，并对应用程序对其他系统进行的所有调用进行分类。查看您选择的库提供的客户端，看看 SDK 是否可以自动对您的应用程序的请求或响应进行仪器跟踪。确定 SDK 提供的客户端是否可用于其他下游系统。对于您的应用程序调用但无法使用 X-Ray 进行检测的外部系统，您应该创建一个自定义子分段，以便在跟踪信息中捕获和识别它们。

在分析应用程序时，请务必创建注释以帮助您识别和搜索请求。例如，您的应用程序可能会使用客户标识符（例如）`customer id`，或者根据不同用户在应用程序中的角色对其进行细分。

您最多可以为每条轨迹创建 50 个注释，但只要区段文档不超过 64 千字节，您就可以创建一个包含一个或多个字段的元数据对象。您应该有选择地使用注解来查找信息，并使用元数据对象提供更多上下文，以帮助在找到请求后对其进行故障排除。

# 配置 X-Ray 采样规则

通过自定义采样规则，您可以控制记录的数据量并修改采样行为，而无需修改或重新部署代码。采样规则向 X-Ray SDK 告知要为一系列条件记录的请求数。默认情况下，X-Ray SDK 每秒记录第一个请求以及任何其他请求的百分之五。每秒一个请求是容器。这可确保只要服务正在处理请求，就会每秒至少记录一个跟踪。百分之五是对超出水库规模的额外请求进行采样的速率。

您应查看并更新默认配置，以确定适合您账户的值。在开发、测试、性能测试和生产环境中，您的要求可能会有所不同。您的应用程序可能要求根据收到的流量或严重程度制定自己的采样规则。您应该从基线开始，并定期重新评估该基准是否符合您的要求。

# 仪表板和可视化效果 CloudWatch

仪表板可帮助您快速关注应用程序和工作负载的关注领域。 CloudWatch提供自动仪表板，您还可以轻松创建使用 CloudWatch 指标的仪表板。 CloudWatch 与单独查看指标相比，仪表板可以提供更多的见解，因为它们可以帮助您关联多个指标并识别趋势。例如，包含已接收订单、内存、CPU 利用率和数据库连接的仪表板可以帮助您在订单数量增加或减少时关联多个 AWS 资源的工作负载指标的变化。

您应该在账户和应用程序级别创建仪表板，以监控工作负载和应用程序。您可以从使用 CloudWatch 自动仪表板开始，这些仪表板是预先配置了 AWS 服务特定指标的服务级别仪表板。自动服务仪表板显示服务的所有标准 CloudWatch 指标。自动仪表板会绘制用于每个服务指标的所有资源图表，并帮助您快速识别账户中的异常值资源。这可以帮助您识别利用率高和低的资源，从而帮助您优化成本。

## 创建跨服务仪表板

您可以通过查看服务的自动服务级别仪表板并使用 “操作” 菜单中的 “添加到仪表板” 选项来创建跨 AWS 服务仪表板。然后，您可以将其他自动仪表板中的指标添加到新的仪表板中，并删除指标以缩小仪表板的关注范围。您还应该添加自己的自定义指标来跟踪关键观察结果（例如，收到的订单或每秒的交易量）。创建自己的自定义跨服务仪表板可帮助您专注于与工作负载最相关的指标。我们建议您创建账户级别的跨服务仪表板，以涵盖关键指标并显示账户中的所有工作负载。

如果您有供云运营团队使用的中央办公空间或公共区域，则可以在大型电视显示器上以全屏模式显示 CloudWatch 仪表板，并自动刷新。

## 创建特定于应用程序或工作负载的仪表板

我们建议您创建特定于应用程序和工作负载的仪表板，重点关注生产环境中每个关键应用程序或工作负载的关键指标和资源。应用程序和工作负载特定的仪表板侧重于您的自定义应用程序或工作负载指标以及影响其性能的重要 AWS 资源指标。

您应该定期评估和自定义 CloudWatch 应用程序或工作负载仪表板，以便在事件发生后跟踪关键指标。引入或停用功能时，您还应该更新特定于应用程序或工作负载的仪表板。除了记录和监控之外，更新工作负载和特定于应用程序的仪表板应该是持续提高质量的必要活动。

## 创建跨账户或跨区域控制面板

AWS 资源主要是区域性的，指标、警报和仪表板特定于部署资源的区域。这可能需要您更改区域以查看跨区域工作负载和应用程序的指标、仪表板和警报。如果您将应用程序和工作负载分成多个帐户，则

可能还需要重新进行身份验证并登录每个帐户。但是， CloudWatch 支持从单个账户查看跨账户和跨区域数据，这意味着您可以在单个账户和区域中查看指标、警报、仪表板和日志小部件。如果您有集中式日志和监控帐户，这将非常有用。

账户所有者和应用程序团队所有者应为账户特定的跨区域应用程序创建仪表板，以便在集中位置有效地监控关键指标。 CloudWatch控制面板自动支持跨区域小组件，这意味着您无需进一步配置即可创建包含来自多个区域的指标的控制面板。

一个重要的例外是 Logs Insights 微件，因为只能显示您当前登录的账户和地区的日志数据。CloudWatch 您可以使用指标筛选器从日志中创建特定于区域的指标，这些指标可以显示在跨区域控制面板上。然后，当您需要进一步分析这些日志时，可以切换到特定区域。

运营团队应创建集中式控制面板，用于监控重要的跨账户和跨区域指标。例如，您可以创建一个跨账户控制面板，其中包含每个账户和地区的聚合 CPU 使用率。您还可以使用指标数学来汇总多个账户和地区的数据并控制面板数据。

## 使用公制数学来微调可观察性和警报

您可以使用指标数学来帮助计算与您的工作负载相关的格式和表达式的指标。计算出的指标可以保存并在仪表板上查看，以便进行跟踪。例如，标准 Amazon EBS 卷指标提供了在特定时间段内执行的读取 (`VolumeReadOps`) 和写入 () 操作的数量。`VolumeWriteOps`

但是， AWS 提供了有关 IOPS 中亚马逊 EBS 卷性能的指南。通过将和除以为这些指标选择的时间段，您可以用公制数学绘制`VolumeReadOps`并计算出您的 Amazon EBS 交易量的`VolumeWriteOps` IOPS。

在此示例中，我们对周期内的 IOPS 求和，然后除以周期长度得出 IOPS。然后，您可以针对此指标数学表达式设置警报，以便在卷的 IOPS 接近其卷类型的最大容量时提醒您。有关使用指标数学来监控带有指标的亚马逊弹性文件系统 (Amazon EFS) 文件系统的更多信息和示例，请参阅 AWS 博客上的亚马逊 CloudWatch CloudWatch 指标数学简化了对您的 Amazon EFS 文件系统的近乎实时的监控等。

## 使用适用于亚马逊 ECS、Amazon EKS 和 Lambda 的自动控制面板以及 CloudWatchContainer 洞察和 Lambda Insights CloudWatch

CloudWatch 容器见解为在 Amazon ECS 和 Amazon EKS 上运行的容器工作负载创建动态的自动控制面板。您应该启用 Container Insights，使其能够观察 CPU、内存、磁盘、网络和诊断信息，例如容器重启故障。Container Insights 生成动态仪表板，您可以在集群、容器实例或节点、服务、任务、容器

和单个容器级别快速筛选这些仪表板。Cont ainer Insights 是在集群和节点或容器实例级别配置的，具体取决于 AWS 服务。

与容器见解类似， CloudWatch Lambda Insights 为您的 Lambda 函数创建动态的自动控制面板。该解决方案收集、汇总和汇总系统级指标，包括 CPU 时间、内存、磁盘和网络。它还会收集、汇总和汇总诸如冷启动和 Lambda 工作程序关闭之类的诊断信息，以帮助您隔离和快速解决 Lambda 函数的问题。Lambda 在函数级别启用，不需要任何代理。

容器见解和 Lambda Insights 还可以帮助您快速切换到应用程序或性能日志、X-Ray 跟踪和服务地图，以可视化您的容器工作负载。它们都使用 CloudWatch 嵌入式指标格式来捕获 CloudWatch 指标和性能日志。

您可以使用容器见解和 Lambda Insights 捕获的指标为工作负载创建共享 CloudWatch 控制面板。为此，您可以通过 Contain CloudWatch er Insights 筛选和查看自动仪表板，然后选择允许您将显示的指标添加到标准 CloudWatch仪表板的 “添加到控制面板” 选项。然后，您可以删除或自定义指标，并添加其他指标以正确表示您的工作量。

# CloudWatch 与 AWS 服务集成

AWS 提供了许多服务，其中包括用于日志记录和指标的其他配置选项。这些服务通常使您能够为 CloudWatch 日志输出配置日志，为 CloudWatch 指标输出配置指标。用于提供这些服务的底层基础设施由管理 AWS 且无法访问，但是您可以使用已配置服务的日志和指标选项来获得进一步的见解并解决问题。例如，您可以将 VPC 流日志发布到 CloudWatch或配置要向其发布日志的亚马逊关系数据库服务 (Amazon RDS) 实例 CloudWatch。

大多数 AWS 服务通过集成来记录其 API 调用 AWS CloudTrail。 CloudTrail 还支持与 CloudWatch 日志集成，这意味着您可以搜索和分析 AWS 服务中的活动。您还可以使用或 Amazon EventBridge 创建和配置自动化和通知，其中包含 AWS 服务中执行的特定操作的事件规则。某些服务直接与集成 EventBridge。您也可以创建通过交付的事件 CloudTrail。

# 用于生成控制面板和可视化的 Amazon Managed Grafana

Amazon Manag ed Grafana 可用于观察和可视化您的工作负载。 AWS Amazon Managed Grafana 可帮助您大规模可视化和分析您的运营数据。Grafan a 是一个开源分析平台，无论指标存储在何处，都可以帮助您查询、可视化、提醒和了解指标。如果您的组织已经使用 Grafana 对现有工作负载进行可视化，并且您想将覆盖范围扩展到工作负载，则 Amazon Managed Grafana 特别有用。 AWS 您可以通过将 Amazon Managed Grafana CloudWatch 添加为数据源来使用，这意味着您可以使用指标创建可视化效果。 CloudWatchAmazon Managed Grafana AWS Organizations 支持，您可以使用来自多个账户和地区的指标 CloudWatch 来集中控制面板。

下表提供了使用 Amazon Managed Grafana 代替 CloudWatch 控制面板的优势和注意事项。根据最终用户、工作负载和应用程序的不同要求，混合方法可能是合适的。

| | |
|---|---|
| 创建与亚马逊托管 Grafana 和开源 Grafana 支持的数据源集成的可视化效果和控制面板 | Amazon Managed Grafana 可帮助您根据许多不同的数据源（包括指标）创建可视化和控制面板。 CloudWatch Amazon Managed Grafana 包含许多内置数据源， AWS 涵盖服务、开源软件和 COTS 软件。有关这方面的更多信息，请参阅 Amazon Managed Grafana 文档中的内置数据源。您还可以通过将工作空间升级到 Grafana Enterprise 来增加对更多数据源的支持。Grafana 还支持数据源插件，允许您与不同的外部系统进行通信。 CloudWatch 仪表板需要 CloudWatch指标或 CloudWatch Logs Insights 查询才能在 CloudWatch仪表板上显示数据。 |
| 将仪表板解决方案的访问权限与 AWS 账户访问权限分开管理 | Amazon Managed Grafana 需要使用 AWS IAM Identity Center （ IAM 身份中心） AWS Organizations 以及进行身份验证和授权。这使您能够使用您可能已经在 IAM 身份中心中使用的联合身份验证来向 Grafana 对用户进行身份验证，或者。 AWS Organizations但是，如果您没有使用 IAM 身份中心或 AWS Organizations，则将其设置为亚马逊托管 Grafana 设置过程的一部分。如果您的组织限制了 IAM Identity |

| | |
|---|---|
| | Center 的使用或，这可能会成为一个问题 AWS Organizations。 |
| 通过集成，跨多个账户和地区摄取和访问数据 AWS Organizations | Amazon Managed Grafana AWS Organizations 与集成，使您能够在所有账户中读取 AWS 来自 OpenSearch 亚马逊服务 CloudWatch 等来源的数据。这使得创建仪表板成为可能，这些仪表板使用您的账户中的数据来显示可视化效果。要自动启用跨数据访问权限 AWS Organizations，您需要在管理账户中设置您的 Amazon Managed Grafana 工作空间。 AWS Organizations 根据管理账户AWS Organizations 的最佳实践，不建议这样做。相比之下，它 CloudWatch 还支持跨账户、跨区域的指标控制面板。 CloudWatch |
| 使用开源社区中提供的高级可视化控件和 Grafana 定义 | Grafana 提供了大量可视化效果，供您在创建仪表板时使用。还有一个由社区贡献的大型仪表板库，您可以根据自己的要求对其进行编辑和重复使用。 |
| 在新的和现有的 Grafana 部署中使用仪表板 | 如果您已经在使用 Grafana，则可以从 Grafana 部署中导入和导出控制面板，并对其进行自定义，以便在亚马逊托管 Grafana 中使用。Amazon Managed Grafana 允许您将 Grafana 作为仪表板解决方案进行标准化。 |
| 工作空间、权限和数据源的高级设置和配置 | Amazon Managed Grafana 使您能够创建多个 Grafana 工作空间，这些工作空间拥有自己的一组已配置的数据源、用户和策略。这可以帮助您满足更高级的用例要求以及高级安全配置。如果您的团队还没有这些技能，则高级功能可能需要他们积累使用 Grafana 的经验。 |

# 使用 CloudWatch FAQ 设计和实现日志记录和监控

本节提供了有关设计和实施日志和监控解决方案的常见问题的答案 CloudWatch。

## 我在哪里存储我的 CloudWatch 配置文件？

Amazon EC2 的 CloudWatch 代理可以应用存储在配置目录中的多个 CloudWatch 配置文件。理想情况下，您应该将 CloudWatch配置存储为一组文件，因为您可以进行版本控制并在多个帐户和环境中再次使用它们。有关这方面的更多信息，请参阅本指南的管理 CloudWatch 配置部分。或者，您可以将配置文件存储在上的存储库中， GitHub 并在配置新的 EC2 实例时自动检索配置文件。

## 警报发出后，如何在我的服务管理解决方案中创建工单？

您将服务管理系统与亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题集成，并将警报配置 CloudWatch 为在警报发出时通知 SNS 主题。您的集成系统会收到 SNS 消息，并可以使用您的服务管理系统 APIs 或 SDKs创建票证。

## 如何使用 CloudWatch 来捕获容器中的日志文件？

可以将 Amazon ECS 任务和 Amazon EKS 容器配置为自动将 STDOUT 和 STDERR 输出发送到。CloudWatch记录容器化应用程序的推荐方法是让容器将其输出发送到 STDOUT 和 STDERR。十二因子应用程序宣言中也对此进行了介绍。

但是，如果您想向其发送特定的日志文件， CloudWatch 则可以在您的 Amazon EKS 容器或 Amazon ECS 任务定义中将卷挂载到应用程序写入其批文件的位置，然后使用 Fluentd 或 Fluent Bit 的边车容器将日志发送到。 CloudWatch您应该考虑将容器中的特定日志文件符号链接到`/dev/stdout`和`/dev/stderr`。有关这方面的更多信息，请参阅 Docker 文档中的查看容器或服务的日志。

## 如何监控 AWS 服务的运行状况问题？

您可以使用AWS Health Dashboard来监控 AWS 运行状况事件。您也可以参考aws-health-tools GitHub 存储库，获取与 AWS 健康事件相关的自动化解决方案示例。

## 在没有代理支持的情况下，如何创建自定义 CloudWatch 指标？

您可以使用嵌入式指标格式来提取指标。 CloudWatch您还可以使用 AWS SDK（例如 put_metric_data）AWS CLI（例如 put-metric-data）或 AWS API（例如 PutMetricData）来创建自定义指标。您应

该考虑如何长期维护任何自定义逻辑。一种方法是使用集成嵌入式指标格式支持的 Lambda 来创建指标，并使用 CloudWatch 事件事件计划规则来确定指标的周期。

## 如何将我现有的日志和监控工具与集成 AWS？

您应参考软件或服务供应商提供的指南，以便与集成 AWS。您可以使用代理软件、SDK 或 API 向他们的解决方案发送日志和指标。您也可以使用根据供应商规格配置的开源解决方案，例如 Fluentd 或 Fluent Bit。您还可以将 AWS 软件开发工具包和 CloudWatch 日志订阅筛选器与 Lambda 和 Kinesis Data Streams 配合使用来创建自定义日志处理器和采集器。最后，如果您使用多个账户和区域，您还应该考虑如何集成软件。

# 资源

## 简介

- AWS Well-Architected

## 目标业务成果

- logging-monitoring-apg-guide-示例
- 云计算的六大优势

## 规划您的 CloudWatch 部署

- AWS Organizations 术语和概念
- AWS Systems Manager 快速设置
- 使用 CloudWatch代理从 Amazon EC2 实例和本地服务器收集指标和日志
- cloudwatch-config-s3bucket.yaml
- 使用向导创建 CloudWatch 代理配置文件
- Enterprise DevOps：为什么要运行自己构建的东西
- Exporting log data to Amazon S3
- Amazon 服务中的精细访问控制 OpenSearch
- Lambda 配额
- 手动创建或编辑 CloudWatch 代理配置文件
- 通过订阅实时处理日志数据
- 可供构建的工具 AWS

## 为 EC2 实例和本地服务器配置 CloudWatch 代理

- 亚马逊 EC2 指标维度
- 可突发性能实例
- CloudWatch 代理预定义指标集

- 使用 procstat 插件收集流程指标
- 为 procsta CloudWatch t 配置代理
- 管理对您的 EC2 实例的详细监控
- 摄取高基数日志并生成带有嵌入式指标格式的指标 CloudWatch
- 使用日志组和日志流
- 列出您的实例的可用 CloudWatch 指标
- PutLogEvents
- 使用 collectd 检索自定义指标
- 使用 StatsD 检索自定义指标

## CloudWatch 适用于 Amazon EC2 和本地服务器的代理安装方法

- 在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色
- 为混合环境创建托管实例激活
- 创建用于 CloudWatch 代理的 IAM 角色和用户
- 使用命令行下载并配置 CloudWatch 代理
- 如何将使用 Systems Manager 代理和统一 CloudWatch 代理的本地服务器配置为仅使用临时证书？
- 堆栈集操作的先决条件
- 使用竞价型实例

## 在 Amazon ECS 上进行日志记录和监控

- amazon-cloudwatch-logs-for-fluentbit
- 亚马逊 ECS CloudWatch 指标
- Amazon ECS Container Insights 指标
- 亚马逊 ECS 容器代理
- 亚马逊 ECS 启动类型
- 部署 CloudWatch 代理以收集 Amazon ECS 上的 EC2 实例级别指标
- ecs_cluster_with_cloudwatch_linux.yaml
- ecs_cw_emf_example
- ecs_firelense_emf_example

- ecs-task-nginx-firelense.json
- 检索亚马逊 ECS 优化的 AMI 元数据
- 使用 awslogs 日志驱动程序
- 使用客户端库生成嵌入式指标格式日志

## Amazon EKS 中的日志记录和监控

- Amazon EKS 控制面板日志记录
- amazon_eks_managed_node_group_launch_config.yaml
- Amazon EKS 节点
- amazon-eks-nodegroup.yaml
- 亚马逊 EKS 服务等级协议
- 容器洞察 Prometheus 指标监控
- 使用 Prometheus 控制平面指标
- Fargate 日志记录
- Fargate 上适用于亚马逊 EKS 的 Fluent Bit
- 在 Fargate 上使用 Amazon EKS 时如何捕获应用程序日志
- 安装 CloudWatch 代理以收集 Prometheus 指标
- 安装 Kubernetes 指标服务器
- kubernetes /仪表板
- Kubernetes Horizontal Pod Autoscaler
- Kubernetes 控制平面组件
- Kubernetes 吊舱
- 启动模板支持
- 托管节点组
- 托管节点更新行为
- 指标服务器
- 使用 Prometheus 和 Grafana 在 Fargate 上监控 Amazon EKS
- prometheus_jmx
- prometheus /jmx_exporter
- 抓取其他 Prometheus 来源并导入这些指标

- 自行管理的节点
- 将日志发送到 CloudWatch 日志
- 将 FluentD 设置为 DaemonSet a 以将日志发送到日志 CloudWatch
- 在 Amazon EKS 和 Kubernetes 上设置 Java/JMX 示例工作负载
- 添加新 Prometheus 抓取目标的教程：Prometheus API 服务器指标
- 垂直吊舱自动扩缩器

## 的日志和指标 AWS Lambda

- Lambda 调用错误
- 日志记录 — 适用于 Python 的日志工具
- 使用客户端库生成嵌入式指标格式日志
- 使用 Lambda 函数指标

## 搜索和分析日志 CloudWatch

- Beats 家族
- 弹性 Logstash
- 弹性堆栈
- 将 CloudWatch 日志数据流式传输到 Amazon OpenSearch 服务

## 带有的警报选项 CloudWatch

- amazon-cloudwatch-auto-alarms
- AWS 适用于 Jira 服务管理云的服务管理连接器
- AWS 适用于 Jira 服务管理数据中心的服务管理连接器
- AWS 的服务管理连接器 ServiceNow

## 监控应用程序和服务可用性

- 配置 DNS 故障转移

## 使用跟踪应用程序 AWS X-Ray

- Amazon ECS 任务联网
- 在 X-Ray 控制台中配置采样规则
- 运行 Windows PowerShell 命令或脚本
- 在亚马逊 EC2 上运行 X-Ray 守护程序
- 向 X-Ray 发送跟踪数据
- X-Ray 中的服务图

## 仪表板和可视化效果 CloudWatch

- Amazon M CloudWatch etric Math 简化了对您的 Amazon EFS 文件系统的近乎实时的监控
- 设置 CloudWatch容器见解
- 使用公制数学

## CloudWatch 与 AWS 服务集成

- AWS CloudTrail 支持的服务和集成
- 来自 AWS 服务 Amazon 的活动 EventBridge
- 通过以下方式交付的 AWS 服务事件 AWS CloudTrail
- CloudTrail 使用日志监控 CloudWatch 日志文件
- 将数据库日志发布到 CloudWatch 日志
- 将流日志发布到 CloudWatch 日志

## 用于生成控制面板和可视化的 Amazon Managed Grafana

- 管理账户的最佳实践 AWS Organizations
- 亚马逊 Managed Grafana 的内置数据源
- 中的跨账户和跨区域控制面板 CloudWatch
- Grafana 插件

# 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 |
| --- | --- | --- |
| 更新了日志信息 | 更新了有关登录的部分 AWS Lambda。 | 2023 年 4 月 17 日 |
| 更新了配置信息 | 更新并重命名了有关创建和存储 CloudWatch 配置的部分。 | 2023 年 2 月 9 日 |
| 更新了指标信息 | 更新了 Amazon ECS 指标部分中的自定义应用程序指标信息。 | 2023 年 1 月 31 日 |
| 已删除预览通知 | 亚马逊托管 Grafana 现已正式上市。 | 2022 年 5 月 25 日 |
| 删除部分 | CloudWatch 不再支持 SDK 指标。 | 2022 年 1 月 7 日 |
| 初次发布 | — | 2021 年 4 月 30 日 |

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。