AWS规范性指导

使用 Amazon 设计和实现日志记录和监控 CloudWatch



AWS规范性指导: 使用 Amazon 设计和实现日志记录和监控 CloudWatch

Copyright © 2023 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 拥有的其它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

介绍 1
目标业务成果 3
加快运营准备 3
提高卓越操作 3
提高操作可见性 3
扩展运营并降低管理费用 3
规划 CloudWatch 部署 4
CloudWatch 在集中式或分布式账户中使用 4
管理 CloudWatch 代理配置文件 6
管理 CloudWatch配置 6
示例：将 CloudWatch 配置文件存储在 7
配置 CloudWatch EC2 实例和本地服务器的代理 9
配置 CloudWatch 代理人 9
为 EC2 实例配置日志捕获 10
为 EC2 实例配置指标捕获 11
系统级 CloudWatch 配置 12
配置系统级日志 12
配置系统级指标 13
应用程序级 CloudWatch 配置 14
配置应用程序级日志 14
配置应用程序级指标 14
适用于 Amazon EC2 和本地服务器的 CloudWatch 代理安装方法 16
安装 CloudWatch 使用 Systems Manager 分销商和状态管理器 16
设置州经理和分销商 CloudWatch 代理部署和配置 17
使用 Systems Manager 快速设置并手动更新创建的 Systems Manager 资源 17
使用AWS CloudFormation而不是快速设置 18
在单个账户和区域中使用AWS CloudFormation堆 18
使用多个区域和多个账户中的自定义快速设置AWS CloudFormationStackSets 19
配置本地服务器的注意事 20
临时 EC2 实例的注意事项 20
使用自动化解决方案部署 CloudWatch 代理人 21
部署 CloudWatch 使用用户数据脚本预配实例期间的代理 21
包括 CloudWatch AMI 中的代理 21
Amazon ECS 日志记录和监控 23
CloudWatch使用 EC2 启动类型进行配置 23
EC2 和 Fargate 启动类型的Amazon ECS 容器日志 24
在 Amazon ECS 上FireLens使用自定义日志路由 24
Amazon ECS 的指标 25
在 Amazon ECS 中创建自定义应用程序指标 25
Amazon EKS 上的日志记录和监控 27
日志记录 Amazon EKS 27
Amazon EKS 控制层面日志记录 27
Amazon EKS 节点和应用程序记录 27
在 Fargate 上登录亚马逊 EKS 29
Amazon EKS 和 Kubernetes 指标 29
Kubernetes 控制层面指标 29
Kubernetes 的节点和系统指标 29
应用程序指标 30
Fargate 上的亚马逊 EKS 指标 30
Amazon EKS 上的 Prometheus 监控 31
的日志和指标AWS Lambda 32
Lambda 函数日志记录 32
将日志从发送到其他目的地 CloudWatch 32
Lambda 函数指标 33

系统指标 33
应用程序指标 33
搜索和分析日志 CloudWatch 34
使用 “应用程序洞察” 共同监控和分析 CloudWatch 应用程序 34
使用日志洞察进行 CloudWatch 日志分析 35
使用亚马逊 OpenSearch 服务进行日志分析 37
CloudWatch 提供警报选项 39
使用 CloudWatch 警报以监控和警报 39
使用 CloudWatch 用于监控和报警的异常检测 39
在多个区域和账户中发生警报 39
使用 EC2 实例标签自动创建警报 40
监控应用和服务可用性 41
使用跟踪应用AWS X-Ray 42
部署 X-Ray 守护程序以便在 Amazon EC2 上跟踪应用程和服务 42
在 Amazon ECS 或 Amazon EKS 上部署 X-Ray 守护程序跟踪应用程和服务 42
配置 Lambda 以跟踪对 X-Ray 的请求 43
分析 X-Ray 应用程 43
配置 X-Ray 采样规则 43
使用 CloudWatch 进行仪表板和可视 44
创建跨服务控制面板 44
创建应用程序或工作负载特定仪 44
创建跨账户或跨区域的控制面板 44
使用指标数学来微调可观察性和警报 45
使用适用于 Amazon ECS、Amazon EKS 和 Lambda 的自动控制面板 CloudWatchContainer 见解和 CloudWatch Lambda Insights 45
与 CloudWatch 集成AWS服务 46
亚马逊托管 Grafana 用于仪表板和可视化 47
常见问题 49
在何处存储 CloudWatch 如何配置文件？ 49
警报发出时，如何在我的服务管理解决方案中创建票证？ 49
如何使用？ CloudWatch 要在我的容器中捕获日志文件？ 49
我如何监控健康问题AWS如何？ 49
如何创建自定义 CloudWatch 衡量指标何时不存在代理支持？ 49
如何将我现有的日志记录和监控工具与AWS? 50
资源 51
介绍 51
有针对性的业务成果 51
规划 CloudWatch 部署 51
为 EC2 实例和本地部署服务器配置 CloudWatch 代理 51
CloudWatch Amazon EC2 和本地服务器的代理安装方法 52
Amazon ECS 上的日志记录和监控 52
Amazon EKS 上的日志记录和监控 52
的日志记录和指标AWS Lambda 52
搜索和分析日志 CloudWatch 53
带有警报功能的选项 CloudWatch 53
监控应用程序和服务可用性 53
使用跟踪应用程序AWS X-Ray 54
带有的仪表板和可视化效果 CloudWatch 54
CloudWatch 与AWS服务集成 54
亚马逊管理的 Grafana 用于仪表板制作和可视化 54
文档历史记录 55
..... lvi

使用 Amazon 设计和实现日志记录和监控 CloudWatch

Khurram Nizami，Amazon Web Services (AWS)

2023 年 4 月（文件历史记录 (p. 55)）

本指南可帮助您使用亚马逊 CloudWatch和相关的Amazon Web Services (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Kubernetes服务 (Amazon EKS) 和本地服务器的工作负载设计和实施日志记录和监控。AWS Lambda该指南适用于管理AWS云端工作负载的运营团队、 DevOps 工程师和应用程序工程师。

您的日志记录和监控方法应基于 Well-Architected FrameAWS work 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。架构完善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用于安全或成本优化的日志记录和监控，因为这些主题需要深入评估。有许多支持安全记录和监控的AWS服务，包括、、Amazon Inspector AWS CloudTrailAWS Config、Amaz on Detecti ve、Amazon Macie GuardDuty、亚马逊和AWS Security Hub。您还可以使用AWS Cost ExplorerAWS预算和CloudWatch 计费指标来优化成本。

下表概述了您的日志记录和监控解决方案应解决的六个方面。

捕获和采集日志文件和指标	识别、配置系统和应用程序日志及指标，并将其发送给来自不同来源的AWS服务。
搜索和分析日志	搜索和分析日志以进行操作管理、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，并据此采取行动。
监控应用程序和服务可用性	通过持续监控服务可用性来减少停机时间并提高实现服务级别目标的能力。
追踪应用程序	跟踪系统和外部依赖项中的应用程序请求，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	创建侧重于系统和工作负载的相关指标和观察结果的仪表板，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，提供可靠、可扩展且灵活的解决方案。除了用于监控和分析 CloudWatch 的日志集成外，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。本指南还涵盖了以下用于日志记录和监控的AWS服务：

- AWS Systems Manager分销商、Systems Manager 器、状态管理器和Syst ems Manager 自动化，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CloudWatch和 Synthetics 用于监控应用程序和服务的可用性
- Amazon Managed Prometheus Manageus Manageus Manageus

- AWS X-Ray用于应用程序跟踪和运行时分析
- Amazon Managed Grafana 用于可视化和分析来自多个来源（例如， CloudWatch亚马逊 OpenSearch 服务和亚马逊 Amazon Timestream）的数据

您选择的AWS计算服务也会影响您的日志记录和监控解决方案的实施和配置。例如，Amaz CloudWatch on EC2、亚马逊 ECS、亚马逊 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环境时可观察性有限，这会延迟识别问题并增加故障排除和解决问题所花费的时间。您的日志和监控解决方案至少必须解决操作系统 (OS) 级别日志和指标的系统层问题，以及应用程序日志和指标的应用程序层。该指南提供了一种针对不同计算类型（包括下表中概述的三种计算类型）处理这两个层的推荐方法。

长时间运行且不可变的 EC2 实例	多个AWS区域或账户中多个操作系统 (OS) 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及指标，包括不同配置的示例。
Serverless (无服务器)	您的 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。

本指南提供了一个日志记录和监控解决方案，该解决方案涉及 CloudWatch以下领域的相关AWS服务：

- 规划 CloudWatch 部署 (p. 4)— 规划 CloudWatch 部署的注意事项和集中 CloudWatch 配置指南。
- 配置 CloudWatch EC2 实例和本地服务器的代理 (p. 9)— 系统级和应用程序级日志记录和指标的 CloudWatch 配置详细信息。
- 适用于 Amazon EC2 和本地服务器的 CloudWatch 代理安装方法 (p. 16)— 安装 CloudWatch 代理的方法，包括使用 Systems Manager 在多个区域和账户之间进行自动部署。
- Amazon ECS 日志记录和监控 (p. 23)— 在 Amazon ECS 中配置 CloudWatch 集群级和应用程序级日志记录和指标的指南。
- Amazon EKS 上的日志记录和监控 (p. 27)— 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。
- Amazon EKS 上的 Prometheus 监控 (p. 31)— 介绍普罗米修斯的亚马逊托管服务与Prometheus 的 Contain CloudWatch er InsigPrometheus 监控并进行了比较。
- 的日志和指标AWS Lambda (p. 32)— CloudWatch 为您的 Lambda 函数配置指南。
- 搜索和分析日志 CloudWatch (p. 34)— 使用 Amazon App CloudWatch lication Insights、Lo CloudWatch gs Insights 分析您的日志以及将日志分析扩展到亚马逊 OpenSearch 服务的方法。
- CloudWatch 提供警报选项 (p. 39)— 介绍 CloudWatch 警报和 CloudWatch 异常检测，并提供有关警报创建和设置的指导。
- 监控应用和服务可用性 (p. 41)— 引入并比较 Sy CloudWatch nthetics 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AWS X-Ray (p. 42)— 使用适用于 Amazon EC2、亚马逊 ECS、亚马逊 EKS 和 Lambda 的 X-Ray 进行应用程序跟踪的简介和设置
- 使用 CloudWatch 进行仪表板和可视 (p. 44)— CloudWatch 仪表板简介，用于提高跨AWS工作负载的可观察性。
- 与 CloudWatch 集成AWS服务 (p. 46)— 说明如何与各种AWS服务 CloudWatch 集成。
- 亚马逊托管 Grafana 用于仪表板和可视化 (p. 47)— 介绍亚马逊管理的 Grafana 并将其与 CloudWatch 用于仪表板和可视化的对比。

本指南在这些领域中使用了实施示例，也可以从AWS示例 GitHub 存储库中获取。

目标业务成果

创建一个专为AWS云对于实现云计算的六大优势. 您的日志记录和监控解决方案应该可以帮助您的 IT 组织实现有利于业务流程、业务合作伙伴、员工和客户的业务成果。在实施与AWS框架完善的框架：

加快运营准备

启用日志记录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。如果过分依赖手动流程，运营准备就绪性可能会很快成为瓶颈，而且还可以缩短 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有限。这可能会增加长期停机、客户不满意和业务流程失败的风险。

你可以使用本指南的方法来标准化和自动化你的日志记录和监控AWS云。然后，新的工作负载需要极少的手动准备和干预来生产日志记录和 这还有助于减少为跨多个账户和区域的不同工作负载大规模创建日志记录和监控标准所需的时间和步骤。

提高卓越操作

本指南提供了多种日志记录和监控最佳实践，帮助不同的工作负载实现业务目标卓越操作. 本指南还提供详细示例和开源、可重复使用的模板您可以与基础架构即代码 (iAC) 方法一起使用该方法来实施架构良好的日志记录和监控解决方案AWS服务。提高卓越运营是迭代性的，需要持续改进。该指南就如何持续改进日志记录和监控实践提供了建议。

提高操作可见性

您的业务流程和应用程序可能受到不同的 IT 资源的支持，并托管在不同的计算类型上，无论是在本地还是在AWS云。您的运营可见性可能受到日志记录和监控策略实施不一致和不完整的限制。采用全面的日志记录和监控方法可帮助您快速识别、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方法，以提高完整的运营可见性并缩短解决 (MTTR) 故障的平均时间。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 (SLA) 要求。

扩展运营并降低管理费用

您可以通过本指南扩展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的方法和示例（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）。当您的云采用成熟和增长，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 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。
- 必须记录和监控的计算类型和操作系统。
- 是否既有本地位置，又有AWS基础设施。
- 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。
- 必须与您的日志记录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，应识别和应用对日志、监控和警报的更新。然后可以主动发现这些问题，并在将future 加以预防。

您必须确保始终如一地安装和配置用于捕获和采集日志和指标的软件和服务。既定的日志记录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，用于不同的领域（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 来捕获和提取多个操作系统和计算类型的日志和指标。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的操作系统和环境进行安装和配置的软件代理。以下部分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 (p. 4)
- 管理 CloudWatch 代理配置文件 (p. 6)

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从多个账户和区域捕获日志和指标。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单个账户来捕获日志和指标。通常，多账户和多区域部署需要混合方法，以支持安全、分析、运营和工作负载所有者的要求。

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领域。

账户结构	您的组织可能有多个单独的帐户（例如，非生产和生产工作负载的帐户），或者针对特定环境中的单个应用程序的数千个帐户。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。这使他们能够在记录和监控中发挥积极作用。我们还建议您使用单独的日志帐户聚合所有工作负载日志，以进行分析、聚合、趋势和集中化操作。单独的日志账户也可用于安全、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。日志应保留在工作负载的帐户中，以便于访问和故障排除。如果将日志和指标维护在与工作负载不同的账户中，则用户可能需要定期在不同账户之间切换。

	使用集中式帐户向授权用户提供日志信息，而无需授予工作负载帐户的访问权限。这可以简化分析工作负载的访问要求，在这些工作负载中，需要聚合在多个账户中运行的工作负载。集中式日志账户还可以有其他搜索和聚合选项，例如 Amazon Serv OpenSearch ice 集群。Amazon S erv OpenSearch ice 为您的日志提供直至字段级别的精细访问控制。当您拥有需要专门访问和权限的敏感或机密数据时，精细的访问控制非常重要。
运营	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，需要访问日志才能进行监控。集中记录和监控可以更轻松地识别趋势、搜索、汇总和对所有账户和工作负载进行分析。如果您的组织使用 “你构建，你运行” 的方法，那么工作负载所有者需要在他们的账户中记录和监控信息。DevOps除了分布式工作负载所有权外，可能需要混合方法来满足中央操作和分析的需求。
Environment	您可以选择将日志和指标托管在生产账户的中心位置，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不同的账户中，具体取决于安全要求和账户架构。这有助于防止生产期间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 订阅筛选器实时处理日志。您可以使用订阅过滤器将日志实时传输到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使用日志和指标，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 Kinesis Dat a Firehose hose — Kinesis Data 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。您不需要管理 Amazon Kinesis 数据流中的分片数量，无需额外编码即可直接连接到Amazon Simple Storage Service（Amazon S3）、Amazon Serv OpenSearch ice 或 Amazon Redshift 直接连接。如果您想将日志集中到这些AWS服务中，Kinesis Data Firehose 是一个有效的解决方案。
- Amazon Kinesis Dat a Streams — 如果您需要与 Kinesis Data Firehose 不支持的服务集成并实现额外的处理逻辑，Kinesis Data Streams 是合适的解决方案。您可以在您的账户和区域中创建 Amazon CloudWatch Logs 目的地，在中央账户中指定 Kinesis 数据流以及授予其在流中放置记录的权限的AWS Identity and Access Management (IAM) 角色。Kinesis Data Streams 为您的日志数据提供了一个灵活的开放式landing zone，然后这些数据可供不同的选项使用。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，您必须为数据流配置分片，使其大小适合生成的日志数据。Kinesis Data Streams 充当您的日志数据的临时中介或队列，您可以将数据存储在 Kinesis 流中一到 365 天。Kinesis Data Streams 还支持回放功能，这意味着您可以重放未消耗的数据。
- Amazon Serv OpenSearch ic 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。当您配置日志组以将数据传输到 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 OpenSearch 集群建立网络连接。除了自定义对 Amazon Serv OpenSearch ice 的摄取外，您还可以自定义 Lambda 函数以执行额外的预处理。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，则需要预置和管理消耗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。这意味着您不需要预置和管理计算资源即可处理传入数据。如果您选择使用 Lambda，请确保您的解决方案与 L ambda 配额兼容。

您可能需要以文件格式处理或共享存储在 Lo CloudWatch gs 中的日志数据。您可以创建导出任务，在特定日期或时间范围内将日志组导出到 Amazon S3。例如，您可以选择每天将日志导出到 Amazon S3 进行分析

和审计。Lambda 可用于自动化此解决方案。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运送和集中到一个集中账户和区域。

CloudWatch 代理配置还可以在部分中指定`credentials`字`agent`段。这指定一个在向不同账户发送指标和日志时使用的 IAM 角色时使用的 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区域中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 以您选择的语言编写自己的自定义处理应用程序，从您的账户读取日志和指标，并将数据发送到集中账户或其他目的地以进行进一步处理和监控。

管理 CloudWatch 代理配置文件

我们建议您创建一个标准 Amazon CloudWatch 代理配置，其中包括您想要在所有 Amazon Elastic Comppute Cloud (Amazon EC2) 实例和本地服务器中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独特的配置。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，您可以将您的 CloudWatch 配置作为文件集中管理 Amazon S3 中的文件或您选择的版本控制工具。

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，它不包含任何日志文件配置。如果您使用 S ystems Manager 快速设置作为 CloudWatch 代理，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定制的，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 ystems

如果您要在少量 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 参数来管理配置效果很好。当您将 CloudWatch 配置存储为参数存储参数时，您可以使用 CloudWatch 代理配置工具（`amazon-cloudwatch-agent-ctl`在 Linux 上）从参数存储中读取和应用配置，而无需将配置文件复制到您的实例。您可以使用 AmazonCloudWatch-ManageAgent Systems 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 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-ManageAgent Comman d 文档以包含这些参数。

CloudWatch 配置文件

如果您有许多 AWS 账户和区域并且管理多个 CloudWatch配置文件，则将配置作为文件进行管理可能会很有效。 CloudWatch 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。 您可以将安全规则应用于单个文件夹或文件，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 之外进行协作。 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 代理配置目录来集体应用配置，而无需单独应用每个配置文件。对于 Linux，可以在以下位置找到 CloudWatch 配置目录`/opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d`。对于 Windows，可以在以下位置找到配置目录`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。配置文件应存储在所需账户和区域可以访问的中心位置（例如 S3 存储桶）。 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/标准/windows/ec2	为Amazon EC2 存储特定于 Windows 的标准 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/本地	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进一步对不同 Windows 版本、服务器类型和环境的标准操作系统配置进行分类。
/config/standard/linux/ec2	存储您的Amazon EC2 特定于 Linux 的标准 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/本地	为本地服务器存储特定于 Linux 的标准 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 Amazon ECS 容器实例，则存储特定于Amazon Elastic Comple Service (Amazon ECS) 的 CloudWatch 配置文件。这些配置可以附加到用于 Amazon ECS 特定系统级日志记录和监控的标准 Amazon EC2 配置中。
/配置/ <application_name>	存储特定于应用程序的 CloudWatch 配置文件。您可以使用其他文件夹和环境和版本前缀进一步对应用程序进行分类。

示例：将 CloudWatch 配置文件存储在

本节提供了一个使用 Amazon S3 存储 CloudWatch配置文件的示例，以及一个用于检索和应用 CloudWatch 配置文件的自定义 Systems Manager 运行手册。这种方法可以解决使用 Systems Manager 参数存储参数进行大规模 CloudWatch配置时遇到的一些挑战：

- 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。参数存储是一项区域服务，必须在使用 CloudWatch 代理的每个区域中更新相同的参数。
- 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。您必须单独从 Parameter Store 检索每个 CloudWatch 配置，并且每当添加新配置时都要更新检索方法。相比之下， CloudWatch 它提供了一个用于存储配置文件的配置目录，并在该目录中应用每个配置，而无需单独指定它们。
- 如果您使用多个帐户，则必须确保每个新帐户在其参数存储中均具有所需的 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，所有账户和区域均可访问该存储桶。然后，您可以使用系统管理器自动化运行手册和系统管理器状态管理器将这些 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 AWS CloudFormation 模板创建一个 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，用于向组织内的所有账户授予读取权限。

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

或者，您可以使用版本控制系统（例如 AWS CodeCommit）来存储您的配置文件。 GitHub如果您想自动检索存储在版本控制系统中的配置文件，则必须管理或集中管理凭据存储，并更新用于跨账户和区域检索凭证的 Systems Manager Automation 运行手册。

配置 CloudWatch EC2 实例和本地服务器的代理

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，每个操作系统都有独特的安装和配置要求来捕获和提取

如果您选择使用 EC2 实例，则可以对实例和操作系统配置进行高级别的控制。但是，这种更高级别的控制和责任要求您监控和调整配置以实现更高效的使用。您可以通过建立日志记录和监控标准，并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。

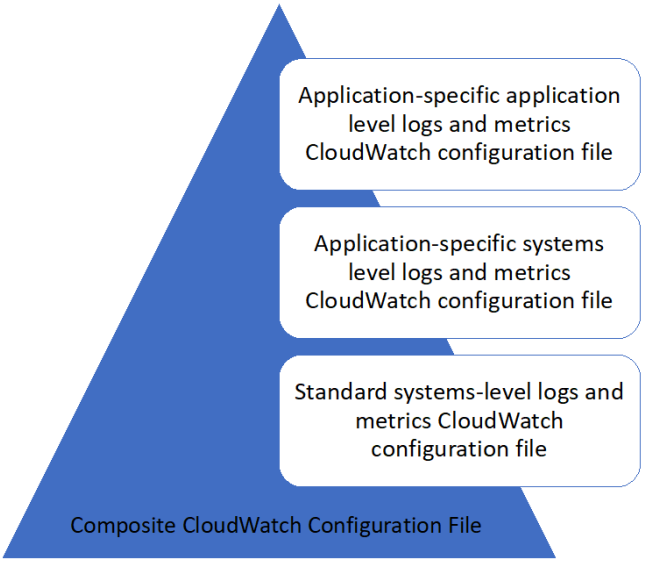
将 IT 投资迁移或扩展到AWS云可以利用 CloudWatch 以实现统一的日志记录和监控解决方案。 CloudWatch 定价意味着您按增量方式为要捕获的指标和日志付费。您还可以使用类似的方法捕获本地服务器的日志和指标 CloudWatch 代理安装过程与 Amazon EC2 相同。

在开始安装和部署 CloudWatch 之前，请确保评估系统和应用程序的日志记录和指标配置。确保为要使用的操作系统定义了需要捕获的标准日志和指标。系统日志和指标是日志记录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 Linux 和 Windows 而言不同。除了特定于 Linux 版本或发行版的指标和日志文件之外，Linux 发行版还有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 CloudWatch 代理人

CloudWatch 使用CloudWatch 代理和代理配置文件特定于每个操作系统的。我们建议您在开始安装之前定义组织的标准指标和日志捕获配置 CloudWatch 在你的账户中大规模代理。

您可以组合使用多个 CloudWatch 代理配置以形成复合 CloudWatch 代理配置。推荐的一种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。下图说明了如何组合满足不同要求的多种 CloudWatch 配置文件类型以形成复合 CloudWatch 配置：



还可以针对特定环境或要求进一步分类和配置这些日志和指标。例如，您可以为不受监管的开发环境定义一个较小的日志和指标子集，而且可以为受监管的生产环境定义更小、精度更高的日志和指标集，精度更高。

为 EC2 实例配置日志捕获

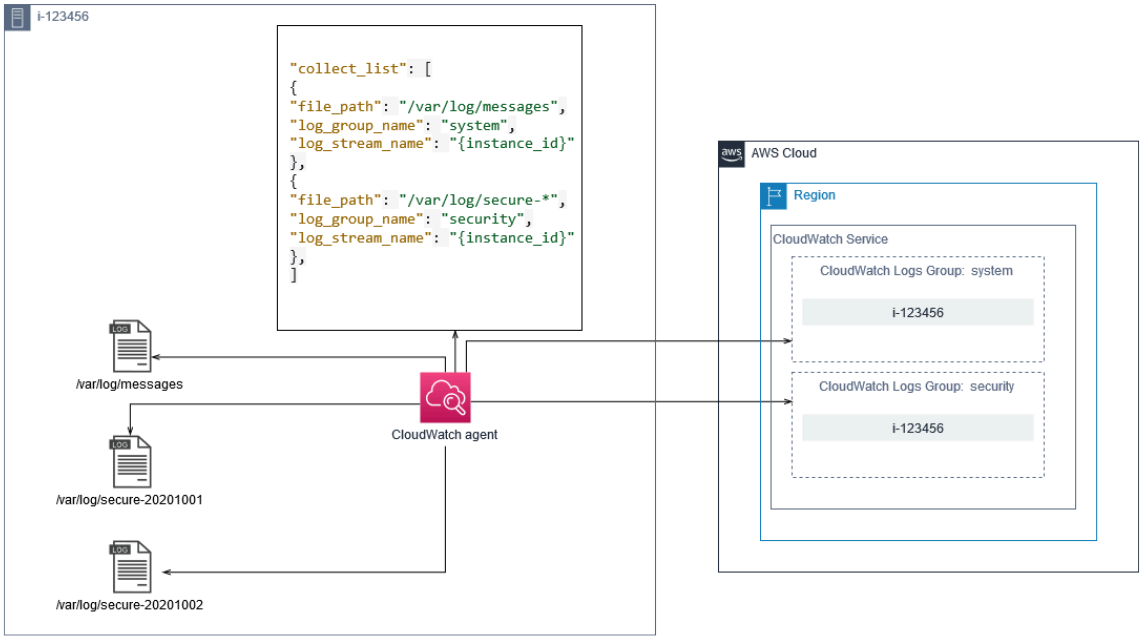
默认情况下，Amazon EC2 不监控或捕获日志文件。相反，会捕获日志文件并将其提取到 CloudWatch 通过的日志 CloudWatch EC2 实例上安装的代理软件，AWSAPI，或者AWS Command Line Interface(AWS CLI)。我们建议使用 CloudWatch 代理将日志文件提取到 CloudWatch Amazon EC2 和本地服务器的日志。

您可以搜索和筛选日志，以及提取指标并根据 CloudWatch 中日志文件的模式修补运行自动化。CloudWatch 支持纯文本、空格分隔和 JSON 格式的过滤器和模式语法选项，JSON 格式的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。

这些区域有： CloudWatch 代理使用一个配置文件，该文件定义了要发送到 CloudWatch 的日志和指标。CloudWatch 然后将每个日志文件捕获为日志流然后将这些日志流分组为日志组. 这有助于您在 EC2 实例中跨日志执行操作，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。您可以使用`instance_id`、`hostname`、`local_hostname`，或者`ip_address`对于日志流和日志组名称中的动态替换，这意味着您可以使用相同的 CloudWatch 跨多个 EC2 实例的代理配置文件。

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



日志组定义了它们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。基于日志文件名称的默认分组行为可帮助您搜索、创建指标并针对账户和区域中的 EC2 实例的日志文件特定的数据进行警报。你应该评估是否需要进一步改进日志组。例如，您的账户可能由多个业务单位共享，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。这种方法允许您将分析和故障排除集中在相关的 EC2 实例上。

如果多个环境使用一个帐户，则可以为每个环境中运行的工作负载分开日志记录。下表显示了日志组命名约定，其中包括业务部门、项目或应用程序以及环境。

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

您还可以将 EC2 实例的所有日志文件分组到同一个日志组中。这使得跨一组日志文件搜索和分析单个 EC2 实例变得更容易。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 实例都有特定的用途，则 下表显示了如何格式化日志组和日志流命名以支持此方法。

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，启用 EC2 实例可进行基本监控，标准指标集（例如，CPU、网络或与存储相关的指标）会自动发送到 CloudWatch 每 5 分钟。 CloudWatch 例如，指标可能因实例系列而异，可突增性能实例有 CPU 积分的指标。Amazon EC2 标准指标包含在您的实例价格中。如果您启用详细监控对于 EC2 实例，您可以在一分钟时段内接收数据。周期频率会影响您的 CloudWatch 成本，因此请确保评估所有 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。

本地服务器不包含任何默认指标 CloudWatch 并且必须使用 CloudWatch 代理，AWS CLI，或者AWS用于捕获指标的 SDK。这意味着您必须在 CloudWatch 配置文件。您可以创建独特的 CloudWatch 配置文件，其中包括本地服务器的标准 EC2 实例指标，并在标准之外应用该指标 CloudWatch 配置。

指标在 CloudWatch 是通过指标名称和零个或多个维度进行唯一定义的，并且在指标命名空间中进行唯一分组。由提供的指标AWS服务有一个以开头的命名空间`AWS`（例如，`AWS/EC2`）和非AWS指标是自定义指标。使用配置和捕获的指标 CloudWatch 代理都被视为自定义指标。因为创建的指标数量会影响您的 CloudWatch 成本，您应评估是所有 EC2 实例还是仅需要部分 EC2 实例每个指标。例如，您可以为生产工作负载定义一组完整的指标，但将这些指标中的一小部分用于非生产工作负载。

`CWAgent`是由 CloudWatch 代理。与日志组类似，指标命名空间组织了一组指标，以便可以在一个位置找到它们。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账户，此方法很有用。你也可以将你的命名空间命名约定与你的 CloudWatch 日志组命名约定。

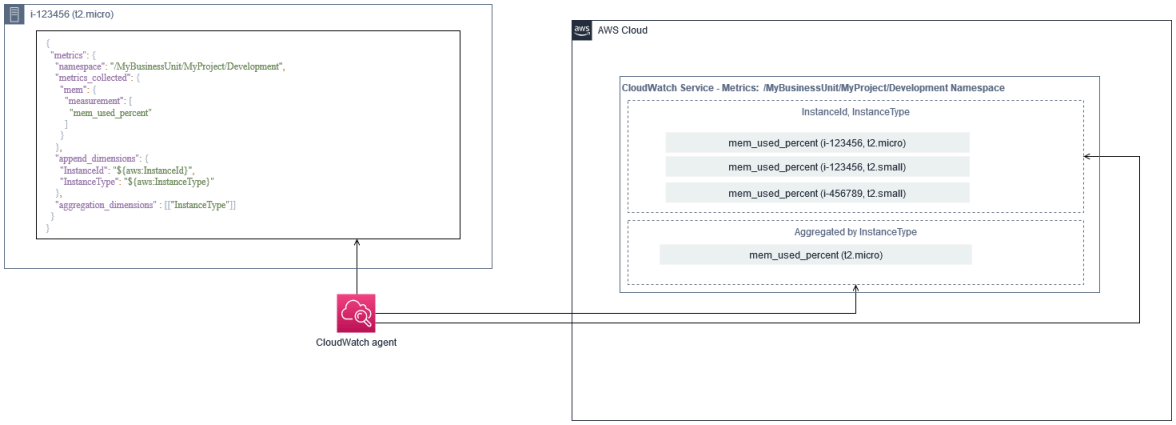
指标还通过它们的维度来标识，这有助于您根据一组条件分析它们，也是记录观测值所依据的属性。Amazon EC2 包括单独的指标适用于的 EC2 实例`InstanceId`和`AutoScalingGroupName`维度。您还可以使用`ImageId`和`InstanceType`如果您启用详细监控，则维度。例如，Amazon EC2 为 CPU 利用率提供了一个单独的 EC2 实例指标，其中`InstanceId`维度，此外还有单独的 CPU 利用率指标`InstanceType`维度。这有助于您分析每个唯一 EC2 实例的 CPU 利用率，以及特定 EC2 实例的所有 EC2 实例实例类型.

添加更多维度会增加分析能力，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产生一个新的量度。例如，如果您针对`InstanceId`维度，那么这是每个 EC2 实例的新指标。如果您的组织运行数千个 EC2 实例，这将导致成千上万个指标并导致成本更高。要控制和预测成本，请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，但这些维度中的一小部分用于非生产工作负载。

您可以使用`append_dimensions`属性可将维度添加到您的中定义的一个或所有指标 CloudWatch 配置。你也可以动态附加`ImageId`、`InstanceId`、`InstanceType`, 和`AutoScalingGroupName`到你的所有指标 CloudWatch 配置。或者，您可以使用`append_dimensions`该指标上的属性。 CloudWatch 还可以聚合您使用`aggregation_dimensions`财产。

例如，您可以将使用的内存聚合在`InstanceType`维度以查看每种实例类型的所有 EC2 实例使用的平均内存。如果您使用`t2.micro`在区域中运行的实例，您可以确定工作负载是否使用`t2.micro`类过度利用或未充分利用所提供的内存。利用率不足可能是工作负载使用具有不需要内存容量的 EC2 类的标志。相比之下，过度利用可能是工作负载使用内存不足的 Amazon EC2 类的标志。

下图显示了一个示例。 CloudWatch 使用自定义命名空间、添加的维度和聚合的指标配置InstanceType.



系统级 CloudWatch 配置

系统级指标和日志是监控和记录解决方案的核心组成部分， CloudWatch 代理有适用于 Windows 和 Linux 的特定配置选项。

我们建议使用CloudWatch 配置文件向导或配置文件架构来定义 CloudWatch 计划支持的每个操作系统的代理配置文件。其他特定于工作负载的操作系统级日志和指标可以在单独的中定义 CloudWatch 配置文件并附加到标准配置中。这些唯一的配置文件应单独存储在 S3 存储桶中，您的 EC2 实例可以在其中检索它们。为此目的设置 S3 存储桶的示例在管理 CloudWatch配置 (p. 6)本指南的部分。您可以使用州管理器和分销商自动检索和应用这些配置。

配置系统级日志

系统级日志对于诊断和故障排除本地或在AWS云。您的日志捕获方法应包括操作系统生成的所有系统和安全日志。操作系统生成的日志文件可能因操作系统版本而异。

这些区域有： CloudWatch 代理支持通过提供事件日志名称来监视 Windows 事件日志。你可以选择要监视的 Windows 事件日志（例如System、Application，或者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。下表定义了应监控的常见默认日志文件，但是您应该检查/etc/rsyslog.conf要么/etc/syslog.conf文件来确定系统日志文件的具体设置。

Fedora 分配 （Amazon Linux、CentOS、Red Hat Enterprise Linux）	/var/log/boot.log*— 启动日志
	/var/log/dmesg— 内核日志
	/var/log/secure— 安全和身份验证日志
	/var/log/messages— 常规系统日志
	/var/log/cron*— Cron 日志
	/var/log/cloud-init-output.log— 来自的输出Userdata启动脚本
Debian (Ubuntu)	/var/log/syslog— 启动日志
	/var/log/cloud-init-output.log— 来自的输出Userdata启动脚本

		/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要监控的日志。您应评估并决定这些代理或应用程序生成的日志文件，并通过标识它们的文件位置将它们包含在配置中。例如，您应该包括 Systems Manager 和 CloudWatch 代理登录到您的配置中。下表提供了 Windows 和 Linux 的这些代理日志的位置。

Windows	CloudWatch 代理	$Env:ProgramData\Amazon\AmazonCloudWatchAgent\Logs\amazon-cloudwatch-agent.log
	Systems Manager 代理	%PROGRAMDATA%\Amazon\SSM\Logs\amazon-ssm-agent.log %PROGRAMDATA%\Amazon\SSM\Logs\errors.log %PROGRAMDATA%\Amazon\SSM\Logs\audits\amazon-ssm-agent-audit-YYYY-MM-DD
Linux	CloudWatch 代理	/opt/aws/amazon-cloudwatch-agent/logs/amazon-cloudwatch-agent.log
	Systems Manager 代理	/var/log/amazon/ssm/amazon-ssm-agent.log /var/log/amazon/ssm/errors.log /var/log/amazon/ssm/audits/amazon-ssm-agent-audit-YYYY-MM-DD

如果日志文件在 CloudWatch 代理配置但未找到。当您想为 Linux 维护单个日志配置时，这非常有用，而不是为每个发行版维护单独的配置。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。要包括这些指标，您必须安装并配置 CloudWatch EC2 实例上的代理。这些区域有： CloudWatch 代理配置向导创建 CloudWatch 使用的配置预定义指标您可以根据需要添加或删除指标。确保您查看预定义的指标集以确定所需的适当级别。

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。应将这些指标定义存储、版本化和维护在单独的 CloudWatch 代理配置文件，并在中心位置（例如 Amazon S3）共享，以便重复使用和自动化。

标准 Amazon EC2 指标不会在本地服务器中自动捕获。这些指标必须在 CloudWatch 本地实例使用的代理配置文件。您可以使用 CPU 利用率等指标为本地实例创建单独的指标配置文件，并将这些指标附加到标准指标配置文件中。

应用程序级 CloudWatch 配置

应用程序日志和指标是由正在运行的应用程序生成的，并且特 确保定义了充分监控组织经常使用的应用程序所需的日志和指标。例如，您的组织可能已针对基于 Web 的应用程序在 Microsoft 互联网信息服务器 (IIS) 上标准化。你可以创建标准日志和指标 CloudWatch IIS 的配置，也可以在整个组织中使用。特定于应用程序的配置文件可以存储在集中位置（例如 S3 存储桶）中，工作负载所有者可以访问或通过自动检索进行访问，然后复制到 CloudWatch 配置目录。这些区域有： CloudWatch 代理会自动将在每个 EC2 实例或服务器的配置文件目录中找到的 CloudWatch 配置文件合并为一个复合 CloudWatch 配置。最终结果是 CloudWatch 包括组织的标准系统级配置以及所有相关的应用程序级别的配置 CloudWatch 配置。

工作负载所有者应为所有关键应用程序和组件识别并配置日志文件和指标。

配置应用程序级日志

应用程序级日志记录取决于应用程序是否为商业应用程序 off-the-shelf (COTS) 或自定义开发的应用程序。COTS 应用程序及其组件可能会为日志配置和输出提供几个选项，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 或第三方应用程序都不允许您从根本上更改日志记录（例如，更新应用程序的代码以包含其他不可配置的日志语句或格式）。至少，您应该为 COTS 或第三方应用程序配置日志记录选项，以记录警告和错误级别的信息，最好采用 JSON 格式。

您可以将定制开发的应用程序与 CloudWatch 通过将应用程序的日志文件包含在 CloudWatch 配置。自定义应用程序提供了更好的日志质量和控制能力，因为除了包括任何其他必需的详细信息之外，您还可以自定义日志输出格式、分类和分离组件输出以单独的 确保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，以便分析和处理变得更加轻松。

你也可以写信给 CloudWatch 使用的日志流 CloudWatch 日志`PutLogEvents`API 调用或使用AWSSDK。您可以使用 API 或 SDK 满足自定义日志记录要求，例如将日志记录协调到一组分布式组件和服务器之间的单个日志流。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 代理来读取日志文件并将其流式传输到 CloudWatch。

您还应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在 CloudWatch 日志组。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计算失败的登录尝试。

您还可以使用CloudWatch 嵌入式指标格式在应用程序日志文件中。

配置应用程序级指标

自定义指标是不是直接提供的指标AWS向的服务 CloudWatch 它们发布在自定义命名空间中 CloudWatch 指标。所有应用程序指标都被视为自 CloudWatch 指标。应用程序指标可能与 EC2 实例、应用程序组件、API 调用甚至业务函数对齐。您还必须考虑为指标选择的维度的重要性和基本性。基数度高的维度会生成大量自定义指标，并可能会增加 CloudWatch 成本.

CloudWatch 可以通过多种方式来捕获应用程序级指标，包括以下方法：

- 通过定义要从procstat 插件.
- 应用程序将指标发布到 Windows 性能监视器，此指标在 CloudWatch 配置。
- 指标筛选器和模式应用于 CloudWatch 中的应用程序的日志。
- 应用程序写入 CloudWatch 使用登录 CloudWatch 嵌入式指标格式。
- 应用程序将指标发送到 CloudWatch 通过 API 或AWSSDK。
- 应用程序将指标发送到收集要么StatsD已配置的守护进程 CloudWatch 代理。

您可以使用 procstat 来使用 CloudWatch 代理监控和衡量关键应用程序进程。如果应用程序不再运行关键进程，这可以帮助您发出警报并采取措施（例如，通知或重启进程）。您还可以衡量应用程序进程的性能特征，并在特定进程异常行为时发出警报。

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`my_process`衡量`cpu_time`并包括自定义`application_version`维度。您也可以使用多个 CloudWatch 如果不同指标有不同的维度，则应用程序的代理配置文件。

如果你的应用程序在 Windows 上运行，你应评估它是否已经向 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 还可以与 Windows 性能监视器集成，您可以捕获其中已有的任何指标。

确保您查看应用程序提供的日志记录格式和日志信息，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，以确定错误消息和异常停机的表示方式。您还应查看之前报告的问题，以确定是否可以捕获指标以防止问题再次发生。您还应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，定义可以使用 CloudWatch 嵌入式指标格式，AWS或AWSAPI。建议的方法是使用嵌入式指标格式。您可以使用AWS提供了开源嵌入式指标格式库，以帮助您以所需格式编写报表。您还需要更新您的特定于应用程序 CloudWatch 配置以包括嵌入式指标格式代理。这会导致 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 CloudWatch 发送嵌入式指标格式指标。

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

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

自动执行 CloudWatch 代理的安装过程可帮助您快速、一致地部署它并捕获所需的日志和指标。自动化 CloudWatch 代理安装有多种方法，包括多账户和多区域支持。讨论了以下自动安装方法：

- 安装 CloudWatch 使用 Systems Manager 分销商和系统管理器州管理器 (p. 16)— 如果 EC2 实例和本地服务器正在运行 Systems Manager 代理，建议使用此方法。这可以确保 CloudWatch 代理程序会保持更新，您可以报告并修复没有 CloudWatch 代理。此方法还扩展到支持多个账户和区域。
- 部署 CloudWatch 代理作为 EC2 实例预配期间用户数据脚本的一部分 (p. 21)— Amazon EC2 允许您定义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AWS CloudFormation和AWSService Catalog 产品。如果针对不符合标准的特定工作负载有自定义的代理安装和配置方法，则此方法可能在需要的基础上适用。
- 在 Amazon 系统映像 (AMI) 中包含 CloudWatch 代理 (p. 21)— 您可以在适用于 Amazon EC2 的自定义 AMI 中安装 CloudWatch 代理。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。

安装 CloudWatch 使用 Systems Manager 分销商和状态管理器

您可以使用 Systems Manager 状态管理器状态管理器和系统管理器分发器自动安装 CloudWatch 服务器和 EC2 实例上的代理。分销商包括`AmazonCloudWatchAgent` AWS安装最新 CloudWatch 代理版本的托管软件包。

此安装方法包含以下先决条件：

- Systems Manager 代理必须已安装并运行在您的服务器或 EC2 实例上。系统管理器代理已预装在亚马逊 Linux、亚马逊 Linux 2 和一些 AMI 上。还必须在其他映像或本地虚拟机和服务器上安装和配置代理。
- IAM 角色或具有规定的 CloudWatch 和 Systems Manager 权限必须附加到 EC2 实例或在本地服务器的凭证文件中定义。例如，您可以创建包含AWS托管策略：`AmazonSSMManagedInstanceCore`对于 Systems Manager 和`CloudWatchAgentServerPolicy`CloudWatch。您可以使用ssm-cloud watch 实例角色 .yaml AWS CloudFormation用于部署包含这两个策略的 IAM 角色和实例配置文件的模板。还可以修改此模板以包含 EC2 实例的其他标准 IAM 权限。对于本地服务器或虚拟机，应配置 CloudWatch 使用的代理Systems Manager 服务角色这是针对本地服务器配置的。有关此问题的更多信息，请参阅如何配置使用 Systems Manager 代理和统一的本地服务器 CloudWatch 代理只使用临时证书？中的AWS知识中心。

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：

- 自动安装多个操作系统— 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— 州经理会自动定期检查每个 EC2 实例是否具有最新的 CloudWatch 版本。
- 合规性报告— Systems Manager 合规性仪表板显示哪些 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— 启动到您账户中的新 EC2 实例会自动接收 CloudWatch 代理。

但是，在选择此方法之前，您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 代理人，那么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— 分销商使用默认配置文件执行安装。如果你使用自定义配置文件或多个 CloudWatch 配置文件，您必须在安装后更新配置。
- 多区域或多账户设置— 必须在每个账户和地区建立州长关联。必须更新多账户环境中的新账户以包括州经理关联。你需要集中或同步 CloudWatch 配置，以便多个账户和区域可以检索并应用所需的标准。

设置州经理和分销商 CloudWatch 代理部署和配置

您可以使用Systems Manager 快速设置快速配置 Systems Manager 功能，包括自动安装和更新 CloudWatch EC2 实例上的代理。快速安装程序部署了AWS CloudFormation根据您的选择部署和配置 Systems Manager 资源的堆栈。

以下列表提供了快速安装程序执行的两项重要操作以实现自动化 CloudWatch 代理安装和更新：

1. 创建 Systems Manager 定制文档— 快速安装程序创建以下 Systems Manager 文档以供状态管理器一起使用 文档名称可能会有所不同，但内容保持不变：
   - `CreateAndAttachIAMToInstance`— 创建`AmazonSSMRoleForInstancesQuickSetup`角色和实例配置文件（如果不存在）并附加`AmazonSSMManagedInstanceCore`策略角色。这不包括必需的`CloudWatchAgentServerPolicy`IAM 策略。您必须更新此策略并更新此 Systems Manager 文档以包括此策略，如下一节所述。
   - `InstallAndManageCloudWatchDocument`— 安装 CloudWatch 使用分销商代理，并使用默认设置一次性配置每个 EC2 实例 CloudWatch 使用`AWS-ConfigureAWSPackage`Systems Manager 文档。
   - `UpdateCloudWatchDocument`— 更新 CloudWatch CloudWatch`AWS-ConfigureAWSPackage`Systems Manager 文档。更新或卸载代理程序不会删除现有的 CloudWatch EC2 实例中的配置文件。
2. 创建状态管理器关联— 创建状态管理器关联并配置为使用自定义创建的 Systems Manager 文档。州管理员关联名称可能会有所不同，但配置保持不变：
   - `ManageCloudWatchAgent`— 运行`InstallAndManageCloudWatchDocument`Systems Manager 为每个 EC2 实例记录一次。
   - `UpdateCloudWatchAgent`— 运行`UpdateCloudWatchDocument`Systems Manager 每 30 天为每个 EC2 实例记录一次。
   - 运行`CreateAndAttachIAMToInstance`Systems Manager 为每个 EC2 实例记录一次。

您必须增加和自定义已完成的快速安装配置，以包括 CloudWatch 权限和支持自定义 CloudWatch 配置。具体而言，`CreateAndAttachIAMToInstance`和`InstallAndManageCloudWatchDocument`将需要更新文档。您可以手动更新快速设置创建的 Systems Manager 文档。此外，也可以使用自己的 CloudFormation 模板以配置相同的资源并提供必要的更新，以及配置和部署其他 Systems Manager 资源，而不是使用快速设置。

> **Important**
>
> 创建快速设置AWS CloudFormation堆栈以根据您的选择部署和配置 Systems Manager 资源。如果更新快速安装选项，则可能需要手动重新更新 Systems Manager 文档。

以下各节介绍了如何手动更新快速安装程序创建的 Systems Manager 资源，以及如何使用自己的资源AWS CloudFormation模板来执行更新的快速设置。建议使用自己的AWS CloudFormation模板，以避免手动更新快速安装程序创建的资源和AWS CloudFormation.

使用 Systems Manager 快速设置并手动更新创建的 Systems Manager 资源

必须更新快速设置方法创建的 Systems Manager 资源，以包括所需的 CloudWatch 代理权限和支持多个 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 CloudWatch 可

从多个账户访问的配置。创建 S3 存储桶以存储 CloudWatch 配置文件在管理 CloudWatch配置 (p. 6)本指南部分。

更新CreateAndAttachIAMToInstanceSystems Manager 文档

此 Systems Manager 文档由快速安装程序创建，检查 EC2 实例是否附加了现有的 IAM 实例配置文件。如果确实如此，它会附加AmazonSSMManagedInstanceCore对现有角色的策略。这可以保护您的现有 EC2 实例不会丢失AWS可能通过现有实例配置文件分配的权限。你需要在本文档中添加一个步骤才能附加CloudWatchAgentServerPolicy针对已附加实例配置文件的 EC2 实例的 IAM 策略。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才能同时包含CloudWatchAgentServerPolicyIAM 策略。

查看完成创建我并将其附加到实例。YAML示例文档并将其与快速安装程序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，快速安装程序创建的文档可能与提供的示例文档不同，因此请确保进行必要的调整。该示例文档包括快速设置选项，用于每天扫描实例以查找缺失的补丁程序，因此包括了 Systems Manager 补丁管理器的策略。

更新InstallAndManageCloudWatchDocumentSystems Manager 文档

快速安装程序创建的此 Systems Manager 文档将安装 CloudWatch 代理并使用默认值对其进行配置 CloudWatch 代理配置。默认 CloudWatch 配置与基本的、预定义的指标集保持一致。您必须替换默认配置步骤并添加步骤才能下载 CloudWatch 来自您的配置文件 CloudWatch 配置 S3 存储桶。

查看完成安装和管理 CloudWatch 文档 .YAML更新了文档并将其与快速安装程序创建的文档进行比较。快速设置创建的文档可能会有所不同，因此请确保您已进行了必要的调整。编辑现有文档以包含必要的步骤和更改。

使用AWS CloudFormation而不是快速设置

您可以使用，而不是使用快速设置AWS CloudFormation以配置 Systems Manager。使用此方法，您可以根据自己的具体要求自定义 Systems Manager 配置。此方法还可以避免手动更新快速安装程序为支持自定义而创建的配置的 Systems Manager 资源 CloudWatch 配置。

快速设置功能还使用AWS CloudFormation然后创建AWS CloudFormation堆栈设置为根据您的选择部署和配置 Systems Manager 资源。在你可以使用之前AWS CloudFormation堆栈集，您必须创建AWS CloudFormation StackSets 以支持跨多个账户或区域的部署。快速安装程序创建支持多区域或多账户部署所需的角色AWS CloudFormationStackSets。您必须完成的先决条件AWS CloudFormation StackSets 如果要在多个区域或单个账户和区域中配置和部署多个账户中的 Systems Manager 资源。有关此问题的更多信息，请参阅堆栈集操作的先决条件中的AWS CloudFormation文档中）。

查看AWS-Quicksetup-ssmhostmgmt.YAML AWS CloudFormation自定义快速设置模板。

您应该查看AWS CloudFormation模板并根据您的要求进行调整。你应该使用版本控制AWS CloudFormation模板，您可以使用该模板并以增量方式测试更改以确认所需结果。此外，您应该执行云安全审查，以确定是否根据组织的要求进行任何政策调整。

您应该部署AWS CloudFormation堆叠在单个测试账户和区域中，然后执行任何必要的测试用例来自定义和确认所需的结果。然后，您可以在单个账户中将部署重新部署到多个区域，然后再升至多个账户和多个区域。

在单个账户和区域中使用AWS CloudFormation堆

如果您只使用单个账户和区域，则可以将完整示例部署为AWS CloudFormation堆栈而非AWS CloudFormation堆栈集。但是，如果可能的话，我们建议您使用多账户、多区域堆栈集方法，即使只使用单个账户和区域。使用AWS CloudFormation StackSets 使 future 更容易扩展到其他账户和地区。

使用以下步骤部署AWS-Quicksetup-ssmhostmgmt.YAML AWS CloudFormation作为一个模板AWS CloudFormation堆叠在单个账户和区域中：

1. 下载模板并将其签入您首选的版本控制系统（例如，AWS CodeCommit)。
2. 自定义默认值AWS CloudFormation参数值基于组织的要求。
3. 自定义州经理关联时间表。
4. 使用自定义 Systems Manager 文档`InstallAndManageCloudWatchDocument`逻辑 ID。确认 S3 存储桶前缀与包含 CloudWatch 配置。
5. 检索并记录包含您的 S3 存储桶的 Amazon 资源名称 (ARN) CloudWatch 配置。有关此问题的更多信息，请参阅管理 CloudWatch配置 (p. 6)本指南中的部分。一个示例cloud watch config-s3-bucket.yaml AWS CloudFormation模板可用，其中包括存储桶策略，用于提供读取访问权限AWS Organizations账户。
6. 部署自定义的快速设置AWS CloudFormation模板与您的 S3 存储桶相同的账户：
   - 对于`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. 部署带有和不带 IAM 角色的测试 EC2 实例，以确认 EC2 实例与 CloudWatch 一起使用。

- 应用`AttachIAMToInstance`State Manager 关联。这是 Systems Manager 运行本手册，配置为按计划运行。使用 runbook 的状态管理器关联不会自动应用于新的 EC2 实例，可以配置为按计划运行。有关更多信息，请参阅 。使用 State Manager 使用触发器运行自动化在 Systems Manager 文档中。
- 确认 EC2 实例附加了所需的 IAM 角色。
- 通过确认 EC2 实例在 Systems Manager 中可见，确认 Systems Manager 是否正常工作。
- 确认 CloudWatch 通过查看代理工作正常 CloudWatch 日志和指标基于 CloudWatch S3 存储桶中的配置。

使用多个区域和多个账户中的自定义快速设置AWS CloudFormationStackSets

如果您使用多个账户和区域，则可以部署AWS-Quicksetup-ssmhostmgmt.YAML AWS CloudFormation模板作为堆栈集。你必须完成AWS CloudFormationStackSet 先决条件在使用堆栈集之前。这些要求取决于您是否使用部署堆栈集自行管理要么托管式服务许可.

我们建议您部署具有服务托管权限的堆栈集，以便新帐户自动接收自定义的快速安装程序。您必须从AWS Organizations管理帐户或委托管理员帐户。您应该从具有委派管理员权限的用于自动化的集中帐户部署堆栈集，而不是AWS Organizations管理账户。我们还建议您通过定位在一个区域中拥有单个或少量帐户的测试组织单位 (OU) 来测试堆栈集部署。

1. 完成从在单个账户和区域中使用AWS CloudFormation堆 (p. 18)本指南中的部分。
2. 登录到AWS Management Console，打开AWS CloudFormation控制台然后选择创建 StackSet：
   - 选择模板准备就绪和上传模板文件. 上传AWS CloudFormation您根据自己的要求定制的模板。
   - 指定堆栈集详细信息：
     - 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。
     - 对于`CloudWatchConfigBucketARN`参数，输入您的 ARN CloudWatch 配置的 S3 存储桶。
     - 指定堆栈集选项，选择是否将服务管理的权限与AWS Organizations或自行管理的权限。
       - 如果选择自我管理的权限，请输入AWS CloudForms 堆栈集管理角色和AWS CloudForms 堆栈集执行角色IAM 角色详细信息。管理员角色必须存在于账户中，且每个目标账户中必须存在执行角色
     - 适用于托管式服务使用的权限AWS Organizations，建议您首先部署到测试 OU 而不是整个组织。
       - 选择是否要启用自动部署。建议您选择Enabled (已启用). 对于账户移除行为，推荐的设置为删除堆栈.

- 适用于自行管理权限，输入AWS您要设置的账户的账户 ID。如果您使用自我管理的权限，则必须为每个新账户重复此过程。
- 输入要使用的地区 CloudWatch 和 Systems Manager。
- 通过查看中的状态来确认部署是否成功操作 和堆栈实例堆栈集的选项卡。
- 测试系 Systems Manager 和 CloudWatch 在已部署的帐户中正常工作，请按照在单个账户和区域中使用AWS CloudFormation堆 (p. 18)本指南中的部分。

配置本地服务器的注意事

这些区域有： CloudWatch 本地服务器和虚拟机的代理是通过使用与 EC2 实例类似的方法来安装和配置的。但是，下表提供了在安装和配置 CloudWatch 本地服务器和虚拟机上的代理。

指向 CloudWatch 代理到用于 Systems Manager 的同一临时凭证。	当您在包含本地服务器的混合环境中设置 Systems Manager 时，您可以使用 IAM 角色激活系统管理器。您应该使用为 EC2 实例创建的角色，其中包括`CloudWatchAgentServerPolicy`和`AmazonSSMManagedIns`策。 这将导致 Systems Manager 代理检索临时证书并将其写入本地凭证文件。你可以指出你的 CloudWatch 代理配置到同一个文件。你可以使用这个过程将使用 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中的AWS知识中心。 您还可以通过定义单独的 Systems Manager Automation Runbook 和状态管理器关联以及使用标签定位本地实例来自动执行此过程。创建时激活 Systems Manager对于本地实例，您应包括一个标签，用于将实例标识为本地实例。
考虑使用拥有 VPN 的帐户和地区或AWS Direct Connect访问和AWSPrivateLink。	您可以使用AWS Direct Connect要么AWS Virtual Private Network(AWS VPN) 在本地网络和您的虚拟私有云 (VPC) 之间建立私有连接。AWSPrivateLink 建立私有连接 CloudWatch 使用接口 VPC 终端节点进行日志。如果您有限制阻止数据通过公共互联网发送到公共服务终端节点，则此方法非常有用。
所有指标必须包含在 CloudWatch 配置文件。	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，然后将配置附加到标准 CloudWatch 平台的指标配置。

临时 EC2 实例的注意事项

EC2 实例是临时的，或者短暂的，如果它们是 Amazon EC2 Auto Scaling 配置的，则 Amazon EMR，Amazon EC2 Spot 实例，或者AWS Batch. 临时 EC2 实例可能会导致大量 CloudWatch 没有关于其运行时源的其他信息的公共日志组下的流。

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 ID、Amazon EMR 集群名称或 Auto Scaling 组名称。对于新启动的 EC2 实例，此信息可能会有所不同，您可能必须在运行时检索和配置它。您可以通过编写 CloudWatch 启动时的代理配置文件，然后重新启动代理以包含更新的配置文件。这样可以使用动态运行时信息向 CloudWatch 交付日志和指标。

您还应确保指标和日志是由 CloudWatch 在临时 EC2 实例终止之前代理。这些区域有： CloudWatch 代理商包括`flush_interval`参数，可以配置为定义刷新日志和指标缓冲区的时间间隔。您可以根据工作负载降低此值并停止 CloudWatch 并强制缓冲区在 EC2 实例终止之前刷新。

使用自动化解决方案部署 CloudWatch 代理人

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

- 验证自动化是否涵盖了您支持的操作系统和操作系统版本。如果自动化脚本不支持组织的所有操作系统，则应为不受支持的操作系统定义替代解决方案。
- 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，或者定期卸载并重新安装代理。您可以使用计划程序或自动化解决方案功能定期检查和更新代理程序。
- 验证是否可以确认代理安装和配置合规性. 自动化解决方案应使您能够确定系统何时没有安装代理程序或代理程序何时无法正常工作。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。

部署 CloudWatch 使用用户数据脚本预配实例期间的代理

如果您不打算使用 Systems Manager 并希望有选择性地将 CloudWatch 用于 EC2 实例，则可以使用此方法。通常，这种方法是一次性使用的，或者在需要专门配置时使用。AWS提供直接链接(对于) CloudWatch 可以在启动或用户数据脚本中下载的代理。代理安装软件包可以在没有用户交互的情况下以静默方式运行，这意味着您可以在自动部署中使用它们。如果你使用这种方法，你应该评估以下注意事项：

- 用户无法安装代理或配置标准指标的风险增加. 用户可以在不包括必要的步骤的情况下预配实例来安装 CloudWatch 代理。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。
- 安装脚本必须是操作系统特定的，适用于不同的操作系统版. 如果你打算同时使用 Windows 和 Linux，你需要单独的脚本。Linux 脚本还应该根据发行版具有不同的安装步骤。
- 你必须定期更新 CloudWatch 在可用时使用新版本的代理. 如果将系统管理器与状态管理器结合使用，则可以自动执行此操作，但您也可以配置用户数据脚本以在实例启动时重新运行。这些区域有： CloudWatch 然后在每次重新启动时更新并重新安装代理。
- 您必须自动检索和应用标准 CloudWatch 配置. 如果将系统管理器与状态管理器结合使用，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。

包括 CloudWatch AMI 中的代理

使用这种方法的优点是，您不必等待 CloudWatch 要安装和配置的代理程序，您可以立即开始日志记录和监控。这有助于您更好地监控实例配置和启动步骤，以防实例无法启动。如果您不打算使用 Systems Manager 代理，此方法也是合适的。如果你使用这种方法，你应该评估以下注意事项：

- 必须存在更新过程，因为 AMI 可能不包括最新的 CloudWatch 代理版本. 这些区域有： CloudWatch AMI 中安装的代理仅在上次创建 AMI 时才是最新的。您应该包括一种额外的方法，用于定期更新代理以及在配置 EC2 实例时更新代理。如果使用 Systems Manager，则可以使用安装 CloudWatch 使用 Systems Manager 分销商和状态管理器 (p. 16)本指南中为此提供的解决方案。如果不使用 Systems Manager，则可以使用用户数据脚本在实例启动和重新启动时更新代理。
- 您的 CloudWatch 实例启动时必须检索代理配置文件. 如果不使用 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。
- 这些区域有： CloudWatch 必须在你的 CloudWatch 配置已更新.

- AWS证书不能保存在 AMI 中. 确保没有本地AWS凭证存储在 AMI 中。如果您使用 Amazon EC2，则可以将必要的 IAM 角色应用于实例并避免使用本地证书。如果您使用本地实例，则应在启动之前自动或手动更新实例凭证 CloudWatch 代理。

Amazon ECS 日志记录和监控

Amazon Elastic Container Service (Amazon ECS) 为运行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；这些启动类型是AWS Fargate Amazon EC2。两种启动类型都集成，CloudWatch但配置和支持各不相同。

以下部分可帮助您了解如何在 Amazon ECS 上使用CloudWatch日志记录和监控。

主题

- CloudWatch使用 EC2 启动类型进行配置 (p. 23)
- EC2 和 Fargate 启动类型的Amazon ECS 容器日志 (p. 24)
- 在 Amazon ECS 上FireLens使用自定义日志路由 (p. 24)
- Amazon ECS 的指标 (p. 25)

CloudWatch使用 EC2 启动类型进行配置

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。Amazon ECS 优化的 AMI 预先安装了 Amazon ECS 容器代理，可为 Amazon ECS 集群提供CloudWatch指标。

这些默认指标包含在 Amazon ECS 的费用中，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）。您可以使用预置具有 EC2 启动类型的 Amazon ECS 集群，这会创建一个AWS CloudFormation堆栈来部署具有启动配置的Amazon EC2 Auto Scaling群组。AWS Management Console但是，这种方法意味着您无法选择自定义 AMI 或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。您可以使用本指南安装 CloudWatch 使用 Systems Manager 分销商和状态管理器 (p. 16)部分中的 EC2 实例安装方法。但是，Amazon ECS AMI 不包含所需的系统管理器代理。在创建 Amazon ECS 集群时，您应使用带有用户数据脚本的自定义启动配置，该脚本用于安装 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 运行和更新您的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配置存储在用于CloudWatch配置的 S3 存储桶中，并使用状态管理器自动应用它们。

您应确保应用于 Amazon ECS 容器实例的 IAM 角色或实例配置文件包含必需的`CloudWatchAgentServerPolicy`和`AmazonSSMManagedInstanceCore`策略。你可以使用 ecs_cluster_with_cloudwatch_linux.yamlAWS CloudFormation 模板预置基于 Linux 的 Amazon ECS 集群。此模板使用自定义启动配置创建 Amazon ECS 集群，用于安装 Systems Manager 并部署自定义CloudWatch配置以监控特定于 Amazon ECS 的日志文件。

您应该捕获您的 Amazon ECS 容器实例的以下日志以及标准 EC2 实例日志：

- 亚马逊 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。

亚马逊 ECS 容器实例应使用最新的 Amazon ECS 优化的 AMI 和容器代理。 AWS使用 Amazon 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 Amazon ECS 优化的 AMI 的参数存储参数格式，从参数存储中检索最新优化的 AMI。您可以引用公共参数存储参数，该参数引用AWS CloudFormation模板中最新的 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AWS CloudFormation模板可以在各个区域和账户之间重复使用，而无需更新 AMI。您可以通过参考特定版本来控制向您的组织部署更新的 Amazon ECS AMI，这有助于您在测试之前防止使用经过优化的新 AMI。

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。日志记录是在容器级别使用日志驱动程序配置的。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日志驱动程序来捕获容器输出并将其传输到CloudWatch日志。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。

日志组的默认命名以及上的“自动配置CloudWatch日志”选项使用的选项AWS Management Console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。我们建议您使用群组名称，根据组织的要求对日志进行分组。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本进行更新，这意味着任务定义可能使用了image_name与image_tag任务定义当前使用的版本不同。有关更多信息和命名建议，请参阅本指南的规划 CloudWatch 部署 (p. 4)部分。

如果您使用持续集成和持续交付 (CI/CD) 管道或自动化流程，则可以在每个新的 Docker 映像版本中为应用程序创建新的任务定义修订版。例如，作为 CI/CD 流程的一部分，您可以将 Docker 映像名称、图像标签、修订版或其他重要信息包含在任务定义修订版和日志配置中。GitHub

在 Amazon ECS 上FireLens使用自定义日志路由

FireLensfor Amazon ECS 可帮助您将日志路由到 Fluentd 或 Fluen t Bit，这样您就可以将容器日志直接发送到AWS服务和AWS合作伙伴网络 (APN) 目的地，并支持将日志传输到CloudWatch日志。

AWS为 Fluent Bit 提供了 Docker 镜像，其中预装了Amazon Kinesis Data Streams、亚马逊 Kinesis Data Firehose 和CloudWatch Logs 的插件。您可以使用FireLens日志驱动程序代替日志驱动程序来对发送awslogs到 Logs 的日志进行CloudWatch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环境变量向您的容器公开。这意味着您可以使用 Fluent Logger 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可用的值登录 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。它还启动了 FFireLens luent Bit 容器作为用于记录的附带车。

Amazon ECS 的指标

Amazon ECS 通过 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 是一项在集群、容器实例、服务和任务级别提供 CPU 利用率、内存利用率、网络流量和存储等指标的CloudWatch功能。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 利用率。Container Insights 将`ECS/ContainerInsights`自定义指标发布到自定义命名空间，您可以使用这些指标绘制图表、警报和仪表板。

您可以通过为每个单独的 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，则可以在 Amazon ECS 集群上将CloudWatch代理作为守护程序容器启动。您可以使用 cwagent-ecs-instance-metric-cfn.yamlAWS CloudFormation 模板将CloudWatch代理部署为亚马逊 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 守护程序CloudWatch容器部署的CloudWatch代理包括额外的磁盘、内存和 CPU 指标，例
如`instance_cpu_reserved_capacity`和`InstanceId`维`instance_memory_reserved_capacity`度为`ClusterName``ContainerInstanceId`、、。容器实例级别的指标由 Container Insights 使用 CloudWatch嵌入式指标格式实现。您可以使用本指南设置州经理和分销商 CloudWatch 代理部署和配置 (p. 17)部分中的方法为 Amazon ECS 容器实例配置额外的系统级指标。

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嵌入的指标格式语句。

以下示例中的`CW_CONFIG_CONTENT`环境变量设置为 `Scwagentconfig` 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，不再有必要。

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

如果您在多个账户和地区部署 Amazon ECS，则可以使用AWS Secrets Manager密钥存储您的 CloudWatch配置并配置密钥策略以与您的组织共享。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`环境变量以连接到用作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_emf_emple 示例 Python 应用程序将指标以嵌入式指标格式发送到配置为嵌入式指标格式端点的CloudWatch代理侧车容器。

的 Flu ent Bit 插件也CloudWatch可用于发送嵌入的指标格式消息。你还可以使用 ecs_firelense_emf_emf_emplase 示例 Python 应用程序以嵌入式指标格式将指标发送到 Firelens for Amazon ECS Sidecar 容器。

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

Amazon EKS 上的日志记录和监控

Amazon Elastic Kubernetes Service (Amazon EKS) 集成与 CloudWatch Kubernetes 控制平面的日志。控制平面由 Amazon EKS 作为托管服务提供，您可以不安装 CloudWatch 代理即可打开日志记录. 这些区域有：CloudWatch 还可以部署代理来捕获 Amazon EKS 节点和容器日志。Fluent Bit 和 Fluentd也支持将容器日志发送到 CloudWatch 日志。

CloudWatch Container Insights 在集群、节点、任务和服务级别为 Amazon EKS 提供综合指标监控解决方案。Amazon EKS 还支持多种指标捕获选项Prometheus. Amazon EKS 控制层面提供指标终端节点以 Prometheus 格式公开指标。您可以将 Prometheus 部署到 Amazon EKS 集群中以使用这些指标。

您还可以设置 CloudWatch 代理抓 Prometheus 指标然后创建 CloudWatch 指标，除了消耗其他 Prometheus 终端节点之外。针对 Prometheus 的容器见解监控还可以从受支持的容器化工作负载和系统中自动发现和捕获 Prometheus 指标。

您可以安装和配置 CloudWatch 亚马逊 EKS 节点上的代理，与 Amazon EC2 与分销商和州经理的方法类似，将 Amazon EKS 节点与标准系统日志记录和监控配置保持一致。

日志记录 Amazon EKS

Kubernetes 日志记录可以分为控制平面日志记录、节点日志记录和应用程序日志记录。这些区域有：Kubernetes 控制层面是一组组组件，用于管理 Kubernetes 集群并生成用于审计和诊断的日志。使用亚马逊 EKS，您可以为不同的控制平面组件打开日志然后将它们发送到 CloudWatch。

Kubernetes 还运行系统组件，例如`kubelet`和`kube-proxy`在运行 Pod 的每个 Kubernetes 节点上。这些组件在每个节点中写入日志，你可以配置 CloudWatch 和容器见解来捕获每个 Amazon EKS 节点的这些日志。

容器被分组为豆荚在 Kubernetes 集群中，并计划在 Kubernetes 节点上运行。大多数容器化应用程序都会写入标准输出和标准错误，容器引擎会将输出重定向到日志记录驱动程序。在 Kubernetes 中，容器日志位于`/var/log/pods`节点上的目录。您可以配置 CloudWatch 以及容器见解来捕获每个 Amazon EKS Pod 的这些日志。

Amazon EKS 控制层面日志记录

Amazon EKS 群集由用于 Kubernetes 集群的高可用性、单租户控制平面和运行容器的 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 集群控制层面节点集成在一起 CloudWatch 你可以为特定的控制平面组件打开日志记录。

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提供Kubernetes 终端节点的服务级别协议 (SLA).

Amazon EKS 节点和应用程序记录

建议使用CloudWatch Container Insights以捕获 Amazon EKS 的日志和指标。容器见解使用 CloudWatch 代理，Fluent Bit 或 Fluentd 用于将日志捕获到 CloudWatch。Container Insights 还提供自动控制面板，其中包含您所捕获的 CloudWatch 指标。容器见解作为 CloudWatch 进行部署 DaemonSet 和 Fluent Bit DaemonSet 在每个亚马逊 EKS 节点上运行。容器见解不支持 Fargate 节点，因为节点由AWS而且不支持 DaemonSets。本指南单独介绍了亚马逊 EKS 的 Fargate 日志记录。

下表显示了 CloudWatch 捕获的日志组和日志默认 Fluentd 或 Fluent Bit 日志捕获配置适用于 Amazon EKS。

/aws/containerinsights/Cluster_Name/application	中的所有日志文件/var/log/containers. 此目录提供了指向所有 Kubernetes 容器日志的符号链接/var/log/pods目录结构。这会捕获写入到的应用程序容器日志stdout要么stderr. 它还包括 Kubernetes 系统容器的日志，例如aws-vpc-cni-init、kube-proxy, 和coreDNS.
/aws/containerinsights/Cluster_Name/host	中的日志/var/log/dmesg、/var/log/secure, 和/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 Fluent Bit 或 Fluentd 的容器见解进行日志记录，则可以使用 CloudWatch Amazon EKS 节点上安装的代理。Amazon EKS 节点是 EC2 实例，这意味着您应该将它们包含在 Amazon EC2 的标准系统级日志记录方法中。如果你安装 CloudWatch 代理使用分销商和州经理，那么 Amazon EKS 节点也包含在 CloudWatch 代理安装、配置和更新。

下表显示了特定于 Kubernetes 的日志，如果您不使用 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	此目录提供了指向所有 Kubernetes 容器日志的符号链接/var/log/pods目录结构。这有效地捕获写入到的应用程序容器日志stdout要么stderr. 这包括 Kubernetes 系统容器的日志，例如aws-vpc-cni-init、kube-proxy, 和coreDNS. 重要提示：如果您使用容器见解，则不需要这样做。
var/log/aws-routed-eni/ipamd.log /var/log/aws-routed-eni/plugin.log	可以在这里找到 L-IPAM 守护程序的日志

您必须确保 Amazon EKS 节点安装并配置 CloudWatch 代理发送适当的系统级日志和指标。但是，Amazon EKS 优化 AMI 不包含系统管理器代理。使用启动模板，您可以自动执行 Systems Manager 代理安装和默认设置 CloudWatch 通过用户数据部分实施的启动脚本捕获重要的 Amazon EKS 特定日志的配置。Amazon EKS 节点使用 Auto Scaling 组作为托管节点组或者作为自行管理的节点.

对于托管节点组，您可以提供启动模板其中包括用于自动执行 Systems Manager 代理安装的用户数据部分和 CloudWatch 配置。你可以自定义和使用amazon_eks_Managed_node_group_launch_config.yaml AWS CloudFormation模板来创建安装 Systems Manager 代理的启动模板， CloudWatch 代理，还将 Amazon EKS 特定的日志记录配置添加到 CloudWatch 配置目录。此模板可用于更新您的 Amazon EKS 托管节点组启动模板，使用 infrastructure-as-code （ iAC ）方法。每次更新AWS CloudFormation模板提供了启动模板的新版本。然后，您可以更新节点组以使用新模板版本并使用托管式生命周期在不停机的情况下更新节点。确保应用于托管节点组的 IAM 角色和实例配置文件包括CloudWatchAgentServerPolicy和AmazonSSMManagedInstanceCore AWS托管策略。

借助自我管理的节点，您可以直接为 Amazon EKS 节点配置和管理生命周期和更新策略。自管节点允许您在 Amazon EKS 群集上运行 Windows 节点，Bottlerocket，以及其他选项. 您可以使用AWS CloudFormation将自管节点部署到 Amazon EKS 群集中，这意味着您可以对 Amazon EKS 群集使用 iAC 和托管更改方法。AWS提供亚马逊 eks-nodegroup.yaml AWS CloudFormation您可以按原样使用或自定义的模板。该模

板为群集中的 Amazon EKS 节点预置所有必需的资源（例如，单独的 IAM 角色、安全组、Amazon EC2 Auto Scaling 组和启动模板）。这些区域有：亚马逊 eks-nodegroup.yaml AWS CloudFormation模板是安装所需的系统管理器代理的更新版本， CloudWatch 代理，还将 Amazon EKS 特定的日志记录配置添加到 CloudWatch 配置目录。

在 Fargate 上登录亚马逊 EKS

使用 Fargate 上的 Amazon EKS，您可以在不分配或管理 Kubernetes 节点的情况下部署 Pod。这就无需为 Kubernetes 节点捕获系统级日志。要从 Fargate pod 中捕获日志，您可以使用 Fluent Bit 直接将日志转发到 CloudWatch。这使您能够自动将日志路由到 CloudWatch 无需进一步的配置或在 Fargate 上为您的亚马逊 EKS 容器配置一个边卡容器。有关此问题的更多信息，请参阅Fargate 日志记录Amazon EKS 文件中的亚马逊 EKS 的流利位在AWS博客。此解决方案捕获STDOUT和STDERR从容器中输入/输出 (I/O) 流并将其发送到 CloudWatch 通过 Fluent Bit，基于为 Fargate 上的 Amazon EKS 集群建立的 Fluent Bit 配置。

Amazon EKS 和 Kubernetes 指标

Kubernetes 提供了一个指标 API，允许您访问资源使用量指标（例如，节点和 Pod 的 CPU 和内存使用情况），但 API 只提供时间点信息，而不提供历史指标。这些区域有：Kubernetes 指标服务器通常用于 Amazon EKS 和 Kubernetes 部署，以聚合指标、提供有关指标的短期历史信息以及支持诸如Horizontal Pod Autoscaler.

亚马逊 EKS 通过 Kubernetes API 服务器公开控制平面指标以 Prometheus 格式和 CloudWatch 可以捕获和摄取这些指标。 CloudWatch 还可以对容器见解进行配置，以便为您的 Amazon EKS 节点和 Pod 提供全面的指标捕获、分析和警报。

Kubernetes 控制层面指标

Kubernetes 通过使用 Prometheus 格式公开控制平面指标`/metrics`HTTP API 终端节点。你应该安装Prometheus在 Kubernetes 集群中使用 Web 浏览器绘制和查看这些指标。您还可以摄取暴露的指标由 Kubernetes API 服务器进入 CloudWatch。

Kubernetes 的节点和系统指标

Kubernetes 提供 Prometheus指标-服务器你可以部署和运行在 Kubernetes 集群上获取集群、节点和 POD 级 CPU 和内存统计信息。这些指标与Horizontal Pod Autoscaler和Vertical Pod Autoscaler. CloudWatch 还可以提供这些指标。

如果您使用Kubernetes 控制面板或者是水平和垂直的 pod 自动缩放器。Kubernetes 控制面板可帮助您浏览和配置 Kubernetes 集群、节点、pod 和相关配置，并从 Kubernetes 指标服务器查看 CPU 和内存指标。您可以按照以下步骤为单个集群部署此解决方案：部署 Kubernetes 控制面板Amazon EKS 文件中的。

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）。这些指标的目的是 point-in-time 分析而不是历史分析。Kubernetes 仪表板部署`dashboard-metrics-scraper`以便在短时间窗口内存储来自 Kubernetes 指标服务器的指标。

容器见解使用的容器化版本 CloudWatch 在 Kubernetes 中运行的代理 DaemonSet 以发现集群中所有正在运行的容器并提供节点级别指标。它在性能堆栈的每个层收集性能数据。您可以使用快速入门AWS快速入门或单独配置容器见解。快速入门将使用 CloudWatch 使用 Fluent Bit 进行代理和日志记录，因此您只需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之外，您还应使用您为 Amazon EC2 定义的标准捕获系统级指标。您可以使用设置州经理和分销商 CloudWatch 代理部署和配置 (p. 17)本指南的部分将安装和配置 CloudWatch Amazon EKS 集群的代理。您可以更新 Amazon EKS 特定的 CloudWatch 配置文件，以包括指标以及 Amazon EKS 特定的日志配置。

这些区域有： CloudWatch 拥有 Prometheus 支持的代理商可以自动发现和抓取 Prometheus 指标支持的容器化工作负载和系统. 它把它们作为 CloudWatch 以嵌入式指标格式进行日志以便分析 CloudWatch 记录见解并自动创建 CloudWatch 指标。

> Important
>
> 您必须部署专用版本的 CloudWatch 代理来收集 Prometheus 指标。这是独立于的代理 CloudWatch 为 Container Insights 部署了代理。您可以使用prometheus_jmx示例 Java 应用程序，其中包括 CloudWatch 代理和 Amazon EKS pod 部署，以演示 Prometheus 指标发现。有关更多信息，请参阅 。在 Amazon EKS 和 Kubernetes 上设置 Java/JMX 示例工作负载在 CloudWatch 文档中。您还可以配置 CloudWatch 代理来捕获 Amazon EKS 集群中运行的其他 Prometheus 目标的指标。

应用程序指标

您可以使用CloudWatch 嵌入式指标格式。要获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。这些区域有： CloudWatch 代理可以配置为亚马逊 EKS 容器中的 sidecar 容器. 这些区域有： CloudWatch 代理配置存储为 Kubernetes ConfigMap 然后由你的阅读 CloudWatch 代理 sidecar 容器启动嵌入式度量格式端点。

您还可以将应用程序设置为 Prometheus 目标，并在 Prometheus 支持下配置 CloudWatch 代理，以发现、抓取指标并将其提取到 CloudWatch 中。例如，您可以使用开源 JMX 导出器使用你的 Java 应用程序来公开 JMX Beans 以供 Prometheus 消费 CloudWatch 代理。

如果您不想使用嵌入式指标格式，还可以使用AWSAPI要么AWS 开发工具包. 但是，我们建议不要采用这种方法，因为它混合了监控和应用程序逻辑。

Fargate 上的亚马逊 EKS 指标

Fargate 会自动配置 Amazon EKS 节点以运行 Kubernetes Pod，因此您无需监控和收集节点级别指标。但是，您必须监控 Fargate 上 Amazon EKS 节点上运行的 Pod 的指标。容器见解目前不适用于 Fargate 上的 Amazon EKS，因为它需要目前不支持的以下功能：

- 目前不支持 DaemonSets。容器见解是通过运行 CloudWatch 作为代理 DaemonSet 在每个群集节点上。
- 不支持 HostPath 永久性卷。这些区域有： CloudWatch 代理容器使用 HostPath 持久性卷作为收集容器指标数据的先决条件。
- Fargate 防止特权容器和访问主机信息。

您可以使用Fargate 的内置日志路由器将嵌入式指标格式语句发送到 CloudWatch。日志路由器使用 Fluent Bit，它具有 CloudWatch 插件，可以配置为支持嵌入式指标格式语句。

您可以通过在 Amazon EKS 集群中部署 Prometheus 服务器以从 Fargate 节点收集指标，从而检索和捕获 Fargate 节点的 POD 级指标。由于 Prometheus 需要持久存储，因此如果您使用 Amazon 弹性文件系统 (Amazon EFS) 进行持久性存储，则可以在 Fargate 上部署 Prometheus。您还可以在 Amazon EC2 支持的节点上部署 Prometheus。有关更多信息，请参阅 。开启监控 Amazon EKSAWS Fargate使用 Prometheus 和 Grafana在AWS博客。

Amazon EKS 上的 Prometheus 监控

Amazon Managed Service for Prometheus提供可扩展、安全、AWS面向开源 Prometheus 的托管服务。您可以使用 Prometheus 查询语言 (ProMQL) 监控容器化工作负载的性能，而无需管理用于摄取、存储和查询运营指标的底层基础设施。您可以使用以下方法从亚马逊 EKS 和亚马逊 ECS 收集 Prometheus 指标AWS对于 Distro OpenTelemetry （ ADOT ）或者 Prometheus 服务器作为收集代理。

Prometheus CloudWatch Container Insights 监控使您可以配置和使用 CloudWatch 代理来发现来自亚马逊 ECS、Amazon EKS 和 Kubernetes 工作负载的 Prometheus 指标，并将它们作为 CloudWatch 指标提取。这个解决方案是合适的， CloudWatch 是你的主要可观察性和监控解决方案。但是，以下列表概述了面向 Prometheus 的亚马逊托管服务为摄取、存储和查询 Prometheus 指标提供了更多灵活性的使用案例：

- 面向 Prometheus 的亚马逊托管服务使您能够使用部署在 Amazon EKS 或自管 Kubernetes 中的现有 Prometheus 服务器，并将其配置为写入针对 Prometheus 的亚马逊托管服务，而不是本地配置的数据存储。这消除了为您的 Prometheus 服务器及其基础设施管理高可用性数据存储所带来的无差别繁重的工作。当您拥有成熟的 Prometheus 部署时，面向 Prometheus 的亚马逊托管服务是合适的选择，而且您想在AWS云。
- Grafana 直接支持 Prometheus 作为可视化的数据源。如果您想将 Grafana 用于 Prometheus 而不是 CloudWatch 用于监控容器的仪表板，然后面向 Prometheus 的亚马逊托管服务可以满足您的要求。面向 Prometheus 的亚马逊托管服务与亚马逊托管 Grafana 集成，以提供托管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，这 CloudWatch 代理程序以嵌入式指标格式提取 Prometheus 指标进入 CloudWatch 导致的日志 CloudWatch 指标。使用以下方法可以查询嵌入式指标格式日志。 CloudWatch 记录见解。
- 如果您不打算使用 CloudWatch 为了进行监控和捕获指标，那么您应该将面向 Prometheus 的亚马逊托管服务与您的 Prometheus 服务器和 Grafana 之类的可视化解决方案一起使用。你需要配置你的 Prometheus 服务器以从你的 Prometheus 目标中抓取指标，然后将服务器配置为将远程写入您的 Amazon Managed Service of Prometheus 工作区. 如果你使用亚马逊托管 Grafana，那么你可以使用随附的插件直接将亚马逊托管 Grafana 与您的 Prometheus 亚马逊托管服务数据源集成. 由于指标数据存储在针对 Prometheus 的亚马逊托管服务中，因此没有依赖关系来部署 CloudWatch 代理或将数据提取到 CloudWatch 的要求。这些区域有： CloudWatch Prometheus 需要代理人进行 Container Insights 监控。

您还可以使用 ADOT Collector 从 Prometheus 仪器的应用程序中抓取数据，然后将指标发送到针对 Prometheus 的亚马逊托管服务。有关 ADOT Collector 的更多信息，请参阅AWSOpenTelemetry 的 Distro文档中）。

的日志和指标AWS Lambda

Lambda 无需管理和监控工作负载的服务器，可自动使用 CloudWatch指标和 CloudWatch 日志，无需进一步配置或检测应用程序的代码。本节帮助您了解 Lambda 使用的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 Lambda 函数，以优化性能和诊断应用程序级问题。

Lambda 函数日志记录

Lambda 可自动将来自 Lambda 函数的标准输出和标准错误消息流式传输到 CloudWatch 日志，无需日志驱动程序。Lambda 还会自动预置运行 Lambda 函数的容器，并将它们配置为在单独的日志流中输出日志消息。

对 Lambda 函数的后续调用可以重用相同的容器并输出到同一个日志流。Lambda 还可以预置新容器并将调用输出到新的日志流。

首次调用您的 Lambda 函数时，Lambda 会自动创建日志组。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，其`/aws/lambda/<YourLambdaFunctionName>`格式为。在日志组中为每个 Lambda 函数实例创建单独的日志流。Lambda 对使用`YYYY/MM/DD/[<FunctionVersion>]<InstanceId>`格式的日志流有标准命名惯例。由生成`InstanceId`，AWS用于识别 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 查询这些消息。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记录，以帮助对日志消息进行格式和分类。例如，如果您的 Lambda 函数是用 Python 编写的，则可以使用 Python 日志模块来记录消息并控制输出格式。Lambda 原生使用 Python 日志库来处理用 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实验室创建了 AWSLambda Powertools for Python 开发者工具包，可以更轻松地使用冷启动等关键数据丰富日志消息。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，然后根据环境和您的要求进行调整。除了使用的库外，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)
```

将日志从发送到其他目的地 CloudWatch

您可以使用订阅筛选器将日志发送到其他目的地（例如，Amazon OpenSearch 服务或 Lambda 函数）。如果您不使用 Amazon Serv OpenSearch ice，则可以使用 Lambda 函数处理日志，并使用软件开发AWS工具包将其发送到您选择的AWS服务。

您还可以在 Lambda 函数中使用AWS云外部日志目标的软件开发工具包，将日志语句直接发送到您选择的目的地。如果您选择此选项，我们建议您考虑延迟、额外处理时间、错误和重试处理以及操作逻辑与 Lambda 函数耦合的影响。

Lambda 函数指标

Lambda 使您无需管理或扩展服务器即可运行代码，这几乎消除了系统级审计和诊断的负担。但是，了解您的 Lambda 函数在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 Lambda 函数的大小，有效监控和衡量性能可以改善用户体验并降低成本。通常，以 Lambda 函数形式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 还为具有这些指标的每个 Lambda 函数提供单独的监控控制面板。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诊断和支持 Lambda 部署。

调用错误使您的 Lambda 函数无法运行。这些错误发生在代码运行之前，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人员和工作负载所有者。这些错误通常与配置或权限错误有关，可能是由于您的配置或权限的更改而发生的。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`指标捕获并识别您的 Lambda 函数运行失败。您应从 Lambda 函数调用中返回格式化响应，其中包含用于确定运行是完全、部分还是成功失败的信息。

CloudWatch 提供 CloudWatch Lambda Insights，您可以为单个 Lambda 函数启用这些见解。Lambda Insights 收集、聚合和汇总系统指标（如CPU、磁盘和网络使用情况）。Lambda Insights 还收集、聚合和汇总诊断信息（如冷启动和 Lambda 工件关闭），以帮助您隔离和快速解决问题。

Lambda Insights 使用嵌入式指标格式根据您的 Lambda 函数名称使用日志流名称前缀自动向日志组发送性能信息。`/aws/lambda-insights/`这些性能日志事件创建的 CloudWatch 指标是自动 CloudWatch 仪表板的基础。我们建议您在性能测试和生产环境中启用 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嵌入式指标格式语句并将其发送到 CloudWatch。集成的 Lambda CloudWatch 日志记录工具配置为处理和提取适当格式的嵌入式指标格式语句。

搜索和分析日志 CloudWatch

将日志和指标捕获到一致的格式和位置后，除了识别和解决问题外，您还可以对其进行搜索和分析，以帮助提高运营效率。我们建议您以格式正确的格式（例如 JSON）捕获日志，以便于搜索和分析日志。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工作AWS负载。

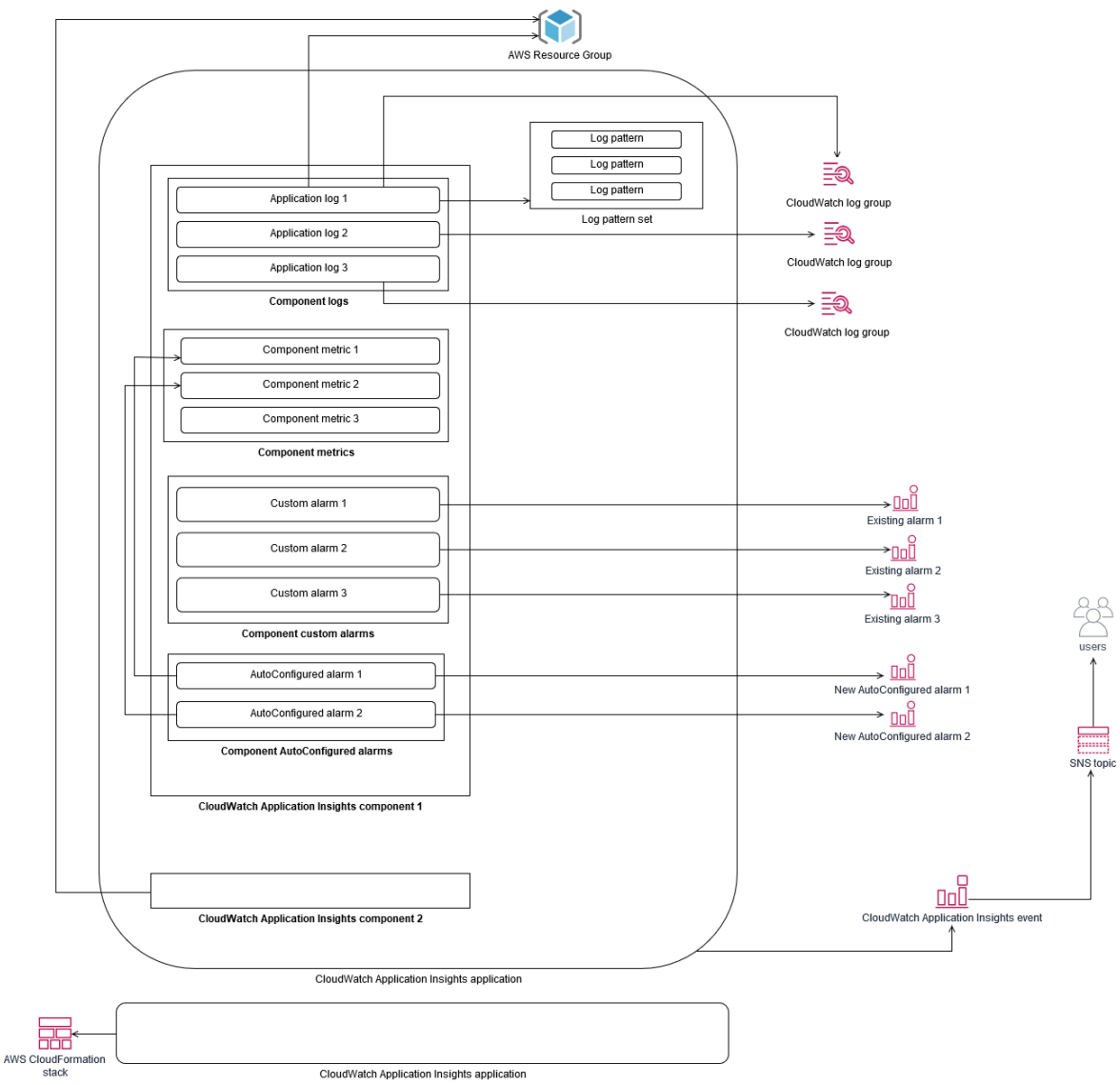
CloudWatch 提供多种功能来帮助分析日志和指标，例如CloudWatch Application Insight s 用于共同定义和监控跨不同AWS资源的应用程序的指标和日志，CloudWatch 异常检测可显示您的异常情况指标和 CloudWatch Log Insights，通过交互方式搜索并分析 Lo CloudWatch gs 中的日志数据。

使用 “应用程序洞察” 共同监控和分析 CloudWatch 应用程序

应用程序所有者可以使用 Amazon A CloudWatch pplication Insights 为工作负载设置自动监控和分析。除了为账户中的所有工作负载配置标准系统级监控外，还可以配置此功能。通过 Applicat CloudWatch ion 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，可帮助团队划分记录和监控职责。

CloudWatch Application Insights 使用资源组来确定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变成 Application Insights CloudWatch 应用程序中单独定义的组件。您的 Applicat CloudWatch ion Insights 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，您可以定义应用于组件和 Application Insights CloudWatch 应用程序的日志模式集。日志模式集是要根据正则表达式搜索的日志模式集合，以及检测到该模式时的低、中或高严重性。对于指标，您可以从特定服务和支持的指标列表中为每个组件选择要监控的指标。对于警报，A CloudWatch pplication Insights 会自动为受监控的指标创建和配置标准或异常检测警报。 CloudWatch Application Insights 可自动配置日志中概述的技术的指标和日志捕获，以及 CloudWatch 文档中 App CloudWatch lication Insights 支持的指标。下图显示了 Applicat CloudWatch ion Insights 组件及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，以使用 CloudWatch 日志和指标进行监控。



由 Applicat CloudWatch ion 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 Application Insights 配置 CloudWatch 应用程序的先决条件。 CloudWatch Application Insights 使用Systems Manager 来安装和更新 CloudWatch 代理。在 App CloudWatch lication Insights 中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 App CloudWatch lication Insights 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。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，应确保配置不重叠。

使用日志洞察进行 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以 JSON 格式构建，L CloudWatch ogs Insights 会自动在多个日志组中的日志流中发现 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future 使用。Log

CloudWatch s Insights 的查询语法支持诸如聚合函数之类的函数，例如 sum ()、avg ()、count ()、min () 和 max ()，这些函数可能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。这有助于根据需要捕获生成特定指标所需的数据点，而不是主动将其捕获为自定义指标，从而降低 CloudWatch 监控成本。这对于具有高基数且会产生大量指标的维度尤其有效。 CloudWatch Container Insights 也采用这种方法来捕获详细的性能数据，但仅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根据嵌入式 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，您可以运行以下查询来查看最新 20 个出现内存页错误的 pod：

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

使用亚马逊 OpenSearch 服务进行日志分析

CloudWatch 与 Amazon S OpenSearch er vice 集成，允许您使用订阅筛选器将 CloudWatch 日志数据从日志组流式传输到您选择的 Amazon Serv OpenSearch ice 集群。您可以使用 CloudWatch 主要的日志和指标捕获和分析，然后在以下用例中使用 Amazon S OpenSearch ervice 对其进行扩展：

- 细粒度的数据访问控制 — Amazon Serv OpenSearch ice 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、区域和基础设施汇总和搜索日志-您可以将来自多个账户和区域的日志流式传输到一个通用的 Amazon Serv OpenSearch ice 集群中。您的集中化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 日志传输到 Amazon Serv OpenSearch ice 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 Serv OpenSearch ice 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 Elast ic Beats 系列数据采集器和 Logstash，它们在将日志数据发送到亚马逊 OpenSearch 服务之前支持日志丰富和转换。
- 现有的运营管理解决方案使用 LogstashElasticSearch、Kibana (ELK) 堆栈进行日志记录和监控 — 您可能已经在 Amazon OpenSearch Service 或开源 Elasticsearch 上投入了大量资金，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、logstash 和 O pen Distro for ElasticSearch API）将日志直接发送到亚马逊 OpenSearch 服务并绕过 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 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器，以将其日志流式传输到 Amazon Serv OpenSearch ice。您可以为要传输到 Amazon Serv OpenSearch ice

的每个单独的日志组手动配置订阅筛选器。或者，您可以部署一个自动为 ElasticSearch 集群订阅新日志组的解决方案。您可以将日志流式传输到同一账户或集中账户中的 ElasticSearch 集群。将日志流式传输到同一个账户中的 ElasticSearch 集群有助于工作负载所有者更好地分析和支持他们的工作负载。

您应该考虑在集中式或共享账户中设置 ElasticSearch 集群，以汇总您的账户、区域和应用程序的日志。例如，AWS Control Tower设置一个用于集中记录的日志存档帐户。在中创建新账户时AWS Control Tower，其AWS CloudTrail和AWS Config日志将传送到该集中账户中的 S3 存储桶。仪表的日志记录AWS Control Tower用于配置、更改和审核日志。

要使用 Amazon OpenSearch Service 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 OpenSearch Service 集群部署到您的集中式日志账户，并在其他账户中配置日志组，将日志流式传输到集中式 Amazon OpenSearch 服务集群。

您可以创建单独的 Amazon Serv OpenSearch ice 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 Serv OpenSearch ice 集群可以帮助您降低安全和可用性风险，而拥有一个通用 Amazon Serv OpenSearch ice 集群可以更轻松地在同一个集群内搜索和关联数据。

CloudWatch 提供警报选项

对重要指标执行一次性自动分析，有助于在问题影响工作负载之前检测并解决问题。 CloudWatch 通过在特定时间段内使用多个统计数据，可以轻松绘制和比较多指标。您可以使用 CloudWatch 以搜索具有所需维度值的所有指标，以查找分析所需的指标。

我们建议您首先包含一组初始指标和维度，用作监控工作负载的基线，从而开始采用指标捕获方法。随着时间的推移，工作负载会成熟，您可以添加额外的指标和维度来帮助您进一步分析和支持它。应用程序或工作负载可能使用多AWS资源并拥有自己的自定义指标，则应将这些资源分组到命名空间下，以便更易于识别它们。

您还应考虑日志记录和监控数据之间的关联性，以便快速识别相关的日志记录和监控数据以诊断具体问题。您可以使用CloudWatch ServiceLens将跟踪、指标、日志和警报关联起来以诊断问题。您还应考虑在工作负载的日志中包括额外的维度和标识符，以帮助您快速搜索和识别系统和服务中的问题。

使用 CloudWatch 警报以监控和警报

您可以使用CloudWatch 警报以减少工作负载或应用程序中的手动监控。首先，您应该检查为每个工作负载组件捕获的指标，然后为每个指标确定适当的阈值。确保您确定在超出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，自动创建新票证并运行运营工作流程。例如，AWS提供AWS的服务管理连接器ServiceNow和Jira 服务台以帮助您迅速设置集成。这种方法对于确保已提出的警报得到确认并与可能已在这些产品中定义的现有运营工作流程保持一致至关重要。

您还可以为具有不同阈值和评估周期的同一指标创建多个警报，这有助于建立上报流程。例如，如果您拥有`OrderQueueDepth`跟踪客户订单的指标，您可以在短短的平均一分钟时间内定义一个较低的阈值，通过电子邮件通知应用程序团队成员或Slack. 您还可以在同一阈值的 15 分钟内为同一指标定义另一个警报，并通知应用程序团队和应用程序团队的负责人。最后，您可以为 30 分钟内的硬平均阈值定义第三个警报，通知上层管理层并通知先前通知的所有团队成员。创建多个警报可帮助您针对不同情况采取不同的操作。您可以从简单的通知过程开始，然后根据需要进行调整和改进。

使用 CloudWatch 用于监控和报警的异常检测

您可以使用CloudWatch 异常检测如果您不确定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。 CloudWatch 异常检测对于可能会发生定期、可预测的活动变化的指标特别有用，例如，在截止时间之前增加当天送达的每日采购订单。异常检测可启用自动调整的阈值，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，然后配置 CloudWatch 根据异常值进行警报。

例如，您可以为`CPUUtilization`指标和`AVG`EC2 实例的统计数据。然后异常检测会使用长达 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频段的多个警报来建立警报上报流程，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，请参阅根据异常检测创建 CloudWatch 告警中的 CloudWatch 文档中）。

在多个区域和账户中发生警报

应用程序和工作负载所有者应为跨越多个区域的工作负载创建应用级警报。我们建议在部署工作负载的每个账户和区域内创建单独的警报。您可以通过使用与账户和区域无关的功能来简化和自动化此流程AWS

CloudFormation StackSets 和模板来部署具有所需警报的应用程序资源。模板您可以配置警报操作以针对常见的 Amazon Simple Notification Service (Amazon SNS) 主题，这意味着无论账户或地区如何，都使用相同的通知或补救措施。

在多账户和多区域环境中，我们建议您为账户和地区创建聚合警报，以便通过使用AWS CloudFormation StackSets 和聚合指标，例如平均`CPUUtilization`跨所有 EC2 实例。

您还应考虑为标准配置的每个工作负载创建标准警报 CloudWatch 您捕获的指标和日志。例如，您可以为每个 EC2 实例创建单独的警报，以监控 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，每天监控平均 CPU 使用率低于 10%。这些警报有助于中央运营团队与特定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 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。您可以通过使用亚马逊云监控自动警报解决方案，可为您的 EC2 实例自动创建一组标准 CloudWatch 警报，并根据 EC2 实例标签创建自定义警报。该解决方案不再需要手动创建标准警报，在使用 CloudEndure 等工具的 EC2 实例的大规模迁移期间可能非常有用。您还可以使用部署此解决方案AWS CloudFormation StackSets 以支持多个区域和账户。有关更多信息，请参阅 。使用标签创建和维护亚马逊 CloudWatch Amazon EC2 实例的警报以警报告在AWS博客。

监控应用和服务可用性

CloudWatch 可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。您还应监控应用程序和工作负载的可用性和可访问性方面。您可以使用主动监控方法来实现此目标Amazon Route 53 运行状况检查和CloudWatch Synthetics.

如果要通过 HTTP 或 HTTPS 监控与网页的连接，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，可以使用 Route 53 运行状况检查。Route 53 运行状况检查以 10 秒或 30 秒的间隔启动来自您指定的区域的连接。您可以为运行状况检查选择多个区域，每个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用于运行状况检查评估的前 5,120 个字节的数据中，则可以在 HTTP 或 HTTPS 请求的响应正文中搜索特定子字符串。如果返回 2xx 或 3xx 响应，则会将 HTTP 或 HTTPS 请求视为运行状况良好。Route 53 运行状况检查可以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，并且希望在其中一个终端节点运行状况不佳时执行相同的通知，则可以执行此操作。如果您使用 Route 53 作为 DNS，则可以将 Route 53 配置为故障切换到另一个 DNS 条目如果运行状况检查变得不健康。对于每个关键工作负载，您应考虑为对正常操作至关重要的外部端点设置 Route 53 运行状况检查。Route 53 运行状况检查有助于避免将故障转移逻辑写入应用程序。

CloudWatch 合成器允许您将 Canary 定义为脚本，以评估工作负载的运行状况和可用性。Canate 是用 Node.js 或 Python 编写的脚本，可以使用 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 Canary 都可以对不同的终端执行多次 HTTP 或 HTTPS 调用。这意味着您可以监控一系列步骤的运行状况，例如使用案例或具有下游依赖关系的终端节点。Canate 创建 CloudWatch 包括运行的每个步骤的指标，以便您可以独立发出警报和衡量不同步骤。尽管加那利群岛比 Route 53 运行状况检查需要更多的规划和努力来开发，但它们为您提供了高度可定制的监控和评估方法。加那利群岛还支持在虚拟私有云 (VPC) 内运行的私有资源，这使得当您没有终端节点的公有 IP 地址时，它们非常适合监控可用性。只要 VPC 内部连接到终端节点，您还可以使用 Canary 监控本地工作负载。当您的工作负载包括本地存在的终端节点时，这一点尤为重要。

使用跟踪应用AWS X-Ray

通过应用程序发出的请求可能包括对在本地服务器、Amazon EC2、容器或 Lambda 中运行的数据库、应用程序和 Web 服务的调用。通过实施应用程序跟踪，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以跨多个组件跟踪应用程序请求。X-Ray 采样和可视化请求服务图当它们流过你的应用程序组件并且每个组件都表示为区段时。X-Ray 会生成跟踪标识符，以便在请求流过多个组件时可以关联请求，这有助于您从端到端查看请求。您可以通过包含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 配置和检测应用程序中的每个服务器或终端节点。X-Ray 是通过调用 X-Ray 服务在应用程序代码中实现的。X-Ray 还提供AWS适用于多种语言的 SDK，包括自动向 X-Ray 发送数据的分析客户端。X-Ray SDK 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，您可以在 Amazon EC2 和 Amazon ECS 上安装并运行该守护程序，以便将数据中 X-Ray 会为应用程序创建跟踪，从运行服务请求的 X-Ray 守护程序的服务器和容器中捕获性能数据。X-Ray 会自动计算你的来电AWS通过修补 Amazon DynamoDB 等服务作为子区段AWS发工具包。X-Ray 还可以自动与 Lambda 函数集成。

如果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，则可以创建用于封装对外部服务的呼叫的子区段. X-Ray 相关 CloudWatch 如果您使用的是应用程序跟踪的日志和指标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 守护程序以便在 Amazon EC2 上跟踪应用程和服务

您需要在运行应用程序组件或微服务的 EC2 实例上安装并运行 X-Ray 守护程序。您可以使用用户数据脚本在配置 EC2 实例时部署 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 构建过程中。当 EC2 实例是短暂的时候，这可能特别有用。

您应该使用状态管理器来确保在 EC2 实例上始终安装 X-Ray 守护程序。对于 Amazon EC2 EC2 的Windows 实例，你可以使用 Systems ManagerAWS-运行 PowerShell 脚本文档运行Windows 脚本下载并安装 X-Ray 代理。对于 Linux 上的 EC2 实例，您可以使用AWS-runshellScript 文档来运行 Linux 脚本将代理下载并安装为服务.

您可以使用 Systems ManagerAWS-运行 RemoteScript 文档在多账户环境中运行脚本。您必须创建一个可从所有账户访问的 S3 存储桶，我们建议使用基于组织的存储桶策略创建 S3 存储桶如果您使用AWS Organizations. 然后，您将脚本上传到 S3 存储桶，但请确保 EC2 实例的 IAM 角色有权访问存储桶和脚本。

您还可以配置状态管理器以将脚本与安装了 X-Ray 代理的 EC2 实例关联。由于您的所有 EC2 实例可能不需要或使用 X-Ray，因此您可以定位与实例标签的关联。例如，您可以根据存在的`InstallAWSXRayDaemonWindows`要么`InstallAWSXRayDaemonLinux`标签。

在 Amazon ECS 或 Amazon EKS 上部署 X-Ray 守护程序跟踪应用程和服务

你可以部署X-Ray 守护作为面向基于容器的工作负载的 sidecar 容器，例如 Amazon ECS 或 Amazon EKS。然后，如果您使用 Amazon ECS，应用程序容器就可以通过容器链接连接到 sidecar 容器；如果您使用，则该容器可以直接连接到 localhost 上的 sidecar 容器awsvpc 网络模式.

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通过本地主机连接到您指定的容器端口上的守护程序。

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。您无需安装适用于 Lambda 的 X-Ray 守护程序，因为守护进程完全由 Lambda 管理，用户无法配置。你可以使用你的 Lambda 函数启用它AWS Management Console然后检查活动跟踪在 X-Ray 控制台中选项。

如需进一步分析，您可以将 X-Ray 开发工具包与 Lambda 函数绑定，以便记录传出调用以及添加注释或元数据。

分析 X-Ray 应用程

您应该评估符合应用程序编程语言的 X-Ray SDK，并对应用程序对其他系统发出的所有调用进行分类。查看您选择的库提供的客户端，看开发工具包是否可以针对应用程序的请求或响应自动进行仪器跟踪。确定 SDK 提供的客户端是否可用于其他下游系统。对于应用程序调用但无法使用 X-Ray 进行仪器的外部系统，应创建自定义子区段来捕获并在跟踪信息中识别它们。

在检测应用程序时，请确保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为客户使用标识符，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。

您最多可以为每个跟踪创建 50 个批注，但是，只要区段文档不超过 64 千字节，则可以创建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释来查找信息，并使用元数据对象提供更多上下文，以帮助在查找请求后对其进行故障排除。

配置 X-Ray 采样规则

By自定义采样规则，您可以控制您记录的数据量以及修改采样行为而不必修改或重新部署您的代码。采样规则告知 X-Ray 开发工具包要为一系列条件记录的请求数。默认情况下，X-Ray SDK 将记录。第一个请求每秒和任何额外请求的 5%. 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。5% 是对超出容器尺寸的额外请求进行采样的比率。

您应该查看并更新默认配置，以确定账户的合适值。在开发环境、测试环境、性能测试环境和生产环境中，您的要求可能不 您可能有一些应用程序需要根据接收的流量或严重程度自己的采样规则。您应该从基线开始，然后定期重新评估基准是否符合您的要求。

使用 CloudWatch 进行仪表板和可视

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 提供了自动仪表板，您还可以轻松创建使用 CloudWatch 指标。 CloudWatch 仪表板提供的洞察力比单独查看指标更多，因为它们可以帮助您关联多个指标并确定趋势。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跨多个关联工作负载指标的变化AWS当您的订单数量增加或减少时的资源。

您应该在账户和应用程序级别创建仪表板来监控工作负载和应用程序。您可以使用开始使用 CloudWatch 自动仪表板，它们是AWS预先配置了服务特定指标的服务级别仪表板。自动服务仪表板显示所有标准 CloudWatch 服务的指标。自动仪表板将绘制每个服务指标使用的所有资源，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利用率的资源，从而帮助您优化成本。

创建跨服务控制面板

您可以通过查看自动服务级别控制面板来创建跨服务仪表板AWS服务和使用添加到控制面板选项来自操作菜单。然后，您可以将其他自动仪表板中的指标添加到新仪表板中，并删除指标以缩小仪表板的焦点。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的跨服务仪表板，其中涵盖关键指标并显示账户中的所有工作负载。

如果您的云运营团队有中央办公空间或公共区域，则可以显示 CloudWatch 大型电视显示器上的仪表板以全屏模式自动刷新。

创建应用程序或工作负载特定仪

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仪表板专注于自定义应用程序或工作负载指标AWS影响其性能的资源指标。

你应该定期评估和自定义 CloudWatch 应用程序或工作负载仪表板，可在事件发生后跟踪关键 在引入或停用功能时，还应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，对工作负载和特定于应用程序的仪表板的更新应该是持续提高质量的必要活动。

创建跨账户或跨区域的控制面板

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。如果你有一个集中的日志记录和监控帐户，这非常有用。

账户所有者和应用程序团队所有者应为特定于账户的跨区域应用程序创建仪表板，以便在集中位置有效监控关键指标。CloudWatch 仪表板自动支持跨区域小部件，这意味着您可以创建包含来自多个区域的指标的仪表板，而无需进一步配

一个重要的例外是 CloudWatch 日志见解小组件，因为只能为您当前登录的账户和地区显示日志数据。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，当你需要进一步分析这些日志时，你可以切换到特定的区域。

运营团队应创建集中式控制面板，以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，其中包括每个账户和区域的总 CPU 利用率。您还可以使用指标数学跨多个账户和地区聚合和控制数据。

使用指标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工作负载相关的格式和表达式中的指标。计算的指标可以保存并在仪表板上查看，以便进行跟踪。例如，标准的 Amazon EBS 数量指标提供了读取数量 (`VolumeReadOps`) 然后写 (`VolumeWriteOps`) 在特定时间段内执行的操作。

但是，AWS提供了有关 IOPS 中 Amazon EBS 卷性能的指南。您可以在指标数学中绘制并计算 Amazon EBS 体积的 IOPS，方法是添加`VolumeReadOps`和`VolumeWriteOps`然后除以为这些指标选择的时间段。

在此示例中，我们总结期间内的 IOPS，然后除以期间长度以获得 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。有关使用指标数学监控 Amazon Elastic File System (Amazon EFS) 文件系统的更多信息和示例 CloudWatch 指标，请参阅亚马逊 CloudWatch 指标数学简化了对 Amazon EFS 文件系统的近实时监控等在AWS博客。

使用适用于 Amazon ECS、Amazon EKS 和 Lambda 的自动控制面板 CloudWatchContainer 见解和 CloudWatch Lambda Insights

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、自动控制面板。您应该启用容器 Insights，以便能够观察 CPU、内存、磁盘、网络和诊断信息，例如容器重启失败。容器见解会生成动态仪表板，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。Container Insights在群集和节点或容器实例级别配置这取决于AWS服务。

类似于容器见解， CloudWatch Lambda Insights 为您的 Lambda 函数创建动态、自动的仪表板。此解决方案收集、聚合和汇总系统级指标，包括 CPU 时间、内存、磁盘和网络。它还收集、聚合和汇总诊断信息（如冷启动和 Lambda 工件关闭），以帮助您隔离和快速解决 Lambda 函数的问题。Lambda 已在函数级别启用，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 跟踪和服务映射以可视化容器工作负载。他们都使用 CloudWatch 要捕获的嵌入式指标格式 CloudWatch 指标和性能日志。

你可以创建共享 CloudWatch 使用容器见解和 Lambda Insights 捕获的指标的工作负载仪表板。您可以通过筛选和查看自动控制面板来达到此目的。 CloudWatch 容器见解，然后选择添加到仪表板选项允许您将显示的指标添加到标准 CloudWatch 控制面板。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负载。

与 CloudWatch 集成AWS服务

AWS提供了许多服务，其中包括日志记录和指标的其他配置选项 这些服务通常使您能够配置 CloudWatch 日志输出的日志和 CloudWatch 指标输出的指标。用于提供这些服务的底层基础架构由AWS和无法访问，但是您可以对预配的服务使用日志记录和指标选项来获取进一步的见解并解决问题。例如，您可以发布将 VPC 流日志发送到 CloudWatch，或者你也可以将 Amazon Relational Database Service（Amazon RDS）实例配置为将日志发布到 CloudWatch.

大多数AWS使用记录 API 调用集成到AWS CloudTrail. CloudTrail 也支持与 的集成 CloudWatch 日志这意味着你可以搜索和分析活动AWS服务。您还可以使用 Amazon Service CloudWatch 或 Amazon Events EventBridge 创建和配置自动化和通知 CloudWatch 在中执行的特定操作的事件事件规则AWS服务。某些服务直接集成和 CloudWatch 事件和 EventBridge。您还可以创建通过 CloudTrail 交付的活动.

亚马逊托管 Grafana 用于仪表板和可视化

Amazon Managed Grafana可以用来观察和可视化你的AWS工作负载。亚马逊托管 Grafana 可帮助您大规模可视化和分析您的运营数据。Grafana是一个开源分析平台，可帮助您查询、可视化、提醒和了解您的指标，无论它们存储在何处。如果您的组织已经使用 Grafana 对现有工作负载进行可视化，并且您希望将覆盖范围扩展到AWS工作负载。您可以将亚马逊托管 Grafana 与 CloudWatch 通过将其添加为数据源，这意味着您可以使用创建可视化效果 CloudWatch指标。Amazon MGrafanaAWS Organizations你可以使用集中管理仪表板 CloudWatch 多个账户和区域中的指标。

下表列出了使用 Amazon Managed Grafana 而不是 Grafana 的优势和注意事项 CloudWatch 用于控制面板。根据最终用户、工作负载和应用程序的不同要求，混合方法可能更合适。

创建可视化和仪表板，与亚马逊托管 Grafana 和开源 Grafana 支持的数据源集成	Amazon Managed Grafana 可帮助您从许多不同的数据源创建可视化和仪表板，包括 CloudWatch 指标。Amazon Managed Grafana 包含许多内置数据源，这些数据源跨越AWS服务、开源软件和 COTS 软件。有关此问题的更多信息，请参阅内置数据源在亚马逊托管 Grafana 文档中。您还可以通过将工作区升级到Grafana 企业企业. Grafana 也支持数据源插件允许您与不同的外部系统进行通信。CloudWatch控制面机制需要 CloudWatch 指标或 CloudWatch Logs Insights 查询要显示的数据显示在 CloudWatch 控制面板。
将对仪表板解决方案的访问权限与您的控制面板解决方案分开管理AWS账户访问	亚马逊托管 Grafana 需要使用AWS IAM Identity Center (successor to AWS Single Sign-On) (IAM 身份中心) 和AWS Organizations进行身份验证和授权。这样，您就可以使用可能已经用于 IAM Identity Center 的身份联合身份验证用户向 Grafana 或AWS Organizations. 但是，如果您没有使用 IAM 身份中心或AWS Organizations，然后将其设置为亚马逊托管 Grafana 设置流程的一部分。如果您的组织限制了 IAM Identity Center 的使用，或者AWS Organizations.
使用跨多个账户和区域摄取和访问数据AWS Organizations一体化	Amazon MGrafanaAWS Organizations使您能够从中读取数据AWS来源，例如 CloudWatch 和 Amazon OpenSearch 为您的所有账户提供服务。这使得创建仪表板成为可能，这些仪表板使用账户中的数据显示可视化。自动启用数据访问权限AWS Organizations中，您需要在 Amazon Managed Grafana 工作空间中设置AWS Organizations管理账户。基于以下原因不建议这样做AWS Organizations管理账户的最佳实践. 相比之下， CloudWatch 也支持跨账户、跨区域的控制面板 CloudWatch 指标.
使用开源社区中提供的高级可视化控件和 Grafana 定义	Grafana 提供了大量可视化集合，您可以在创建仪表板时使用这些可视化效果。还有一个大型的社区贡献仪表板库，您可以根据自己的要求编辑和重复使用这些仪表板。

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 部署中导入和导出仪表板，并对其进行自定义以在亚马逊托管 Grafana 中使用。亚马逊托管 Grafana 允许您将 Grafana 作为仪表板解决方案进行标准化。
工作区、权限和数据源的高级设置和配置	借助 Amazon Managed Grafana，您可以创建多个 Grafana 工作区，这些工作区具有自己的已配置数据源、用户和策略集。这可以帮助您满足更高级的使用案例要求以及高级安全配置。如果你的团队还不具备这些技能，这些高级功能可能需要他们增加使用 Grafana 的经验。

使用设计和实施日志记录和监控 CloudWatch 常见问题

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

在何处存储 CloudWatch 如何配置文件？

这些区域有： CloudWatch 适用于 Amazon EC2 的代理可以应用存储在 CloudWatch 配置目录。理想情况下，您应将 CloudWatch 配置存储为一组文件，因为您可以对多个账户和环境进行版本控制并再次使用它们。有关此问题的更多信息，请参阅管理 CloudWatch配置 (p. 6)本指南的部分。或者，您可以将配置文件存储在存储库中 GitHub 并在预配置新的 EC2 实例时自动检索配置文件。

警报发出时，如何在我的服务管理解决方案中创建票证？

您将服务管理系统与 Amazon Simple Notification Service (Amazon SNS) 主题集成并配置 CloudWatch 警报用于在发出警报时通知 SNS 主题。您的集成系统会收到 SNS 消息，并可以使用服务管理系统 API 或 SDK 创建票证。

如何使用？ CloudWatch 要在我的容器中捕获日志文件？

可以将亚马逊 ECS 任务和 Amazon EKS pod 配置为自动将标准输出和 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输出发送到 STDOUT 和 STDERR。这也包括在十二因素应用宣言.

但是，如果你想将特定的日志文件发送到 CloudWatch 然后，您可以在 Amazon EKS pod 或 Amazon ECS 任务定义中挂载卷到应用程序将写入批量文件的位置，然后使用 Fluentd 或 Fluent Bit 的 sidecar 容器将日志发送到 CloudWatch。你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`. 有关此问题的更多信息，请参阅查看容器或服务的日志在 Docker 文档中。

我如何监控健康问题AWS如何？

您可以使用AWS Health Dashboard进行监控AWS运行状况事件。您也可以参阅aws-Health 工具 GitHub 相关的示例自动化解决方案库AWS运行状况事件。

如何创建自定义 CloudWatch 衡量指标何时不存在代理支持？

您可以使用嵌入式指标格式将指标提取到 CloudWatch 中。您还可以使用AWSSDK（例如，put_metric_data),AWS CLI（例如，put-metric-data)，或AWS（例如，PutMetricData) 创建自定义指

标。你应该考虑如何长期维护任何自定义逻辑。一种方法是使用带有集成嵌入式指标格式支持的 Lambda 来创建指标，以及 CloudWatch 事件事件计划规则以确定指标的周期。

如何将我现有的日志记录和监控工具与AWS?

你应该参考软件或服务供应商提供的指导来集成AWS. 您可以使用代理软件、SDK 或提供的 API 向其解决方案发送日志和指标。您也许还可以使用按照供应商的规格配置的开源解决方案，例如 Fluentd 或 Fluent Bit。您也可以使用AWS开发工具包 CloudWatch 使用 Lambda 和 Kinesis Data Streams 记录订阅筛选条件，以创建自定义日志处理器和发货人。最后，如果您使用多个账户和区域，您还应考虑如何集成软件。

资源

介绍

- AWSWell-Architected

有针对性的业务成果

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

规划 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 实例和本地部署服务器中收集指标和日志
- cloudwatch-config-s3-bucket.yaml
- 使用向导创建 CloudWatch 代理配置文件
- 企业 DevOps：为什么要运行自己构建的内容
- 将日志数据导出到 Amazon S3
- Amazon Serv OpenSearch ice 中的精细访问控制
- Lambda 配额
- 手动创建或编辑 CloudWatch 代理配置文件
- 使用订阅实时处理日志数据
- 可供开发的工具AWS

为 EC2 实例和本地部署服务器配置 CloudWatch 代理

- Amazon EC2 指标维度
- 具爆发能力的实例
- CloudWatch 代理预定义指标集
- 使用 procstat 插件收集流程指标
- 为 procstat 配置 CloudWatch 代理
- 对实例启用或禁用详细监控
- 采用 CloudWatch 嵌入式指标格式摄取高基数日志并生成指标
- 使用日志组和日志流
- 列出实例的可用 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 Amazon EC2 和本地服务器的代理安装方法

- 为混合环境创建 IAM 服务角色
- 为混合环境创建托管式实例激活
- 创建 IAM 角色和用户以用于代 CloudWatch 理
- 使用命令行下载和配置 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凭证？
- 堆栈集操作的先决条件
- 使用竞价型实例

Amazon ECS 上的日志记录和监控

- amazon-cloudwatch-logs-for-fluent-bit
- 亚马逊 ECS CloudWatch 指标
- Amazon ECS 容器洞察指标
- Amazon ECS 容器代理
- Amazon ECS 启动类型
- 部署 CloudWatch 代理以收集 Amazon ECS 上的 EC2 实例级指标
- ecs_cluster_with_cloudwatch_linux.yaml
- ecs_cw_emf_example
- ecs_firelense_emf_example
- ecs-task-nginx-firelense.json
- 检索 Amazon 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 上的日志记录和监控

- Amazon EKS 控制面板日志记录
- amazon_eks_managed_node_group_launch_config.yaml
- Amazon EKS
- amazon-eks-nodegroup.yaml
- 亚马逊 EKS 服务等级协议
- Container Insights Prometh
- 使用 Prometheus 控制平面指标
- 部署 Kubernetes 控制面板 (Web UI)
- Fargate 日志记录
- Fluent Bit 适用于亚马逊 EKS 在 Fargate
- 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 Metrics Server
- 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 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 a DaemonSet 以将日志发送到 Lo CloudWatch gs
- 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
- 垂直吊舱自动调节器

的日志记录和指标AWS Lambda

- Lambda 调用错误
- 记录 — Python 的日志记录工具
- 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 日志数据传输到亚马逊 OpenSearch 服务

带有警报功能的选项 CloudWatch

- amazon-cloudwatch-auto-alarms
- AWS用于 Jira 服务管理的服务管理连接器
- 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 Amazon EC2 上运行 X-Ray 守护进程
- 将跟踪数据发送到 X-Ray
- X-Ray 中的服务图

带有的仪表板和可视化效果 CloudWatch

- Amazon M CloudWatch etric Math 简化了对亚马逊 EFS 文件系统的近乎实时的监控
- 设置 CloudWatch 容器见解
- 使用指标数学

CloudWatch 与AWS服务集成

- AWS CloudTrail 支持的服务和集成
- CloudWatch 来自支持服务的事件事件示例
- 活动通过以下方式提供 CloudTrail
- CloudTrail 使用日志监视 CloudWatch 日志文件
- 将数据库引擎日志发布到 Lo CloudWatch gs
- 将流日志发布到 Lo CloudWatch gs

亚马逊管理的 Grafana 用于仪表板制作和可视化

- 中的管理账户的最佳实践AWS Organizations
- Amazon 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

下表描述了本指南的重大更改。如果您想收到有关future 更新的通知，可以订阅 RSS 提要。

变更	说明	日期
更新了日志信息 (p. 55)	更新了有关日志记录的部分AWS Lambda。	2023 年 4 月 17 日
更新了配置信息 (p. 55)	更新并重命名了关于创建和存储 CloudWatch 配置的部分。	2023 年 2 月 9 日
更新的指标信息 (p. 55)	更新了 A mazon ECS 指标部分中的自定义应用程序指标信息。	2023 年 1 月 31 日
删除了预览通知 (p. 55)	亚马逊Grafana 包指标。	2022 年 5 月 25 日
已删除的部分 (p. 55)	CloudWatch 不再支持。	2022 年 1 月 7 日
首次出版 (p. 55)	—	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。