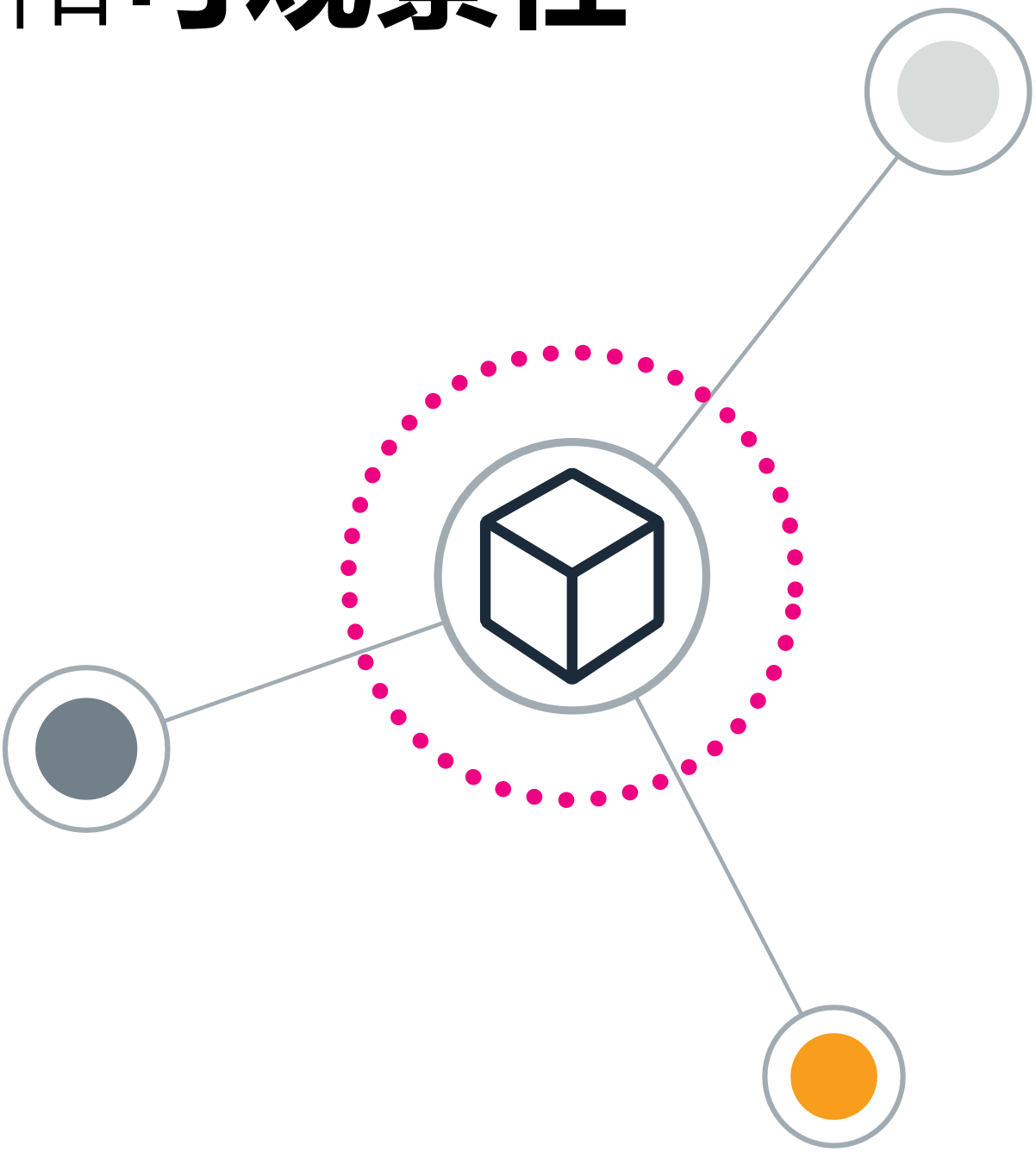
容器监控和可观察性



简介

现在世界上超过 50% 的人口通过手机连接到互联网，开发新的应用程序已经成为任何数字业务的核心要求。这一核心需求的出现为基础设施领域带来了范式转变。作为云之旅的一部分，公司正在采用云原生技术来提高开发这些新应用程序的速度。云本地计算基金会进行的一项调查显示，采用云原生技术的最大好处是部署时间更快。随之而来的是可扩展性、云可移植性和可用性的提高。

容器已经成为最受欢迎的云原生基础设施组件之一，可以更快地开发和测试新应用程序。除了像 Docker 这样的容器技术，开发运维团队还利用像 Kubernetes 这样的容器编排技术来大规模管理和操作容器。

本白皮书强调了监控在 AWS 容器环境中的作用，以及有效监控这些容器环境的关键注意事项。它还考察了在确定哪个 AWS 容器服务最适合面向微服务的应用程序方面具有决定作用的因素。最后，阐述了容器监控和可观察性策略的重要性。

Splunk和 AWS: 转变容器化应用程序的实时企业监控

Splunk 可观察性解决方案包括 SignaFx 基础设施监控、SignaFx 微服务 APM 和 Splunk 云。Splunk 解决方案可在您的 Amazon Web Services (AWS) 环境中为您提供:

- 对您的整个环境进行实时 (1 秒) 度量和监控，包括基础设施、应用程序、微服务、容器和 AWS Lambda
- 见解，帮助您优化 AWS 环境的成本和容量
- 与 AWS 服务直接集成，包括预构建的仪表板
- 整合了对整个组织中每个人的可见性和监控

Splunk 使开发运维和 SRE 团队能够更轻松地监控和管理容器环境，并优化内部或云中的资源和成本。在 AWS 上同时使用容器和功能 (无服务器) 的客户可以为其整个环境提供端到端的实时监控解决方案。除了监控之外，SignalFx 基础设施监控还提供了成本优化器工具，可以直接了解保留的实例和按需 Amazon EC2 实例的成本和资源分配情况。

Splunk 一直在帮助**拥有复杂容器环境的客户**，包括在 Amazon ECS 管理的 Amazon EC2 实例上的 Docker 中运行的应用程序、有状态应用程序 (例如在 EC2 实例上的 Docker 中运行的不带调度程序的数据库) 、在 ECS 中运行的无状态应用程序，以及在 Mesosphere (现为 D2iQ) 上运行的 Apache Spark 等流式应用程序。除了 Amazon EKS 和 AWS Fargate，Splunk 还支持 Amazon EC2、Amazon ECS 以及 200 多个现成的云技术和应用程序集成。

容器基础知识

什么是容器

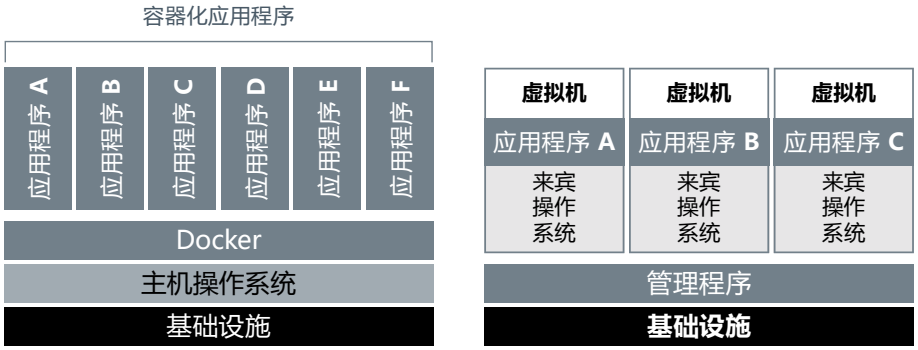
容器为开发软件提供了一种新的敏捷方法，使开发人员能够实现应用程序的可移植性，并能够在同一操作系统上运行多个应用程序，而无需共享依赖关系。容器提供了一种将应用程序的代码、配置和依赖项打包成单个对象的标准方法。如图所示，每个容器共享主机操作系统内核及其二进制文件和库，使它们非常轻 (兆字节) 。借助该抽象层次，容器可以在几秒钟内上下旋转，释放运行不可变基础设施的能力。

格式和开放标准

2013 年，Docker 启动了一个开源容器运行时项目，从而开始了容器采用的蔓延。从那以后，围绕这项技术出现了许多开放标准。有些是互补的，有些是竞争的，但有几个已经成为精英。以下是一些更受欢迎的开放标准:

- **OCI** (开放容器计划) 由 Linux 基金会管理，OCI 标准得到许多供应商的支持，并管理映像和运行时规范。
- **CNI** (容器网络接口) CNCF (云原生计算基金会) 项目，包括用于编写插件的规范和库，以便在 Linux 容器中配置网络接口。
- **Kubernetes CRI** (容器运行时计划) 虽然 Docker 是最受欢迎的容器运行时，但该领域仍在不断发展。CRI 使开发人员能够使用各种各样的容器运行时。

在企业规模上运行容器需要决定使用哪个容器运行时，以及应该采用哪个容器引擎、编排器、存储和网络解决方案。



容器化与虚拟化应用程序堆栈

云中的容器

如果您使用云基础设施来支持您的企业应用程序，AWS 云可以提供针对运行容器而优化的基础设施资源，以及一组编排服务，使您能够轻松地在生产环境中构建和运行容器化应用程序。这些服务允许根据您的需求存储、管理和运行容器，从小型实验到关键的生产应用。AWS 支持**符合 OCI 标准的容器**，包括 **Docker** 容器。

AWS 容器管理工具

容器管理工具可以分为三类，如下图所示：注册、计算和编排。通过 AWS 提供的服务，您可以安全存储和管理容器映像、编排（用于管理容器何时何地运行），以及为您的容器提供支持的灵活计算引擎。AWS 可以帮助您管理容器及其部署，因此您不必担心底层基础设施。

注册表

Amazon Elastic Container 注册表

高度可用和安全的容器注册表，
使其易于存储和管理容器图像。

了解更多信息 »

编排

Amazon Elastic Container 服务

完全托管的容器编排，与其他 AWS 服务无缝集成。

了解更多信息 »

Amazon Elastic Kubernetes 服务

使用 Kubernetes 轻松部署、
编排和扩展容器化应用程序。

了解更多信息 »

计算

AWS Fargate

无服务器计算引擎，可运行任何规模的生产容器。

了解更多信息 »

Amazon EC2

在虚拟机基础设施上运行容器，
完全控制配置和扩展。

了解更多信息 »

AWS 容器服务注意事项

容器编排器处理容器化工作负载的部署、维护和扩展，帮助公司操作容器。Docker 推出两年后，Kubernetes (K8s) 在社区中掀起了一场风暴，成为了占主导地位的容器编排技术。Kubernetes 从 Google 中剥离而出，由工程师构建，他们的任务是管理每周上下旋转的 20 多亿个容器。Kubernetes 是目前最流行的容器编排技术。在 CNCF 的一项调查中，78% 的受访者表示他们在生产中使用 Kubernetes。

AWS 云可以提供针对运行容器而优化的基础设施资源，以及一组编排服务，使您能够轻松地在生产环境中构建和运行容器化应用程序。这包括围绕 Kubernetes 提供的许多托管服务。例如，Amazon EKS 是 Kubernetes 编排服务的 AWS 版本，它跨多个 AWS 可用性区域为您运行 Kubernetes 管理基础设施，以消除单点故障。Amazon EKS 是经过认证的 Kubernetes，所以您可以使用合作伙伴和 Kubernetes 社区现有的工具和插件。

关于哪种用例从每种类型的 AWS 容器服务中受益的 AWS 建议：

SignalFx 为 AWS 容器服务提供内置监控

If you want to...	Consider using	Because...
Store, encrypt, and manage container images	Amazon ECR	ECR compresses and encrypts your container images, making them fast to start and available to run anywhere.
Run containerized applications or build microservices	Amazon ECS	ECS eliminates the need for you to install and operate your own container orchestration software or manage and scale a cluster of virtual machines.
Manage containers with Kubernetes	Amazon EKS	EKS provisions and manages a conformant, upstream Kubernetes control plane and is integrated with many AWS services to provide scalability and security for your applications.
Run containers without managing servers	AWS Fargate	Fargate scales and manages the infrastructure required to run your containers. This removes the need to choose server types, decide when to scale your clusters, or optimize cluster packing.
Run containers with server-level control	Amazon EC2	EC2 virtual machines gives you control of your server clusters and provide a broad range of customization options.

SignalFx 内置监控

当选择 AWS 提供的一个或多个容器服务和工具时，除了您的主要用例之外，您需要记住以下考虑事项：

- 用例 (应用程序、微服务架构、混合云或多云需求)
- 需要由用户管理的实体
- 集群配置灵活性要求
- 集群管理需求
- Kubernetes 环境中工作节点的控制
- 对云服务器 instance7.Docker 控制的控制能力
- 容器编排工具的自由选择
- 可观察性需求、实时应用程序性能监控需求和监控规模
- 定价注意事项

下表提供了每个 AWS 容器服务的优势和推荐使用的详细信息。它还强调了为服务提供最佳服务的不同容器使用场景的关键指标和注意事项。基于您的用例和对基础设施和容器组件的控制水平，您可以选择这些服务。

容器服务	优点	推荐使用	CloudWatch Metrics
Amazon ECS	• 比用于管理容器的 Kubernetes 接口简单得多 • 非常类似于基于服务器的工作负载，因此更容易迁移到基于云的容器 • 与 AWS 服务完美集成，并通过 AWS 服务组件（例如应用程序负载平衡器 (ALB)、Route 53 和 CloudWatch) 提供网络和支持组件	• 对于那些开发运维资源有限的组织来说，使用 ECS 时的学习曲线要小得多，这些组织不希望围绕容器中的 Pod 概念来重新构建应用程序 • 如果您正在利用 “软件定义的基础设施”，例如 Terraform、Elastic Beanstalk，您通常会发现 ECS 在这些应用中会得到很好的支持 • 如果您需要在任务级别控制扩展，那么您需要使用 ECS，否则 Fargate 会将容器和服务器级别的扩展抽象为一个更简单的任务扩展模型 • AWS 控制和管理集群，允许客户进行一些集群配置，如果您的应用程序需要对 Docker 进行完全控制，这将很有帮助；然而，容器编排是 AWS 专有的，使用开源调度器 • 定价：您需要为 EC2 实例、EBS 卷以及用于部署和设置应用程序服务器的任何附加 AWS 服务付费 — ECS 不收取额外费用，但如果您使用 ECS Fargate 选项，您需要为容器化应用程序请求的 vCPU 和内存资源付费	• 容器、磁盘和网络的 CPU 和内存预留和利用率指标 • Amazon ECS 集群和 ECS 服务，运行任务计数

虽然大多数托管 Kubernetes 的供应商正在发布其容器解决方案的混合版本，但多云支持仍然微不足道。这促使许多用户继续采用 Kubernetes 作为他们自己管理的独立环境。如果您的应用程序需要对容器放置和流量路由决策进行更多控制，您可以选择 **Amazon ECS** 或基于 **Amazon EC2** 的容器环境。

容器服务	优点	推荐使用	CloudWatch Metrics
Amazon EC2	• 成熟、高度可用的 AWS 云服务器实例，用于托管您的托管容器应用程序环境 • 根据您的应用程序需求灵活选择容器管理软件 • 对云服务器实例和运行在其上的应用程序的最高级别控制，使用户能够管理服务器集群的配置、修补和扩展 — 您可以决定使用哪种类型的服务器、在集群中运行哪些应用程序和多少个容器来优化利用率，以及何时应该在集群中添加或删除服务器	• 如果您的应用程序需要非常健壮、可扩展和严格控制的混合容器环境，无论是在云中还是在内部，在 EC2 服务器实例上使用 Kubernetes 都是一个不错的选择 • 您可以完全自由地选择您喜欢的编排工具，例如 Kubernetes，或者，如果您需要简单的容器管理并希望自己管理它，可以选择 Docker Swarm • 用户可以完全控制服务器集群，并为您的容器环境提供更广泛的定制选项，这可能是支持某些特定应用程序或法规遵从性和政府要求所必需的 • 使用自我管理的 EC2 集群部署容器更适合应用程序的可移植性 • 定价：您需要为 EC2 实例、EBS 卷以及用于部署和设置应用服务器的任何附加 AWS 服务付费，例如存储、网络、LBS、消息传递等	• EC2 实例 – 集群、服务和任务级别的 CPU 和内存利用率、内存预留 • 可用性和 AWS 区域指标 • 磁盘、网络和端口监控指标 • 可扩展性指标

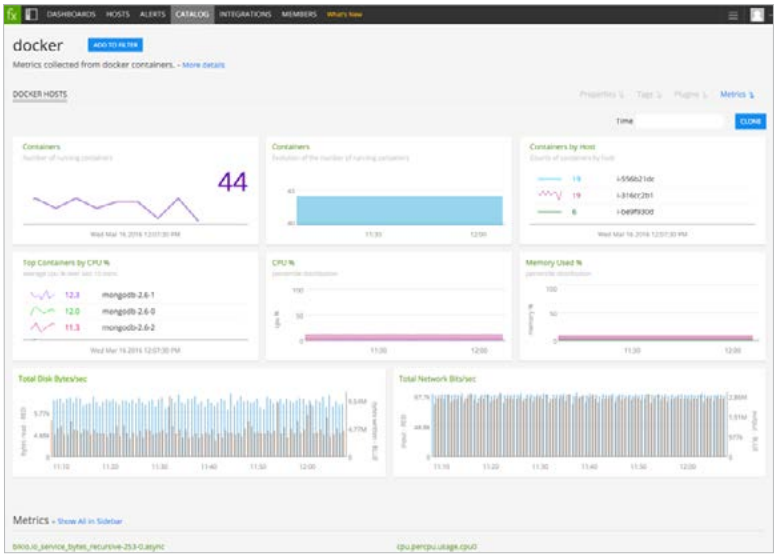
容器服务	优点	推荐使用	CloudWatch Metrics
Amazon EKS	• 与 Amazon 服务紧密集成 • Amazon 管理您的 Kubernetes 集群控制平面 • 与 IAM、VPC、汽车扩展模型集成	• 如果您已经在为 Kubernetes 使用内部或云提供的主机，或者正在寻求利用事实上的行业标准来进行容器的开源编排，那么 EKS 可以提供 Kubernetes 的许多好处，而无需承担托管和配置 Kubernetes 环境的操作责任 • 如果您正在利用 AWS，但希望让您的基础设施便于其他云提供商使用，或者需要多云基础设施 • 您拥有必备的开发运维技能，您应用程序需要控制容器的管理方式 • 定价：除了 EC2、EBS 和您的应用程序使用的其他 AWS 服务 (容器除外) 的成本之外，还有额外的 EKS 控制平面成本	• AWS CloudWatch 有助于监控 Kubernetes 控制平面日志，包括：审计、API 服务器、验证器、控制器管理器和调度器日志 • 容器、Pod、节点、网络、集群、命名空间和 AWS 服务的状态

容器服务	优点	推荐使用	CloudWatch Metrics
AWS Fargate	• 现代应用程序的无服务器容器化环境，让开发人员专注于编写代码，而不用担心容器配置和扩展 • Fargate 是一个完全托管的解决方案，用户只需设置他们的应用程序任务，并将其部署在 AWS 管理的 ECS 实例上，这允许用户运行容器化的应用程序，而无需管理服务器或集群 • 使用 Fargate，用户只需定义他们的任务或服务操作所需的 CPU 和内存量，并且只为 CPU 和内存时间付费，而不是 EC2 实例	• Fargate 推荐用于容器化的应用程序，这些应用程序在使用 EC2 或 ECS 时自动高效地扩展很复杂，并且最终通常会有大量未使用的 CPU 和内存资源 • 对于响应事件、按需或按计划运行且不需要专用 EC2 服务器实例 (无论是直接使用还是作为 ECS 选项的一部分) 的任务，这是一个不错的选择 • 最适合小型微服务、不需要对集群或服务器或容器编排进行任何控制的按需作业以及非常有限的 Docker 控制 • 定价：您只需为每个任务定义的资源付费 — 定价是根据从您开始下载容器映像 (Docker pull) 到 AWS 任务终止 (四舍五入到最近的一秒钟，最低收费为 1 分钟) 这段时间内使用的 vCPU 和内存资源计算的	• CloudWatch Log Streams for Fargate • 按任务、容器和服务分类的 Fargate 指标 **(截至 2019 年 7 月的预览模式)**

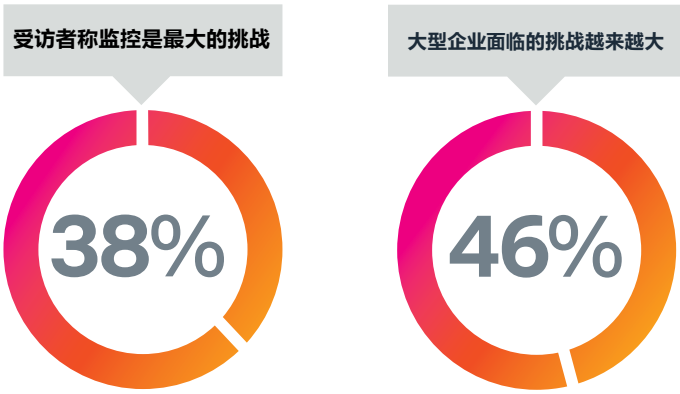
容器服务	优点	推荐使用	CloudWatch Metrics
Amazon ECR	• Amazon ECR 使用 AWS IAM 支持具有基于资源的权限的私有 Docker 存储库，以便特定用户或 Amazon EC2 实例可以访问存储库和图像 • 与 Amazon ECS 和 Docker CLI 紧密集成，使您能够简化开发和生产工作流程	• 开发人员可以使用 Docker CLI 来推、拉和管理基于容器的应用程序部署映像 • 如果您的应用程序基础设施完全基于 AWS，使用 ECR 会有益处，因为它不需要操作您自己的容器存储库，也无需担心扩展底层基础设施。	• 针对添加到存储库中的新图像，监控 “放置图像” 通知事件

容器监控和可观察性策略

选择正确的容器监控和可观察性策略的困难程度，不亚于选择正确的运行时和编排技术时的困难程度。这是让您的容器架构为生产做好准备的最关键的组件之一。如果您在没有一个可靠的监控和可观察性策略的情况下将新容器推向生产，您将会出现问题。



根据 CNCF 对近 2400 名 CNCF 成员的调查，对于采用新的云原生架构的开发运维团队而言，他们最关心的问题是监控和可观察性。



调查结果：容器监控是最大的挑战

来源：https://www.cncf.io/blog/2018/08/29/cncf-survey-use-of-cloud-native-technologies-in-production-has-grown-over-200-percent/

当计划和执行容器监控和可观察性策略时，您需要考虑以下几点。

考虑事项 1: 集成

有许多容器技术可供选择，就像有许多云原生技术一样。了解您的容器技术在更广泛的云原生堆栈中的集成程度至关重要。例如，如果您的 **Elasticsearch** 数据位于监控系统 A 中，而您的容器遥测数据位于监控系统 B 中，那么成功的可能性不大。为了确保您拥有做出有效决策所需的所有数据，您需要一个监控平台，该平台具有捕获和关联数据所需的所有集成，并具有可观察性的单一真实来源。当谈到集成时，您需要考虑健壮性、广度和覆盖范围。

推荐策略

选择高度集成到云原生生态系统的监控供应商，例如 SignalFx 与 AWS 的结合。

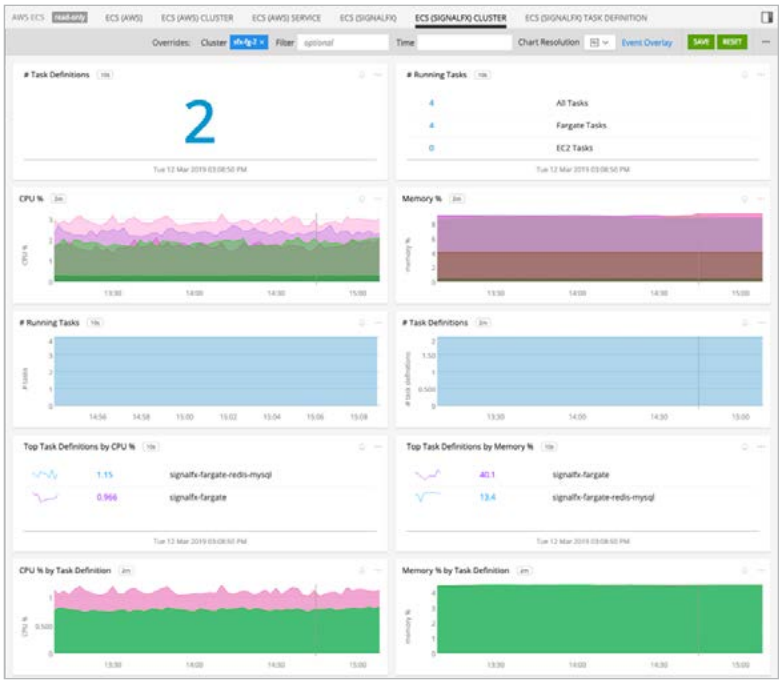
考虑事项 2: 发现和摄取

由于容器的寿命非常短 (几秒/几分钟) ，因此在容器上下旋转时发现新容器并将其与正确的主机 (EC2 等) 和微服务相关联以实现更快的 MTTR 非常重要。容器是轻量级的，因此在它们上面运行传统监控供应商使用的重量级代理是没有意义的。

推荐策略

选择使用轻量级代理自动发现新容器和服务的供应商。借助 SignalFx 基础设施监控，您可以看到详细的指标，这些指标可以使用 **SignalFx AWS 容器资源监控仪表板**提供实时主动的故障排除见解。选择基于开放标准的解决方案，而不是利用其重量级和/或专有代理的解决方案。这减轻了对与供应商捆绑在一起的担忧，并获得了生态系统友好的好处。

下面的 SignalFx Amazon ECS 监控仪表板显示了与 ECS 任务相关的性能指标:



SignalFx Amazon ECS 监控仪表板

考虑事项 3: 扩展

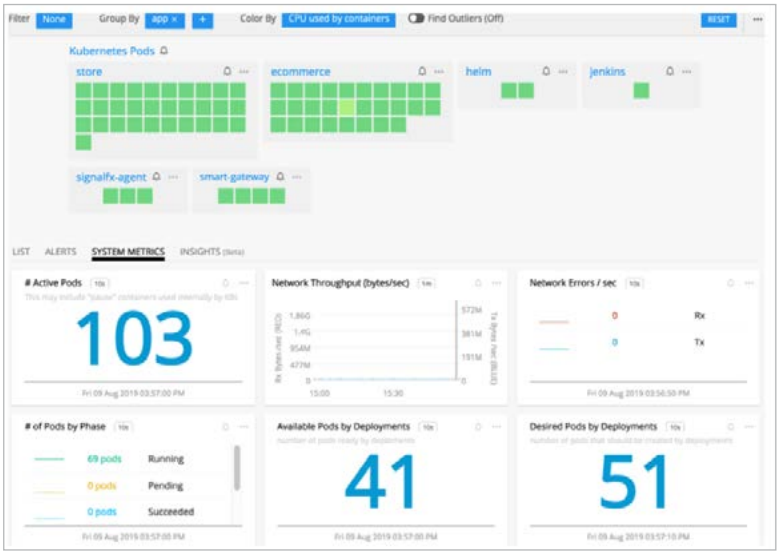
微服务和容器的激增导致了需要分析的数据点激增。当前趋势将继续增加数据点的数量: 装箱是增加每台主机的容器数量的常见策略, 这意味着每台主机有更多的数据。团队这样做是为了帮助节省云计算成本。此外, 计算实例的适当规模意味着, 实例比传统的大型主机多得多。

细粒度的可观察性用例 (例如基于每个客户或类别的监控/警报) 会导致传统监控工具的性能下降。高数据量阻碍了重要的用例 (例如, 在一年内搜索 1000 个容器相当于 310 亿个数据点) 、容量规划, 甚至减缓了停机期间的根本原因分析。

推荐策略

构建自己的解决方案来处理扩展带来了严重的权衡考虑，最终可能不值得感觉为难。请记住，随着您将来继续扩展容器，数据量将继续快速增长。自主开发的系统不仅要能处理今天的数据量，还要能在未来继续保持良好的性能。这将需要大量的投资、护理和馈送。您不仅需要扩展数据存储，这样就不需要在多个孤立的数据库中分割数据，而且查询必须继续以更高的数据量快速响应。最后，数据突发意味着，峰值负载比正常情况高得多，在非高峰时段为峰值容量调配资源会浪费大量资金。

有商业选择。当许多监控团队使用 K8s 和容器时，通常会选择开源的时间序列数据库 (TSDB)。然而，在某个时候，您会感受到上面讨论的问题的痛苦。专注于您的核心业务并简单地利用托管监控解决方案 (例如 SignalFx 基础设施监控) 可能会更好地利用您的时间、开发人员和资金，该解决方案是在考虑容器的情况下构建的，如下所示。



用于显示容器环境系统指标的 SignalFx 仪表板

考虑事项 4: 延迟

传统的批处理监控系统运行完整的查询，并在每次评估警报时加载完整的数据集，这种系统不能满足现代需求。这些系统不能每隔几分钟刷新查询一次以上，这意味着最终用户不能以精细的粒度跟踪应用程序。由于容器的短暂性 (寿命短)，它们还会错过重要的警报和分析。

推荐策略

如果实施得当，流式分析技术在监控方面优于批处理方法。流式分析不仅可以支持更大的数据量和并发用户/查询，还可以实时触发更新和警报。我们建议您构建或购买数据流监控解决方案，这样您就可以在几秒钟内对容器数据发出警报并进行分析。这使得能够更快地对问题采取行动，甚至通过网络挂钩触发警报，以执行自动回滚和其他操作，从而在机器运行时修复问题。

考虑事项 5: 变动

变动是最被低估的，也可能是监控中最痛苦的问题。从技术的角度来看，当一个数据源被另一个等价的数据源替换时，就会发生变动。容器的这种变动造成了元数据的爆炸，因为数据源的身份不断变化。导致这种变动的一些用例是 CI/CD 部署 (一些组织将其整个环境蓝绿色化) 、自动扩展、云中的短期点实例以及影响许多指标的大规模标记 (例如，向所有指标添加一个像"customer_id"这样的新标记) 。随着时间的推移，变动不仅会导致元数据的累积，有时还会突发发生 (例如，在对整个应用程序或服务进行蓝绿色化时) 。许多监控系统无法处理如此突然的大量元数据，也无法足够快地对其进行处理或索引。

推荐策略

变动对您的监控系统来说可能很危险，并且随着时间的推移，影响可能会越来越严重。如果您每天都在推送代码，那么一年后您将不得不处理 365 倍以上的元数据，而容器则更具挑战性。

如果您已经构建或计划构建一个系统，以下是您需要考虑的一些事情：为处理突发事件提供容量的计划，或者减缓代码推送以平滑突发事件的计划。如果历史上对数据的查询对您来说是一个重要的用例，您可能最终会在每个服务或应用程序的基础上，实现某种"预聚合"或"预计算"方案来处理这个问题。然而，您需要在警报的及时性和准确性之间进行权衡。大多数开源工具不会让您两者兼得。

变动问题也广泛存在于当今的监控供应商中。它们在小型概念验证 (POC) 中可能很有效，但是性能会随着时间和环境规模而降低。选择一个在构建系统时考虑到变动问题的供应商。

考虑事项 6: 可见性

在当今分布式、容器化的环境中，很难找到问题的根本原因。有一个解决方案可以帮助您可视化堆栈中的数据，这对于在突发事件时从噪音中提取信号非常重要。

推荐策略

利用 Grafana 进行一般可视化，或利用 Sysdig 深入到容器可视化情况，再深入到内核级别，都是一些选项，但它们只能解决可观察性难题的一部分。注意不要堆积太多的点工具。相反，考虑尽可能整合您的工具以节省成本。您可能希望选择一个端到端的可观察性平台，而不是拼凑出能够覆盖整个堆栈宽度的点工具，以及用于快速实现价值的预构建仪表板。

考虑事项 7: 故障排除

将您的基础设施与该基础设施所服务的应用程序相关联非常重要。随着一些公司采用容器，他们这样做的心态是将他们的整体应用程序分成微服务，或者为他们的新应用程序重新开始分布式服务架构。这使得在事务数据遍历您的容器支持的这些服务时，也能够捕获事务数据变得至关重要。

推荐策略

利用基于尾部的分布式跟踪解决方案，该解决方案观察所有穿过应用程序层的事务，并存储 p90 和 p99 跟踪以进行历史分析。此外，确保您的解决方案将帮助您的团队以有指导的方式解决问题。当今系统的分布式特性在您的服务之间创建了相互依赖关系，这对于观察是至关重要的，并且需要能够梳理出，在您的服务之间的通信中，大部分延迟发生在何处。您还需要快速回答底层基础设施是否导致延迟或错误增加，因此您需要一个能将基础设施指标和应用程序跟踪联系在一起的解决方案。

Splunk 是您的容器可观察性战略的核心

SignalFx 从单一管理平台为 AWS 基础设施、Kubernetes 平台、App Mesh 数据平面 (Envoy、Docker 容器和微服务) 提供统一监控。无论您是在 Amazon EKS、ECS 上部署 App Mesh，还是在 EC2 上部署 Kubernetes，SignalFx 都可以提供全面的全堆栈监控。

SignalFx 基础设施监控提供了与 CloudWatch 的强大集成，为**基础设施导航器**提供了 CloudWatch 支持的模式，并包括许多**内置仪表板**，帮助您开始监控 Amazon Web Services。您还可以使用 **SignalFx 智能代理**监控 Amazon Web Services 实例及其上面运行的服务。SignalFx 智能代理提供了比 AWS CloudWatch 更高程度的定制，并且可能更适合您希望以更好的分辨率查看指标，或者对发送的指标进行详细控制很重要的情况。**SignalFx 智能代理**是一个在 Go 中编写的指标代理，用于在各种不同的环境中监控基础设施和应用程序服务。它安装了 100 多个用于收集数据的捆绑监视器，包括基于 Python 的插件，例如 Mongo、Redis 和 Docker。SignalFx 智能代理最初作为 Kubernetes 集成进行开发，现在从容器和非容器环境中收集主机性能、应用程序和服务级别指标。

有关 SignalFx 监控的 AWS 服务和基础设施的完整列表，请参见：https://docs.signalfx.com/en/latest/integrations/amazon-web-services.html

正如您将在接下来几页的表格中看到的那样，SignalFx 技术具有独特的优势，可以满足云原生堆栈的新监控标准，并帮助您集中控制可观察性策略。

考虑事项	SignalFx	开源	传统监控解决方案
集成	数百个集成到最流行的云原生技术 (包括 Prometheus)	大多数开源工具都缺乏预构建的集成，除了 Prometheus，它拥有一些集成	数百个集成到最流行的云原生技术
发现和摄取	Docker、Kubernetes 和其他流行的云原生技术的智能代理自动发现和配置	大多数开源工具缺乏自动发现和自动配置功能，但是 Prometheus 确实提供了一些有限的自动发现功能	容器和 Kubernetes 的专有代理和有限代理自动发现和配置
扩展	流式架构以高分辨率无缝处理 100000 个组件	开源 TSDB 不是为水平可扩展性而构建的，大约有 5000 个组件性能下降	批处理架构将可扩展性限制在 10000 个组件以下
延迟	当新容器向上或向下旋转时，流式架构可以触发警报，在几秒钟内更新图表	• 所有开源工具都有基于拉的 (批处理) 体系结构，每隔几分钟就轮询数据以更新图表和触发警报，这将错过来自短暂容器的关键数据，这些容器动态地上下旋转数据集	批处理架构限制了警报的及时性，造成了更新图表的延迟 (大约几分钟)，并导致查询需要几分钟到几小时才能完成，限制了有效监控容器的能力 - 限制了关键用例
变动	SignalFx 有一个专门构建的元数据存储 (独立于 TSDB)，用于处理大量元数据和与高容器变动相关的高基数	开源 TSDB 只有一个用于时间序列和元数据的数据库，并且通常会限制您可以应用于指标的标签数量 (< 1000 个)	由于整体性能下降，不鼓励在指标上使用超过 1000 个标签

续 ▼

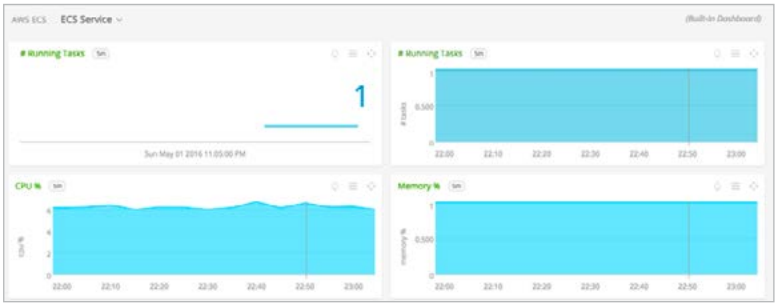
正如您将在接下来几页的表格中看到的那样，SignalFx 技术具有独特的优势，可以满足云原生堆栈的新监控标准，并帮助您集中控制可观察性策略。

考虑事项	SignalFx	开源	传统监控解决方案
可见性	• 全堆栈可观察性，本机支持指标、分布式跟踪，并将功能深度链接到同类最佳的日志解决方案 • 镜像仪表板功能减少了仪表板的蔓延，并在团队和组织中推广了最佳实践内容	• 对容器指标的有限可见性阻碍了完全的可观察性 • 针对指标、跟踪和日志的单点解决方案缺乏全套功能，需要付出巨大努力才能单独配置和集成在一起	• 点解决方案和轻度集成套件的组合，可跨指标、跟踪和日志实现可见性 • 通过随机采样对分布式跟踪的有限可见性阻碍了完全的可观察性
故障排除	• Outlier Analyzer™ 使用数据科学来快速分析相关跟踪，以确定导致高延迟的因素，从而显著降低 MTTR • 紧密集成的指标和分布式跟踪平台支持上下文关联和快速问题隔离，通过跟踪/跨度矩阵化 (RED+) 实现唯一性识别、历史基线化以及计算聚合和百分比	• Zipkin 和 Jaeger 等开源跟踪工具缺乏用户友好的强大 UI。它们经常强制进行逐迹分析，这很费时，而且会降低 MTTR • 事务的随机抽样 (将数据量减少到可管理的水平) 会遗漏关键的异常值和异常痕迹，妨碍了对 P99 问题的理解和缓解 • 仅使用 APM 的孤立工具 – 用户无法将事务性能与基础设施运行状况联系起来	• 缺乏内置的分析和手动的逐迹分析，导致 MTTR 降低 • 随机抽样会遗漏关键的异常值和异常痕迹，妨碍了对 P99 问题的理解和缓解

利用 SignalFx 基础设施监控来监控 AWS 容器服务

在选择任何 AWS 容器服务时，监控和可观察性是关键考虑因素之一。容器化环境具有高度的动态性和短暂性，这导致使用传统的基础设施监控或 APM 解决方案很难对其进行监控。每天甚至每小时都有数百个 (如果不是数千个) 组件上下旋转，基于批处理的监控解决方案无法跟上这种变化，在许多情况下甚至无法看到新组件，更不用说实现多维实时分析。

SignalFx 流分析引擎 SignalFlow 是唯一一个能够跟上容器化环境的动态性而不影响性能和被大量警报淹没的引擎。我们的一些客户正在运行行业中要求最苛刻的基于容器的生产环境，每天都有数百万个组件发生变动。

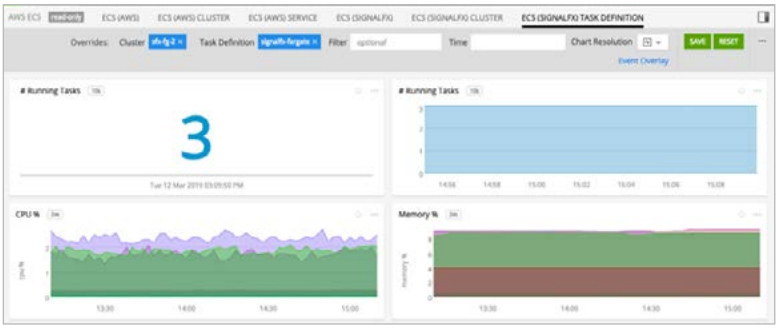


指标监控 — 关注使用 CloudWatch 的单一 ECS 服务

创建适合您特定容器环境的监控计划

除了前面列出的容器注意事项之外，我们还建议您创建一个监控计划，解决与您的容器环境相关的具体问题，例如：

- 监控目标
- 需要监控的资源类型和服务列表
- 监控频率
- 工具和集成要求
- 监控角色与职责
- 通知和处理警报的流程以及自动化水平



指标监控 — 使用 SignalFx 智能代理关注单个 ECS 任务定义

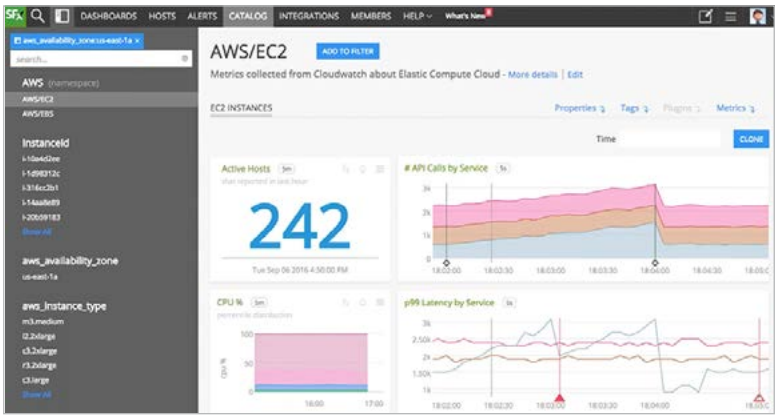
SignalFx 帮助您实时密切监控 AWS 容器服务和资源。通过我们与 **AWS 的现成集成**，使用 SignalFx 监控Amazon ECS。为了更深入地了解您的 ECS 环境，SignalFx 的**智能代理**可以自动发现服务，并针对您在 ECS 中运行的容器，提供更深入的指标。

SignalFx 可以帮助您更好地监控和管理您的容器环境。它弥合了您需要监控的内容和 CloudWatch 提供的内容之间的差距。此外，与 CloudWatch 的编程界面相反，它提供了一种更简单的方法来可视化您的监控数据。Amazon ECS 收集指标时，每分钟收集多个数据点。然后，在将数据发送到 CloudWatch 之前，将它们聚合到一个数据点。因此，使用 CloudWatch，一个样本计数实际上是一分钟内多个数据点的总和。SignalFx 允许您以一秒钟的精细频率监控您的容器指标。AWS 允许您访问两周的历史指标和日志。这些数据可用于建立基线和识别故障排除模式。

借助 SignalFx，您可以根据订阅计划获得更丰富、更细粒度的指标汇总、**分辨率和长达一年的数据保留期**。SignalFx AWS Optimizer 为您提供了可行见解来了解节省成本的机会和未充分利用的 EC2 投资。您可以通过 InstanceType、AWS 区域、AWS 可用性区域以及特定于您的设置的类别 (例如服务、团队或源自 EC2 实例标签的任何其他维度) 来查看使用模式和成本属性。

SignalFx 提供了与 CloudWatch 的强大集成，为基础设施导航器提供了 CloudWatch 支持的模式，并包括许多内置仪表板，帮助您开始监控 Amazon Web Services。您还可以使用 SignalFx 智能代理监控 Amazon Web Services 实例及其上面运行的服务。SignalFx 智能代理提供了比 AWS CloudWatch 更高程度的定制，并且可能更适合您希望以更好的分辨率查看指标，或者对发送的指标进行详细控制很重要的情况。

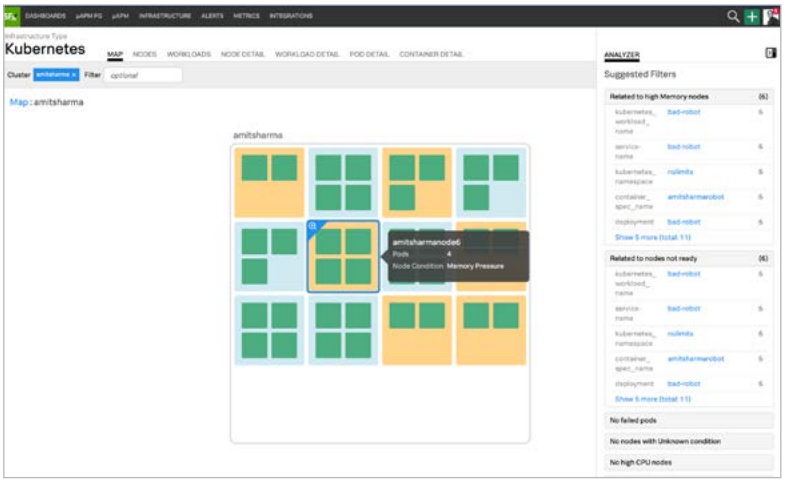
如果您想了解如何在 ECS 中部署 SignalFx 智能代理，请参阅 智能代理 ECS 部署指南 以获取指导。此外，智能代理可以部署在 ECS 任务中，以监控 AWS Fargate 容器。有关更详细的部署说明，请参阅 智能代理 Fargate 部署指南。



SignalFx AWS EC2 监控仪表板

Kubernetes Navigator

Kubernetes Navigator 包含在 SignalFx 基础设施监控中，是一款企业级交钥匙 Kubernetes 监控解决方案，提供了一种简单直观的方式来了解和管理 EKS 环境的性能。我们与数百位客户合作，设计了一个适用于每个团队的解决方案，不管他们的成熟度和 Kubernetes 的操作经验如何，均可使用。Kubernetes Navigator 为开始云原生之旅的团队带来了立竿见影的价值，同时也解决了世界上最复杂的 Kubernetes 大规模部署的监控挑战。



使用 Kubernetes Navigator，团队能够比以往更快地检测、分类和解决性能问题。开发运维和 SRE 团队可以利用以下特点，成功应对与大规模运营 Kubernetes 相关的复杂性：

- 动态集群地图：一种直观的方式来即时了解 Kubernetes 集群的运行状况
- 钻取：通过快速深入来快速有效地排除故障
- 在上下文中记录日志：深入链接到上下文日志，以获得精细的见解，消除上下文切换并加快根本原因分析
- Kubernetes 分析器：AI 驱动的分析加快故障排除

动态集群地图

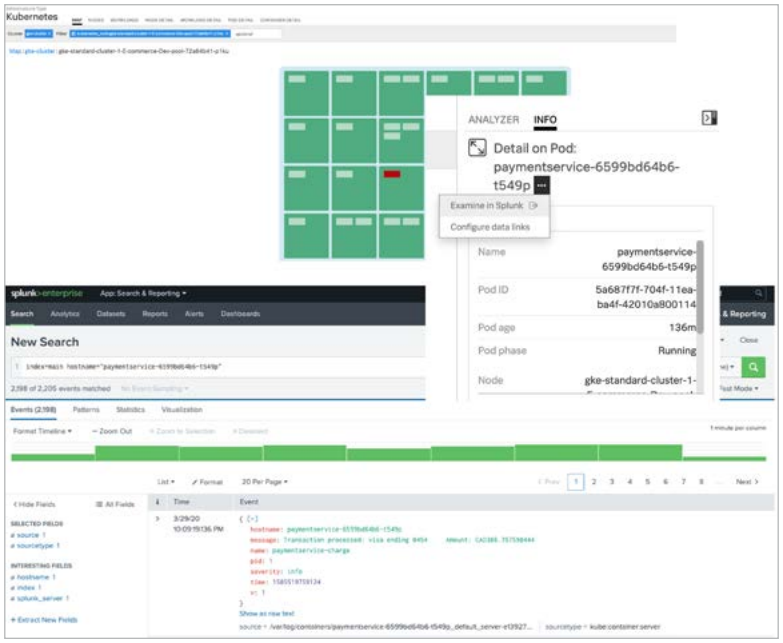
从鸟瞰图开始，Kubernetes Navigator 使团队能够通过直观的分层导航，快速了解整个 Kubernetes 环境的性能。在几秒钟内选择、过滤或搜索任何 Kubernetes 实体，例如节点、Pod 和容器级别。SignalFx 自动发现 Kubernetes 组件和容器化服务，以立即监控您的整个堆栈。了解动态 Kubernetes 组件之间的关系，并快速修复相互依赖的性能问题。

钻取

对整个 Kubernetes 环境的全局概览有助于团队了解整个系统的运行情况。同样重要的是，当团队缩小到问题的来源时，要对单个组件有一个细化、详细的视图 - 从节点向下钻取到 Pod，再到容器，最后到工作负载。我们的流式架构能够在几秒钟内进行大规模搜索和筛选，进而进行深入分析。

在上下文中记录日志

无缝地转向日志，并获得对应用程序、Kubernetes 和容器日志的精细可见性，以关联整个堆栈的性能，而无需任何上下文切换。对 Kubernetes 和 API 服务器审计日志的生命周期事件的可见性有助于您理解和维护您的安全性和合规性状态。



Kubernetes 分析器

为了理解性能异常背后的"原因"，Kubernetes Navigator 利用 AI 驱动的分析，自动提出见解和建议，以实时准确地回答是什么导致了整个 Kubernetes 集群的异常，包括节点、Pod、容器和工作负载。在下面的示例中，Kubernetes 分析器自动检测到一种模式，该模式会对一些 Kubernetes 节点造成内存压力。在这种情况下，具有无限内存限制的容器最终会消耗掉这些节点上的所有可用内存，耗尽 Kubernetes 调度的其他 Pod。这种情况通常被称为噪音邻居问题。使用建议的筛选器，SRE 团队可以在几分钟内缩小到潜在的问题。复杂的算法，包括历史性能基线和突然变化，可以检测系统级问题，例如 Goroutines 的突然增加或容器重启，并在几秒钟内发出警报。

总结

虽然新的云原生技术 (例如容器) 能够加快创新速度，实现更灵活、更有弹性的应用程序开发，但这些新技术也增加了监控和故障排除用例的复杂性。在您采用容器时，实施正确的监控、故障排除和整体可观察性策略至关重要。SignalFx 支持 AWS 容器基础设施和服务集成，使公司更容易管理其在 K8s 环境中的应用性能，并最大限度地提高大规模采用容器的好处。

客户越来越多地采用容器、编排和基于微服务架构的应用程序，以更快地交付创新成果，并使应用程序更具弹性。AWS 容器服务和 SignalFx 平台通过流式智能补充了现场可靠性工程师和运营团队的工作，以帮助在容器环境中检测问题和排除故障。AWS 和 SignalFx 还可以共同加速代码部署，并允许为应用程序自动捕获性能数据。SignalFx 解决方案为客户提供了预构建的仪表板，该仪表板具有性能指标和其他可视化功能，并具备系统范围的监控、可观察性和定向故障排除功能，这些都是在生产环境中大规模采用容器的关键要求。

借助企业信任的经验证的解决方案，让您的可观察性投资经得起未来考验，以实现大规模的最高级用例。立即注册免费试用 SignalFx。

要了解更多关于 SignalFx 基础设施监控和 SignalFx 微服务 APM 的信息，请访问 https://www.splunk.com/en_us/devops.html，或者联系销售部，获取更多信息。

20-splunk-AWS-container-monitoring-observability-WP-111