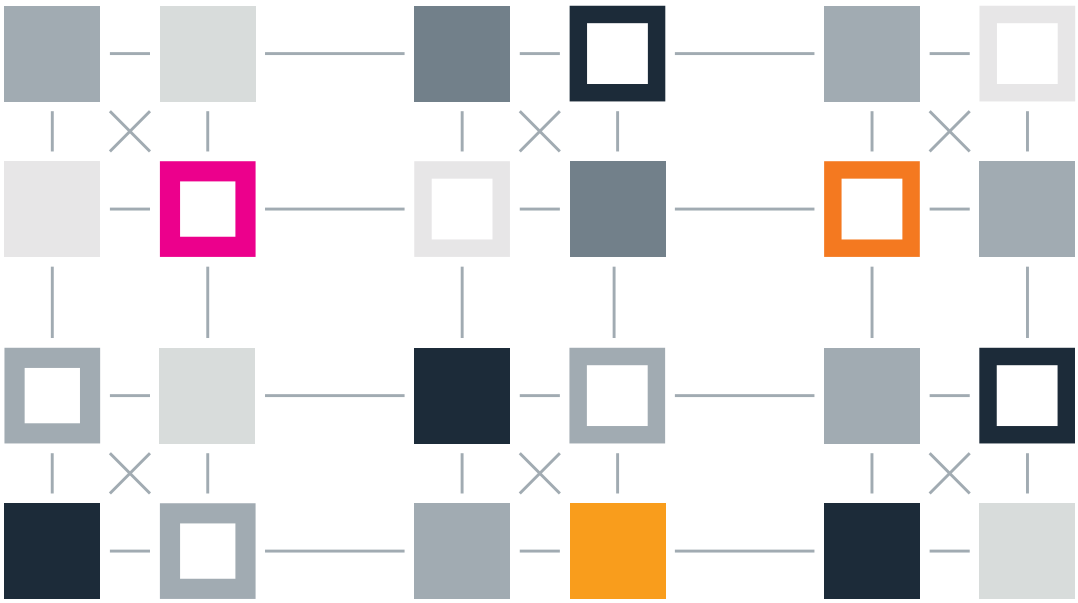
监控 Kubernetes 上的微服务



根据 CNCF 最新的两年一次的调查，对于采用 Kubernetes 的组织来说，监控是最大的挑战之一。

本白皮书深入探讨了如何最好地监控部署在Kubernetes 上的 Kubernetes 集群和微服务，以查明性能问题，最大限度地缩短平均解决时间 (MTTR)。

实时管理性能问题是实现完美终端用户体验的可控方式。企业需要开发运行 Kubernetes 集群的最佳实践，以及了解整个堆栈的性能，包括所有微服务的分布式跟踪。

容器化应用程序编排背后的大脑

Kubernetes是一个领先的容器编排解决方案，用于部署容器化应用程序。各行业的企业都在采用容器化微服务，以便获得：

- **速度** — 加快创新步伐
- **规模** — 跨不同地区和云提供商提供优化的应用体验
- **效率** — 通过优化资源消耗，以更低的成本提供更多的价值

Kubernetes 有几个逻辑层来支持各种用例，并管理容器化微服务的生命周期。

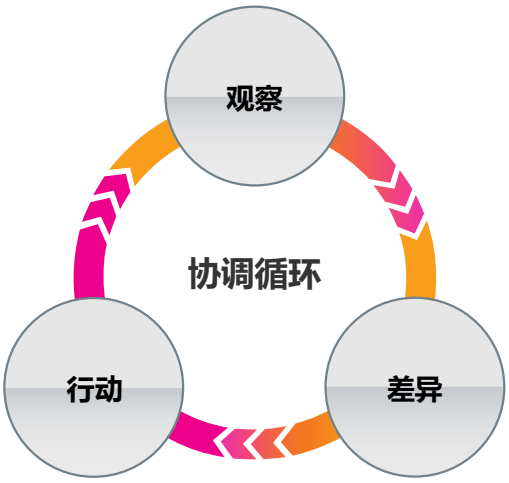
Kubernetes 部署哲学有三个基本支柱

基础设施抽象	声明性配置	不变
计算服务器、存储卷和网络等基础设施资源的应用程序编排	Kubernetes 续监控已部署的应用程序和 Kubernetes 对象的状态；它还实现了操作者所表达的系统的期望状态，例如一个服务应该运行多少个副本	继续 Docker 提出的不变性要求，容器图像是不可变的，Kubernetes 对象也是不可变的，例如，Kubernetes 服务的不同版本是完全独立的，不可互换

为什么使用 Kubernetes?

除了容器化给应用程序开发人员带来的好处之外，Kubernetes 还以多种不同的方式简化了操作：

自动化应用程序生命周期管理	云互操作性	自愈系统
Kubernetes 将应用程序生命周期管理自动化为端到端解决方案，包括应用程序配置、自动扩展、复制管理以及 CI/CD	Kubernetes 提供了对基础设施的抽象，因此应用程序可以真正跨不同的云进行移植	Kubernetes 保护应用程序免受任何破坏系统稳定的故障或不可靠行为的影响。它不断地采取行动来确保系统的状态符合期望的状态，如声明性配置中所表达的那样

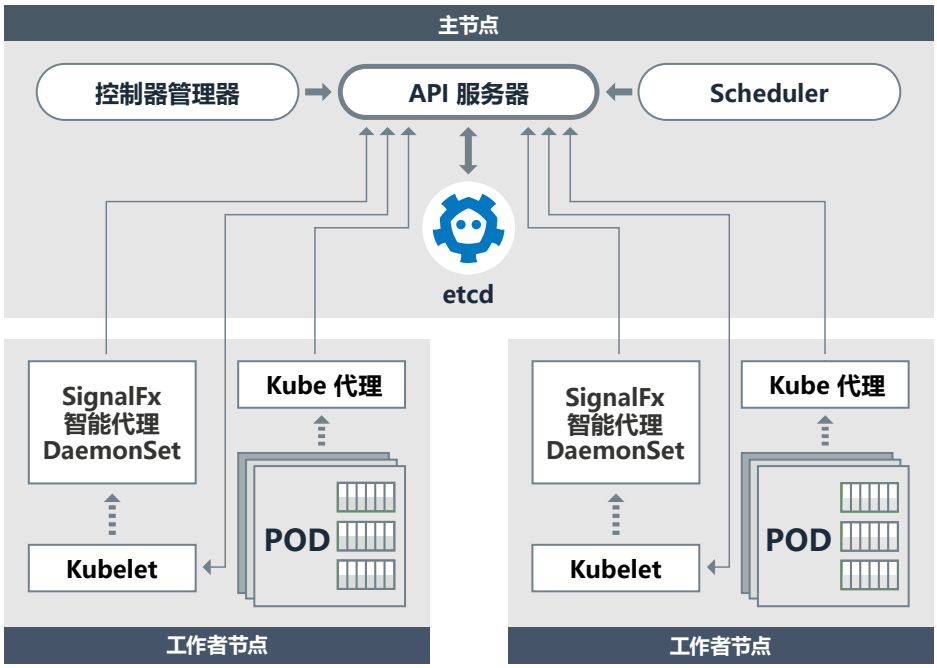


Kubernetes 控制器管理器不断地将期望状态与实际状态进行比较，评估它们之间的差异，并通过采取适当的措施来协调这些差异。

了解 Kubernetes 组件

Kubernetes 是一个将一组机器组装成单一单元的系统，可以通过 API 使用。Kubernetes 进一步将计算资源分成两个组:

工作者节点和主节点



以下是在主节点和工作者节点上运行的主要组件:

主节点

API 服务器

- 通往 Kubernetes 集群的门户
- 调解来自客户端和存储在 etcd 中的 API 对象的所有请求
- 执行身份验证和基于角色的访问控制 RBAC
- 请求验证和准入控制
- 更多信息：**http://bit.ly/k8s-apiserver**

控制管理器

- 实施 Kubernetes 内置控制循环的守护进程 — 滚动部署、副本集、工作者节点数等。
- 更多信息：**http://bit.ly/k8s-ctrl-mgr**

Scheduler

- 基于多种因素（相似性、可用资源、标签、QoS 等）来决定 Pod 应该在哪里运行。
- 更多信息： **http://bit.ly/k8s-scheduler**

etcd

- Kubernetes 集群的核心；保存所有 Kubernetes 对象的键值信息

更多信息：**http://bit.ly/2COkMx9**

工作者节点

Kubelet — 每个员工的代理

- 使用 PodSpec 启动 Pod（由一个或多个容器组成的组），并确保所有 Pod 运行正常
- 与容器交互 — 例如Docker
- 更多信息：**http://bit.ly/k8s-kubelet**

Kube 代理 — 每个员工的代理

- Kubernetes 服务的网络代理和负载平衡器
- 更多信息：**http://bit.ly/k8s-proxy**

这些是大多数应用程序所需的 Kubernetes 插件:

kube-dns

- 在 Kubernetes 上作为 Pod 和服务进行配置
- 在 Kubernetes 中，每个服务都会获得 DNS 条目
- kube-dns 解析集群中所有服务的 DNS

kubectl

- Kubernetes 的官方命令行
- 幕后使用基于 REST 的 对 Kubernetes API 服务器的 API 调用

关键 Kubernetes 构造和对象:

命名空间	Pod
单个集群的虚拟分割	一个或多个容器的逻辑分组，由 Kubernetes 管理
节点	**ReplicaSet**
Kubernetes 的基础设施结构（工人和主组件的主机）	确保给定数量的 Pod 运行的连续循环
角色	**Ingresses**
基于角色的 Kubernetes 集群访问控制	管理托管服务的外部 HTTP 流量
部署	**服务**
管理 ReplicaSet、Pod 定义、更新和其他概念	逻辑层，为动态 Pod 提供 IP、DNS 等持久性

Kubernetes 环境中的监控挑战

根据 CNCF 的最新调查，复杂性、监控和安全性是采用 Kubernetes 的组织面临的最大挑战。过去的监控策略在云原生时代不起作用，主要是因为:

- 要监控的组件有很多
- 容器是短暂和动态的
- Kubernetes 根据最佳资源利用率自动安排 Pod。虽然它确实提高了效率，但它也增加了对 Pod 部署和运行位置的不可预测性，除非使用 afÞnity 或 taints 来表达特定的意图

这导致 Kubernetes 环境会出现以下一组状况:

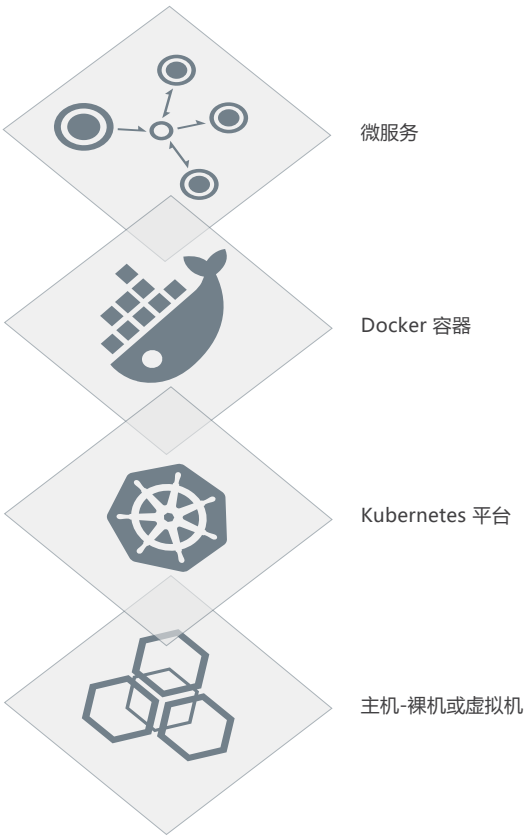
- 要监控的组件有很多
- 容器是短暂的，并且具有不可预测的 Pod 位置
- 容器化微服务的故障排除更加复杂

要监控的组件更多

在单芯片世界中，只有两个组件需要监控 - 应用程序和部署应用程序的主机。

在云原生世界中，由 Kubernetes 协调的容器化应用程序有多个组件需要监控:

- 主机
- Kubernetes 平台本身
- Docker 容器
- 容器化微服务



容器的短暂性和不可预测的放置位置

与传统的长时间运行的主机模型不同，现代基于微服务的应用程序通常部署在动态且短暂的容器上。Kubernetes 确保运行所需数量的应用程序副本。Kubernetes 会将 Pod 放在它认为合适的节点上，除非通过节点 afÞnity 或 taints 明确指示不要这样做。事实上，让 Kubernetes 调度 Pod 是这种自调整系统的关键设计目标。

传统的监控方法在这些高度动态的环境中不起作用，因为它们倾向于通过使用主机名或 IP 地址来跟踪长期运行的主机。对于容器化环境，监控工具必须提供即时的服务发现并自动检测容器的生命周期事件，同时在容器在几秒钟内启动或重启时调整度量收集。

监控微服务的性能

确定微服务环境中的问题比单一服务环境更具挑战性，因为请求在堆栈的不同层之间以及跨多个服务传递。现代监控工具必须监控这些相互关联的层，同时有效地关联应用程序和基础设施行为，以简化故障排除。

要监控的关键性能指标

POD 指标

监控所有 Kubernetes 对象非常重要，这样可以确保每个集群均运行正常，并且资源利用率得到优化。监控 Kubernetes Pod 指标以及部署和服务将有助于确定 Kubernetes 是否在您的环境中按预期工作。

POD	所需 Pod 数量
	可用 Pod 数量
	按阶段划分的 Pod（）失败、挂起、运行）
	每次部署所需的 Pod
	每个服务所需的 Pod
	按部署划分的可用 Pod
	按服务划分的可用 Pod
	ReplicaSet 所需的 Pod
	每个节点运行的 Pod

资源利用率指标

跟踪资源利用率也很重要，以确保您的应用程序和 Kubernetes 集群保持健康。

- Docker Socket 提供容器指标和节点级资源利用率指标，例如 CPU 和内存使用情况。Kubernetes 提供 collectd 指标以及由 Kubernetes 发出的指标。
- 将CPU、内存、磁盘 IO 和网络性能指标与应用程序性能和 Kubernetes 事件关联起来，有助于更快地找到性能问题的根本原因。
- 监控 Docker 和 Kubernetes 事件，例如容器或 Pod 生命周期事件，有助于查明错误配置或资源匮乏。

使用 SignalFx 监控 Kubernetes

Kubernetes 在集群、Pod 和服务级别提供了关于其组件和应用程序资源使用的详细信息，**这些信息可以通过**大量的监控解决方案轻松收集，但使这些数据可操作的任务留给了最终用户。

SignalFx 使您能够通过灵活、开放的仪器和预构建的仪表板来监控 Kubernetes，这些仪表板提供了对其环境各层的即时可见性。通过将流分析与分布式跟踪相结合，SignalFx 用户可以超越基本数据收集，利用实时警报和性能分析在几秒钟内发现并解决问题。

从云提供商收集指标

对于像 AWS Elastic Container Service for Kubernetes (EKS)、Google Container Engine (GKE) 和 Azure Kubernetes Service 这样的托管 Kubernetes 服务的基本监控，收集指标的最简单方法是将 SignalFx 与 AWS CloudWatch、Google Stackdriver 和 Azure Monitor 等服务相集成。

这种方法最为直接，它使 SignalFx 能够收集 Kubernetes 指标，而无需安装代理或修改应用程序代码。但是，默认情况下，这些服务被配置为以相对不频繁的间隔聚合和报告指标 () 默认情况下，AWS CloudWatch每 5 分钟更新一次) ，并且不提供对部署在 Kubernetes 集群上的特定服务的深入了解。

全自动 Kubernetes 监控

为了更深入地了解服务和更细粒度地监控容器指标，我们建议在您的 Kubernetes 集群中安装 SignalFx 智能代理。

智能代理是建立在 collectd 之上的开源指标收集代理，它为监控内容提供自动服务发现和配置。这种方法的一个优点是智能代理能够以 1 秒钟的分辨率向 SignalFx 提交指标标准，这使得它特别适合 Kubernetes Pod 的短暂特性。

智能代理在 Kubernetes 中作为 DaemonSet 运行，以确保它安装在集群中的每个节点上。它从 Kubernetes 收集数据，并使用 cAdvisor 从运行的 Docker 容器中获取资源和性能特征。自动发现 Kubernetes 组件和容器化服务的零接触配置可即时监控整个堆栈。通过一个简单的步骤便可以安装 SignalFx 智能代理:

```
helm install signalfx --set signalFxAccessToken=<
YOUR_ACCESS_TOKEN> --set
clusterName=<YOUR_CLUSTER_NAME> signalfx/
signalfx-agent
```

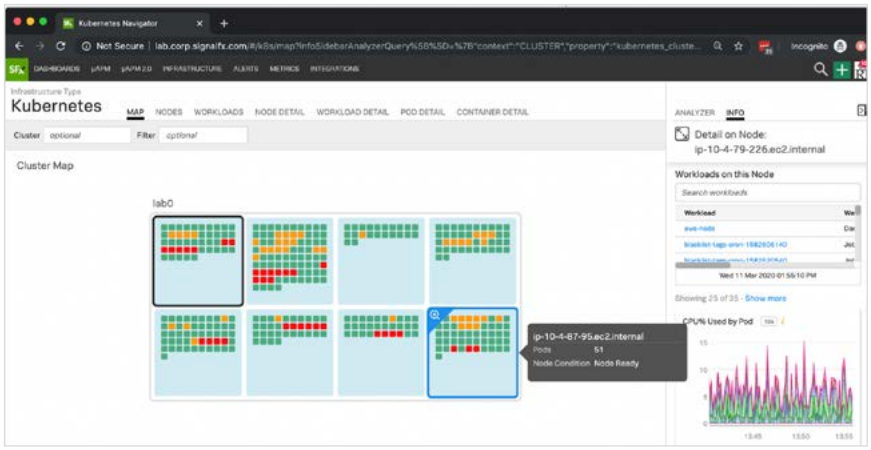
智能代理不仅可以监控上游 Kubernetes，还能够从托管 Kubernetes 服务 (例如 AWS Elastic Container Service for Kubernetes (EKS) 和 Google Container Engine (GKE)) 以及 Openshift 和 Pivotal Kubernetes Service (PKS) 等平台收集指标。

智能代理还提供了监视器，可以从 Prometheus 和 StatsD 等其他指标协议中收集数据，因此拥有现有指标管道的团队可以轻松利用 SignalFx。

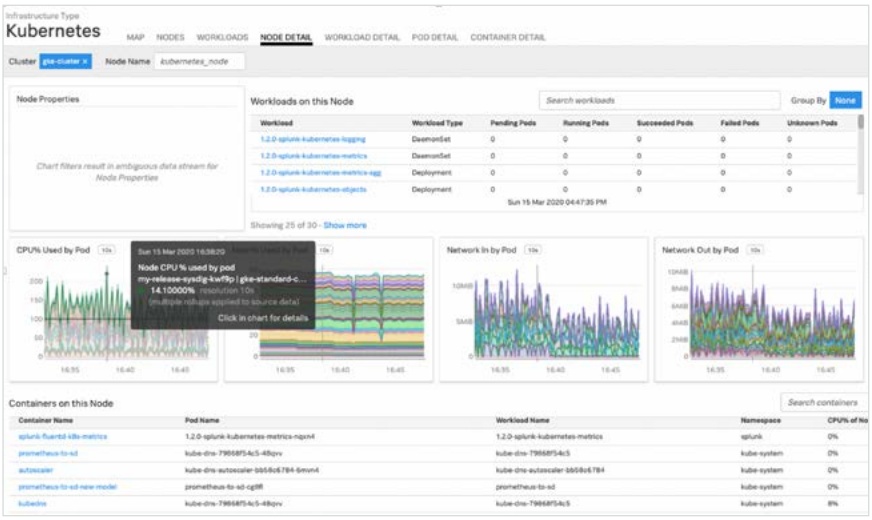
浏览您的 Kubernetes 环境

预构建 Kubernetes 仪表板

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。 选择、过滤或搜索任何 Kubernetes 实体，并向下钻取以进行详细分析，例如节点、Pod 和容器级别。了解动态 Kubernetes 组件之间的关系，并快速修复由嘈杂邻居引起的相互依赖的性能问题。



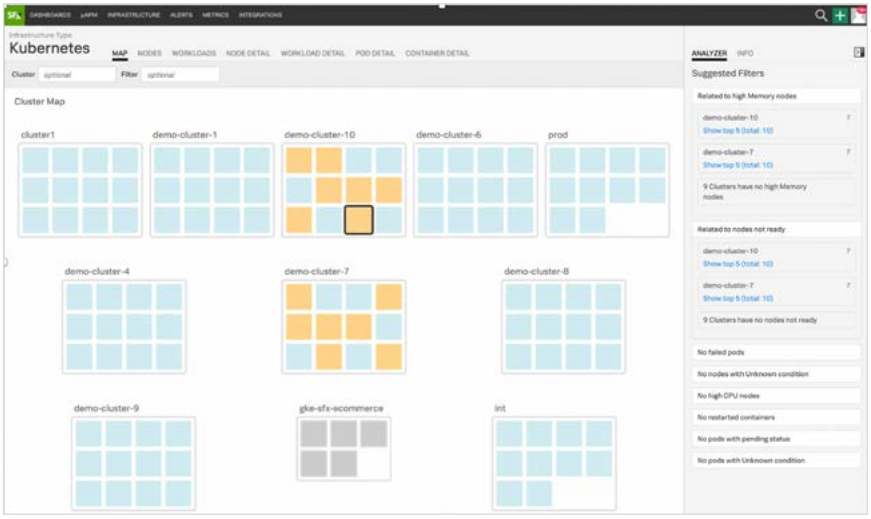
深入到单个节点将显示该特定节点的系统指标，以及该节点上运行的特定服务的仪表板。



您还可以查看您的 Kubernetes 集群中运行的所有 Pod，并跟踪特定 Pod 或集群中所有 Pod 的活动。从这里，可以深入到每个 Pod 中的单个 Docker 容器。

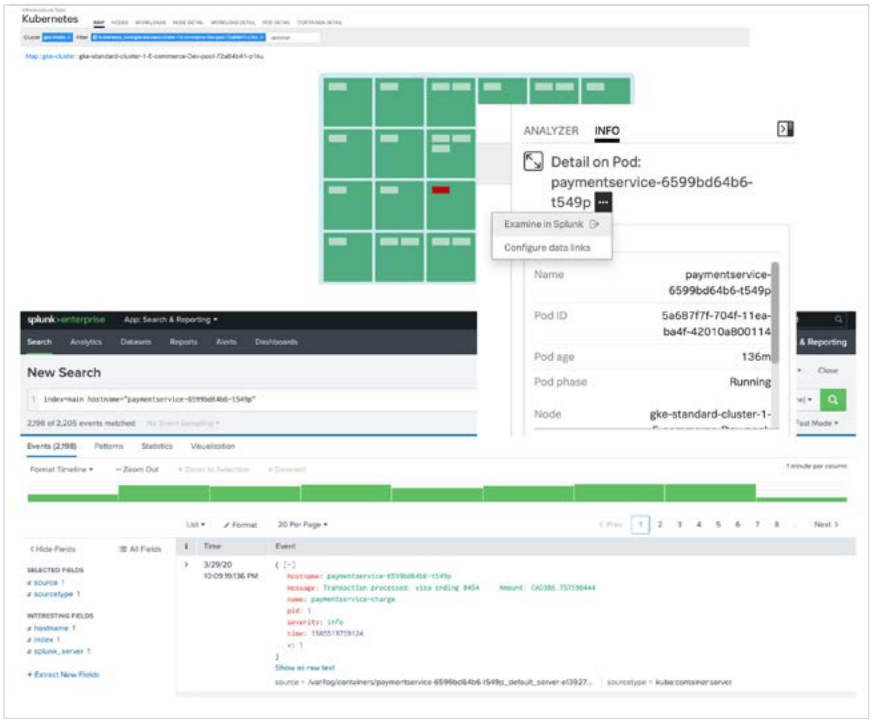
Kubernetes Analyzer

AI 驱动的分析可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 - 节点、容器和工作负载。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。



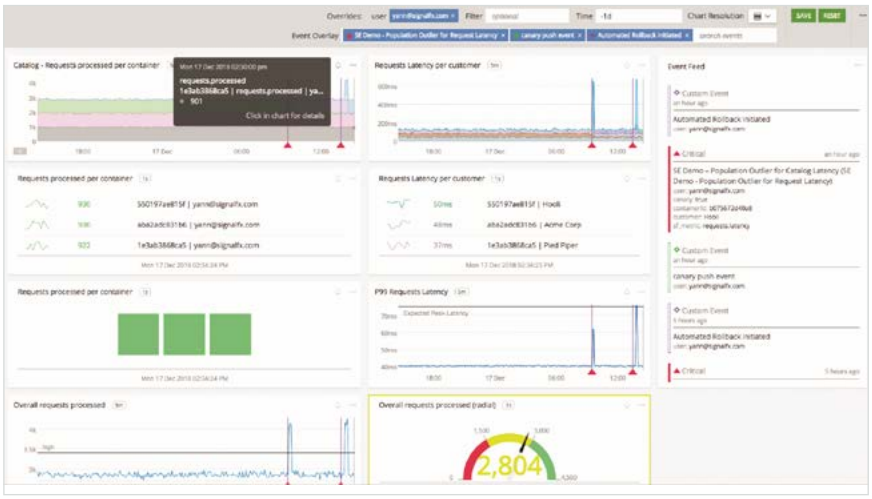
上下文中的日志

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。



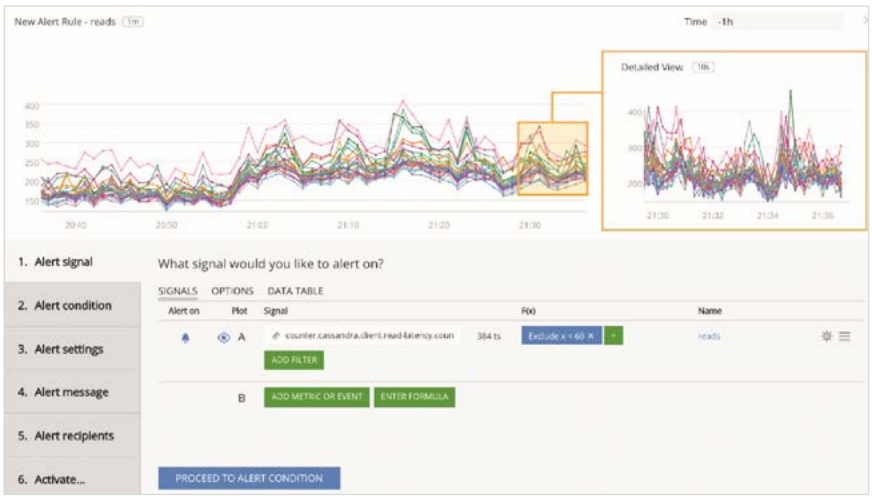
创建自定义服务仪表板

SignalFx 允许您查看容器指标和其他性能指标，以提供您环境中每层的可见性。例如，希望监控 canary 部署的服务所有者可能会创建以下仪表板，在测量请求延迟的图表旁边显示容器指标，以及跟踪代码推送、警报和采取的任何补救措施的事件馈送。



实时检测问题

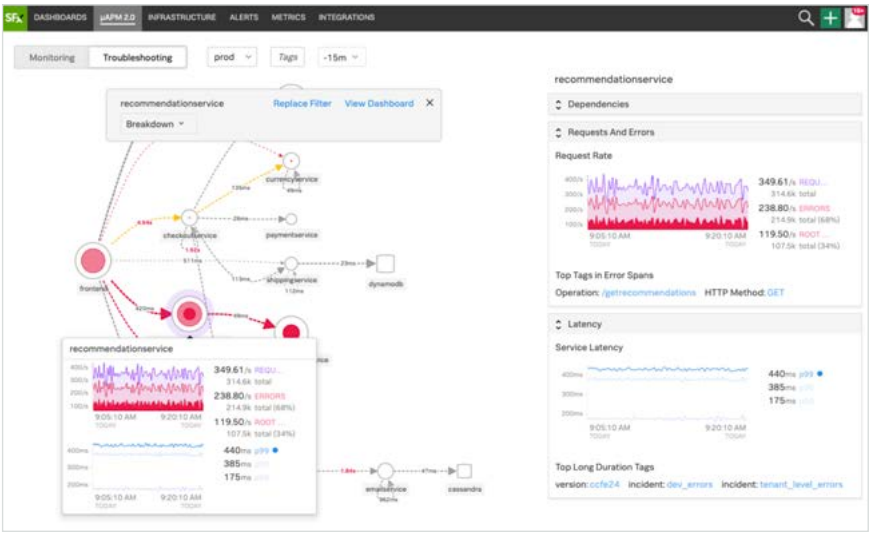
有了 SignalFx，设置警报不必是准确性和及时性之间的折衷。通过将 Kubernetes 集群的高分辨率指标与 SignalFx 中的统计函数和算法库相结合，主动对问题发出警报。根据历史数据预览警报，以便在部署前对其进行调整，并将数据科学应用于警报，以避免在非常短暂的环境中出现常见的误报和警报风暴。



利用分布式跟踪进行定向故障排除

SignalFx 微服务 APM™ 为用户提供分布式跟踪功能，以便在发生影响性能的事件时深入挖掘。无论您的 Kubernetes 环境中出现什么问题，您都可以从实时警报直接导航到应用程序跟踪，并将基础设施、Kubernetes 和您的微服务之间的性能趋势关联起来。SignalFx 解决方案建立在 **NoSample™** 全保真架构之上，该架构吸收并分析所有踪迹，因此，永远不会漏检离群值和异常。

SignalFx Outlier Analyzer™ 通过在离群点跟踪中显示最常见的模式来提供定向故障排除，使您能够快速缩小异常 Kubernetes 节点、集群、云区域或应用程序特定的标签或标记。



提高开发运维速度

一旦来自 Kubernetes 的数据流入 SignalFx，开发运维团队便可以更轻松地利用监控最佳实践和解决方案来解决整个组织中的常见问题。SignalFx 提供了开箱即用的即时可见性，通过零接触内置仪表板，可以看到主机、容器、Pod 和 Kubernetes 对象，同时还允许团队根据其特定需求定制仪表板、警报和通知。

中央团队可以使用 SignalFx API 以编程方式创建监控内容和定义分析，并利用 SignalFx 服务局来控制用户和团队的使用、访问和编辑权限，从而将监控作为代码进行交付。

```
curl \  --request post \ --header "x-sf-token: your_access_
token" \ --header "content-type: application/json" \ --data \'{
"name": "cpu load", "programtext": "data('cpu.load').publish()"
}' \ https://api.signalfx.com/v2/chart
```

Splunk 受到领先组织的信任

全球各行各业的企业都在利用 SignalFx 加速他们的云原生之旅，并满怀信心地在他们的组织中采用 Kubernetes。以下仅列出了一些客户。

更多信息： https://www.splunk.com/en_us/customers.html

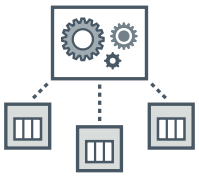
立即开始使用 SignalFx 基础设施监控

准备好了解 SignalFx 如何加速 Kubernetes 在贵组织中的采用了吗?

我们的客户成功经理和解决方案工程师将与您的团队合作，加速您对 Kubernetes 的采用，并实时监控 Kubernetes。

注册免费试用

开始使用



将 SignalFx 智能代理部署为 DaemonSet

了解更多信息

开始监控 Kubernetes 环境

了解更多信息

要了解有关实时监控和故障排除 Kubernetes 环境的 Kubernetes 导航器的更多信息，请下载免费的 Kubernetes 导航器数据表。

splunk>

了解更多信息： www.splunk.com/asksales

www.splunk.com

20-signalfx-monitoring-microservices-kubernetes-WP-111-A4