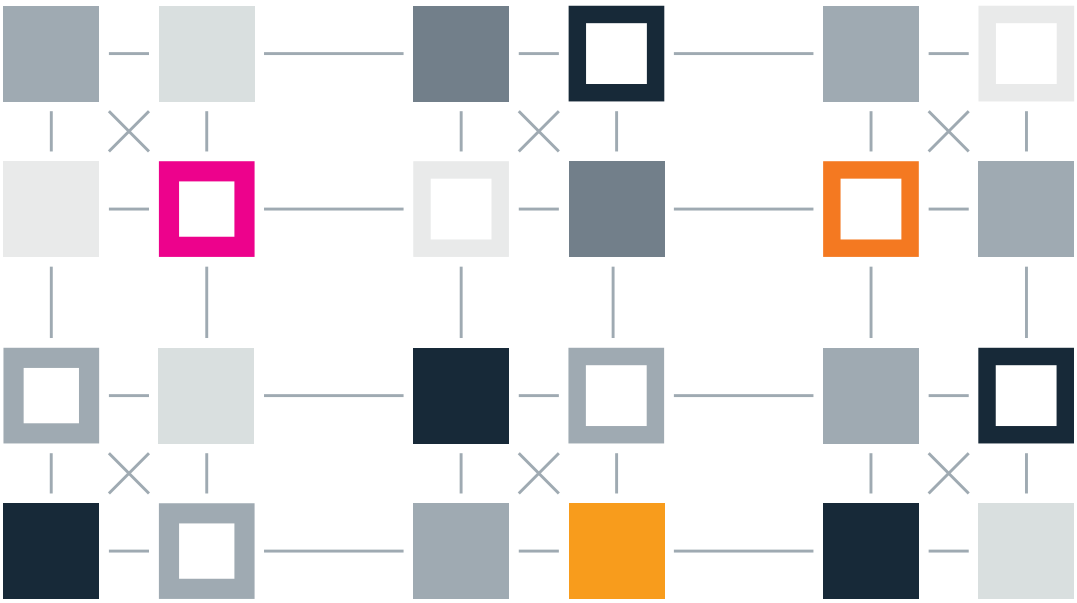
Kubernetes上のマイクロサービスの監視



半年ごとに行われるCNCF調査の最新版によれば、Kubernetesを採用する組織にとって、監視が最大の課題の1つとなっています。

このホワイトペーパーでは、KubernetesクラスターおよびKubernetes上にデプロイされたマイクロサービスの両方を最適な方法で監視してパフォーマンスの問題を特定し、平均解決時間(MTTR)を最小限に抑える方法を説明します。

パフォーマンスの問題をリアルタイムで管理できれば、エンドユーザーに円滑なエクスペリエンスを確実に提供できます。そのためには、企業はベストプラクティスを策定し、Kubernetesクラスターを適切に運用するとともに、すべてのマイクロサービスの分散トレーシングを含めスタック全体にわたるパフォーマンスを正確に把握することが必要です。

コンテナ化されたアプリケーションのオーケストレーションを支える頭脳

Kubernetesは、コンテナ化されたアプリケーションのデプロイを支援する、コンテナオーケストレーションソリューションの代表格です。あらゆる業界で多くの企業が、以下のメリットを得ようと、コンテナ化されたマイクロサービスを採用しています。

- **スピード** — イノベーションを加速化
- **規模** — 地域やクラウドプロバイダーを問わず、最適化されたアプリケーションエクスペリエンスを提供
- **効率** — リソース消費の最適化により、コストを低減しつつ価値を拡大

Kubernetesは論理層をいくつか持ち、これによって多様なユースケースに対応し、コンテナ化されたマイクロサービスのライフサイクルを管理します。

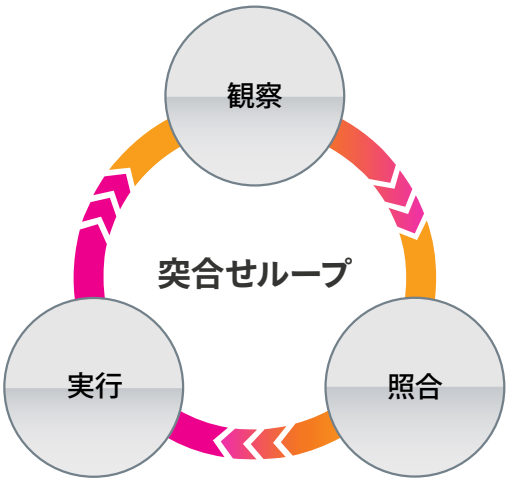
Kubernetesデプロイメントの理念には、3本の大きな柱が存在します。

インフラの抽象化	宣言型設定	不変性
コンピューターサーバー、ストレージボリューム、ネットワークなどのインフラリソースからアプリケーションオーケストレーションを行います。	Kubernetesはデプロイ済みのアプリケーションとKubernetesオブジェクトの状態を継続的に監視します。また、特定のサービスのレプリカをいくつ実行するかといった、運用者が定義したシステムの理想の状態を実装します。	Kubernetesオブジェクトは、変更できないコンテナイメージを持つDockerによって規定された不変性要件を維持する上、それ自体も変更できません。たとえば、Kubernetesサービスの各バージョンは完全に独立しており、互換性を持ちません。

Kubernetesのメリット

Kubernetesは、コンテナ化のメリットをアプリケーション開発者に提供するだけでなく、運用についても以下のようにさまざまな点で効率化します。

アプリケーションのライフサイクル管理の自動化	クラウドの相互運用性	自己修復システム
Kubernetesは、エンドツーエンドのソリューションとしてアプリケーションのライフサイクル管理を自動化します。アプリケーションプロビジョニング、自動スケーリング、レプリケーション管理に加え、CI/CDなどに対応します。	Kubernetesはインフラを抽象化するので、アプリケーションをクラウド間で自由に移動することができます。	Kubernetesは、システムの安定性を損なうあらゆる障害や信頼性の低い振る舞いからアプリケーションを保護します。宣言型設定で表明されたとおりの理想の状態にシステムを保つために、アクションを継続的に起こします。

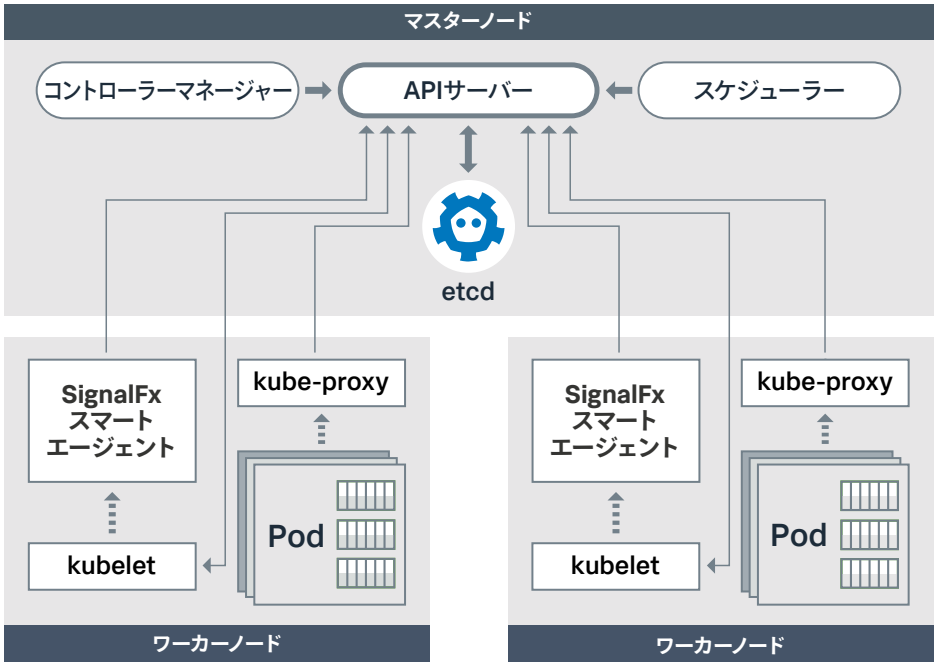


Kubernetesコントローラーマネージャーが理想の状態と実際の状態を継続的に比較し、
両者の差を評価して適切なアクションを起こすことで、その差を解消します。

Kubernetesのコンポーネントの理解

Kubernetesは、複数のマシンを単一のユニットにまとめあげ、APIを通じて利用可能にする仕組みです。さらに、コンピューターリソースを以下の2つのグループに分類します。

ワーカーノードとマスターノード



ワーカーノードとマスターノードで稼動する主なコンポーネントは以下のとおりです。

マスターノード

APIサーバー

- Kubernetesクラスターへのゲートウェイ
- クライアントからのすべてのリクエストと、etcdに格納されたAPIオブジェクトを仲介する
- 認証とロールベースアクセス制御(RBAC)を実施する
- 検証とアドミッションコントロールをリクエストする
- 詳細：http://bit.ly/k8s-apiserver

コントローラーマネージャー

- Kubernetesに組み込まれたコントロールループを実行するデーモンプロセスで、ローリングデプロイメント、ワーカーノードの数、レプリカセットなどを管理する
- 詳細：http://bit.ly/k8s-ctrl-mgr

スケジューラー

- アフィニティ、利用可能なリソース、ラベル、QoSなどの複数の要因に基づき、Podを実行すべき場所を判断する
- 詳細：http://bit.ly/k8s-scheduler

etcd

- Kubernetesクラスターの中核で、すべてのKubernetesオブジェクトのキーバリュー情報を保持する

詳細：http://bit.ly/2COkMx9

ワーカーノード

kubelet — 各ワーカー上のエージェント

- PodSpecを使用してPod(1つ以上のコンテナからなるグループ)を起動し、各Podの稼動と健全性を確保する
- Dockerなどのコンテナとやり取りする
- 詳細：http://bit.ly/k8s-kubelet

kube-proxy — 各ワーカー上のエージェント

- Kubernetesサービスのネットワークプロキシ兼ロードバランサー
- 詳細：http://bit.ly/k8s-proxy

ほとんどのアプリケーションには、次のKubernetesアドオンが必要です。

kube-dns

- Kubernetes上にPod兼サービスとしてプロビジョニングされる
- Kubernetesでは、各サービスにDNSエントリが付与される
- kube-dnsは、クラスター内にあるすべてのサービスのDNSを解決する

kubectl

- Kubernetesの公式コマンドライン
- バックグラウンドでは、KubernetesのAPIサーバーに対してRESTベースのAPIコールを使用する

Kubernetesの主な構成要素とオブジェクト：

Namespace	Pod
単一クラスターの仮想セグメンテーション	Kubernetesによって管理される1つ以上のコンテナを、論理的にグルーピングしたもの
ノード	**ReplicaSet**
Kubernetesのインフラファブリック(ワーカーおよびマスターコンポーネントのホスト)	特定数のPodの稼動を確保する継続的なループ
ロール	**Ingress**
Kubernetesクラスターに対するロールベースアクセス制御	外部からホスト型サービスへ向かうHTTPトラフィックの管理
Deployment	**サービス**
ReplicaSet、Podの定義、アップデートといった概念の管理	IPやDNSなどの不変性を動的なPodに提供する論理層

Kubernetes環境を監視する上での課題

最新のCNCF調査によれば、Kubernetesを採用する組織にとっての最大の課題として、複雑さ、監視、セキュリティが挙げられています。従来の監視戦略では、クラウドネイティブ時代の以下のような状況に対応できません。

- 監視するコンポーネントが大幅に増えている
- コンテナが短命かつ動的である
- Kubernetesは、リソース使用率を最適にするようにPodを自動的にスケジューリングする。これによって効率化が実現する一方、アフィニティやTaintを使用して特定の意図が表明されていない限り、Podのデプロイ先および実行先を予測できない

その結果、Kubernetes環境は以下のような状態になります。

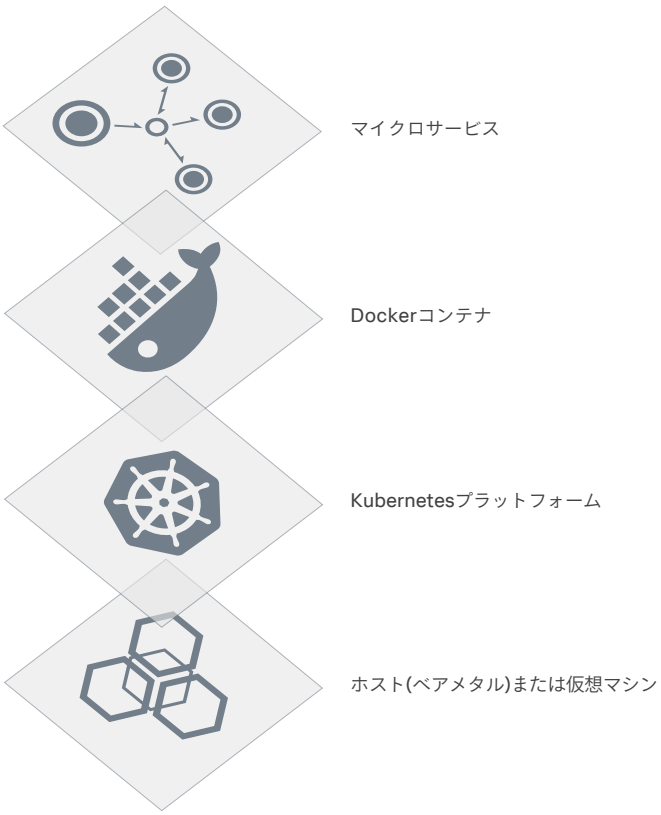
- 監視するコンポーネントが大幅に増えている
- コンテナが流動的なため、Podの配置を予測できない
- コンテナ化されたマイクロサービスでは、トラブルシューティングがより複雑

監視するコンポーネントの増大

モノリシックな環境において、監視するコンポーネントは、アプリケーションとそのデプロイ先であるホストという2つだけです。

これに対して、クラウドネイティブの環境では、Kubernetesがオーケストレーションを手がけるコンテナ化されたアプリケーションに関して、監視を必要とするコンポーネントが以下のように複数存在します。

- ホスト
- Kubernetesプラットフォーム本体
- Dockerコンテナ
- コンテナ化されたマイクロサービス



コンテナの流動的な変化と配置予測の難しさ

従来の長期稼動型ホストモデルと異なり、マイクロサービスベースの最新のアプリケーションは通常、動的かつ短命なコンテナにデプロイされます。Kubernetesは、理想とされる数のアプリケーションレプリカが確実に実行されるようにします。また、ノードアフィニティやTaintを通じて特定のノードを回避する指示がない限り、Kubernetesが適切と判断したノードにPodを配置します。実際、この自己調整システムについては、KubernetesによるPodのスケジューリングが重要な設計目標となります。

従来の監視方法ではホスト名やIPアドレスを使用して長期稼動型ホストを追跡するのが一般的なので、こうした極めて動的な環境に対応できません。コンテナ化された環境では、監視ツールによる即時のサービスディスカバリーとコンテナのライフサイクルイベントの自動検出とともに、コンテナのスピンアップ時や再起動時にはメトリクス収集を秒単位で調整して対応する必要があります。

マイクロサービスのパフォーマンスの監視

マイクロサービス環境では、モノリシックな環境と比べて問題の特定が難しくなります。これは、リクエストがスタックの異なる層やさまざまなサービスの間を行き交うためです。最新の監視ツールは、こうした相関関係を持つ層を監視すると同時に、アプリケーションとインフラの挙動を効率的に関連付けることで、トラブルシューティングを合理化する必要があります。

監視すべき主要なパフォーマンスメトリクス

Podのメトリクス

各クラスターの健全性と最適なリソース使用状況を維持するために、すべてのKubernetesオブジェクトを監視することが不可欠です。KubernetesのPod、Deployment、サービスのメトリクスを監視することによって、環境内でKubernetesが意図されたとおりに稼動しているかを判断できます。

Pod	理想とされるPodの数
	利用可能なPodの数
	フェーズごとのPod(Failed、Pending、Running)
	Deploymentごとに理想とされるPod
	サービスごとに理想とされるPod
	Deploymentごとに利用可能なPod
	サービスごとに利用可能なPod
	ReplicaSetごとに理想とされるPod
	ノードごとに稼動中のPod

リソース使用状況のメトリクス

アプリケーションおよびKubernetesクラスターの健全性を維持するためには、リソース使用状況の追跡も重要です。

- Docker Socketは、コンテナのメトリクス、およびCPUやメモリー使用率といったノードレベルのリソース使用状況に関するメトリクスを提供します。Kubernetesは自身が生成したメトリクスに加え、collectdメトリクスを提供します。
- CPU、メモリー、ディスクI/O、ネットワークパフォーマンスに関するメトリクスを、アプリケーションパフォーマンスおよびKubernetesイベントと関連付けることによって、パフォーマンス上の問題の根本原因をより素早く特定することができます。
- コンテナやPodのライフサイクルイベントといった、DockerおよびKubernetesイベントの監視は、設定ミスやリソース枯渇の発見につながります。

SignalFxを活用したKubernetesの監視

Kubernetesは、コンポーネントおよびアプリケーションのリソース使用状況に関する詳細情報をクラスター、Pod、およびサービスレベルで提供します。この情報は多彩な監視ソリューションによって簡単に収集できますが、データから実用的なインサイトを引き出す作業はエンドユーザーに任されています。

SignalFxでは、柔軟性の高いオープンなインストルメンテーションおよび事前構築済みのダッシュボードを通じてKubernetesを監視できるので、環境のさまざまな層を即座に可視化できます。SignalFxユーザーは、ストリーム分析と分散トレーシングを組み合わせることで、基本的なデータ収集にとどまらず、リアルタイムのアラート生成とパフォーマンス分析を基に問題を秒単位で検出して解決することが可能です。

クラウドプロバイダーからのメトリクスの収集

AWS Elastic Container Service for Kubernetes (EKS)、Google Kubernetes Engine (GKE)、Azure Kubernetes Serviceなどのマネージド型Kubernetesサービスの基本的な監視を行う上で、メトリクスを最もシンプルに収集する方法は、SignalFxをAWS CloudWatch、Google Stackdriver、Azure Monitorといったサービスと連携させる方法です。

この方法は最もシンプルであり、エージェントをインストールしたりアプリケーションコードを変更したりしなくても、SignalFxでKubernetesメトリクスの収集が可能になります。ただし、これらのサービスは、メトリクスの収集およびレポーティング頻度はデフォルトで比較的低く設定されている上(AWS CloudWatchの更新頻度はデフォルトで5分おき)、Kubernetesクラスターにデプロイ済みの特定のサービスに関するインサイトは提供されません。

完全に自動化されたKubernetes監視

サービスの詳細なインサイトを取得し、コンテナのメトリクスをきめ細かく監視するためには、Kubernetesクラスター全体にSignalFxスマートエージェントをインストールすることをおすすめします。

スマートエージェントは、collectdをベースにしたオープンソースのメトリクス収集エージェントで、監視コンテンツのために設定とサービスの検出を自動で行います。この方法のメリットは、スマートエージェントはメトリクスをSignalFxに1秒の解像度で送信できるので、エフェメラルな性質を持つKubernetesPodと非常に相性が良いことです。

スマートエージェントはKubernetesでDaemonSetとして稼動することで、クラスター内の各ノードに確実にインストールされるようにします。Kubernetesからデータを収集するほか、cAdvisorを使用して、稼動中のDockerコンテナからリソースおよびパフォーマンスの傾向に関する情報を取得します。ゼロタッチ設定、さらにKubernetesコンポーネントおよびコンテナ化されたサービスのオートディスカバリーにより、スタック全体を即座に監視します。SignalFxスマートエージェントのインストールは、以下のとおりシンプルなワンステップで完了します。

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

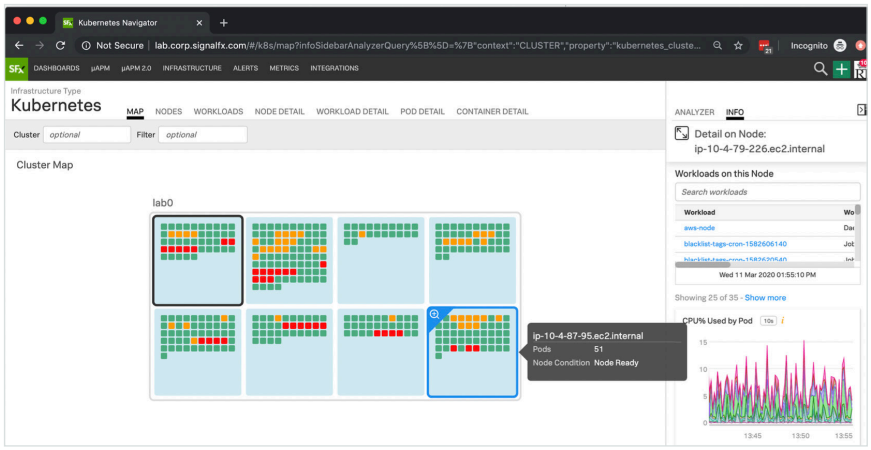
スマートエージェントが監視できるのは、アップストリームのKubernetesだけではありません。AWS Elastic Container Service for Kubernetes (EKS)やGoogle Kubernetes Engine (GKE)といったマネージド型Kubernetesサービス、さらにOpenshiftやPivotal Kubernetes Service (PKS)といったプラットフォームからもメトリクスを収集することが可能です。

スマートエージェントはまた、PrometheusやStatsDといったその他のメトリクスプロトコルからもデータを収集可能な監視機能を提供するので、メトリクスパイプラインがすでにある場合にもSignalFxを容易に活用できます。

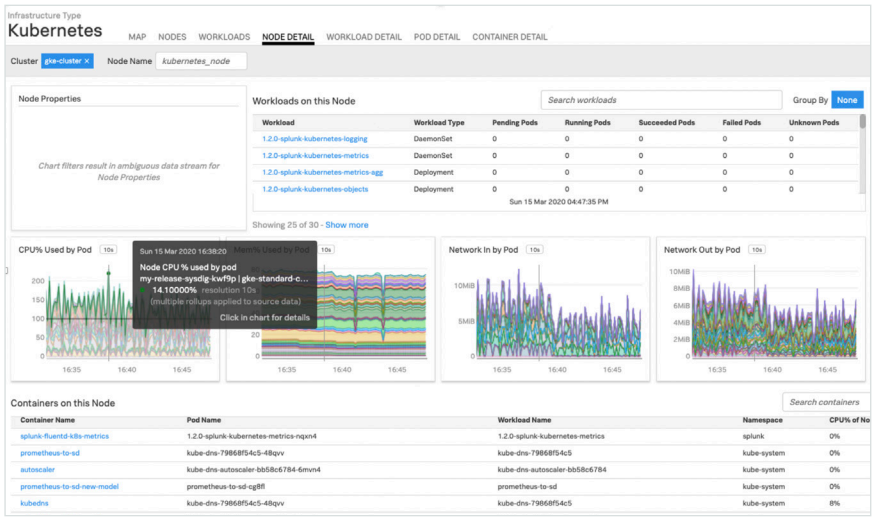
Kubernetes環境全体の把握

事前構築済みのKubernetesダッシュボード

まずは全体像として、Kubernetes Navigatorの分かりやすい階層型のナビゲーションによって、ユーザーはKubernetes環境全体のパフォーマンスを素早く理解できます。任意のKubernetesエンティティを選択、フィルタリング、検索したうえでドリルダウンして、ノード、Pod、コンテナレベルなどへの詳細な分析に秒単位でたどり着けます。これによって、動的なKubernetesコンポーネントの相互関係を把握して、「ノイジーネイバー (うるさい隣人)」から生じる、相互依存性に伴うパフォーマンスの問題を迅速に解決することが可能です。



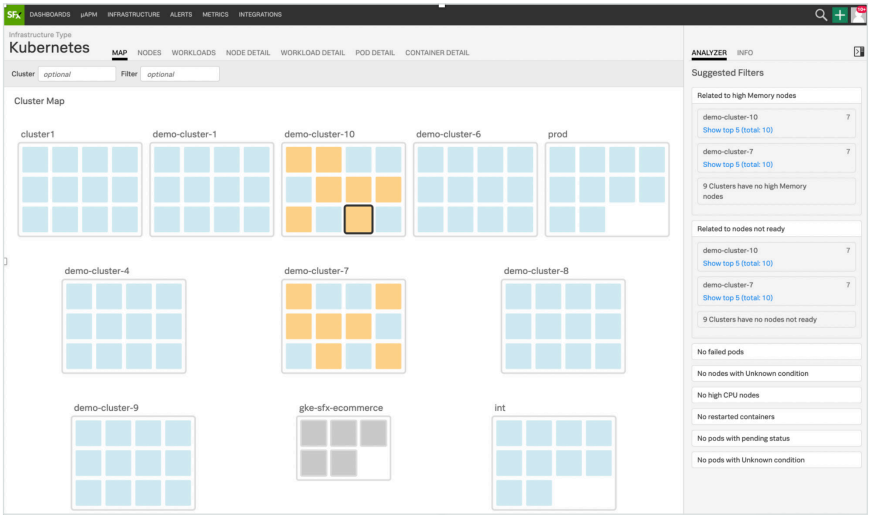
個々のノードにドリルダウンすると、そのノードのシステムメトリクスと合わせて、そのノードで稼動中のサービスのダッシュボードが表示されます。



Kubernetesクラスターで稼動しているすべてのPodを表示したり、クラスター内の特定のPodやすべてのPodのアクティビティを追跡したりすることも可能です。ここから、各Pod内の個々のDockerコンテナにドリルダウンすることができます。

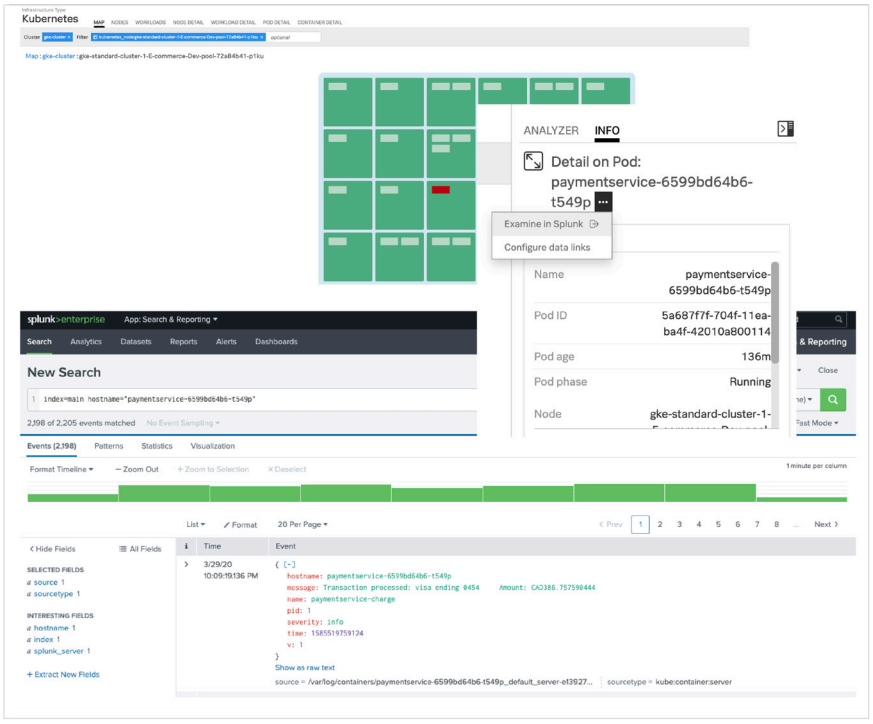
Kubernetes Analyzer

AIベースの分析によって、Kubernetesクラスター全体(ノード、Pod、コンテナ、ワークロード)で異常値が発生している原因をリアルタイムで正確に理解するためのインサイトと推奨事項が自動的に導き出されます。goroutineやコンテナの再起動の急増のようなシステムレベルの問題も、Historical Performance Baselines、Sudden Changeなどの高度なアルゴリズムによって秒単位で検出し、アラートを発行します。



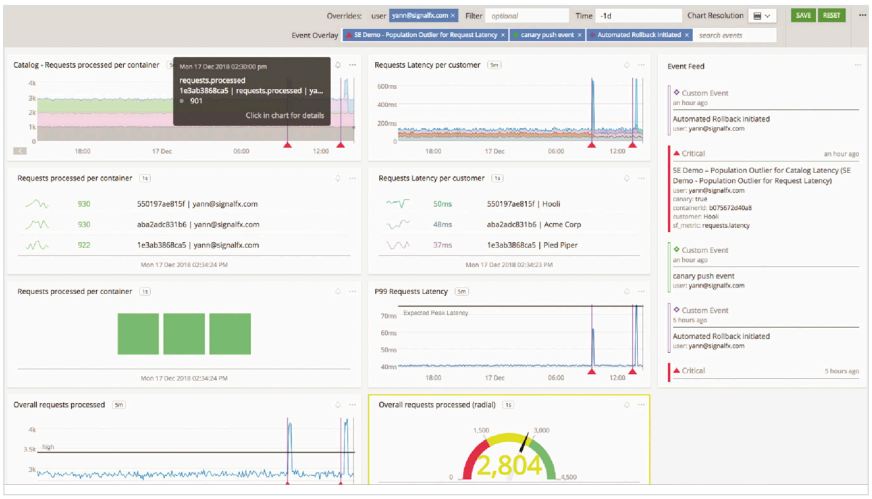
コンテキストにおけるログ

ログをシームレスに切り替え、アプリケーション、Kubernetes、およびコンテナのログを詳細に可視化し、コンテキストを切り替えることなくスタック全体でパフォーマンスの相関付けを行うことができます。また、KubernetesおよびAPIサーバーの監査ログにあるライフサイクルイベントを可視化することで、セキュリティ体制とコンプライアンス体制の把握と管理が行いやすくなります。



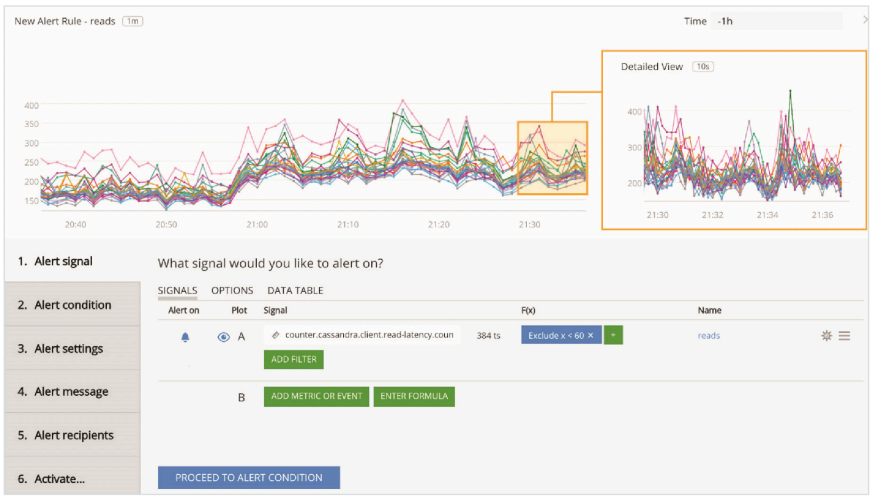
カスタムサービスダッシュボードの作成

SignalFxでは、コンテナのメトリクスをその他のパフォーマンス指標を合わせて表示できるため、環境内の各層にわたって可視化することが可能です。たとえば、カナリアデプロイを監視したいサービスオーナーであれば、以下のようなダッシュボードを作成することができます。この場合は、コンテナのメトリクスの隣に、リクエストレイテンシを測定するグラフとコードプッシュ、アラート、修復作業の実施を追跡するイベントフィードが表示されています。



問題のリアルタイム検出

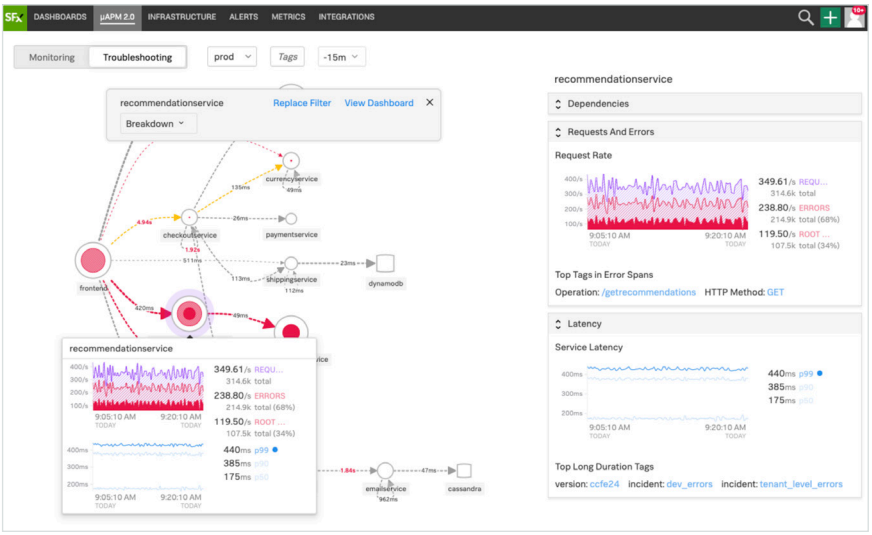
SignalFxでアラートを設定する際には、正確さと即時性のバランスで悩む必要がありません。Kubernetesクラスターから届く高解像度のメトリクスに、SignalFxの統計機能とアルゴリズムのライブラリを組み合わせることで、問題に関するアラートをプロアクティブに生成することができます。また、アラートをデプロイ前に履歴データと照合して調整できる上、エフェメラルな環境でありがちな誤検知や大量のアラートも、データサイエンスの適用によって回避可能です。



分散トレーシングを活用した指示ありトラブルシューティング

SignalFx Microservices APM™は、分散トレーシング機能を提供し、パフォーマンスに影響のあるイベントの発生時に詳細の把握を可能にします。Kubernetes環境内のどこで問題が発生した場合でも、リアルタイムアラートからアプリケーショントレースへと直接移動し、インフラ、Kubernetes、マイクロサービスの間のパフォーマンスの傾向を相関付けることができます。SignalFxソリューションは、**NoSample™**による完全忠実なアーキテクチャを基盤としています。このアーキテクチャはすべてのトレースを取りこんで分析するため、外れ値も異常も取りこぼしがありません。

SignalFx Outlier Analyzer™は、トラブルシューティングの方向性を提供します。外れ値トレースで最も頻出しているパターンをあぶり出し、異常のあるKubernetesノード、クラスター、クラウドリージョン、アプリケーション固有のラベルやタグへの素早い絞り込みを可能にします。



DevOpsの加速化

KubernetesからのデータをSignalFxに取りこめば、DevOpsチームは監視のベストプラクティスを組織全体で実践しやすくなり、頻出している問題をスムーズに解決できるようになります。SignalFxはゼロタッチ設定とビルトインのダッシュボードを備えているので、ホスト、コンテナ、Pod、Kubernetesオブジェクトを迅速に可視化できます。また、ダッシュボード、アラート、通知を固有のニーズに合わせてカスタマイズすることも可能です。管理チームはSignalFxのAPIを使用して監視機能をコードとして配信でき、これによってプログラムで監視コンテンツを作成して分析を定義できます。さらに、SignalFxサービスビューロを活用してユーザーとチームの使用権限、アクセス権限、編集権限をコントロールすることも可能です。

```
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は一流企業の信頼を獲得

全世界であらゆる業界の企業が、SignalFxを活用してクラウドネイティブへの移行を加速させ、Kubernetesの活用を着実に進めています。以下のお客様をはじめとする多数の企業にご利用いただいています。

詳細：https://www.splunk.com/ja_jp/customers.html

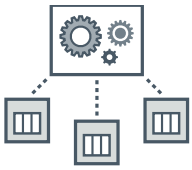
SignalFXによるインフラ監視を始めよう

SignalFxによって実現する、Kubernetesの高速デプロイにご興味ございますか？

Splunkのカスタマーサクセスマネージャーとソリューションエンジニアは、お客様のチームと連携してKubernetesの高速デプロイとリアルタイム監視をサポートする体制を整えてお待ちしております。

無料トライアルに申し込む

申し込み



DaemonSetとしてSignalFxスマートエージェントをデプロイする

詳細はこちら

Kubernetes環境の監視を開始する

詳細はこちら

あらゆる規模のKubernetes環境のリアルタイム監視とトラブルシューティングを実現するKubernetes Navigatorの詳細は、Kubernetes Navigatorのデータシート(無料)をダウンロードしてご確認ください。

splunk>

お問い合わせ：https://www.splunk.com/ja_jp/talk-to-sales.html
〒100-0004 千代田区大手町1-1-1 大手町パークビルディング 8階

www.splunk.com/ja_jp
splunkjp@splunk.com

© 2020 Splunk Inc. 無断複写・転載を禁じます。 Splunk, Splunk>, Data-to-Everything, D2EおよびTurn Data Into Doingは、米国およびその他の国におけるSplunk Inc.の商標または登録商標です。他のすべてのブランド名、製品名、もしくは商標は、それぞれの所有者に帰属します。

20-signalfx-monitoring-microservices-kubernetes-WP-A4-JA-202009