Наблюдаемость

Обзор

Обзор

Мониторинг

Введение

Установка

Overview

Архитектур

Основные понятия

Monitoring

Alarms

Notifications

Monitoring Dashboard

Руководств

ML

Как сделат

Распределённое трассирование

Введение

Ограничения использования

Установка

Установка Jaeger Operator

Развертывание экземпляра Jaeger

Архитектур

Основные ком

Поток данных

Основные понятия

Telemetry

OpenTelemetry

Span

Trace

Instrumentation

OpenTelemetry Collector

Jaeger

Руководств

Как сделат

Устранение

Логи

О сервисе логирования

События

Введение

Ограничения по использованию

Events

Operation Procedures

Event Overview

Инспекция

Введение

Ограничения использования

Архитектура

Inspection

Component Health Status

Руководств

Обзор

Модуль Observability является ключевой функцией платформы ACP, обеспечивающей комплексные возможности мониторинга и наблюдаемости для облачно-нативных приложений.

Этот модуль объединяет четыре основных столпа наблюдаемости:

- **Synthetic monitoring (probe)** для проактивного тестирования конечных точек
- **Centralized logging** для централизованного управления и анализа логов
- **Real-time monitoring** для сбора метрик и оповещений в реальном времени
- **Distributed tracing** для сквозного отслеживания запросов между микросервисами

Объединяя эти возможности в единой платформе, он позволяет организациям получить полную видимость производительности приложений, быстро диагностировать проблемы, обеспечивать надежность системы и оптимизировать пользовательский опыт по всему технологическому стеку.

Мониторинг

Введение

Введение

Установка

Установка

Overview

Installation Preparation

Install the ACP Monitoring with Prometheus Plugin via console

Install the ACP Monitoring with Prometheus Plugin via YAML

Install the ACP Monitoring with VictoriaMetrics Plugin via console

Install the ACP Monitoring with VictoriaMetrics Plugin via YAML

Архитектура

Архитектура модуля монитор

- Общее описание архитектуры
- Система мониторинга
- Система оповещений
- Система уведомлений

Руководство по выбору комп

- Важные замечания
- Список компонентов
- Сравнение архитектур
- Сравнение функций
- Рекомендации по схемам установки

Планирова

- Предположени
- Prometheus
- VictoriaMetrics

Основные понятия

Основные понятия

- Monitoring
- Alarms
- Notifications
- Monitoring Dashboard

Руководства

Управление метриками

- Просмотр метрик, предоставляемых ко
- Просмотр всех метрик, хранящихся в Pr
- Просмотр всех встроенных метрик, опре
- Интеграция внешних метрик

Управление оповещениями

- Обзор функции
- Ключевые возможности
- Преимущества функции
- Создание политик оповещений через UI
- Создание оповещений по ресурсам чер

Управлени

- Обзор функци
- Ключевые фун
- Notification Ser
- Notification Cor
- Notification Tem

Создание оповещений по событиям чер
Создание политик оповещений через al
Настройка заглушения оповещений
Рекомендации по настройке правил оповещений

Notification rule
Настройка пра

Управление Probe

Обзор функции
Blackbox мониторинг
Оповещения Blackbox
Настройка модуля мониторинга BlackboxExporter
Создание элементов Blackbox мониторинга и оповещений через CLI
Справочная информация

Как сделать

Резервное копирование и вос Prometheus

Обзор функции
Сценарии использования
Предварительные требования
Процедуры работы
Результаты операции
Дополнительная информация
Следующие шаги

Резервное копирование и вос VictoriaMetrics

Обзор функции
Сценарии использования
Предварительные требования
Процедуры
Результат операции
Дополнительная информация
Последующие действия

Сбор сетев именами

Обзор функци
Сценарий испо
Предварительны
Порядок дейст
Результаты опе
Дополнительна
Последующие

Введение

Модуль Monitoring является ключевым компонентом набора средств наблюдаемости платформы ACP, предоставляющим комплексные возможности мониторинга и оповещения для администраторов платформы и операционных команд.

Этот модуль обеспечивает четыре основные функции мониторинга:

- **Сбор метрик** для получения данных о производительности в реальном времени с кластеров, узлов, приложений и контейнеров
- **Панели мониторинга** для интуитивной визуализации и анализа состояния системы и тенденций производительности
- **Оповещения** для проактивного выявления проблем с помощью настраиваемых правил и пороговых значений
- **Уведомления** для своевременной доставки информации об оповещениях операционному персоналу

Интегрируя эти возможности с open-source компонентами, такими как Prometheus и VictoriaMetrics, модуль позволяет организациям поддерживать надежность системы, предотвращать простои, снижать операционные затраты и обеспечивать оптимальную производительность всей инфраструктуры.

Установка

Содержание

- Overview
- Installation Preparation
- Install the ACP Monitoring with Prometheus Plugin via console
  - Installation Procedures
  - Access Method
- Install the ACP Monitoring with Prometheus Plugin via YAML
  - 1. Check available versions
  - 2. Create a ModuleInfo
  - 3. Verify installation
- Install the ACP Monitoring with VictoriaMetrics Plugin via console
  - Prerequisites
  - Installation Procedures
- Install the ACP Monitoring with VictoriaMetrics Plugin via YAML
  - 1. Check available versions
  - 2. Create a ModuleInfo
  - 3. Verify installation

Overview

Компонент мониторинга служит инфраструктурой для функций мониторинга, оповещений, инспекции и проверки состояния в модуле наблюдаемости. В этом документе описывается, как установить ACP Monitoring с плагином Prometheus или ACP Monitoring с плагином VictoriaMetrics в кластере.

Installation Preparation

> **INFO**
>
> Некоторые компоненты Monitoring требуют значительных ресурсов. Рекомендуется запускать их на infra-узлах и задавать nodeSelector и tolerations, чтобы обеспечить их работу только на этих узлах. Если вы оцениваете продукт и не выделяли infra-узлы, можно убрать эти настройки, чтобы компоненты запускались на всех узлах.
>
> Для рекомендаций по планированию infra-узлов смотрите Cluster Node Planning.

Перед установкой компонентов мониторинга убедитесь, что выполнены следующие условия:

- Выбран соответствующий компонент мониторинга, согласно Monitoring Component Selection Guide.
- При установке в рабочий кластер убедитесь, что кластер `global` может получить доступ к порту 11780 рабочего кластера.
- Если необходимо использовать storage class или постоянное хранилище для данных мониторинга, создайте соответствующие ресурсы в разделе **Storage** заранее.

Install the ACP Monitoring with Prometheus Plugin via console

Installation Procedures

1. Перейдите в **App Store Management** > **Cluster Plugins** и выберите целевой кластер.
2. Найдите плагин **ACP Monitoring with Prometheus** и нажмите **Install**.

3. Настройте следующие параметры:

Parameter	Description
Scale Configuration	Поддерживает три конфигурации: **Small Scale**, **Medium Scale** и **Large Scale**: - Значения по умолчанию основаны на рекомендованных нагрузочных тестах платформы - Можно выбрать или настроить квоты в зависимости от фактического масштаба кластера - Значения по умолчанию обновляются с версиями платформы; для фиксированных конфигураций рекомендуется использовать кастомные настройки
Storage Type	- **LocalVolume**: Локальное хранилище с данными на указанных узлах - **StorageClass**: Автоматическое создание persistent volume с использованием storage class - **PV**: Использование существующих persistent volume **Примечание**: Конфигурация хранилища не может быть изменена после установки
Replica Count	Задает количество подов компонента мониторинга **Примечание**: Prometheus поддерживает только установку на одном узле
Parameter Configuration	Параметры данных для компонента мониторинга можно настроить по необходимости

4. Нажмите **Install** для завершения установки.

Access Method

После завершения установки компоненты доступны по следующим адресам (замените `<>` на актуальные значения):

Component	Access Address
Thanos	`<platform_access_address>/clusters/<cluster>/prometheus`
Prometheus	`<platform_access_address>/clusters/<cluster>/prometheus-0`
Alertmanager	`<platform_access_address>/clusters/<cluster>/alertmanager`

Install the ACP Monitoring with Prometheus Plugin via YAML

1. Check available versions

Убедитесь, что плагин опубликован, проверив наличие ресурсов ModulePlugin и ModuleConfig в кластере `global`:

```
# kubectl get moduleplugin | grep prometheus
prometheus                        30h
# kubectl get moduleconfig | grep prometheus
prometheus-v4.1.0                 30h
```

Это означает, что ModulePlugin `prometheus` существует в кластере и версия `v4.1.0` опубликована.

2. Create a ModuleInfo

Создайте ресурс ModuleInfo для установки плагина без параметров конфигурации:

```yaml
kind: ModuleInfo
apiVersion: cluster.alauda.io/v1alpha1
metadata:
  name: global-prometheus
  labels:
    cpaas.io/cluster-name: global
    cpaas.io/module-name: prometheus
    cpaas.io/module-type: plugin
spec:
  version: v4.1.0
  config:
    storage:
      type: LocalVolume
      capacity: 40
      nodes:
        - xxx.xxx.xxx.xx
      path: /cpaas/monitoring
      storageClass: ""
      pvSelectorK: ""
      pvSelectorV: ""
    replicas: 1
    components:
      prometheus:
        retention: 7
        scrapeInterval: 60
        scrapeTimeout: 45
        resources: null
      nodeExporter:
        port: 9100
        resources: null
      alertmanager:
        resources: null
      kubeStateExporter:
        resources: null
      prometheusAdapter:
        resources: null
      thanosQuery:
        resources: null
    size: Small
  valuesOverride:
    ait/chart-kube-prometheus:
      global:
        nodeSelector:
```

```
        node-role.kubernetes.io/infra: "true"
      tolerations:
      - effect: NoSchedule
        key: node-role.kubernetes.io/infra
        value: reserved
        operator: Equal
```

Пример настройки ресурсов для prometheus:

```
spec:
  config:
    components:
      prometheus:
        resources:
          limits:
            cpu: 2000m
            memory: 2000Mi
          requests:
            cpu: 1000m
            memory: 1000Mi
```

Подробности смотрите в Monitor Component Capacity Planning

Справочник по полям YAML (Prometheus):

Field path	Description
`metadata.labels.cpaas.io/cluster-name`	Имя целевого кластера, в котором установлен плагин.
`metadata.labels.cpaas.io/module-name`	Должно быть `prometheus`.
`metadata.labels.cpaas.io/module-type`	Должно быть `plugin`.
`metadata.name`	Имя ModuleInfo (например, `<cluster>-prometheus`).
`spec.version`	Версия плагина для установки.
`spec.config.storage.type`	Тип хранилища: `LocalVolume`,

Field path	Description
	`StorageClass` или `PV`.
`spec.config.storage.capacity`	Размер хранилища для Prometheus (Gi). Рекомендуется минимум 30 Gi.
`spec.config.storage.nodes`	Список узлов при `storage.type=LocalVolume`. Поддерживается до 1 узла.
`spec.config.storage.path`	Путь LocalVolume при `storage.type=LocalVolume`.
`spec.config.storage.storageClass`	Имя StorageClass при `storage.type=StorageClass`.
`spec.config.storage.pvSelectorK`	Ключ селектора PV при `storage.type=PV`.
`spec.config.storage.pvSelectorV`	Значение селектора PV при `storage.type=PV`.
`spec.replicas`	Количество реплик; применимо только для типов `StorageClass` / `PV`.
`spec.config.components.prometheus.retention`	Количество дней хранения данных.
`spec.config.components.prometheus.scrapeInterval`	Интервал сбора данных в секундах; применяется к ServiceMonitors без `interval`.
`spec.config.components.prometheus.scrapeTimeout`	Таймаут сбора данных в секундах; должен быть меньше `scrapeInterval`.

Field path	Description
`spec.config.components.prometheus.resources`	Настройки ресурсов для Prometheus.
`spec.config.components.nodeExporter.port`	Порт Node Exporter (по умолчанию 9100).
`spec.config.components.nodeExporter.resources`	Настройки ресурсов для Node Exporter.
`spec.config.components.alertmanager.resources`	Настройки ресурсов для Alertmanager.
`spec.config.components.kubeStateExporter.resources`	Настройки ресурсов для Kube State Exporter.
`spec.config.components.prometheusAdapter.resources`	Настройки ресурсов для Prometheus Adapter.
`spec.config.components.thanosQuery.resources`	Настройки ресурсов для Thanos Query.
`spec.config.size`	Масштаб мониторинга: `Small`, `Medium` или `Large`.
`spec.valuesOverride.ait/chart-kube-prometheus.global.nodeSelector`	Необязательно. NodeSelector для компонента Monitoring. Используется для выбора узлов, на которых будет работать компонент мониторинга. Обычно применяется для infra-узлов.
`spec.valuesOverride.ait/chart-kube-prometheus.global.tolerations`	Необязательно. Tolerations для компонента Monitoring. Если nodeSelector выбирает узлы с taint, используйте этот параметр для указания ключа, значения и эффекта toleration.

Field path	Description
	Обычно применяется для infra-узлов.

3. Verify installation

Так как имя ModuleInfo меняется при создании, найдите ресурс по метке, чтобы проверить статус и версию плагина:

```
kubectl get moduleinfo -l cpaas.io/module-name=prometheus
NAME                                             CLUSTER         MODULE
DISPLAY_NAME   STATUS    TARGET_VERSION   CURRENT_VERSION   NEW_VERSION
global-e671599464a5b1717732c5ba36079795          global          promethe
us    prometheus     Running   v4.1.0           v4.1.0            v4.1.0
```

Объяснение полей:

- `NAME`: имя ресурса ModuleInfo
- `CLUSTER`: кластер, в котором установлен плагин
- `MODULE`: имя плагина
- `DISPLAY_NAME`: отображаемое имя плагина
- `STATUS`: статус установки; `Running` означает успешную установку и работу
- `TARGET_VERSION`: версия, которая должна быть установлена
- `CURRENT_VERSION`: версия до установки
- `NEW_VERSION`: последняя доступная версия для установки

Install the ACP Monitoring with VictoriaMetrics Plugin via console

Prerequisites

- Если устанавливается только агент VictoriaMetrics, убедитесь, что VictoriaMetrics Center установлен в другом кластере.

Installation Procedures

1. Перейдите в **App Store Management** > **Cluster Plugins** и выберите целевой кластер.
2. Найдите плагин **ACP Monitoring with VictoriaMetrics** и нажмите **Install**.
3. Настройте следующие параметры:

Parameter	Description
Scale Configuration	Поддерживает три конфигурации: **Small Scale**, **Medium Scale** и **Large Scale**: - Значения по умолчанию основаны на рекомендованных нагрузочных тестах платформы - Можно выбрать или настроить квоты в зависимости от фактического масштаба кластера - Значения по умолчанию обновляются с версиями платформы; для фиксированных конфигураций рекомендуется использовать кастомные настройки
Install Agent Only	- Off: Установить полный набор компонентов VictoriaMetrics - On: Установить только компонент сбора VMAgent, который зависит от VictoriaMetrics Center
VictoriaMetrics Center	Выберите кластер, в котором установлен полный набор компонентов VictoriaMetrics
Storage Type	- **LocalVolume**: Локальное хранилище с данными на указанных узлах - **StorageClass**: Автоматическое создание persistent volume с использованием storage class - **PV**: Использование существующих persistent volume
Replica Count	Задает количество подов компонента мониторинга: - Тип хранилища LocalVolume не поддерживает несколько реплик

Parameter	Description
	- Для других типов хранилища смотрите подсказки на экране для настройки
Parameter Configuration	Параметры данных для компонента мониторинга можно настроить **Примечание**: Данные могут временно превышать период хранения перед удалением

4. Нажмите **Install** для завершения установки.

Install the ACP Monitoring with VictoriaMetrics Plugin via YAML

1. Check available versions

Убедитесь, что плагин опубликован, проверив наличие ресурсов ModulePlugin и ModuleConfig в кластере `global`:

```
# kubectl get moduleplugin | grep victoriametrics
victoriametrics                        30h
# kubectl get moduleconfig | grep victoriametrics
victoriametrics-v4.1.0                 30h
```

Это означает, что ModulePlugin `victoriametrics` существует в кластере и версия `v4.1.0` опубликована.

2. Create a ModuleInfo

Создайте ресурс ModuleInfo для установки плагина без параметров конфигурации:

```yaml
kind: ModuleInfo
apiVersion: cluster.alauda.io/v1alpha1
metadata:
  name: business-1-victoriametrics
  labels:
    cpaas.io/cluster-name: business-1
    cpaas.io/module-name: victoriametrics
    cpaas.io/module-type: plugin
spec:
  version: v4.1.0
  config:
    storage:
      type: LocalVolume
      capacity: 40
      nodes:
        - xxx.xxx.xxx.xx
      path: /cpaas/monitoring
      storageClass: ""
      pvSelectorK: ""
      pvSelectorV: ""
    replicas: 1
    agentOnly: false
    agentReplicas: 1
    crossClusterDependency:
      victoriametrics: ""
    components:
      nodeExporter:
        port: 9100
        resources: null
      vmstorage:
        retention: 7
        resources: null
      kubeStateExporter:
        resources: null
      vmalert:
        resources: null
      prometheusAdapter:
        resources: null
      vmagent:
        scrapeInterval: 60
        scrapeTimeout: 45
        resources: null
      vminsert:
```

```
        resources: null
      alertmanager:
        resources: null
      vmselect:
        resources: null
    size: Small
  valuesOverride:
    ait/chart-victoriametrics:
      global:
        nodeSelector:
          node-role.kubernetes.io/infra: "true"
        tolerations:
        - effect: NoSchedule
          key: node-role.kubernetes.io/infra
          value: reserved
          operator: Equal
```

Пример настройки ресурсов для vmagent:

```
spec:
  config:
    components:
      vmagent:
        resources:
          limits:
            cpu: 2000m
            memory: 2000Mi
          requests:
            cpu: 1000m
            memory: 1000Mi
```

Подробности смотрите в Monitor Component Capacity Planning

Справочник по полям YAML (VictoriaMetrics):

Field path	Description
`metadata.labels.cpaas.io/cluster-name`	Имя целевого кластера, в котором установлен плагин.

Field path	Description
`metadata.labels.cpaas.io/module-name`	Должно быть `victoriametrics`.
`metadata.labels.cpaas.io/module-type`	Должно быть `plugin`.
`metadata.name`	Имя ModuleInfo (например, `<cluster>-victoriametrics`).
`spec.version`	Версия плагина для установки.
`spec.config.storage.type`	Тип хранилища: `LocalVolume`, `StorageClass` или `PV`.
`spec.config.storage.capacity`	Размер хранилища для VictoriaMetrics (Gi). Рекомендуется минимум 30 Gi.
`spec.config.storage.nodes`	Список узлов при `storage.type=LocalVolume`. Поддерживается до 1 узла.
`spec.config.storage.path`	Путь LocalVolume при `storage.type=LocalVolume`.
`spec.config.storage.storageClass`	Имя StorageClass при `storage.type=StorageClass`.
`spec.config.storage.pvSelectorK`	Ключ селектора PV при `storage.type=PV`.
`spec.config.storage.pvSelectorV`	Значение селектора PV при `storage.type=PV`.
`spec.replicas`	Количество реплик; LV не поддерживает несколько реплик.

Field path	Description
`spec.config.components.vmstorage.retention`	Количество дней хранения данных для vmstorage.
`spec.config.components.vmagent.scrapeInterval`	Интервал сбора данных в секундах; применяется к ServiceMonitors без `interval`.
`spec.config.components.vmagent.scrapeTimeout`	Таймаут сбора данных в секундах; должен быть меньше `scrapeInterval`.
`spec.config.components.vmstorage.resources`	Настройки ресурсов для vmstorage.
`spec.config.components.nodeExporter.port`	Порт Node Exporter (по умолчанию 9100).
`spec.config.components.nodeExporter.resources`	Настройки ресурсов для Node Exporter.
`spec.config.components.alertmanager.resources`	Настройки ресурсов для Alertmanager.
`spec.config.components.kubeStateExporter.resources`	Настройки ресурсов для Kube State Exporter.
`spec.config.components.prometheusAdapter.resources`	Настройки ресурсов для Prometheus Adapter (используется для HPA/ кастомных метрик).
`spec.config.components.vmagent.resources`	Настройки ресурсов для vmagent.
`spec.config.size`	Масштаб мониторинга: `Small`, `Medium` или `Large`.

Field path	Description
`spec.valuesOverride.ait/chart-victoriametrics.global.nodeSelector`	Необязательно. NodeSelector для компонента Monitoring. Используется для выбора узлов, на которых будет работать компонент мониторинга. Обычно применяется для infra-узлов.
`spec.valuesOverride.ait/chart-victoriametrics.global.tolerations`	Необязательно. Tolerations для компонента Monitoring. Если nodeSelector выбирает узлы с taint, используйте этот параметр для указания ключа, значения и эффекта toleration. Обычно применяется для infra-узлов.

3. Verify installation

Так как имя ModuleInfo меняется при создании, найдите ресурс по метке, чтобы проверить статус и версию плагина:

```
kubectl get moduleinfo -l cpaas.io/module-name=victoriametrics
NAME                                             CLUSTER         MODULE
DISPLAY_NAME     STATUS    TARGET_VERSION   CURRENT_VERSION   NEW_VERSION
global-e671599464a5b1717732c5ba36079795          global          victoria
metrics   victoriametrics  Running   v4.1.0           v4.1.0            v
4.1.0
```

Объяснение полей:

- `NAME`: имя ресурса ModuleInfo
- `CLUSTER`: кластер, в котором установлен плагин
- `MODULE`: имя плагина
- `DISPLAY_NAME`: отображаемое имя плагина

- `STATUS`: статус установки; `Running` означает успешную установку и работу
- `TARGET_VERSION`: версия, которая должна быть установлена
- `CURRENT_VERSION`: версия до установки
- `NEW_VERSION`: последняя доступная версия для установки

Архитектура

Архитектура модуля монитор

- Общее описание архитектуры
- Система мониторинга
- Система оповещений
- Система уведомлений

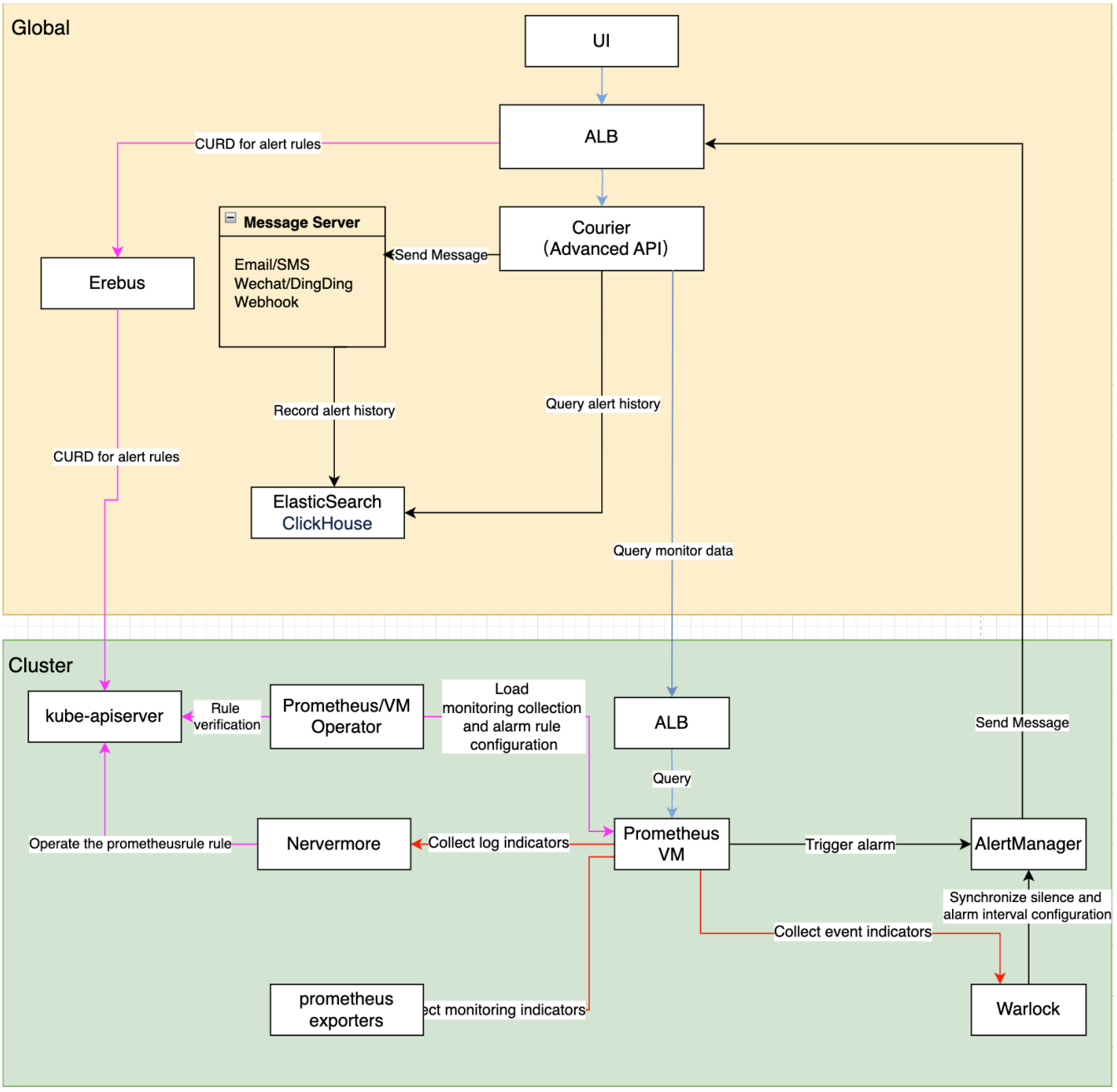
Руководство по выбору комп

- Важные замечания
- Список компонентов
- Сравнение архитектур
- Сравнение функций
- Рекомендации по схемам установки

Планирова

- Предположени
- Prometheus
- VictoriaMetrics

Архитектура модуля мониторинга



Содержание

Общее описание архитектуры

Система мониторинга

Сбор и хранение данных

Запрос и визуализация данных

Система оповещений

Управление правилами оповещений

Рабочий процесс обработки оповещений

Отображение статуса оповещений в реальном времени

Система уведомлений

Управление конфигурацией уведомлений

Управление сервером уведомлений

Общее описание архитектуры

Система мониторинга состоит из следующих основных функциональных модулей:

1. Система мониторинга
   - Сбор и хранение данных: сбор и сохранение метрик мониторинга из различных источников
   - Запрос и визуализация данных: предоставление гибких возможностей для запросов и визуализации данных мониторинга
2. Система оповещений
   - Управление правилами оповещений: настройка и управление политиками оповещений
   - Генерация оповещений и уведомлений: оценка правил оповещений и отправка уведомлений
   - Отображение текущего статуса оповещений в реальном времени
3. Система уведомлений
   - Конфигурация уведомлений: управление шаблонами уведомлений, группами контактов и политиками

- Сервер уведомлений: управление конфигурацией различных каналов уведомлений

Система мониторинга

Сбор и хранение данных

1. Обязанности операторов Prometheus/VictoriaMetrics:
   - Загрузка и валидация конфигураций сбора мониторинга
   - Загрузка и валидация конфигураций правил оповещений
   - Синхронизация конфигураций с инстансами Prometheus/VictoriaMetrics
2. Источники данных мониторинга:
   - Nevermore: генерирует метрики, связанные с логами
   - Warlock: генерирует метрики, связанные с событиями
   - Prometheus/VictoriaMetrics: обнаруживает и собирает метрики различных экспортеров через ServiceMonitor

Запрос и визуализация данных

1. Процесс запроса данных мониторинга:
   - Браузер инициирует запрос (Путь: `/platform/monitoring.alauda.io/v1beta1`)
   - ALB перенаправляет запрос в компонент Courier
   - API Courier обрабатывает запрос:
     - Встроенные метрики: получает PromQL через интерфейс индикаторов и выполняет запрос
     - Пользовательские метрики: напрямую пересылает PromQL в компонент мониторинга
   - Панель мониторинга получает данные и отображает их
2. Процесс управления панелью мониторинга:

- Пользователи обращаются к ALB кластера `global` (Путь: `/kubernetes/cluster_name/apis/ait.alauda.io/v1alpha2/MonitorDashboard`)
- ALB перенаправляет запрос в компонент Erebus
- Erebus маршрутизирует запрос в целевой кластер мониторинга
- Компонент Warlock отвечает за:
  - Проверку легитимности конфигурации панели мониторинга
  - Управление ресурсом MonitorDashboard CR

Система оповещений

Управление правилами оповещений

Процесс конфигурации правил оповещений:

1. Пользователи обращаются к ALB кластера `global` (Путь: `/kubernetes/cluster_name/apis/monitoring.coreos.com/v1/prometheusrules`)
2. Запрос проходит через ALB -> Erebus -> kube-apiserver целевого кластера
3. Обязанности компонентов:
   - Оператор Prometheus/VictoriaMetrics:
     - Проверка легитимности правил оповещений
     - Управление ресурсом PrometheusRule CR
   - Nevermore: прослушивание и обработка метрик оповещений по логам
   - Warlock: прослушивание и обработка метрик оповещений по событиям

Рабочий процесс обработки оповещений

1. Оценка оповещений:
   - Правила оповещений определяются в PrometheusRule/VMRule
   - Prometheus/VictoriaMetrics периодически оценивают правила

2. Уведомление об оповещениях:

   - При срабатывании оповещения отправляются в Alertmanager
   - Alertmanager -> ALB -> API Courier
   - API Courier отвечает за отправку уведомлений

3. Хранение оповещений:

   - История оповещений хранится в ElasticSearch/ClickHouse

Отображение статуса оповещений в реальном времени

1. Сбор статуса:

   - Компонент Courier кластера `global` генерирует метрики:
     - cpaas_active_alerts: текущие активные оповещения
     - cpaas_active_silences: текущие конфигурации тишины
   - Глобальный Prometheus собирает данные каждые 15 секунд

2. Отображение статуса:

   - Фронтенд запрашивает и отображает статус в реальном времени через API Courier

Система уведомлений

Управление конфигурацией уведомлений

Процесс управления шаблонами уведомлений, группами контактов и политиками уведомлений:

1. Пользователи обращаются к стандартному API кластера `global` через браузер

   - Путь доступа: `/apis/ait.alauda.io/v1beta1/namespaces/cpaas-system`

2. Управление соответствующими ресурсами:

- Шаблон уведомления: apiVersion: "ait.alauda.io/v1beta1", kind: "NotificationTemplate"
- Группа контактов уведомлений: apiVersion: "ait.alauda.io/v1beta1", kind: "NotificationGroup"
- Политика уведомлений: apiVersion: "ait.alauda.io/v1beta1", kind: "Notification"

3. Courier отвечает за:

- Проверку легитимности шаблонов уведомлений
- Проверку легитимности групп контактов уведомлений
- Проверку легитимности политик уведомлений

Управление сервером уведомлений

1. Пользователи обращаются к ALB кластера `global` через браузер
   - Путь доступа: `/kubernetes/global/api/v1/namespaces/cpaas-system/secrets`
2. Управление и отправка конфигураций сервера уведомлений
   - Имя ресурса: platform-email-server
3. Courier отвечает за:
   - Проверку легитимности конфигурации сервера уведомлений

Руководство по выбору компонента мониторинга

При установке мониторинга кластера платформа предоставляет два компонента мониторинга на выбор: VictoriaMetrics и Prometheus. В этой статье подробно описаны характеристики и применимые сценарии этих двух компонентов, что поможет вам сделать наиболее подходящий выбор.

Содержание

- Важные замечания
- Список компонентов
  - Компоненты, связанные с Prometheus
  - Компоненты, связанные с VictoriaMetrics
- Сравнение архитектур
  - Архитектура Prometheus
  - Архитектура VictoriaMetrics
- Сравнение функций
- Рекомендации по схемам установки
  - Обзор архитектуры установки мониторинга
    - Метод установки Prometheus
    - Метод установки VictoriaMetrics
  - Рекомендации по выбору
    - Сценарии, подходящие для использования VictoriaMetrics
    - Сценарии, подходящие для использования Prometheus

Важные замечания

- При установке компонентов мониторинга кластера можно выбрать только один из VictoriaMetrics или Prometheus.
- Начиная с версии 3.18, VictoriaMetrics получила статус Beta, что соответствует условиям использования в производственной среде.
- VictoriaMetrics подходит для сценариев с требованиями высокой доступности и мониторинга нескольких кластеров.
- Prometheus подходит для сценариев мониторинга одного кластера, особенно небольшого масштаба.

Список компонентов

Компоненты, связанные с Prometheus

Название компонента	Описание функции
Prometheus Server	Основной сервер, отвечающий за сбор, хранение и запросы мониторинговых данных
Exporters	Компоненты сбора мониторинговых данных, которые предоставляют метрики мониторинга через HTTP-интерфейсы
AlertManager	Центр управления оповещениями, обрабатывающий правила оповещений и уведомления
PushGateway	Поддерживает push-режим для мониторинговых данных, используется для передачи данных в специальных сетевых условиях

Компоненты, связанные с VictoriaMetrics

Название компонента	Описание функции
VMStorage	Движок хранения мониторинговых данных
VMInsert	Компонент записи данных, отвечающий за распределение и хранение данных
VMSelect	Компонент сервиса запросов, предоставляющий возможности запросов данных
VMAlert	Компонент оценки и обработки правил оповещений
VMAgent	Компонент сбора метрик мониторинга

Сравнение архитектур

Архитектура Prometheus

Prometheus — зрелая система мониторинга с открытым исходным кодом и второй проект CNCF, получивший статус graduated после Kubernetes. Имеет следующие характеристики:

- Мощные возможности сбора данных.
- Гибкий язык запросов PromQL.
- Обширная экосистема.
- Поддержка мониторинга кластера масштаба до тысячи узлов.

Архитектура VictoriaMetrics

VictoriaMetrics — высокопроизводительная база данных временных рядов и решение для мониторинга следующего поколения с такими преимуществами:

- Более высокий коэффициент сжатия данных.
- Меньшее потребление ресурсов.

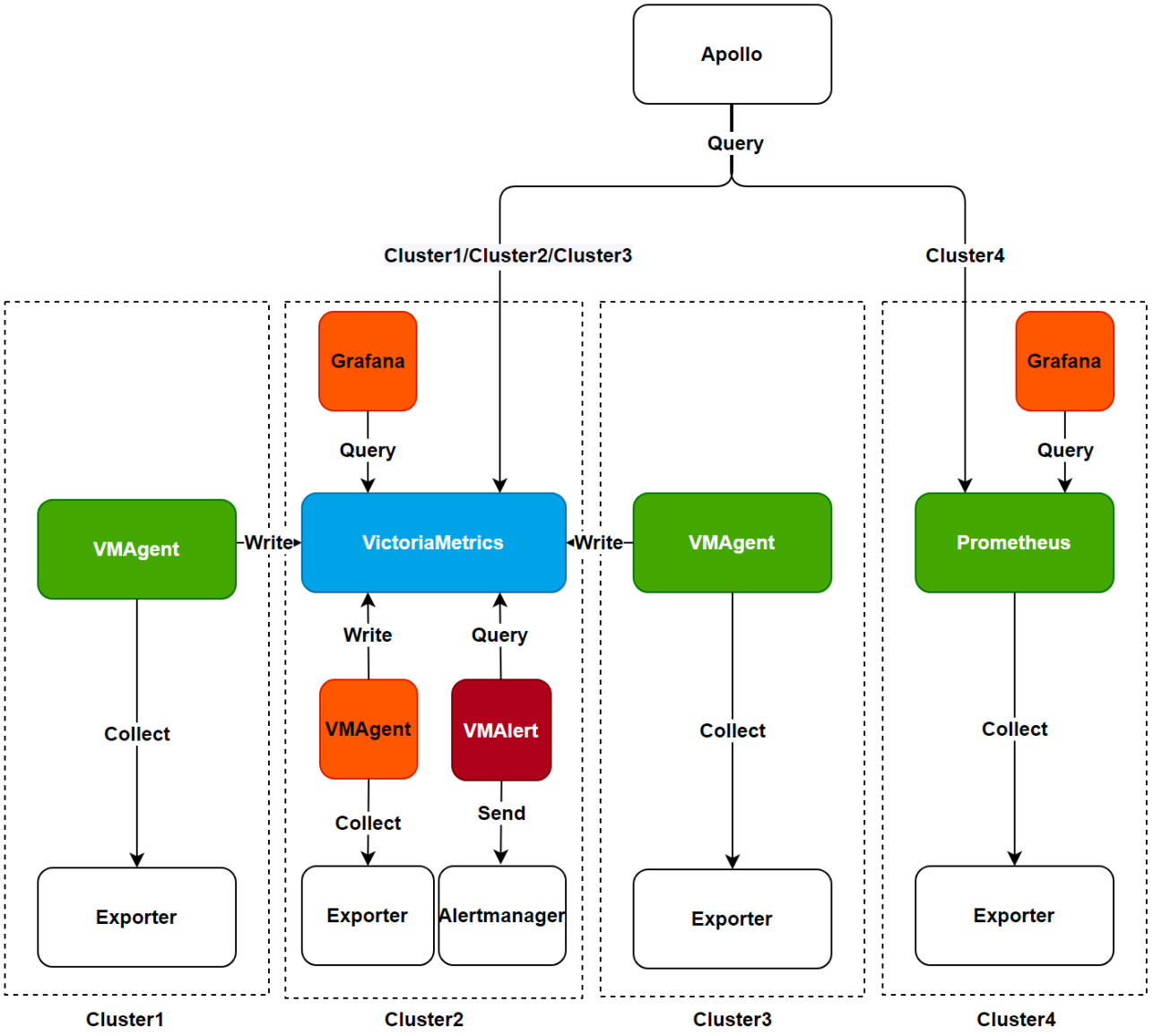
- Родная поддержка высокой доступности кластера.
- Более простое управление эксплуатацией и обслуживанием.

Сравнение функций

Функция	Prometheus	VictoriaMetrics	Описание
Установка с высокой доступностью	❌	✅	VictoriaMetrics поддерживает настоящую кластерную высокую доступность с лучшей согласованностью данных
Установка на одном узле	✅	✅	Оба поддерживают режим установки на одном узле
Долговременное хранение данных	Требуется удалённое хранилище	Поддерживается нативно	VictoriaMetrics более подходит для долговременного хранения данных
Эффективность использования ресурсов	Выше	Лучше	VictoriaMetrics обеспечивает лучшую эффективность использования ресурсов
Поддержка сообщества	Очень зрелая	Быстро развивается	У Prometheus более крупная экосистема сообщества

Рекомендации по схемам установки

Обзор архитектуры установки мониторинга



На приведённой выше схеме показана архитектура установки и поток данных компонентов мониторинга, поддерживаемых платформой. Платформа предоставляет следующие два варианта установки на выбор:

Примечание: При замене компонентов мониторинга убедитесь, что существующие компоненты полностью удалены, и данные мониторинга не поддерживают миграцию между компонентами.

Метод установки Prometheus

Этот метод соответствует архитектуре **cluster4** на схеме выше:

- Используются компоненты Prometheus для сбора и обработки мониторинговых данных.
- Запросы и отображение данных осуществляются через панель мониторинга.

- Подходит для сценариев с одним кластером.

Метод установки VictoriaMetrics

VictoriaMetrics поддерживает два режима установки:

1. **Режим установки в одном кластере**
   - Соответствует архитектуре **cluster2** на схеме выше.
   - Все компоненты VictoriaMetrics устанавливаются в одном кластере.
   - Для сбора данных используется VMAgent, который записывает данные в VictoriaMetrics.
   - VMAlert отвечает за оценку правил оповещений.
   - Запросы и отображение данных осуществляются через панель мониторинга. **Совет**: Рекомендуется использовать этот режим при объёме данных менее 1 миллиона в секунду.
2. **Режим установки для нескольких кластеров**
   - Соответствует архитектурам **cluster1/cluster2/cluster3** на схеме выше.
   - VMAgent устанавливается в рабочем кластере в качестве агента сбора данных.
   - VMAgent записывает данные в VictoriaMetrics в центральном кластере мониторинга.
   - Поддерживается единое управление мониторингом нескольких кластеров. **Совет**: Перед установкой VMAgent убедитесь, что сервисы VictoriaMetrics установлены в кластере мониторинга.

Рекомендации по выбору

Сценарии, подходящие для использования VictoriaMetrics

- **Требования к высокой производительности и масштабируемости**: Подходит для сценариев мониторинга с высокими объёмами данных и долговременным хранением.
- **Оптимизация затрат**: Необходимость оптимизировать затраты на хранение и вычислительные ресурсы.

- **Требования к высокой доступности**: Необходима гарантия высокой доступности компонентов мониторинга.
- **Управление несколькими кластерами**: Требуется единое управление данными мониторинга нескольких кластеров.

Сценарии, подходящие для использования Prometheus

- **Один кластер малого масштаба**: Небольшой масштаб мониторинга без требований к высокой доступности.
- **Пользователи с уже существующим Prometheus**: Имеется полноценная система мониторинга на базе Prometheus.
- **Простые требования к стабильности**: Предпочтение простому и надёжному решению мониторинга.
- **Глубокая интеграция с экосистемой**: Тесная интеграция с экосистемой Prometheus и высокие затраты на миграцию.

Планирование ёмкости компонента мониторинга

Компонент мониторинга отвечает за хранение данных метрик, собранных с одного или нескольких кластеров в платформе. Поэтому необходимо заранее оценить масштаб вашего мониторинга и спланировать ресурсы, необходимые для компонента мониторинга, согласно рекомендациям в этом документе.

Содержание

- Предположения и методология
  - Prometheus
    - Малый масштаб — 10 рабочих узлов, 500 подов с двумя контейнерами
    - Средний масштаб — 50 рабочих узлов, 2000 подов с двумя контейнерами
    - Большой масштаб — 500 рабочих узлов, 10000 подов с двумя контейнерами
  - VictoriaMetrics
    - Малый масштаб — 10 рабочих узлов, 500 подов с двумя контейнерами
    - Средний масштаб — 50 рабочих узлов, 2000 подов с двумя контейнерами
    - Большой масштаб — 500 рабочих узлов, 10000 подов с двумя контейнерами

Предположения и методология

- Данные в этом документе получены из контролируемых лабораторных отчетов о производительности и предназначены в качестве базового ориентира для

планирования в продакшене.

- В примерах с дисковым хранением время хранения составляет 7 дней; для других целей хранения корректируйте пропорционально.
- Базовые параметры хранения соответствуют приведенному выше предупреждению (SSD, ~6000 IOPS, ~250MB/s чтение/запись, отдельный монтируемый том).
- Тестовые нагрузки охватывали типичные страницы мониторинга, такие как "acp ns overview page" и "platform region detail page".

Prometheus

Ниже приведены рекомендации по масштабированию для Prometheus и связанных компонентов (Thanos Query, Thanos Sidecar и др.).

Малый масштаб — 10 рабочих узлов, 500 подов с двумя контейнерами

- Скорость приема метрик: ~2800 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	Пр
courier-api	courier	2	2C	4Gi	-	-
kube-prometheus-thanos-query	thanos-query	1	1C	1Gi	-	-
prometheus-kube-prometheus-0	prometheus	1	2C	8Gi	20G	~1 за

Средний масштаб — 50 рабочих узлов, 2000 подов с двумя контейнерами

- Скорость приема метрик: ~7294 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	Пр
courier-api	courier	2	4C	4Gi	-	-
kube-prometheus-thanos-query	thanos-query	1	2.5C	8Gi	-	-
prometheus-kube-prometheus-0	prometheus	1	4C	8Gi	40G	~3 за

Большой масштаб — 500 рабочих узлов, 10000 подов с двумя контейнерами

- Скорость приема метрик: ~41575 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	Пр
courier-api	courier	2	6C	4Gi	-	-
kube-prometheus-thanos-query	thanos-query	1	2C	6Gi	-	В р ра мо ис 2 р
prometheus-kube-	prometheus	1	8C	20Gi	100G	Пи ~1

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	Пр
prometheus-0						заг дн

VictoriaMetrics

Ниже приведены рекомендации по масштабированию для компонентов VictoriaMetrics.

Малый масштаб — 10 рабочих узлов, 500 подов с двумя контейнерами

- Скорость приема метрик: ~3274 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	Пр
courier-api	courier	1	2C	4Gi	-	-
vmselect-cluster	proxy	1	1C	200Mi	-	-
vmselect	vmselect	1	500m	1Gi	-	-
vmstorage-cluster	vmstorage	1	500m	2Gi	3G	~1. заг дне

Средний масштаб — 50 рабочих узлов, 2000 подов с двумя контейнерами

- Скорость приема метрик: ~6940 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	Пр
courier-api	courier	2	4C	4Gi	-	-
vmselect-cluster	proxy	1	1C	200Mi	-	-
vmselect	vmselect	1	2C	2Gi	-	-
vmstorage-cluster	vmstorage	1	2C	2Gi	10G	~2. зап дне

Большой масштаб — 500 рабочих узлов, 10000 подов с двумя контейнерами

- Скорость приема метрик: ~34300 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	Пр
courier-api	courier	2	6C	4Gi	-	-
vmselect-cluster	proxy	1	2C	200Mi	-	-
vmselect	vmselect	1	5C	3Gi	-	-
vmstorage-cluster	vmstorage	1	2C	6Gi	30G	~16 зап дне

Основные понятия

Содержание

- Monitoring
  - Metrics
  - PromQL
  - Built-in Indicators
  - Exporter
  - ServiceMonitor
- Alarms
  - Alarm Rules
  - Alarm Policies
- Notifications
  - Notification Policies
  - Notification Templates
- Monitoring Dashboard
  - Dashboard
  - Panels
  - Data Sources
  - Variables

Monitoring

Metrics

Метрики используются для количественного описания состояния работы системы, и каждая метрика состоит из четырёх основных элементов:

- Metric Name: используется для идентификации объекта мониторинга, например, `cpu_usage`
- Metric Value: конкретное измеренное значение, например, `85.5`
- Timestamp: фиксирует время измерения
- Labels: используются для многомерной классификации данных, например, `{pod="nginx-1", namespace="default"}`

PromQL

PromQL — это язык запросов для Prometheus, используемый для запроса и агрегации метрик из системы мониторинга.

Built-in Indicators

Платформа имеет предустановленный набор часто используемых метрик мониторинга на основе многолетнего опыта эксплуатации. Вы можете использовать эти метрики напрямую при настройке правил оповещений или создании дашбордов без дополнительной конфигурации.

Exporter

Exporter — это компонент для сбора данных мониторинга, основные задачи которого включают:

- Сбор исходных данных мониторинга с целевой системы
- Преобразование данных в стандартный формат временных рядов метрик
- Предоставление метрик для запросов через HTTP-интерфейс

ServiceMonitor

ServiceMonitor используется для декларативного управления конфигурациями мониторинга и в основном определяет:

- Критерии выбора целей мониторинга
- Конфигурацию интерфейсов сбора метрик
- Параметры выполнения задач сбора (интервалы, таймауты и т.д.)

Alarms

Alarm Rules

Правила оповещений определяют конкретные условия срабатывания тревог:

- Alarm Expression: описание условий срабатывания с помощью выражений PromQL
- Alarm Threshold: явные пороговые значения для срабатывания
- Duration: продолжительность, в течение которой условия должны непрерывно выполняться
- Alarm Level: различение степени серьёзности тревог (например, P0/P1/P2)

Alarm Policies

Политики оповещений объединяют несколько правил оповещений для единой настройки:

- Alarm Targets: целевой охват правил
- Notification Method: каналы отправки оповещений
- Sending Interval: интервал повторной отправки оповещений

Notifications

Notification Policies

Политики уведомлений управляют правилами отправки сообщений об оповещениях:

- Recipients: целевые пользователи для уведомлений
- Notification Channels: поддерживаемые методы отправки сообщений
- Notification Templates: определение формата содержимого сообщений

Notification Templates

Шаблоны уведомлений настраивают формат отображения сообщений об оповещениях:

- Title Template: формат заголовка сообщения об оповещении
- Content Template: организация деталей оповещения
- Variable Replacement: поддержка динамического заполнения данными

Monitoring Dashboard

Dashboard

Дашборд — это набор нескольких связанных панелей, предоставляющий общий обзор состояния системы. Поддерживает гибкое расположение и может организовывать панели в строки или столбцы.

Panels

Панели — визуальное представление данных мониторинга, поддерживающее различные типы отображения.

Data Sources

Конфигурация источников данных мониторинга. В настоящее время поддерживаются только компоненты мониторинга текущего кластера, кастомные источники данных пока не поддерживаются.

Variables

Переменные служат заполнителями значений и могут использоваться в запросах метрик. Через селектор переменных в верхней части дашборда можно динамически изменять условия запросов, что позволяет обновлять содержимое графиков в реальном времени.

Руководства

Управление метриками

- Просмотр метрик, предоставляемых ко
- Просмотр всех метрик, хранящихся в Pr
- Просмотр всех встроенных метрик, опре
- Интеграция внешних метрик

Управление оповещениями

- Обзор функции
- Ключевые возможности
- Преимущества функции
- Создание политик оповещений через UI
- Создание оповещений по ресурсам чер
- Создание оповещений по событиям чер
- Создание политик оповещений через al
- Настройка заглушения оповещений
- Рекомендации по настройке правил оповещений

Управлени

- Обзор функци
- Ключевые фун
- Notification Ser
- Notification Cor
- Notification Tem
- Notification rule
- Настройка пра

Управление Probe

- Обзор функции
- Blackbox мониторинг
- Оповещения Blackbox
- Настройка модуля мониторинга BlackboxExporter
- Создание элементов Blackbox мониторинга и оповещений через CLI
- Справочная информация

ни

- кци
- е па
- е па
- ıане
- нкци

Управление метриками

Система мониторинга платформы основана на метриках, собираемых Prometheus / VictoriaMetrics. В этом документе описано, как управлять этими метриками.

Содержание

- Просмотр метрик, предоставляемых компонентами платформы
- Просмотр всех метрик, хранящихся в Prometheus / VictoriaMetrics
  - Предварительные требования
  - Процедуры
- Просмотр всех встроенных метрик, определённых платформой
  - Предварительные требования
  - Процедуры
- Интеграция внешних метрик
  - Предварительные требования
  - Процедуры

Просмотр метрик, предоставляемых компонентами платформы

Метод мониторинга компонентов кластера в платформе заключается в извлечении метрик, предоставляемых через `ServiceMonitor`. Метрики в платформе доступны

публично через эндпоинт `/metrics`. Вы можете просмотреть метрики конкретного компонента платформы, используя следующий пример команды:

```
curl -s http://<Component IP>:<Component metrics port>/metrics | grep 'TY
PE\|HELP'
```

Пример вывода:

```
# HELP controller_runtime_active_workers Number of currently used workers
per controller
# TYPE controller_runtime_active_workers gauge
# HELP controller_runtime_max_concurrent_reconciles Maximum number of con
current reconciles per controller
# TYPE controller_runtime_max_concurrent_reconciles gauge
# HELP controller_runtime_reconcile_errors_total Total number of reconcil
iation errors per controller
# TYPE controller_runtime_reconcile_errors_total counter
# HELP controller_runtime_reconcile_time_seconds Length of time per recon
ciliation per controller
```

Просмотр всех метрик, хранящихся в Prometheus / VictoriaMetrics

Вы можете просмотреть список доступных метрик в кластере, чтобы на их основе написать необходимый PromQL.

Предварительные требования

1. У вас есть ваш пользовательский Token
2. У вас есть адрес платформы

Процедуры

> Выполните следующую команду для получения списка метрик с помощью `curl`:

```
curl -k -X 'GET' -H 'Authorization: Bearer <Your token>' 'https://<Yo
ur platform access address>/v2/metrics/<Your cluster name>/prometheu
s/label/__name__/values'
```

Пример вывода:

```
{
  "status": "success",
  "data": [
    "ALERTS",
  "ALERTS_FOR_STATE",
  "advanced_search_cached_resources_count",
  "alb_error",
  "alertmanager_alerts",
  "alertmanager_alerts_invalid_total",
  "alertmanager_alerts_received_total",
  "alertmanager_cluster_enabled"]
}
```

Просмотр всех встроенных метрик, определённых платформой

Для упрощения использования пользователями платформа встроила большое количество часто используемых метрик. Вы можете напрямую использовать эти метрики при настройке оповещений или панелей мониторинга, не определяя их самостоятельно. Ниже описано, как просмотреть эти метрики.

Предварительные требования

1. У вас есть ваш пользовательский Token
2. У вас есть адрес платформы

Процедуры

Выполните следующую команду для получения списка метрик с помощью `curl`:

```
curl -k -X 'GET' -H 'Authorization: Bearer <Your token>' 'https://<Your platform access address>/v2/metrics/<Your cluster name>/indicators'
```

Пример вывода:

```
[
  {
  "alertEnabled": true, (1)
  "annotations": {
   "cn": "Использование CPU контейнерами в компоненте вычислений",
   "descriptionEN": "Cpu utilization for pods in workload",
   "descriptionZH": "CPU utilization of containers in the compute com
ponent",
   "displayNameEN": "CPU utilization of the pods",
   "displayNameZH": "CPU utilization of containers in the compute com
ponent",
   "en": "Cpu utilization for pods in workload",
   "features": "SupportDashboard", (2)
   "summaryEN": "CPU usage rate {{.externalLabels.comparison}}{{.exte
rnalLabels.threshold}} of Pod ({{.labels.pod}})",
   "summaryZH": "CPU usage rate {{.externalLabels.comparison}}{{.exte
rnalLabels.threshold}} of pod ({{.labels.pod}})"
  },
  "displayName": "Использование CPU контейнерами в компоненте вычисле
ний",
  "kind": "workload",
  "multipleEnabled": true,  (3)
  "name": "workload.pod.cpu.utilization",
  "query": "avg by (kind,name,namespace,pod) (avg by (kind,name,names
pace,pod,container)(cpaas_advanced_container_cpu_usage_seconds_total_
irate5m{kind=~\"{{.kind}}\",name=~\"{{.name}}\",namespace=~\"{{.names
pace}}\",container!=\"\",container!=\"POD\"}) / avg by (kind,name,nam
espace,pod,container)(cpaas_advanced_kube_pod_container_resource_limi
ts{kind=~\"{{.kind}}\",name=~\"{{.name}}\",namespace=~\"{{.namespac
e}}\",resource=\"cpu\"}))", (4)
  "summary": "CPU usage rate {{.externalLabels.comparison}}{{.externa
lLabels.threshold}} of pod ({{.labels.pod}})",
  "type": "metric",
  "unit": "%",
  "legend": "{{.namespace}}/{{.pod}}",
  "variables": [ (5)
   "namespace",
   "name",
   "kind"
  ]
 }
]
```

1. Поддерживает ли эта метрика использование для настройки оповещений
2. Поддерживает ли эта метрика использование в панелях мониторинга
3. Поддерживает ли эта метрика использование при настройке оповещений для нескольких ресурсов
4. Определённое для метрики выражение PromQL
5. Переменные, которые можно использовать в выражении PromQL метрики

Интеграция внешних метрик

Помимо встроенных метрик платформы, вы также можете интегрировать метрики, предоставляемые вашими приложениями или сторонними приложениями через `ServiceMonitor` или `PodMonitor`. В этом разделе в качестве примера используется Elasticsearch Exporter, установленный в виде pod в том же кластере.

Предварительные требования

Вы установили ваше приложение и открыли метрики через указанные интерфейсы. В этом документе предполагается, что ваше приложение установлено в namespace `cpaas-system` и предоставляет эндпоинт `http://<elasticsearch-exporter-ip>:9200/_prometheus/metrics`.

Процедуры

1. Создайте Service/Endpoint для Exporter, чтобы открыть метрики

```
apiVersion: v1
kind: Service
metadata:
  labels:
    chart: elasticsearch
    service_name: cpaas-elasticsearch
  name: cpaas-elasticsearch
  namespace: cpaas-system
spec:
  clusterIP: 10.105.125.99
  ports:
  - name: cpaas-elasticsearch
    port: 9200
    protocol: TCP
    targetPort: 9200
  selector:
    service_name: cpaas-elasticsearch
  sessionAffinity: None
  type: ClusterIP
```

2. Создайте объект `ServiceMonitor` для описания метрик, предоставляемых вашим приложением:

```yaml
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    app: cpaas-monitor
    chart: cpaas-monitor
    heritage: Helm
    prometheus: kube-prometheus (1)
    release: cpaas-monitor
  name: cpaas-elasticsearch-Exporter
  namespace: cpaas-system (2)
spec:
  jobLabel: service_name (3)
  namespaceSelector: (4)
    any: true
  selector: (5)
    matchExpressions:
      - key: service_name
        operator: Exists
  endpoints:
    - port: cpaas-elasticsearch (6)
      path: /_prometheus/metrics (7)
      interval: 60s (8)
      honorLabels: true
      basicAuth: (9)
        password:
          key: ES_PASSWORD
          name: acp-config-secret
        username:
          key: ES_USER
          name: acp-config-secret
```

(1) К какому Prometheus должен быть синхронизирован ServiceMonitor; оператор будет слушать соответствующий ресурс ServiceMonitor на основе конфигурации serviceMonitorSelector в Prometheus CR. Если метки ServiceMonitor не совпадают с конфигурацией serviceMonitorSelector в Prometheus CR, этот ServiceMonitor не будет мониториться оператором.

(2) В каких namespace оператор будет слушать ServiceMonitor на основе конфигурации serviceMonitorNamespaceSelector в Prometheus CR; если

ServiceMonitor не находится в serviceMonitorNamespaceSelector Prometheus CR, этот ServiceMonitor не будет мониториться оператором.

3 Метрики, собираемые Prometheus, будут иметь метку job, значение которой соответствует значению метки сервиса, указанной в jobLabel.

4 ServiceMonitor сопоставляется с соответствующим Service на основе конфигурации namespaceSelector.

5 ServiceMonitor сопоставляется с Service на основе конфигурации selector.

6 ServiceMonitor сопоставляется с портом Service на основе конфигурации port.

7 Путь доступа к Exporter, по умолчанию /metrics.

8 Интервал, с которым Prometheus опрашивает метрики Exporter.

9 Если для доступа к пути Exporter требуется аутентификация, необходимо добавить данные аутентификации; также поддерживается bearer token, tls-аутентификация и другие методы.

3. Проверьте, мониторится ли ServiceMonitor Prometheus
   Зайдите в UI компонента мониторинга и проверьте, существует ли задача `cpaas-elasticsearch-exporter`.

   - Адрес UI Prometheus: `https://<Your platform access address>/clusters/<Cluster name>/prometheus-0/targets`
   - Адрес UI VictoriaMetrics: `https://<Your platform access address>/clusters/<Cluster name>/vmselect/vmui/?#/metrics`

Управление оповещениями

Содержание

- Обзор функции
- Ключевые возможности
- Преимущества функции
- Создание политик оповещений через UI
  - Предварительные условия
  - Процедура
    - Выбор типа оповещения
    - Настройка правил оповещений
    - Другие настройки
  - Дополнительные замечания
- Создание оповещений по ресурсам через CLI
  - Предварительные условия
  - Процедура
- Создание оповещений по событиям через CLI
  - Предварительные условия
  - Процедура
- Создание политик оповещений через alert Templates
  - Предварительные условия
  - Процедура
    - Создание шаблона оповещений
    - Создание политик оповещений с использованием alert Templates
- Настройка заглушения оповещений

Настройка через UI

Настройка через CLI

Рекомендации по настройке правил оповещений

Обзор функции

Функция управления оповещениями платформы предназначена для помощи пользователям в комплексном мониторинге и своевременном обнаружении аномалий системы. Используя предустановленные системные оповещения и гибкие возможности создания пользовательских оповещений, в сочетании со стандартизированными шаблонами оповещений и многоуровневым механизмом управления, она предоставляет полное решение по оповещениям для операционного и технического персонала.

Будь то администраторы платформы или бизнес-пользователи, они могут удобно настраивать и управлять политиками оповещений в пределах своих полномочий для эффективного мониторинга ресурсов платформы.

Ключевые возможности

- **Встроенные системные политики оповещений**: Предустановлены богатые правила оповещений на основе типовых идей диагностики сбоев для кластеров `global` и рабочих кластеров.
- **Пользовательские правила оповещений**: Поддерживается создание правил оповещений на основе различных источников данных, включая предустановленные показатели мониторинга, пользовательские показатели, black-box мониторинг, данные логов платформы и события платформы.
- **Управление шаблонами оповещений**: Поддерживается создание и управление стандартизированными шаблонами оповещений для быстрого применения к похожим ресурсам.
- **Интеграция уведомлений об оповещениях**: Поддерживается отправка информации об оповещениях операционному и техническому персоналу через различные каналы.

- **Изоляция просмотра оповещений**: Разграничение оповещений управления платформой и бизнес-оповещений, что обеспечивает фокусировку персонала с разными ролями на соответствующей информации.
- **Просмотр оповещений в реальном времени**: Предоставляет оповещения в реальном времени с концентрированным отображением количества ресурсов, находящихся в состоянии оповещения, и подробной информацией об оповещениях.
- **Просмотр истории оповещений**: Поддерживается просмотр исторических записей оповещений за определённый период, что облегчает анализ последних состояний мониторинга операционным персоналом и администраторами.

Преимущества функции

- **Всестороннее покрытие мониторинга**: Поддерживается мониторинг различных типов ресурсов, таких как кластеры, узлы и вычислительные компоненты, а также предоставляются богатые встроенные системные политики оповещений, которые можно использовать без дополнительной настройки.
- **Эффективное управление оповещениями**: Стандартизированные конфигурации через шаблоны оповещений повышают эффективность работы, а разделение видов просмотра оповещений облегчает персоналу с разными ролями быстро находить релевантные оповещения.
- **Своевременное обнаружение проблем**: Уведомления об оповещениях автоматически запускаются для обеспечения своевременного обнаружения проблем, поддерживается многоканальная отправка оповещений для проактивного предотвращения проблем.
- **Надёжное управление правами доступа**: Строгий контроль доступа к политикам оповещений гарантирует безопасность и управляемость информации об оповещениях.

Создание политик оповещений через UI

Предварительные условия

- Настроена политика уведомлений (если требуется автоматическая отправка оповещений).
- В целевом кластере установлены компоненты мониторинга (требуется при создании политик оповещений с использованием показателей мониторинга).
- В целевом кластере установлены компоненты хранения логов и сбора логов (требуется при создании политик оповещений с использованием логов и событий).

Процедура

1. Перейдите в **Центр эксплуатации и обслуживания** > **alerts** > **alert Policies**.
2. Нажмите **Create Alert Policy**.
3. Настройте основную информацию.

Выбор типа оповещения

Оповещение по ресурсу

- Типы оповещений классифицируются по типу ресурса (например, состояние deployment в namespace).
- Описание выбора ресурса:
  - По умолчанию "Any", если параметр не выбран, поддерживается автоматическое связывание с новыми ресурсами.
  - При выборе "Select All" применяется только к текущему ресурсу.
  - При выборе нескольких namespace имена ресурсов поддерживают регулярные выражения (например, `cert.*`).

Оповещение по событию

- Типы оповещений классифицируются по конкретным событиям (например, аномальное состояние Pod).
- По умолчанию выбираются все ресурсы под указанным ресурсом и поддерживается автоматическое связывание с новыми ресурсами.

Настройка правил оповещений

Нажмите **Add Alert Rule** и настройте параметры в зависимости от типа оповещения:

Параметры оповещений по ресурсу

Параметр	Описание
Expression	Алгоритм метрики мониторинга в формате Prometheus, например, `rate(node_network_receive_bytes{instance="$server",device!~"lo"}[5m])`
Metric Unit	Единица измерения пользовательской метрики мониторинга, можно ввести вручную или выбрать из предустановленных единиц платформы
Legend Parameter	Управляет именем, соответствующим кривой на графике, формат `{{.LabelName}}`, например, `{{.hostname}}`
Time Range	Временное окно для запросов логов/событий
Log Content	Поля запроса для содержимого логов (например, Error), несколько полей связаны через OR
Event Reason	Поля запроса для причин событий (Reason, например, BackOff, Pulling, Failed и т.д.), несколько полей связаны через OR
Trigger Condition	Условие, состоящее из операторов сравнения, порогов оповещения и длительности (опционально). Определяет, срабатывает ли оповещение на основе сравнения текущих значений/количества логов/событий с порогом оповещения, а также длительности нахождения значений в пределах порога.
alert Level	Делится на четыре уровня: Critical, Serious, Warning и Info. Можно установить разумный уровень оповещения в зависимости от влияния правил оповещений на бизнес для соответствующих ресурсов.

Параметры оповещений по событию

Параметр	Описание
Time Range	Временное окно для запросов событий

Параметр	Описание
Event Monitoring Item	Поддерживает мониторинг уровней событий или причин событий, несколько полей связаны через OR
Trigger Condition	Основано на количестве событий для оценки сравнения
alert Level	Определение аналогично уровням оповещений по ресурсу

Другие настройки

1. Выберите одну или несколько созданных политик уведомлений.
2. Настройте интервалы отправки оповещений.
   - Global: Использовать конфигурацию по умолчанию платформы.
   - Custom: Можно задать разные интервалы отправки в зависимости от уровней оповещений.
   - При выборе "Do Not Repeat" уведомления отправляются только при срабатывании и восстановлении оповещения.

Дополнительные замечания

1. В опции "More" правила оповещения можно задать labels и annotations.
2. Пожалуйста, обратитесь к Prometheus Alerting Rules Documentation↗ для настройки labels и annotations.
3. Внимание: не используйте переменную `$value` в labels, так как это может вызвать исключения оповещений.

Создание оповещений по ресурсам через CLI

Предварительные условия

- Настроена политика уведомлений (если требуется автоматическая отправка оповещений).

- В целевом кластере установлены компоненты мониторинга (требуется при создании политик оповещений с использованием показателей мониторинга).
- В целевом кластере установлены компоненты хранения логов и сбора логов (требуется при создании политик оповещений с использованием логов и событий).

Процедура

1. Создайте новый YAML-файл конфигурации с именем `example-alerting-rule.yaml`.
2. Добавьте ресурсы PrometheusRule в YAML-файл и отправьте его. Следующий пример создаёт новую политику оповещений с именем policy:

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  annotations:
    alert.cpaas.io/cluster: global # Имя кластера, где расположено опов
ещение
    alert.cpaas.io/kind: Cluster # Тип ресурса,
    alert.cpaas.io/name: global # Объект ресурса, поддерживает одиночны
й, множественный (через |) или любой (.*)
    alert.cpaas.io/namespace: cpaas-system # Namespace, где расположен
объект оповещения, поддерживает одиночный, множественный (через |) или
любой (.*)
    alert.cpaas.io/notifications: '["test"]'
    alert.cpaas.io/repeat-config: '{"Critical":"never","High":"5m","Med
ium":"5m","Low":"5m"}'
    alert.cpaas.io/rules.description: '{}'
    alert.cpaas.io/rules.disabled: '[]'
    alert.cpaas.io/subkind: ''
    cpaas.io/description: ''
    cpaas.io/display-name: policy # Отображаемое имя политики оповещени
й
  labels:
    alert.cpaas.io/owner: System
    alert.cpaas.io/project: cpaas-system
    cpaas.io/source: Platform
    prometheus: kube-prometheus
    rule.cpaas.io/cluster: global
    rule.cpaas.io/name: policy
    rule.cpaas.io/namespace: cpaas-system
  name: policy
  namespace: cpaas-system
spec:
  groups:
    - name: general # имя правила оповещения
      rules:
        - alert: cluster.pod.status.phase.not.running-tx1ob-e998f0b9485
4ee1eade5ae79279e005a
          annotations:
            alert_current_value: '{{ $value }}' # Уведомление о текущем
значении, оставить по умолчанию
          expr: (count(min by(pod)(kube_pod_container_status_ready{}) !
=1) or on() vector(0))>2
          for: 30s # Длительность
```

```
        labels:
          alert_cluster: global # Имя кластера, где расположено опове
щение
          alert_for: 30s # Длительность
          alert_indicator: cluster.pod.status.phase.not.running # Имя
индикатора правила оповещения (пользовательское имя индикатора)
          alert_indicator_aggregate_range: '30' # Время агрегации пра
вила оповещения в секундах
          alert_indicator_blackbox_name: '' # Имя black-box мониторин
га
          alert_indicator_comparison: '>' # Метод сравнения правила о
повещения
          alert_indicator_query: '' # Запрос логов правила оповещения
(только для лог-оповещений)
          alert_indicator_threshold: '2' # Порог правила оповещения
          alert_indicator_unit: '' # Единица измерения индикатора пра
вила оповещения
          alert_involved_object_kind: Cluster # Тип объекта, к которо
му относится правило оповещения: Cluster|Node|Deployment|Daemonset|Stat
efulset|Middleware|Microservice|Storage|VirtualMachine
          alert_involved_object_name: global # Имя объекта, к котором
у относится правило оповещения
          alert_involved_object_namespace: '' # Namespace объекта, к
которому относится правило оповещения
          alert_name: cluster.pod.status.phase.not.running-tx1ob # Им
я правила оповещения
          alert_namespace: cpaas-system # Namespace, где расположено
правило оповещения
          alert_project: cpaas-system # Имя проекта объекта, к которо
му относится правило оповещения
          alert_resource: policy # Имя политики оповещений, где распо
ложено правило
          alert_source: Platform # Тип данных политики оповещений: Pl
atform - данные платформы, Business - бизнес-данные
          severity: High # Уровень серьёзности правила оповещения: Cr
itical - критический, High - серьёзный, Medium - предупреждение, Low -
информация
```

Создание оповещений по событиям через CLI

Предварительные условия

- Настроена политика уведомлений (если требуется автоматическая отправка оповещений).
- В целевом кластере установлены компоненты мониторинга (требуется при создании политик оповещений с использованием показателей мониторинга).
- В целевом кластере установлены компоненты хранения логов и сбора логов (требуется при создании политик оповещений с использованием логов и событий).

Процедура

1. Создайте новый YAML-файл конфигурации с именем `example-alerting-rule.yaml`.
2. Добавьте ресурсы PrometheusRule в YAML-файл и отправьте его. Следующий пример создаёт новую политику оповещений с именем policy2:

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  annotations:
    alert.cpaas.io/cluster: global
    alert.cpaas.io/events.scope:
      '[{"names":["argocd-gitops-redis-ha-haproxy"],"kind":"Deployment","operator":"=","namespaces":["*"]}]'
      # names: Имя ресурса для оповещения по событию; оператор не действует, если имя пустое.
      # kind: Тип ресурса, вызывающего оповещение по событию.
      # namespace: Namespace, к которому принадлежит ресурс, вызывающий оповещение. Пустой массив означает ресурс без namespace; при ns ['*'] — все namespace.
      # operator: Селектор =, !=, =~, !~
    alert.cpaas.io/kind: Event # Тип оповещения, Event (оповещение по событию)
    alert.cpaas.io/name: '' # Используется для оповещений по ресурсам; для оповещений по событиям остаётся пустым
    alert.cpaas.io/namespace: cpaas-system
    alert.cpaas.io/notifications: '["acp-qwtest"]'
    alert.cpaas.io/repeat-config: '{"Critical":"never","High":"5m","Medium":"5m","Low":"5m"}'
    alert.cpaas.io/rules.description: '{}'
    alert.cpaas.io/rules.disabled: '[]'
    cpaas.io/description: ''
    cpaas.io/display-name: policy2
  labels:
    alert.cpaas.io/owner: System
    alert.cpaas.io/project: cpaas-system
    cpaas.io/source: Platform
    prometheus: kube-prometheus
    rule.cpaas.io/cluster: global
    rule.cpaas.io/name: policy2
    rule.cpaas.io/namespace: cpaas-system
  name: policy2
  namespace: cpaas-system
spec:
  groups:
    - name: general
      rules:
        - alert: cluster.event.count-6sial-34c9a378e3b6dda8401c2d728994ce2f
```

```
          # 6sial-34c9a378e3b6dda8401c2d728994ce2f можно настроить для
обеспечения уникальности
          annotations:
            alert_current_value: '{{ $value }}' # Уведомление о текущем
значении, оставить по умолчанию
          expr: round(((avg
            by(kind,namespace,name,reason)(increase(cpaas_event_count{n
amespace=~".*",id="policy2-cluster.event.count-6sial"}[300s])))
            + (avg
            by(kind,namespace,name,reason)(abs(increase(cpaas_event_cou
nt{namespace=~".*",id="policy2-cluster.event.count-6sial"}[300s]))))))
            / 2)>2
          # id в policy2 должен быть именем политики оповещений; 6sial
должно совпадать с предыдущим именем правила оповещения
          for: 15s # Длительность
          labels:
            alert_cluster: global # Имя кластера, где расположено опове
щение
            alert_for: 15s # Длительность
            alert_indicator: cluster.event.count # Имя индикатора прави
ла оповещения (пользовательское имя индикатора)
            alert_indicator_aggregate_range: '300' # Время агрегации пр
авила оповещения в секундах
            alert_indicator_blackbox_name: ''
            alert_indicator_comparison: '>' # Метод сравнения правила о
повещения
            alert_indicator_event_reason: ScalingReplicaSet # Причина с
обытия
            alert_indicator_threshold: '2' # Порог правила оповещения
            alert_indicator_unit: pieces # Единица измерения индикатора
правила оповещения; для оповещений по событиям не меняется
            alert_involved_object_kind: Event
            alert_involved_object_options: Single
            alert_name: cluster.event.count-6sial # Имя правила оповеще
ния
            alert_namespace: cpaas-system # Namespace, где расположено
правило оповещения
            alert_project: cpaas-system # Имя проекта объекта, к которо
му относится правило оповещения
            alert_repeat_interval: 5m
            alert_resource: policy2 # Имя политики оповещений, где расп
оложено правило
            alert_source: Platform # Тип данных политики оповещений: Pl
atform - данные платформы, Business - бизнес-данные
```

```
          severity: High # Уровень серьёзности правила оповещения: Cr
itical - критический, High - серьёзный, Medium - предупреждение, Low -
информация
```

Создание политик оповещений через alert Templates

alert templates — это комбинация правил оповещений и политик уведомлений, ориентированных на похожие ресурсы. С помощью alert templates легко и быстро создавать политики оповещений для кластеров, узлов или вычислительных компонентов на платформе.

Предварительные условия

- Настроена политика уведомлений (если требуется автоматическая отправка оповещений).
- В целевом кластере установлены компоненты мониторинга (требуется при создании политик оповещений с использованием показателей мониторинга).

Процедура

Создание шаблона оповещений

1. В левой навигационной панели нажмите **Центр эксплуатации и обслуживания** > **alerts** > **alert Templates**.
2. Нажмите **Create alert Template**.
3. Настройте основную информацию шаблона оповещений.
4. В разделе **alert Rules** нажмите **Add alert Rule** и добавьте правила оповещений согласно описанию параметров ниже:

Параметр	Описание
Expression	Алгоритм метрики мониторинга в формате Prometheus, например, `rate(node_network_receive_bytes{instance="$server",device!~"lo"}[5m])`

Параметр	Описание
Metric Unit	Единица измерения пользовательской метрики мониторинга, можно ввести вручную или выбрать из предустановленных единиц платформы
Legend Parameter	Управляет именем, соответствующим кривой на графике, формат `{{.LabelName}}`, например, `{{.hostname}}`
Time Range	Временное окно для запросов логов/событий
Log Content	Поля запроса для содержимого логов (например, Error), несколько полей связаны через OR
Event Reason	Поля запроса для причин событий (Reason, например, BackOff, Pulling, Failed и т.д.), несколько полей связаны через OR
Trigger Condition	Условие, состоящее из операторов сравнения, порогов оповещения и длительности (опционально).
alert Level	Делится на четыре уровня: Critical, Serious, Warning и Info. Можно установить разумный уровень оповещения в зависимости от влияния правил оповещений на бизнес для соответствующих ресурсов.

5. Нажмите **Create**.

Создание политик оповещений с использованием alert Templates

1. В левой навигационной панели нажмите **Центр эксплуатации и обслуживания** > **alerts** > **alert Policies**.
   Совет: Вы можете переключать целевой кластер через верхнюю навигационную панель.
2. Нажмите кнопку раскрытия рядом с кнопкой **Create alert Policy** > **Template Create alert Policy**.
3. Настройте некоторые параметры, ориентируясь на описания ниже:

Параметр	Описание
Template Name	Имя используемого шаблона оповещений. Шаблоны классифицированы по кластеру, узлу и вычислительному компоненту. При выборе шаблона можно просмотреть правила оповещений, политики уведомлений и другую информацию, заданную в шаблоне оповещений.
Resource Type	Выберите, является ли шаблон шаблоном политики оповещений для **Cluster**, **Node** или **Computing Component**; будет отображено соответствующее имя ресурса.

4. Нажмите **Create**.

Настройка заглушения оповещений

Поддерживается заглушение оповещений для кластеров, узлов и вычислительных компонентов. При настройке заглушения для конкретной политики оповещений можно контролировать, что все правила в рамках этой политики не будут отправлять уведомления при срабатывании в течение заданного периода заглушения. Можно задать постоянное заглушение или заглушение на определённое время.

Например: при обновлении или обслуживании платформы многие ресурсы могут показывать аномальные состояния, что приводит к множеству срабатывающих оповещений и частому получению уведомлений операционным персоналом до завершения обновления или обслуживания. Настройка заглушения для политики оповещений позволяет избежать такой ситуации.

Примечание: Когда статус заглушения сохраняется до времени окончания заглушения, настройка заглушения автоматически снимается.

Настройка через UI

1. В левой навигационной панели нажмите **Центр эксплуатации и обслуживания** > **alerts** > **alert Policies**.
2. Нажмите кнопку действий справа от политики оповещений, которую нужно заглушить > **Set Silence**.

3. Включите переключатель **alert Silence**.
   Совет: Этот переключатель управляет действием настройки заглушения. Чтобы отменить заглушение, просто выключите переключатель.
4. Настройте соответствующие параметры согласно описаниям ниже:
   Совет: Если не выбран диапазон заглушения или имя ресурса, по умолчанию используется **Any**, что означает, что последующие действия **Delete/Add** ресурсов будут соответствовать **Delete Silence/Add Silence** политикам оповещений; если выбрано "Select All", это применяется только к текущему выбранному диапазону ресурсов, и последующие действия **Delete/Add** ресурсов не обрабатываются.

Параметр	Описание
Silence Range	Область ресурсов, в которой действует настройка заглушения.
Resource Name	Имя объекта ресурса, на который направлена настройка заглушения.
Silence Time	Временной диапазон заглушения оповещений. Оповещение переходит в состояние заглушения с начала времени заглушения, и если политика оповещений остаётся в состоянии оповещения или срабатывает после окончания времени заглушения, уведомления возобновляются. **Permanent**: настройка заглушения действует до удаления политики оповещений. **Custom**: пользовательские настройки времени начала и окончания заглушения, интервал не менее 5 минут.

5. Нажмите **Set**.
   Совет: С момента установки заглушения до начала времени заглушения статус заглушения политики оповещений считается **Silence Waiting**. В этот период при срабатывании правил уведомления отправляются как обычно; после начала и до окончания заглушения статус считается **Silencing**, и при срабатывании правил уведомления не отправляются.

Настройка через CLI

1. Укажите имя ресурса политики оповещений, для которой нужно настроить заглушение, и выполните команду:

```
kubectl edit PrometheusRule <TheNameOfThealertPolicyYouWantToSet>
```

2. Измените ресурс, добавив аннотации заглушения, как показано в примере, и отправьте изменения.

```
---
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  annotations:
    alert.cpaas.io/cluster: global
    alert.cpaas.io/kind: Node
    alert.cpaas.io/name: 0.0.0.0
    alert.cpaas.io/namespace: cpaas-system
    alert.cpaas.io/notifications: '[]'
    alert.cpaas.io/rules.description: '{}'
    alert.cpaas.io/rules.disabled: '[]'
    alert.cpaas.io/rules.version: '23'
    alert.cpaas.io/silence.config:
      '{"startsAt":"2025-02-08T08:01:37Z","endsAt":"2025-02-22T08:01:37
Z","creator":"leizhu@alauda.io","resources":{"nodes":[{"name":"192.168.
36.11","ip":"192.168.36.11"},{"name":"192.168.36.12","ip":"192.168.36.1
2"},{"name":"192.168.36.13","ip":"192.168.36.13"}]}}'
      # Конфигурация заглушения для политики оповещений на уровне узло
в, включая время начала, окончания, создателя и т.д.; если диапазон заг
лушения включает конкретные узлы, добавьте информацию resources.node, к
ак показано выше. Если требуется заглушение для всех ресурсов, поле res
ources не нужно.
    # alert.cpaas.io/silence.config: '{"startsAt":"2025-02-08T08:04:50
Z","endsAt":"2199-12-31T00:00:00Z","creator":"leizhu@alauda.io","name":
["alb-operator-ctl","apollo"],"namespace":["cpaas-system"]}'
    # Конфигурация заглушения для политики оповещений на уровне worklоа
d, включая время начала, окончания, создателя и т.д.; если диапазон заг
лушения включает конкретные workloads, добавьте name и namespace, как п
оказано выше. Если требуется заглушение для всех ресурсов, поля name и
namespace не нужны.
    # Установка endsAt в 2199-12-31T00:00:00Z означает постоянное заглу
шение.
    alert.cpaas.io/subkind: ''
    cpaas.io/creator: leizhu@alauda.io
    cpaas.io/description: ''
    cpaas.io/display-name: policy3
    cpaas.io/updated-at: 2025-02-08T08:01:42Z
  labels:
  ## Исключены нерелевантные данные
```

Рекомендации по настройке правил оповещений

Большее количество правил оповещений не всегда означает лучший результат. Избыточные или сложные правила могут привести к лавине оповещений и увеличить нагрузку на обслуживание. Рекомендуется ознакомиться с приведёнными ниже рекомендациями перед настройкой правил, чтобы пользовательские правила достигали своих целей при сохранении эффективности.

- **Используйте минимально необходимое количество новых правил**: Создавайте только те правила, которые соответствуют вашим конкретным требованиям. Используя минимальное количество правил, вы создадите более управляемую и централизованную систему оповещений в среде мониторинга.
- **Фокусируйтесь на симптомах, а не на причинах**: Создавайте правила, которые уведомляют пользователей о симптомах, а не о корневых причинах этих симптомов. Это гарантирует, что при появлении соответствующих симптомов пользователи получат оповещения и смогут исследовать причины, вызвавшие эти оповещения. Такой подход значительно сокращает общее количество необходимых правил.
- **Планируйте и оценивайте потребности перед изменениями**: Сначала определите, какие симптомы важны и какие действия вы хотите, чтобы пользователи выполняли при их появлении. Затем оцените существующие правила, чтобы решить, можно ли их изменить для достижения целей без создания новых правил для каждого симптома. Модифицируя существующие правила и тщательно создавая новые, вы упростите систему оповещений.
- **Предоставляйте чёткие сообщения оповещений**: При создании сообщений включайте описание симптомов, возможных причин и рекомендуемых действий. Информация должна быть ясной, краткой и содержать процедуры устранения неполадок или ссылки на дополнительную релевантную информацию. Это помогает пользователям быстро оценивать ситуацию и реагировать соответствующим образом.
- **Разумно устанавливайте уровни серьёзности**: Присваивайте уровням серьёзности правила, чтобы указать, как пользователи должны реагировать при срабатывании оповещений. Например, классифицируйте оповещения с уровнем Critical как требующие немедленных действий от ответственных лиц. Установив уровни серьёзности, вы поможете пользователям принимать решения при получении оповещений и обеспечите своевременную реакцию на критические проблемы.

Управление уведомлениями

Содержание

- Обзор функции
- Ключевые функции
- Notification Server
  - Corporate Communication Tool Server
  - Email Server
  - Webhook Type Server
  - Настройка пользовательских заголовков запросов для уведомлений Webhook
- Notification Contact Group
  - Связать группу контактов уведомлений с Secret заголовков
- Notification Template
  - Создание шаблона уведомления
  - Reference Variables
  - Special Formatting Markup Language in Emails
- Notification rule
  - Предварительные условия
  - Порядок действий
- Настройка правила уведомлений для проектов
  - Предварительные условия
  - Порядок действий

Обзор функции

С помощью уведомлений вы можете интегрировать функции мониторинга и оповещения платформы для своевременной отправки информации о предварительных предупреждениях получателям уведомлений, напоминая соответствующим сотрудникам принять необходимые меры для устранения проблем или предотвращения сбоев.

Ключевые функции

- **Notification Server**: сервер уведомлений предоставляет сервисы для отправки сообщений уведомлений группам контактов на платформе, например, сервер электронной почты.
- **Notification Contact Group**: группа контактов уведомлений — это набор получателей уведомлений с похожими логическими характеристиками, что позволяет снизить нагрузку на обслуживание за счёт категоризации сущностей, получающих уведомления.
- **Notification Template**: шаблон уведомления — это стандартизированная структура, состоящая из настраиваемого содержимого, переменных содержимого и параметров форматирования. Используется для стандартизации содержания и формата сообщений оповещений в стратегиях уведомлений. Например, настройка темы и содержимого email-уведомлений.
- **Notification rule**: правило уведомления — это набор правил, определяющих, как отправлять сообщения уведомлений конкретным контактам. Использование правила уведомления необходимо для сценариев, таких как оповещения, проверки и аутентификация при входе, требующих уведомления внешних сервисов.

Notification Server

Сервер уведомлений предоставляет сервисы для отправки сообщений уведомлений получателям на платформе. В настоящее время платформа поддерживает следующие серверы уведомлений:

- **Corporate Communication Tool Server**: поддерживает интеграцию с встроенными приложениями WeChat Work, DingTalk и Feishu для отправки уведомлений отдельным

пользователям.

- **Email Server**: отправляет уведомления по электронной почте через email-сервер.
- **Webhook Type Server**: поддерживает интеграцию с ботами групп корпоративного WeChat, DingTalk, Feishu или отправку WebHook на ваш указанный сервер.

> **WARNING**
>
> Можно добавить только один сервер корпоративного коммуникационного инструмента.

Corporate Communication Tool Server

WeChat Work

1. Настройте параметры сервера уведомлений согласно приведённому ниже примеру. После заполнения параметров переключитесь на кластер `global` в **Cluster Management** > **Resource Management** и создайте объект ресурса.

```
# Методы получения corpId, corpSecret, agentId для WeChat Work можно на
йти в официальной документации: https://developer.work.weixin.qq.com/do
cument/path/90665
apiVersion: v1
kind: Secret
type: NotificationServer
metadata:
  labels:
    cpaas.io/notification.server.type: CorpWeChat
    cpaas.io/notification.server.category: Corp
  name: platform-corp-wechat-server
  namespace: cpaas-system
data:
  displayNameZh: 企业微信             # Отображаемое имя сервера на китай
ском, по умолчанию в base64
  displayNameEn: WeChat              # Отображаемое имя сервера на англи
йском, по умолчанию в base64
  corpId:                            # Корпоративный ID, по умолчанию в
base64
  corpSecret:                        # Секрет приложения, по умолчанию в
base64
  agentId:                           # ID корпоративного приложения, по
умолчанию в base64
```

2. После создания необходимо обновить **WeChat Work ID** пользователя в **User Role Management** > **User Management** платформы или в **Personal Information** пользователя, чтобы обеспечить нормальный приём сообщений.

DingTalk

1. Настройте параметры сервера уведомлений согласно приведённому ниже примеру. После заполнения параметров переключитесь на кластер `global` в **Cluster Management** > **Resource Management** и создайте объект ресурса.

```yaml
# Метод получения appKey, appSecret, agentId для DingTalk: https://open
-dev.dingtalk.com/fe/app#/corp/app
apiVersion: v1
kind: Secret
type: NotificationServer
metadata:
  labels:
    cpaas.io/notification.server.type: CorpDingTalk
    cpaas.io/notification.server.category: Corp
  name: platform-corp-dingtalk-server
  namespace: cpaas-system
data:
  displayNameZh: 钉钉                # Отображаемое имя сервера на китай
ском, по умолчанию в base64
  displayNameEn: DingTalk           # Отображаемое имя сервера на англи
йском, по умолчанию в base64
  appKey:                           # Ключ приложения, по умолчанию в b
ase64
  appSecret:                        # Секрет приложения, по умолчанию в
base64
  agentId:                          # agent_id приложения, по умолчанию
в base64
```

2. После создания необходимо обновить **DingTalk ID** пользователя в **User Role Management** > **User Management** платформы или в **Personal Information** пользователя, чтобы обеспечить нормальный приём сообщений.

Feishu

1. Настройте параметры сервера уведомлений согласно приведённому ниже примеру. После заполнения параметров переключитесь на кластер `global` в **Cluster Management** > **Resource Management** и создайте объект ресурса.

```
# Методы получения appId, appSecret для Feishu: https://open.feishu.cn/app/
apiVersion: v1
kind: Secret
type: NotificationServer
metadata:
  labels:
    cpaas.io/notification.server.type: CorpFeishu
    cpaas.io/notification.server.category: Corp
  name: platform-corp-feishu-server
  namespace: cpaas-system
data:
  displayNameZh: 飞书                  # Отображаемое имя сервера на китайском, по умолчанию в base64
  displayNameEn: Feishu                # Отображаемое имя сервера на английском, по умолчанию в base64
  appId:                               # ID приложения, по умолчанию в base64
  appSecret:                           # Секрет приложения, по умолчанию в base64
```

2. После создания необходимо обновить **Feishu ID** пользователя в **User Role Management** > **User Management** платформы или в **Personal Information** пользователя, чтобы обеспечить нормальный приём сообщений.

Email Server

1. В левой навигационной панели нажмите **Platform Settings** > **Notification Server**.
2. Нажмите **Configure Now**.
3. Следуйте инструкциям для настройки соответствующих параметров.

Параметр	Описание
Service Address	Адрес сервера уведомлений, поддерживающего протокол SMTP, например, `smtp.yeah.net`.
Port	Номер порта сервера уведомлений. При включённом флажке **Use SSL** необходимо указать SSL-порт.

Параметр	Описание
Server Configuration	**Use SSL**: Secure Socket Layer (SSL) — стандартная технология безопасности. Переключатель SSL управляет установкой зашифрованного соединения между сервером и клиентом. **Skip Insecure Verification**: переключатель insecureSkipVerify управляет проверкой сертификата клиента и имени хоста сервера. Если включён, сертификаты и соответствие имени хоста в сертификате имени сервера проверяться не будут.
Sender Email	Email-аккаунт отправителя на сервере уведомлений, используемый для отправки уведомлений по электронной почте.
Enable Authentication	Если требуется аутентификация, настройте имя пользователя и код авторизации для email-сервера.

4. Нажмите **OK**.

Webhook Type Server

Поддерживает интеграцию с ботами групп корпоративного WeChat, DingTalk, Feishu или отправку HTTP-запросов на ваш указанный Webhook сервер.

Корпоративный бот группы WeChat

1. В левой навигационной панели нажмите **Cluster Management** > **Cluster**.
2. Нажмите кнопку операций рядом с кластером `global` > **CLI Tool**.
3. Выполните следующую команду на мастер-узле кластера `global`:

```
kubectl patch secret -n cpaas-system platform-wechat-server -p '{"dat
a":{"enable":"dHJ1ZQo="}}'
```

Подсказка: `dHJ1ZQo=` — это значение true, закодированное в base64; чтобы отключить, замените `dHJ1ZQo=` на `ZmFsc2UK`, что является значением false в base64.

Бот группы DingTalk

1. В левой навигационной панели нажмите **Cluster Management** > **Cluster**.
2. Нажмите кнопку операций рядом с кластером `global` > **CLI Tool**.
3. Выполните следующую команду на мастер-узле кластера `global`:

```
kubectl patch secret -n cpaas-system platform-dingtalk-server -p '{"dat
a":{"enable":"dHJ1ZQo="}}'
```

Подсказка: `dHJ1ZQo=` — это значение true, закодированное в base64; чтобы отключить, замените `dHJ1ZQo=` на `ZmFsc2UK`, что является значением false в base64.

Бот группы Feishu

1. В левой навигационной панели нажмите **Cluster Management** > **Cluster**.
2. Нажмите кнопку операций рядом с кластером `global` > **CLI Tool**.
3. Выполните следующую команду на мастер-узле кластера `global`:

```
kubectl patch secret -n cpaas-system platform-feishu-server -p '{"dat
a":{"enable":"dHJ1ZQo="}}'
```

Подсказка: `dHJ1ZQo=` — это значение true, закодированное в base64; чтобы отключить, замените `dHJ1ZQo=` на `ZmFsc2UK`, что является значением false в base64.

Webhook Server

1. В левой навигационной панели нажмите **Cluster Management** > **Cluster**.
2. Нажмите кнопку операций рядом с кластером `global` > **CLI Tool**.
3. Выполните следующую команду на мастер-узле кластера `global`:

```
kubectl patch secret -n cpaas-system platform-webhook-server -p '{"dat
a":{"enable":"dHJ1ZQo="}}'
```

Подсказка: `dHJ1ZQo=` — это значение true, закодированное в base64; чтобы отключить, замените `dHJ1ZQo=` на `ZmFsc2UK`, что является значением false в base64.

Настройка пользовательских заголовков запросов для уведомлений Webhook

Если целевой Webhook-эндпоинт требует пользовательские HTTP-заголовки, создайте Secret в пространстве имён `cpaas-system` и позже свяжите его с группой контактов уведомлений.

1. В левой навигационной панели нажмите **Cluster Management** > **Cluster**.
2. Нажмите кнопку операций рядом с кластером `global` > **CLI Tool**.
3. Создайте YAML-файл, аналогичный следующему, на мастер-узле кластера `global`.

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: webhook-header-secret
  namespace: cpaas-system
type: NotificationSender
data:
  X-Secret: <base64-encoded-header-value>
```

4. Сохраните YAML-файл как `webhook-header.yaml` и примените ресурс.

```
kubectl apply -f webhook-header.yaml
```

INFO

1. Каждый ключ в `data` используется как имя HTTP-заголовка.
2. Каждое значение в `data` должно быть закодировано в base64 перед применением к кластеру.
3. Если эндпоинт требует несколько заголовков, добавьте дополнительные пары ключ-значение в `data`.

Notification Contact Group

Группа контактов уведомлений — это набор получателей уведомлений с похожими логическими характеристиками. Например, вы можете задать команду эксплуатации и обслуживания как группу контактов уведомлений для удобного выбора и управления при настройке стратегий уведомлений.

INFO

1. Платформа поддерживает различные серверы уведомлений, и соответствующие параметры конфигурации для типов уведомлений будут отображаться в зависимости от настройки сервера уведомлений.
2. Если необходимо использовать сервер типа Webhook в качестве получателя уведомлений, необходимо настроить соответствующий URL в группе контактов уведомлений.
3. Если Webhook-эндпоинт требует пользовательские заголовки запросов, необходимо связать группу контактов уведомлений с Secret типа `NotificationSender` в пространстве имён `cpaas-system`.

1. В левой навигационной панели нажмите **Operations Center** > **Notifications**.
2. Перейдите на вкладку **Notification Contact Group**.
3. Нажмите **Create Notification Contact Group** и настройте соответствующие параметры согласно инструкции ниже.

Параметр	Описание
Email	Добавьте email для всей группы контактов уведомлений. Платформа будет отправлять уведомления на этот email и на email всех контактов в группе.
Webhook URL/WeChat Group Bot/DingTalk Group Bot/Feishu Group Bot	Укажите соответствующий URL метода уведомления в зависимости от настроенного сервера уведомлений. После настройки контакты в этой группе будут уведомляться этим методом.
Contact Configuration	Нажмите **Add Contact**, чтобы добавить существующих пользователей платформы в группу контактов. Убедитесь в

Параметр	Описание
	корректности контактной информации выбранных контактов (телефон, email, callback интерфейс), чтобы избежать пропуска уведомлений.

4. Нажмите **Add**.

Связать группу контактов уведомлений с Secret заголовков

Если настроенный Webhook-эндпоинт требует пользовательские заголовки запросов, свяжите группу контактов с Secret, созданным в разделе Настройка пользовательских заголовков запросов для уведомлений Webhook.

1. В левой навигационной панели нажмите **Cluster Management** > **Cluster**.
2. Нажмите кнопку операций рядом с кластером `global` > **CLI Tool**.
3. Отредактируйте соответствующий ресурс `NotificationGroup` на мастер-узле кластера `global`.

```
kubectl edit notificationgroups.ait.alauda.io -n cpaas-system <notification-group-name>
```

4. Добавьте аннотацию `cpaas.io/notification.webhook.config` в `metadata.annotations`. Значение должно быть именем Secret, созданного для пользовательских заголовков запросов.

```
apiVersion: ait.alauda.io/v1beta1
kind: NotificationGroup
metadata:
  name: <notification-group-name>
  namespace: cpaas-system
  annotations:
    cpaas.io/notification.webhook.config: webhook-header-secret
```

INFO

URL Webhook настраивается в группе контактов уведомлений, а пользовательские заголовки запросов настраиваются через связанный Secret.

Notification Template

Шаблон уведомления — это стандартизированная структура, состоящая из настраиваемого содержимого, переменных содержимого и параметров форматирования. Используется для стандартизации содержания и формата сообщений оповещений в стратегиях уведомлений.

Администраторы платформы или операционные сотрудники могут задавать шаблоны уведомлений для настройки содержания и формата сообщений уведомлений в зависимости от различных методов оповещения, помогая пользователям быстро получать критически важную информацию об оповещениях и повышать эффективность работы.

INFO

Платформа поддерживает различные серверы уведомлений, и соответствующие шаблоны уведомлений будут отображаться в зависимости от настройки сервера уведомлений. Если сервер уведомлений не настроен, соответствующие шаблоны уведомлений по умолчанию не отображаются.

Создание шаблона уведомления

1. В левой навигационной панели нажмите **Operations Center** > **Notifications**.
2. Перейдите на вкладку **Notification Template**.
3. Нажмите **Create Notification Template**.
4. В разделе **Basic Information** настройте следующие параметры.

Параметр	Описание
Message Type	Выберите тип сообщения в зависимости от цели уведомления. **Alert Message**: отправляет сообщения оповещений, вызванных правилами оповещений, в связке с функцией оповещений платформы; **Component Exception Message**: отправляет уведомления, вызванные исключениями в некоторых компонентах.

5. В разделе **Template Configuration** настройте переменные и параметры форматирования, опираясь на различные типы шаблонов.

INFO

1. Содержимое шаблона может состоять только из переменных, отображаемых имён переменных и специального языка разметки форматирования, поддерживаемого платформой. Переменные и другие элементы можно свободно комбинировать при соблюдении синтаксических правил.
2. В шаблоне можно использовать только переменные, поддерживаемые платформой. Можно изменять отображаемые имена переменных и формат содержимого, но нельзя изменять сами переменные. См. Reference Variables и Special Formatting Markup Language in Emails.
3. Платформа предоставляет стандартное содержимое шаблонов уведомлений для различных типов уведомлений на основе реальных сценариев эксплуатации, что удовлетворяет большинству потребностей в настройке сообщений уведомлений. При отсутствии особых требований можно использовать содержимое шаблона по умолчанию.

6. Нажмите **Create**.

Reference Variables

Переменные — это ключи меток или аннотаций в сообщениях уведомлений (NotificationMessage), оформленные как `{{.labelKey}}`. Для удобства быстрого получения ключевой информации пользователям можно назначать переменным настраиваемые отображаемые имена; например: `Alert Level: {{ .externalLabels.severity }}`.

Когда правило уведомления отправляет сообщения уведомлений пользователям на основе шаблона уведомления, переменные в шаблоне ссылаются на соответствующие значения меток в сообщении уведомления (фактические данные мониторинга). В итоге пользователям отправляются данные мониторинга в стандартизированном формате содержимого.

Платформа по умолчанию предоставляет следующие базовые переменные:

Отображаемое имя	Переменная	Описание
Alert Status	`{{ .externalLabels.status }}`	Например: Alerting.
Alert Level	`{{ .externalLabels.severity }}`	Например: Critical.
Alert Cluster	`{{ .labels.alert_cluster }}`	Например: кластер 1, где произошло оповещение.
Alert Object	`{{ .externalLabels.object }}`	Тип и имя ресурса, где произошло оповещение, например, node 192.168.16.53.
rule Name	`{{ .labels.alert_resource }}`	Имя правила оповещения, например, cpaas-node-rules.
Alert Description	`{{ .externalLabels.summary }}`	Описание правила оповещения.
Trigger Value	`{{ .externalLabels.currentValue }}`	Мониторинговое значение, вызвавшее оповещение.
Alert Time	`{{ dateFormatWithZone .startsAt "2006-01-02 15:04:05" "Asia/Chongqing" }}`	Время начала оповещения.
Recovery Time	`{{ dateFormatWithZone .endsAt "2006-01-02 15:04:05" "Asia/Chongqing" }}`	Время окончания оповещения.

Отображаемое имя	Переменная	Описание
Metric Name	`{{ .labels.alert_indicator }}`	Название мониторинговой метрики.

Special Formatting Markup Language in Emails

В email-уведомлениях используются распространённые HTML-теги форматирования, описанные в таблице ниже:

Элемент содержимого	Тег	Описание
Текст	-	Поддерживается ввод текста на китайском/ английском.
Шрифт	`<font color="#FF0000">Установить цвет шрифта</font>` `<b>Жирный шрифт</b>`	Установка формата шрифта.
Заголовок	`<h1>Заголовок уровня 1</h1>`, поддерживается до h6 (заголовок 6).	Установка уровня заголовка.
Абзац	`<p>Абзац</p>`	Вставка обычного абзаца текста.
Цитата	`<q>Цитата</q>`	Вставка короткой цитаты.
Гиперссылка	`<a href="//www.example.com">Гиперссылка</a>`	Вставка гиперссылки.

Notification rule

Правило уведомления — это набор правил, определяющих, как отправлять сообщения уведомлений конкретным контактам. Использование стратегий уведомлений необходимо для сценариев, требующих уведомления внешних сервисов, таких как оповещения, проверки и аутентификация при входе.

> **INFO**
>
> Платформа поддерживает различные серверы уведомлений, и режимы уведомлений, соответствующие типам уведомлений, будут отображаться в зависимости от настройки сервера уведомлений. Если сервер уведомлений не настроен, соответствующие режимы уведомлений по умолчанию не отображаются.

Предварительные условия

Для использования **Corporate Communication Tool Server** для уведомления контактов пользователям необходимо сначала изменить контактную информацию в **Personal Information**, указав свой `WeChat Work ID`.

Порядок действий

1. В левой навигационной панели нажмите **Operations Center** > **Notifications**.
2. Нажмите **Create Notification rule** и настройте соответствующие параметры согласно инструкции ниже.

Параметр	Описание
Notification Contact Group	Группа контактов уведомлений — логический набор получателей уведомлений, которых платформа уведомляет указанным методом уведомления.
Notification Recipients	Выберите одного или нескольких получателей уведомлений, платформа будет отправлять уведомления согласно контактным данным в **Personal Information** получателей.

Параметр	Описание
Notification Method	Поддерживает несколько методов, включая **WeChat Work**, **DingTalk**, **Feishu**, **Corporate WeChat Group Bot**, **DingTalk Group Bot**, **Feishu Group Bot**, **WebHook URL**, поддерживается множественный выбор. **Примечание**: некоторые параметры отображаются после настройки сервера уведомлений.
Notification Template	Выберите шаблон уведомления для отображения информации уведомления.

3. Нажмите **Create**.

Настройка правила уведомлений для проектов

Стратегии уведомлений, шаблоны уведомлений и группы контактов уведомлений на платформе изолированы по арендаторам. Как администратор проекта, вы не сможете просматривать или использовать стратегии уведомлений, шаблоны уведомлений или группы контактов, настроенные другими проектами или администраторами платформы. Поэтому необходимо руководствоваться этим документом для настройки подходящих стратегий уведомлений для вашего проекта.

Предварительные условия

1. Вы связались с администратором платформы для завершения настройки сервера уведомлений.
2. Если требуется уведомление через корпоративные коммуникационные инструменты, необходимо убедиться, что уведомляемые контакты корректно настроили свои идентификаторы коммуникационных инструментов в **Personal Information**.

Порядок действий

1. В представлении **Project Management** нажмите ***Project Name***.
2. В левой навигационной панели нажмите **Notifications**.

3. Перейдите на вкладку **Notification Contact Group**, ознакомьтесь с разделом Notification Contact Group и создайте группу контактов уведомлений.

> **TIP**
>
> Если нет необходимости управлять контактами уведомлений через группу контактов уведомлений или уведомлять сервер уведомлений типа webhook, этот шаг можно пропустить.

4. Перейдите на вкладку **Notification Template**, ознакомьтесь с разделом Notification Template и создайте шаблон уведомления.
5. Перейдите на вкладку **Notification rule**, ознакомьтесь с разделом Notification rule и создайте правило уведомления.

Управление мониторинговыми панелями

Содержание

- Обзор функции
  - Основные возможности
  - Преимущества
  - Сценарии использования
  - Предварительные условия
  - Взаимосвязь между панелями мониторинга и компонентами мониторинга
- Управление панелями
  - Создание дашборда
  - Импорт дашборда
  - Добавление переменных
  - Добавление панелей
  - Добавление групп
  - Переключение дашбордов
  - Другие операции
- Управление панелями
  - Описание панелей
  - Описание конфигурации панелей
    - Общие параметры
    - Специальные параметры для панелей
- Создание панелей мониторинга через CLI
- Общие функции и переменные
  - Общие функции

Общие переменные

Пример использования переменной 1

Пример использования переменной 2

Примечания при использовании встроенных метрик

Обзор функции

Платформа предоставляет мощный функционал управления панелями мониторинга, предназначенный для замены традиционных инструментов Grafana, предлагая пользователям более комплексный и гибкий опыт мониторинга. Эта функция агрегирует различные данные мониторинга внутри платформы, представляя единый обзор мониторинга, что значительно повышает эффективность вашей настройки.

Основные возможности

- Поддержка настройки пользовательских панелей мониторинга как для бизнес-вью, так и для платформенных вью.
- Возможность просмотра публично расшаренных панелей, настроенных в платформенных вью, из бизнес-вью с изоляцией данных по namespace, к которому принадлежит бизнес.
- Поддержка управления панелями внутри дашборда: добавление, удаление, изменение панелей, масштабирование панелей и перемещение панелей с помощью drag-and-drop.
- Возможность установки пользовательских переменных в дашборде для фильтрации данных запросов.
- Поддержка настройки групп в дашборде для управления панелями. Группы могут отображаться повторно на основе пользовательских переменных.
- Поддерживаемые типы панелей: trend, step line chart, bar chart, horizontal bar chart, bar gauge chart, gauge chart, table, stat chart, XY chart, pie chart, text.
- Функция однокликового импорта панелей из Grafana.

Преимущества

- Поддержка пользовательских сценариев мониторинга без ограничений предопределёнными шаблонами, что позволяет добиться по-настоящему персонализированного мониторинга.
- Богатый набор вариантов визуализации, включая линейные графики, столбчатые диаграммы, круговые диаграммы, а также гибкие варианты компоновки и стилей.
- Глубокая интеграция с ролевыми правами платформы, позволяющая бизнес-вью определять собственные панели мониторинга при обеспечении изоляции данных.
- Глубокая интеграция с различными функциями контейнерной платформы, обеспечивающая мгновенный доступ к данным мониторинга контейнеров, сетей, хранилищ и т. д., предоставляя пользователям комплексное наблюдение за производительностью и диагностику неисправностей.
- Полная совместимость с JSON панелей Grafana, что облегчает миграцию с Grafana для дальнейшего использования.

Сценарии использования

- **Управление IT-операциями**: В составе команды IT-операций вы можете использовать панели мониторинга для унифицированного отображения и управления различными метриками производительности контейнерной платформы, такими как CPU, память, сетевой трафик и др. Настраивая отчёты мониторинга и правила оповещений, вы сможете своевременно обнаруживать и локализовать проблемы системы, повышая эффективность эксплуатации.
- **Анализ производительности приложений**: Для разработчиков и тестировщиков приложений панели мониторинга предлагают разнообразные варианты визуализации для наглядного отображения состояния работы приложений и потребления ресурсов. Вы можете создавать специализированные виды мониторинга, адаптированные под разные сценарии приложений, для глубокого анализа узких мест производительности и предоставления основы для оптимизации.
- **Управление мультикластером**: Для пользователей, управляющих несколькими контейнерными кластерами, панели мониторинга могут агрегировать данные мониторинга из разных кластеров, позволяя получить общее представление о состоянии системы.

- **Диагностика неисправностей**: При возникновении проблем в системе панели мониторинга предоставляют комплексные данные о производительности и инструменты анализа для быстрого выявления корневой причины. Вы можете оперативно просматривать колебания соответствующих метрик на основе информации об оповещениях для углублённого анализа неисправностей.

Предварительные условия

В настоящее время панели мониторинга поддерживают только просмотр данных мониторинга, собираемых компонентами мониторинга, установленными в платформе. Поэтому перед настройкой панели мониторинга необходимо подготовить следующее:

- Убедитесь, что в кластере, для которого вы хотите настроить панель мониторинга, установлены компоненты мониторинга, а именно плагин **ACP Monitor с Prometheus** или **ACP Monitor с VictoriaMetrics**.
- Убедитесь, что данные, которые вы хотите отображать на панели, были собраны компонентами мониторинга.

Взаимосвязь между панелями мониторинга и компонентами мониторинга

- Ресурсы панелей мониторинга хранятся в Kubernetes кластере. Вы можете переключать вью между разными кластерами с помощью вкладки **Cluster** в верхней части.
- Панели мониторинга зависят от компонентов мониторинга в кластере для запроса источников данных. Поэтому перед использованием панелей мониторинга убедитесь, что в текущем кластере успешно установлены компоненты мониторинга и они работают нормально.
- Панель мониторинга по умолчанию запрашивает данные мониторинга из соответствующего кластера. Если вы установите плагин **VictoriaMetrics** в режиме proxy в кластере, мы автоматически запросим кластер хранения для получения соответствующих данных без необходимости специальной настройки.

Управление панелями

Дашборд — это коллекция из одной или нескольких панелей, организованных и расположенных в одной или нескольких строках для предоставления чёткого обзора релевантной информации. Эти панели могут запрашивать исходные данные из источников и преобразовывать их в ряд визуальных эффектов, поддерживаемых платформой.

Создание дашборда

1. Нажмите **Create Dashboard**, следуйте инструкциям ниже для настройки соответствующих параметров.

Параметр	Описание
Folder	Папка, в которой находится дашборд; можно ввести или выбрать существующую папку.
Label	Метка для панели мониторинга; вы можете быстро находить существующие дашборды, фильтруя по верхним меткам при переключении.
Set as Main Dashboard	Если включено, текущий дашборд будет установлен как основной после успешного создания; при повторном входе в функцию мониторинга по умолчанию будет отображаться основной дашборд.
Variables	Добавьте переменные при создании дашборда для использования в качестве параметров метрик в добавленных панелях, которые также могут использоваться как фильтры на главной странице дашборда.

2. После добавления нажмите **Create** для завершения создания дашборда. Далее необходимо **добавить переменные**, **добавить панели** и **добавить группы** для завершения общей компоновки.

Импорт дашборда

Платформа поддерживает прямой импорт JSON из Grafana для конвертации в панель мониторинга для отображения.

- В настоящее время поддерживается только Grafana JSON версии V8+; более низкие версии запрещены к импорту.

- Если какие-либо панели в импортированном дашборде не входят в поддерживаемый платформой диапазон, они могут отображаться как **unsupported panel types**, но вы можете изменить настройки панели для нормального отображения.
- После импорта дашборда вы можете выполнять любые операции управления как обычно, без отличий от панелей, созданных в платформе.

Добавление переменных

1. В области формы переменных нажмите **Add**.

Query

Переменные типа `Query` позволяют фильтровать данные на основе измерений временных рядов. Выражение запроса можно задать для динамического вычисления и генерации результатов запроса.

Параметр	Описание
Query Settings	При определении настроек запроса, помимо использования PromQL для запроса временных рядов, платформа также предоставляет некоторые общие переменные и функции. См. Common Functions and Variables.
Regular Expression	С помощью регулярных выражений можно отфильтровать нужные значения из содержимого, возвращаемого запросами переменных. Это делает каждое имя опции в переменной более ожидаемым. Вы можете предварительно просмотреть, соответствуют ли отфильтрованные значения ожиданиям, в **Variable Value Preview**.
Selection Settings	- **Multiple Selection**: При выборе из верхних фильтров на главной странице дашборда позволяет выбрать несколько опций одновременно. Необходимо ссылаться на эту переменную в выражении запроса панелей для просмотра данных, соответствующих значению переменной. - **All**: Если отмечено, в фильтрах будет доступна опция **All** для выбора всех данных переменной.

Constant

Постоянные переменные — это статические переменные с фиксированными значениями, которые не меняются в течение всего дашборда. Обычно используются для

хранения идентификаторов окружения, фиксированных порогов или параметров конфигурации, которые нужно использовать в нескольких панелях без отображения в фильтрах.

Параметр	Описание
Constant Value	Значение постоянной переменной.

Custom

Пользовательские переменные позволяют определить предопределённый список статических опций, которые отображаются в виде выпадающих фильтров на дашборде. Обычно используются для ручного выбора конкретных сервисов, команд или категорий без необходимости динамических запросов данных.

Параметр	Описание
Custom Settings	Введите значения опций через запятую, используя формат display_name : value для каждой опции (например, Production : prod, Staging : stage, Development : dev), или просто перечислите значения, если имя отображения совпадает со значением.

Textbox

Переменные типа Textbox позволяют пользователям вводить текст напрямую, обычно используются для указания конкретных значений или параметров, не требующих динамических запросов данных.

Параметр	Описание
Textbox Value	Значение по умолчанию для переменной.

2. Нажмите **OK** для добавления одной или нескольких переменных.

Добавление панелей

Добавьте несколько панелей в текущий созданный дашборд для отображения данных по разным ресурсам.

Совет: Вы можете настроить размер панели, щёлкнув по правому нижнему углу; щёлкните в любом месте панели для изменения порядка панелей.

1. Нажмите **Add Panel**, настройте параметры согласно инструкции ниже.

- **Panel Preview**: Область динамически отображает данные, соответствующие добавленным метрикам.
- **Add Metric**: Настройте заголовок панели и метрики мониторинга в этой области.
- **Способ добавления**: Поддерживается использование встроенных метрик или нативно настроенных метрик. Оба способа объединяются и действуют одновременно.
  - **Built-in Metrics**: Выберите часто используемые метрики и параметры легенды, встроенные в платформу, для отображения данных в текущей панели.
    - **Примечание**: Все метрики, добавленные в панель, должны иметь единый юнит; нельзя добавлять метрики с разными единицами в одну панель.
  - **Native**: Настройте единицу метрики, выражение метрики и параметры легенды. Выражение метрики следует синтаксису PromQL; подробности см. в PromQL Official Documentation ↗.
- **Legend Parameters**: Управляет именами, соответствующими кривым на панелях. Можно использовать текст или шаблоны:
  - Правило: вводимое значение должно быть в формате `{{.xxxx}}`; например, `{{.hostname}}` заменится на значение метки hostname, возвращаемое выражением.
  - **Совет**: Если ввести некорректно отформатированный параметр легенды, имена кривых будут отображаться в исходном формате.
- **Instant Switch**: При включённом переключателе **Instant** данные запрашиваются через интерфейс Query Prometheus в режиме мгновенного значения и сортируются, как в статистических и gauge диаграммах. Если выключен, используется метод `query_range` для запроса серии данных за определённый период.
- **Panel Settings**: Поддерживает выбор различных типов панелей для визуализации метрик. См. Manage Panels.

2. Нажмите **Save** для завершения добавления панелей.
3. Можно добавить одну или несколько панелей в дашборд.

4. После добавления панелей доступны следующие операции для настройки отображения и размера:

   - Щёлкните по правому нижнему углу панели для настройки размера.
   - Щёлкните в любом месте панели для изменения порядка панелей.
   - Нажмите кнопку **Edit** для изменения настроек панели.
   - Нажмите кнопку **Delete** для удаления панели.
   - Нажмите кнопку **Copy** для копирования панели.

5. После настройки нажмите кнопку **Save** на странице дашборда для сохранения изменений.

Добавление групп

Группы — это логические разделители внутри дашборда, которые могут группировать панели.

1. Нажмите выпадающее меню **Add Panel** > **Add Group**, настройте параметры согласно инструкции.

- **Group**: Имя группы.
- **Repeat**: Поддерживает отключение повторов или выбор переменных для текущих панелей.

  - **Disable Repeat**: Не выбирать переменную, использовать созданную по умолчанию группу.
  - **Parameter Variables**: Выберите переменные, созданные в текущих панелях, и дашборд сгенерирует строку идентичных подгрупп для каждого соответствующего значения переменной. Подгруппы не поддерживают изменение, удаление или перемещение панелей.

2. После добавления группы доступны следующие операции для управления отображением панелей в дашборде:

   - Группы можно сворачивать или разворачивать для скрытия части содержимого дашборда. Панели в свернутых группах не отправляют запросы.

- Переместите панель в группу, чтобы она управлялась этой группой. Группа управляет всеми панелями между ней и следующей группой.
- При сворачивании группы можно также перемещать все панели, управляемые этой группой, вместе.
- Сворачивание и разворачивание групп также считается изменением дашборда. Чтобы сохранить это состояние при следующем открытии, нажмите **Save**.

Переключение дашбордов

Установите созданный пользовательский дашборд мониторинга как основной. При повторном входе в функцию мониторинга по умолчанию будет отображаться основной дашборд.

1. В левой навигационной панели нажмите **Operations Center** > **Monitoring** > **Monitoring Dashboards**.
2. По умолчанию открывается основной дашборд мониторинга. Нажмите **Switch Dashboard**.
3. Вы можете найти дашборды, фильтруя по меткам или ищя по имени, и переключать основные дашборды с помощью переключателя **Main Dashboard**.

Другие операции

Вы можете нажать кнопку операций справа на странице дашборда для выполнения нужных действий.

Операция	Описание
YMAL	Открывает фактический код CR ресурса дашборда, хранящегося в Kubernetes кластере. Можно изменять весь контент дашборда, редактируя параметры в YAML.
Export Expression	Позволяет экспортировать метрики и соответствующие выражения запросов, используемые в текущем дашборде, в формате CSV.
Copy	Копирует текущий дашборд; вы можете редактировать панели по необходимости и сохранить как новый дашборд.

Операция	Описание
Settings	Изменяет основную информацию текущего дашборда, например, метки и добавление дополнительных переменных.
Delete	Удаляет текущий дашборд мониторинга.

Управление панелями

Платформа предоставляет различные методы визуализации для поддержки разных сценариев использования. В этом разделе представлены типы панелей, параметры конфигурации и способы использования.

Описание панелей

№	Название панели	Описание	Рекомендуемые сценарии использования
1	Trend Chart	Отображает тенденцию данных во времени с помощью одной или нескольких линий.	Показывает тренды во времени, например, изменения использования CPU, памяти и т. д.
2	Step Line Chart	Расширение линейного графика с соединением точек горизонтальными и вертикальными сегментами, образующими ступенчатую структуру.	Подходит для отображения временных меток дискретных событий, например, количества оповещений.
3	Bar Chart	Использует вертикальные прямоугольные столбцы для представления величины данных, высота столбцов соответствует значению.	Столбчатые диаграммы интуитивны для сравнения значений, полезны для выявления закономерностей и аномалий, подходят для сценариев с акцентом на изменение значений, например, количество подов, узлов и т. д.

№	Название панели	Описание	Рекомендуемые сценарии использования
4	Horizontal Bar Chart	Аналогично столбчатой диаграмме, но использует горизонтальные прямоугольные столбцы.	При большом количестве измерений данных горизонтальные диаграммы лучше используют пространство и повышают читаемость.
5	Gauge Chart	Использует полукруглые или кольцевые формы для отображения текущего значения индикатора и его доли от общего.	Интуитивно отражает текущее состояние ключевых индикаторов мониторинга, например, использование CPU и памяти системы. Рекомендуется использовать пороги оповещений с изменением цвета для обозначения аномалий.
6	Gauge Bar Chart	Использует вертикальные прямоугольные столбцы для отображения текущих значений индикаторов и их доли.	Интуитивно отражает текущее состояние ключевых индикаторов, например, прогресс выполнения задач и нагрузку системы. При наличии нескольких категорий одного индикатора предпочтительнее использовать gauge bar chart, например, доступное дисковое пространство.
7	Pie Chart	Использует сектора для отображения пропорционального соотношения частей и целого.	Подходит для демонстрации состава общих данных по разным измерениям, например, доли кодов ответов 4XX, 3XX и 2XX за период.
8	Table	Организует данные в формате строк и столбцов, облегчая просмотр и сравнение конкретных значений.	Подходит для отображения структурированных многомерных данных, например, подробной информации об узлах, подах и т. д.
9	Stat Chart	Отображает текущее значение одного ключевого индикатора,	Подходит для отображения в реальном времени важных показателей мониторинга, таких

№	Название панели	Описание	Рекомендуемые сценарии использования
		обычно с текстовым пояснением.	как количество подов, узлов, текущее число оповещений и т. д.
10	Scatter Plot	Использует декартовы координаты для отображения серии точек данных, отражая корреляцию между двумя переменными.	Подходит для анализа взаимосвязей между двумя индикаторами, выявления закономерностей, таких как линейная корреляция и кластеризация по распределению точек, помогает обнаружить связи между метриками.
11	Text Card	Отображает ключевую текстовую информацию в формате карточки, обычно содержит заголовок и краткое описание.	Подходит для представления текстовой информации, например, описаний панелей и пояснений по устранению неполадок.

Описание конфигурации панелей

Общие параметры

Параметр	Описание
Basic Information	Выберите подходящий тип панели в зависимости от выбранных метрик, добавьте заголовки и описания; можно добавить одну или несколько ссылок, к которым можно быстро перейти, выбрав имя ссылки рядом с заголовком.
Standard Settings	Единицы измерения для нативных метрик. Кроме того, gauge charts и gauge bars поддерживают настройку поля **Total Value**, которое отображается как процент от **Current Value/Total Value** на диаграмме.
Tooltips	Всплывающие подсказки для отображения данных в реальном времени при наведении на панели, поддерживают выбранную сортировку.

Параметр	Описание
Threshold Parameters	Настройка порогов для панелей; при включении пороги отображаются выбранными цветами, позволяя визуально оценивать значения.
Value	Настройка метода вычисления значения, например, последнее значение или минимальное. Применимо только к stat charts и gauge charts.
Value Mapping	Переопределение заданных значений, диапазонов, регулярных выражений или специальных значений, например, определение 100 как полного загрузки. Применимо к stat charts, таблицам и gauge charts.

Специальные параметры для панелей

Тип панели	Параметр	Описание
Trend Chart	**Graph Style**	Можно выбрать между линейным графиком и графиком с заливкой (area); линейные графики больше отражают изменения тренда, а area-графики акцентируют внимание на изменениях общей и частичной пропорции. Выбирайте в зависимости от потребностей.
Gauge Chart	**Gauge Chart Settings**	**Show Direction**: При необходимости просмотра нескольких метрик на одном графике можно задать горизонтальное или вертикальное расположение. **Unit Redefinition**: Можно задать отдельные единицы измерения для каждой метрики; если не задано, используется единица из **Standard Settings**.
Stat Chart	**Stat Chart Settings**	**Show Direction**: При необходимости просмотра нескольких метрик на одном графике можно задать горизонтальное или вертикальное расположение. **Graph Mode**: Можно добавить график в stat chart для отображения тренда метрики во времени.
Pie Chart	**Pie Chart Settings**	**Maximum Number of Slices**: Позволяет уменьшить количество секторов на круговой диаграмме, чтобы снизить

Тип панели	Параметр	Описание
		влияние категорий с низкой долей, но большим количеством. Избыточные сектора объединяются в **Others**. **Label Display Fields**: Настройка полей, отображаемых в метках круговой диаграммы.
Pie Chart	**Graph Style**	Можно выбрать стиль отображения: pie или donut.
Table	**Table Settings**	**Hide Columns**: Позволяет уменьшить количество столбцов в таблице для фокусировки на основных данных. **Column Alignment**: Настройка выравнивания данных в столбце. **Display Name and Unit**: Изменение названий столбцов и единиц измерения.
Text Card	**Graph Style**	**Style**: Можно выбрать редактирование содержимого текстовой карточки в формате rich-text или HTML.

Создание панелей мониторинга через CLI

1. Создайте новый YAML файл конфигурации с именем `example-dashboard.yaml`.
2. Добавьте ресурс MonitorDashboard в YAML файл и отправьте его. Пример ниже создаёт панель мониторинга с именем demo-v2-dashboard1:

```
kind: MonitorDashboard
apiVersion: ait.alauda.io/v1alpha2
metadata:
  annotations:
    cpaas.io/dashboard.version: '3'
    cpaas.io/description: '{"zh":"描述信息","en":""}' # Поле описания
    cpaas.io/operator: admin
  labels:
    cpaas.io/dashboard.folder: demo-v2-folder1 # Папка
    cpaas.io/dashboard.is.home.dashboard: 'False' # Является ли основны
м дашбордом?
  name: demo-v2-dashboard1 # Имя
  namespace: cpaas-system # Namespace (все создания вью управления прои
сходят в этом ns)
spec:
  body: # Все информационные поля
    titleZh: 更新显示名称 # Встроенное поле для отображения названия на к
итайском (создаётся для китайского языка)
    title: english_display_name # Встроенное поле для отображения назва
ния на английском (создаётся для английского языка) Встроенные дашборды
могут иметь двуязычный перевод.
    templating: # Пользовательские переменные
      list:
        - hide: 0 # 0 — не скрывать; 1 — скрывать только метку; 2 — скр
ывать метку и значение
          label: 集群 # Отображаемое имя встроенной переменной (метка за
даётся в зависимости от языка, например, cluster для английского)
          name: cluster # Имя встроенной переменной (уникальное)
          options: # Определение опций выпадающего списка; если запрос
возвращает данные, используются они, иначе — options. Можно задать знач
ение по умолчанию (обычно для установки дефолтных значений)
            - selected: false # Выбрано по умолчанию или нет
              text: global
              value: global
          type: custom # Тип переменной: custom (встроенная) или query
(поддерживаются только эти типы; импорт Grafana поддерживает constant c
ustom interval (после импорта преобразуется в custom переменную и не по
ддерживает авто))
        - allValue: '' # Выбрать все, передавая опции в формате xxx|xxx
|xxx; можно задать allValue для преобразования (Grafana получает все да
нные текущей переменной как xxx|xxx|xxx, корректировки обеспечивают сог
ласованность)
          current: null # Текущее значение переменной; если не задано,
```

```
по умолчанию первое в списке
          definition: query_result(kube_namespace_labels) # Выражение з
апроса для получения данных
          hide: 0 # 0 — не скрывать; 1 — скрывать только метку; 2 — скр
ывать метку и значение
          includeAll: true # Включить выбор всех
          label: ns # Отображаемое имя переменной
          multi: true # Разрешить множественный выбор
          name: ns # Имя переменной (уникальное)
          options: []
          query: ''
          regex: /.*namespace=\"(.*?)\".*/ # Регулярное выражение для и
звлечения значений переменной
          sort: 2 # Сортировка: 1 — по алфавиту по возрастанию; 2 — по
алфавиту по убыванию (поддерживаются только эти два); 3 — по числу по в
озрастанию; 4 — по числу по убыванию
          type: query # Тип переменной
    time: # Время дашборда
      from: now-30m # Время начала
      to: now # Время окончания
    repeat: '' # Конфигурация повторения строк; выбирается пользователь
ская переменная
    collapsed: 'false' # Конфигурация свёрнутости/развёрнутости строк
    description: '123' # Описание (всплывающая подсказка после заголовк
а)
    targets: # Источники данных
      - indicator: cluster.node.ready # Метрика
        expr: sum (cpaas_pod_number{cluster=\"\"}>0) # Выражение PromQL
        instant: false # Режим запроса: true — мгновенное значение
        legendFormat: '' # Легенда
        range: true # По умолчанию запрос диапазона при получении данны
х
        refId: 指标1 # Уникальный идентификатор для отображаемого имени
источника данных
    gridPos: # Позиционная компоновка дашборда
      h: 8 # Высота
      w: 12 # Ширина (ширина соответствует 24 сеточным единицам)
      x: 0 # Горизонтальная позиция
      y: 0 # Вертикальная позиция
    panels: # Данные панелей
      title: 图表标题tab # Название панели
      type: table # Тип панели; поддерживаются timeseries, barchart, st
at, gauge, table, bargauge, row, text, pie (step chart, scatter plot, b
ar chart настраиваются через атрибут drawStyle)
```

```
      id: a2239830-492f-4d27-98f3-cb7ecb77c56f # Уникальный идентификат
op
      links: # Ссылки
        - targetBlank: true # Открывать в новой вкладке
          title: '1' # Имя
          url: '1' # URL
      transformations: # Трансформации данных
        - id: 'organize' # Тип organize; используется для сортировки, п
ерестановки порядка, отображения полей, видимости
          options:
            excludeByName: # Скрытые поля
              cluster_cpu_utilization: true
            indexByName: # Сортировка
              cluster_cpu_utilization: 0,
              Time: 1
            renameByName: # Переименование
              Time: ''
              cluster_cpu_utilization: '222'
        - id: 'merge' # Объединение данных
          options:
      fieldConfig: # Определение свойств и внешнего вида панели
        defaults: # Конфигурация по умолчанию
          custom: # Пользовательские графические атрибуты
            align: 'left' # Выравнивание таблицы: left, center, right
            cellOptions: # Конфигурация порогов таблицы
              type: color-text # Поддерживается только текст для цветов
ой настройки порогов
            spanNulls: false # true — соединять null значения; false —
не соединять; число == 0 — соединять null значения как 0
            drawStyle: line # Типы панелей: line, bars для столбчатых,
points для точечных
            fillOpacity: 20 # Существует при drawStyle area (пока не по
ддерживается настройка, area по умолчанию 20)
            thresholdsStyle: # Настройка отображения порогов (пока подд
ерживается только линия)
              mode: line # Формат отображения порогов (area пока не под
держивается)
            lineInterpolation: 'stepBefore' # Настройка step chart; по
умолчанию поддерживается только stepBefore (stepAfter будет добавлен по
зже)
          decimals: 3 # Количество десятичных знаков
          min: 0 # Минимальное значение (пока не поддерживается в конфи
гурации страницы, только в адаптированных импортированных)
          max: 1 # Максимальное значение (конфигурация страницы применя
```

```
ется только к stat, gauge, barGauge, pie)
          unit: '%' # Единица измерения
          mappings: # Конфигурация сопоставления значений (поддерживают
ся только типы value и range; специальные типы поддерживаются в данных)
            - options: # Правила сопоставления значений
                '1': # Соответствующее значение
                  index: 0
                  text: 'Running' # Отображается как Running при значен
ии 1
              type: value # Тип сопоставления — значение
            - options: # Правила сопоставления диапазона
                from: 2 # Начало диапазона
                to: 3 # Конец диапазона
                result: # Результат сопоставления
                  index: 1
                  text: 'Error' # Значения от 2 до 3 отображаются как E
rror
              type: range # Тип сопоставления — диапазон
            - type: special # Тип сопоставления — специальные случаи
              options:
                match: null # nan null null+nan empty true false
                result:
                  text: xxx
                  index: 2
          thresholds: # Конфигурация порогов
            mode: absolute # Режим порогов, абсолютные значения (пока п
оддерживаются только абсолютные и процентные; процентные пока не поддер
живаются)
            steps: # Шаги порогов
              - color: '#a7772f' # Цвет порога
                value: '2' # Значение порога
              - color: '#007AF5' # Значение по умолчанию без значения —
базовое
        overrides: # Конфигурация переопределений
          - matcher:
              id: byName # Сопоставление по имени поля
              options: node # Соответствующее имя
            properties: # Конфигурация переопределений; id поддерживает
displayName и unit
              - id: displayName # Переопределение отображаемого имени
                value: '1' # Переопределённое имя
              - id: unit # Переопределение единицы
                value: GB/s # Значение единицы
              - id: noValue # Отображение при отсутствии значения
```

```
                value: No value display
      options:
        orientation: horizontal # Управление направлением компоновки па
нелей; применяется к gauge и barGauge (stat будет поддерживаться позже)
        legend: # Конфигурация легенды
          calcs: # Методы вычисления (отображаются только при расположе
нии легенды справа)
            - latest # Пока поддерживается только последнее значение
          placement: right # Положение легенды (right или bottom; по ум
олчанию bottom)
          placementRightTop: '' # Конфигурация верхнего правого угла
          showLegend: true # Отображать легенду или нет
        tooltip: # Всплывающие подсказки
          mode: multi # Режим мультивыбора (поддерживается только mult
i); отображаются все данные при наведении мыши
          sort: asc # Сортировка: asc или desc
        reduceOptions: # Метод вычисления значения (для агрегации данны
х)
          calcs: # Методы вычисления (latest, minimum, maximum, averag
e, sum)
            - latest
          limit: 3 # Ограничение количества секторов в pie chart
        textMode: 'value' # Конфигурация stat; определяет стиль отображ
ения значения метрики; варианты: auto, value, value_and_name, name, non
e (пока не поддерживается в конфигурации страницы, но поддерживается пр
и импорте)
        colorMode: 'value' # Конфигурация stat; определяет режим цвета
для отображения значений метрик; варианты: none, value, background (по
умолчанию value; не поддерживается в конфигурации, но адаптируется при
импорте)
        displayLabels: ['name', 'value', 'percent'] # Поля, отображаемы
е в метках pie chart
        pieType: 'pie' # Тип pie chart; варианты: pie и donut
        mode: 'html' # Режим text chart; варианты: html и richText
        content: '<div>xxx</div>' # Содержимое text chart
        footer:
          enablePagination: true # Включение пагинации в таблице
```

Общие функции и переменные

Общие функции

При определении настроек запроса, помимо использования PromQL, платформа предоставляет следующие общие функции для удобства настройки:

Функция	Назначение
label_names()	Возвращает все метки в Prometheus, например, label_names().
label_values(label)	Возвращает все доступные значения для имени метки во всех мониторимых метриках Prometheus, например, label_values(job).
label_values(metric, label)	Возвращает все доступные значения для имени метки в указанной метрике Prometheus, например, label_values(up, job).
metrics(metric)	Возвращает все имена метрик, удовлетворяющих заданному регулярному выражению, например, metrics(cpaas_active).
query_result(query)	Возвращает результат запроса для указанного Prometheus запроса, например, query_result(up).

Общие переменные

При определении настроек запроса можно комбинировать общие функции в переменные для быстрого создания пользовательских переменных. Ниже приведены распространённые определения переменных для справки:

Имя переменной	Функция запроса
cluster	`label_values(cpaas_cluster_info,cluster)`
node	`label_values(node_load1, instance)`
namespace	`query_result(kube_namespace_labels)`
deployment	`label_values(kube_deployment_spec_replicas{namespace="$namespace"}, deployment)`

Имя переменной	Функция запроса
daemonset	`label_values(kube_daemonset_status_number_ready{namespace="$namespace"}, daemonset)`
statefulset	`label_values(kube_statefulset_replicas{namespace="$namespace"}, statefulset)`
pod	`label_values(kube_pod_info{namespace=~"$namespace"}, pod)`
vmcluster	`label_values(up, vmcluster)`
daemonset	`label_values(kube_daemonset_status_number_ready{namespace="$namespace"}, daemonset)`

Пример использования переменной 1

Использование функции **query_result(query)** для запроса значения: `node_load5` и извлечения IP.

1. В **Query Settings** заполните `query_result(node_load5)`.
2. В области **Variable Value Preview** пример превью: `node_load5{container="node-exporter",endpoint="metrics",host_ip="192.168.178.182",instance="192.168.178.182:9100"}`.
3. В **Regular Expression** заполните `/.*instance="(.*?):.*/` для фильтрации значения.
4. В области **Variable Value Preview** пример превью: `192.168.176.163`.

Пример использования переменной 2

1. Добавьте первую переменную: namespace, используя функцию **query_result(query)** для запроса значения: `kube_namespace_labels` и извлечения namespace.
   - **Query Settings**: `query_result(kube_namespace_labels)`.

- **Variable Value Preview**: `kube_namespace_labels{container="exporter-kube-state", endpoint="kube-state-metrics", instance="12.3.188.121:8080", job="kube-state", label_cpaas_io_project="cpaas-system", namespace="cert-manager", pod="kube-prometheus-exporter-kube-state-55bb6bc67f-lpgtx", project="cpaas-system", service="kube-prometheus-exporter-kube-state"}`.
- **Regular Expression**: `/.+namespace=\"(.*?)\".*/`.
- В области **Variable Value Preview** пример превью включает несколько namespace, таких как `argocd`, `cpaas-system` и другие.

2. Добавьте вторую переменную: deployment, ссылаясь на ранее созданную переменную:

   - **Query Settings**: `kube_deployment_spec_replicas{namespace=~"$namespace"}`.
   - **Regular Expression**: `/.+deployment="(.*?)",.*/`.

3. Добавьте панель в текущий дашборд и используйте ранее добавленные переменные, например:

   - Имя метрики: **pod Memory Usage under Compute Components**.
   - Пара ключ-значение: `kind`: `Deployment`, `name`: `$deployment`, `namespace`: `$namespace`.

4. После добавления и сохранения панелей вы можете просмотреть соответствующую информацию на главной странице дашборда.

Примечания при использовании встроенных метрик

> **WARNING**
>
> Следующие метрики используют пользовательские переменные `namespace`, `name` и `kind`, которые не поддерживают **множественный выбор** или выбор **всех**.
>
> - `namespace` поддерживает выбор только одного конкретного namespace;
> - `name` поддерживает только три типа вычислительных компонентов: `deployment`, `daemonset`, `statefulset`;

- `kind` поддерживает указание только одного из типов: `Deployment`, `DaemonSet`, `StatefulSet`.

- `workload.cpu.utilization`
- `workload.memory.utilization`
- `workload.network.receive.bytes.rate`
- `workload.network.transmit.bytes.rate`
- `workload.gpu.utilization`
- `workload.gpu.memory.utilization`
- `workload.vgpu.utilization`
- `workload.vgpu.memory.utilization`

Управление Probe

Содержание

- Обзор функции
- Blackbox мониторинг
  - Предварительные требования
  - Порядок действий
- Оповещения Blackbox
  - Предварительные требования
  - Порядок действий
- Настройка модуля мониторинга BlackboxExporter
  - Порядок действий
- Создание элементов Blackbox мониторинга и оповещений через CLI
  - Предварительные требования
  - Порядок действий
- Справочная информация

Обзор функции

Функция probe платформы реализована на основе Blackbox Exporter, позволяя пользователям выполнять проверку сети через ICMP, TCP или HTTP для быстрого выявления сбоев, происходящих на платформе.

В отличие от систем white-box мониторинга, которые опираются на различные метрики мониторинга, уже доступные на платформе, blackbox мониторинг фокусируется на результатах. Когда white-box мониторинг не может охватить все факторы, влияющие на доступность сервиса, blackbox мониторинг может быстро обнаружить сбои и сгенерировать оповещения на основе этих сбоев. Например, если конечная точка API работает ненормально, blackbox мониторинг может оперативно выявить такие проблемы для пользователей.

> **WARNING**
>
> Функция probe не поддерживает использование ICMP для обнаружения IPv6-адресов на узлах с версиями ядра 3.10 и ниже. Для использования данного сценария необходимо обновить версию ядра на узле до 3.11 или выше.

Blackbox мониторинг

Для создания элемента blackbox мониторинга можно выбрать метод проверки ICMP, TCP или HTTP для периодического опроса указанного целевого адреса.

Предварительные требования

Компоненты мониторинга должны быть установлены в кластере и функционировать корректно.

Порядок действий

1. В левой навигационной панели нажмите **Operations Center** > **Monitoring** > **Blackbox Monitoring**.
   Совет: Blackbox мониторинг является функцией на уровне кластера. Для переключения между кластерами используйте верхнюю панель навигации.
2. Нажмите **Create Blackbox Monitoring Item**.
3. Настройте соответствующие параметры согласно следующим инструкциям.

Параметр	Описание
Метод проверки	**ICMP**: Проверка путем пинга доменного имени или IP-адреса, введенного в поле **Target Address**, для проверки доступности сервера. **TCP**: Проверка бизнес-порта хоста путем прослушивания `<domain:port>` или `<IP:port>`, указанного в **Target Address**. **HTTP**: Проверка URL, введенного в **Target Address**, для проверки доступности веб-сайта. **Совет**: Метод HTTP по умолчанию поддерживает только GET-запросы; для POST-запросов смотрите Настройка модуля мониторинга BlackboxExporter.
Интервал проверки	Интервал времени между проверками.
Целевой адрес	Целевой адрес для проверки, максимум 128 символов. Формат ввода зависит от метода проверки: **ICMP**: доменное имя или IP-адрес, например, `10.165.94.31`. **TCP**: `<domain:port>` или `<IP:port>`, например, `172.19.155.133:8765`. **HTTP**: URL, начинающийся с http или https, например, `http://alauda.cn/`.

4. Нажмите **Create**.

   После успешного создания вы можете в реальном времени просматривать последние результаты проверки на странице списка, а также на основе элементов blackbox мониторинга создавать политики оповещений. При обнаружении сбоя автоматически сработает оповещение для уведомления ответственных лиц о необходимости устранения.

> **WARNING**
>
> После успешного создания элементов blackbox мониторинга системе требуется около 5 минут для синхронизации конфигурации. В течение этого времени проверки не выполняются, и результаты проверки недоступны для просмотра.

Оповещения Blackbox

Предварительные требования

- Компоненты мониторинга должны быть установлены в кластере и функционировать корректно.
- Элемент blackbox мониторинга должен быть успешно создан, и система должна завершить синхронизацию конфигурации, чтобы результаты проверки были видны на странице blackbox мониторинга.

Порядок действий

1. В левой навигационной панели нажмите **Operations Center** > **Alerts** > **Alert Policies**. **Совет**: Политики оповещений являются функцией на уровне кластера. Для переключения между кластерами используйте верхнюю панель навигации. Убедитесь, что вы переключились на кластер, в котором только что настроен элемент blackbox мониторинга.
2. Нажмите **Create Alert Policy**.
3. Настройте соответствующие параметры согласно следующим инструкциям; для получения дополнительной информации о параметрах смотрите Создание политик оповещений.

- **Тип оповещения**: выберите **Resource Alert**.
- **Тип ресурса**: выберите **Cluster**.
- Нажмите **Add Alert Rule**.
  - **Тип оповещения**: выберите **Blackbox Alert**.
  - **Элемент blackbox мониторинга**: выберите нужный элемент blackbox мониторинга.
  - **Имя метрики**: выберите метрику, которую хотите контролировать и по которой будет срабатывать оповещение. В настоящее время платформа поддерживает метрики **Connectivity** и **HTTP Status Code**.
    - **Connectivity**: доступна для всех элементов blackbox мониторинга, условие срабатывания **“!= 1”** означает, что целевой адрес элемента blackbox

мониторинга недоступен.

- **HTTP Status Code**: доступна, если метод проверки выбранного элемента blackbox мониторинга — **HTTP**. Можно указать трехзначное положительное число в качестве значения условия срабатывания, например, при условии **“> 299”** оповещения срабатывают при кодах ответа 3XX, 4XX или 5XX.

- **Политика уведомлений**: выберите заранее настроенную политику.
- Нажмите **Add**.

4. Нажмите **Create**. После отправки политики оповещений она появится в списке политик.

Настройка модуля мониторинга BlackboxExporter

Вы также можете расширить функциональность blackbox мониторинга, добавляя настраиваемые модули мониторинга в конфигурационный файл BlackboxExporter. Например, добавив модуль **http_post_2xx** в конфигурационный файл, при выборе метода проверки blackbox мониторинга `HTTP` будет возможна проверка статуса POST-запросов.

Конфигурационный файл blackbox мониторинга находится в namespace, где установлен компонент Prometheus кластера, по умолчанию называется `cpaas-monitor-prometheus-blackbox-exporter`, имя можно изменить в зависимости от фактического.

> **TIP**
>
> Этот конфигурационный файл является ресурсом ConfigMap, связанным с namespace, который можно быстро просмотреть и обновить через функцию управления платформы **Cluster Management > Resource Management**.

Порядок действий

1. Обновите конфигурационный файл blackbox мониторинга, добавив настраиваемые модули мониторинга в **key** `modules`.

В качестве примера добавления модуля **http_post_2xx**:

```
blackbox.yaml: |
  modules:
    http_post_2xx:                    # Модуль проверки HTTP POST
      prober: http
      timeout: 5s
      http:
        method: POST                  # Метод запроса для проверки
        headers:
          Content-Type: application/json
        body: '{}'                    # Тело запроса, отправляемое при п
роверке
```

Полные примеры YAML конфигурации файла blackbox мониторинга смотрите в Справочной информации.

2. Активируйте конфигурацию одним из следующих способов.

   - Перезапустите компонент Blackbox Exporter **cpaas-monitor-prometheus-blackbox-exporter**, удалив его Pod.
   - Выполните следующую команду для вызова API перезагрузки и обновления конфигурационного файла:

     ```
     curl -X POST -v <Pod IP>:9115/-/reload
     ```

Создание элементов Blackbox мониторинга и оповещений через CLI

Предварительные требования

- Политики уведомлений должны быть настроены (если требуется автоматическая отправка оповещений).
- В целевом кластере должны быть установлены компоненты мониторинга.

Порядок действий

1. Создайте новый YAML-файл конфигурации с именем `example-probe.yaml`.
2. Добавьте ресурс PrometheusRule в YAML-файл и отправьте его. В следующем примере создается новая политика оповещений с именем `prometheus-liveness`:

```yaml
apiVersion: monitoring.coreos.com/v1
kind: Probe
metadata:
  annotations:
    cpaas.io/creator: jhshi@alauda.io # Создатель элемента probe
    cpaas.io/updated-at: '2021-05-25T08:08:45Z' # Время последнего обно
вления элемента probe
    cpaas.io/display-name: 'Prometheus prober' # Описание элемента prob
e
  creationTimestamp: '2021-05-10T02:04:33Z' # Время создания элемента p
robe
  labels:
    prometheus: kube-prometheus # Значение метки, используемой для имен
и prometheus
  name: prometheus-liveness # Имя элемента probe
  namespace: cpaas-system # Namespace, используемый для namespace prome
theus
spec:
  jobName: prometheus-liveness # Имя элемента probe
  prober:
    url: cpaas-monitor-prometheus-blackbox-exporter:9115 # URL для метр
ик Blackbox, полученный из features
  module: http_2xx # Имя модуля элемента probe
  targets:
    staticConfig:
      static:
        - http://www.prometheus.io # Целевой адрес элемента probe
      labels:
        module: http_2xx # Имя модуля элемента probe
        prober: http # Метод проверки элемента probe
  interval: 30s # Интервал проверки элемента probe
  scrapeTimeout: 10s
```

3. Создайте новый YAML-файл конфигурации с именем `example-alerting-rule.yaml`.

4. Добавьте ресурс PrometheusRule в YAML-файл и отправьте его. В следующем примере создается новая политика оповещений с именем `policy`:

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  annotations:
    alert.cpaas.io/cluster: global # Имя кластера, в котором находится оповещение
    alert.cpaas.io/kind: Cluster # Тип ресурса
    alert.cpaas.io/name: global # Имя кластера, в котором находится элемент blackbox мониторинга
    alert.cpaas.io/namespace: cpaas-system # Namespace, используемый для namespace prometheus, оставьте по умолчанию
    alert.cpaas.io/notifications: '["test"]'
    alert.cpaas.io/repeat-config: '{"Critical":"never","High":"5m","Medium":"5m","Low":"5m"}'
    alert.cpaas.io/rules.description: '{}'
    alert.cpaas.io/rules.disabled: '[]'
    alert.cpaas.io/subkind: ''
    cpaas.io/description: ''
    cpaas.io/display-name: policy # Отображаемое имя политики оповещений
  labels:
    alert.cpaas.io/owner: System
    alert.cpaas.io/project: cpaas-system
    cpaas.io/source: Platform
    prometheus: kube-prometheus
    rule.cpaas.io/cluster: global
    rule.cpaas.io/name: policy
    rule.cpaas.io/namespace: cpaas-system
  name: policy
  namespace: cpaas-system
spec:
  groups:
    - name: general # Имя правил оповещений
      rules:
        - alert: cluster.blackbox.probe.success-y97ah-9833444d918cab96c43e9ab6efc172cf
          annotations:
            alert_current_value: '{{ $value }}' # Текущее значение для уведомления, оставьте по умолчанию
          expr:
            max by (job, instance) (probe_success{job=~"test",
            instance=~"https://demo.at-servicecenter.com/"})!=1
            # Сценарий оповещения по доступности, обязательно измените
```

```
имя элемента blackbox мониторинга и целевой адрес
          for: 30s # Продолжительность
          labels:
            alert_cluster: global # Имя кластера, в котором находится о
повещение
            alert_for: 30s # Продолжительность
            alert_indicator: cluster.blackbox.probe.success # Оставьте
без изменений
            alert_indicator_aggregate_range: '0' # Оставьте без изменен
ий
            alert_indicator_blackbox_instance: https://demo.at-servicec
enter.com/ # Целевой адрес blackbox мониторинга
            alert_indicator_blackbox_name: test # Имя элемента blackbox
мониторинга
            alert_indicator_comparison: '!=' # Оставьте без изменений д
ля оповещений по доступности
            alert_indicator_query: '' # Используется для оповещений по
логам, не нужно настраивать
            alert_indicator_threshold: '1' # Пороговое значение правила
оповещения, 1 означает доступность, оставьте без изменений
            alert_indicator_unit: '' # Единица измерения метрик правила
оповещения
            alert_involved_object_kind: Cluster # Оставьте без изменени
й для blackbox оповещений
            alert_involved_object_name: global # Кластер, в котором нах
одится элемент blackbox мониторинга
            alert_involved_object_namespace: '' # Namespace объекта, к
которому относится правило оповещения
            alert_name: cluster.blackbox.probe.success-y97ah # Имя прав
ила оповещения
            alert_namespace: cpaas-system # Namespace, в котором находи
тся правило оповещения
            alert_project: cpaas-system # Имя проекта объекта, к которо
му относится правило оповещения
            alert_resource: policy # Имя политики оповещений, в которой
находится правило
            alert_source: Platform # Тип данных правила оповещения: Pla
tform — данные платформы, Business — бизнес-данные
            severity: High # Уровень правила оповещения: Critical — кри
тический, High — высокий, Medium — средний, Low — низкий
        - alert: cluster.blackbox.http.status.code-235el-99b0095b6b6669
415043e14ae84f43bc
          annotations:
            alert_current_value: '{{ $value }}'
```

```
            alert_notifications: '["message"]'
          expr:
            max by(job, instance) (probe_http_status_code{job=~"test",
            instance=~"https://demo.at-servicecenter.com/"})>200
            # Сценарий оповещения по HTTP статус-коду, обязательно изме
ните имя элемента blackbox мониторинга и целевой адрес
          for: 30s
          labels:
            alert_cluster: global
            alert_for: 30s
            alert_indicator: cluster.blackbox.http.status.code
            alert_indicator_aggregate_range: '0'
            alert_indicator_blackbox_instance: https://demo.at-servicec
enter.com/
            alert_indicator_blackbox_name: test
            alert_indicator_comparison: '>'
            alert_indicator_query: ''
            alert_indicator_threshold: '299' # Пороговое значение для п
равил оповещений, в сценариях оповещений по HTTP статус-кодам должно бы
ть трехзначное число, например, статусы больше 299 (3XX, 4XX, 5XX) озна
чают ошибку
            alert_indicator_unit: ''
            alert_involved_object_kind: Cluster
            alert_involved_object_name: global
            alert_involved_object_namespace: ''
            alert_involved_object_options: Single
            alert_name: cluster.blackbox.http.status.code-235el
            alert_namespace: cpaas-system
            alert_project: cpaas-system
            alert_resource: policy33
            alert_source: Platform
            severity: High
```

Справочная информация

Полный пример YAML конфигурационного файла для blackbox мониторинга приведен ниже:

```
apiVersion: v1
data:
  blackbox.yaml: |
    modules:
      http_2xx_example:               # Пример HTTP проверки
        prober: http
        timeout: 5s                   # Таймаут проверки
        http:
          valid_http_versions: ["HTTP/1.1", "HTTP/2.0"]
# Версия в возвращаемой информации, обычно по умолчанию
          valid_status_codes: []  # По умолчанию 2xx
# Диапазон допустимых кодов ответа; если возвращенный код входит в этот д
иапазон, проверка считается успешной
          method: GET                 # Метод запроса
          headers:                    # Заголовки запроса
            Host: vhost.example.com
            Accept-Language: en-US
            Origin: example.com
          no_follow_redirects: false  # Разрешать ли перенаправления
          fail_if_ssl: false
          fail_if_not_ssl: false
          fail_if_body_matches_regexp:
            - "Could not connect to database"
          fail_if_body_not_matches_regexp:
            - "Download the latest version here"
          fail_if_header_matches: # Проверка отсутствия установки cookies
            - header: Set-Cookie
              allow_missing: true
              regexp: '.*'
          fail_if_header_not_matches:
            - header: Access-Control-Allow-Origin
              regexp: '(\*|example\.com)'
          tls_config:                  # TLS конфигурация для https запро
сов
            insecure_skip_verify: false
          preferred_ip_protocol: "ip4" # по умолчанию "ip6"
# Предпочитаемая версия IP протокола
          ip_protocol_fallback: false  # Без fallback на "ip6"
      http_post_2xx:                   # Пример HTTP проверки с телом зап
роса
        prober: http
        timeout: 5s
        http:
```

```
method: POST                    # Метод запроса для проверки
headers:
  Content-Type: application/json
body: '{"username":"admin","password":"123456"}'
```

Как сделать

Резервное копирование и вос Prometheus

- Обзор функции
- Сценарии использования
- Предварительные требования
- Процедуры работы
- Результаты операции
- Дополнительная информация
- Следующие шаги

Резервное копирование и вос VictoriaMetrics

- Обзор функции
- Сценарии использования
- Предварительные требования
- Процедуры
- Результат операции
- Дополнительная информация
- Последующие действия

Сбор сетев именами

- Обзор функции
- Сценарий испо
- Предварительны
- Порядок дейст
- Результаты опе
- Дополнительна
- Последующие

Резервное копирование и восстановление данных мониторинга Prometheus

Содержание

- Обзор функции
- Сценарии использования
- Предварительные требования
- Процедуры работы
  - Резервное копирование данных
    - Метод 1: Резервное копирование каталога хранения (рекомендуется)
    - Метод 2: Резервное копирование с помощью снимка (snapshot)
  - Восстановление данных
- Результаты операции
- Дополнительная информация
  - Описание формата данных TSDB
  - Особенности резервного копирования данных
- Следующие шаги

Обзор функции

Данные мониторинга Prometheus хранятся в формате TSDB (Time Series Database) и поддерживают функции резервного копирования и восстановления. Данные мониторинга сохраняются в заданном пути внутри контейнера Prometheus (указывается в конфигурации `storage.tsdb.path`, по умолчанию `/prometheus`).

```
template:
  spec:
    containers:
      - args:
          - '--storage.tsdb.path=/prometheus' # Каталог для хранения данн
ых мониторинга в контейнере Prometheus
```

Сценарии использования

- Сохранение исторических данных мониторинга при миграции системы
- Предотвращение потери данных из-за непредвиденных инцидентов
- Миграция данных мониторинга на новый экземпляр Prometheus

Предварительные требования

- Установлен плагин ACP Monitoring с Prometheus (имя вычислительного компонента — `prometheus-kube-prometheus-0`, тип — `StatefulSet`)
- Привилегии администратора кластера
- Достаточно свободного места для хранения данных в целевом расположении

Процедуры работы

Резервное копирование данных

Перед началом резервного копирования обратите внимание: когда Prometheus сохраняет данные мониторинга, он сначала помещает собранные данные в кэш, а затем периодически записывает их в каталог хранения. Следующие методы резервного

копирования используют каталог хранения в качестве источника данных, поэтому данные, находящиеся в кэше на момент резервного копирования, не включаются.

Метод 1: Резервное копирование каталога хранения (рекомендуется)

1. Используйте команду kubectl cp для резервного копирования:

```
kubectl cp -n cpaas-system prometheus-kube-prometheus-0-0:/prometheus -c prometheus <target storage path>
```

2. Резервное копирование из бекенда хранения (в зависимости от типа хранилища, выбранного при установке):

- **LocalVolume**: копировать из каталога `/cpaas/monitoring/prometheus`
- **PV**: копировать из каталога монтирования PV (рекомендуется установить для PV параметр **persistentVolumeReclaimPolicy** в значение `Retain`)
- **StorageClass**: копировать из каталога монтирования PV

Метод 2: Резервное копирование с помощью снимка (snapshot)

1. Включите Admin API:

```
kubectl edit -n cpaas-system prometheus kube-prometheus-0
```

Добавьте конфигурацию:

```
spec:
  enableAdminAPI: true
```

Примечание: после обновления и сохранения конфигурации Pod Prometheus (имя Pod: prometheus-kube-prometheus-0-0) перезапустится. Дождитесь, пока все Pods перейдут в состояние Running, прежде чем выполнять дальнейшие действия.

2. Создайте снимок:

```
curl -XPOST <Prometheus Pod IP>:9090/api/v1/admin/tsdb/snapshot
```

Восстановление данных

1. Скопируйте резервные данные в контейнер Prometheus:

   ```
   kubectl cp ./prometheus-backup cpaas-system/prometheus-kube-prometheus-0-0:/prometheus/
   ```

2. Переместите данные в указанный каталог:

   ```
   kubectl exec -it -n cpaas-system prometheus-kube-prometheus-0-0 -c prometheus sh
   mv /prometheus/prometheus-backup/* /prometheus/
   ```

   Упрощение: если при установке плагина тип хранилища — **LocalVolume**, достаточно скопировать резервные данные напрямую в каталог `/cpaas/monitoring/prometheus/prometheus-db/` на узле, где установлен плагин.

Результаты операции

- После завершения резервного копирования в целевом каталоге будет доступен полный набор данных мониторинга в формате TSDB
- После завершения восстановления Prometheus автоматически загрузит исторические данные мониторинга

Дополнительная информация

Описание формата данных TSDB

Пример структуры данных в формате TSDB:

```
├── 01FXP317QBANGAX1XQAXCJK9DB
│   ├── chunks
│   │   └── 000001
│   ├── index
│   ├── meta.json
│   └── tombstones
├── chunks_head
│   ├── 000022
│   └── 000023
├── queries.active
└── wal
    ├── 00000020
    ├── 00000021
    ├── 00000022
    ├── 00000023
    └── checkpoint.00000019
        └── 00000000
```

Особенности резервного копирования данных

- Резервные данные не включают кэшированные данные на момент резервного копирования
- Рекомендуется выполнять резервное копирование данных регулярно
- При использовании PV-хранилища рекомендуется установить параметр **persistentVolumeReclaimPolicy** в значение `Retain`

Следующие шаги

- Проверить корректность восстановления данных мониторинга
- Регулярно планировать задачи резервного копирования данных
- При использовании метода резервного копирования со снимком можно отключить Admin API после завершения операций восстановления

Резервное копирование и восстановление данных мониторинга VictoriaMetrics

Содержание

- Обзор функции
- Сценарии использования
- Предварительные требования
- Процедуры
  1. Подтвердите путь хранения
  2. Выполните резервное копирование данных
  3. Выполните восстановление данных
- Результат операции
- Дополнительная информация
- Последующие действия

Обзор функции

Функция резервного копирования и восстановления данных мониторинга VictoriaMetrics позволяет выполнять регулярное резервное копирование данных мониторинга и восстанавливать данные при необходимости, обеспечивая безопасность и доступность данных мониторинга.

Сценарии использования

- Регулярное резервное копирование данных мониторинга для предотвращения потери данных
- Миграция данных при переносе системы
- Восстановление после сбоев
- Воссоздание данных тестовой среды

Предварительные требования

- В кластере установлен плагин ACP Monitoring с VictoriaMetrics
- Обеспечено достаточное пространство для хранения резервных копий
- Имеется доступ к пути хранения данных VictoriaMetrics

Процедуры

1. Подтвердите путь хранения

Данные мониторинга VictoriaMetrics хранятся в указанном пути контейнера, который задаётся параметром `-storageDataPath`, по умолчанию `/vm-data`.

Пример конфигурации:

```
spec:
  template:
    spec:
      containers:
        - args:
            - '-storageDataPath=/vm-data'
```

Примечание: Имя вычислительного компонента в плагине ACP Monitoring с VictoriaMetrics — `vmstorage-cluster`, тип — `StatefulSet`.

2. Выполните резервное копирование данных

Для резервного копирования используйте инструмент vmbackup; подробные инструкции см. в официальной документации vmbackup ↗.

3. Выполните восстановление данных

Для восстановления резервных данных используйте инструмент vmrestore; подробные инструкции см. в официальной документации vmrestore ↗.

Результат операции

После завершения резервного копирования вы получите полный файл резервной копии данных мониторинга. После выполнения операции восстановления ваши данные мониторинга будут восстановлены в состояние на момент создания резервной копии.

Дополнительная информация

- Официальная документация VictoriaMetrics ↗
- Лучшие практики резервного копирования данных ↗
- Устранение неполадок при восстановлении данных ↗

Последующие действия

- Проверить целостность резервных данных
- Настроить регулярное расписание резервного копирования
- Периодически тестировать процесс восстановления
- Контролировать статус выполнения задач резервного копирования

Сбор сетевых данных с сетевых интерфейсов с пользовательскими именами

Содержание

- Обзор функции
- Сценарий использования
- Предварительные условия
- Порядок действий
- Результаты операции
- Дополнительная информация
- Последующие действия

Обзор функции

Платформа поддерживает сбор сетевых данных с сетевых интерфейсов с пользовательскими именами путем изменения конфигурации компонента мониторинга, что позволяет просматривать информацию о сетевом трафике этих интерфейсов на странице мониторинга.

Сценарий использования

Это актуально, когда на ваших узлах используются сетевые интерфейсы с пользовательскими именами (не соответствующие шаблонам `eth.|en.|wl.*|ww.*`) и требуется мониторинг данных сетевого трафика этих интерфейсов в платформе.

Предварительные условия

- Создан кластер рабочих нагрузок
- У вас есть права администратора платформы
- Известны соглашения об именовании пользовательских сетевых интерфейсов

Порядок действий

1. Нажмите на иконку CLI-инструмента в верхней навигационной панели платформы.
2. Выберите **global**.
3. В кластере `global` найдите имя ресурса moduleinfo, соответствующее вашему кластеру рабочих нагрузок:

   ```
   kubectl get moduleinfo | grep -E 'prometheus|victoriametrics'
   ```

   Пример вывода:

   ```
   global-6448ef7f7e5e3924c1629fad826372e7     global     prometheus
   prometheus                          Running   v3.15.0-zz231204040711-9d
   1fc12474c2   v3.15.0-zz231204040711-9d1fc12474c2   v3.15.0-zz2312040407
   11-9d1fc12474c2
   ovn-0954f21f0359720e8c115804376b3e7e        ovn        prometheus
   prometheus                          Running   v3.15.0-zz231204040711-9d
   1fc12474c2   v3.15.0-zz231204040711-9d1fc12474c2   v3.15.0-zz2312040407
   11-9d1fc12474c2
   ```

4. Отредактируйте ресурс moduleinfo кластера рабочих нагрузок:

```
kubectl edit moduleinfo <имя ресурса moduleinfo кластера рабочих нагруз
ок>
```

5. Добавьте или измените поле valuesOverride:

```
spec:
  valuesOverride:# Если этого поля нет, необходимо добавить valuesOverr
ide под spec вместе с параметрами ниже
    ait/chart-cpaas-monitor:
      global:
        indicator:
          networkDevice: eth.*|em.*|en.*|wl.*|ww.*|[A-Z].*i|custom_inte
rface # Замените custom_interface на пользовательское регулярное выраже
ние для корректного сопоставления имени сетевого интерфейса
```

6. После ожидания 10 минут проверьте сетевые графики на странице мониторинга узла, чтобы убедиться, что изменения вступили в силу.

Результаты операции

После применения конфигурации вы сможете просматривать на странице мониторинга платформы следующие данные сетевых интерфейсов с пользовательскими именами:

- Данные сетевого трафика
- Пропускная способность сети
- Статистика сетевых пакетов

Дополнительная информация

- Для получения дополнительной информации о мониторинге сети обратитесь к Документации по архитектуре мониторинга

Последующие действия

- Отслеживайте показатели производительности пользовательских сетевых интерфейсов
- Настройте правила оповещений на основе данных мониторинга

Распределённое трассирование

Введение

Введение

Ограничения использования

Установка

Установка

Установка Jaeger Operator

Развертывание экземпляра Jaeger

Установка OpenTelemetry Operator

Развертывание экземпляров OpenTelemetry

Включение переключателя функций

Удаление Tracing

Архитектура

Архитектура

Основные компоненты

Поток данных

Основные понятия

Основные понятия

Telemetry

OpenTelemetry

Span

Trace

Instrumentation

OpenTelemetry Collector

Jaeger

Руководства

Query Tracing

Feature Overview

Main Features

Feature Advantages

Tracing Query

Query Result Analysis

Query Trace Logs

Обзор функции

Основные возможности

Предварительные требования

Операции с запросами логов

Как сделать

Беспрепятственная интеграц

- Обзор возможностей
- Сценарии использования
- Предварительные требования
- Шаги для настройки
- Результаты работы

Бизнес-логи, связанные с TraceID

- Предпосылки
- Добавление TraceID в логи Java-приложения
- Добавление TraceID в логи Python-приложения
- Метод проверки

Устранение неполадок

Невозможно выполнить запр

- Описание проблемы
- Анализ первопричин
- Решение для первопричины 1
- Решение для первопричины 2

Неполные данные трассировки

- Описание проблемы
- Анализ причин
- Решение для причины 1
- Решение для причины 2

Введение

Модуль Distributed Tracing является ключевым компонентом набора средств наблюдаемости платформы ACP, обеспечивающим сквозное отслеживание и анализ запросов в распределённых микросервисных архитектурах.

Этот модуль предоставляет четыре основные возможности трассировки:

- **Сбор трассировок** для автоматизированного сбора данных о распределённых запросах с помощью автоматической инъекции OpenTelemetry или интеграции SDK
- **Хранение трассировок** для масштабируемого сохранения данных трассировки с использованием Elasticsearch в качестве бэкенд-хранилища
- **Визуализация трассировок** для многомерного анализа через настраиваемые UI-дашборды и отображение зависимостей сервисов
- **Запрос трассировок** для точного поиска и фильтрации по TraceID, именам сервисов, тегам и другим расширенным условиям поиска

Интегрируя эти возможности со стандартами OpenTelemetry и компонентами с открытым исходным кодом, модуль позволяет организациям быстро выявлять аномалии сервисов, анализировать узкие места производительности, отслеживать полный жизненный цикл запросов и оптимизировать производительность распределённых систем в рамках их микросервисной архитектуры.

Содержание

Ограничения использования

Ограничения использования

При использовании трассировки следует учитывать следующие ограничения:

- **Балансировка стратегий сэмплирования и производительности**
  - В условиях высокой нагрузки сбор данных трассировки может оказывать определённое давление на производительность и хранение в Elasticsearch; рекомендуется разумно настраивать коэффициент сэмплирования в зависимости от бизнес-условий.

Установка

> **WARNING**
>
> **Этот документ по развертыванию применим только к сценариям интеграции платформы контейнеров с системой трассировки.**
>
> Компоненты **Tracing** и **Service Mesh** являются взаимоисключающими. Если вы уже развернули компонент Service Mesh, пожалуйста, сначала удалите его.

Данное руководство предоставляет администраторам кластера процесс установки системы трассировки на кластер Alauda Container Platform.

Требования:

- У вас есть доступ к кластеру Alauda Container Platform с учетной записью, обладающей правами `platform-admin-system`.
- У вас установлен CLI `kubectl`.
- Компонент `Elasticsearch` настроен для хранения данных трассировки, включая URL доступа и информацию `Basic Auth`.

Содержание

Установка Jaeger Operator

Установка Jaeger Operator через веб-консоль

Развертывание экземпляра Jaeger

Установка OpenTelemetry Operator

Установка OpenTelemetry Operator через веб-консоль

Развертывание экземпляров OpenTelemetry

Включение переключателя функций

Удаление Tracing

Удаление экземпляра OpenTelemetry

Удаление OpenTelemetry Operator

Удаление экземпляра Jaeger

Удаление Jaeger Operator

Установка Jaeger Operator

Установка Jaeger Operator через веб-консоль

Вы можете установить Jaeger Operator из раздела **Marketplace → OperatorHub** в Alauda Container Platform, где перечислены доступные операторы.

Шаги

- В режиме **Administrator** веб-консоли выберите **кластер**, в котором хотите развернуть Jaeger Operator, затем перейдите в **Marketplace → OperatorHub**.
- Используйте поле поиска, чтобы найти `Alauda build of Jaeger` в каталоге. Нажмите на заголовок **Alauda build of Jaeger**.
- Ознакомьтесь с вводной информацией об операторе на странице **Alauda build of Jaeger**. Нажмите **Install**.
- На странице **Install**:
  - Выберите **Manual** для **Upgrade Strategy**. При стратегии одобрения `Manual` OLM создаст запросы на обновление. Как администратор кластера, вы должны вручную одобрять запросы OLM для обновления оператора до новой версии.
  - Выберите канал **stable (Default)**.
  - Выберите **Recommended** для **Installation Location**. Установите оператор в рекомендуемое пространство имен `jaeger-operator`, чтобы оператор мог мониторить и быть доступным во всех пространствах имен кластера.

- Нажмите **Install**.
- Убедитесь, что статус отображается как **Succeeded**, чтобы подтвердить корректную установку Jaeger Operator.
- Проверьте, что все компоненты Jaeger Operator успешно установлены. Войдите в кластер через терминал и выполните команду:

```
kubectl -n jaeger-operator get csv
```

Пример вывода

```
NAME                    DISPLAY          VERSION   REPLACES   PHASE
jaeger-operator.vx.x.0  Jaeger Operator  x.x.0                Succeed
ed
```

Если поле `PHASE` показывает `Succeeded`, значит оператор и его компоненты установлены успешно.

Развертывание экземпляра Jaeger

Экземпляр Jaeger и связанные с ним ресурсы можно установить с помощью скрипта `install-jaeger.sh`, который принимает три параметра:

- `--es-url`: URL доступа к Elasticsearch.
- `--es-user-base64`: имя пользователя `Basic Auth` для Elasticsearch, закодированное в base64.
- `--es-pass-base64`: пароль `Basic Auth` для Elasticsearch, закодированный в base64.

Скопируйте скрипт установки из **DETAILS**, войдите в кластер, где хотите установить, сохраните его как `install-jaeger.sh` и выполните после предоставления прав на выполнение:

▶ **DETAILS**

Пример запуска скрипта:

```
./install-jaeger.sh --es-url='https://xxx' --es-user-base64='xxx' --es-pa
ss-base64='xxx'
```

Пример вывода скрипта:

```
CLUSTER_NAME: <cluster>
ES_URL: https://xxx
ES_USER_BASE64: xxx
ES_PASS_BASE64: xxx
TARGET_NAMESPACE: cpaas-system
JAEGER_INSTANCE_NAME: jaeger-prod
JAEGER_BASEPATH_SUFFIX: /acp/jaeger
JAEGER_ES_INDEX_PREFIX: acp-tracing-<cluster>
PLATFORM_URL: https://xxx
configmap/jaeger-prod-oauth2-proxy created
secret/jaeger-prod-oauth2-proxy created
secret/jaeger-prod-es-basic-auth created
serviceaccount/jaeger-prod-sa created
role.rbac.authorization.k8s.io/jaeger-prod-role created
rolebinding.rbac.authorization.k8s.io/jaeger-prod-rb created
jaeger.jaegertracing.io/jaeger-prod created
podmonitor.monitoring.coreos.com/jaeger-prod-monitor created
ingress.networking.k8s.io/jaeger-prod-query created
Jaeger UI access address: <platform-url>/clusters/<cluster>/acp/jaeger
Jaeger installation completed
```

Установка OpenTelemetry Operator

Установка OpenTelemetry Operator через веб-консоль

Вы можете установить OpenTelemetry Operator из раздела **Marketplace → OperatorHub** в Alauda Container Platform, где перечислены доступные операторы.

Шаги

- В режиме **Administrator** веб-консоли выберите **кластер**, в котором хотите развернуть OpenTelemetry Operator, затем перейдите в **Marketplace** → **OperatorHub**.
- Используйте поле поиска, чтобы найти `Alauda build of OpenTelemetry` в каталоге. Нажмите на заголовок **Alauda build of OpenTelemetry**.
- Ознакомьтесь с вводной информацией об операторе на странице **Alauda build of OpenTelemetry**. Нажмите **Install**.
- На странице **Install**:
  - Выберите **Manual** для **Upgrade Strategy**. При стратегии одобрения `Manual` OLM создаст запросы на обновление. Как администратор кластера, вы должны вручную одобрять запросы OLM для обновления оператора до новой версии.
  - Выберите канал **alpha (Default)**.
  - Выберите **Recommended** для **Installation Location**. Установите оператор в рекомендуемое пространство имен `opentelemetry-operator`, чтобы оператор мог мониторить и быть доступным во всех пространствах имен кластера.
- Нажмите **Install**.
- Убедитесь, что статус отображается как **Succeeded**, чтобы подтвердить корректную установку OpenTelemetry Operator.
- Проверьте, что все компоненты OpenTelemetry Operator успешно установлены. Войдите в кластер через терминал и выполните команду:

```
kubectl -n opentelemetry-operator get csv
```

Пример вывода

```
NAME                                DISPLAY                    VERSION   REPL
ACES   PHASE
openTelemetry-operator.vx.x.0   OpenTelemetry Operator   x.x.0
Succeeded
```

Если поле `PHASE` показывает `Succeeded`, значит оператор и его компоненты установлены успешно.

Развертывание экземпляров OpenTelemetry

Экземпляры OpenTelemetry и связанные с ними ресурсы можно установить с помощью скрипта `install-otel.sh`.

Скопируйте скрипт установки из **DETAILS**, войдите в кластер, где хотите установить, сохраните его как `install-otel.sh` и выполните после предоставления прав на выполнение:

▶ **DETAILS**

Пример запуска скрипта:

```
./install-otel.sh
```

Пример вывода скрипта:

```
CLUSTER_NAME: cluster-xxx
serviceaccount/otel-collector created
clusterrolebinding.rbac.authorization.k8s.io/otel-collector:cpaas-system:
cluster-admin created
opentelemetrycollector.opentelemetry.io/otel created
instrumentation.opentelemetry.io/acp-common-java created
servicemonitor.monitoring.coreos.com/otel-collector-monitoring created
servicemonitor.monitoring.coreos.com/otel-collector created
OpenTelemetry installation completed
```

Включение переключателя функций

Система трассировки находится на этапе **Alpha** и требует ручного включения переключателя функции `acp-tracing-ui` в разделе **Feature Switch**.

Затем перейдите в представление **Container Platform** и откройте **Observability** → **Tracing**, чтобы просмотреть функцию трассировки.

Удаление Tracing

Удаление экземпляра OpenTelemetry

Войдите в установленный кластер и выполните следующие команды для удаления экземпляра OpenTelemetry и связанных ресурсов.

```
kubectl -n cpaas-system delete servicemonitor otel-collector-monitoring
kubectl -n cpaas-system delete servicemonitor otel-collector
kubectl -n cpaas-system delete instrumentation acp-common-java
kubectl -n cpaas-system delete opentelemetrycollector otel
kubectl delete clusterrolebinding otel-collector:cpaas-system:cluster-adm
in
kubectl -n cpaas-system delete serviceaccount otel-collector
```

Удаление OpenTelemetry Operator

Вы можете удалить OpenTelemetry Operator через режим **Administrator** в веб-консоли.

Шаги

- В разделе **Marketplace** → **OperatorHub** используйте поле поиска, чтобы найти `Alauda build of OpenTelemetry`.
- Нажмите на заголовок **Alauda build of OpenTelemetry** для перехода к деталям.
- На странице деталей нажмите кнопку **Uninstall** в правом верхнем углу.
- В окне **Uninstall "opentelemetry-operator"?** нажмите **Uninstall**.

Удаление экземпляра Jaeger

Войдите в установленный кластер и выполните следующие команды для удаления экземпляра Jaeger и связанных ресурсов.

```
kubectl -n cpaas-system delete ingress jaeger-prod-query
kubectl -n cpaas-system delete podmonitor jaeger-prod-monitor
kubectl -n cpaas-system delete jaeger jaeger-prod
kubectl -n cpaas-system delete rolebinding jaeger-prod-rb
kubectl -n cpaas-system delete role jaeger-prod-role
kubectl -n cpaas-system delete serviceaccount jaeger-prod-sa
kubectl -n cpaas-system delete secret jaeger-prod-oauth2-proxy
kubectl -n cpaas-system delete secret jaeger-prod-es-basic-auth
kubectl -n cpaas-system delete configmap jaeger-prod-oauth2-proxy
```

Удаление Jaeger Operator

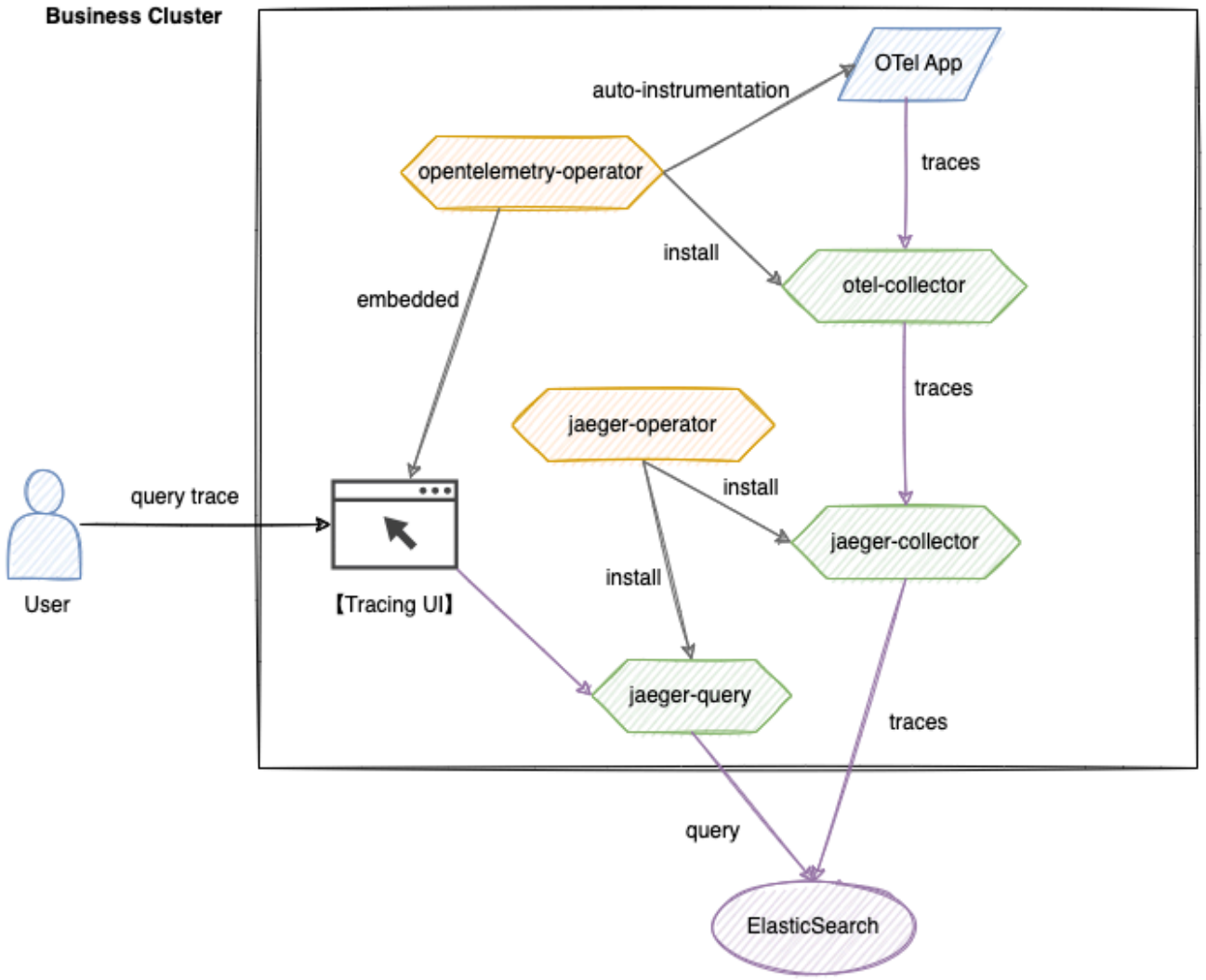
Вы можете удалить Jaeger Operator через режим **Administrator** в веб-консоли.

Шаги

- В разделе **Marketplace** → **OperatorHub** используйте поле поиска, чтобы найти `Alauda build of Jaeger`.
- Нажмите на заголовок **Alauda build of Jaeger** для перехода к деталям.
- На странице деталей нажмите кнопку **Uninstall** в правом верхнем углу.
- В окне **Uninstall "jaeger-operator"?** нажмите **Uninstall**.

Архитектура

Данная архитектура построена на технологическом стеке OpenTelemetry и Jaeger, обеспечивая полный жизненный цикл управления распределённым трассированием. Система состоит из пяти основных модулей: сбор данных, передача, хранение, запросы и визуализация.



Содержание

Основные компоненты

Поток данных

Основные компоненты

1. **Система OpenTelemetry**

   - **opentelemetry-operator**
     Оператор на уровне кластера, отвечающий за развертывание и управление компонентом otel-collector, обеспечивающий возможность автоматического внедрения OTel.
   - **otel-collector**
     Принимает данные трассировки от приложений, фильтрует и группирует их, затем пересылает в jaeger-collector.
   - **Tracing UI**
     Самостоятельно разработанный интерфейс визуализации, интегрированный с API jaeger-query, поддерживающий многомерные условия запросов.

2. **Система Jaeger**

   - **jaeger-operator**
     Разворачивает и управляет компонентами jaeger-collector и jaeger-query.
   - **jaeger-collector**
     Принимает данные трассировки, пересланные и обработанные otel-collector, выполняет преобразование формата и записывает их в Elasticsearch.
   - **jaeger-query**
     Предоставляет API для запросов трассировки, поддерживая многокритериальный поиск, включая TraceID и метки.

3. **Слой хранения**

   - **Elasticsearch**
     Распределённый движок хранения, поддерживающий эффективную запись и извлечение огромного объёма данных Span.

Поток данных

- **Процесс записи**

  `Application -> otel-collector -> jaeger-collector -> Elasticsearch`

  Приложение генерирует данные Span через SDK или автоматическое внедрение, которые стандартизируются otel-collector и затем сохраняются в Elasticsearch через jaeger-collector.

- **Процесс запроса**

  `User -> Tracing UI -> jaeger-query -> Elasticsearch`

  Пользователь отправляет условия запроса через UI, jaeger-query извлекает данные из Elasticsearch; UI визуализирует результаты на основе возвращённых данных.

Основные понятия

Содержание

- Telemetry
- OpenTelemetry
- Span
- Trace
- Instrumentation
- OpenTelemetry Collector
- Jaeger

Telemetry

Телеметрия — это данные, генерируемые системами и их поведением, включая трассы, метрики и логи.

OpenTelemetry

OpenTelemetry — это фреймворк и набор инструментов для наблюдаемости, предназначенный для создания и управления телеметрическими данными, такими как трассы, метрики и логи. Важно, что OpenTelemetry не зависит от поставщика, то есть может работать с различными системами наблюдаемости, включая open-source инструменты, такие как Jaeger и Prometheus, а также коммерческие продукты.

Span

Span — это базовый строительный блок распределённого трассирования, представляющий конкретную операцию или единицу работы. Каждый span фиксирует определённые действия в рамках запроса, помогая понять детали того, что происходило во время выполнения операции.

Span содержит имя, временные данные, структурированные сообщения логов и другие метаданные (атрибуты), которые вместе дают полную картину операции.

Trace

Trace фиксирует путь запроса (от приложения или конечного пользователя) по много-сервисной архитектуре (например, микросервисы и serverless-приложения).

Trace состоит из одного или нескольких span-ов. Первый span называется корневым span-ом и представляет весь жизненный цикл запроса от начала до конца. Дочерние span-ы под корневым span-ом предоставляют более детальную контекстную информацию о процессе запроса (или различных шагах, составляющих запрос).

Без трасс было бы довольно сложно определить коренную причину проблем с производительностью в распределённых системах. Трассы упрощают отладку и понимание распределённых систем, разбивая поток запросов через них.

Instrumentation

Для обеспечения наблюдаемости система должна пройти процесс **инструментирования**: компоненты кода системы должны генерировать трассы↗, метрики↗ и логи↗.

С помощью OpenTelemetry вы можете инструментировать ваш код двумя основными способами:

1. Решения на основе кода↗: используя официальные API и SDK для большинства языков↗

2. Решения с нулевым инструментированием ↗

Решения на основе кода обеспечивают более глубокое понимание и более богатые телеметрические данные изнутри вашего приложения. Вы можете генерировать телеметрию в вашем приложении с помощью OpenTelemetry API, что является важным дополнением к данным, создаваемым решениями с нулевым инструментированием.

Решения с нулевым инструментированием отлично подходят для быстрого старта или когда вы не можете изменить приложение, из которого нужно получить телеметрию. Они могут предоставлять богатые телеметрические данные через используемые библиотеки или среду выполнения. Другими словами, они передают информацию о событиях, происходящих на границах (Edges) приложения.

Эти два подхода могут использоваться одновременно.

OpenTelemetry Collector

OpenTelemetry Collector — это агент, не зависящий от поставщика, который может принимать, обрабатывать и экспортировать телеметрические данные. Он поддерживает приём данных в различных форматах (таких как OTLP, Jaeger, Prometheus и многие коммерческие/проприетарные инструменты) и отправку этих данных в один или несколько бекендов. Также поддерживается обработка и фильтрация телеметрии перед экспортом.

Для дополнительной информации смотрите Collector ↗.

Jaeger

Jaeger — это open-source **система распределённого трассирования**. Она предназначена для мониторинга и диагностики сложных распределённых систем на основе микросервисной архитектуры, помогая разработчикам визуализировать трассы запросов, анализировать узкие места производительности и устранять аномалии. Jaeger совместим со стандартом **OpenTracing** (теперь частью OpenTelemetry), поддерживает множество языков программирования и хранилищ данных, и является ключевым инструментом наблюдаемости в облачно-нативной среде.

Руководства

Query Tracing

- Feature Overview
- Main Features
- Feature Advantages
- Tracing Query
- Query Result Analysis

Query Trace Logs

- Обзор функции
- Основные возможности
- Предварительные требования
- Операции с запросами логов

Query Tracing

Содержание

- Feature Overview
- Main Features
- Feature Advantages
- Tracing Query
  - Step 1: Combine Query Conditions
  - Step 2: Execute Query
- Query Result Analysis
  - Span List
  - Time-Series Waterfall Chart
  - Span Details

Feature Overview

Функция распределённого трассирования запросов предоставляет возможности полного трассирования цепочки вызовов в архитектуре микросервисов за счёт сбора метаданных межсервисных вызовов, помогая пользователям быстро локализовать проблемы в межсервисных вызовах. Эта функция решает следующие задачи:

- **Трассировка цепочки запроса**: Восстановление полного пути запроса в сложных распределённых системах.

- **Анализ узких мест производительности**: Выявление аномальных узлов вызова по времени выполнения в цепочке.
- **Определение корня ошибки**: Быстрая локализация точки возникновения проблемы с помощью маркировки ошибок.

Применимые сценарии включают:

- Быстрое обнаружение аномальных сервисов при устранении неисправностей в продуктивной среде.
- Выявление участков с высокой задержкой вызовов при оптимизации производительности.
- Проверка взаимосвязей межсервисных вызовов после выпуска новой версии.

Ключевые ценности:

- Повышение наблюдаемости распределённых систем.
- Сокращение среднего времени восстановления (MTTR).
- Оптимизация производительности межсервисных вызовов.

Main Features

- Многомерный поиск: Поддержка 6 комбинаций условий запроса, таких как TraceID, имя сервиса, метки и др.
- Визуальный анализ: Интуитивное отображение иерархии вызовов и распределения времени с помощью временных водопадных диаграмм.
- Точная локализация: Поддержка фильтрации по ошибочным Span и вторичных поисков с метками.

Feature Advantages

- **Быстрая идентификация проблем**: Сужение области поиска с помощью многомерных условий ускоряет локализацию проблем.

- **Визуальное представление**: Использование временных водопадных диаграмм для наглядного отображения связей вызовов снижает сложность и повышает эффективность анализа ошибок.
- **Гибкость и разнообразие**: Поддержка как простых, так и сложных комбинаций запросов, адаптированных под различные сценарии эксплуатации и разработки.

Tracing Query

1 Step 1: Combine Query Conditions

Совет: Условия запроса можно комбинировать. Вы можете уточнить запрос, добавляя несколько условий.

Query Condition	Description
TraceID	Уникальный идентификатор полной цепочки, по которому можно выполнить поиск конкретного трассирования.
Service	Сервис, инициирующий или принимающий запрос вызова (необходимо выбрать или ввести).
Label	Можно фильтровать результаты запроса, вводя метки (Tag), поддерживаются метки из деталей Span.
Span Duration Greater Than	Spans с длительностью больше или равной *введённому значению* (мс).
Only Search Error Spans	Ошибочные Spans — это Spans, у которых значение тега **error** равно `true`.
Span Type	**Root Span**: Поиск корневых Span, инициированных указанным **сервисом**. Этот режим используется, когда указанный сервис является инициатором всего запроса. **Service Entry Span**: Поиск первого Span, созданного при вызове указанного **сервиса** в роли сервера.

Query Condition	Description
Maximum Query Count	Максимальное количество Span, которые можно запросить, по умолчанию 200. **Совет**: Для производительности платформа отображает максимум 1000 Span за один раз. Если количество Span, соответствующих условиям, превышает **максимальное количество запросов**, можно уточнить условия или сузить временной диапазон для поэтапного поиска.

2 Step 2: Execute Query

- После выбора условий и ввода значений нажмите кнопку **Add to Query Conditions**, текущие условия отобразятся в области результатов **Query Conditions**, что запустит запрос.
- Также можно развернуть **Common Query Conditions** для быстрого добавления недавно использованных условий поиска.

Query Result Analysis

После ввода условий и выполнения поиска на странице появится область с результатами запроса.

Span List

Слева в области результатов отображается список Span, соответствующих условиям, с базовой информацией: имя сервиса, вызываемый интерфейс или метод обработки запроса, длительность и время начала.

Time-Series Waterfall Chart

Временная водопадная диаграмма справа на странице результатов наглядно показывает связи вызовов между Span в одном трассировании. Основные особенности использования временных водопадных диаграмм в анализе трассировок:

1. Раскрытие сверху вниз: В диаграмме вызовы (Span) обычно раскрываются вниз от верхней части, каждая горизонтальная полоса представляет вызов сервиса или процесс. Положение отражает логический порядок вызовов.
2. Выравнивание по временной оси: Горизонтальная ось — время. Длина полосы показывает длительность вызова, что позволяет интуитивно сравнивать временные отношения между вызовами.
3. Отступы: Отступы показывают иерархию вызовов, чем глубже отступ, тем глубже уровень вызова в цепочке.
4. Интерактивность и подробности: Клик по полосам диаграммы отображает более детальную информацию о вызове.

Span Details

Клик по строке Span в диаграмме раскрывает подробную информацию о Span, включая:

- Service: сервис, к которому относится Span.
- Span Duration (ms): длительность Span.
- URL: URL, к которому обращается сервис, соответствует **http.url** в тегах Span.
- Tag: метки Span в виде пар ключ-значение, которые можно использовать для расширенного поиска по меткам. Кнопка рядом с меткой позволяет добавить текущее условие метки в запрос для более точного поиска.
- JSON: исходная JSON-структура Span для детального изучения внутренней информации.

Query Trace Logs

Содержание

- Обзор функции
- Основные возможности
- Предварительные требования
- Операции с запросами логов
  - Доступ к логам трейсинга
  - Фильтрация логов
    - По имени Pod
    - По временному диапазону
    - По условиям запроса
    - Содержит Trace ID
  - Расширенные операции
    - Экспорт логов
    - Настройка отображаемых полей
    - Просмотр контекста логов

Обзор функции

Trace Logs позволяют пользователям выполнять запросы и анализировать логи, связанные с конкретным распределённым трейсом, используя его уникальный TraceID.

Эта функция помогает разработчикам и операторам быстро находить проблемы, понимать потоки запросов и связывать бизнес-логи с контекстами трейсинга.

Основные преимущества:

- **Анализ первопричин**: выявление ошибок и проблем с задержками в микросервисах распределённых систем.
- **Корреляция контекста**: связывание бизнес-логов с трассировочными спанами для сквозной видимости.
- **Эффективная фильтрация**: фильтрация логов по Pod или TraceID для фокусировки на релевантных данных.

Применимые сценарии:

- Отладка сбоев транзакций между сервисами.
- Анализ узких мест производительности в сложных рабочих процессах.
- Аудит потоков обработки запросов для соответствия требованиям.

Основные возможности

- **Запрос по TraceID**: получение всех логов, связанных с конкретным трейсом по его TraceID.
- **Фильтрация по Pod**: просмотр логов из конкретных Pod, участвующих в трейсинге.
- **Экспорт логов**: экспорт отфильтрованных логов в настраиваемых форматах.
- **Просмотр контекста логов**: изучение записей логов до и после целевой записи для более глубокого анализа.

Предварительные требования

TIP

Перед выполнением запросов логов по TraceID необходимо инструментировать ваши сервисы для включения TraceID в бизнес-логи. Следуйте руководству Business Log Correlation with TraceID Guide для настройки.

Поведение по умолчанию:

- Отображает логи за весь период трейсинга.
- Для трейсинга короче 1 минуты выполняет запрос логов в течение 1 минуты после времени начала трейсинга.

Операции с запросами логов

1 Доступ к логам трейсинга

1. После выполнения запроса трейсинга кликните по конкретному трейсу для просмотра деталей.
2. Перейдите на вкладку **View Log** в панели визуализации трейсинга.

2 Фильтрация логов

По имени Pod

В селекторе **Pod Name** выберите целевой Pod из списка участвующих сервисов.

По временному диапазону

В селекторе **Time Range** выберите нужный временной интервал.

По условиям запроса

Введите ключевые слова в текстовое поле **Query Conditions** для фильтрации логов по конкретному содержимому.

Содержит Trace ID

Отметьте чекбокс **Contain Trace ID**.

3

Расширенные операции

Экспорт логов

1. Нажмите **Export**.
2. Выберите поля для включения с помощью чекбоксов столбцов.
3. Выберите формат экспорта (JSON/CSV).

Настройка отображаемых полей

Нажмите **Set**. Переключайте видимость полей логов в панели отображения.

Просмотр контекста логов

1. Нажмите **Insight** рядом с любой записью лога.
2. Изучите 5 предыдущих и 5 последующих логов вокруг целевой временной метки.
3. Прокручивайте вверх/вниз мышью для загрузки дополнительных логов.

Как сделать

Беспрепятственная интеграци

- Обзор возможностей
- Сценарии использования
- Предварительные требования
- Шаги для настройки
- Результаты работы

Бизнес-логи, связанные с TraceID

- Предпосылки
- Добавление TraceID в логи Java-приложения
- Добавление TraceID в логи Python-приложения
- Метод проверки

Беспрепятственная интеграция трассировки в Java-приложениях

INFO

Автоматически внедряемый OpenTelemetry Java Agent ↗ поддерживает версии `Java 8+`.

Содержание

- Обзор возможностей
- Сценарии использования
- Предварительные требования
- Шаги для настройки
- Результаты работы

Обзор возможностей

Трассировка — это ключевая функция наблюдаемости в распределённых системах, которая позволяет полностью записывать пути вызовов и данные о производительности запросов внутри системы. В этой статье описывается, как добиться беспрепятственной интеграции трассировки в Java-приложениях с помощью автоматического внедрения OpenTelemetry Java Agent.

Сценарии использования

Java-приложения могут быть интегрированы для следующих задач:

- Быстрое добавление возможностей трассировки в Java-приложения
- Избежание изменений исходного кода приложения
- Развёртывание сервисов с помощью Kubernetes
- Визуализация взаимосвязей вызовов между сервисами и анализ узких мест производительности

Предварительные требования

Перед использованием данной функции убедитесь, что:

- Целевой сервис развернут на платформе Alauda Container Platform
- Сервис использует JDK версии Java 8 или выше
- У вас есть права на редактирование Deployment в целевом namespace
- На платформе выполнено развёртывание трассировки

Шаги для настройки

Для Java-приложения, которое необходимо интегрировать с трассировкой Alauda Container Platform, требуется выполнить следующие настройки:

- Настроить аннотации автоматического внедрения для Java Deployment.
- Установить переменную окружения `SERVICE_NAME`.
- Установить переменную окружения `SERVICE_NAMESPACE`.

Пример адаптации Deployment:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-java-deploy
spec:
  template:
    metadata:
      annotations:
        instrumentation.opentelemetry.io/inject-java: cpaas-system/acp-co
mmon-java (1)
      labels:
        app.kubernetes.io/name: my-java-app
    spec:
      containers:
      - env:
        - name: SERVICE_NAME (2)
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.labels['app.kubernetes.io/name']
        - name: SERVICE_NAMESPACE (3)
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.namespace
```

(1) Выберите инструментальную настройку `cpaas-system/acp-common-java` для внедрения Java Agent.

(2) Настройте переменную окружения `SERVICE_NAME`, которая может быть связана через метки или иметь фиксированное значение.

(3) Настройте переменную окружения `SERVICE_NAMESPACE`, значение которой берётся из `metadata.namespace`.

Результаты работы

После адаптации Java-приложения:

- Если в новом поде Java-приложения присутствует init-контейнер `opentelemetry-auto-instrumentation-java`, это означает успешное внедрение.
- Отправьте тестовые запросы в Java-приложение.
- В представлении **Container Platform** выберите **проект**, **кластер** и **namespace**, где находится Java-приложение.
- Перейдите на страницу **Observability** -> **Tracing**, чтобы просмотреть данные трассировки и диаграмму временной последовательности Java-приложения.

TIP

В этой статье разработчикам будет показано, как интегрировать методы **получения TraceID** и **добавления TraceID в логи приложения** в код приложения, что подходит для backend-разработчиков с некоторым опытом разработки.

Бизнес-логи, связанные с TraceID

Содержание

Предпосылки

Добавление TraceID в логи Java-приложения

Добавление TraceID в логи Python-приложения

Метод проверки

Предпосылки

- Для корректного связывания нескольких автоматически отправляемых спанов (различных модулей/узлов/сервисов, вызываемых в рамках одного запроса) в один трейс, в HTTP-заголовках запроса сервиса передаются TraceID и другая информация, используемая для связывания трейса.
- Трейс представляет собой процесс вызова одного запроса, а TraceID — уникальный идентификатор, определяющий этот запрос. Наличие TraceID в логах позволяет связать трассировку с логами приложения.

Исходя из вышеизложенного, в этой статье будет объяснено, как получить TraceID из HTTP-заголовков запроса и добавить его в логи приложения, что позволит точно выполнять поиск логов на платформе по TraceID.

Добавление TraceID в логи Java-приложения

TIP

- Приведённые ниже примеры основаны на фреймворке **Spring Boot** и используют **Log4j** и **Logback** для иллюстрации.
- Ваше приложение должно соответствовать следующим требованиям:
  - Тип и версия библиотеки логирования должны соответствовать следующим требованиям:

Logging Library	Требование по версии
Log4j 1	1.2+
Log4j 2	2.7+
Logback	1.0+

  - В приложение внедрён Java Agent.

Способ 1: Настройка `logging.pattern.level`

Измените параметр `logging.pattern.level` в конфигурации приложения следующим образом:

```
logging.pattern.level = trace_id=%mdc{trace_id}
```

Способ 2: Настройка `CONSOLE_LOG_PATTERN`

1. Измените конфигурационный файл logback следующим образом.

> **TIP**
>
> Здесь приведён пример вывода в консоль, где `%X{trace_id}` указывает на значение ключа `trace_id`, полученное из MDC.

```
<property name="CONSOLE_LOG_PATTERN"
    value="${CONSOLE_LOG_PATTERN:-%clr(%d{yyyy-MM-dd HH:mm:ss.SSS}){fai
nt} [trace_id=%X{trace_id}] %clr(${LOG_LEVEL_PATTERN:-%5p}) %clr(${PID:
- }){magenta} %clr(---){faint} %clr([%15.15t]){faint} %clr(%-40.40logge
r{39}){cyan} %clr(:){faint} %m%n${LOG_EXCEPTION_CONVERSION_WORD:-%wE
x}}"/>
```

2. В классе, где необходимо выводить логи, добавьте аннотацию `@Slf4j` и используйте объект log для вывода логов, как показано ниже:

```
@RestController
@Slf4j
public class ProviderController {

    @GetMapping("/hello")
    public String hello(HttpServletRequest request) {
        log.info("request /hello");
        return "hello world";
    }
}
```

Добавление TraceID в логи Python-приложения

1. В коде приложения добавьте следующий код для получения TraceID из заголовков запроса. Пример кода приведён ниже и может быть адаптирован по необходимости:

> **TIP**
>
> Функция **getForwardHeaders** извлекает информацию о трассировке из заголовков запроса, где значение `x-b3-traceid` является TraceID.

```
    def getForwardHeaders(request):
        headers = {}
        incoming_headers = [
            'x-request-id',  # Все приложения должны передавать x-reques
t-id для access-логов и согласованных решений по трассировке/выборке ло
гов
            'x-b3-traceid',  # Заголовок B3 trace, совместимый с Zipkin,
OpenCensusAgent и Stackdriver
            'x-b3-spanid',
            'x-b3-parentspanid',
            'x-b3-sampled',
            'x-b3-flags',
        ]
        for ihdr in incoming_headers:
            val = request.headers.get(ihdr)
            if val is not None:
                headers[ihdr] = val

        return headers
```

2. В коде приложения добавьте следующий код для включения полученного TraceID в логи. Пример кода приведён ниже и может быть адаптирован по необходимости:

```
headers = getForwardHeaders(request)
tracing_section = ' [%(x-b3-traceid)s,%(x-b3-spanid)s] ' % headers
logging.info(tracing_section + "Oops, unexpected error happens.")
```

Метод проверки

1. Нажмите на **Tracing** в левой навигационной панели.
2. В критериях запроса выберите TraceID, введите TraceID для поиска и нажмите **Add to query**.
3. В отображённых ниже данных трейса нажмите **View Log** рядом с TraceID.
4. На странице **Log Query** отметьте **Contain Trace ID**; система отобразит только лог-данные, содержащие TraceID.

Устранение неполадок

Невозможно выполнить запр

Описание проблемы

Анализ первопричин

Решение для первопричины 1

Решение для первопричины 2

Неполные данные трассировки

Описание проблемы

Анализ причин

Решение для причины 1

Решение для причины 2

Невозможно выполнить запрос необходимого трассирования

Содержание

- Описание проблемы
- Анализ первопричин
  1. Слишком низкий уровень выборки трассирования
  2. Ограничения Elasticsearch по времени обновления
- Решение для первопричины 1
- Решение для первопричины 2

Описание проблемы

При выполнении запроса трассирования в сервисной сетке могут возникать ситуации, когда целевое трассирование не удаётся получить.

Анализ первопричин

1. Слишком низкий уровень выборки трассирования

Если параметр уровня выборки для трассирования установлен слишком низко, система будет собирать данные трассирования пропорционально. В периоды недостаточного

объёма запросов или в часы низкой нагрузки это может привести к тому, что выборочные данные окажутся ниже порога видимости.

2. Ограничения Elasticsearch по времени обновления

По умолчанию для индекса Elasticsearch установлена конфигурация `"refresh_interval": "10s"`, что приводит к задержке в 10 секунд перед обновлением данных из буфера памяти в состояние, доступное для поиска. При запросе недавно сгенерированного трассирования результаты могут отсутствовать, так как данные ещё не были сохранены.

Эта конфигурация индекса эффективно снижает нагрузку на слияние данных в Elasticsearch, улучшая скорость индексирования и скорость первого запроса, но в то же время снижает степень актуальности данных в реальном времени.

Решение для первопричины 1

- Соответственно увеличить уровень выборки в зависимости от требований.
- Использовать более продвинутые методы выборки, например, tail sampling.

Решение для первопричины 2

Настроить интервал обновления через параметр запуска `--es.asm.index-refresh-interval` компонента `jaeger-collector`, значение по умолчанию — `10s`.

Если значение этого параметра установить в `"null"`, конфигурация `refresh_interval` для индекса не будет применяться.

Примечание: Установка значения `"null"` повлияет на производительность и скорость запросов Elasticsearch.

Неполные данные трассировки

Содержание

Описание проблемы
Анализ причин
1. Задержка сохранения данных
2. Ограничение временного диапазона
Решение для причины 1
Решение для причины 2

Описание проблемы

Результаты запросов трассировки демонстрируют следующие проблемы с неполными данными:

- В недавних запросах (за последние 30 минут) отсутствуют некоторые спаны.
- Трассировки старше 1 часа испытывают разрывы соединения.

Анализ причин

1. Задержка сохранения данных

Процесс записи в Elasticsearch требует последовательного выполнения шагов: буфер памяти → translog → сегментные файлы, что может приводить к задержкам видимости недавно записанных данных.

2. Ограничение временного диапазона

По умолчанию, когда `jaeger-query` запрашивает спаны, соответствующие трассировке, временной диапазон расширяется на один час до и после времени начала спана.

Например, если спан начинается в `08:12:30` и заканчивается в `08:12:32`, то временной диапазон для запроса этой трассировки будет с `07:12:30` по `09:12:32`.

Таким образом, если длительность трассировки превышает 1 час, запрос по этому спану может не вернуть полную трассировку.

Решение для причины 1

Подождите немного и обновите страницу, чтобы повторить запрос.

Решение для причины 2

Если длительность трассировки в вашей среде большая, вы можете настроить временной диапазон запроса для одной трассировки с помощью параметра запуска `--es.asm.span-trace-query-time-adjustment-hours` в `jaeger-query`.

Значение этого параметра по умолчанию — `1` час, и вы можете увеличить его при необходимости.

Логи

О сервисе логирования

О сервисе логирования

Модуль Logging является ключевым компонентом набора средств наблюдаемости платформы ACP, предоставляющим комплексные возможности управления логами для эффективной и надежной обработки журналов.

Этот модуль обеспечивает четыре основные функции логирования:

- **Сбор логов** для автоматизированного сбора журналов из приложений, контейнеров и компонентов инфраструктуры
- **Хранение логов** для масштабируемого и долговременного сохранения с использованием ElasticSearch и ClickHouse
- **Запрос логов** для быстрого и гибкого поиска по большим объемам данных журналов

Интегрируя эти возможности с мощными open-source компонентами, такими как Filebeat, ElasticSearch и ClickHouse, модуль позволяет организациям эффективно обрабатывать огромные объемы логов, ускорять устранение неполадок, обеспечивать соответствие требованиям и получать ценные операционные данные в режиме реального времени.

> **Note**
>
> Поскольку выпуски Logging Service осуществляются в ином режиме, чем у Alauda Container Platform, документация Logging Service теперь доступна в виде отдельного набора по адресу Logging Service↗。

События

Введение

Ограничения по использованию

Events

Operation Procedures

Event Overview

Введение

Платформа интегрируется с событиями Kubernetes, регистрируя значимые изменения статуса и различные изменения операционного состояния ресурсов Kubernetes. Она также предоставляет возможности для хранения, запроса и визуализации данных. При возникновении аномалий с ресурсами, такими как кластеры, узлы или Pods, пользователи могут анализировать события для определения конкретных причин.

На основе выявленных корневых причин из событий пользователи могут создавать политики оповещений для рабочих нагрузок. Когда количество критических событий достигает порога оповещения, оповещения могут автоматически запускаться для уведомления соответствующего персонала с целью своевременного вмешательства, что снижает операционные риски на платформе.

Содержание

Ограничения по использованию

Ограничения по использованию

Эта функция зависит от сервиса логирования. Пожалуйста, убедитесь, что в платформе предварительно установлены плагины ACP Log Essentails, ACP Log Collector и ACP Log Storage.

Note

Поскольку выпуски Logging Service осуществляются в ином режиме, чем у Alauda Container Platform, документация Logging Service теперь доступна в виде отдельного набора по адресу Logging Service↗。

Events

Содержание

Operation Procedures

Event Overview

Operation Procedures

1. Нажмите **Operations Center** > **Events** в левой навигационной панели.
   Совет: Переключайте кластер для просмотра событий с помощью выпадающего списка в верхней навигационной панели.

Event Overview

Страница событий по умолчанию отображает обзор значимых событий, произошедших за последние 30 минут (вы можете выбрать или настроить временной диапазон), а также записи событий ресурсов.

- **Significant Event Overview**: Эта карточка показывает причины значимых событий и количество ресурсов, у которых произошли такие события в выбранном временном диапазоне.
  - **Примечание**: Если один и тот же ресурс испытывал данный тип события несколько раз, количество ресурсов не суммируется.

  - Например: если количество ресурсов для событий перезапуска узла равно 20, это означает, что в выбранном временном диапазоне 20 ресурсов столкнулись с такими событиями, и один и тот же ресурс мог испытать это несколько раз.

- **Resource Event Records**: Ниже области обзора значимых событий отображаются все записи событий, соответствующие условиям запроса в выбранном временном диапазоне. Вы можете отфильтровать соответствующие типы событий, нажав на карточку значимого события, либо развернуть вид и ввести условия запроса для поиска. Условия запроса следующие:

  - **Resource Type**: Тип Kubernetes-ресурса, у которого произошло событие, например Pod.
  - **Namespace**: Пространство имён Kubernetes-ресурса, в котором произошло событие.
  - **Event Reason**: Причина возникновения события.
  - **Event Level**: Значимость события, например Warning.
  - **Resource Name**: Имя Kubernetes-ресурса, у которого произошло событие. Можно выбрать или ввести несколько имён.

TIP

- Нажмите на иконку просмотра рядом с именем ресурса в записи события, чтобы увидеть подробную информацию о событии в всплывающем диалоговом окне **Event Details**.
- Цвет иконки слева от причины события указывает уровень события. Зелёная иконка означает, что уровень этого события — `Normal`, и его можно игнорировать; оранжевая иконка означает, что уровень события — `Warning`, что указывает на аномалию с ресурсом, и за этим событием следует следить, чтобы предотвратить инциденты.

Инспекция

Введение

Введение

Ограничения использования

Архитектура

Архитектура

Inspection

Component Health Status

Руководства

Inspection

Execute Inspection

Inspection Configuration

Inspection Report Explanation

Статус здоровья компонентов

Процедуры работы

Введение

Модуль Inspection является ключевым компонентом набора средств наблюдаемости платформы ACP, обеспечивающим автоматизированные возможности инспекции и оценки для комплексного мониторинга ресурсов и управления рисками.

Этот модуль предоставляет четыре основные функции инспекции:

- **Инспекция ресурсов** для автоматизированной оценки кластеров, узлов, подов, сертификатов и других ресурсов платформы с целью выявления рисков и паттернов использования
- **Мониторинг в реальном времени** для отслеживания прогресса задач инспекции и мгновенного отображения состояния работы ресурсов
- **Визуальная отчетность** для интуитивного отображения результатов инспекции, включая риски ресурсов, информацию об использовании и операционные данные
- **Генерация отчетов** для скачивания отчетов инспекции в форматах PDF или Excel с подробным анализом и рекомендациями

Интегрируя эти возможности с управлением доступом на основе ролей и автоматизированными алгоритмами оценки, модуль позволяет организациям снижать затраты на ручную инспекцию, проактивно выявлять аномалии ресурсов, снижать бизнес-риски и поддерживать оптимальную производительность платформы посредством систематических оценок состояния.

Содержание

Ограничения использования

Ограничения использования

- Некоторые элементы инспекции на платформе зависят от наличия в кластерах установленных компонентов мониторинга. Пожалуйста, убедитесь, что в каждом кластере заранее установлен либо плагин ACP Monitoring с Prometheus, либо плагин ACP Monitoring с VictoriaMetrics.
- Инспекция платформы поддерживает отправку результатов инспекции по электронной почте. Пожалуйста, убедитесь, что конфигурация сервера уведомлений по электронной почте была выполнена заранее.

С помощью функционала инспекции контейнерной платформы пользователи могут более эффективно управлять и поддерживать контейнерную среду, повышая стабильность и безопасность системы.

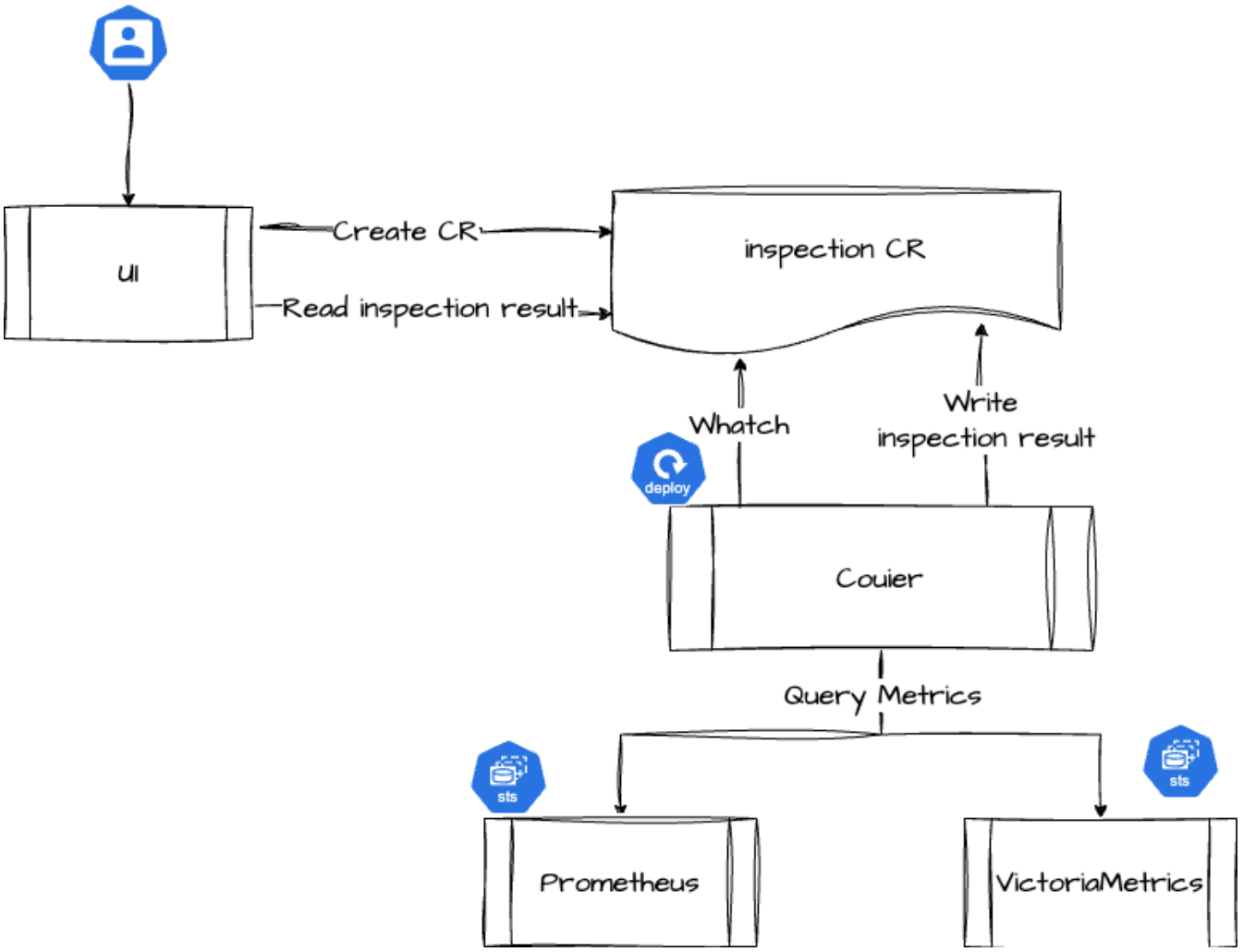
Архитектура

Содержание

Inspection

Component Health Status

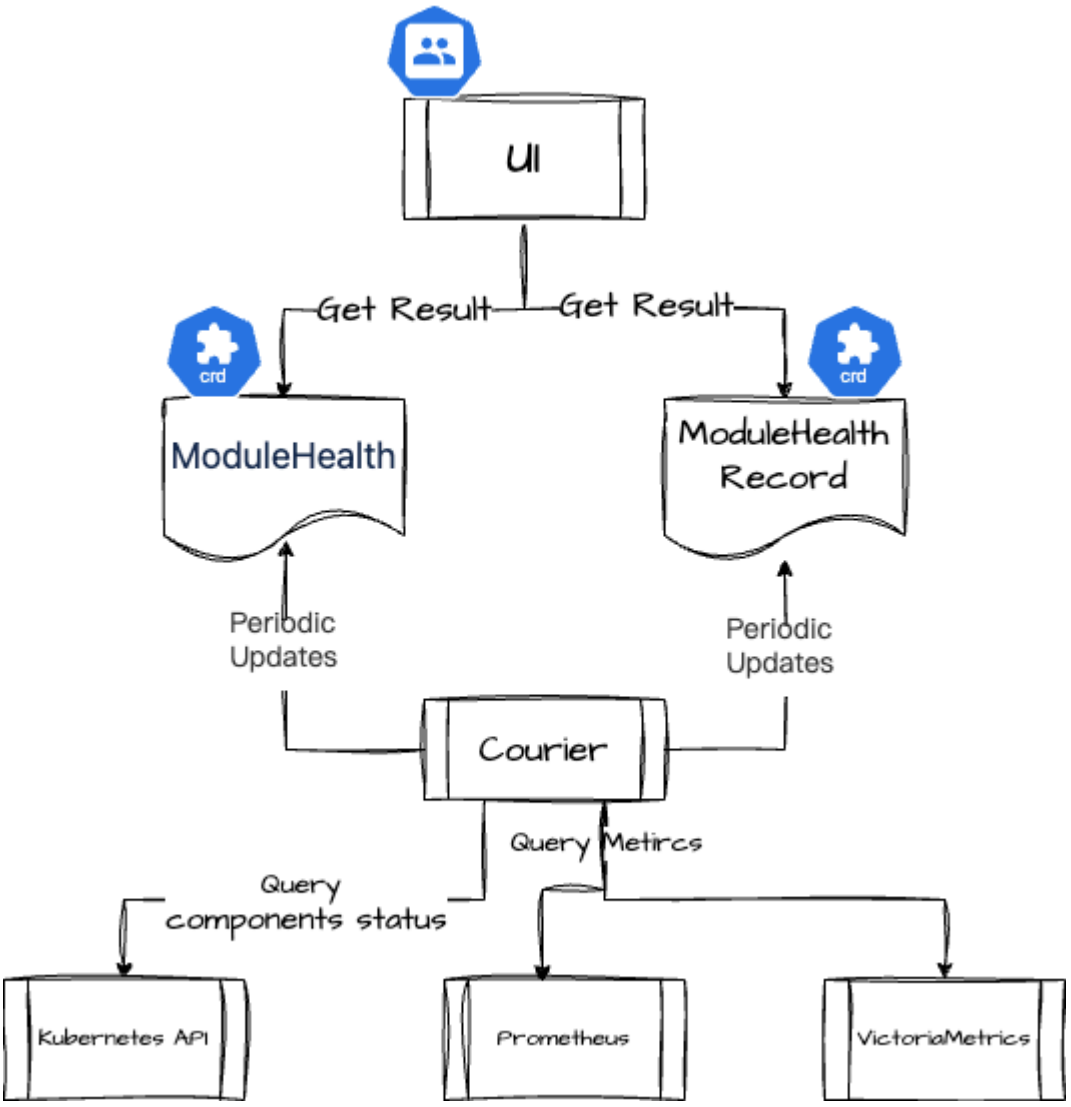
Inspection



Модуль инспекции совместно предоставляется платформенным компонентом Courier и компонентом мониторинга, включая следующие бизнес-процессы:

- Создание задачи инспекции: Платформа отправляет CR типа инспекции в кластер `global`.
- Выполнение задачи инспекции: Компонент Courier отслеживает создание CR типа инспекции и запрашивает у компонентов мониторинга каждого кластера различные метрики, связанные с инспекцией.
- Запись результатов инспекции: После завершения оценки каждого пункта инспекции компонент Courier записывает результаты инспекции обратно в соответствующий CR инспекции.
- Просмотр результатов инспекции: Пользователи могут проверить статус и результаты задач инспекции через платформу, где данные будут получены из соответствующего CR инспекции.

Component Health Status



Состояние здоровья компонентов совместно предоставляется платформенным компонентом Courier и компонентом мониторинга, включая следующие бизнес-процессы:

- Предопределённый список мониторинга компонентов: Платформа имеет предопределённые два типа CRD в кластере `global` для определения списка компонентов для мониторинга и методов мониторинга:
  - ModuleHealth: Определяет компоненты, которые необходимо мониторить, и методы мониторинга.
  - ModuleHealthRecord: Определяет результаты мониторинга соответствующих компонентов в каждом кластере.
- Регулярный мониторинг состояния компонентов: Courier отслеживает ModuleHealth, проверяет указанные функции и затем записывает результаты инспекции в CR ресурсы ModuleHealth и ModuleHealthRecord.

- Определение состояния компонентов: Courier запрашивает данные у Kubernetes и компонентов мониторинга для определения фактического состояния компонентов и существующих проблем.
  - Kubernetes: Проверяет, установлен ли компонент и нормальное ли количество реплик компонента.
  - Prometheus / VictoriaMetrics: На основе метрик, предоставляемых каждым компонентом, запрашивает и определяет, может ли компонент нормально предоставлять сервисы.
- Просмотр состояния здоровья компонентов: Пользователи могут проверить состояние здоровья каждого компонента через платформу, где данные будут получены из соответствующих CR ресурсов ModuleHealth и ModuleHealthRecord.

Руководства

Inspection

Execute Inspection

Inspection Configuration

Inspection Report Explanation

Статус здоровья компонентов

Процедуры работы

Inspection

Содержание

- Execute Inspection
- Inspection Configuration
- Inspection Report Explanation
  - Most Recent Inspection
  - Resource Risk Inspection
  - Resource Utilization Inspection

Execute Inspection

1. Нажмите **Operation Center** > **Inspection** > **Basic Inspection** в левой навигационной панели.
   Совет: На странице инспекции отображается информация об инспекции с последней проверки. Во время процесса инспекции вы можете в реальном времени просматривать данные ресурсов завершённых проверок.
2. На странице Basic Inspection поддерживаются следующие действия:
   - **Execute Inspection**: Нажмите кнопку **Inspection** в правом верхнем углу страницы для выполнения инспекции платформы.
   - **Download Inspection Report**: Нажмите кнопку **Download Report** в правом верхнем углу страницы, выберите формат отчёта (PDF или Excel) в появившемся

диалоговом окне и нажмите для скачивания, после чего отчёт соответствующего формата будет загружен на ваш локальный компьютер.

- Отчёт инспекции в формате PDF не включает страницу с деталями рисков ресурсов;
- Отчёт инспекции в формате Excel содержит все данные инспекции;
- Поддерживается одновременная загрузка отчётов в обоих форматах.

Inspection Configuration

Inspection Configuration	Description
Scheduled Inspection	Правила времени выполнения автоматических задач, поддерживается ввод выражений Crontab. **Совет**: Нажмите на поле ввода, чтобы развернуть предустановленные платформой **Trigger Rule Templates**, выберите подходящий шаблон и быстро настройте правила триггера с минимальными изменениями.
Inspection Record Retention	Количество сохраняемых записей инспекций.
Email Notification	Выберите контакты для уведомлений по электронной почте. **Примечание**: Контакты для уведомлений должны иметь настроенный email.
Inspection Report Name	Имя, которое будет использоваться встроенным шаблоном уведомлений инспекции платформы для информирования контактов.
Inspection Configuration Items	Измените пороги предупреждений или отключите пункты инспекции в соответствии с дефолтными элементами инспекции платформы для сертификатов, хостов кластера и подов.

Inspection Report Explanation

Most Recent Inspection

В области информации **Most Recent Inspection** можно просмотреть данные последней инспекции:

- **Inspection Time**: Время начала и окончания последней инспекции.
- **Total Number of Inspection Resources**: Общее количество ресурсов (кластеры, узлы, поды, сертификаты), проверенных в последней инспекции.
- **Risks**: Количество ресурсов с рисками, включая категории **Fault** и **Warning**.

Resource Risk Inspection

На странице **Resource Risk Inspection** можно просмотреть обзор информации о рисках для кластеров `global`, собственных кластеров, подключённых кластеров, а также всех узлов, подов и сертификатов в этих кластерах.

Нажмите кнопку **Risk Details** на карточке соответствующего типа ресурса (**Cluster**, **Node**, **pod**, **Certificate**), чтобы перейти на страницу с деталями рисков для данного типа ресурса. На странице деталей можно просмотреть информацию о последней инспекции ресурса, а также список ресурсов с ошибками и предупреждениями.

- Нажмите на имя ресурса, чтобы перейти на страницу деталей ресурса.
- Нажмите кнопку раскрытия справа от поля **Name** в списке, чтобы развернуть условия и причины срабатывания ошибок и предупреждений.

Для объяснения критериев оценки состояния риска (Fault, Warning) для каждого ресурса смотрите таблицу ниже.

Примечание: Для оценки ошибок и предупреждений каждого типа ресурса используется несколько условий; если данные инспекции ресурса соответствуют хотя бы одному из условий, это считается данными с риском.

Resource Type	Inspection Scope	Fault Judgment Conditions	Warning Judgment Conditions
Cluster	- `global` cluster - Self-built cluster - Accessed cluster	- Статус кластера **Abnormal**; - Нарушено соединение с apiserver	- После увеличения масштаба кластера (число узлов/подов/метрик) не обновлены конфигурации ресурсов компонентов мониторинга. - После увеличения объёма логов и частоты их сбора не обновлены конфигурации ресурсов лог-компонентов. - Использование CPU кластера превышает 60%; - Использование памяти кластера превышает 60%; - Любой под компонента ETCD кластера находится в состоянии, отличном от Running; - Любой хост кластера находится в состоянии, отличном от Ready; - Разница во времени между любыми двумя узлами кластера превышает 40 секунд; - Процент запросов CPU кластера (фактические запросы / всего) превышает 60%; - Процент запросов памяти кластера (фактические запросы / всего) превышает 80%; - Компоненты мониторинга не установлены в кластере; - Компоненты мониторинга кластера работают с ошибками;

Resource Type	Inspection Scope	Fault Judgment Conditions	Warning Judgment Conditions
			- Любой под компонента **kube-controller-manager** кластера находится в состоянии, отличном от Running; - Любой под компонента **kube-scheduler** кластера находится в состоянии, отличном от Running; - Любой под компонента **kube-apiserver** кластера находится в состоянии, отличном от Running.
Node	- Все управляющие узлы - Все вычислительные узлы	- Статус узла **Abnormal**; - Под компонента **node-exporter** на узле находится в состоянии, отличном от Running; - Под компонента **kubelet** на узле находится в состоянии, отличном от Running.	- Свободных inode на узле менее 1000 - Использование CPU узла превышает 60%; - Использование памяти узла превышает 60%; - Использование дискового пространства каталога узла превышает 60%; - Системная нагрузка узла превышает 200% и длится более 15 минут; - За последние сутки произошло как минимум одно событие NodeDeadlock (зависание узла); - За последние сутки произошло как минимум одно событие NodeOOM (недостаток памяти); - За последние сутки произошло как минимум одно событие

Resource Type	Inspection Scope	Fault Judgment Conditions	Warning Judgment Conditions
			NodeTaskHung (зависание задачи).
pod	Все поды	- Статус пода **Error**; - Под находится в состоянии запуска более 5 минут.	- Использование CPU пода превышает 80%; - Использование памяти пода превышает 80%; - Количество перезапусков пода за последние 5 минут больше или равно 1.
Certificate	- Сертификаты Certmanager - Сертификаты Kubernetes	Статус сертификата **Expired**.	Срок действия сертификата менее 29 дней.

Resource Utilization Inspection

Нажмите вкладку **Resource Utilization Inspection**, чтобы перейти на страницу **Resource Utilization Inspection**.

На странице **Resource Utilization Inspection** можно просмотреть общий объём, использование и процент использования CPU, памяти и диска для кластеров `global`, подключённых и собственных кластеров, а также количество ресурсов, таких как кластеры, узлы, поды и проекты на платформе.

- **Resource Usage Statistics**: Можно просмотреть общий объём и общий процент использования CPU, памяти и диска для глобальных, подключённых и собственных кластеров.
- **Platform Resource Quantity**: Можно просмотреть количество ресурсов, работающих на платформе.

Статус здоровья компонентов

Страница статуса здоровья платформы отображает статистические данные о состоянии здоровья установленных на платформе функций. Если у вашей учетной записи есть права управления или аудита, связанные с платформой, вы также можете просматривать подробные данные о состоянии здоровья конкретных функций, включая: список кластеров, в которых функция не установлена, статус здоровья кластеров с установленной функцией и данные обнаружения компонентов внутри кластеров, связанных с функцией. Это поможет быстро выявлять проблемы и повышать операционную эффективность платформы.

Содержание

Процедуры работы

Процедуры работы

1. Перейдите на страницу просмотра установленных продуктов или в центр платформы (управление платформой, управление проектами, центр операций).
2. Нажмите кнопку с вопросительным знаком в правом верхнем углу панели навигации > **Platform Health Status**.
3. Проверьте карточку функции; на карточке отображается информация о состоянии здоровья функции. Если в компонентах функции обнаружены аномалии, это будет отражено на карточке как `fault`.

4. Нажмите на значение health/fault на карточке функции, чтобы развернуть подробную страницу состояния здоровья справа, где можно просмотреть детальную информацию о проблемах в неисправных компонентах.