Menu

Наблюдаемость

Обзор

Обзор

Мониторинг

Введение

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

Основные понятия

Monitoring

Alarms

Notifications

Monitoring Dashboard

Руководства

Как сделать

Распределённое трассирование

Введение

Ограничения использования

Установка

Установка Jaeger Operator

Развёртывание экземпляра Jaeger

Установка OpenTelemetry Operator

Развёртывание экземпляров OpenTelemetry

Включение переключателя функций

Удаление Tracing

Архитектура

Основные компоненты

Поток данных

Основные понятия

Телеметрия

OpenTelemetry

Span

Trace

Инструментирование

OpenTelemetry Collector

Jaeger

Руководства

Как сделать

Устранение неполадок

Логи

Введение

Установка

Планирование установки

Установка Alauda Container Platform Log Storage с ElasticSearch через консоль

Установка Alauda Container Platform Log Storage с ElasticSearch через YAML

Установка Alauda Container Platform Log Storage с Clickhouse через консоль

Установка Alauda Container Platform Log Storage с Clickhouse через YAML

Установка плагина Alauda Container Platform Log Collector

Установка плагина Alauda Container Platform Log Collector через YAML

Архитектура

Основные понятия

Open Source Components

Основные понятия функциональности

Ключевые технические термины

Модель потока данных

Руководства

Как сделать

События

Введение

Ограничения использования

События

Процедуры работы

Обзор событий

Инспекция

Введение

Ограничения по использованию

Архитектура

Inspection

Component Health Status

Руководства

Menu

Обзор

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

Этот модуль объединяет четыре основных компонента 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.

Menu

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 мониторинга

Общее описание архитектуры

Система мониторинга

Система оповещений

Система уведомлений

Руководство по выбору компонента мониторинга

Важные замечания

Список компонентов

Сравнение архитектур

Сравнение функций

Рекомендации по схеме установки

Планирование ёмкости компонента мониторинга

Предположения и методология

Prometheus

VictoriaMetrics

Основные понятия

Основные понятия

Monitoring

Alarms

Notifications

Monitoring Dashboard

Руководства

Управление метриками

Просмотр метрик, предоставляемых компонентами платформы

Просмотр всех метрик, хранящихся в Prometheus / VictoriaMetrics

Просмотр всех встроенных метрик, определённых платформой

Интеграция внешних метрик

Управление оповещениями

Обзор функции

Ключевые возможности

Преимущества функции

Создание политик оповещений через UI

Создание оповещений по ресурсам через CLI

Создание оповещений по событиям через CLI

Создание политик оповещений через alert Templates

Настройка заглушения оповещений

Рекомендации по настройке правил оповещений

Управление уведомлениями

Обзор функции

Основные функции

Notification Server

Notification Contact Group

Notification Template

Notification rule

Настройка правила уведомления для проектов

Управление мониторинговыми панелями

Обзор функции

Управление панелями

Управление панелями

Создание панелей мониторинга через CLI

Общие функции и переменные

Управление Probe

Обзор функции

Мониторинг Blackbox

Оповещения Blackbox

Настройка модуля мониторинга BlackboxExporter

Создание элементов мониторинга Blackbox и оповещений через CLI

Справочная информация

Как сделать

Резервное копирование и восстановление данных мониторинга Prometheus

Обзор функции

Сценарии использования

Предварительные требования

Порядок выполнения операций

Результаты операции

Дополнительная информация

Следующие шаги

Резервное копирование и восстановление данных мониторинга VictoriaMetrics

Обзор функции

Сценарии использования

Предварительные условия

Процедуры

Результат операции

Дополнительная информация

Последующие действия

Сбор сетевых данных с сетевых интерфейсов с пользовательскими именами

Обзор функции

Сценарий использования

Предварительные требования

Порядок действий

Результаты операции

Дополнительная информация

Последующие действия

Menu

Введение

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

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 персоналу эксплуатации

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

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

Перед установкой компонентов мониторинга убедитесь, что выполнены следующие условия:

- Выбран соответствующий компонент мониторинга, руководствуясь Руководством по выбору компонента мониторинга.
- При установке в рабочем кластере убедитесь, что кластер `global` может получить доступ к порту 11780 рабочего кластера.
- Если необходимо использовать классы хранилищ или постоянные тома 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 установлены на основе рекомендуемых значений нагрузочного тестирования платформы

Parameter	Description
	- Можно выбрать или настроить квоты в зависимости от фактического масштаба кластера - Значения по умолчанию будут обновляться с версиями платформы; для фиксированных конфигураций рекомендуется использовать пользовательские настройки
Storage Type	- **LocalVolume**: Локальное хранилище с данными, сохранёнными на указанных узлах - **StorageClass**: Автоматически создаёт постоянные тома с помощью класса хранилища - **PV**: Использует существующие постоянные тома **Примечание**: Конфигурация хранилища не может быть изменена после установки
Replica Count	Устанавливает количество подов компонента мониторинга **Примечание**: Prometheus поддерживает только установку на одном узле
Parameter Configuration	Параметры данных для компонента мониторинга могут быть отрегулированы 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:

```
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
```

Пример настройки ресурсов, например для prometheus:

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

Для подробностей см. Планирование ёмкости компонентов мониторинга

Справка по полям YAML (VictoriaMetrics):

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

Field path	Description
`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 множественные реплики.
`spec.config.components.vmstorage.retention`	Количество дней хранения данных для vmstorage.
`spec.config.components.vmagent.scrapeInterval`	Интервал сбора в секундах; применяется к ServiceMonitors без `interval`.
`spec.config.components.vmagent.scrapeTimeout`	Таймаут сбора 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.

Field path	Description
`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`.

3. Verify installation

Так как имя ModuleInfo изменяется при создании, найдите ресурс по метке для проверки статуса плагина и версии:

```
kubectl get moduleinfo -l cpaas.io/module-name=victoriametrics
NAME                                          CLUSTER         MODULE
DISPLAY_NAME     STATUS    TARGET_VERSION   CURRENT_VERSION   NEW_VERSION
global-e671599464a5b1717732c5ba36079795          global          victoriametrics
victoriametrics  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, предназначенная для установки
- `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 установлены на основе рекомендуемых значений нагрузочного тестирования платформы - Можно выбрать или настроить квоты в зависимости от фактического масштаба кластера - Значения по умолчанию будут обновляться с версиями платформы; для фиксированных конфигураций рекомендуется использовать пользовательские настройки
Install Agent Only	- Off: Устанавливает полный набор компонентов VictoriaMetrics - On: Устанавливает только компонент сбора VMAgent, который зависит от VictoriaMetrics Center

Parameter	Description
VictoriaMetrics Center	Выберите кластер, в котором установлен полный набор компонентов VictoriaMetrics
Storage Type	- **LocalVolume**: Локальное хранилище с данными, сохранёнными на указанных узлах - **StorageClass**: Автоматически создаёт постоянные тома с помощью класса хранилища - **PV**: Использует существующие постоянные тома
Replica Count	Устанавливает количество подов компонента мониторинга: - Тип хранилища LocalVolume не поддерживает множественные реплики - Для других типов хранилища следуйте подсказкам на экране для настройки
Parameter Configuration	Параметры данных для компонента мониторинга могут быть отрегулированы **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:

```
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
```

Пример настройки ресурсов, например для prometheus:

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

Для подробностей см. Планирование ёмкости компонентов мониторинга

Справка 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`, `StorageClass` или `PV`.

Field path	Description
`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 в секундах; применяется к ServiceMonitors без `interval`.
`spec.config.components.prometheus.scrapeTimeout`	Таймаут сбора в секундах; должен быть меньше `scrapeInterval`.
`spec.config.components.prometheus.resources`	Настройки ресурсов для Prometheus.

Field path	Description
`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`.

3. Verify installation

Так как имя ModuleInfo изменяется при создании, найдите ресурс по метке для проверки статуса плагина и версии:

```
kubectl get moduleinfo -l cpaas.io/module-name=prometheus
NAME                                             CLUSTER         MODULE
DISPLAY_NAME   STATUS    TARGET_VERSION   CURRENT_VERSION   NEW_VERSION
global-e671599464a5b1717732c5ba36079795          global          prometheus    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, предназначенная для установки
- `CURRENT_VERSION`: версия до установки
- `NEW_VERSION`: последняя доступная версия для установки

Menu

Архитектура

Архитектура модуля мониторинга

Общее описание архитектуры

Система мониторинга

Система оповещений

Система уведомлений

Руководство по выбору компонента мониторинга

Важные замечания

Список компонентов

Сравнение архитектур

Сравнение функций

Рекомендации по схеме установки

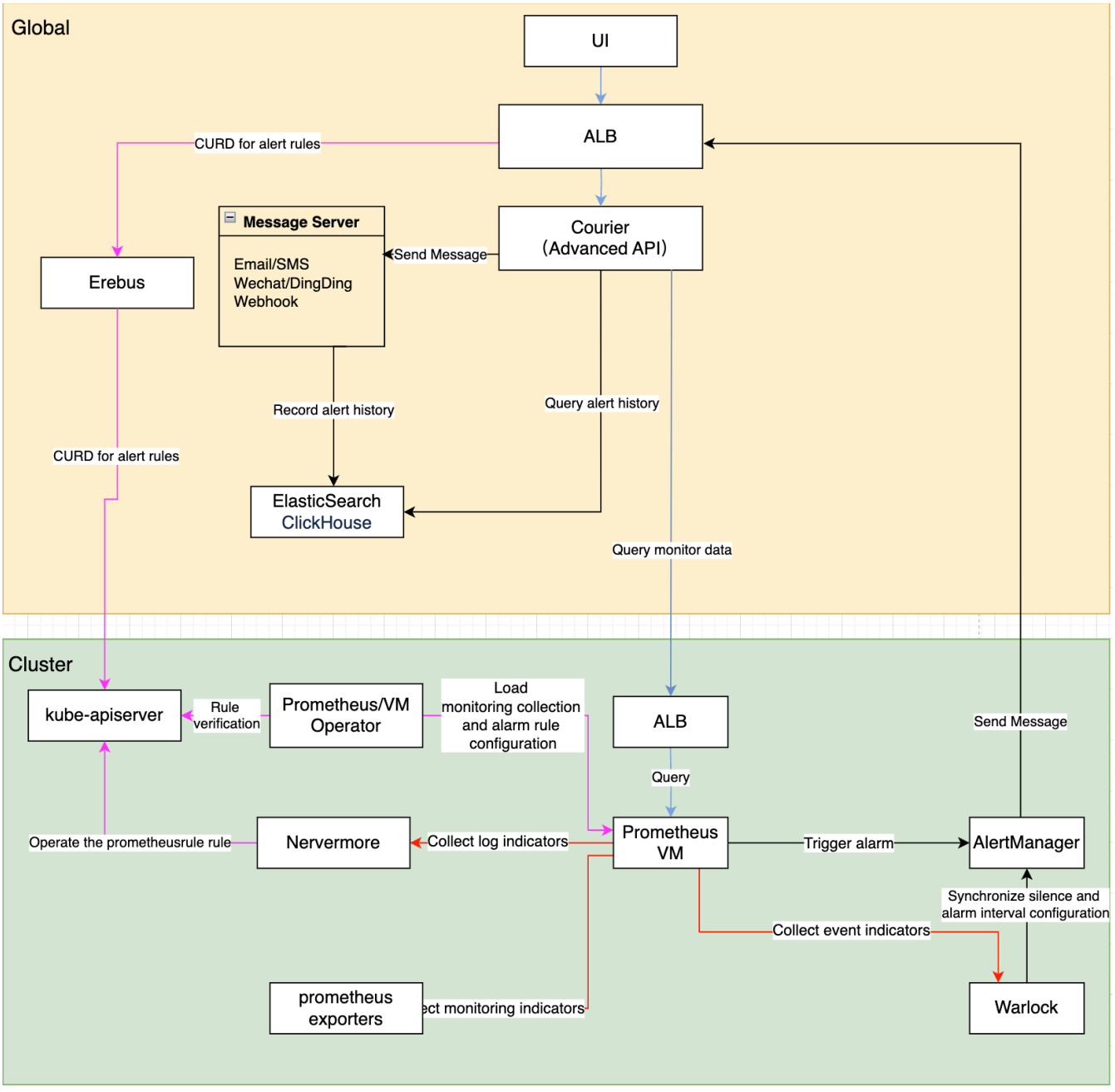
Планирование ёмкости компонента мониторинга

Предположения и методология

Prometheus

VictoriaMetrics

Архитектура модуля мониторинга



Содержание

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 запроса 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
- Статус оповещений в реальном времени: предоставление актуального статуса оповещений системы

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 оператора Prometheus/VictoriaMetrics:

- Загрузка и валидация конфигураций сбора мониторинга
- Загрузка и валидация конфигураций правил оповещений
- Синхронизация конфигураций с экземплярами 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 компоненту 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 компоненту Erebus
- Erebus маршрутизирует запрос в целевой кластер мониторинга
- Компонент Warlock отвечает за:
  - Проверку корректности 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 корректности 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 оценивает правила

2. Уведомление об оповещениях:

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

3. Хранение оповещений:

- История оповещений сохраняется в ElasticSearch/ClickHouse

Статус оповещений в реальном времени

1. Сбор статуса:

- Компонент Courier кластера `global` генерирует метрики:
  - cpaas_active_alerts: текущие активные оповещения
  - cpaas_active_silences: текущие конфигурации заглушек
- 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 корректности шаблонов уведомлений
- Проверку корректности групп контактов уведомлений
- Проверку корректности 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 корректности 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

- 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 для небольших масштабов.

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 для выполнения запросов
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 — база данных временных рядов и решение для мониторинга следующего поколения с высокими показателями производительности, обладающее следующими преимуществами:

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

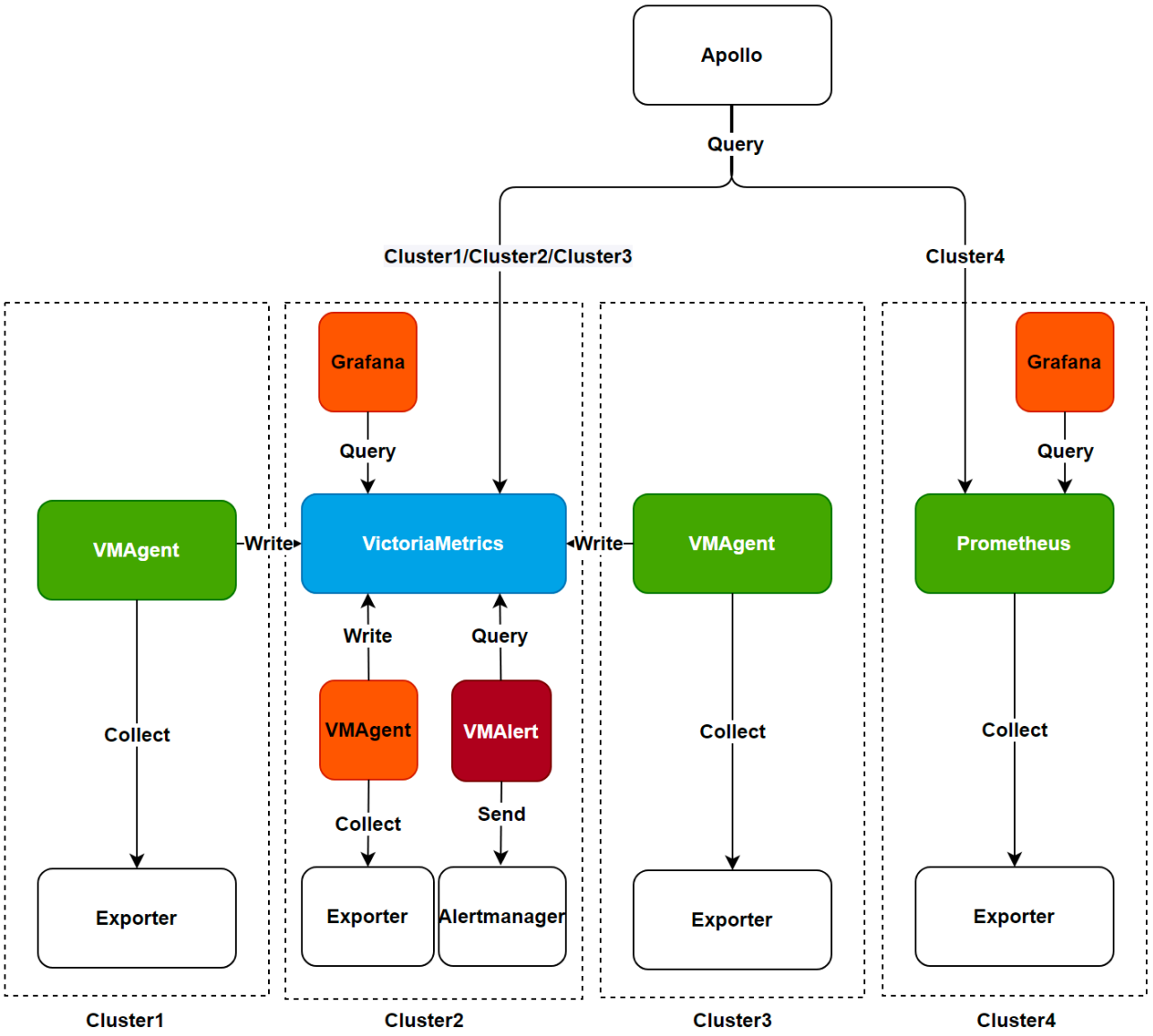
- Родная поддержка высокой доступности кластера.
- Более простое управление эксплуатацией и обслуживанием.

Сравнение функций

Функция	Prometheus	VictoriaMetrics	Описание
Установка с высокой доступностью	❌	✅	VictoriaMetrics поддерживает настоящую высокую доступность кластера 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 развивающаяся	У Prometheus более крупная экосистема сообщества

Рекомендации по схеме установки

Обзор архитектуры установки мониторинга



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

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

Метод установки Prometheus

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

- Использует компоненты Prometheus для сбора и обработки данных мониторинга.
- 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**: убедитесь, что сервисы VictoriaMetrics установлены в кластере мониторинга перед установкой VMAgent.

Рекомендации по выбору

Сценарии, подходящие для использования VictoriaMetrics

- **Потребности в высокой производительности и масштабируемости**: подходят сценарии мониторинга с высокими потоками данных и долговременным хранением.

- **Оптимизация затрат**: необходимость оптимизировать затраты на хранение и вычислительные ресурсы.
- **Требования к высокой доступности**: требуется обеспечение 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 существующей системой Prometheus**: уже 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, высокая стоимость миграции.

Планирование ёмкости компонента мониторинга

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

Содержание

Предположения и методология

- Данные в этом документе получены из контролируемых лабораторных отчетов по производительности и предназначены в качестве базового ориентира для планирования в продакшене.

- В примерах для дискового пространства задан срок хранения 7 дней; для других сроков хранения корректируйте пропорционально.
- Базовые параметры хранения соответствуют приведённому 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 размеру для Prometheus и связанных компонентов (Thanos Query, Thanos Sidecar и др.) в зависимости от масштаба.

Малый масштаб — 10 воркер-нод, 500 подов с двумя контейнерами

- Скорость приёма метрик: ~2800 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	П
courier-api	courier	2	2C	4Gi	-	-
kube-prometheus-thanos-query	thanos-query	1	1C	1Gi	-	-
prometheus-kube-prometheus-0	prometheus	1	2C	8Gi	20G	~ 3

Средний масштаб — 50 воркер-нод, 2000 подов с двумя контейнерами

- Скорость приёма метрик: ~7294 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	П
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	~ з

Большой масштаб — 500 воркер-нод, 10000 подов с двумя контейнерами

- Скорость приёма метрик: ~41575 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	
courier-api	courier	2	6C	4Gi	-	-
kube-prometheus-thanos-query	thanos-query	1	2C	6Gi	-	В р м и 2
prometheus-kube-prometheus-0	prometheus	1	8C	20Gi	100G	П ~ з д

VictoriaMetrics

Ниже приведены рекомендации по размеру для компонентов VictoriaMetrics в зависимости от масштаба.

Малый масштаб — 10 воркер-нод, 500 подов с двумя контейнерами

- Скорость приёма метрик: ~3274 сэмплов/секунду

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

Средний масштаб — 50 воркер-нод, 2000 подов с двумя контейнерами

- Скорость приёма метрик: ~6940 сэмплов/секунду

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	П
courier-api	courier	2	4C	4Gi	-	-
vmselect-cluster	proxy	1	1C	200Mi	-	-
vmselect	vmselect	1	2C	2Gi	-	-

Компонент	Контейнер	Реплики	Лимит CPU	Лимит памяти	Диск (если применимо)	П
vmstorage-cluster	vmstorage	1	2C	2Gi	10G	~ за д

Большой масштаб — 500 воркер-нод, 10000 подов с двумя контейнерами

- Скорость приёма метрик: ~34300 сэмплов/секунду

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

Основные понятия

Содержание

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 данных метрик из системы мониторинга.

Built-in Indicators

Платформа содержит предустановленный набор часто используемых метрик мониторинга, основанных на многолетнем опыте эксплуатации. Вы можете использовать эти метрики напрямую при настройке правил оповещений или создании панелей мониторинга без дополнительной конфигурации.

Exporter

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

- Сбор исходных данных мониторинга из 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

- Настройки интерфейсов сбора метрик
- Параметры выполнения задач сбора (интервалы, тайм-ауты и т.д.)

Alarms

Alarm Rules

Правила оповещений определяют конкретные условия срабатывания оповещений:

- Alarm Expression: описание условий срабатывания оповещения с помощью выражений PromQL
- Alarm Threshold: явные граничные значения для срабатывания
- Duration: продолжительность, в течение которой условия должны непрерывно выполняться
- Alarm Level: различение степени серьёзности оповещений (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 об оповещениях
- 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 данных

Monitoring Dashboard

Dashboard

Дашборд — это коллекция нескольких связанных панелей, предоставляющая общий обзор состояния системы. Поддерживает гибкое расположение элементов 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 в качестве заполнителей значений и могут использоваться в запросах метрик. С помощью селектора переменных в верхней части дашборда можно динамически изменять условия запросов, что позволяет обновлять содержимое графиков в реальном времени.

Menu

Руководства

Управление метриками

Просмотр метрик, предоставляемых компонентами платформы

Просмотр всех метрик, хранящихся в Prometheus / VictoriaMetrics

Просмотр всех встроенных метрик, определённых платформой

Интеграция внешних метрик

Управление оповещениями

Обзор функции

Ключевые возможности

Преимущества функции

Создание политик оповещений через UI

Создание оповещений по ресурсам через CLI

Создание оповещений по событиям через CLI

Создание политик оповещений через alert Templates

Настройка заглушения оповещений

Рекомендации по настройке правил оповещений

Управление уведомлениями

Обзор функции

Основные функции

Notification Server

Notification Contact Group

Notification Template

Notification rule

Настройка правила уведомления для проектов

Управление мониторинговыми панелями

Обзор функции

Управление панелями

Управление панелями

Создание панелей мониторинга через CLI

Общие функции и переменные

Управление Probe

Обзор функции

Мониторинг Blackbox

Оповещения Blackbox

Настройка модуля мониторинга BlackboxExporter

Создание элементов мониторинга Blackbox и оповещений через CLI

Справочная информация

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

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 'TYPE\|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 concurrent
reconciles per controller
# TYPE controller_runtime_max_concurrent_reconciles gauge
# HELP controller_runtime_reconcile_errors_total Total number of reconciliation errors
per controller
# TYPE controller_runtime_reconcile_errors_total counter
# HELP controller_runtime_reconcile_time_seconds Length of time per reconciliation per
controller
```

Просмотр всех метрик, хранящихся в Prometheus / VictoriaMetrics

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

Предварительные условия

1. У вас есть пользовательский токен
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>/prometheus/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 платформой встроено большое количество часто используемых метрик. Вы можете напрямую использовать эти метрики при настройке оповещений или панелей мониторинга без необходимости их самостоятельного определения. Ниже описано, как просмотреть эти метрики.

Предварительные условия

1. У вас есть пользовательский токен
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 component",
   "displayNameEN": "CPU utilization of the pods",
   "displayNameZH": "CPU utilization of containers in the compute component",
   "en": "Cpu utilization for pods in workload",
   "features": "SupportDashboard", (2)
   "summaryEN": "CPU usage rate {{.externalLabels.comparison}}
{{.externalLabels.threshold}} of Pod ({{.labels.pod}})",
   "summaryZH": "CPU usage rate {{.externalLabels.comparison}}
{{.externalLabels.threshold}} of pod ({{.labels.pod}})"
  },
  "displayName": "Использование CPU контейнерами в компоненте вычислений",
  "kind": "workload",
  "multipleEnabled": true,  (3)
  "name": "workload.pod.cpu.utilization",
  "query": "avg by (kind,name,namespace,pod) (avg by
(kind,name,namespace,pod,container)
(cpaas_advanced_container_cpu_usage_seconds_total_irate5m{kind=~\"
{{.kind}}\",name=~\"{{.name}}\",namespace=~\"
{{.namespace}}\",container!=\"\",container!=\"POD\"}) / avg by
(kind,name,namespace,pod,container)
(cpaas_advanced_kube_pod_container_resource_limits{kind=~\"{{.kind}}\",name=~\"
{{.name}}\",namespace=~\"{{.namespace}}\",resource=\"cpu\"}))", (4)
  "summary": "Уровень использования CPU {{.externalLabels.comparison}}
{{.externalLabels.threshold}} пода ({{.labels.pod}})",
  "type": "metric",
  "unit": "%",
  "legend": "{{.namespace}}/{{.pod}}",
  "variables": [ (5)
   "namespace",
   "name",
   "kind"
  ]
 }
]
```

1. Поддерживает ли эта метрика настройку оповещений
2. Поддерживает ли эта метрика использование в панелях мониторинга
3. Поддерживает ли эта метрика настройку оповещений для нескольких ресурсов
4. Определённый для метрики PromQL-запрос
5. Переменные, которые можно использовать в PromQL-запросе 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 условия

Вы установили своё 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

```yaml
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`, описывающий метрики, предоставляемые 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 отслеживаться оператором.

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

ServiceMonitor не находится в serviceMonitorNamespaceSelector Prometheus CR, он не будет отслеживаться 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; поддерживаются также bearer token, tls-аутентификация и другие методы.

3. Проверьте, отслеживается 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

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 рамках своих полномочий для эффективного мониторинга ресурсов платформы.

Ключевые возможности

- **Встроенные системные политики оповещений**: Предустановлены богатые правила оповещений, основанные на типичных сценариях диагностики сбоев для кластеров `global` и рабочих кластеров.
- **Пользовательские правила оповещений**: Поддерживается создание правил оповещений на основе различных источников данных, включая предустановленные метрики мониторинга, пользовательские метрики, 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 для операционного и технического персонала и администраторов.

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 операционную эффективность, а разделение просмотров оповещений облегчает персоналу с разными ролями быстро находить соответствующие 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 на основе метрик мониторинга).
- В целевом кластере установлены компоненты хранения логов и сбора логов (требуется при создании политик оповещений на основе логов и событий).

Процедура

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

Выбор типа оповещения

Оповещение по ресурсу

- Типы оповещений классифицированы по типу ресурса (например, состояние deployment в namespace).
- Описание выбора ресурса:
  - По умолчанию "Any", если параметр не выбран, поддерживается автоматическая привязка к вновь добавленным ресурсам.
  - 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 классифицированы по конкретным событиям (например, аномальное состояние Pod).
- По умолчанию выбираются все ресурсы под указанным ресурсом, поддерживается автоматическая привязка к вновь добавленным ресурсам.

Настройка правил оповещений

Нажмите **Add Alert Rule** и настройте параметры в соответствии с типом 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 платформой единиц
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 др.), 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 такое же, как для уровней 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 будут отправляться только при срабатывании и восстановлении оповещения.

Дополнительные примечания

1. В разделе "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 на основе метрик мониторинга).
- В целевом кластере установлены компоненты хранения логов и сбора логов (требуется при создании политик оповещений на основе 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. В следующем примере создаётся новая политика оповещений с именем policy:

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  annotations:
    alert.cpaas.io/cluster: global # Имя кластера, где расположено оповещение
    alert.cpaas.io/kind: Cluster # Тип ресурса
    alert.cpaas.io/name: global # Объект ресурса, поддерживает одиночный,
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","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 правила оповещения
      rules:
        - alert: cluster.pod.status.phase.not.running-tx1ob-e998f0b94854ee1eade5ae79279e00
          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 оповещение
            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 для правила
оповещения в секундах
            alert_indicator_blackbox_name: '' # Имя black-box мониторинга
            alert_indicator_comparison: '>' # Метод сравнения для правила оповещения
            alert_indicator_query: '' # Запрос к логам правила оповещения (только для
лог-оповещений)
            alert_indicator_threshold: '2' # Порог для правила оповещения
            alert_indicator_unit: '' # Единица измерения индикатора правила
оповещения
            alert_involved_object_kind: Cluster # Тип объекта, к которому относится
правило оповещения:
Cluster|Node|Deployment|Daemonset|Statefulset|Middleware|Microservice|Storage|VirtualMachi
            alert_involved_object_name: global # Имя объекта, к которому относится
правило оповещения
            alert_involved_object_namespace: '' # Namespace объекта, к которому относит
правило оповещения
            alert_name: cluster.pod.status.phase.not.running-tx1ob # Имя правила
оповещения
            alert_namespace: cpaas-system # Namespace, где расположено правило
оповещения
            alert_project: cpaas-system # Имя проекта объекта, к которому относится
правило оповещения
            alert_resource: policy # Имя политики оповещений, где расположено
правило оповещения
            alert_source: Platform # Тип данных политики оповещений: Platform - данны
платформы, Business - бизнес-данные
            severity: High # Уровень серьёзности правила оповещения: Critical -
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 на основе метрик мониторинга).

- В целевом кластере установлены компоненты хранения логов и сбора логов (требуется при создании политик оповещений на основе 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. В следующем примере создаётся новая политика 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 по событию. 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{namespace=~".*",id="policy2-cluster.event.count-6sial"}
[300s])))
            + (avg
            by(kind,namespace,name,reason)
(abs(increase(cpaas_event_count{namespace=~".*",id="policy2-cluster.event.count-
6sial"}[300s]))))
            / 2)>2
          # id в policy2 должен совпадать с именем политики оповещений; 6sial
должен соответствовать имени предыдущего правила оповещения
          for: 15s # Длительность
          labels:
            alert_cluster: global # Имя кластера, где расположено оповещение
            alert_for: 15s # Длительность
            alert_indicator: cluster.event.count # Имя индикатора правила
оповещения (пользовательское имя индикатора)
            alert_indicator_aggregate_range: '300' # Время агрегации для правила
оповещения в секундах
            alert_indicator_blackbox_name: ''
            alert_indicator_comparison: '>' # Метод сравнения для правила
оповещения
            alert_indicator_event_reason: ScalingReplicaSet # Причина события.
            alert_indicator_threshold: '2' # Порог для правила оповещения
            alert_indicator_unit: pieces # Единица измерения индикатора правила
оповещения; для оповещений по событиям остаётся без изменений
            alert_involved_object_kind: Event
            alert_involved_object_options: Single
            alert_name: cluster.event.count-6sial # Имя правила оповещения
            alert_namespace: cpaas-system # Namespace, где расположено правило
оповещения
            alert_project: cpaas-system # Имя проекта объекта, к которому
относится правило оповещения
            alert_repeat_interval: 5m
            alert_resource: policy2 # Имя политики оповещений, где расположено
правило оповещения
            alert_source: Platform # Тип данных политики оповещений: Platform -
данные платформы, Business - бизнес-данные
            severity: High # Уровень серьёзности правила оповещения: Critical -
критический, High - серьёзный, Medium - предупреждение, Low - информация
```

Создание политик оповещений через alert Templates

alert templates — это сочетание 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 на основе метрик 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 базовую информацию шаблона оповещений.

4. В разделе **alert Rules** нажмите **Add alert Rule** и добавьте правила оповещений согласно описаниям параметров ниже:

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 платформой единиц

Параметр	Описание
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 др.), 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.

1. 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 панель навигации.

2. Нажмите кнопку раскрытия рядом с кнопкой **Create alert Policy** > **Template Create alert Policy**.

3. Настройте некоторые параметры, опираясь на описания ниже:

Параметр	Описание
Template Name	Имя используемого шаблона оповещений. Шаблоны классифицированы по кластерам, узлам и вычислительным компонентам. При выборе шаблона можно просмотреть правила

Параметр	Описание
	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.

1. 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 настроить постоянное заглушение или заглушение на заданное 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 предотвратить такую ситуацию.

Примечание: Когда статус заглушения достигает времени окончания, 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 применением 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 удалению/добавлению заглушения в политиках оповещений; если выбрано "Select All", заглушение применяется только к текущему выбранному диапазону ресурсов, и последующие действия **Delete/Add** ресурсов не обрабатываются.

Параметр	Описание
Silence Range	Область ресурсов, на которую распространяется настройка заглушения.
Resource Name	Имя объекта ресурса, на который направлено заглушение.
Silence Time	Временной диапазон заглушения оповещений. Оповещение переходит в состояние заглушения в начале 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**: пользовательская настройка времени начала и окончания заглушения с интервалом не менее 5 минут.

5. Нажмите **Set**.

   Совет: С момента установки заглушения до начала времени заглушения статус политики оповещений считается **Silence Waiting**. В этот период при срабатывании правил уведомления отправляются как обычно; с начала заглушения до её окончания статус политики — **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-
22T08:01:37Z","creator":"leizhu@alauda.io","resources":{"nodes":
[{"name":"192.168.36.11","ip":"192.168.36.11"},
{"name":"192.168.36.12","ip":"192.168.36.12"},
{"name":"192.168.36.13","ip":"192.168.36.13"}]}}'
      # Конфигурация заглушения для политик оповещений на уровне узлов, включая время начала, окончания, создателя и т.д.; если диапазон заглушения включает конкретные узлы, добавьте информацию resources.node как показано выше. Если требуется заглушение для всех ресурсов, поле resources не нужно.
    # alert.cpaas.io/silence.config: '{"startsAt":"2025-02-
08T08:04:50Z","endsAt":"2199-12-31T00:00:00Z","creator":"leizhu@alauda.io","name":
["alb-operator-ctl","apollo"],"namespace":["cpaas-system"]}'
    # Конфигурация заглушения для политик оповещений на уровне workload, включая время начала, окончания, создателя и т.д.; если диапазон заглушения включает конкретные workload, добавьте имя и namespace как показано выше. Если требуется заглушение для всех ресурсов, поля name и namespace не нужны.
    # Установка поля endsAt в 2199-12-31T00:00:00Z означает постоянное заглушение.
    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. Использование минимального количества правил позволяет создать 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 внесением изменений**: 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. Модификация существующих правил и тщательное создание новых помогает упростить систему оповещений.
- **Предоставляйте чёткие сообщения оповещений**: При создании сообщений оповещений 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 принимать соответствующие меры.
- **Разумно устанавливайте уровни серьёзности**: Назначайте уровням серьёзности правила, чтобы указать, как пользователи должны реагировать при срабатывании оповещений. Например, классифицируйте оповещения с уровнем Critical как требующие немедленных действий соответствующего персонала. Установление

уровней серьёзности помогает пользователям принимать решения о реакции на оповещения и обеспечивает своевременное реагирование на критические проблемы.

Управление уведомлениями

Содержание

Обзор функции

С помощью уведомлений вы можете интегрировать функции мониторинга и оповещения платформы для своевременной отправки информации о предварительном

предупреждении получателям уведомлений, напоминая соответствующему персоналу принять необходимые меры для решения проблем или предотвращения сбоев.

Основные функции

- **Notification Server**: сервер уведомлений предоставляет сервисы для отправки сообщений уведомлений контактным группам на платформе, например, сервер электронной почты.
- **Notification Contact Group**: контактная группа 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. Он используется для стандартизации содержания и формата сообщений оповещений для стратегий 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 корпоративными группами 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.

```yaml
# Методы получения corpId, corpSecret, agentId для WeChat Work можно найти в официальной документации: https://developer.work.weixin.qq.com/document/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 китайском, по умолчанию закодировано в base64
  displayNameEn: WeChat             # Отображаемое имя сервера на английском, по умолчанию закодировано в base64
  corpId:                           # ID компании, по умолчанию закодировано в base64
  corpSecret:                       # Секрет приложения, по умолчанию закодировано в base64
  agentId:                          # ID корпоративного приложения, по умолчанию закодировано 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 китайском, по умолчанию закодировано в base64
  displayNameEn: DingTalk           # Отображаемое имя сервера на английском, по умолчанию закодировано в base64
  appKey:                           # Ключ приложения, по умолчанию закодировано в base64
  appSecret:                        # Секрет приложения, по умолчанию закодировано в base64
  agentId:                          # agent_id приложения, по умолчанию закодировано 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.

```yaml
# 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 закодировано в base64
  displayNameEn: Feishu              # Отображаемое имя сервера на английском, по умолчанию закодировано в base64
  appId:                             # ID приложения, по умолчанию закодировано в base64
  appSecret:                         # Секрет приложения, по умолчанию закодировано 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 ниже для настройки соответствующих параметров.

Параметр	Описание
Service Address	Адрес сервера уведомлений, поддерживающего протокол SMTP, например, `smtp.yeah.net`.

Параметр	Описание
Port	Номер порта сервера уведомлений. При включённом **Use SSL** необходимо указать порт SSL.
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 корпоративными группами WeChat, DingTalk, Feishu или отправку HTTP-запросов на указанный сервер Webhook.

Корпоративный бот группы WeChat

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

```
kubectl patch secret -n cpaas-system platform-wechat-server -p '{"data":
{"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 действий рядом с кластером `global` > **CLI Tool**.
3. Выполните следующую команду на мастер-ноде кластера `global`:

```
kubectl patch secret -n cpaas-system platform-dingtalk-server -p '{"data":
{"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 действий рядом с кластером `global` > **CLI Tool**.
3. Выполните следующую команду на мастер-ноде кластера `global`:

```
kubectl patch secret -n cpaas-system platform-feishu-server -p '{"data":
{"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 действий рядом с кластером `global` > **CLI Tool**.
3. Выполните следующую команду на мастер-ноде кластера `global`:

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

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

Notification Contact Group

Контактная группа уведомлений — это набор получателей уведомлений с похожими логическими характеристиками. Например, вы можете назначить команду эксплуатации и обслуживания в качестве контактной группы 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 настроек сервера уведомлений.
2. Если необходимо использовать сервер типа Webhook в качестве получателя уведомлений, необходимо настроить соответствующий URL в контактной группе уведомлений.

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 приведённым ниже инструкциям.

Параметр	Описание
Email	Добавьте email для всей контактной группы уведомлений. Платформа будет отправлять

Параметр	Описание
	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 способом.
Contact Configuration	Нажмите **Add Contact**, чтобы добавить существующих пользователей платформы в контактную группу. Убедитесь в корректности контактных данных выбранных контактов (телефон, email, callback интерфейс), чтобы избежать пропуска уведомлений.

4. Нажмите **Add**.

Notification Template

Шаблон уведомления — это стандартизированная структура, состоящая из настраиваемого содержимого, переменных содержимого и параметров форматирования. Он используется для стандартизации содержания и формата сообщений оповещений для стратегий уведомлений.

Администраторы платформы или операционный персонал могут создавать шаблоны уведомлений для настройки содержания и формата сообщений уведомлений в зависимости от различных способов оповещения, помогая пользователям быстро получать ключевую информацию об оповещениях и повышать эффективность работы.

INFO

Платформа поддерживает различные серверы уведомлений, и соответствующие шаблоны типов уведомлений будут отображаться в зависимости от настроек 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 оповещения, в сочетании с функцией оповещения платформы; **Component Exception Message**: отправляет уведомления, вызванные исключениями в отдельных компонентах.

5. В разделе **Template Configuration** настройте переменные и параметры форматирования содержимого, ориентируясь 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. Вы можете изменять отображаемые имена переменных и формат содержимого, но не можете изменять саму переменную. 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

покрывает большинство потребностей в настройке сообщений уведомлений. При отсутствии особых требований можно использовать содержимое шаблонов по умолчанию.

1. 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 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, в котором 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.

Отображаемое имя	Переменная	Описание
Alert Description	`{{ .externalLabels.summary }}`	Описание правила оповещения.
Trigger Value	`{{ .externalLabels.currentValue }}`	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.
Metric Name	`{{ .labels.alert_indicator }}`	Название метрики мониторинга.

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-го	Установка уровня заголовка.

Элемент содержимого	Тег	Описание
	уровня).	
Абзац	`<p>Абзац</p>`	Вставка обычного текста абзаца.
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 правил 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 настроек сервера уведомлений. Если сервер уведомлений не настроен, соответствующие режимы уведомлений по умолчанию не отображаются.

Предварительные требования

Для использования **Corporate Communication Tool Server** для уведомления контактов пользователям необходимо сначала изменить контактную информацию в **Personal Information**, введя 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 следующим инструкциям.

Параметр	Описание
Notification Contact Group	Контактная группа уведомлений — логический набор получателей уведомлений, которых платформа будет уведомлять указанным способом.
Notification Recipients	Выберите одного или нескольких получателей уведомлений, и платформа будет отправлять уведомления согласно контактным данным в **Personal Information** получателей.
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 уведомления для проектов

Стратегии уведомлений, шаблоны уведомлений и контактные группы уведомлений платформы изолированы по арендаторам. В качестве администратора проекта вы не сможете просматривать или использовать стратегии уведомлений, шаблоны уведомлений или контактные группы уведомлений, настроенные другими проектами или администраторами платформы. Поэтому вам необходимо руководствоваться этим документом для настройки подходящих стратегий уведомлений для вашего проекта.

Предварительные требования

1. Вы связались с администратором платформы для завершения настройки сервера уведомлений.
2. Если требуется уведомлять через корпоративные коммуникационные инструменты, необходимо также 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 контактную группу уведомлений.

> **TIP**
>
> Если вам не нужно управлять контактами уведомлений через контактную группу 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

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 мониторинговый обзор, что значительно повышает эффективность вашей настройки.

Основные возможности

- Поддержка настройки пользовательских панелей мониторинга как для бизнес-обзоров, так и для обзоров платформы.
- Возможность просмотра публично общих панелей, настроенных в обзорах платформы, из бизнес-обзоров с изоляцией данных по namespace, к которому принадлежит бизнес.
- Поддержка управления панелями внутри панели мониторинга: добавление, удаление, изменение панелей, масштабирование панелей и перемещение панелей с помощью drag-and-drop.
- Возможность установки пользовательских переменных внутри панели для фильтрации данных запросов.
- Поддержка настройки групп внутри панели 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 импорта панелей Grafana одним кликом.

Преимущества

- Поддержка пользовательских сценариев мониторинга без ограничений предопределёнными шаблонами, что позволяет достичь действительно персонализированного мониторинга.
- Предоставляет богатый набор вариантов визуализации, включая линейные графики, столбчатые диаграммы, круговые диаграммы, а также гибкие варианты компоновки и стилизации.
- Бесшовная интеграция с ролевыми разрешениями платформы, позволяющая бизнес-обзорам определять собственные панели мониторинга с обеспечением изоляции данных.
- Глубокая интеграция с различными функциями контейнерной платформы, обеспечивающая мгновенный доступ к данным мониторинга контейнеров, сетей, хранилищ и т.д., предоставляя пользователям всестороннее наблюдение за производительностью и диагностику неисправностей.
- Полная совместимость с JSON панелей Grafana, что облегчает миграцию с Grafana для продолжения использования.

Сценарии использования

- **Управление IT-операциями**: В составе команды IT-операций вы можете использовать панели мониторинга для унификации 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 выявлять 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 настраивать специализированные виды мониторинга 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 аналитические инструменты для быстрого выявления корневой причины. Вы можете оперативно просматривать колебания соответствующих метрик на основе информации оповещений для углублённого анализа неисправностей.

Предварительные требования

В настоящее время панели мониторинга поддерживают только просмотр данных мониторинга, собранных 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 обзоры 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 будет запрашивать 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

Панель мониторинга — 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 панели

1. Нажмите **Create Dashboard**, ориентируйтесь на следующие инструкции для настройки соответствующих параметров.

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

1. После добавления нажмите **Create** для завершения создания панели. Далее необходимо **добавить переменные**, **добавить панели** и **добавить группы** для завершения общего дизайна компоновки.

Импорт панели

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

- В настоящее время поддерживается только Grafana JSON версии V8+; более низкие версии запрещены к импорту.
- Если в импортируемой панели есть типы, не поддерживаемые платформой, они могут отображаться как **unsupported panel types**, но вы можете изменить настройки панели для нормального отображения.
- После импорта панели вы можете выполнять любые действия по управлению, 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 размерностей временных рядов. Выражение запроса может быть задано для динамического вычисления и генерации результатов.

Параметр	Описание
Query Settings	При определении настроек запроса, помимо использования PromQL для запроса временных рядов, платформа также предоставляет некоторые общие переменные и функции. Смотрите 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 панели 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 появится опция **All** для выбора всех данных переменной.

Constant

Постоянные переменные — это статические переменные с фиксированными значениями, которые не меняются на протяжении всей панели, обычно используются для хранения идентификаторов окружения, фиксированных порогов или параметров конфигурации, к которым нужно обращаться из нескольких панелей 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 панели, 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), либо 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.

Параметр	Описание
Textbox Value	Значение по умолчанию для текстового поля.

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

Добавление панелей

Добавьте несколько панелей в текущую созданную панель мониторинга для отображения данных различных ресурсов.

Совет: Размер панели можно настроить, кликнув по правому нижнему углу; для изменения порядка панелей кликните по любой области панели.

1. Нажмите **Add Panel**, ориентируйтесь на следующие инструкции для настройки параметров.

- **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 нативных пользовательских метрик. Оба метода 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 единичный формат; нельзя добавлять метрики с разными единицами в одну панель.
  - **Native**: Настройте единицу метрики, выражение метрики и параметры легенды. Выражение метрики следует синтаксису PromQL; подробности смотрите в PromQL Official Documentation ↗.
- **Legend Parameters**: Управляет именами, соответствующими кривым на панелях. Можно использовать текст или шаблоны:
  - Правило: Введённое значение должно быть в формате `{{.xxxx}}`; например, `{{.hostname}}` заменится на значение метки hostname, возвращаемой выражением.
  - **Совет**: При вводе некорректного формата параметра легенды имена кривых будут отображаться в исходном виде.
- **Instant Switch**: При включённом переключателе **Instant** будет выполняться запрос мгновенных значений через интерфейс Query Prometheus и сортировка, как в статистических и gauge диаграммах. Если выключен, используется метод `query_range` для вычисления серии данных за определённый период.

- **Panel Settings**: Поддерживается выбор различных типов панелей для визуализации метрик. Смотрите Manage Panels.

2. Нажмите **Save** для завершения добавления панелей.

3. Можно добавить одну или несколько панелей в рамках панели мониторинга.

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

   - Кликните по правому нижнему углу панели для настройки размера.
   - Кликните по любой области 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 внесения изменений нажмите кнопку **Save** на странице панели для сохранения.

Добавление групп

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

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

- **Group**: Название группы.
- **Repeat**: Поддерживается 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; панель мониторинга сгенерирует строку идентичных подгрупп для каждого соответствующего значения переменной. Подгруппы не поддерживают изменение, удаление или перемещение панелей.

1. После добавления группы доступны следующие операции для управления отображением панелей внутри панели:

- Группы можно сворачивать или разворачивать для скрытия части содержимого. Панели внутри свернутых групп не отправляют запросы.
- Перемещайте панели в группу, чтобы они управлялись этой группой. Группа управляет всеми панелями между ней и следующей группой.
- При сворачивании группы можно перемещать все панели, управляемые этой группой, вместе.
- Сворачивание и разворачивание групп также считается изменением панели. Чтобы сохранить это состояние при следующем открытии, нажмите **Save**.

Переключение панелей

Установите созданную пользовательскую панель мониторинга как основную. При повторном входе в функцию мониторинга по умолчанию будет отображаться основная панель.

1. В левом навигационном меню выберите **Operations Center** > **Monitoring** > **Monitoring Dashboards**.
2. По умолчанию открывается основная панель мониторинга. Нажмите **Switch Dashboard**.
3. Можно найти панели, фильтруя по меткам или поиском по имени, и переключать основные панели с помощью переключателя **Main Dashboard**.

Другие операции

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

Операция	Описание
YMAL	Открывает фактический код ресурса CR панели, хранящийся в Kubernetes-кластере. Вы можете изменить всё содержимое панели, редактируя параметры в YAML.

Операция	Описание
Export Expression	Экспортирует метрики и соответствующие выражения запросов, используемые в текущей панели, в формате CSV.
Copy	Копирует текущую панель; вы можете редактировать панели по необходимости и сохранить как новую панель.
Settings	Изменяет основную информацию текущей панели, например, смену меток и добавление переменных.
Delete	Удаляет текущую панель мониторинга.

Управление панелями

Платформа предоставляет различные методы визуализации для поддержки разных сценариев. В этом разделе описаны типы панелей, параметры настройки 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.

№	Название панели	Описание	Рекомендуемые сценарии использования
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, количество узлов и т.д.
4	Horizontal Bar Chart	Аналогично столбчатой диаграмме, но использует горизонтальные столбцы.	При большом количестве размерностей данных горизонтальные столбчатые 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, загрузку CPU и использование памяти системы. Рекомендуется использовать пороги с изменением цвета для индикации 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 задачи и нагрузку системы. При наличии нескольких категорий одного индикатора рекомендуется использовать именно этот тип,

№	Название панели	Описание	Рекомендуемые сценарии использования
			например, доступное дисковое пространство.
7	Pie Chart	Использует сектора для отображения пропорционального соотношения частей и целого.	Подходит для демонстрации состава данных по разным размерностям, например, доли ответов 4XX, 3XX, 2XX за период.
8	Table	Организует данные в виде 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, обычно с текстовым пояснением.	Подходит для отображения в реальном времени важных индикаторов мониторинга, например, количества подов, узлов, текущего числа 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, через распределение точек, помогая обнаружить связи между метриками.
11	Text Card	Отображает ключевую текстовую информацию в формате карточки,	Подходит для представления текстовой информации, например, описаний панелей и

№	Название панели	Описание	Рекомендуемые сценарии использования
		обычно содержит заголовок и краткое описание.	пояснений по устранению неполадок.

Описание параметров панели

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 для быстрого доступа, выбирая имя ссылки рядом с заголовком.
Standard Settings	Единицы измерения для нативных метрик. Кроме того, gauge и bar gauge 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.
Threshold Parameters	Настройка порогов для панелей; при включении пороги отображаются выбранными цветами, позволяя визуально оценивать значения относительно порогов.
Value	Метод вычисления значения, например, последнее или минимальное значение. Применимо только к stat и gauge диаграммам.
Value Mapping	Переопределение заданных значений, диапазонов, регулярных выражений или специальных значений, например, определение 100 как полной загрузки. Применимо к stat, table и gauge диаграммам.

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 тенденции изменений индикаторов, 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 в круговой диаграмме, чтобы снизить влияние категорий с низкой долей, но большим количеством. Избыточные секторы объединяются в **Others**. **Label Display Fields**: Настройка полей, отображаемых в метках круговой диаграммы.
Pie Chart	**Graph Style**	Можно выбрать стиль отображения: pie или donut.

Тип панели	Параметр	Описание
Table	**Table Settings**	**Hide Columns**: Позволяет уменьшить количество столбцов для фокусировки на основных. **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 создания панели 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 основной панелью?
  name: demo-v2-dashboard1 # Имя
  namespace: cpaas-system # Namespace (все создания в области управления происходят в этом ns)
spec:
  body: # Все информационные поля
    titleZh: 更新显示名称 # Встроенное поле для отображения названия на китайском (создаётся под китайский язык)
    title: english_display_name # Встроенное поле для отображения названия на английском (создаётся под английский язык) Встроенные панели могут иметь двуязычный перевод.
    templating: # Пользовательские переменные
      list:
        - hide: 0 # 0 - не скрывать; 1 - скрывать только метку; 2 - скрывать метку и значение
          label: 集群 # Отображаемое имя встроенной переменной (метка задаётся в зависимости от языка, например, cluster на английском)
          name: cluster # Имя встроенной переменной (уникальное)
          options: # Определение опций выпадающего списка; если запрос возвращает данные, используется он, иначе - options. Можно задать значение по умолчанию (обычно для установки дефолта)
            - selected: false # Выбрана ли по умолчанию
              text: global
              value: global
          type: custom # Тип переменной: custom (встроенная) или query (запрос). Импорт Grafana поддерживает constant, custom, interval (после импорта будет преобразовано в custom, auto не поддерживается)

        - allValue: '' # Выбрать все, передаётся в формате xxx|xxx|xxx; можно задать allValue для преобразования (Grafana возвращает все данные текущей переменной в формате xxx|xxx|xxx, корректировка для согласованности)
          current: null # Текущее значение переменной; если не задано, по умолчанию первое в списке
          definition: query_result(kube_namespace_labels) # Выражение запроса для
```

```
получения данных
          hide: 0 # 0 - не скрывать; 1 - скрывать только метку; 2 - скрывать метку и значение
          includeAll: true # Включать опцию "все"
          label: ns # Отображаемое имя переменной
          multi: true # Разрешить множественный выбор
          name: ns # Имя переменной (уникальное)
          options: []
          query: ''
          regex: /.*namespace=\"(.*?)\".*/ # Регулярное выражение для извлечения значений переменной
          sort: 2 # Сортировка: 1 - по алфавиту по возрастанию; 2 - по алфавиту по убыванию (поддерживаются только эти два); 3 - по числу по возрастанию; 4 - по числу по убыванию
          type: query # Тип переменной
    time: # Временной диапазон панели
      from: now-30m # Начало
      to: now # Конец
    repeat: '' # Конфигурация повторения строк; выбирается пользовательская переменная
    collapsed: 'false' # Конфигурация свернутости строк
    description: '123' # Описание (всплывающая подсказка после заголовка)
    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 запрашивать диапазон
        refId: 指标1 # Уникальный идентификатор для отображения имени источника данных
    gridPos: # Позиция и размер на панели
      h: 8 # Высота
      w: 12 # Ширина (из 24 сеточных единиц)
      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, stat, gauge, table, bargauge, row, text, pie (step chart, scatter plot, bar chart настраиваются через drawStyle)
      id: a2239830-492f-4d27-98f3-cb7ecb77c56f # Уникальный идентификатор
      links: # Ссылки
        - targetBlank: true # Открывать в новой вкладке
          title: '1' # Имя
```

```
          url: '1' # URL
      transformations: # Трансформации данных
        - id: 'organize' # Тип organize; используется для сортировки,
перестановки, отображения полей
          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' # Слияние данных
          options:
      fieldConfig: # Настройка свойств и внешнего вида панели
        defaults: # Настройки по умолчанию
          custom: # Пользовательские графические атрибуты
            align: 'left' # Выравнивание таблицы: left, center, right
            cellOptions: # Настройка порогов таблицы
              type: color-text # Поддерживается только текст для цветовой
настройки порогов
            spanNulls: false # true - соединять null значения; false - не соединять;
число == 0 - соединять null как 0
            drawStyle: line # Типы панелей: line, bars для столбчатых, points для
точечных
            fillOpacity: 20 # Присутствует при drawStyle area (пока не
поддерживается настройка, по умолчанию 20)
            thresholdsStyle: # Настройка отображения порогов (пока
поддерживается только линия)
              mode: line # Формат отображения порогов (area пока не
поддерживается)
            lineInterpolation: 'stepBefore' # Настройка step chart; по умолчанию
поддерживается только stepBefore (stepAfter будет добавлен позже)
          decimals: 3 # Количество знаков после запятой
          min: 0 # Минимальное значение (пока не поддерживается в конфигурации
страницы, поддерживается в адаптированных импортах)
          max: 1 # Максимальное значение (конфигурация страницы применяется
только к stat, gauge, barGauge, pie)
          unit: '%' # Единица измерения
          mappings: # Настройка отображения значений (поддерживаются value и
range; специальные типы на данных)
            - options: # Правила отображения значений
                '1': # Значение
```

```
                index: 0
                text: 'Running' # Отображается как Running при значении 1
            type: value # Тип отображения value
          - options: # Правила отображения диапазонов
              from: 2 # Начало диапазона
              to: 3 # Конец диапазона
              result: # Результат отображения
                index: 1
                text: 'Error' # Значения от 2 до 3 отображаются как Error
            type: range # Тип отображения range
          - type: special # Специальный тип отображения
            options:
              match: null # nan null null+nan empty true false
              result:
                text: xxx
                index: 2
        thresholds: # Настройка порогов
          mode: absolute # Режим порогов, абсолютные значения (пока
поддерживаются только абсолютные и процентные; процентные пока не
поддерживаются)
          steps: # Шаги порогов
            - color: '#a7772f' # Цвет порога
              value: '2' # Значение порога
            - color: '#007AF5' # Цвет по умолчанию, базовый
      overrides: # Переопределения
        - matcher:
            id: byName # Совпадение по имени поля
            options: node # Имя поля
          properties: # Переопределяемые свойства; id поддерживает displayName,
unit
            - id: displayName # Переопределение имени отображения
              value: '1' # Новое имя
            - id: unit # Переопределение единицы
              value: GB/s # Новая единица
            - id: noValue # Отображение при отсутствии значения
              value: No value display
    options:
      orientation: horizontal # Направление компоновки панелей; применяется к
gauge и barGauge (stat будет поддерживаться позже)
      legend: # Настройки легенды
        calcs: # Методы вычисления (отображаются при положении легенды
справа)
          - latest # Поддерживается только последнее значение
        placement: right # Положение легенды (right или bottom; по умолчанию
```

```
bottom)
          placementRightTop: '' # Настройка верхнего правого угла
          showLegend: true # Отображать легенду
        tooltip: # Всплывающие подсказки
          mode: multi # Режим мультивыбора (поддерживается только multi)
Отображает все данные при наведении мыши
          sort: asc # Сортировка: asc или desc
        reduceOptions: # Метод вычисления значения (для агрегирования данных)
          calcs: # Методы вычисления (latest, minimum, maximum, average, sum)
            - latest
          limit: 3 # Ограничение количества секторов в pie
        textMode: 'value' # Настройка stat; определяет стиль отображения значения
метрики; варианты: auto, value, value_and_name, name, none (пока не поддерживается
в конфигурации страницы, но поддерживается в импорте)
        colorMode: 'value' # Настройка stat; определяет цветовой режим
отображения значений; варианты: none, value, background (по умолчанию value; не
поддерживается в конфигурации, но адаптировано в импорте)
        displayLabels: ['name', 'value', 'percent'] # Поля, отображаемые в метках
pie
        pieType: 'pie' # Тип pie диаграммы; варианты pie и donut
        mode: 'html' # Режим текстовой диаграммы; варианты html и richText
        content: '<div>xxx</div>' # Содержимое текстовой диаграммы
        footer:
          enablePagination: true # Включение пагинации в таблице
```

Общие функции и переменные

Общие функции

При определении настроек запросов, 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 метриках Prometheus, например, label_values(job).

Функция	Назначение
label_values(metric, label)	Возвращает все доступные значения для имени метки в указанной метрике, 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 запроса Prometheus, например, query_result(up).

Общие переменные

При определении настроек запросов можно комбинировать общие функции в переменные для быстрого определения 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="$namespace"}, deployment)`
daemonset	`label_values(kube_daemonset_status_number_ready{namespace="$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)`

Имя переменной	Функция запроса
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 предварительного просмотра: `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 предварительного просмотра: `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 предварительного просмотра 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=~"$namespace"}`.
   - **Regular Expression**: `/.+deployment="(.*?)",.*/`.
3. Добавьте панель в текущую панель и ссылайтесь на ранее добавленные переменные, например:
   - Имя метрики: **pod Memory Usage under Compute Components**.
   - Пара ключ-значение: `kind`: `Deployment`, `name`: `$deployment`, `namespace`: `$namespace`.
4. После добавления панелей и их сохранения вы сможете просматривать соответствующую информацию на главной странице панели.

Примечания при использовании встроенных метрик

> **WARNING**
>
> Следующие метрики используют пользовательские переменные `namespace`, `name` и `kind`, которые не поддерживают **множественный выбор** и 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

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 белого ящика (white-box), которые опираются на различные метрики мониторинга, уже доступные на платформе, мониторинг черного ящика (blackbox) фокусируется на результатах. Когда мониторинг белого ящика не может

охватить все факторы, влияющие на доступность сервиса, мониторинг черного ящика может быстро обнаружить сбои и выдать оповещения на основе этих сбоев. Например, если конечная точка API работает ненормально, мониторинг черного ящика 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.

Мониторинг Blackbox

Для создания элемента мониторинга blackbox вы можете 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: Мониторинг blackbox 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**: Проверяет доступность сервера путем ping доменного имени или IP-адреса, введенного в поле **Target Address**. **TCP**: Проверяет бизнес-порт хоста, слушая `<domain:port>` или `<IP:port>`, указанный в **Target Address**. **HTTP**: Проверяет URL, введенный 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 на основе элементов мониторинга blackbox создавать политики оповещений. При обнаружении сбоя автоматически сработает оповещение для уведомления ответственных лиц о необходимости устранения.

WARNING

После успешного создания элементов мониторинга blackbox 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 мониторинга blackbox должен быть успешно создан, и система должна завершить синхронизацию конфигурации, чтобы результаты проверки были видны на странице мониторинга blackbox.

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 был настроен элемент мониторинга blackbox.

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 мониторинга Blackbox**: выберите нужный элемент мониторинга blackbox.
  - **Имя метрики**: выберите метрику, по которой хотите мониторить и получать оповещения. В настоящее время платформа поддерживает метрики **Connectivity** и

HTTP Status Code.

- **Connectivity**: доступна для всех элементов мониторинга blackbox, условие срабатывания **“!= 1”** означает, что целевой адрес элемента мониторинга недоступен.
- **HTTP Status Code**: доступна, если метод проверки выбранного элемента мониторинга blackbox — **HTTP**. В качестве значения условия срабатывания можно ввести трехзначное положительное число, например, условие **“> 299”** означает, что оповещения срабатывают при кодах ответа 3XX, 4XX или 5XX.

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

1. Нажмите **Create**. После отправки политика оповещений появится в списке политик.

Настройка модуля мониторинга BlackboxExporter

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

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

TIP

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

Порядок действий

1. Обновите конфигурационный файл мониторинга blackbox, добавив кастомные модули мониторинга в ключ **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 проверке
```

   Полные примеры YAML конфигурации мониторинга blackbox смотрите в Справочной информации.

2. Активируйте конфигурацию одним из следующих способов.

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

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

Создание элементов мониторинга Blackbox 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 обновления элемента probe
    cpaas.io/display-name: 'Prometheus prober' # Описание элемента probe
  creationTimestamp: '2021-05-10T02:04:33Z' # Время создания элемента probe
  labels:
    prometheus: kube-prometheus # Значение метки, используемой для имени prometheus
  name: prometheus-liveness # Имя элемента probe
  namespace: cpaas-system # Namespace, используемый для namespace prometheus
spec:
  jobName: prometheus-liveness # Имя элемента probe
  prober:
    url: cpaas-monitor-prometheus-blackbox-exporter:9115 # URL для метрик 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 мониторинга blackbox
    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 группы 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 мониторинга blackbox и целевой адрес
          for: 30s # Продолжительность
          labels:
```

```
            alert_cluster: global # Имя кластера, в котором находится оповещение
            alert_for: 30s # Продолжительность
            alert_indicator: cluster.blackbox.probe.success # Оставьте без изменений
            alert_indicator_aggregate_range: '0' # Оставьте без изменений
            alert_indicator_blackbox_instance: https://demo.at-servicecenter.com/ #
Целевой адрес мониторинга blackbox
            alert_indicator_blackbox_name: test # Имя элемента мониторинга blackbox
            alert_indicator_comparison: '!=' # Оставьте без изменений для
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 изменений для
оповещений blackbox
            alert_involved_object_name: global # Кластер, в котором находится
элемент мониторинга blackbox
            alert_involved_object_namespace: '' # Namespace объекта, к которому
принадлежит правило оповещения
            alert_name: cluster.blackbox.probe.success-y97ah # Имя правила
оповещения
            alert_namespace: cpaas-system # Namespace, в котором находится правило
оповещения
            alert_project: cpaas-system # Имя проекта объекта, к которому
принадлежит правило оповещения
            alert_resource: policy # Имя политики оповещений, в которой
находится правило оповещения
            alert_source: Platform # Тип данных правила оповещения: Platform -
данные платформы, Business - бизнес-данные
            severity: High # Уровень правила оповещения: Critical - критический,
High - высокий, Medium - средний, Low - низкий
        - alert: cluster.blackbox.http.status.code-235el-
99b0095b6b6669415043e14ae84f43bc
          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 статусу, обязательно измените имя
элемента мониторинга blackbox 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-servicecenter.com/
            alert_indicator_blackbox_name: test
            alert_indicator_comparison: '>'
            alert_indicator_query: ''
            alert_indicator_threshold: '299' # Пороговое значение для правил
оповещений, в сценариях оповещений по HTTP статусу должно быть трехзначное
число, например, коды больше 299 (3XX, 4XX, 5XX) означают 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 мониторинга blackbox 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 диапазон, 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 запросов
            insecure_skip_verify: false
          preferred_ip_protocol: "ip4" # по умолчанию "ip6"                # Предпочтительная версия IP протокола
          ip_protocol_fallback: false  # Без fallback на "ip6"
      http_post_2xx:                 # Пример HTTP проверки с телом запроса
        prober: http
        timeout: 5s
        http:
          method: POST               # Метод запроса для проверки
          headers:
```

```
            Content-Type: application/json
          body: '{"username":"admin","password":"123456"}'                  # Тело,
передаваемое при проверке
      http_basic_auth_example:         # Пример проверки с именем пользователя и
паролем
        prober: http
        timeout: 5s
        http:
          method: POST
          headers:
            Host: "login.example.com"
          basic_auth:                  # Имя пользователя и пароль для проверки
            username: "username"
            password: "mysecret"
      http_custom_ca_example:
        prober: http
        http:
          method: GET
          tls_config:                  # Указать корневой сертификат для проверки
            ca_file: "/certs/my_cert.crt"
      http_gzip:
        prober: http
        http:
          method: GET
          compression: gzip            # Метод сжатия при проверке
      http_gzip_with_accept_encoding:
        prober: http
        http:
          method: GET
          compression: gzip
          headers:
            Accept-Encoding: gzip
      tls_connect:                     # Пример TCP проверки
        prober: tcp
        timeout: 5s
        tcp:
          tls: true                    # Использовать ли TLS
      tcp_connect_example:
        prober: tcp
        timeout: 5s
      imap_starttls:                   # Пример настройки проверки для IMAP почтовых
серверов
        prober: tcp
        timeout: 5s
```

```
        tcp:
          query_response:
            - expect: "OK.*STARTTLS"
            - send: ". STARTTLS"
            - expect: "OK"
            - starttls: true
            - send: ". capability"
            - expect: "CAPABILITY IMAP4rev1"
      smtp_starttls:                    # Пример настройки проверки для SMTP почтовых
серверов
        prober: tcp
        timeout: 5s
        tcp:
          query_response:
            - expect: "^220 ([^ ]+) ESMTP (.+)$"
            - send: "EHLO prober\r"
            - expect: "^250-STARTTLS"
            - send: "STARTTLS\r"
            - expect: "^220"
            - starttls: true
            - send: "EHLO prober\r"
            - expect: "^250-AUTH"
            - send: "QUIT\r"
      irc_banner_example:
        prober: tcp
        timeout: 5s
        tcp:
          query_response:
            - send: "NICK prober"
            - send: "USER prober prober prober :prober"
            - expect: "PING :([^ ]+)"
              send: "PONG ${1}"
            - expect: "^:[^ ]+ 001"
      icmp_example:                     # Пример конфигурации для ICMP проверки
        prober: icmp
        timeout: 5s
        icmp:
          preferred_ip_protocol: "ip4"
          source_ip_address: "127.0.0.1"
      dns_udp_example:                  # Пример DNS-запросов с использованием UDP
        prober: dns
        timeout: 5s
        dns:
          query_name: "www.prometheus.io"                # Доменное имя для
```

```
разрешения
          query_type: "A"              # Тип, соответствующий доменному имени
          valid_rcodes:
          - NOERROR
          validate_answer_rrs:
            fail_if_matches_regexp:
            - ".*127.0.0.1"
            fail_if_all_match_regexp:
            - ".*127.0.0.1"
            fail_if_not_matches_regexp:
            - "www.prometheus.io.\t300\tIN\tA\t127.0.0.1"
            fail_if_none_matches_regexp:
            - "127.0.0.1"
          validate_authority_rrs:
            fail_if_matches_regexp:
            - ".*127.0.0.1"
          validate_additional_rrs:
            fail_if_matches_regexp:
            - ".*127.0.0.1"
      dns_soa:
        prober: dns
        dns:
          query_name: "prometheus.io"
          query_type: "SOA"
      dns_tcp_example:               # Пример DNS-запросов с использованием TCP
        prober: dns
        dns:
          transport_protocol: "tcp" # по умолчанию "udp"
          preferred_ip_protocol: "ip4" # по умолчанию "ip6"
          query_name: "www.prometheus.io"
kind: ConfigMap
metadata:
  annotations:
    skip-sync: 'true'
  labels:
    app.kubernetes.io/instance: cpaas-monitor
    app.kubernetes.io/managed-by: Tiller
    app.kubernetes.io/name: prometheus-blackbox-exporter
    helm.sh/chart: prometheus-blackbox-exporter-1.6.0
  name: cpaas-monitor-prometheus-blackbox-exporter
  namespace: cpaas-system
```

Menu

Как сделать

Резервное копирование и восстановление данных мониторинга Prometheus

Обзор функции

Сценарии использования

Предварительные требования

Порядок выполнения операций

Результаты операции

Дополнительная информация

Следующие шаги

Резервное копирование и восстановление данных мониторинга VictoriaMetrics

Обзор функции

Сценарии использования

Предварительные условия

Процедуры

Результат операции

Дополнительная информация

Последующие действия

Сбор сетевых данных с сетевых интерфейсов с пользовательскими именами

Резервное копирование и восстановление данных мониторинга Prometheus

Содержание

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 данных
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 компонента вычислений — `prometheus-kube-prometheus-0`, тип — `StatefulSet`)
- Права администратора кластера
- Достаточно свободного места для хранения в целевом расположении

Порядок выполнения операций

Резервное копирование данных

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

копирования используют каталог хранения как источник 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 бэкенда 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 Pod будут в статусе Running, прежде чем продолжать.

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 находиться полный набор данных мониторинга в формате TSDB
- После 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 **persistentVolumeReclaimPolicy** в значение `Retain`

Следующие шаги

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

Резервное копирование и восстановление данных мониторинга VictoriaMetrics

Содержание

Обзор функции

Функция резервного копирования и восстановления данных мониторинга VictoriaMetrics позволяет регулярно создавать резервные копии данных мониторинга и восстанавливать данные при необходимости, обеспечивая безопасность и доступность данных мониторинга.

Сценарии использования

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

Предварительные условия

- В кластере установлен плагин ACP Monitoring с VictoriaMetrics
- Обеспечено достаточное место для хранения резервных копий
- Есть 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
- Отслеживать статус выполнения задач резервного копирования

Сбор сетевых данных с сетевых интерфейсов с пользовательскими именами

Содержание

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 применимо, когда на ваших узлах используются сетевые интерфейсы с пользовательскими именами (не соответствующими соглашениям об именах

`eth.|en.|wl.*|ww.*`) и требуется мониторинг сетевого трафика этих интерфейсов на платформе.

Предварительные требования

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

Порядок действий

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

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

   Пример вывода:

   ```
   global-6448ef7f7e5e3924c1629fad826372e7     global     prometheus          prometheus
   Running   v3.15.0-zz231204040711-9d1fc12474c2   v3.15.0-zz231204040711-9d1fc12474c2
   v3.15.0-zz231204040711-9d1fc12474c2
   ovn-0954f21f0359720e8c115804376b3e7e        ovn        prometheus          prometheus
   Running   v3.15.0-zz231204040711-9d1fc12474c2   v3.15.0-zz231204040711-9d1fc12474c2
   v3.15.0-zz231204040711-9d1fc12474c2
   ```

4. Отредактируйте ресурс moduleinfo кластера рабочих нагрузок:

```
kubectl edit moduleinfo <moduleinfo resource name of the workload cluster>
```

5. Добавьте или измените поле valuesOverride:

```
spec:
  valuesOverride:# Если этого поля нет, необходимо добавить поле valuesOverride
под spec вместе с параметрами ниже
    ait/chart-cpaas-monitor:
      global:
        indicator:
          networkDevice: eth.*|em.*|en.*|wl.*|ww.*|[A-Z].*i|custom_interface #
Замените custom_interface на пользовательское регулярное выражение 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 следующие данные сетевых интерфейсов с пользовательскими именами на странице мониторинга платформы:

- 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

Menu

Распределённое трассирование

Введение

Введение

Ограничения использования

Установка

Установка

Установка Jaeger Operator

Развёртывание экземпляра Jaeger

Установка OpenTelemetry Operator

Развёртывание экземпляров OpenTelemetry

Включение переключателя функций

Удаление Tracing

Архитектура

Архитектура

Основные компоненты

Поток данных

Основные понятия

Основные понятия

Телеметрия

OpenTelemetry

Span

Trace

Инструментирование

OpenTelemetry Collector

Jaeger

Руководства

Query Tracing

Обзор функции

Основные возможности

Преимущества функции

Tracing Query

Анализ результатов запроса

Query Trace Logs

Обзор функции

Основные возможности

Требования

Операции с запросами логов

Как сделать

Безвредная интеграция трассировки в Java-приложения

Обзор функции

Сценарии использования

Предварительные требования

Шаги по эксплуатации

Результаты эксплуатации

Бизнес-логи, связанные с TraceID

Предпосылки

Добавление TraceID в логи Java-приложения

Добавление TraceID в логи Python-приложения

Метод проверки

Устранение неполадок

Невозможно выполнить запрос требуемого трассирования

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 основных 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 с OpenTelemetry стандартами и open-source компонентами, 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

При использовании трассировки следует учитывать следующие ограничения:

- **Балансировка стратегий сэмплирования и производительности**
  - В условиях высокой нагрузки сбор данных трассировки может создавать определённую нагрузку 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 учётной 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

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 строку 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 имён `jaeger-operator`, чтобы оператор мог мониторить и быть доступным во всех пространствах имён кластера.
- Нажмите **Install**.

- Убедитесь, что в поле **Status** отображается **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               Succeeded
```

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

Развёртывание 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-pass-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 строку 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 имён `opentelemetry-operator`, чтобы оператор мог мониторить и быть доступным во всех пространствах имён кластера.
- Нажмите **Install**.
- Убедитесь, что в поле **Status** отображается **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   REPLACES   PHASE
openTelemetry-operator.vx.x.0   OpenTelemetry Operator   x.x.0
Succeeded
```

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

Развёртывание 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 в фазе **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-admin
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 строку поиска для поиска `Alauda build of OpenTelemetry`.
- Нажмите на заголовок **Alauda build of OpenTelemetry**, чтобы открыть детали.
- На странице деталей **Alauda build of OpenTelemetry** 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

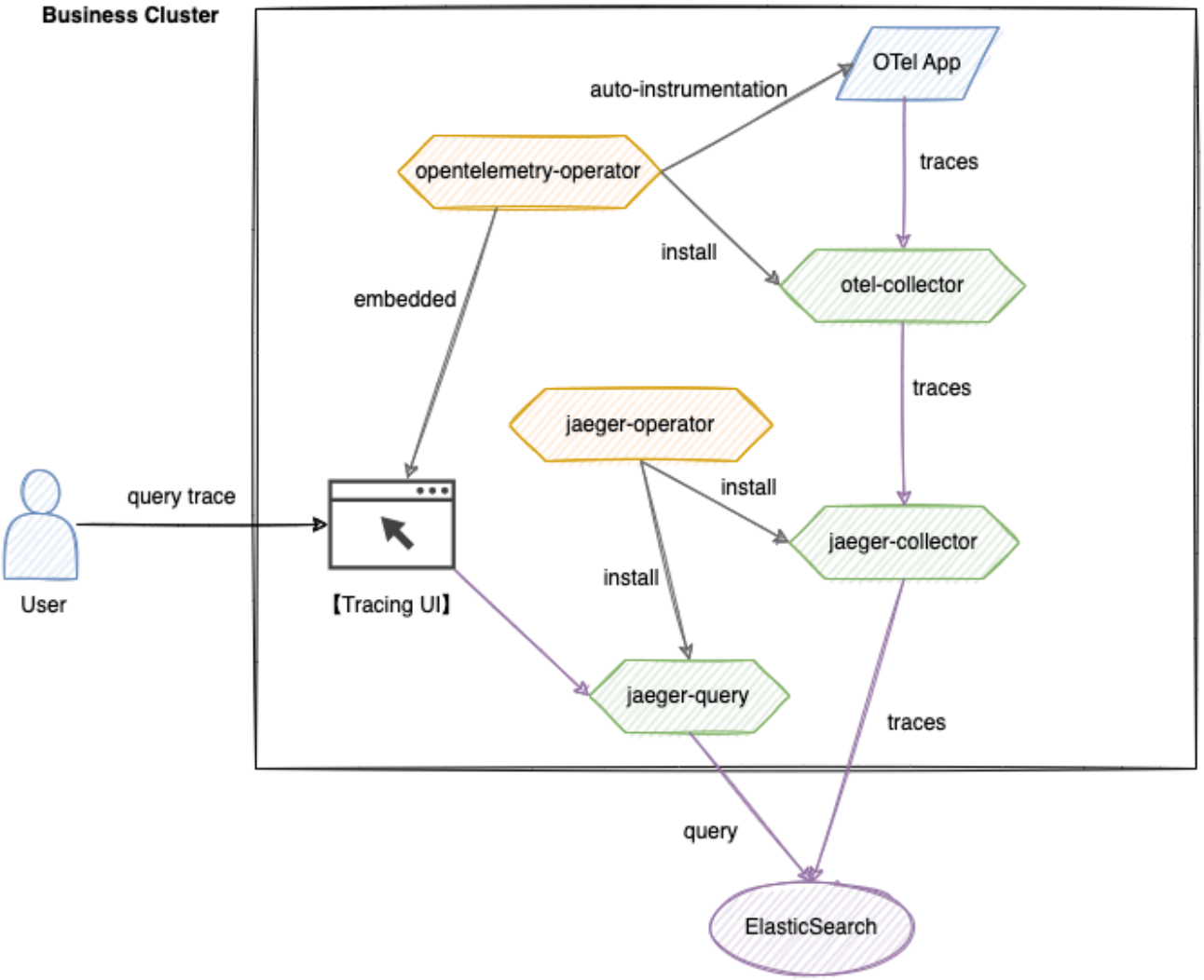
Вы можете удалить Jaeger Operator через режим **Administrator** в веб-консоли.

Шаги

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

Архитектура

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



Содержание

Поток данных

Основные компоненты

1. **Система OpenTelemetry**

   - **opentelemetry-operator**
     Оператор на уровне кластера, отвечающий за развертывание и управление компонентом otel-collector, обеспечивающий возможность автоматической инъекции 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, переданные и обработанные otel-collector, выполняет преобразование формата и записывает их в Elasticsearch.
   - **jaeger-query**
     Предоставляет API для запросов трассировки, поддерживающий многокритериальный поиск, включая TraceID и метки.

3. **Слой хранения**

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

Поток данных

- **Процесс записи**

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

  Приложение генерирует данные Span через SDK или автоматическую инъекцию, 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

Телеметрия

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

OpenTelemetry

OpenTelemetry — это фреймворк↗ и набор инструментов для создания и управления телеметрическими данными, такими как трассы↗, метрики↗ и логи↗. Важно, что OpenTelemetry не зависит от конкретного 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 (root span) 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.

Инструментирование

Для обеспечения наблюдаемости система должна пройти процесс **инструментирования**: то есть код компонентов системы должен генерировать трассы↗ , метрики↗ и логи↗ .

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

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

Решения на основе кода обеспечивают более глубокое понимание и более богатые телеметрические данные изнутри вашего приложения. Вы можете генерировать телеметрию в приложении с помощью OpenTelemetry API, что является важным дополнением к телеметрии, создаваемой решениями без инструментирования кода.

Решения без инструментирования кода 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 телеметрии в различных форматах (например, OTLP, Jaeger, Prometheus и многих коммерческих/проприетарных инструментах) и отправку этих данных в один или несколько бекендов. Также поддерживается обработка и фильтрация телеметрии перед экспортом.

Подробнее см. 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 несколько языков программирования и хранилищ данных, и является ключевым инструментом наблюдаемости в облачно-нативной среде.

Menu

Руководства

Query Tracing

Обзор функции

Основные возможности

Преимущества функции

Tracing Query

Анализ результатов запроса

Query Trace Logs

Обзор функции

Основные возможности

Требования

Операции с запросами логов

Query Tracing

Содержание

Обзор функции

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

- **Трассировка цепочки запросов**: Восстановление полного пути запроса в сложных распределённых системах.
- **Анализ узких мест производительности**: Выявление аномальных узлов вызова по времени выполнения в цепочке.

- **Определение корня неисправности**: Быстрая локализация точки возникновения ошибки с помощью маркировки ошибок.

Применимые сценарии включают:

- Быстрая локализация 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.

Основные возможности

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

Преимущества функции

- **Быстрая идентификация проблем**: сужение области поиска за счёт многомерных условий ускоряет локализацию неисправностей.
- **Визуальное представление**: использование временных водопадных диаграмм для наглядного отображения связей вызовов снижает сложность и повышает

эффективность анализа ошибок.

- **Гибкость и разнообразие**: поддержка как простых запросов, так и сложных комбинаций, адаптированных под различные сценарии эксплуатации и разработки.

Tracing Query

1 Шаг 1: Комбинирование условий запроса

Совет: условия запроса можно комбинировать. Вы можете уточнить поиск, добавляя несколько условий.

Условие запроса	Описание
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 по введённым меткам (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.

Условие запроса	Описание
Maximum Query Count	Максимальное количество Span, которые можно запросить, по умолчанию 200. **Совет**: для производительности платформа отображает не более 1000 Span за один раз. Если количество Span, удовлетворяющих условиям, превышает **максимальное количество запросов**, можно уточнить условия или сузить временной диапазон для поэтапного запроса.

2 Шаг 2: Выполнение запроса

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

Анализ результатов запроса

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

Список Span

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

Временная водопадная диаграмма

Справа в области результатов временная водопадная диаграмма наглядно показывает связи вызовов между Span в одном трассировании. Основные особенности использования временных водопадных диаграмм при анализе трассировки:

1. Раскрытие сверху вниз: в диаграмме вызовы (Span) обычно разворачиваются 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 подробные данные: клик по полосам диаграммы отображает более детальную информацию о вызове.

Детали Span

Кликнув по строке Span на временной водопадной диаграмме, можно развернуть и просмотреть подробную информацию о Span, включая:

- Service: сервис, к которому относится Span.
- Span Duration (ms): длительность Span.
- URL: URL, к которому обращался сервис, соответствует **http.url** в тегах Span.
- Tag: информация о метках Span в виде пар ключ-значение, используется для расширенного поиска по тегам. Кнопка рядом с меткой позволяет добавить текущее условие Tag в условия запроса для более точного поиска.
- JSON: исходная JSON-структура Span для более глубокого изучения внутренней информации.

Query Trace Logs

Содержание

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 данных в настраиваемых форматах.
- **Контекстный просмотр логов**: изучение записей логов до и после целевой записи для более глубокого анализа.

Требования

TIP

Перед выполнением запросов логов по TraceID необходимо сначала настроить ваши сервисы на включение TraceID в бизнес-логи. Следуйте руководству Business Log Correlation with TraceID Guide для деталей настройки.

Поведение по умолчанию:

- Отображает логи за весь период трейсинга.
- Для трейсинга короче 1 минуты запрашивает логи 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 его 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

Установите флажок **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 лога

1. Нажмите **Insight** рядом с любой записью лога.

2. Изучите 5 предыдущих и 5 последующих логов относительно целевой временной метки.

3. Прокручивайте вверх/вниз мышью для загрузки дополнительных логов.

Menu

Как сделать

Безвредная интеграция трассировки в Java-приложения

Обзор функции

Сценарии использования

Предварительные требования

Шаги по эксплуатации

Результаты эксплуатации

Бизнес-логи, связанные с TraceID

Предпосылки

Добавление TraceID в логи Java-приложения

Добавление TraceID в логи Python-приложения

Метод проверки

Безвредная интеграция трассировки в Java-приложения

INFO

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

Содержание

Обзор функции

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

Сценарии использования

Java-приложения могут быть интегрированы для следующих сценариев:

- 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 Alauda Container Platform
- Сервис использует JDK версии Java 8 или выше
- У вас есть права на редактирование Deployment в целевом namespace
- Платформа завершила развёртывание трассировки

Шаги по эксплуатации

Для Java-приложения, которое необходимо интегрировать в трассировку Alauda Container Platform, требуются следующие адаптации:

- 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-common-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 `cpaas-system/acp-common-java` Instrumentation в качестве конфигурации 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 — `metadata.namespace`.

Результаты эксплуатации

После адаптации Java-приложения:

- Если в новом запущенном pod Java-приложения присутствует init-контейнер `opentelemetry-auto-instrumentation-java`, это означает, что внедрение прошло успешно.
- Отправьте тестовые запросы в Java-приложение.
- В представлении **Container Platform** выберите **проект**, **кластер** и **namespace**, в котором находится Java-приложение.
- Перейдите на страницу **Observability** -> **Tracing**, чтобы просмотреть данные трассировки и диаграмму водопада временной шкалы Java-приложения.

TIP

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

Бизнес-логи, связанные с TraceID

Содержание

Предпосылки

- Для корректного объединения 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 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 файл конфигурации logback следующим образом.

TIP

В качестве примера используется вывод в консоль, где `%X{trace_id}` обозначает значение ключа `trace_id`, полученное из MDC.

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

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 трейсе из заголовков запроса, где значение `x-b3-traceid` является TraceID.

```
    def getForwardHeaders(request):
        headers = {}
        incoming_headers = [
            'x-request-id',  # Все приложения должны передавать x-request-id для
access log и согласованных решений по трассировке/выборке логов
            '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 отображаемых 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.

Menu

Устранение неполадок

Невозможно выполнить запрос требуемого трассирования

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 требуемого трассирования

Содержание

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 задан параметр `"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` у `jaeger-collector`, значение по умолчанию — `10s`.

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

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

Неполные данные трассировки

Содержание

Описание проблемы

Результаты запросов трассировки демонстрируют следующие проблемы с неполными данными:

- В последних 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 трассировка длится более 1 часа, 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 спан трассировки в вашей среде длительный, 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.

Menu

Логи

Введение

Введение

Установка

Установка

Планирование установки

Установка Alauda Container Platform Log Storage с ElasticSearch через консоль

Установка Alauda Container Platform Log Storage с ElasticSearch через YAML

Установка Alauda Container Platform Log Storage с Clickhouse через консоль

Установка Alauda Container Platform Log Storage с Clickhouse через YAML

Установка плагина Alauda Container Platform Log Collector

Установка плагина Alauda Container Platform Log Collector через YAML

Архитектура

Архитектура модуля логирования

Общее описание архитектуры

Сбор логов

Потребление и хранение логов

Визуализация логов

Руководство по выбору компонента логирования

Сравнение архитектур

Сравнение функций

Рекомендации по выбору

Планирование ёмкости компонента логирования

ElasticSearch

Clickhouse

Основные понятия

Основные понятия

Open Source Components

Основные понятия функциональности

Ключевые технические термины

Модель потока данных

Руководства

Логи

Анализ запросов логов

Управление временем хранения логов приложений

Настройка частичного исключения логов приложений из сбора

Как сделать

Как архивировать логи в стороннее хранилище

Передача во внешнее NFS

Передача во внешнее S3-хранилище

Как взаимодействовать с внешними кластерами ES Storage

Подготовка ресурсов

Порядок действий

Menu

Введение

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

Этот модуль предоставляет четыре основные функции логирования:

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

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 соблюдение требований и получать ценные операционные данные в режиме реального времени.

Установка

Система логирования платформы состоит из двух плагинов: Alauda Container Platform Log Collector и Alauda Container Platform Log Storage. В этой главе будет представлено руководство по установке этих двух плагинов.

WARNING

1. Кластер `global` может выполнять запросы к данным логов, хранящимся на любом workload кластере внутри платформы. Убедитесь, что кластер `global` имеет доступ к порту 11780 workload кластера.
2. Плагин Alauda Container Platform Log Storage с Clickhouse требует наличия Clickhouse operator. Перед установкой плагина убедитесь, что Clickhouse operator загружен в кластер.

Содержание

Планирование установки

Плагины Alauda Container Platform Log Storage могут быть установлены в любом кластере, и для сбора логов можно выбрать компонент хранения логов любого кластера для взаимодействия с данными хранилища.

Поэтому перед установкой плагина хранения логов необходимо спланировать кластер и узлы, на которых будет установлен компонент хранения логов.

- Избегайте развертывания плагинов хранения логов в глобальном кластере. Вместо этого развертывайте их в workload кластерах, чтобы сбои в управляющем кластере не нарушали процесс решения проблем на основе логов.
- Старайтесь централизовать логи в одном кластере хранения логов. Если объем логов превышает максимальные пороговые значения, распределите логи по нескольким кластерам хранения.
- Разверните как минимум один экземпляр хранения логов в каждой сетевой зоне для локальной агрегации логов, минимизируя трафик по публичной сети между дата-центрами (что ведет к высоким затратам и задержкам).
- Выделите отдельные узлы для хранения логов, избегая совместного развертывания с другими приложениями или компонентами платформы. Хранение логов требует высокой пропускной способности ввода-вывода и может страдать от помех.
- Подключайте выделенные SSD-диски для хранения логов, что значительно повысит производительность.

Установка Alauda Container Platform Log Storage с ElasticSearch через консоль

1. Перейдите в **App Store Management** > **Cluster Plugin** и выберите целевой кластер.
2. На вкладке **Plugins** нажмите кнопку действий справа от **Alauda Container Platform Log Storage with ElasticSearch** > **Install**.
3. Следуйте инструкциям для настройки соответствующих параметров.

Параметр	Описание
Connect External Elasticsearch	Оставьте закрытым для установки плагина хранения логов внутри платформы.
Component installation Settings	**LocalVolume**: Локальное хранилище, данные логов будут храниться в локальном пути хранения выбранного узла. Преимущество этого метода — компонент логирования напрямую привязан к локальному хранилищу, что исключает необходимость доступа к хранилищу по сети и обеспечивает лучшую производительность. **StorageClass**: Динамическое создание ресурсов хранения с помощью классов хранения для сохранения данных логов. Преимущество этого метода — большая гибкость; при наличии нескольких классов хранения для всего кластера администраторы могут выбирать соответствующий класс хранения для компонентов логирования в зависимости от сценариев использования, снижая влияние сбоев хоста на хранение. Однако производительность StorageClass может зависеть от пропускной способности сети и задержек, а также опирается на механизмы избыточности, предоставляемые бекендом хранения, для обеспечения высокой доступности.
Retention Period	Максимальное время хранения логов, событий и аудита в кластере. Данные, превышающие период хранения, будут автоматически удалены.

Параметр	Описание
	Совет: Вы можете создавать резервные копии данных, которые необходимо хранить длительное время. При необходимости обратитесь в техническую поддержку.

4. Нажмите **Install**.

Установка Alauda Container Platform Log Storage с ElasticSearch через YAML

1. Проверка доступных версий

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

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

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

2. Создание ModuleInfo

Создайте ресурс ModuleInfo для установки плагина без параметров конфигурации:

```
apiVersion: cluster.alauda.io/v1alpha1
kind: ModuleInfo
metadata:
  annotations:
    cpaas.io/display-name: logcenter
    cpaas.io/module-name: '{"en": "Alauda Container Platform Log Storage for
Elasticsearch", "zh": "Alauda Container Platform Log Storage for Elasticsearch"}'
  labels:
    cpaas.io/cluster-name: go
    cpaas.io/module-name: logcenter
    cpaas.io/module-type: plugin
    cpaas.io/product: Platform-Center
  name: <cluster>-log-center
spec:
  config:
    clusterView:
      isPrivate: "true"
    components:
      elasticsearch:
        address: ""
        basicAuthSecretName: ""
        hostpath: /cpaas/data/elasticsearch
        httpPort: 9200
        install: true
        k8sNodes:
        - 192.168.139.75
        masterK8sNodes: []
        masterReplicas: 0
        masterResources:
          limits:
            cpu: "2"
            memory: 4Gi
          requests:
            cpu: 200m
            memory: 256Mi
        masterStorageSize: 5
        nodeReplicas: 1
        nodeStorageSize: 200
        resources:
          limits:
            cpu: "4"
            memory: 4Gi
          requests:
```

```
            cpu: "1"
            memory: 1Gi
          tcpPort: 9300
          type: single
        kafka:
          address: ""
          auth: true
          basicAuthSecretName: ""
          exporterPort: 9308
          install: true
          k8sNodes:
          - 192.168.139.75
          port: 9092
          storageSize: 10
          tls: true
          zkElectPort: 3888
          zkExporterPort: 9141
          zkLeaderPort: 2888
          zkPort: 2181
        kibana:
          install: false
        storageClassConfig:
          name: elasticsearch-local-log-sc
          type: LocalVolume
        zookeeper:
          storageSize: 1
      ttl:
        audit: 180
        event: 180
        logKubernetes: 7
        logPlatform: 7
        logSystem: 7
        logWorkload: 7
    version: v4.1.0
```

Справочник по полям YAML:

Путь поля	Описание
`metadata.labels.cpaas.io/cluster-name`	Имя целевого кластера, в котором устанавливается плагин.

Путь поля	Описание
`metadata.name`	Временное имя ModuleInfo; платформа переименует его после создания.
`spec.config.clusterView.isPrivate`	Настройка видимости просмотра кластера.
`spec.config.components.elasticsearch.address`	Адрес внешнего Elasticsearch; оставьте пустым для использования Elasticsearch, установленного платформой.
`spec.config.components.elasticsearch.basicAuthSecretName`	Имя секрета для базовой аутентификации внешнего Elasticsearch; оставьте пустым для платформенного Elasticsearch.
`spec.config.components.elasticsearch.hostpath`	Путь данных для Elasticsearch.
`spec.config.components.elasticsearch.httpPort`	HTTP-порт Elasticsearch, по умолчанию 9200.
`spec.config.components.elasticsearch.install`	Устанавливать ли Elasticsearch через платформу; установите в false при использовании внешнего Elasticsearch.
`spec.config.components.elasticsearch.k8sNodes`	Список IP-адресов узлов для данных Elasticsearch

Путь поля	Описание
	при использовании LocalVolume.
`spec.config.components.elasticsearch.masterK8sNodes`	Список IP-адресов узлов для Elasticsearch Master (только для крупномасштабных установок с LocalVolume).
`spec.config.components.elasticsearch.masterReplicas`	Количество реплик Elasticsearch Master (только для крупномасштабных установок).
`spec.config.components.elasticsearch.masterResources`	Запросы и лимиты ресурсов для Elasticsearch Master (только для крупномасштабных установок).
`spec.config.components.elasticsearch.masterStorageSize`	Размер хранилища для Elasticsearch Master (только для крупномасштабных установок).
`spec.config.components.elasticsearch.nodeReplicas`	Количество реплик для данных Elasticsearch.
`spec.config.components.elasticsearch.nodeStorageSize`	Размер хранилища для данных Elasticsearch (Gi).
`spec.config.components.elasticsearch.resources`	Запросы и лимиты ресурсов для данных Elasticsearch.

Путь поля	Описание
`spec.config.components.elasticsearch.tcpPort`	Внутренний транспортный порт для кластера Elasticsearch, по умолчанию 9300.
`spec.config.components.elasticsearch.type`	Размер кластера Elasticsearch: single/normal/big.
`spec.config.components.kafka.address`	Адрес внешнего Kafka; оставьте пустым для использования Kafka, установленного платформой.
`spec.config.components.kafka.auth`	Включить аутентификацию Kafka, по умолчанию true.
`spec.config.components.kafka.basicAuthSecretName`	Имя секрета для аутентификации внешнего Kafka; оставьте пустым для платформенного Kafka.
`spec.config.components.kafka.exporterPort`	Порт Kafka Exporter, по умолчанию 9308.
`spec.config.components.kafka.install`	Устанавливать ли Kafka через платформу; установите в false при использовании внешнего Kafka.

Путь поля	Описание
`spec.config.components.kafka.k8sNodes`	Список IP-адресов узлов для Kafka при использовании LocalVolume.
`spec.config.components.kafka.port`	Порт Kafka, по умолчанию 9092.
`spec.config.components.kafka.storageSize`	Размер хранилища Kafka (Gi).
`spec.config.components.kafka.tls`	Включить TLS для Kafka, по умолчанию true.
`spec.config.components.kafka.zkElectPort`	Порт выборов Zookeeper, по умолчанию 3888.
`spec.config.components.kafka.zkExporterPort`	Порт Zookeeper Exporter, по умолчанию 9141.
`spec.config.components.kafka.zkLeaderPort`	Порт связи лидера/фолловера Zookeeper, по умолчанию 2888.
`spec.config.components.kafka.zkPort`	Клиентский порт Zookeeper, по умолчанию 2181.
`spec.config.components.kibana.install`	Устанавливать Kibana; Kibana устарела, установите false.
`spec.config.components.storageClassConfig.name`	Для LocalVolume обычно `elasticsearch-local-log-sc`; для StorageClass укажите имя класса.
`spec.config.components.storageClassConfig.type`	Тип хранения: LocalVolume/StorageClass.

Путь поля	Описание
`spec.config.components.zookeeper.storageSize`	Размер хранилища Zookeeper (Gi).
`spec.config.ttl.audit`	Количество дней хранения аудита.
`spec.config.ttl.event`	Количество дней хранения событий.
`spec.config.ttl.logKubernetes`	Количество дней хранения логов Kubernetes.
`spec.config.ttl.logPlatform`	Количество дней хранения логов платформы.
`spec.config.ttl.logSystem`	Количество дней хранения системных логов.
`spec.config.ttl.logWorkload`	Количество дней хранения логов workload.
`spec.version`	Указывает версию плагина для установки, должна совпадать с `.spec.version` в ModuleConfig.

3. Проверка установки

Поскольку имя ModuleInfo меняется после создания, найдите ресурс по метке для проверки статуса и версии плагина:

```
kubectl get moduleinfo -l cpaas.io/module-name=logcenter
NAME                                          CLUSTER         MODULE      DISPLAY_NAME
STATUS    TARGET_VERSION   CURRENT_VERSION   NEW_VERSION
global-e671599464a5b1717732c5ba36079795          global          logcenter   logcenter
Running   v4.0.12          v4.0.12           v4.0.12
```

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` : целевая версия для установки
- `CURRENT_VERSION` : версия до установки
- `NEW_VERSION` : последняя доступная версия для установки

Установка Alauda Container Platform Log Storage с Clickhouse через консоль

1. Перейдите в **App Store Management** > **Cluster Plugin** и выберите целевой кластер.
2. На вкладке **Plugins** нажмите кнопку действий справа от **Alauda Container Platform Log Storage with Clickhouse** > **Install**.
3. Следуйте инструкциям для настройки соответствующих параметров.

Параметр	Описание
Component installation Settings	**LocalVolume**: Локальное хранилище, данные логов будут храниться в локальном пути хранения выбранного узла. Преимущество этого метода — компонент логирования напрямую привязан к локальному хранилищу, что исключает

Параметр	Описание
	необходимость доступа к хранилищу по сети и обеспечивает лучшую производительность. **StorageClass**: Динамическое создание ресурсов хранения с помощью классов хранения для сохранения данных логов. Преимущество этого метода — большая гибкость; при наличии нескольких классов хранения для всего кластера администраторы могут выбирать соответствующий класс хранения для компонентов логирования в зависимости от сценариев использования, снижая влияние сбоев хоста на хранение. Однако производительность StorageClass может зависеть от пропускной способности сети и задержек, а также опирается на механизмы избыточности, предоставляемые бекендом хранения, для обеспечения высокой доступности.
Retention Period	Максимальное время хранения логов, событий и аудита в кластере. Данные, превышающие период хранения, будут автоматически удалены. **Совет**: Вы можете создавать резервные копии данных, которые необходимо хранить длительное время. При необходимости обратитесь в техническую поддержку.

4. Нажмите **Install**.

Установка Alauda Container Platform Log Storage с Clickhouse через YAML

1. Проверка доступных версий

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

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

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

2. Создание ModuleInfo

Создайте ресурс ModuleInfo для установки плагина без параметров конфигурации:

```yaml
apiVersion: cluster.alauda.io/v1alpha1
kind: ModuleInfo
metadata:
  name: global-logclickhouse
  labels:
    cpaas.io/cluster-name: global
    cpaas.io/module-name: logclickhouse
    cpaas.io/module-type: plugin
spec:
  version: v4.1.0
  config:
    components:
      storageClassConfig:
        type: LocalVolume
        name: ""
      clickhouse:
        resources:
          limits:
            cpu: "2"
            memory: 4Gi
          requests:
            cpu: 200m
            memory: 256Mi
        k8sNodes:
          - xxx.xxx.xxx.xx
        hostpath: /cpaas/data/clickhouse
        nodeReplicas: 1
        nodeStorageSize: 200
        type: single
      razor:
        resources:
          limits:
            cpu: "2"
            memory: 1Gi
          requests:
            cpu: 10m
            memory: 256Mi
      vector:
        resources:
          limits:
            cpu: "4"
            memory: 1Gi
          requests:
```

```
            cpu: 10m
            memory: 256Mi
    ttl:
      audit: 180
      event: 180
      logKubernetes: 7
      logPlatform: 7
      logSystem: 7
      logWorkload: 7
```

Справочник по полям YAML (ClickHouse):

Путь поля	Описание
`metadata.name`	Имя ModuleInfo. Рекомендуемый формат: `<target-cluster>-logclickhouse`.
`metadata.labels.cpaas.io/cluster-name`	Целевой кластер, в котором устанавливается плагин.
`metadata.labels.cpaas.io/module-name`	Должно быть `logclickhouse`.
`metadata.labels.cpaas.io/module-type`	Должно быть `plugin`.
`spec.version`	Версия плагина для установки.
`spec.config.components.storageClassConfig.type`	Тип хранения данных ClickHouse: `LocalVolume` или `StorageClass`.
`spec.config.components.storageClassConfig.name`	Имя StorageClass при типе `StorageClass`; оставьте пустым для `LocalVolume`.
`spec.config.components.clickhouse.resources`	Запросы и лимиты ресурсов для ClickHouse.
`spec.config.components.clickhouse.k8sNodes`	Список IP-адресов узлов для ClickHouse при использовании `LocalVolume`.

Путь поля	Описание
`spec.config.components.clickhouse.hostpath`	Локальный путь для данных ClickHouse при использовании `LocalVolume`.
`spec.config.components.clickhouse.nodeReplicas`	Количество реплик при использовании `StorageClass`.
`spec.config.components.clickhouse.nodeStorageSize`	Размер хранилища для данных ClickHouse (Gi).
`spec.config.components.clickhouse.type`	Размер кластера: `single`, `normal` или `big`.
`spec.config.components.razor.resources`	Запросы и лимиты ресурсов для Razor.
`spec.config.components.vector.resources`	Запросы и лимиты ресурсов для Vector.
`spec.config.ttl.audit`	Количество дней хранения аудита.
`spec.config.ttl.event`	Количество дней хранения событий.
`spec.config.ttl.logKubernetes`	Количество дней хранения логов Kubernetes.
`spec.config.ttl.logPlatform`	Количество дней хранения логов платформы.
`spec.config.ttl.logSystem`	Количество дней хранения системных логов.
`spec.config.ttl.logWorkload`	Количество дней хранения логов workload.

Путь поля	Описание
`spec.version`	Указывает версию плагина для установки, должна совпадать с `.spec.version` в ModuleConfig.

3. Проверка установки

Поскольку имя ModuleInfo меняется после создания, найдите ресурс по метке для проверки статуса и версии плагина:

```
kubectl get moduleinfo -l cpaas.io/module-name=logclickhouse
NAME                                          CLUSTER         MODULE      DISPLAY_NAME
STATUS    TARGET_VERSION   CURRENT_VERSION   NEW_VERSION
global-e671599464a5b1717732c5ba36079795          global          logclickhouse
logclickhouse      Running   v4.0.12          v4.0.12          v4.0.12
```

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` : целевая версия для установки
- `CURRENT_VERSION` : версия до установки
- `NEW_VERSION` : последняя доступная версия для установки

Установка плагина Alauda Container Platform Log Collector

1. Перейдите в **App Store Management** > **Cluster Plugin** и выберите целевой кластер.

2. На вкладке **Plugins** нажмите кнопку действий справа от **Alauda Container Platform Log Collector** > **Install**.
3. Выберите **Storage Cluster** (кластер, в котором установлен Alauda Container Platform Log Storage) и нажмите ***Select/Deselect*** для выбора типов логов, которые будут собираться в кластере.
4. Нажмите **Install**.

Установка плагина Alauda Container Platform Log Collector через YAML

1. Проверка доступных версий

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

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

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

2. Создание ModuleInfo

Создайте ресурс ModuleInfo для установки плагина без параметров конфигурации:

```yaml
apiVersion: cluster.alauda.io/v1alpha1
kind: ModuleInfo
metadata:
  annotations:
    cpaas.io/display-name: logagent
    cpaas.io/module-name: '{"en": "Alauda Container Platform Log Collector", "zh":
"Alauda Container Platform Log Collector"}'
  labels:
    cpaas.io/cluster-name: go
    cpaas.io/module-name: logagent
    cpaas.io/module-type: plugin
    cpaas.io/product: Platform-Center
    logcenter.plugins.cpaas.io/cluster: go
  name: <cluster>-log-agent
spec:
  config:
    crossClusterDependency:
      logcenter: go
      logclickhouse: null
    dataSource:
      audit: true
      event: true
      kubernetes: true
      platform: false
      system: false
      workload: true
    storage:
      type: Elasticsearch
  version: v4.1.0
```

Справочник по полям YAML (Log Collector):

Путь поля	Описание
`metadata.annotations.cpaas.io/display-name`	Отображаемое имя плагина.
`metadata.annotations.cpaas.io/module-name`	JSON-строка с интернационализированным именем плагина.

Путь поля	Описание
`metadata.labels.cpaas.io/cluster-name`	Целевой кластер, в котором устанавливается плагин.
`metadata.labels.cpaas.io/module-name`	Должно быть `logagent`.
`metadata.labels.cpaas.io/module-type`	Должно быть `plugin`.
`metadata.labels.cpaas.io/product`	Идентификатор продукта, обычно `Platform-Center`.
`metadata.labels.logcenter.plugins.cpaas.io/cluster`	Имя кластера хранения, в который отправляются логи.
`metadata.name`	Временное имя ModuleInfo; платформа переименует его после создания.
`spec.config.crossClusterDependency.logcenter`	Имя кластера хранения логов на базе Elasticsearch.
`spec.config.crossClusterDependency.logclickhouse`	Установите `null` при использовании Elasticsearch; иначе имя кластера ClickHouse.
`spec.config.dataSource.audit`	Собирать логи аудита.
`spec.config.dataSource.event`	Собирать логи событий.
`spec.config.dataSource.kubernetes`	Собирать логи Kubernetes.
`spec.config.dataSource.platform`	Собирать логи платформы.
`spec.config.dataSource.system`	Собирать системные логи.
`spec.config.dataSource.workload`	Собирать логи workload.
`spec.config.storage.type`	`Elasticsearch` или `Clickhouse`.
`spec.version`	Версия плагина для установки.

3. Проверка установки

Поскольку имя ModuleInfo меняется после создания, найдите ресурс по метке для проверки статуса и версии плагина:

```
kubectl get moduleinfo -l cpaas.io/module-name=logagent
NAME                                          CLUSTER         MODULE      DISPLAY_NAME
STATUS    TARGET_VERSION   CURRENT_VERSION   NEW_VERSION
global-e671599464a5b1717732c5ba36079795          global          logagent    logagent
Running   v4.0.12          v4.0.12           v4.0.12
```

Menu

Архитектура

Архитектура модуля логирования

Общее описание архитектуры

Сбор логов

Потребление и хранение логов

Визуализация логов

Руководство по выбору компонента логирования

Сравнение архитектур

Сравнение функций

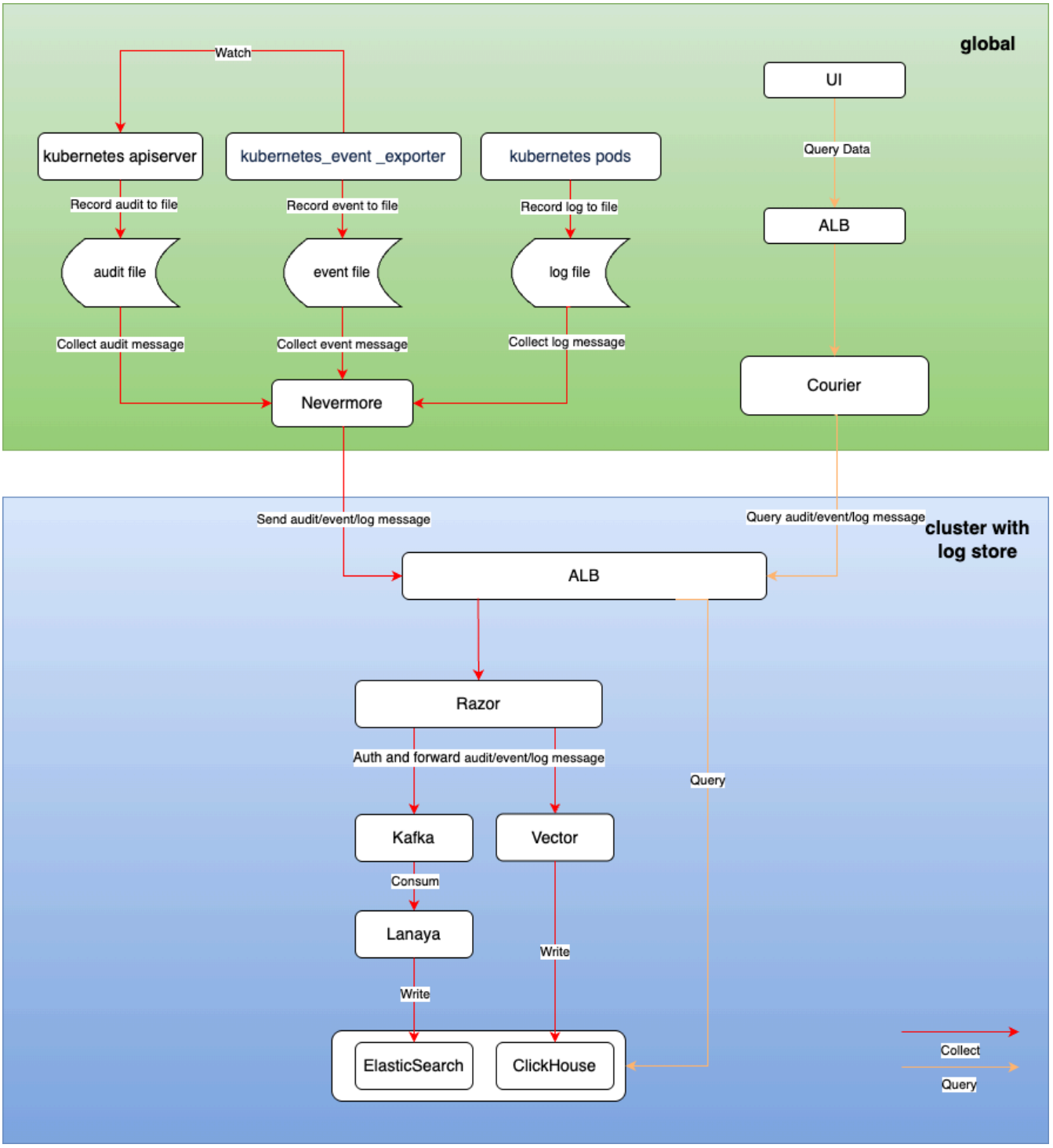
Рекомендации по выбору

Планирование ёмкости компонента логирования

ElasticSearch

Clickhouse

Архитектура модуля логирования



Содержание

Общее описание архитектуры

Система логирования состоит из следующих основных функциональных модулей:

1. Сбор логов

   - Основан на open-source компоненте filebeat
   - Сбор логов: поддерживается сбор логов стандартного вывода, файловых логов, событий Kubernetes и аудитов.

2. Хранение логов

   - Предоставляются два различных решения для хранения логов на базе open-source компонентов Clickhouse и ElasticSearch.
   - Хранение логов: поддерживается долговременное хранение лог-файлов.
   - Управление временем хранения логов: поддерживается управление сроком хранения логов на уровне проекта.

3. Визуализация логов

   - Обеспечивает удобный и надежный поиск логов, экспорт логов и возможности анализа логов.

Сбор логов

Метод установки компонента

nevermore устанавливается как daemonset в namespace cpaas-system каждого кластера. Этот компонент состоит из 4 контейнеров:

Название	Функция
audit	Сбор данных аудита
event	Сбор данных событий
log	Сбор логов (включая стандартный вывод и файловые логи)
node-problem-detector	Сбор информации об аномалиях на узлах

Процесс сбора данных

После того как nevermore собирает информацию аудита/событий/логов, он отправляет данные в кластер хранения логов, проходя аутентификацию через Razor, после чего данные окончательно сохраняются в ElasticSearch или ClickHouse.

Потребление и хранение логов

Razor

Razor отвечает за аутентификацию, прием и пересылку сообщений логов.

- После получения запросов от nevermore из различных рабочих кластеров Razor сначала аутентифицирует их с помощью Token, содержащегося в запросе. При неудачной аутентификации запрос отклоняется.

- Если установлен компонент хранения логов ElasticSearch, соответствующие логи записываются в Kafka кластер.
- Если установлен компонент хранения логов Clickhouse, соответствующие логи передаются в Vector, который в конечном итоге записывает их в Clickhouse.

Lanaya

Lanaya отвечает за потребление и пересылку данных логов в цепочке хранения логов ElasticSearch.

- Lanaya подписывается на темы в Kafka. После получения сообщений по подписке она распаковывает сообщения.
- После распаковки выполняется предварительная обработка сообщений: добавление необходимых полей, преобразование полей и разделение данных.
- В конце сообщения сохраняются в соответствующем индексе ElasticSearch в зависимости от времени и типа сообщения.

Vector

Vector отвечает за обработку и пересылку данных логов в цепочке хранения логов Clickhouse, в конечном итоге сохраняя логи в соответствующей таблице Clickhouse.

Визуализация логов

1. Пользователи могут выполнять запросы аудита/событий/логов через URL-адреса из UI интерфейса продукта для отображения:

- Запрос логов /platform/logging.alauda.io/v1
- Запрос событий /platform/events.alauda.io/v1
- Запрос аудита /platform/audits.alauda.io/v1

1. Запросы обрабатываются компонентом advanced API Courier, который выполняет запросы к данным логов из кластеров хранения ElasticSearch или Clickhouse и возвращает результаты на страницу.

Руководство по выбору компонента логирования

При установке компонента логирования платформа предоставляет два компонента для хранения логов на выбор: ElasticSearch и Clickhouse. В этой статье подробно рассмотрены особенности и применимые сценарии этих двух компонентов, чтобы помочь вам сделать наиболее подходящий выбор.

WARNING

- Для установки компонента хранения логов в кластере можно выбрать только один из ElasticSearch или Clickhouse.
- Для сбора логов и взаимодействия с данными хранилища можно выбрать компонент хранения логов любого кластера.
- В настоящее время продукт DevOps не поддерживает архивирование записей выполнения Jenkins pipeline с использованием Clickhouse. Если вам необходимы функции Jenkins pipeline, пожалуйста, осторожно выбирайте плагин ACP Log Storage с Clickhouse.
- В настоящее время продукт ServiceMesh не поддерживает интеграцию с Clickhouse. Если вам необходимы функции service mesh, пожалуйста, осторожно выбирайте плагин ACP Log Storage с Clickhouse.
- Плагин ACP Log Storage с Clickhouse в настоящее время не поддерживает кластеры с одиночным стеком IPv6 или двойным стеком IPv6.

Содержание

Сравнение архитектур

Архитектура ElasticSearch

ElasticSearch — это распределённый поисковый движок с открытым исходным кодом, построенный на базе Lucene, предназначенный для быстрого полнотекстового поиска и анализа. Его преимущества включают:

- Высокая производительность поиска: поддерживает поиск в реальном времени и может быстро обрабатывать огромные объёмы данных.
- Гибкие возможности запросов: предоставляет мощный DSL для запросов, поддерживающий сложные требования к поиску.
- Масштабируемость: легко масштабируется горизонтально по мере необходимости, подходит для приложений любого размера.
- Поддержка разнообразных данных: способен работать как со структурированными, так и с неструктурированными данными, широко применим.

Архитектура Clickhouse

Clickhouse — это высокопроизводительная колоночная база данных, предназначенная для Online Analytical Processing (OLAP). Его преимущества включают:

- Быстрая обработка данных: поддерживает быстрые запросы и анализ благодаря колоночному хранению и сжатию данных.
- Анализ в реальном времени: способен обрабатывать потоки данных в реальном времени, подходит для сценариев анализа данных в реальном времени.
- Высокая пропускная способность: оптимизирован для производительности при записи и запросах больших объёмов данных, что делает его очень подходящим для сценариев больших данных.

- Гибкая поддержка SQL: совместим со стандартным SQL, легко начать работу, снижая порог использования.

Сравнение функций

	Clickhouse	Elasticsearch	Объяснение
Высокая доступность	Поддерживается	Поддерживается	
Масштабируемость	Поддерживается	Поддерживается	
Опыт работы с запросами	Слабый	Сильный	Elasticsearch предлагает более мощные возможности пои на основе языка Lucene, тогда как Clickhouse поддерживает только SQL-запросы, что ограничивает его возможности поиска.
Использование ресурсов	Низкое	Высокое	При одинаковых требованиях к производительно Clickhouse требуе меньше ресурсов чем Elasticsearch. Например, для поддержки 20 000 логов в секунду Elasticsearch

	Clickhouse	Elasticsearch	Объяснение
			требует 3 es-masters и 7 es-nodes (2c4g+8c16 тогда как Clickhou — всего 3 реплик 2c4g.
Производительность	Высокая	Низкая	При одинаковых ресурсах объём логов, поддерживаемый Clickhouse, значительно превышает объём поддерживаемый Elasticsearch.
Активность сообщества	Средняя	Высокая	Сообщество Elasticsearch активно и облада богатой документацией, в время как сообщество Clickhouse развивается и совершенствуетс

Рекомендации по выбору

- Если вы привыкли использовать Elasticsearch и сильно зависите от языка Lucene, рекомендуется продолжать использовать плагин ACP Log Storage с ElasticSearch.

- Если вы зависите от функций Jenkins pipeline или service mesh платформы, рекомендуется продолжать использовать плагин ACP Log Storage с ElasticSearch.
- Если у вас высокие требования к производительности и потреблению ресурсов компонента логирования, но базовые потребности в запросах логов, рекомендуется выбрать плагин ACP Log Storage с Clickhouse.

Планирование ёмкости компонента логирования

Компонент хранения логов отвечает за сохранение логов, событий и данных аудита, собираемых компонентом сбора логов с одного или нескольких кластеров на платформе. Поэтому необходимо заранее оценить масштаб ваших логов и спланировать ресурсы, необходимые для компонента хранения логов, согласно рекомендациям в этом документе.

WARNING

- Представленные ниже данные являются стандартными показателями, полученными в ходе тестов, проведённых в лабораторных условиях, и предназначены для вашего ориентирования при планировании ресурсов. Вы должны обеспечить, чтобы фактически планируемые ресурсы превышали ресурсы, описанные в тестах, а масштаб логов не превышал соответствующий масштаб.
- Конфигурация диска для приведённых данных: `6000 iops` , `250MB/s скорость чтения и записи` , `SSD с независимым монтированием` . Если ваши фактические ресурсы хранения слабее тестовых, пожалуйста, ориентируйтесь на информацию о конфигурации для более крупного масштаба и при необходимости выделяйте больше CPU и памяти.

Содержание

Clickhouse

Один узел - Всего логов: 18000/c

Три узла - Всего логов: 20000/c

Шесть узлов - Всего логов: 40000/c

Девять узлов - Всего логов: 69000/c

ElasticSearch

Малый масштаб 3 узла - Всего логов: 6300/c

Компонент	Реплики	Лимит CPU	Лимит памяти
ElasticSearch	3	2C	4G
Kafka	3	2C	4G
Zookeeper	3	2C	4G
Lanaya	2	2C	4G
Razor	2	1C	2G

Малый масштаб 5 узлов - Всего логов: 9900/c

Компонент	Реплики	Лимит CPU	Лимит памяти
ElasticSearch	5	2C	4G
Kafka	3	2C	4G
Zookeeper	3	2C	4G
Lanaya	2	2C	4G
Razor	2	1C	2G

Крупный масштаб 3+5 узлов - Всего логов: 25000/с

Компонент	Реплики	Лимит CPU	Лимит памяти
ElasticSearch - Master	3	2C	4G
ElasticSearch - Data	5	8C	16G
Kafka	3	2C	4G
Zookeeper	3	2C	4G
Lanaya	2	2C	4G
Razor	2	1C	2G

Крупный масштаб 3+7 узлов - Всего логов: 30000/с

Компонент	Реплики	Лимит CPU	Лимит памяти
ElasticSearch - Master	3	2C	4G
ElasticSearch - Data	7	8C	16G
Kafka	3	2C	4G
Zookeeper	3	2C	4G
Lanaya	2	2C	4G
Razor	2	1C	2G

Clickhouse

Один узел - Всего логов: 18000/с

Компонент	Реплики	Лимит CPU	Лимит памяти	Примечания
Clickhouse	1	2C	4G	1 реплика 1 шард
Razor	1	1C	1G	-
Vector	1	2C	4G	-

Три узла - Всего логов: 20000/с

Компонент	Реплики	Лимит CPU	Лимит памяти	Примечания
Clickhouse	3	2C	4G	3 реплики 1 шард
Razor	2	1C	1G	-
Vector	2	2C	4G	-

Шесть узлов - Всего логов: 40000/с

Компонент	Реплики	Лимит CPU	Лимит памяти	Примечания
Clickhouse	3	4C	8G	3 реплики 2 шарда
Razor	2	1C	1G	-
Vector	2	4C	8G	-

Девять узлов - Всего логов: 69000/с

Компонент	Реплики	Лимит CPU	Лимит памяти	Примечания
Clickhouse	9	4C	8G	3 реплики 3 шарда
Razor	2	1C	1G	-
Vector	2	4C	8G	-

Основные понятия

Содержание

Open Source Components

Filebeat

Позиционирование: Легковесный сборщик логов
Описание: Компонент для сбора логов с открытым исходным кодом, устанавливаемый на узлах контейнеров, отвечающий за мониторинг файлов логов в реальном времени по заданным путям. Он собирает данные логов через входные модули, обрабатывает их и пересылает логи в Kafka или напрямую доставляет их в компоненты хранения через выходные модули. Поддерживает такие возможности, как агрегация многострочных логов и фильтрация полей для предварительной обработки.

Elasticsearch

Позиционирование: Распределённый движок поиска и аналитики
Описание: Движок полнотекстового поиска на базе Lucene, хранящий данные логов в формате JSON-документов и обеспечивающий поиск с близкой к реальному времени скоростью. Поддерживает динамическое отображение для автоматического распознавания типов полей и достигает быстрой поисковой выдачи по ключевым словам с помощью обратного индексирования, что подходит для поиска по логам и мониторинговых оповещений.

ClickHouse

Позиционирование: Колонко-ориентированная аналитическая база данных
Описание: Высокопроизводительная база данных с колонковым хранением, предназначенная для OLAP-сценариев, реализующая хранение логов на уровне петабайт с использованием движка MergeTree. Поддерживает высокоскоростные агрегирующие запросы, партиционирование по времени и стратегии TTL для данных, что делает её подходящей для анализа логов и статистической отчётности в пакетных вычислениях.

Kafka

Позиционирование: Распределённая очередь сообщений
Описание: Выступает в роли промежуточного программного обеспечения для системы логирования, обеспечивая высокопроизводительную буферизацию логов. При возникновении узких мест в обработке кластера Elasticsearch, принимает данные логов, отправленные Filebeat через Topics, способствуя снижению пиковых нагрузок и асинхронному потреблению, что гарантирует стабильность сбора логов.

Основные понятия функциональности

Конвейер сбора логов

Описание: Полный путь от генерации логов до их хранения, состоящий из четырёх этапов: `Сбор -> Передача -> Буферизация -> Хранение`. Поддерживает два режима конвейера:

- **Режим прямой записи**: Filebeat → Elasticsearch/ClickHouse
- **Режим буфера**: Filebeat → Kafka → Elasticsearch

Индекс

Описание: Логическая единица разбиения данных в Elasticsearch, аналогичная структуре таблицы в базах данных. Поддерживает создание временных роллирующих индексов (например, logstash-2023.10.01) и автоматизированное многоуровневое хранение hot-warm-cold с помощью Index Lifecycle Management (ILM).

Шарды и реплики

Описание:

- **Шард**: Физическая единица хранения, образующаяся при горизонтальном разбиении индекса в Elasticsearch, обеспечивающая распределяемую масштабируемость.
- **Реплика**: Копия каждого шарда, обеспечивающая высокую доступность данных и балансировку нагрузки запросов.

Колонковое хранение

Описание: Основной механизм хранения в ClickHouse, при котором данные сжимаются и хранятся по колонкам, что значительно снижает потребление ввода-вывода. Поддерживает следующие возможности:

- Векторизованный движок выполнения запросов
- Партиционирование и шардинг данных

- Материализованные представления для предварительной агрегации

Ключевые технические термины

Ingest Pipeline

Описание: Конвейер предварительной обработки данных в Elasticsearch, способный выполнять ETL-операции, такие как переименование полей, парсинг Grok и условную логику до записи данных.

Consumer Group

Описание: Механизм параллельного потребления в Kafka, при котором несколько экземпляров в одной группе потребителей могут параллельно обрабатывать сообщения из разных партиций, обеспечивая упорядоченную обработку сообщений.

TTL (Time To Live)

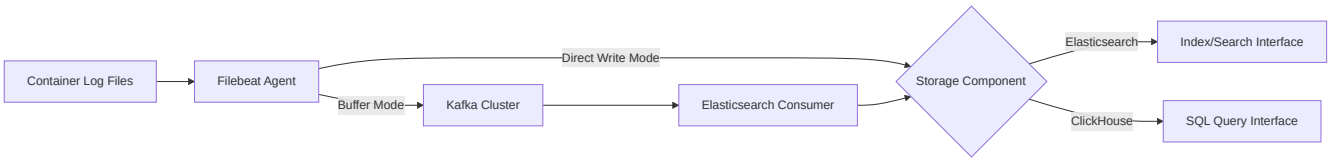
Описание: Стратегия жизненного цикла данных, поддерживающая два способа реализации:

- Elasticsearch: Автоматическое удаление устаревших индексов через политики ILM.
- ClickHouse: Автоматическое удаление партиций таблиц с помощью выражений TTL.

Replication Factor

Описание: Конфигурация избыточности данных на уровне Kafka Topic, определяющая количество реплик сообщений на разных брокерах, повышая надёжность данных.

Модель потока данных


Container Log Files
Filebeat Agent
Direct Write Mode
Buffer Mode
Kafka Cluster
Elasticsearch Consumer
Storage Component
Elasticsearch
Index/Search Interface
ClickHouse
SQL Query Interface

Menu

Руководства

Логи

Анализ запросов логов

Управление временем хранения логов приложений

Настройка частичного исключения логов приложений из сбора

Логи

Содержание

Анализ запросов логов

В панели анализа запросов логов центра операций вы можете просматривать логи стандартного вывода (stdout) учетной записи, под которой выполнен вход, в пределах её прав, включая системные логи, логи продуктов, логи Kubernetes и логи приложений. Через эти логи можно получить информацию о работе ресурсов.

- **Системные логи**: логи с хост-узлов, например: dmesg, syslog/messages, secure и др.
- **Логи продуктов**: логи собственных компонентов платформы и сторонних компонентов, интегрированных с платформой, например: Container-Platform, Platform-

Center, DevOps, Service-Mesh и др.

- **Логи Kubernetes**: логи компонентов, связанных с оркестрацией контейнеров Kubernetes, а также логи, генерируемые kubelet, kubeproxy и docker, например: docker, kube-apiserver, kube-controller-manager, etcd и др.
- **Логи приложений**: логи бизнес-приложений, включая файловые логи и логи стандартного вывода.

Условия запроса логов поддерживают фильтрацию логов в указанном временном диапазоне (выбранном или пользовательском), а результаты запроса отображаются в виде столбчатых диаграмм и стандартного вывода.

> **WARNING**
>
> В целях производительности платформа может отображать максимум 10 000 логов за один раз. Если объем логов на платформе слишком велик за определенный период, пожалуйста, сузьте временной диапазон запроса и выполняйте запрос логов поэтапно.

Поиск логов

1. В левой навигационной панели нажмите **Operations Center** > **Logs** > **Log Query Analysis**.
2. Выберите нужный тип лога, условия запроса, введите ключевые слова для поиска в содержимом логов и нажмите **Search**.

> **TIP**
>
> - Для разных **Log Types** доступны разные условия запроса.
> - Можно выбрать или ввести несколько тегов условий запроса; условия для разных типов ресурсов связаны логическим И (AND). Некоторые теги условий запроса поддерживают множественный выбор; после выбора обязательно нажмите клавишу `Enter` для подтверждения.
> - Условия запроса поддерживают нечеткий поиск; например, условие `pod = nginx` позволит найти логи для `nginx-1`, `nginx-2`.

- Условия поиска по содержимому логов используются только для поиска ключевых слов и поддерживают параметры `AND` и `OR` для ассоциативных запросов. Однако не используйте одновременно `AND` и `OR` в одном запросе.
- Столбчатая диаграмма показывает общее количество логов за текущий временной диапазон запроса и количество логов в разные моменты времени. Клик по столбцу диаграммы позволяет просмотреть логи за период между этим столбцом и следующим.

Экспорт данных логов

Страница может отображать максимум 10 000 записей логов. Если количество полученных логов слишком велико, вы можете использовать функцию экспорта логов для просмотра до 1 миллиона записей.

1. Нажмите кнопку **Export** в правом верхнем углу столбчатой диаграммы и настройте параметры в появившемся диалоговом окне экспорта логов.

   - **Scope**: диапазон экспорта логов, можно выбрать **Current Page** или **All Results**.
     - **Current Page**: экспортирует только результаты запроса на текущей странице, максимум 1000 записей.
     - **All Results**: экспортирует все данные логов, соответствующие текущим условиям запроса, максимум 1 миллион записей.
   - **Fields**: поля для отображения в логах. Можно выбрать, какие поля включать в экспортируемый файл, отметив соответствующие чекбоксы рядом с названием поля.

     Примечание: для разных типов логов доступны разные поля для выбора, выбирайте согласно вашим потребностям.
   - **Format**: формат экспортируемого файла лога, поддерживается `txt` или `csv`. Платформа экспортирует в сжатом формате `gzip`.

2. Нажмите **Export**, и браузер автоматически скачает сжатый файл на ваш локальный компьютер.

Просмотр контекста лога

1. Дважды кликните по области содержимого лога, и в текущем диалоговом окне отобразятся 5 логов до и после времени печати текущего лога, что помогает операторам лучше понять причины возникновения текущих логов ресурсов.
2. Вы можете настроить отображаемые поля контекста лога или экспортировать контекст лога. При экспорте контекста лога выбирать **Scope** не нужно; нажатие кнопки **Export** сразу скачает файл контекста лога на локальный компьютер через браузер.

Управление временем хранения логов приложений

Если политика проекта не установлена, время хранения логов приложений на платформе определяется параметром `Application Log Retention Time` **Log Storage Plugin**, установленного на **Storage Cluster**, выбранном при установке **ACP Log Collector** в кластере, где размещено приложение.

Вы можете задать разное время хранения для **Application Logs** на платформе, добавляя и управляя политиками логов проекта.

> **TIP**
>
> Политики проектов применяются только к **Application Logs** в рамках конкретного проекта. После установки политики проекта время хранения всех логов приложений в этом проекте будет соответствовать политике проекта.

Установка политик хранения администратором платформы

1. В левой навигационной панели нажмите **Operations Center** > **Logs** > **Policy Management**.
2. Нажмите **Add Project Policy**.
3. В выпадающем списке **Project** выберите проект.
4. Установите **Log Retention Time**.

- Используйте кнопки `-` / `+` по обе стороны счетчика для уменьшения/увеличения количества дней хранения или введите значение напрямую. Платформа позволяет задавать время хранения от 1 до 30 дней.
- Если введено десятичное число, оно округляется вверх до целого; если значение меньше 1, оно округляется до 1, и кнопка `-` становится неактивной; если значение больше 30, оно округляется вниз до 30, и кнопка `+` становится неактивной.

5. Нажмите **Add**.

Установка политик хранения администратором проекта

1. Перейдите на страницу деталей текущего проекта.
2. Нажмите кнопку редактирования рядом с полем политики логов, чтобы включить политику логов во всплывающем окне.
3. Установите **Log Retention Time**.

- Используйте кнопки `-` / `+` по обе стороны счетчика для уменьшения/увеличения количества дней хранения или введите значение напрямую. Платформа позволяет задавать время хранения от 1 до 30 дней.
- Если введено десятичное число, оно округляется вверх до целого; если значение меньше 1, оно округляется до 1, и кнопка `-` становится неактивной; если значение больше 30, оно округляется вниз до 30, и кнопка `+` становится неактивной.

Установка политик хранения через CLI

1. Войдите в кластер `global` и выполните команду:

```
kubectl edit project <Project Name>
```

2. Измените yaml согласно примеру ниже, сохраните и отправьте изменения.

```yaml
apiVersion: auth.alauda.io/v1
kind: Project
metadata:
  annotations:
    cpaas.io/creator: mschen1@alauda.io
    cpaas.io/description: ''
    cpaas.io/display-name: ''
    cpaas.io/operator: leizhuc
    cpaas.io/project.esPolicyLastEnabledTimestamp: '2025-02-18T09:53:54Z'
    cpaas.io/updated-at: '2025-02-18T09:53:54Z'
creationTimestamp: '2025-02-13T08:19:11Z'
finalizers:
  - namespace
generation: 1
labels:
  cpaas.io/project: bookinfo
  cpaas.io/project.esIndicesKeepDays: '7' # Время хранения логов приложений в проекте
  cpaas.io/project.esPolicyEnabled: 'true' # Включение политики проекта
  cpaas.io/project.id: '95447321'
  cpaas.io/project.level: '1'
  cpaas.io/project.parent: ''
name: bookinfo
### Остальная информация yaml, не связанная с изменениями, опущена.
```

Настройка частичного исключения логов приложений из сбора

Если вам нужно просматривать только **Реальные логи** определённых приложений в кластере без необходимости их хранения (коллектор будет отбрасывать соответствующие логи), вы можете воспользоваться этим разделом для настройки области остановки сбора логов (кластер, namespace, Pod) для тонкой настройки сбора логов приложений.

Остановить сбор всех логов приложений в кластере

Вы можете обновить **Configuration Parameters ACP Log Collector** кластера, чтобы отключить переключатель сбора **Application Log**, тем самым единообразно обновив область сбора логов для этого кластера. После отключения переключателя сбора для определённого типа логов сбор всех логов этого типа в текущем кластере прекратится.

Остановить сбор логов приложений в конкретном namespace

Вы можете отключить сбор логов для namespace, добавив метку `cpaas.io/log.mute=true` к указанному namespace, тем самым остановив сбор всех логов стандартного вывода и файловых логов для всех Pod в этом namespace.

Варианты настройки:

- **Через командную строку**: после входа на любой управляющий узел кластера выполните команду для обновления метки namespace.

```
kubectl label namespace <Namespace Name> cpaas.io/log.mute=true
```

- **Через интерфейс**: в разделе **Project Management** обновите метку namespace.

  1. В списке проектов в разделе **Project Management** нажмите на ***Project Name***, в котором находится namespace.
  2. В левой навигационной панели нажмите **Namespaces**.
  3. Нажмите на ***Namespace Name***, метку которого нужно обновить.
  4. На вкладке **Details** нажмите кнопку управления справа от **Labels**.
  5. Добавьте метку (Ключ: `cpaas.io/log.mute`, Значение: `true`) или измените значение существующей метки, затем нажмите **Update**.

Остановить сбор логов Pod

Вы можете отключить сбор логов для конкретного Pod, добавив метку `cpaas.io/log.mute=true` к нему, тем самым остановив сбор логов стандартного вывода и файловых логов для этого Pod.

После входа на любой управляющий узел кластера выполните команду для обновления метки Pod.

```
kubectl label pod <Pod Name> -n <Namespace Name> cpaas.io/log.mute=true
```

Примечание: Если Pod принадлежит вычислительному компоненту (Workload), вы можете обновить метки вычислительного компонента (Deployment, StatefulSet, DaemonSet, Job, CronJob), чтобы единообразно обновить метки всех Pod под этим компонентом. Метки сохранятся даже после пересоздания Pod.

Обновить метки вычислительного компонента можно следующим образом.

1. В представлении продукта **Container Platform** переключитесь в namespace, где находится Pod, через верхнюю навигацию.
2. В левой навигационной панели нажмите **Compute Components** > ***Тип вычислительного компонента, к которому принадлежит Pod***.
3. Нажмите кнопку управления справа от нужного вычислительного компонента > **Update**.
4. В правом верхнем углу нажмите **YAML** для перехода в режим редактирования YAML.
5. В поле **spec.template.labels** добавьте метку `cpaas.io/log.mute: 'true'`.

   Пример:

```
spec:
  template:
    metadata:
      namespace: tuhao-test
      creationTimestamp: null
      labels:
        app: spilo
        cpaas.io/log.mute: 'true'
        cluster-name: acid-minimal-cluster
        role: exporter
        middleware.instance/name: acid-minimal-cluster
        middleware.instance/type: PostgreSQL
```

6. Нажмите **Update**.

Menu

Как сделать

Как архивировать логи в стороннее хранилище

Передача во внешнее NFS

Передача во внешнее S3-хранилище

Как взаимодействовать с внешними кластерами ES Storage

Подготовка ресурсов

Порядок действий

Как архивировать логи в стороннее хранилище

В настоящее время логи, генерируемые платформой, сохраняются в компоненте хранения логов; однако период их хранения относительно короткий. Для предприятий с высокими требованиями к соответствию нормативам логи обычно требуют более длительного срока хранения для удовлетворения аудиторских требований. Кроме того, экономический аспект хранения также является одним из ключевых факторов для предприятий.

Исходя из вышеописанных сценариев, платформа предлагает решение для архивирования логов, позволяющее пользователям переносить логи во внешнее NFS или объектное хранилище.

Содержание

Передача во внешнее NFS

Предварительные требования

Ресурс	Описание
NFS	Заранее настроить сервис NFS и определить путь NFS для монтирования.
Kafka	Заранее получить адрес сервиса Kafka.
Адрес образа	Необходимо использовать CLI-инструмент в кластере `global` для выполнения следующих команд и получения адресов образов: - Получить адрес образа alpine: `kubectl get daemonset nevermore -n cpaas-system -o jsonpath='{.spec.template.spec.initContainers[0].image}'` - Получить адрес образа razor: `kubectl get deployment razor -n cpaas-system -o jsonpath='{.spec.template.spec.containers[0].image}'`

Создание ресурсов синхронизации логов

1. Нажмите **Cluster Management** > **Clusters** в левой навигационной панели.
2. Нажмите кнопку действий справа от кластера, в который будут передаваться логи > **CLI Tool**.
3. Измените YAML согласно описанию параметров ниже; после изменения вставьте код в открытый командный интерфейс **CLI Tool** и нажмите Enter для выполнения.

Тип ресурса	Путь поля	Описание
ConfigMap	`data.export.yml.output.compression`	Сжатие текста логов; поддерживаются варианты **none (без сжатия)**, **zlib**, **gzip**.
ConfigMap	`data.export.yml.output.file_type`	Тип экспортируемого файла лога; поддерживаются txt, csv, json.

Тип ресурса	Путь поля	Описание
ConfigMap	`data.export.yml.output.max_size`	Размер одного архивного файла в МБ. При превышении этого значения логи автоматически сжимаются и архивируются согласно настройке compression.
ConfigMap	`data.export.yml.scopes`	Область передачи логов; в настоящее время поддерживаются: системные логи, логи приложений, логи Kubernetes, продуктовые логи.
Deployment	`spec.template.spec.containers[0].command[7]`	Адрес сервиса Kafka.
Deployment	`spec.template.spec.volumes[3].hostPath.path`	Путь NFS для монтирования.
Deployment	`spec.template.spec.initContainers[0].image`	Адрес образа Alpine.
Deployment	`spec.template.spec.containers[0].image`	Адрес образа Razor.

```
cat << "EOF" |kubectl apply -f -
apiVersion: v1
data:
  export.yml: |
    scopes: # Область передачи логов; по умолчанию собираются только логи приложений
      system: false  # Системные логи
      workload: true # Логи приложений
      kubernetes: false # Логи Kubernetes
      platform: false # Продуктовые логи
    output:
      type: local
      path: /cpaas/data/logarchive
      layout: TimePrefixed
      # Размер одного архивного файла в МБ. При превышении этого значения логи автоматически сжимаются и архивируются согласно настройке compression.
      max_size: 200
      compression: zlib    # Опционально: none (без сжатия) / zlib / gzip
      file_type: txt   # Опционально: txt csv json
kind: ConfigMap
metadata:
  name: log-exporter-config
  namespace: cpaas-system

---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    service_name: log-exporter
  name: log-exporter
  namespace: cpaas-system
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 5
  selector:
    matchLabels:
      service_name: log-exporter
  strategy:
    rollingUpdate:
      maxSurge: 0
      maxUnavailable: 1
```

```
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: lanaya
        cpaas.io/product: Platform-Center
        service_name: log-exporter
        version: v1
      namespace: cpaas-system
    spec:
      affinity:
        podAffinity: {}
        podAntiAffinity:
          preferredDuringSchedulingIgnoredDuringExecution:
            - podAffinityTerm:
                labelSelector:
                  matchExpressions:
                    - key: service_name
                      operator: In
                      values:
                        - log-exporter
                topologyKey: kubernetes.io/hostname
              weight: 50
      initContainers:
        - args:
            - -ecx
            - |
              chown -R 697:697 /cpaas/data/logarchive
          command:
            - /bin/sh
          image: registry.example.cn:60080/ops/alpine:3.16 # Адрес образа Alpine
          imagePullPolicy: IfNotPresent
          name: chown
          resources:
            limits:
              cpu: 100m
              memory: 200Mi
            requests:
              cpu: 10m
              memory: 50Mi
          securityContext:
            runAsUser: 0
          terminationMessagePath: /dev/termination-log
```

```
          terminationMessagePolicy: File
          volumeMounts:
            - mountPath: /cpaas/data/logarchive
              name: data
      containers:
        - command:
          - /razor
          - consumer
          - --v=1
          - --kafka-group-log=log-nfs
          - --kafka-auth-enabled=true
          - --kafka-tls-enabled=true
          - --kafka-endpoint=192.168.143.120:9092  # Заполнить согласно реальной
среде
          - --database-type=file
          - --export-config=/etc/log-export/export.yml
          image: registry.example.cn:60080/ait/razor:v3.16.0-beta.3.g3df8e987  #
Образ Razor
          imagePullPolicy: Always
          livenessProbe:
            failureThreshold: 5
            httpGet:
              path: /metrics
              port: 8080
              scheme: HTTP
            initialDelaySeconds: 20
            periodSeconds: 10
            successThreshold: 1
            timeoutSeconds: 3
          name: log-export
          ports:
            - containerPort: 80
              protocol: TCP
          readinessProbe:
            failureThreshold: 5
            httpGet:
              path: /metrics
              port: 8080
              scheme: HTTP
            initialDelaySeconds: 20
            periodSeconds: 10
            successThreshold: 1
            timeoutSeconds: 3
          resources:
```

```
        limits:
          cpu: "2"
          memory: 4Gi
        requests:
          cpu: 440m
          memory: 1280Mi
      securityContext:
        runAsGroup: 697
        runAsUser: 697
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
      volumeMounts:
        - mountPath: /etc/secrets/kafka
          name: kafka-basic-auth
          readOnly: true
        - mountPath: /etc/log-export
          name: config
          readOnly: true
        - mountPath: /cpaas/data/logarchive
          name: data
  dnsPolicy: ClusterFirst
  nodeSelector:
    kubernetes.io/os: linux
  restartPolicy: Always
  schedulerName: default-scheduler
  securityContext:
    fsGroup: 697
  serviceAccount: lanaya
  serviceAccountName: lanaya
  terminationGracePeriodSeconds: 10
  tolerations:
    - effect: NoSchedule
      key: node-role.kubernetes.io/master
      operator: Exists
    - effect: NoSchedule
      key: node-role.kubernetes.io/control-plane
      operator: Exists
    - effect: NoSchedule
      key: node-role.kubernetes.io/cpaas-system
      operator: Exists
  volumes:
    - name: kafka-basic-auth
      secret:
        defaultMode: 420
```

```
            secretName: kafka-basic-auth
        - name: elasticsearch-basic-auth
          secret:
            defaultMode: 420
            secretName: elasticsearch-basic-auth
        - configMap:
            defaultMode: 420
            name: log-exporter-config
          name: config
        - hostPath:
            path: /cpaas/data/logarchive    # Путь NFS для монтирования
            type: DirectoryOrCreate
          name: data
EOF
```

4. После того как статус контейнера изменится на Running, вы сможете просматривать непрерывно архивируемые логи в пути NFS; структура каталогов файлов логов следующая:

```
/cpaas/data/logarchive/$date/$project/$namespace-$cluster/logfile
```

Передача во внешнее S3-хранилище

Предварительные требования

Ресурс	Описание
S3 Storage	Заранее подготовить адрес сервиса S3-хранилища, получить значения `access_key_id` и `secret_access_key`; создать бакет для хранения логов.
Kafka	Заранее получить адрес сервиса Kafka.
Адрес образа	Необходимо использовать CLI-инструмент в кластере `global` для выполнения следующих команд и получения адресов образов: - Получить адрес образа alpine: `kubectl get daemonset nevermore -n cpaas-system -o jsonpath='{.spec.template.spec.initContainers[0].image}'`

Ресурс	Описание
	- Получить адрес образа razor: `kubectl get deployment razor -n cpaas-system -o jsonpath='{.spec.template.spec.containers[0].image}'`

Создание ресурсов синхронизации логов

1. Нажмите **Cluster Management** > **Clusters** в левой навигационной панели.
2. Нажмите кнопку действий справа от кластера, в который будут передаваться логи > **CLI Tool**.
3. Измените YAML согласно описанию параметров ниже; после изменения вставьте код в открытый командный интерфейс **CLI Tool** и нажмите Enter для выполнения.

Тип ресурса	Путь поля	Описание
Secret	`data.access_key_id`	Base64-кодированное значение полученного access_key_id.
Secret	`data.secret_access_key`	Base64-кодированное значение полученного secret_access_key.
ConfigMap	`data.export.yml.output.compression`	Сжатие текста логов; поддерживаются варианты **none (без сжатия)**, **zlib**, **gzip**.
ConfigMap	`data.export.yml.output.file_type`	Тип экспортируемого файла лога;

Тип ресурса	Путь поля	Описание
		поддерживаются txt, csv, json.
ConfigMap	`data.export.yml.output.max_size`	Размер одного архивного файла в МБ. При превышении этого значения логи автоматически сжимаются и архивируются согласно настройке compression.
ConfigMap	`data.export.yml.scopes`	Область передачи логов; в настоящее время поддерживаются: системные логи, логи приложений, логи Kubernetes, продуктовые логи.
ConfigMap	`data.export.yml.output.s3.bucket_name`	Имя бакета.
ConfigMap	`data.export.yml.output.s3.endpoint`	Адрес сервиса S3-хранилища.
ConfigMap	`data.export.yml.output.s3.region`	Регион сервиса S3-хранилища.
Deployment	`spec.template.spec.containers[0].command[7]`	Адрес сервиса Kafka.
Deployment	`spec.template.spec.volumes[3].hostPath.path`	Локальный путь для монтирования, используется для

Тип ресурса	Путь поля	Описание
		временного хранения информации логов. После синхронизации с S3 файлы логов автоматически удаляются.
Deployment	`spec.template.spec.initContainers[0].image`	Адрес образа Alpine.
Deployment	`spec.template.spec.containers[0].image`	Адрес образа Razor.

```
cat << "EOF" |kubectl apply -f -
apiVersion: v1
type: Opaque
data:
  # Должны содержаться следующие два ключа
  access_key_id: bWluaW9hZG1pbg==  # Base64-кодированное значение полученного access_key_id
  secret_access_key: bWluaW9hZG1pbg==  # Base64-кодированное значение полученного secret_access_key
kind: Secret
metadata:
  name: log-export-s3-secret
  namespace: cpaas-system

---
apiVersion: v1
data:
  export.yml: |
    scopes: # Область передачи логов; по умолчанию собираются только логи приложений
      system: false  # Системные логи
      workload: true # Логи приложений
      kubernetes: false # Логи Kubernetes
      platform: false # Продуктовые логи
    output:
      type: s3
      path: /cpaas/data/logarchive

      s3:
        s3forcepathstyle: true
        bucket_name: baucket_name_s3           # Заполнить подготовленным именем бакета
        endpoint: http://192.168.179.86:9000   # Заполнить подготовленным адресом сервиса S3-хранилища
        region: "dummy"                        # Информация о регионе
        access_secret: log-export-s3-secret
        insecure: true

      layout: TimePrefixed
      # Размер одного архивного файла в МБ. При превышении этого значения логи автоматически сжимаются и архивируются согласно настройке compression.
      max_size: 200
      compression: zlib                        # Опционально: none (без сжатия) /
```

```
zlib / gzip
      file_type: txt                          # Опционально: txt, csv, json
kind: ConfigMap
metadata:
  name: log-exporter-config
  namespace: cpaas-system

---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    service_name: log-exporter
  name: log-exporter
  namespace: cpaas-system
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 5
  selector:
    matchLabels:
      service_name: log-exporter
  strategy:
    rollingUpdate:
      maxSurge: 0
      maxUnavailable: 1
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: lanaya
        cpaas.io/product: Platform-Center
        service_name: log-exporter
        version: v1
      namespace: cpaas-system
    spec:
      affinity:
        podAffinity: {}
        podAntiAffinity:
          preferredDuringSchedulingIgnoredDuringExecution:
            - podAffinityTerm:
                labelSelector:
                  matchExpressions:
```

```
                  - key: service_name
                    operator: In
                    values:
                      - log-exporter
              topologyKey: kubernetes.io/hostname
            weight: 50
      initContainers:
        - args:
            - -ecx
            - |
              chown -R 697:697 /cpaas/data/logarchive
          command:
            - /bin/sh
          image: registry.example.cn:60080/ops/alpine:3.16 # Адрес образа Alpine
          imagePullPolicy: IfNotPresent
          name: chown
          resources:
            limits:
              cpu: 100m
              memory: 200Mi
            requests:
              cpu: 10m
              memory: 50Mi
          securityContext:
            runAsUser: 0
          terminationMessagePath: /dev/termination-log
          terminationMessagePolicy: File
          volumeMounts:
            - mountPath: /cpaas/data/logarchive
              name: data
      containers:
        - command:
            - /razor
            - consumer
            - --v=1
            - --kafka-group-log=log-s3
            - --kafka-auth-enabled=true
            - --kafka-tls-enabled=true
            - --kafka-endpoint=192.168.179.86:9092  # Заполнить адресом сервиса
Kafka согласно реальной среде
            - --database-type=file
            - --export-config=/etc/log-export/export.yml
          image: registry.example.cn:60080/ait/razor:v3.16.0-beta.3.g3df8e987  #
Образ Razor
```

```
      imagePullPolicy: Always
      livenessProbe:
        failureThreshold: 5
        httpGet:
          path: /metrics
          port: 8080
          scheme: HTTP
        initialDelaySeconds: 20
        periodSeconds: 10
        successThreshold: 1
        timeoutSeconds: 3
      name: log-export
      ports:
        - containerPort: 80
          protocol: TCP
      readinessProbe:
        failureThreshold: 5
        httpGet:
          path: /metrics
          port: 8080
          scheme: HTTP
        initialDelaySeconds: 20
        periodSeconds: 10
        successThreshold: 1
        timeoutSeconds: 3
      resources:
        limits:
          cpu: "2"
          memory: 4Gi
        requests:
          cpu: 440m
          memory: 1280Mi
      securityContext:
        runAsGroup: 697
        runAsUser: 697
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
      volumeMounts:
        - mountPath: /etc/secrets/kafka
          name: kafka-basic-auth
          readOnly: true
        - mountPath: /etc/log-export
          name: config
          readOnly: true
```

```
            - mountPath: /cpaas/data/logarchive
              name: data
      dnsPolicy: ClusterFirst
      nodeSelector:
        kubernetes.io/os: linux
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext:
        fsGroup: 697
      serviceAccount: lanaya
      serviceAccountName: lanaya
      terminationGracePeriodSeconds: 10
      tolerations:
        - effect: NoSchedule
          key: node-role.kubernetes.io/master
          operator: Exists
        - effect: NoSchedule
          key: node-role.kubernetes.io/control-plane
          operator: Exists
        - effect: NoSchedule
          key: node-role.kubernetes.io/cpaas-system
          operator: Exists
      volumes:
        - name: kafka-basic-auth
          secret:
            defaultMode: 420
            secretName: kafka-basic-auth
        - name: elasticsearch-basic-auth
          secret:
            defaultMode: 420
            secretName: elasticsearch-basic-auth
        - configMap:
            defaultMode: 420
            name: log-exporter-config
          name: config
        - hostPath:
            path: /cpaas/data/logarchive    # Локальный путь временного хранения
логов
            type: DirectoryOrCreate
          name: data
EOF
```

4. После того как статус контейнера изменится на Running, вы сможете просматривать

непрерывно архивируемые логи в бакете.

Как взаимодействовать с внешними кластерами ES Storage

Вы можете взаимодействовать с внешними кластерами Elasticsearch или Kafka, создавая YAML-конфигурации. В зависимости от ваших бизнес-требований, вы можете выбрать взаимодействие только с внешним кластером Elasticsearch (при этом Kafka устанавливается в текущем кластере), либо взаимодействовать одновременно с внешними кластерами Elasticsearch и Kafka.

TIP

Поддерживаемые версии для взаимодействия с внешним Elasticsearch следующие:

- Elasticsearch 6.x поддерживает версии 6.6 - 6.8;
- Elasticsearch 7.x поддерживает версии 7.0 - 7.10.2, рекомендуется использовать 7.10.2.

Содержание

Подготовка ресурсов

Перед взаимодействием необходимо подготовить требуемую информацию для аутентификации.

1. В левой навигационной панели нажмите **Cluster Management** > **Resource Management**, затем переключитесь на кластер, в котором необходимо установить плагин.

2. Нажмите **Create Resource Object** и заполните поле кода, изменив параметры согласно комментариям в коде.

- Учетные данные, необходимые для взаимодействия с внешним Elasticsearch:

```
apiVersion: v1
type: Opaque
data:
  password: dEdWQVduSX5kUW1mc21acg==  # Должно быть закодировано в base64. Команда для справки: echo -n <password_value>| base64
  username: YWRtaW4=                  # Должно быть закодировано в base64. Команда для справки: echo -n <username_value>| base64
kind: Secret
metadata:
  name: elasticsearch-basic-auth      # Имя учетных данных. Убедитесь, что значение elasticsearch.basicAuthSecretName в YAML лог-хранилища совпадает с этим параметром.
  namespace: cpaas-system             # Пространство имён, в котором расположен компонент Elasticsearch, обычно cpaas-system.
```

- Если необходимо использовать внешний кластер Kafka, также нужно создать учетные данные для взаимодействия с внешним Kafka:

```
apiVersion: v1
type: Opaque
data:
  password: dEdWQVduSX5kUW1mc21acg==  # Должно быть закодировано в base64.
Команда для справки: echo -n <password_value>| base64
  username: YWRtaW4=                  # Должно быть закодировано в base64.
Команда для справки: echo -n <username_value>| base64
kind: Secret
metadata:
  name: kafka-basic-auth              # Имя учетных данных. Убедитесь, что
значение kafka.basicAuthSecretName в YAML лог-хранилища совпадает с этим
параметром.
  namespace: cpaas-system             # Пространство имён, в котором расположен
компонент Kafka, обычно cpaas-system.
```

1. Нажмите **Create**.

Порядок действий

1. В левой навигационной панели нажмите **App Store** > **Plugin Management**.
2. В верхней навигации выберите ***Cluster Name***, в котором хотите установить плагин ACP Log Storage с Elasticsearch.
3. Нажмите кнопку действий справа от **ACP Log Storage with Elasticsearch** > **Install**.
4. Включите переключатель **Interface with External Elasticsearch**, настройте YAML-файл, пример взаимодействия и описание параметров приведены ниже:

- Взаимодействие с внешним кластером Elasticsearch при установке Kafka в текущем кластере:

```
elasticsearch:
  install: false
  address: http://fake:9200                # Адрес доступа к внешнему ES,
например, http://192.168.143.252:11780/es_proxy
  basicAuthSecretName: elasticsearch-basic-auth # Учетные данные для
взаимодействия с внешним Elasticsearch, созданные на этапе подготовки.
storageClassConfig:
  type: "LocalVolume"   # По умолчанию LocalVolume. Варианты: "LocalVolume" или
"StorageClass".
kafka:
  auth: true                               # Включена ли аутентификация.
  k8sNodes:
  - log1                                   # Имя узла, полученное командой kubectl
get nodes.
  - log2
  - log3
  storageSize: 10                          # Размер хранилища в Gi, по умолчанию
10 Gi.
```

- Взаимодействие с внешними кластерами Elasticsearch и Kafka одновременно:

```
elasticsearch:
  install: false
  address: http://fake:9200                # Адрес доступа к внешнему ES,
например, http://192.168.143.252:11780/es_proxy
  basicAuthSecretName: elasticsearch-basic-auth # Учетные данные для
взаимодействия с внешним Elasticsearch, созданные на этапе подготовки.
kafka:
  auth: true                               # Включена ли аутентификация.
  install: false
  basicAuthSecretName: kafka-basic-auth  # Учетные данные для взаимодействия с
внешним Kafka, созданные на этапе подготовки.
  address: 192.168.130.169:9092,192.168.130.187:9092,192.168.130.193:9092     #
Адреса доступа к Kafka, разделённые запятыми.
```

Menu

События

Введение

Ограничения использования

События

Процедуры работы

Обзор событий

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 срабатывать для уведомления соответствующего персонала с целью своевременного вмешательства, что снижает операционные риски на платформе.

Содержание

Ограничения использования

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

События

Содержание

Процедуры работы

1. Нажмите **Operations Center** > **Events** в левой навигационной панели.

   Совет: Используйте выпадающий список в верхней навигационной панели для переключения кластера и просмотра событий.

Обзор событий

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

- **Обзор значимых событий**: Эта карточка показывает причину значимых событий и количество ресурсов, у которых произошли такие события в выбранном временном диапазоне.
  - **Примечание**: Если один и тот же ресурс испытывал этот 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 испытывать их несколько раз.

- **Записи событий ресурсов**: 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 просмотр и ввести условия запроса для поиска. Условия запроса следующие:

  - **Тип ресурса**: Тип Kubernetes-ресурса, у которого произошло событие, например Pod.
  - **Namespace**: Пространство имён Kubernetes-ресурса, в котором произошло событие.
  - **Причина события**: Причина возникновения события.
  - **Уровень события**: Значимость события, например Warning.
  - **Имя ресурса**: 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 во всплывающем диалоговом окне **Event Details**.
- Цвет иконки слева от причины события указывает уровень события. Зелёная иконка означает, что уровень события `Normal`, и это событие можно игнорировать; оранжевая иконка означает, что уровень события `Warning`, что указывает на аномалию в ресурсе, и за этим событием следует следить, чтобы предотвратить инциденты.

Menu

Инспекция

Введение

Введение

Ограничения по использованию

Архитектура

Архитектура

Inspection

Component Health Status

Руководства

Inspection

Execute Inspection

Inspection Configuration

Inspection Report Explanation

Component Health Status

Procedures to Operate

Введение

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

Этот модуль обеспечивает 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 алгоритмами автоматизированной оценки, модуль позволяет организациям снижать затраты на ручную инспекцию, проактивно выявлять аномалии ресурсов, снижать бизнес-риски и поддерживать оптимальную производительность платформы посредством систематических проверок состояния.

Содержание

Ограничения по использованию

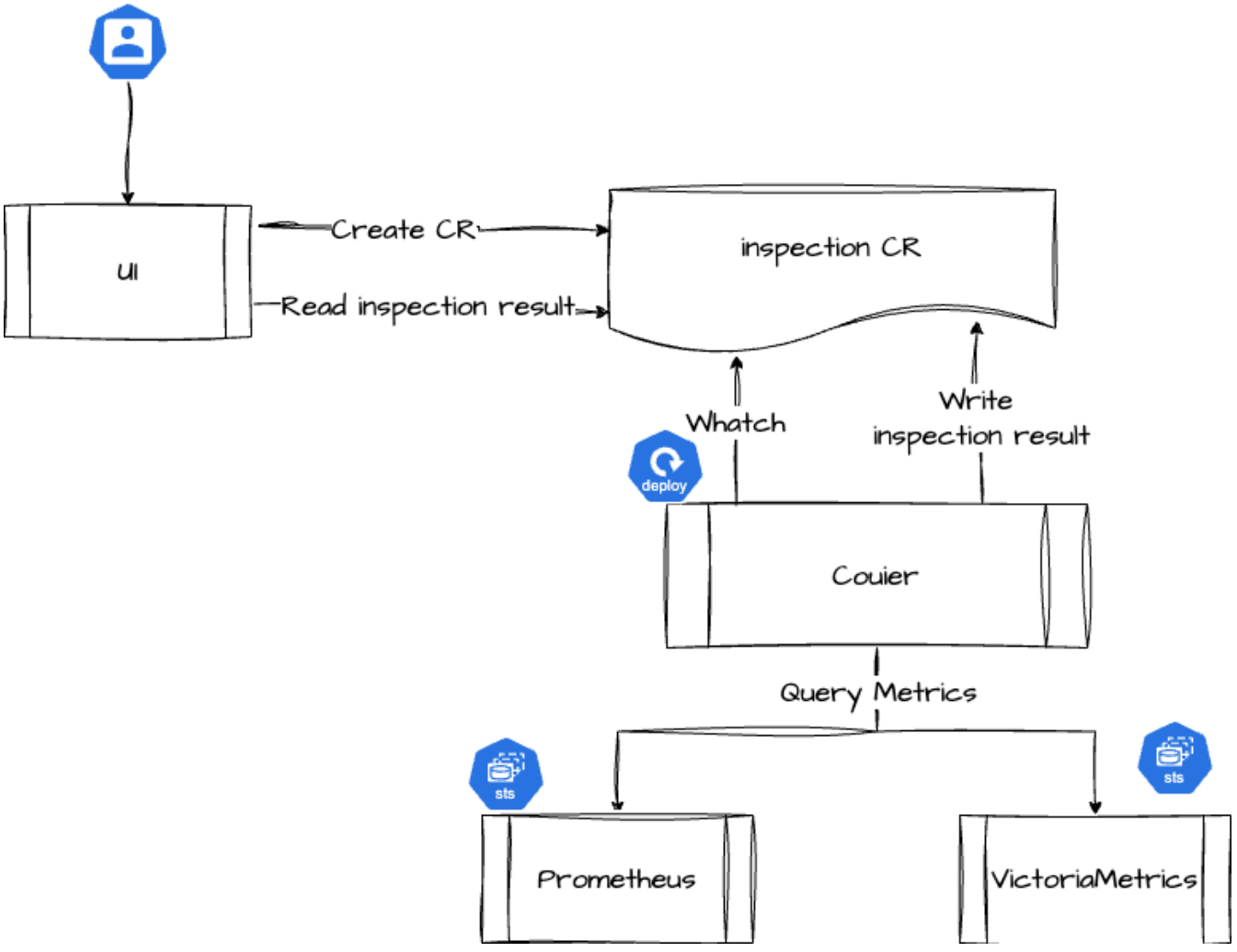
- Некоторые элементы инспекции на платформе зависят от наличия в кластерах установленных компонентов мониторинга. Пожалуйста, убедитесь, что в каждом кластере заранее установлен либо плагин ACP Monitoring с Prometheus, либо плагин ACP Monitoring с VictoriaMetrics.
- Инспекция платформы поддерживает отправку 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

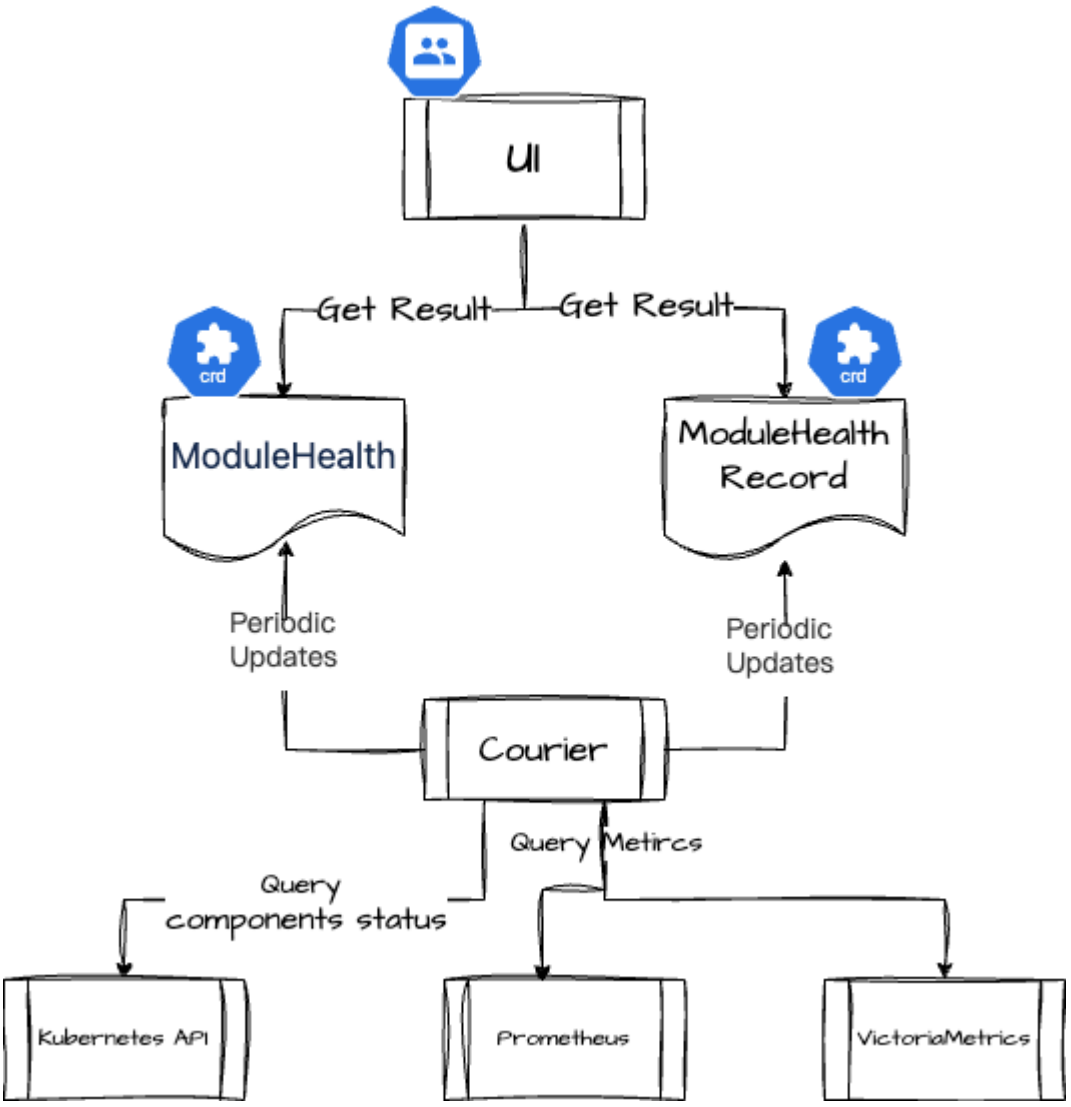
Inspection



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

- Создание задачи инспекции: Платформа отправляет CR типа inspection в кластер `global`.
- Выполнение задачи инспекции: Компонент Courier отслеживает появление CR типа inspection 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



Статус здоровья компонентов совместно предоставляется компонентом платформы 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, подлежащих мониторингу, 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 записывает результаты инспекции в ресурсы CR ModuleHealth и ModuleHealthRecord.
- Определение состояния компонентов: Courier запрашивает данные у Kubernetes и компонентов мониторинга для определения фактического состояния компонентов и

выявления существующих проблем.

  - Kubernetes: Проверяет, установлен ли компонент и нормальное ли количество реплик компонента.
  - Prometheus / VictoriaMetrics: На основе метрик, предоставляемых каждым компонентом, выполняет запросы и определяет, может ли компонент нормально предоставлять сервисы.
- Просмотр статуса здоровья компонентов: Пользователи могут проверить статус здоровья каждого компонента через платформу, где данные будут получены из соответствующих ресурсов CR ModuleHealth и ModuleHealthRecord.

Menu

Руководства

Inspection

Execute Inspection

Inspection Configuration

Inspection Report Explanation

Component Health Status

Procedures to Operate

Inspection

Содержание

Execute Inspection

1. Нажмите **Operation Center** > **Inspection** > **Basic Inspection** в левой навигационной панели.

   Совет: На странице инспекции отображается информация о данных инспекции из последней проверки. Во время процесса инспекции вы можете в реальном времени просматривать данные ресурсов завершённых инспекций.

2. На странице Basic Inspection поддерживаются следующие действия:

   - **Execute Inspection**: Нажмите кнопку **Inspection** в правом верхнем углу страницы, чтобы выполнить инспекцию на платформе.
   - **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, что загрузит отчёт соответствующего формата 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 инспекции.
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 оповещения контактов.
Inspection Configuration Items	Измените пороги предупреждений или отключите элементы инспекции в соответствии с элементами инспекции по умолчанию платформы для сертификатов, хостов кластера и pod-ов.

Inspection Report Explanation

Most Recent Inspection

В области информации **Most Recent Inspection** вы можете просмотреть соответствующую информацию о последней инспекции:

- **Inspection Time**: Время начала и окончания последней инспекции.
- **Total Number of Inspection Resources**: Общее количество ресурсов (кластеров, узлов, pod-ов, сертификатов), проверенных в последней инспекции.
- **Risks**: Количество ресурсов с рисками, включая те, которые классифицированы как **Fault** и **Warning**.

Resource Risk Inspection

На странице **Resource Risk Inspection** вы можете просмотреть обзор информации о рисках для `global` кластеров, собственных кластеров, подключённых кластеров, а также всех узлов, pod-ов 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 этого типа ресурса. На странице деталей вы можете 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 статуса 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 рисковыми данными.

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/pod-ов/метрик) ресурсы компонентов мониторинга не обновлены. - После увеличения объёма логов и частоты сбора логов ресурсы лог-компонентов не обновлены. - Использование CPU кластера превышает 60%; - Использование памяти кластера превышает 60%; - Любой pod компонента ETCD кластера находится не в состоянии Running; - Любой хост кластера находится не в состоянии Ready; - Разница времени между любыми двумя узлами кластера превышает 40 секунд; - Коэффициент запроса CPU кластера (фактическое значение / общее) превышает 60%; - Коэффициент запроса памяти кластера (фактическое значение / общее) превышает 80%; - Компоненты мониторинга не установлены в кластере;

Resource Type	Inspection Scope	Fault Judgment Conditions	Warning Judgment Conditions
			- Компоненты мониторинга кластера работают с ошибками; - Любой pod компонента **kube-controller-manager** кластера находится не в состоянии Running; - Любой pod компонента **kube-scheduler** кластера находится не в состоянии Running; - Любой pod компонента **kube-apiserver** кластера находится не в состоянии Running.
Node	- Все управляющие узлы - Все вычислительные узлы	- Статус узла **Abnormal**; - Pod компонента **node-exporter** на узле находится не в состоянии Running; - Pod компонента **kubelet** на узле находится не в состоянии 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

Resource Type	Inspection Scope	Fault Judgment Conditions	Warning Judgment Conditions
			одно событие NodeOOM (недостаток памяти); - За последние сутки произошло как минимум одно событие NodeTaskHung (зависание задачи); - За последние сутки произошло как минимум одно событие NodeCorruptDockerImage (повреждённый Docker-образ).
pod	Все pod-ы	- Статус pod-а **Error**; - Pod находится в состоянии запуска более 5 минут.	- Использование CPU pod-а превышает 80%; - Использование памяти pod-а превышает 80%; - Количество перезапусков pod-а 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

Перейдите на вкладку **Resource Utilization Inspection**, чтобы открыть страницу **Resource Utilization Inspection**.

На странице **Resource Utilization Inspection** вы можете просмотреть общий объём, использование и коэффициент использования CPU, памяти и диска для `global`

кластеров, подключённых кластеров и собственных кластеров, а также количество ресурсов, таких как кластеры, узлы, pod-ы и проекты на платформе.

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

Component Health Status

Страница состояния 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, состояние 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

Procedures to Operate

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 на странице, где можно просмотреть детальную

информацию о проблемах с неисправными компонентами.