Безопасность

Alauda Container Security

Alauda Container Security

Аутентификация кластера Alauda

Аутентификация кластера Alauda

Безопасность и соответствие

Соответствие требованиям

API Refiner

О службе с

Служба соотве
сканирования
готовые возмо
подробной отч

Конфигурации безопасности платформы

Изменение OIDC Client Secret

Предварительные требования

Обзор OIDC Secret

Процедура

Откат

Отключение метода PKCE Plain

Предварительные требования

Процедура

Проверка

Откат

Пользователи и роли

Пользователь

Группа

Роль

IDP

Пользовательская политика

Мультиарендность (Project)

Введение **Руководства**

Project

Namespaces

Relationship Between Clusters, Projects, and Namespaces

Аудит

Введение

Требования

Процедура

Результаты поиска

Телеметрия

Установка

Предварительные требования

Шаги установки

Включение онлайн-операций

Шаги удаления

Сертификаты

Автоматическая ротация серт **cert-manager** **Сертифика**

Installation

How it works

Operation Considerations

Обзор

Как это работает

Определение сертификатов, управляем

Связанные ресурсы

Мониторин

Мониторинг ст

Встроенные пр

Ротация TLS сертификатов адресов доступа платформы

Предварительные требования

Процедуры

Alauda Container Security

Alauda Container Security — это комплексное решение для обеспечения безопасности, разработанное для Kubernetes и контейнеризированных сред. Оно предоставляет централизованное управление, автоматизированное сканирование уязвимостей, применение политик и проверки соответствия, помогая организациям защищать свою контейнерную инфраструктуру в нескольких кластерах.

Alauda Container Security использует распределённую архитектуру на основе контейнеров, состоящую из Central Services (для управления, API и UI) и Secured Cluster Services (для мониторинга, применения политик и сбора данных). Оно интегрируется с CI/CD пайплайнами, SIEM, системами логирования и поддерживает встроенный сканер уязвимостей Scanner V4.

Note

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

Аутентификация кластера Alauda

Плагин аутентификации кластера Alauda Container Platform обеспечивает независимую интеграцию аутентификации для сценариев сбоев глобального кластера.

Когда глобальный кластер `global` выходит из строя, пользователи всё равно могут войти через этот сервис для доступа и управления кластерами Kubernetes, сохраняя права, соответствующие состоянию до сбоя глобального кластера.

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

Безопасность и соответствие

Соответствие требованиям

Введение

Руководство

Установка Alauda Container Pl

Установка через консоль

Обновлени

Матрица совме

ции

API Refiner

Введение

Введение в продукт

Ограничения

Установка Alauda Container Pl

Установка через консоль

Установка через YAML

Процедура удаления

Конфигурация по умолчанию

Обновлени

Матрица совме

Рекомендации

О службе соответствия платформы контейнеров Alauda

О службе соответствия платформы контейнеров Alauda

Служба соответствия — это модуль платформы, предназначенный для поддержки сканирования на соответствие STIG и операционной системы MicroOS. Он обеспечивает готовые возможности сканирования на соответствие с поддержкой планового сканирования и подробной отчетности.

Соответствие требованиям

Введение

Введение

Установка Alauda Container Platform Compliance с Kyverno

Установка Alauda Container Platform Compliance с Kyverno

Установка через консоль

Установка через YAML

Процедура удаления

Обновление

Обновление

Матрица совместимости

Рекомендации по пути обновления

Руководство

Конфигурация доступа к прие

Зачем Kyverno нужен доступ к реестру?

Быстрый старт

Политика проверки подписи с

Что такое проверка подписи образа?

Быстрый старт

Распространённые сценарии использов

Политика п

Зачем использ

Быстрый старт

Способы созда

Распространён

Политика проверки реестра образов

Что такое проверка реестра образов?

Быстрый старт

Распространённые сценарии

Расширенные шаблоны

Лучшие практики

Политика предотвращения выхода из конте

Что такое предотвращение выхода из ко

Быстрый старт

Основные политики предотвращения вы

Расширенные сценарии

Тестирование и проверка

Лучшие практики

Политика П

Что такое При

Быстрый старт

Основные Пол

Расширенные

ние

Политика сетевой безопасности

Что такое сетевая безопасность?

Быстрый старт

Основные политики сетевой безопасности

Расширенные сценарии

Тестирование и проверка

Политика безопасности томов

Что такое безопасность томов?

Быстрый старт

Введение

ACP предоставляет функциональность соответствия требованиям на основе компонента с открытым исходным кодом Kyverno, позволяя организациям определять и применять политики в своих кластерах Kubernetes.

Эта функция решает задачу поддержания единых стандартов безопасности, управления и эксплуатации, позволяя пользователям создавать пользовательские политики с использованием синтаксиса YAML Kyverno и автоматически проверять ресурсы на соответствие этим политикам.

Функциональность соответствия обеспечивает комплексный мониторинг и отчетность по нарушениям, предлагая как представления на уровне ресурсов, так и на уровне политик через интуитивно понятный интерфейс, помогая командам быстро выявлять несоответствующие ресурсы и принимать соответствующие меры для поддержания желаемого уровня безопасности и соответствия нормативным требованиям.

INFO

Для получения дополнительной информации о Kyverno прочитайте Kyverno Documentation ↗.

Установка Alauda Container Platform Compliance с Kyverno

Alauda Container Platform Compliance с Kyverno — это сервис платформы, который интегрирует Kyverno для управления политиками соответствия на Alauda Container Platform.

Содержание

Установка через консоль

1. Перейдите в раздел **Administrator**
2. В левой навигационной панели выберите **Marketplace** > **Cluster Plugins**
3. Найдите **Alauda Container Platform Compliance with Kyverno** и кликните для просмотра деталей
4. Нажмите **Install** для развертывания плагина

Установка через YAML

1. Проверка доступных версий

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

```
# kubectl get moduleplugins kyverno
NAME      AGE
kyverno   4d20h

# kubectl get moduleconfigs -l cpaas.io/module-name=kyverno
NAME             AGE
kyverno-v4.0.4   4d21h
```

Это означает, что ModulePlugin `kyverno` существует в кластере, а версия `v4.0.4` 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: kyverno
    cpaas.io/module-name: '{"en": "Alauda Container Platform Compliance f
or Kyverno",
      "zh": "Alauda Container Platform Compliance for Kyverno"}'
  labels:
    cpaas.io/cluster-name: global
    cpaas.io/module-name: kyverno
    cpaas.io/module-type: plugin
    cpaas.io/product: Platform-Center
  name: kyverno-global
spec:
  version: v4.2.0
```

Объяснение полей:

- `name`: Временное имя для плагина кластера. Платформа переименует его после создания на основе содержимого в формате `<cluster-name>-<hash of content>`, например, `global-ee98c9991ea1464aaa8054bdacbab313`.
- `label cpaas.io/cluster-name`: Указывает кластер, в который должен быть установлен плагин.
- `label cpaas.io/module-name`: Имя плагина, должно совпадать с ресурсом ModulePlugin.
- `label cpaas.io/module-type`: Фиксированное поле, должно быть `plugin`; отсутствие этого поля приведёт к ошибке установки.
- `.spec.config`: Если соответствующий ModuleConfig пуст, это поле можно оставить пустым.
- `.spec.version`: Указывает версию плагина для установки, должна совпадать с `.spec.version` в ModuleConfig.

Процедура удаления

1. Выполните шаги 1-3 из процесса установки, чтобы найти плагин

2. Нажмите **Uninstall** для удаления плагина

Обновление

INFO

В этом документе изложены принципы пути обновления и поддерживаемая совместимость версий для Alauda Container Platform Compliance for Kyverno.

Содержание

Матрица совместимости

В таблице ниже перечислены поддерживаемые версии Alauda Container Platform Compliance for Kyverno:

Версия Compliance for Kyverno	Поддерживаемая версия Alauda Container Platform
4.3.x	4.2.x, 4.3.x

Версия Compliance for Kyverno	Поддерживаемая версия Alauda Container Platform
4.2.x	4.2.x

Рекомендации по пути обновления

Обновление вместе с ACP

- **Описание**: В ACP 4.1.x и более ранних выпусках плагины Compliance for Kyverno синхронизировались по жизненному циклу с ACP и обновлялись в составе выпуска ACP. Для следующих путей обновления плагины обновляются вместе с ACP до версии 4.3.0 в процессе обновления ACP.
- **Поддерживаемые пути обновления ACP**:
  - `4.0.x -> 4.3.0`
  - `4.1.x -> 4.3.0`
- **Результат**: После завершения обновления все плагины Compliance for Kyverno обновляются до версии 4.3.0.

Обновление минорной версии

- **Описание**: Плагины Compliance for Kyverno поддерживают обновление минорной версии, когда ACP использует совместимую версию.
- **Предварительное условие**: Версия ACP должна соответствовать приведенной выше матрице совместимости.
- **Пример**: `4.2.x -> 4.3.x`

Обновление на уровне патча

- **Описание**: Обновления между любыми версиями патча в пределах одной и той же минорной версии полностью совместимы и могут выполняться напрямую.
- **Пример**: `4.3.0 -> 4.3.x`

Руководство

Конфигурация доступа к прие

Зачем Kyverno нужен доступ к реестру?

Быстрый старт

Политика проверки подписи с

Что такое проверка подписи образа?

Быстрый старт

Распространённые сценарии использов

Политика п

Зачем использ

Быстрый старт

Способы созда

Распространён

Политика проверки реестра образов

Что такое проверка реестра образов?

Быстрый старт

Распространённые сценарии

Расширенные шаблоны

Лучшие практики

Политика предотвращения выхода из конте

Что такое предотвращение выхода из ко

Быстрый старт

Основные политики предотвращения вы

Расширенные сценарии

Тестирование и проверка

Лучшие практики

Политика П

Что такое При

Быстрый старт

Основные Пол

Расширенные

чие

Политика сетевой безопасности

Что такое сетевая безопасность?

Быстрый старт

Основные политики сетевой безопасности

Расширенные сценарии

Тестирование и проверка

Политика безопасности томов

Конфигурация доступа к приватному реестру

В этом руководстве показано, как настроить Kyverno для доступа к приватным контейнерным реестрам. Когда Kyverno необходимо проверить подписи образов или получить детали образа, требуются соответствующие учетные данные для доступа к приватным реестрам — как пропуск для входа в охраняемое здание.

Содержание

Зачем Kyverno нужен доступ к реестру?

Kyverno нужен доступ к реестрам, когда он:

- **Проверяет подписи образов**: загружает данные подписей, чтобы убедиться, что образы правильно подписаны

- **Проверяет метаданные образов**: читает метки, аннотации и информацию манифеста образа
- **Сканирует на уязвимости**: загружает образы для проверки безопасности
- **Проверяет содержимое образов**: инспектирует, что именно находится внутри контейнерных образов

Можно представить это как охранника, которому нужно проверить удостоверение личности — Kyverno должен «увидеть» образы, чтобы их проверить.

Быстрый старт

1. Создайте секрет для реестра

```
# Для приватного реестра компании
kubectl create secret docker-registry my-registry-secret \
  --docker-server=registry.company.com \
  --docker-username=<username> \
  --docker-password=<password> \
  --docker-email=<email@company.com> \
  -n kyverno
```

2. Настройте Kyverno на использование секрета (рекомендуется)

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: kyverno
  namespace: kyverno
imagePullSecrets:
- name: my-registry-secret
```

3. Конфигурация деплоя Kyverno

Если требуется более тонкая настройка, можно изменить деплой Kyverno напрямую:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kyverno
  namespace: kyverno
spec:
  replicas: 1
  selector:
    matchLabels:
      app: kyverno
  template:
    metadata:
      labels:
        app: kyverno
    spec:
      serviceAccountName: kyverno
      imagePullSecrets:
      - name: my-registry-secret
      - name: gcr-secret
      - name: dockerhub-secret
      containers:
      - name: kyverno
        image: ghcr.io/kyverno/kyverno:latest
        env:
        - name: REGISTRY_CREDENTIAL_HELPERS
          value: "ecr-login,gcr,acr-env"  # Включить помощники по учетным данным
        # ... другая конфигурация
```

Политика проверки подписи образа

В этом руководстве показано, как настроить Kyverno для проверки того, что образы контейнеров правильно подписаны перед запуском в кластере Kubernetes. Это похоже на проверку удостоверения личности — только образы с действительными «подписями» допускаются к запуску.

Содержание

Что такое проверка подписи образа?

Проверка подписи образа — это как охранник, проверяющий удостоверения у входа. Она гарантирует:

- **Образы подлинные**: они действительно исходят от заявленного источника
- **Образы не изменены**: никто не модифицировал их после подписания
- **Запускаются только доверенные образы**: неподписанные или неправильно подписанные образы блокируются
- **Аудит**: отслеживание, какие образы и когда были проверены

Быстрый старт

1. Генерация ключей

```
# Создайте пару ключей для подписи (как создание системы удостоверений)
cosign generate-key-pair
# Это создаст: cosign.key (приватный, хранить в секрете) и cosign.pub (пу
бличный, можно распространять)
```

2. Подпишите образы

```
# Подпишите образы (как поставить официальную печать)
cosign sign --key cosign.key registry.company.com/app:v1.0.0
```

3. Создайте базовую политику проверки

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-signed-images
spec:
  validationFailureAction: Enforce  # Блокировать неподписанные образы
  background: false
  rules:
    - name: check-signatures
      match:
        any:
        - resources:
            kinds:
            - Pod
      verifyImages:
      - imageReferences:
        - "registry.company.com/*"  # Проверять образы из реестра компании
        attestors:
        - count: 1
          entries:
          - keys:
              publicKeys: |-
                -----BEGIN PUBLIC KEY-----
                # Вставьте содержимое cosign.pub сюда
                MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8nXRh950IZbRj8Ra/N9sbqOPZrfM
                5/KAQN0/KjHcorm/J5yctVd7iEcnessRQjU917hmKO6JWVGHpDguIyakZA==
                -----END PUBLIC KEY-----
        mutateDigest: true  # Преобразовать теги в безопасный формат digest
```

4. Проверьте работу

```
# Примените политику
kubectl apply -f signature-policy.yaml

# Попытайтесь запустить неподписанный образ (должно не получиться)
kubectl run test --image=nginx:latest

# Попытайтесь запустить подписанный образ (должно сработать)
kubectl run test --image=registry.company.com/app:v1.0.0
```

Распространённые сценарии использования

Сценарий 1: Несколько команд должны подписывать критические образы

Для критических приложений могут потребоваться подписи как от команды разработки, так и от команды безопасности:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-dual-signatures
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: critical-app-signatures
      match:
        any:
        - resources:
            kinds:
            - Pod
      verifyImages:
      - imageReferences:
        - "registry.company.com/critical/*"
        attestors:
        # Подписи обеих команд обязательны
        - count: 1  # Подпись команды безопасности
          entries:
          - keys:
              publicKeys: |-
                -----BEGIN PUBLIC KEY-----
                # Публичный ключ команды безопасности
                MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8nXRh950IZbRj8Ra/N9sb
qOPZrfM
                5/KAQN0/KjHcorm/J5yctVd7iEcnessRQjU917hmKO6JWVGHpDguIyakZ
A==
                -----END PUBLIC KEY-----
        - count: 1  # Подпись команды разработки
          entries:
          - keys:
              publicKeys: |-
                -----BEGIN PUBLIC KEY-----
                # Публичный ключ команды разработки
                MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEyctVd7iEcnessRQjU917h
mKO6JWV
                GHpDguIyakZA8nXRh950IZbRj8Ra/N9sbqOPZrfM5/KAQN0/KjHcorm/J
5==
                -----END PUBLIC KEY-----
        mutateDigest: true
```

Сценарий 2: Разные правила для разных сред

В production требуется строгая проверка, в development — более мягкая:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: environment-specific-verification
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    # Строгие правила для production
    - name: production-must-be-signed
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - production
      verifyImages:
      - imageReferences:
        - "*"  # Все образы должны быть подписаны
        failureAction: Enforce  # Блокировать, если не подписан
        attestors:
        - count: 1
          entries:
          - keys:
              publicKeys: |-
                -----BEGIN PUBLIC KEY-----
                # Ключ для подписи production
                MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8nXRh950IZbRj8Ra/N9sb
qOPZrfM
                5/KAQN0/KjHcorm/J5yctVd7iEcnessRQjU917hmKO6JWVGHpDguIyakZ
A==
                -----END PUBLIC KEY-----
        mutateDigest: true

    # Более мягкие правила для development
    - name: development-warn-unsigned
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
```

```
            - development
            - staging
      verifyImages:
      - imageReferences:
        - "registry.company.com/*"  # Проверять только образы компании
        failureAction: Audit  # Аудит, но разрешать неподписанные образы
        attestors:
        - count: 1
          entries:
          - keys:
              publicKeys: |-
                -----BEGIN PUBLIC KEY-----
                # Ключ для подписи development
                MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEyctVd7iEcnessRQjU917h
mKO6JWV
                GHpDguIyakZA8nXRh950IZbRj8Ra/N9sbqOPZrfM5/KAQN0/KjHcorm/J
5==
                -----END PUBLIC KEY-----
        mutateDigest: true
```

Сценарий 3: Использование сертификатов вместо ключей

В корпоративных средах могут использоваться сертификаты X.509:

```
# Подписать с помощью сертификата
cosign sign --cert company-cert.pem --cert-chain ca-chain.pem \
  registry.company.com/myapp:v1.0.0
```

Политика проверки подписи образа с использованием Secrets

В этом руководстве показано, как использовать Kubernetes Secrets для хранения публичных ключей для проверки подписей образов Kyverno, обеспечивая лучшую безопасность и управление ключами по сравнению с внедрением ключей непосредственно в политики.

Содержание

Сценарий 4: Офлайн-среда с использованием Secrets

Зачем использовать Secrets для публичных ключей?

Использование Kubernetes Secrets для хранения публичных ключей имеет несколько преимуществ:

- **Повышенная безопасность**: ключи надежно хранятся в хранилище Kubernetes Secret
- **Простая ротация ключей**: обновление ключей без изменения политик
- **Контроль доступа**: использование RBAC для управления доступом к секретам

Быстрый старт

1. Генерация и хранение ключей в Secret

```
# Генерация пары ключей cosign
cosign generate-key-pair

# Создание секрета из файла с публичным ключом
kubectl create secret generic cosign-public-key \
  --from-file=cosign.pub=./cosign.pub \
  --namespace=kyverno

# Проверка создания секрета
kubectl get secret cosign-public-key -n kyverno
```

2. Конфигурация RBAC для Kyverno

Создайте Service Account для Kyverno

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: kyverno-secret-reader
  namespace: kyverno
```

Создайте Role для доступа к Secret

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: kyverno
  name: secret-reader
rules:
- apiGroups: [""]
  resources: ["secrets"]
  verbs: ["get", "list", "watch"]
  resourceNames: ["cosign-public-key", "team-keys"]  # Только конкретные
секреты
```

Привяжите Role к Service Account

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: read-secrets
  namespace: kyverno
subjects:
- kind: ServiceAccount
  name: kyverno-secret-reader
  namespace: kyverno
roleRef:
  kind: Role
  name: secret-reader
  apiGroup: rbac.authorization.k8s.io
```

3. Создание политики с использованием ссылки на Secret

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: verify-with-secret
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: check-signatures
      match:
        any:
        - resources:
            kinds: [Pod]
      verifyImages:
      - imageReferences:
        - "registry.company.com/*"
        attestors:
        - count: 1
          entries:
          - keys:
              secret:
                name: cosign-public-key
                namespace: kyverno
                key: cosign.pub
              rekor:
                url: https://rekor.sigstore.dev
        mutateDigest: true
```

4. Тестирование конфигурации

```
# Подписать образ
cosign sign --key cosign.key registry.company.com/app:v1.0.0

# Применить политику
kubectl apply -f verify-with-secret.yaml

# Тест с подписанным образом (должно работать)
kubectl run test --image=registry.company.com/app:v1.0.0

# Тест с неподписанным образом (должно завершиться ошибкой)
kubectl run test-fail --image=nginx:latest
```

Способы создания Secret

Метод 1: Из файла

```
# Создание секрета из существующего файла с публичным ключом cosign
kubectl create secret generic cosign-public-key \
  --from-file=cosign.pub=./cosign.pub \
  --namespace=kyverno
```

Метод 2: Из литеральной строки

```
# Создание секрета с содержимым публичного ключа inline
kubectl create secret generic cosign-public-key \
  --from-literal=cosign.pub="-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8nXRh950IZbRj8Ra/N9sbqOPZrfM
5/KAQN0/KjHcorm/J5yctVd7iEcnessRQjU917hmKO6JWVGHpDguIyakZA==
-----END PUBLIC KEY-----" \
  --namespace=kyverno
```

Метод 3: Из YAML-манифеста

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: cosign-public-key
  namespace: kyverno
  labels:
    app: kyverno
    component: image-verification
type: Opaque
stringData:
  cosign.pub: |
    -----BEGIN PUBLIC KEY-----
    MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8nXRh950IZbRj8Ra/N9sbqOPZrfM
    5/KAQN0/KjHcorm/J5yctVd7iEcnessRQjU917hmKO6JWVGHpDguIyakZA==
    -----END PUBLIC KEY-----
```

```
kubectl apply -f cosign-secret.yaml
```

Распространённые сценарии использования

Сценарий 1: Одна команда с одним секретом

Простая настройка, когда одна команда управляет всеми подписями образов:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: single-team-verification
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: verify-team-signatures
      match:
        any:
        - resources:
            kinds: [Pod, Deployment, StatefulSet, DaemonSet]
      exclude:
        any:
        - resources:
            namespaces: [kube-system, kyverno]

      verifyImages:
      - imageReferences:
        - "registry.company.com/*"
        - "gcr.io/myproject/*"

        failureAction: Enforce

        attestors:
        - count: 1
          entries:
          - keys:
              secret:
                name: team-cosign-key
                namespace: kyverno
                key: cosign.pub
              rekor:
                url: https://rekor.sigstore.dev

        mutateDigest: true
        verifyDigest: true
        required: true
```

Сценарий 2: Несколько команд с разными секретами

Разные команды имеют свои ключи подписи и секреты:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: multi-team-verification
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    # Образы команды frontend
    - name: verify-frontend-images
      match:
        any:
        - resources:
            kinds: [Pod]
            namespaces: [frontend-*]

      verifyImages:
      - imageReferences:
        - "registry.company.com/frontend/*"

        attestors:
        - count: 1
          entries:
          - keys:
              secret:
                name: frontend-team-key
                namespace: kyverno
                key: cosign.pub
              rekor:
                url: https://rekor.sigstore.dev

        mutateDigest: true
        required: true

    # Образы команды backend
    - name: verify-backend-images
      match:
        any:
        - resources:
            kinds: [Pod]
            namespaces: [backend-*]

      verifyImages:
```

```
    - imageReferences:
      - "registry.company.com/backend/*"

      attestors:
      - count: 1
        entries:
        - keys:
            secret:
              name: backend-team-key
              namespace: kyverno
              key: cosign.pub
            rekor:
              url: https://rekor.sigstore.dev

      mutateDigest: true
      required: true
```

Сценарий 3: Критические образы, требующие нескольких подписей

Среды с повышенной безопасностью, где несколько команд должны подписывать критические образы:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: critical-multi-signature
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: verify-critical-images
      match:
        any:
        - resources:
            kinds: [Pod]
            namespaces: [production]

      verifyImages:
      - imageReferences:
        - "registry.company.com/critical/*"

        failureAction: Enforce

        attestors:
        # Подпись команды безопасности (обязательно)
        - count: 1
          entries:
          - keys:
              secret:
                name: security-team-key
                namespace: kyverno
                key: security.pub
              rekor:
                url: https://rekor.sigstore.dev

        # Подпись команды разработки (обязательно)
        - count: 1
          entries:
          - keys:
              secret:
                name: dev-team-key
                namespace: kyverno
                key: development.pub
              rekor:
                url: https://rekor.sigstore.dev
```

```
      # Подпись команды релиза (обязательно)
      - count: 1
        entries:
        - keys:
            secret:
              name: release-team-key
              namespace: kyverno
              key: release.pub
            rekor:
              url: https://rekor.sigstore.dev

      mutateDigest: true
      required: true
```

Сценарий 4: Офлайн-среда с использованием Secrets

Использование секретов в изолированных (air-gapped) средах:

Политика проверки реестра образов

В этом руководстве показано, как настроить Kyverno для контроля того, какие контейнерные реестры могут использоваться в кластере Kubernetes. Реализуются политики контроля доступа к реестрам, чтобы гарантировать, что разворачиваются только образы из одобренных и доверенных реестров.

Содержание

Что такое проверка реестра образов?

Проверка реестра обеспечивает централизованный контроль источников образов. Она позволяет:

- **Контролировать источники образов**: разрешать только образы из доверенных реестров
- **Блокировать рискованные реестры**: предотвращать использование неизвестных или скомпрометированных реестров
- **Обеспечивать соответствие требованиям**: выполнять требования безопасности по источникам образов
- **Разные правила для разных сред**: строгие правила для продакшена, более мягкие для разработки
- **Отслеживать использование**: мониторить, какие реестры используются

Быстрый старт

1. Заблокировать все, кроме корпоративного реестра

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: company-registry-only
spec:
  validationFailureAction: Enforce  # Блокировать неразрешённые образы
  background: false
  rules:
    - name: check-registry
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: "Разрешён только корпоративный реестр: registry.company.com"
        pattern:
          spec:
            containers:
            - image: "registry.company.com/*"
```

2. Проверка

```bash
# Применить политику
kubectl apply -f registry-policy.yaml

# Это должно завершиться ошибкой (nginx из Docker Hub)
kubectl run test --image=nginx:latest

# Это должно сработать (если образы есть в реестре)
kubectl run test --image=registry.company.com/nginx:latest
```

Распространённые сценарии

Сценарий 1: Разрешить несколько доверенных реестров

Организации обычно используют несколько реестров:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: multiple-trusted-registries
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: check-approved-registries
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: "Образы должны поступать из одобренных реестров: корпоративный реестр, GCR или официальные Docker-образы"
        anyPattern:
        - spec:
            containers:
            - image: "registry.company.com/*"      # Корпоративный реестр
        - spec:
            containers:
            - image: "gcr.io/project-name/*"       # Google Container Registry
        - spec:
            containers:
            - image: "docker.io/library/*"         # Только официальные Docker-образы
        - spec:
            containers:
            - image: "quay.io/organization/*"      # Red Hat Quay
```

Сценарий 2: Разные правила для разных сред

Для продакшена нужны строгие правила, для разработки — более гибкие:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: environment-based-registry-rules
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    # Продакшен: только сертифицированные образы
    - name: production-strict-registries
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - production
            - prod-*
      validate:
        message: "В продакшене разрешены только сертифицированные корпоративные образы"
        pattern:
          spec:
            containers:
            - image: "registry.company.com/certified/*"

    # Разработка: разрешено больше реестров
    - name: development-flexible-registries
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - development
            - dev-*
            - staging
            - test-*
      validate:
        message: "В разработке разрешены корпоративный реестр, GCR и официальные Docker-образы"
        anyPattern:
        - spec:
```

```
    containers:
    - image: "registry.company.com/*"
- spec:
    containers:
    - image: "gcr.io/dev-project/*"
- spec:
    containers:
    - image: "docker.io/library/*"
- spec:
    containers:
    - image: "docker.io/organization/*"
```

Сценарий 3: Блокировать конкретные рискованные реестры

Блокировать определённые реестры, разрешая остальные:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: block-risky-registries
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    # Метод 1: Использовать deny-лист
    - name: block-untrusted-registries
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: "Образы из untrusted-registry.com не разрешены"
        deny:
          conditions:
          - key: "{{ request.object.spec.containers[?contains(image, 'untrusted-registry.com')] | length(@) }}"
            operator: GreaterThan
            value: 0

    # Метод 2: Использовать allow-лист для Docker Hub (только официальные образы)
    - name: allow-only-official-dockerhub
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: "Разрешены только официальные образы Docker Hub (docker.io/library/*)"
        deny:
          conditions:
          - key: "{{ request.object.spec.containers[?starts_with(image, 'docker.io/') && !starts_with(image, 'docker.io/library/')] | length(@) }}"
            operator: GreaterThan
            value: 0
```

Сценарий 4: Доступ к реестрам для разных команд

Разным командам разрешён доступ к разным реестрам:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: team-specific-registries
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    # Команда frontend может использовать Node.js образы
    - name: frontend-team-registries
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - frontend-*
      validate:
        message: "Команда frontend может использовать корпоративный реест
р и официальные Node.js образы"
        anyPattern:
        - spec:
            containers:
            - image: "registry.company.com/*"
        - spec:
            containers:
            - image: "docker.io/library/node:*"
        - spec:
            containers:
            - image: "docker.io/library/nginx:*"

    # Команда data может использовать ML/AI реестры
    - name: data-team-registries
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - data-*
            - ml-*
      validate:
        message: "Команда data может использовать корпоративный реестр и
```

```
образы ML/AI"
        anyPattern:
        - spec:
            containers:
            - image: "registry.company.com/*"
        - spec:
            containers:
            - image: "docker.io/tensorflow/*"
        - spec:
            containers:
            - image: "docker.io/pytorch/*"
        - spec:
            containers:
            - image: "nvcr.io/nvidia/*"
```

Расширенные шаблоны

Эффективное использование подстановочных знаков

```
# Примеры шаблонов:
- image: "registry.company.com/*"           # Любой образ из этого реестр
а
- image: "registry.company.com/team-a/*"    # Только образы команды team-
a
- image: "*/database:*"                     # Любой образ базы данных из
любого реестра
- image: "gcr.io/project-*/app:*"          # Любой app из проекта project
-* в GCR
```

Лучшие практики

Начинайте с предупреждений

```
spec:
  validationFailureAction: Audit  # Начинайте с режима аудита, без блокир
овки
```

Исключайте системные неймспейсы

```
rules:
  - name: check-registries
    match:
      any:
      - resources:
          kinds:
          - Pod
    exclude:
      any:
      - resources:
          namespaces:
          - kube-system
          - kyverno
          - kube-public
```

Распространённые ошибки

1. **Неправильный формат образа**:
   - ❌ `registry.company.com:5000/app` (отсутствует протокол)
   - ✅ `registry.company.com/app:latest`
2. **Ошибки с подстановочными знаками**:
   - ❌ `registry.company.com*` (отсутствует слэш)
   - ✅ `registry.company.com/*`
3. **Формат Docker Hub**:
   - ❌ `nginx` (неявный docker.io)
   - ✅ `docker.io/library/nginx`

Политика предотвращения выхода из контейнера

В этом руководстве показано, как настроить Kyverno для предотвращения атак с выходом из контейнера, блокируя конфигурации контейнеров с высоким уровнем риска, которые могут позволить контейнерам выйти за пределы их изоляционных границ.

Содержание

Что такое предотвращение выхода из контейнера?

Предотвращение выхода из контейнера включает обнаружение и блокировку опасных конфигураций контейнеров, которые могут позволить злоумышленникам выйти из изоляции контейнера и получить доступ к хост-системе. Это включает:

- **Привилегированные контейнеры**: контейнеры, работающие с повышенными привилегиями
- **Доступ к пространствам имён хоста**: контейнеры, использующие пространства имён PID, сети или IPC хоста
- **Монтирование путей хоста**: контейнеры, монтирующие пути файловой системы хоста
- **Опасные capabilities**: контейнеры с избыточными Linux capabilities
- **Доступ к портам хоста**: контейнеры, привязывающиеся к сетевым портам хоста

Быстрый старт

1. Блокировка привилегированных контейнеров

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-privileged-containers
  annotations:
    policies.kyverno.io/title: Disallow Privileged Containers
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Privileged mode disables most security mechanisms and must not be a
llowed.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: privileged-containers
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Privileged mode is disallowed. The fields spec.containers[*].se
curityContext.privileged,
          spec.initContainers[*].securityContext.privileged, and spec.eph
emeralContainers[*].securityContext.privileged
          must be unset or set to false.
        pattern:
          spec:
            =(ephemeralContainers):
              - =(securityContext):
                  =(privileged): "false"
            =(initContainers):
              - =(securityContext):
                  =(privileged): "false"
            containers:
              - =(securityContext):
                  =(privileged): "false"
```

2. Тестирование политики

```
# Применить политику
kubectl apply -f disallow-privileged-containers.yaml

# Попытка создать привилегированный контейнер (должно не пройти)
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-privileged
spec:
  containers:
  - name: nginx
    image: nginx
    securityContext:
      privileged: true
EOF

# Попытка создать обычный контейнер (должно пройти)
kubectl run test-normal --image=nginx

# Очистка
kubectl delete pod test-privileged test-normal --ignore-not-found
```

Основные политики предотвращения выхода из контейнера

Политика 1: Запрет доступа к пространствам имён хоста

Запретить контейнерам доступ к пространствам имён хоста:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-host-namespaces
  annotations:
    policies.kyverno.io/title: Disallow Host Namespaces
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Host namespaces (Process ID namespace, Inter-Process Communication
namespace, and
      network namespace) allow access to shared information and can be us
ed to elevate
      privileges. Pods should not be allowed access to host namespaces.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-namespaces
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Sharing the host namespaces is disallowed. The fields spec.host
Network,
          spec.hostIPC, and spec.hostPID must be unset or set to false.
        pattern:
          spec:
            =(hostPID): "false"
            =(hostIPC): "false"
            =(hostNetwork): "false"
```

Политика 2: Запрет монтирования путей хоста

Блокировать контейнеры от монтирования путей файловой системы хоста:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-host-path
  annotations:
    policies.kyverno.io/title: Disallow Host Path
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod,Volume
    policies.kyverno.io/description: >-
      HostPath volumes let Pods use host directories and volumes in conta
iners.
      Using host resources can be used to access shared data or escalate
privileges
      and should not be allowed.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-path
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          HostPath volumes are forbidden. The field spec.volumes[*].hostP
ath must be unset.
        pattern:
          spec:
            =(volumes):
              - X(hostPath): "null"
```

Политика 3: Запрет использования портов хоста

Запретить контейнерам привязываться к сетевым портам хоста:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-host-ports
  annotations:
    policies.kyverno.io/title: Disallow Host Ports
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Access to host ports allows potential snooping of network traffic a
nd should not be
      allowed, or at minimum restricted to a known list.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-ports-none
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Use of host ports is disallowed. The fields spec.containers[*].
ports[*].hostPort,
          spec.initContainers[*].ports[*].hostPort, and spec.ephemeralCon
tainers[*].ports[*].hostPort
          must either be unset or set to 0.
        pattern:
          spec:
            =(ephemeralContainers):
              - =(ports):
                  - =(hostPort): 0
            =(initContainers):
              - =(ports):
                  - =(hostPort): 0
            containers:
              - =(ports):
                  - =(hostPort): 0
```

Политика 4: Запрет опасных capabilities

Блокировать контейнеры от добавления опасных Linux capabilities:

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-capabilities-strict
  annotations:
    policies.kyverno.io/title: Disallow Capabilities (Strict)
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Adding capabilities other than `NET_BIND_SERVICE` is disallowed. In
addition,
      all containers must explicitly drop `ALL` capabilities.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: require-drop-all
      match:
        any:
        - resources:
            kinds:
            - Pod
      preconditions:
        all:
        - key: "{{ request.operation || 'BACKGROUND' }}"
          operator: NotEquals
          value: DELETE
      validate:
        message: >-
          Containers must drop `ALL` capabilities.
        foreach:
        - list: request.object.spec.[ephemeralContainers, initContainers,
containers][]
          deny:
            conditions:
              all:
              - key: ALL
                operator: AnyNotIn
                value: "{{ element.securityContext.capabilities.drop || `
[]` }}"
    - name: adding-capabilities
      match:
```

```
        any:
        - resources:
            kinds:
            - Pod
      preconditions:
        all:
        - key: "{{ request.operation || 'BACKGROUND' }}"
          operator: NotEquals
          value: DELETE
      validate:
        message: >-
          Any capabilities added other than NET_BIND_SERVICE are disallow
ed.
        foreach:
        - list: request.object.spec.[ephemeralContainers, initContainers,
containers][]
          deny:
            conditions:
              any:
              - key: "{{ element.securityContext.capabilities.add || `[]`
}}"
                operator: AnyNotIn
                value:
                - NET_BIND_SERVICE
```

Политика 5: Требовать запуск контейнеров не от root

Обеспечить запуск контейнеров от пользователей, отличных от root:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-run-as-nonroot
  annotations:
    policies.kyverno.io/title: Require Run As Non-Root User
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Containers must run as a non-root user. This policy ensures runAsNo
nRoot is set to true.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: run-as-non-root
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Running as root is not allowed. Either the field spec.securityC
ontext.runAsNonRoot
          must be set to true, or the field spec.containers[*].securityCo
ntext.runAsNonRoot
          must be set to true.
        anyPattern:
        - spec:
            securityContext:
              runAsNonRoot: "true"
        - spec:
            containers:
            - securityContext:
                runAsNonRoot: "true"
```

Расширенные сценарии

Сценарий 1: Политики для разных окружений

Разные уровни безопасности для разных окружений:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: environment-container-security
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Production: Строгая безопасность
    - name: production-strict-security
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - production
            - prod-*
      validate:
        message: "Production environments require strict container securi
ty"
        pattern:
          spec:
            =(hostPID): "false"
            =(hostIPC): "false"
            =(hostNetwork): "false"
            securityContext:
              runAsNonRoot: "true"
            containers:
            - securityContext:
                privileged: "false"
                runAsNonRoot: "true"
                capabilities:
                  drop:
                  - ALL

    # Development: Более разрешительная, но всё ещё безопасная
    - name: development-basic-security
      match:
        any:
        - resources:
            kinds:
            - Pod
```

```
            namespaces:
            - development
            - dev-*
            - staging
      validate:
        message: "Development environments require basic container securi
ty"
        pattern:
          spec:
            =(hostPID): "false"
            =(hostIPC): "false"
            containers:
            - securityContext:
                =(privileged): "false"
```

Сценарий 2: Исключения для конкретных нагрузок

Разрешить определённые нагрузки с контролируемыми исключениями:

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: workload-specific-security
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: system-workloads-exception
      match:
        any:
        - resources:
            kinds:
            - Pod
      exclude:
        any:
        - resources:
            namespaces:
            - kube-system
            - kyverno
        - resources:
            kinds:
            - Pod
            names:
            - "monitoring-*"
            - "logging-*"
      validate:
        message: "Container security policies apply to application worklo
ads"
        pattern:
          spec:
            =(hostNetwork): "false"
            containers:
            - securityContext:
                =(privileged): "false"
```

Тестирование и проверка

Тест привилегированного контейнера

```
# Это должно быть заблокировано
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-privileged
spec:
  containers:
  - name: test
    image: nginx
    securityContext:
      privileged: true
EOF
```

Тест доступа к пространствам имён хоста

```
# Это должно быть заблокировано
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-host-network
spec:
  hostNetwork: true
  containers:
  - name: test
    image: nginx
EOF
```

Тест монтирования путей хоста

```
# Это должно быть заблокировано
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-hostpath
spec:
  containers:
  - name: test
    image: nginx
    volumeMounts:
    - name: host-vol
      mountPath: /host
  volumes:
  - name: host-vol
    hostPath:
      path: /
EOF
```

Тест корректного безопасного контейнера

```
# Это должно быть разрешено
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-secure
spec:
  securityContext:
    runAsNonRoot: true
    runAsUser: 1000
  containers:
  - name: test
    image: nginx
    securityContext:
      allowPrivilegeEscalation: false
      capabilities:
        drop:
        - ALL
      readOnlyRootFilesystem: true
      runAsNonRoot: true
      runAsUser: 1000
EOF
```

Лучшие практики

1. Начинайте с режима аудита

```
spec:
  validationFailureAction: Audit  # Начинайте с предупреждений, а не блок
ировки
```

2. Исключайте системные пространства имён

```
exclude:
  any:
  - resources:
      namespaces:
      - kube-system
      - kyverno
      - kube-public
```

Политика Принудительного Применения Security Context

В этом руководстве показано, как настроить Kyverno для обеспечения правильных security context контейнеров, гарантируя их запуск с соответствующими настройками безопасности и ограничениями.

Содержание

Что такое Принудительное Применение Security Context?

Принудительное применение security context подразумевает контроль за запуском контейнеров путём установки параметров, связанных с безопасностью. Правильная настройка security context предотвращает:

- **Повышение привилегий root**: Запуск контейнеров от имени пользователя root
- **Атаки с повышением привилегий**: Получение контейнерами расширенных прав
- **Небезопасное выполнение процессов**: Запуск контейнеров с опасными возможностями
- **Манипуляции с файловой системой**: Контейнеры с доступной для записи корневой файловой системой
- **Обход механизмов безопасности**: Контейнеры, обходящие меры безопасности

Быстрый старт

1. Политика Требования Запуска Не от Root

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-run-as-nonroot
  annotations:
    policies.kyverno.io/title: Require Run As Non-Root User
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Контейнеры должны запускаться не от root-пользователя. Эта политика гарантирует, что runAsNonRoot установлен в true.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: run-as-non-root
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Запуск от root не разрешён. Либо поле spec.securityContext.runAsNonRoot
          должно быть установлено в true, либо поле spec.containers[*].securityContext.runAsNonRoot
          должно быть установлено в true.
        anyPattern:
        - spec:
            securityContext:
              runAsNonRoot: "true"
        - spec:
            containers:
            - securityContext:
                runAsNonRoot: "true"
```

2. Тестирование Политики

```
# Применить политику
kubectl apply -f require-run-as-nonroot.yaml

# Попытка создать контейнер, явно запускающийся от root (должно провалить
ся)
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-root
spec:
  containers:
  - name: nginx
    image: nginx
    securityContext:
      runAsUser: 0
      runAsNonRoot: false
EOF

# Попытка создать контейнер с не-root пользователем (должно сработать)
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-nonroot
spec:
  securityContext:
    runAsNonRoot: true
    runAsUser: 1000
  containers:
  - name: nginx
    image: nginx
EOF

# Очистка
kubectl delete pod test-root test-nonroot --ignore-not-found
```

Основные Политики Security Context

Политика 1: Запрет Повышения Привилегий

Запретить контейнерам повышать привилегии:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-privilege-escalation
  annotations:
    policies.kyverno.io/title: Disallow Privilege Escalation
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Повышение привилегий, например через set-user-ID или set-group-ID,
не должно быть разрешено.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: privilege-escalation
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Повышение привилегий запрещено. Поля
          spec.containers[*].securityContext.allowPrivilegeEscalation,
          spec.initContainers[*].securityContext.allowPrivilegeEscalatio
n,
          и spec.ephemeralContainers[*].securityContext.allowPrivilegeEsc
alation
          должны быть установлены в false.
        pattern:
          spec:
            =(ephemeralContainers):
              - securityContext:
                  allowPrivilegeEscalation: "false"
            =(initContainers):
              - securityContext:
                  allowPrivilegeEscalation: "false"
            containers:
              - securityContext:
                  allowPrivilegeEscalation: "false"
```

Политика 2: Требование Диапазона User ID

Обеспечить запуск контейнеров с определённым диапазоном user ID:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-user-id-range
  annotations:
    policies.kyverno.io/title: Require User ID Range
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Контейнеры должны запускаться с user ID в определённом диапазоне дл
я предотвращения повышения привилегий.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: user-id-range
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Контейнеры должны запускаться с user ID в диапазоне от 1000 до
65535.
        deny:
          conditions:
            any:
            # Проверка securityContext на уровне pod
            - key: "{{ request.object.spec.securityContext.runAsUser || 0
}}"
              operator: LessThan
              value: 1000
            - key: "{{ request.object.spec.securityContext.runAsUser || 0
}}"
              operator: GreaterThan
              value: 65535
            # Проверка securityContext на уровне контейнеров
            - key: "{{ request.object.spec.containers[?securityContext.ru
nAsUser && (securityContext.runAsUser < `1000` || securityContext.runAsUs
er > `65535`)] | length(@) }}"
              operator: GreaterThan
```

```
        value: 0
```

Политика 3: Требование Не-Root Групп

Обеспечить запуск контейнеров с не-root group ID:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-non-root-groups
  annotations:
    policies.kyverno.io/title: Require Non-Root Groups
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Контейнеры должны запускаться с не-root group ID или дополнительным
и группами.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: non-root-groups
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Контейнеры должны запускаться с не-root group ID. Либо spec.sec
urityContext.runAsGroup,
          либо spec.containers[*].securityContext.runAsGroup должны быть
установлены и не равны 0.
        deny:
          conditions:
            any:
            # Проверка runAsGroup на уровне pod равен 0
            - key: "{{ request.object.spec.securityContext.runAsGroup ||
0 }}"
              operator: Equals
              value: 0
            # Проверка, если у любого контейнера runAsGroup равен 0
            - key: "{{ request.object.spec.containers[?securityContext.ru
nAsGroup == `0`] | length(@) }}"
              operator: GreaterThan
              value: 0
```

Политика 4: Ограничение Seccomp Профилей

Обеспечить использование безопасных seccomp профилей:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: restrict-seccomp-strict
  annotations:
    policies.kyverno.io/title: Restrict Seccomp (Strict)
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Seccomp профиль должен быть явно установлен в одно из разрешённых з
начений.
      Профиль Unconfined и отсутствие профиля запрещены.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: seccomp-strict
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Использование пользовательских Seccomp профилей запрещено. Поле
          spec.securityContext.seccompProfile.type должно быть установлен
о в RuntimeDefault или Localhost.
        anyPattern:
        - spec:
            securityContext:
              seccompProfile:
                type: RuntimeDefault
        - spec:
            securityContext:
              seccompProfile:
                type: Localhost
        - spec:
            containers:
            - securityContext:
                seccompProfile:
                  type: RuntimeDefault
        - spec:
```

```
containers:
- securityContext:
    seccompProfile:
      type: Localhost
```

Политика 5: Требование Сброса ВСЕХ Возможностей

Обеспечить сброс всех capabilities контейнерами:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-drop-all-capabilities
  annotations:
    policies.kyverno.io/title: Require Drop ALL Capabilities
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Контейнеры должны сбрасывать все capabilities и добавлять только т
е, которые необходимы.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: require-drop-all
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Контейнеры должны сбрасывать ВСЕ capabilities.
        foreach:
        - list: request.object.spec.[ephemeralContainers, initContainers,
containers][]
          deny:
            conditions:
              all:
              - key: ALL
                operator: AnyNotIn
                value: "{{ element.securityContext.capabilities.drop || `
[]` }}"
```

Политика 6: Ограничение AppArmor Профилей

Контроль использования AppArmor профилей:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: restrict-apparmor-profiles
  annotations:
    policies.kyverno.io/title: Restrict AppArmor Profiles
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      На поддерживаемых хостах по умолчанию применяется профиль runtime/default AppArmor.
      Базовая политика должна предотвращать переопределение или отключение профиля по умолчанию.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: apparmor-profiles
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Профиль AppArmor должен быть установлен в runtime/default или пользовательский профиль.
          Профили Unconfined не допускаются.
        pattern:
          metadata:
            =(annotations):
              =(container.apparmor.security.beta.kubernetes.io/*): "!unconfined"
```

Расширенные Сценарии

Сценарий 1: Security Context для Разных Сред

Различные требования безопасности для разных сред:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: environment-security-contexts
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Продакшн: Строгие security context
    - name: production-strict-security
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - production
            - prod-*
      validate:
        message: "В продакшн-средах требуются строгие security context"
        pattern:
          spec:
            securityContext:
              runAsNonRoot: "true"
              runAsUser: "1000-65535"
              runAsGroup: "1000-65535"
              seccompProfile:
                type: RuntimeDefault
            containers:
            - securityContext:
                allowPrivilegeEscalation: "false"
                readOnlyRootFilesystem: "true"
                runAsNonRoot: "true"
                capabilities:
                  drop:
                  - ALL

    # Разработка: Базовые требования безопасности
    - name: development-basic-security
      match:
        any:
        - resources:
            kinds:
```

```
        - Pod
        namespaces:
        - development
        - dev-*
        - staging
    validate:
      message: "В средах разработки требуются базовые security context"
      pattern:
        spec:
          containers:
          - securityContext:
              allowPrivilegeEscalation: "false"
              runAsNonRoot: "true"
```

Сценарий 2: Security Context для Разных Приложений

Различные security context для разных типов приложений:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: application-security-contexts
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Базы данных: Специфичные user/group ID
    - name: database-security-context
      match:
        any:
        - resources:
            kinds:
            - Pod
            selector:
              matchLabels:
                app.type: database
      validate:
        message: "Приложения баз данных должны использовать специфичные s
ecurity context"
        pattern:
          spec:
            securityContext:
              runAsUser: "999"
              runAsGroup: "999"
              fsGroup: "999"
            containers:
            - securityContext:
                runAsNonRoot: "true"
                readOnlyRootFilesystem: "true"

    # Веб-приложения: Стандартный security context
    - name: web-app-security-context
      match:
        any:
        - resources:
            kinds:
            - Pod
            selector:
              matchLabels:
                app.type: web
      validate:
```

```
        message: "Веб-приложения должны использовать стандартные security
context"
        pattern:
          spec:
            containers:
            - securityContext:
                runAsNonRoot: "true"
                allowPrivilegeEscalation: "false"
                capabilities:
                  drop:
                  - ALL
                  add:
                  - NET_BIND_SERVICE
```

Сценарий 3: Пошаговое Применение Security Context

Реализация прогрессивных требований к security context:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: graduated-security-contexts
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Уровень 1: Базовая безопасность (все namespaces)
    - name: basic-security-level
      match:
        any:
        - resources:
            kinds:
            - Pod
      exclude:
        any:
        - resources:
            namespaces:
            - kube-system
            - kyverno
      validate:
        message: "Все контейнеры должны иметь базовые security context"
        pattern:
          spec:
            containers:
            - securityContext:
                allowPrivilegeEscalation: "false"

    # Уровень 2: Усиленная безопасность (чувствительные namespaces)
    - name: enhanced-security-level
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - finance-*
            - hr-*
            - security-*
      validate:
        message: "Чувствительные namespaces требуют усиленных security co
ntext"
```

```
        pattern:
          spec:
            securityContext:
              runAsNonRoot: "true"
            containers:
            - securityContext:
                readOnlyRootFilesystem: "true"
                capabilities:
                  drop:
                  - ALL

    # Уровень 3: Максимальная безопасность (критические namespaces)
    - name: maximum-security-level
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - critical-*
            - payment-*
      validate:
        message: "Критические namespaces требуют максимальных security co
ntext"
        pattern:
          spec:
            securityContext:
              runAsNonRoot: "true"
              runAsUser: "1000-1999"
              runAsGroup: "1000-1999"
              seccompProfile:
                type: RuntimeDefault
            containers:
            - securityContext:
                allowPrivilegeEscalation: "false"
                readOnlyRootFilesystem: "true"
                runAsNonRoot: "true"
                capabilities:
                  drop:
                  - ALL
```

Сценарий 4: Применение Security Context на основе

Namespace

Реализация политики безопасности namespace на основе меток:

```
apiVersion: kyverno.io/v1
kind: Policy
metadata:
  annotations:
    policies.kyverno.io/category: Pod Security Standards
    policies.kyverno.io/description: 'Строго следовать конфигурации Secur
ity Context в образе для автоматического добавления, allowPrivilegeEscala
tion:
      false, capabilities drop ALL,  runAsNonRoot: true, seccompProfile:
RuntimeDefault'
    policies.kyverno.io/minversion: 1.13.0
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/title: Add precise Security Context configuration
  creationTimestamp: "2025-06-04T03:26:54Z"
  generation: 1
  labels:
    velero.io/backup-name: app-backup-20250604111354
    velero.io/restore-name: app-recovery
  name: add-exact-security-context
  namespace: test-1
spec:
  admission: true
  background: true
  emitWarning: false
  rules:
  - match:
      any:
      - resources:
          kinds:
          - Pod
    context:
    - name: namespaceInfo
      apiCall:
        urlPath: "/api/v1/namespaces/{{request.namespace}}"
        jmesPath: "metadata.labels.\"pod-security.kubernetes.io/enforce\"
|| 'restricted'"
    preconditions:
      all:
      - key: "{{ namespaceInfo }}"
        operator: NotEquals
        value: "privileged"
    mutate:
```

```
      foreach:
      - list: request.object.spec.containers
        patchStrategicMerge:
          spec:
            containers:
            - name: '{{ element.name }}'
              securityContext:
                allowPrivilegeEscalation: false
                capabilities:
                  drop:
                  - ALL
                runAsNonRoot: true
                seccompProfile:
                  type: RuntimeDefault
    name: add-container-security-context
    skipBackgroundRequests: true
  - match:
      any:
      - resources:
          kinds:
          - Pod
    context:
    - name: namespaceInfo
      apiCall:
        urlPath: "/api/v1/namespaces/{{request.namespace}}"
        jmesPath: "metadata.labels.\"pod-security.kubernetes.io/enforce\"
|| 'restricted'"
    preconditions:
      all:
      - key: "{{ namespaceInfo }}"
        operator: NotEquals
        value: "privileged"
      - key: '{{ request.object.spec.initContainers || `[]` | length(@)
}}'
        operator: GreaterThan
        value: 0
    mutate:
      foreach:
      - list: request.object.spec.initContainers
        patchStrategicMerge:
          spec:
            initContainers:
            - name: '{{ element.name }}'
              securityContext:
```

```
                allowPrivilegeEscalation: false
                capabilities:
                  drop:
                  - ALL
                runAsNonRoot: true
                seccompProfile:
                  type: RuntimeDefault
    name: add-init-container-security-context
    skipBackgroundRequests: true
  - match:
      any:
      - resources:
          kinds:
          - Pod
    context:
    - name: namespaceInfo
      apiCall:
        urlPath: "/api/v1/namespaces/{{request.namespace}}"
        jmesPath: "metadata.labels.\"pod-security.kubernetes.io/enforce\"
|| 'restricted'"
    preconditions:
      all:
      - key: "{{ namespaceInfo }}"
        operator: NotEquals
        value: "privileged"
      - key: '{{ request.object.spec.ephemeralContainers || `[]` | length
(@) }}'
        operator: GreaterThan
        value: 0
    mutate:
      foreach:
      - list: request.object.spec.ephemeralContainers
        patchStrategicMerge:
          spec:
            ephemeralContainers:
            - name: '{{ element.name }}'
              securityContext:
                allowPrivilegeEscalation: false
                capabilities:
                  drop:
                  - ALL
                runAsNonRoot: true
                seccompProfile:
                  type: RuntimeDefault
```

```
    name: add-ephemeral-container-security-context
    skipBackgroundRequests: true
  validationFailureAction: Audit
```

Если вы не ожидаете, что определённые namespaces будут автоматически получать security context, пожалуйста, добавьте метку для namespace `pod-security.kubernetes.io/enforce: privileged`

Тестирование и Валидация

Тест Root Контейнера (Должен Провалиться)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-root-user
spec:
  containers:
  - name: test
    image: nginx
    securityContext:
      runAsUser: 0
EOF
```

Тест Повышения Привилегий (Должен Провалиться)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-privilege-escalation
spec:
  containers:
  - name: test
    image: nginx
    securityContext:
      allowPrivilegeEscalation: true
EOF
```

Тест Отсутствия Сброса Capabilities (Должен Провалиться)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-missing-drop-all
spec:
  containers:
  - name: test
    image: nginx
    securityContext:
      capabilities:
        add:
        - NET_ADMIN
EOF
```

Тест Корректного Безопасного Контейнера (Должен Пройти)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-secure-context
spec:
  securityContext:
    runAsNonRoot: true
    runAsUser: 1000
    runAsGroup: 1000
    seccompProfile:
      type: RuntimeDefault
  containers:
  - name: test
    image: nginx
    securityContext:
      allowPrivilegeEscalation: false
      readOnlyRootFilesystem: true
      runAsNonRoot: true
      runAsUser: 1000
      capabilities:
        drop:
        - ALL
        add:
        - NET_BIND_SERVICE
EOF
```

Политика сетевой безопасности

В этом руководстве показано, как настроить Kyverno для применения политик сетевой безопасности, которые контролируют доступ контейнеров к сети и предотвращают сетевые атаки.

Содержание

Тест привязки порта хоста (должно не пройти)

Тест сервиса NodePort (должно не пройти)

Тест корректной сетевой конфигурации (должно пройти)

Что такое сетевая безопасность?

Сетевая безопасность включает контроль того, как контейнеры получают доступ и взаимодействуют с сетевыми ресурсами. Правильная сетевая безопасность предотвращает:

- **Доступ к сети хоста**: Контейнеры, получающие доступ к сетевым интерфейсам хоста
- **Повышение привилегий через сеть**: Использование сетевого доступа для получения повышенных прав
- **Сканирование портов и разведка**: Несанкционированные действия по обнаружению сети
- **Латеральное перемещение**: Контейнеры, получающие доступ к нежелательным сетевым ресурсам
- **Экфильтрация данных**: Несанкционированные сетевые коммуникации

Быстрый старт

1. Запретить доступ к сети хоста

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-host-network
  annotations:
    policies.kyverno.io/title: Disallow Host Network
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Доступ к сети хоста позволяет потенциально прослушивать сетевой тра
фик и не должен разрешаться.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-network
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Использование сети хоста запрещено. Поле spec.hostNetwork должн
о быть unset или установлено в false.
        pattern:
          spec:
            =(hostNetwork): "false"
```

2. Тестирование политики

```
# Применить политику
kubectl apply -f disallow-host-network.yaml

# Попытка создать pod с сетью хоста (должно не пройти)
kubectl run test-hostnet --image=nginx --overrides='{"spec":{"hostNetwor
k":true}}'

# Попытка создать обычный pod (должно пройти)
kubectl run test-normal --image=nginx
```

Основные политики сетевой безопасности

Политика 1: Запретить порты хоста

Запретить контейнерам привязываться к портам сети хоста:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-host-ports
  annotations:
    policies.kyverno.io/title: Disallow Host Ports
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Доступ к портам хоста позволяет потенциально прослушивать сетевой т
рафик и не должен
      разрешаться, либо должен быть ограничен известным списком.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-ports-none
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Использование портов хоста запрещено. Поля spec.containers[*].p
orts[*].hostPort,
          spec.initContainers[*].ports[*].hostPort и spec.ephemeralContai
ners[*].ports[*].hostPort
          должны быть unset или установлены в 0.
        pattern:
          spec:
            =(ephemeralContainers):
              - =(ports):
                  - =(hostPort): 0
            =(initContainers):
              - =(ports):
                  - =(hostPort): 0
            containers:
              - =(ports):
                  - =(hostPort): 0
```

Политика 2: Ограничить диапазон портов хоста

Разрешить использование определённых диапазонов портов хоста для контролируемого доступа:

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: restrict-host-port-range
  annotations:
    policies.kyverno.io/title: Restrict Host Port Range
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Порты хоста, если используются, должны находиться в разрешённом диа
пазоне для предотвращения конфликтов и проблем безопасности.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-port-range
      match:
        any:
        - resources:
            kinds:
            - Pod
      preconditions:
        all:
        - key: "{{ request.object.spec.containers[].ports[?hostPort] | le
ngth(@) }}"
          operator: GreaterThan
          value: 0
      validate:
        message: >-
          Порты хоста должны находиться в разрешённом диапазоне 30000-327
67.
        foreach:
        - list: request.object.spec.[ephemeralContainers, initContainers,
containers][].ports[]
          preconditions:
            any:
            - key: "{{ element.hostPort }}"
              operator: GreaterThan
              value: 0
          deny:
            conditions:
              any:
```

```
    - key: "{{ element.hostPort }}"
      operator: LessThan
      value: 30000
    - key: "{{ element.hostPort }}"
      operator: GreaterThan
      value: 32767
```

Политика 3: Требовать NetworkPolicies

Обеспечить наличие у pod’ов связанных NetworkPolicies для контроля трафика:

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-network-policies
  annotations:
    policies.kyverno.io/title: Require Network Policies
    policies.kyverno.io/category: Network Security
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod,NetworkPolicy
    policies.kyverno.io/description: >-
      Pod’ы должны иметь связанные NetworkPolicies для контроля сетевого
трафика.
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: require-netpol
      match:
        any:
        - resources:
            kinds:
            - Pod
      exclude:
        any:
        - resources:
            namespaces:
            - kube-system
            - kyverno
      context:
      - name: netpols
        apiCall:
          urlPath: "/apis/networking.k8s.io/v1/namespaces/{{ request.name
space }}/networkpolicies"
          jmesPath: "items[?spec.podSelector.matchLabels.app == '{{ reque
st.object.metadata.labels.app }}'] | length(@)"
      validate:
        message: >-
          Pod должен иметь связанную NetworkPolicy. Создайте NetworkPolic
y, которая выбирает этот pod.
        deny:
          conditions:
            all:
            - key: "{{ netpols }}"
```

```
      operator: Equals
      value: 0
```

Политика 4: Ограничить типы Service

Контролировать, какие типы сервисов могут создаваться:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: restrict-service-types
  annotations:
    policies.kyverno.io/title: Restrict Service Types
    policies.kyverno.io/category: Network Security
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Service
    policies.kyverno.io/description: >-
      Ограничить типы Service, чтобы предотвратить открытие сервисов во в
нешние сети.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: restrict-nodeport
      match:
        any:
        - resources:
            kinds:
            - Service
      validate:
        message: >-
          Сервисы типа NodePort не разрешены. Используйте ClusterIP или L
oadBalancer.
        pattern:
          spec:
            type: "!NodePort"
    - name: restrict-loadbalancer
      match:
        any:
        - resources:
            kinds:
            - Service
            namespaces:
            - development
            - dev-*
            - staging
      validate:
        message: >-
          Сервисы типа LoadBalancer не разрешены в средах разработки.
        pattern:
```

```
    spec:
      type: "!LoadBalancer"
```

Политика 5: Контроль конфигураций Ingress

Обеспечить безопасную конфигурацию Ingress:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: secure-ingress-configuration
  annotations:
    policies.kyverno.io/title: Secure Ingress Configuration
    policies.kyverno.io/category: Network Security
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Ingress
    policies.kyverno.io/description: >-
      Ресурсы Ingress должны быть настроены безопасно с использованием TL
S и правильных аннотаций.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: require-tls
      match:
        any:
        - resources:
            kinds:
            - Ingress
      validate:
        message: >-
          Ingress должен использовать TLS. Поле spec.tls должно быть указ
ано.
        pattern:
          spec:
            tls:
            - hosts:
              - "*"
    - name: require-security-annotations
      match:
        any:
        - resources:
            kinds:
            - Ingress
      validate:
        message: >-
          Ingress должен иметь аннотации безопасности для SSL redirect и
HSTS.
        pattern:
          metadata:
```

```
    annotations:
      nginx.ingress.kubernetes.io/ssl-redirect: "true"
      nginx.ingress.kubernetes.io/force-ssl-redirect: "true"
```

Политика 6: Ограничить конфигурацию DNS

Контролировать настройки DNS для предотвращения атак на DNS:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: restrict-dns-configuration
  annotations:
    policies.kyverno.io/title: Restrict DNS Configuration
    policies.kyverno.io/category: Network Security
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Ограничить конфигурацию DNS для предотвращения перехвата DNS и экфи
льтрации данных.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: restrict-dns-policy
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Пользовательская политика DNS не разрешена. Используйте только
Default или ClusterFirst.
        pattern:
          spec:
            =(dnsPolicy): "Default | ClusterFirst"
    - name: restrict-custom-dns
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Пользовательская конфигурация DNS не разрешена в продуктивных с
редах.
        pattern:
          spec:
            X(dnsConfig): "null"
```

Расширенные сценарии

Сценарий 1: Сетевые политики для разных сред

Различные ограничения сети для разных сред:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: environment-network-security
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Продакшн: строгий контроль сети
    - name: production-network-restrictions
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - production
            - prod-*
      validate:
        message: "В продакшн-средах требуется строгая сетевая безопасность"
        pattern:
          spec:
            hostNetwork: "false"
            dnsPolicy: "ClusterFirst"
            containers:
            - ports:
              - =(hostPort): 0

    # Разработка: базовая сетевая безопасность
    - name: development-network-restrictions
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - development
            - dev-*
            - staging
      validate:
        message: "В средах разработки требуется базовая сетевая безопасность"
```

```
      pattern:
        spec:
          hostNetwork: "false"
```

Сценарий 2: Сетевые политики для разных типов приложений

Различные сетевые политики для разных типов приложений:

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: application-network-policies
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Приложения базы данных: без внешнего сетевого доступа
    - name: database-network-policy
      match:
        any:
        - resources:
            kinds:
            - Pod
            selector:
              matchLabels:
                app.type: database
      validate:
        message: "Приложения базы данных не могут использовать сеть хоста
или порты хоста"
        pattern:
          spec:
            hostNetwork: "false"
            containers:
            - ports:
              - =(hostPort): 0

    # Веб-приложения: контролируемый доступ к портам
    - name: web-app-network-policy
      match:
        any:
        - resources:
            kinds:
            - Pod
            selector:
              matchLabels:
                app.type: web
      validate:
        message: "Веб-приложения могут использовать только стандартные HT
TP/HTTPS порты"
        foreach:
        - list: request.object.spec.containers[].ports[]
```

```
    deny:
      conditions:
        any:
        - key: "{{ element.containerPort }}"
          operator: AnyNotIn
          value:
          - 80
          - 443
          - 8080
          - 8443
```

Сценарий 3: Принудительное разделение сети

Обеспечить сегментацию сети между разными уровнями:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: network-segmentation-enforcement
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: frontend-backend-separation
      match:
        any:
        - resources:
            kinds:
            - Pod
            selector:
              matchLabels:
                tier: frontend
      validate:
        message: "Frontend pod’ы не могут напрямую обращаться к backend с
ети"
        deny:
          conditions:
            any:
            - key: "{{ request.object.metadata.labels.tier }}"
              operator: Equals
              value: backend
    - name: require-network-labels
      match:
        any:
        - resources:
            kinds:
            - Pod
      exclude:
        any:
        - resources:
            namespaces:
            - kube-system
            - kyverno
      validate:
        message: "Pod’ы должны иметь метки уровня сети для сегментации"
        pattern:
          metadata:
            labels:
```

```
      tier: "frontend | backend | database"
```

Тестирование и проверка

Тест доступа к сети хоста (должно не пройти)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-host-network
spec:
  hostNetwork: true
  containers:
  - name: test
    image: nginx
EOF
```

Тест привязки порта хоста (должно не пройти)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-host-port
spec:
  containers:
  - name: test
    image: nginx
    ports:
    - containerPort: 80
      hostPort: 8080
EOF
```

Тест сервиса NodePort (должно не пройти)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Service
metadata:
  name: test-nodeport
spec:
  type: NodePort
  ports:
  - port: 80
    targetPort: 80
    nodePort: 30080
  selector:
    app: test
EOF
```

Тест корректной сетевой конфигурации (должно пройти)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-secure-network
  labels:
    app: web-app
    tier: frontend
spec:
  dnsPolicy: ClusterFirst
  containers:
  - name: test
    image: nginx
    ports:
    - containerPort: 80
      protocol: TCP
---
apiVersion: v1
kind: Service
metadata:
  name: test-service
spec:
  type: ClusterIP
  ports:
  - port: 80
    targetPort: 80
  selector:
    app: web-app
EOF
```

Политика безопасности томов

В этом руководстве показано, как настроить Kyverno для применения политик безопасности томов, которые ограничивают использование опасных типов томов и конфигураций, способных скомпрометировать безопасность контейнеров.

Содержание

Что такое безопасность томов?

Безопасность томов включает контроль над тем, какие типы томов могут монтироваться в контейнеры и каким образом к ним осуществляется доступ. Правильная безопасность томов предотвращает:

- **Доступ к файловой системе хоста**: Несанкционированный доступ к директориям хоста
- **Повышение привилегий**: Использование томов для получения повышенных прав
- **Экфильтрация данных**: Доступ к конфиденциальным данным хоста через монтирование томов
- **Выход из контейнера**: Нарушение изоляции контейнера через доступ к томам
- **Небезопасные типы томов**: Использование типов томов, обходящих механизмы безопасности

Быстрый старт

1. Ограничение типов томов

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: restrict-volume-types
  annotations:
    policies.kyverno.io/title: Restrict Volume Types
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod,Volume
    policies.kyverno.io/description: >-
      Only allow safe volume types. This policy restricts volumes to conf
igMap, csi,
      downwardAPI, emptyDir, ephemeral, persistentVolumeClaim, projected,
and secret.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: restrict-volume-types
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Only the following types of volumes may be used: configMap, cs
i, downwardAPI,
          emptyDir, ephemeral, persistentVolumeClaim, projected, and secr
et.
        foreach:
        - list: "request.object.spec.volumes || []"
          deny:
            conditions:
              all:
              - key: "{{ element.keys(@) }}"
                operator: AnyNotIn
                value:
                - name
                - configMap
                - csi
                - downwardAPI
                - emptyDir
```

```
    - ephemeral
    - persistentVolumeClaim
    - projected
    - secret
```

2. Тестирование политики

```
# Применить политику
kubectl apply -f restrict-volume-types.yaml

# Попытка создать pod с томом hostPath (должно не пройти)
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-hostpath
spec:
  containers:
  - name: nginx
    image: nginx
    volumeMounts:
    - name: host-vol
      mountPath: /host
  volumes:
  - name: host-vol
    hostPath:
      path: /
EOF

# Создать тестовый ConfigMap
kubectl create configmap test-config --from-literal=key=value

# Попытка создать pod с разрешённым томом (должно пройти)
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-configmap
spec:
  containers:
  - name: nginx
    image: nginx
    volumeMounts:
    - name: config-vol
      mountPath: /config
  volumes:
  - name: config-vol
    configMap:
      name: test-config
EOF
```

```
# Очистка
kubectl delete pod test-hostpath test-configmap --ignore-not-found
kubectl delete configmap test-config --ignore-not-found
```

Основные политики безопасности томов

Политика 1: Запрет томов HostPath

Запретить контейнерам монтировать пути файловой системы хоста:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-host-path
  annotations:
    policies.kyverno.io/title: Disallow Host Path
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod,Volume
    policies.kyverno.io/description: >-
      HostPath volumes let Pods use host directories and volumes in conta
iners.
      Using host resources can be used to access shared data or escalate
privileges
      and should not be allowed.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-path
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          HostPath volumes are forbidden. The field spec.volumes[*].hostP
ath must be unset.
        pattern:
          spec:
            =(volumes):
              - X(hostPath): "null"
```

Политика 2: Ограничение томов HostPath (только для чтения)

Разрешить определённые тома hostPath с доступом только для чтения:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: restrict-host-path-readonly
  annotations:
    policies.kyverno.io/title: Restrict Host Path (Read-Only)
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod,Volume
    policies.kyverno.io/description: >-
      HostPath volumes which are allowed must be read-only and restricted
to specific paths.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-path-readonly
      match:
        any:
        - resources:
            kinds:
            - Pod
      preconditions:
        all:
        - key: "{{ request.object.spec.volumes[?hostPath] | length(@) }}"
          operator: GreaterThan
          value: 0
      validate:
        message: >-
          HostPath volumes must be read-only and limited to allowed path
s.
        foreach:
        - list: "request.object.spec.volumes[?hostPath]"
          deny:
            conditions:
              any:
              # Запрет, если путь не в списке разрешённых
              - key: "{{ element.hostPath.path }}"
                operator: AnyNotIn
                value:
                - "/var/log"
                - "/var/lib/docker/containers"
                - "/proc"
```

```
        - "/sys"
    foreach:
    - list: "request.object.spec.containers[].volumeMounts[?name]"
      deny:
        conditions:
          any:
          # Запрет, если монтирование тома не только для чтения
          - key: "{{ element.readOnly || false }}"
            operator: Equals
            value: false
```

Политика 3: Запрет привилегированных типов томов

Блокировать типы томов, которые могут обходить механизмы безопасности:

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-privileged-volumes
  annotations:
    policies.kyverno.io/title: Disallow Privileged Volume Types
    policies.kyverno.io/category: Pod Security Standards (Baseline)
    policies.kyverno.io/severity: high
    policies.kyverno.io/subject: Pod,Volume
    policies.kyverno.io/description: >-
      Certain volume types are considered privileged and should not be al
lowed.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: disallow-privileged-volumes
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Privileged volume types are not allowed: hostPath, gcePersisten
tDisk,
          awsElasticBlockStore, gitRepo, nfs, iscsi, glusterfs, rbd, flex
Volume,
          cinder, cephFS, flocker, fc, azureFile, azureDisk, vsphereVolum
e, quobyte,
          portworxVolume, scaleIO, storageos.
        foreach:
        - list: "request.object.spec.volumes || []"
          deny:
            conditions:
              any:
              - key: "{{ element.keys(@) }}"
                operator: AnyIn
                value:
                - hostPath
                - gcePersistentDisk
                - awsElasticBlockStore
                - gitRepo
```

```
    - nfs
    - iscsi
    - glusterfs
    - rbd
    - flexVolume
    - cinder
    - cephFS
    - flocker
    - fc
    - azureFile
    - azureDisk
    - vsphereVolume
    - quobyte
    - portworxVolume
    - scaleIO
    - storageos
```

Политика 4: Требовать корневую файловую систему только для чтения

Обеспечить использование контейнерами корневой файловой системы только для чтения:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: require-readonly-rootfs
  annotations:
    policies.kyverno.io/title: Require Read-Only Root Filesystem
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      A read-only root file system helps to enforce an immutable infrastr
ucture strategy;
      the container only needs to write on the mounted volume that persis
ts the state.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: readonly-rootfs
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Root filesystem must be read-only. Set readOnlyRootFilesystem t
o true.
        foreach:
        - list: request.object.spec.[ephemeralContainers, initContainers,
containers][]
          deny:
            conditions:
              any:
              - key: "{{ element.securityContext.readOnlyRootFilesystem |
| false }}"
                operator: Equals
                value: false
```

Политика 5: Контроль разрешений монтирования томов

Ограничить разрешения и пути монтирования томов:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: control-volume-mounts
  annotations:
    policies.kyverno.io/title: Control Volume Mount Permissions
    policies.kyverno.io/category: Pod Security Standards (Restricted)
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod,Volume
    policies.kyverno.io/description: >-
      Control where volumes can be mounted and with what permissions.
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: restrict-mount-paths
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Volume mounts to sensitive paths are not allowed.
        foreach:
        - list: request.object.spec.[ephemeralContainers, initContainers,
containers][].volumeMounts[]
          deny:
            conditions:
              any:
              # Блокировать монтирование в чувствительные системные пути
              - key: "{{ element.mountPath }}"
                operator: AnyIn
                value:
                - "/etc"
                - "/root"
                - "/var/run/docker.sock"
                - "/var/lib/kubelet"
                - "/var/lib/docker"
                - "/usr/bin"
                - "/usr/sbin"
                - "/sbin"
                - "/bin"
```

```
    - name: require-readonly-sensitive-mounts
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: >-
          Mounts to /proc and /sys must be read-only.
        foreach:
        - list: request.object.spec.[ephemeralContainers, initContainers,
containers][].volumeMounts[]
          preconditions:
            any:
            - key: "{{ element.mountPath }}"
              operator: AnyIn
              value:
              - "/proc"
              - "/sys"
          deny:
            conditions:
              any:
              - key: "{{ element.readOnly || false }}"
                operator: Equals
                value: false
```

Расширенные сценарии

Сценарий 1: Политики томов для разных окружений

Различные ограничения томов для разных окружений:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: environment-volume-security
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Production: Строгий контроль томов
    - name: production-volume-restrictions
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - production
            - prod-*
      validate:
        message: "Production environments allow only secure volume types"
        foreach:
        - list: "request.object.spec.volumes || []"
          deny:
            conditions:
              all:
              - key: "{{ element.keys(@) }}"
                operator: AnyNotIn
                value:
                - name
                - configMap
                - secret
                - persistentVolumeClaim
                - emptyDir

    # Development: Более либеральные, но безопасные
    - name: development-volume-restrictions
      match:
        any:
        - resources:
            kinds:
            - Pod
            namespaces:
            - development
```

```
        - dev-*
        - staging
    validate:
      message: "Development environments allow additional volume types"
      foreach:
      - list: "request.object.spec.volumes || []"
        deny:
          conditions:
            any:
            - key: "{{ element.keys(@) }}"
              operator: AnyIn
              value:
              - hostPath  # Всё ещё блокировать hostPath в dev
              - nfs       # Блокировать сетевые файловые системы
```

Сценарий 2: Политики томов для разных типов приложений

Различные политики томов для разных типов приложений:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: application-volume-policies
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    # Приложения баз данных: Разрешить постоянное хранилище
    - name: database-volume-policy
      match:
        any:
        - resources:
            kinds:
            - Pod
            selector:
              matchLabels:
                app.type: database
      validate:
        message: "Database applications must use persistent volumes"
        pattern:
          spec:
            volumes:
            - persistentVolumeClaim: {}

    # Веб-приложения: Ограничить безопасными томами
    - name: web-app-volume-policy
      match:
        any:
        - resources:
            kinds:
            - Pod
            selector:
              matchLabels:
                app.type: web
      validate:
        message: "Web applications can only use safe volume types"
        foreach:
        - list: "request.object.spec.volumes || []"
          deny:
            conditions:
              all:
              - key: "{{ element.keys(@) }}"
```

```
    operator: AnyNotIn
    value:
    - name
    - configMap
    - secret
    - emptyDir
    - projected
```

Сценарий 3: Ограничения по размеру томов и ресурсам

Контроль размеров томов и использования ресурсов:

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: volume-resource-limits
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: limit-emptydir-size
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: "EmptyDir volumes must have size limits"
        foreach:
        - list: "request.object.spec.volumes[?emptyDir]"
          deny:
            conditions:
              any:
              - key: "{{ element.emptyDir.sizeLimit || '' }}"
                operator: Equals
                value: ""
    - name: limit-emptydir-memory
      match:
        any:
        - resources:
            kinds:
            - Pod
      validate:
        message: "EmptyDir memory volumes are not allowed"
        foreach:
        - list: "request.object.spec.volumes[?emptyDir]"
          deny:
            conditions:
              any:
              - key: "{{ element.emptyDir.medium || '' }}"
                operator: Equals
                value: "Memory"
```

Тестирование и проверка

Тест тома HostPath (должен не пройти)

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: test-hostpath
spec:
  containers:
  - name: nginx
    image: nginx
    volumeMounts:
    - name: host-vol
      mountPath: /host
  volumes:
  - name: host-vol
    hostPath:
      path: /
EOF
```

API Refiner

Введение

Введение в продукт

Ограничения

Установка Alauda Container Pl

Установка через консоль

Установка через YAML

Процедура удаления

Конфигурация по умолчанию

Обновлени

Матрица совме

Рекомендации

Введение

Содержание

Введение в продукт

ACP API Refiner — это сервис фильтрации данных, предоставляемый платформой Alauda Container Platform, который повышает безопасность мультиарендности и изоляцию данных в Kubernetes-средах. Он фильтрует данные ответов Kubernetes API на основе прав пользователя, проектов, кластеров и пространств имён, а также поддерживает фильтрацию на уровне полей, включение и десенситизацию данных.

Ограничения

К ACP API Refiner применяются следующие ограничения:

- Ресурсы должны содержать определённые метки, связанные с арендатором, для изоляции данных:
  - `cpaas.io/project`
  - `cpaas.io/cluster`

  - `cpaas.io/namespace`
  - `kubernetes.io/metadata.name`
  - Необязательно: `cpaas.io/creator`
- Запросы LabelSelector не поддерживают логические операции OR
- Привязки пользователей на уровне платформы не фильтруются
- Фильтрация применяется только к операциям API GET и LIST

Установка Alauda Container Platform API Refiner

Alauda Container Platform API Refiner — это сервис платформы, который фильтрует данные ответов Kubernetes API. Он предоставляет возможности фильтрации по проекту, кластеру и namespace, а также поддерживает исключение, включение и десенситизацию полей в ответах API.

Содержание

Установка через консоль

1. Перейдите в раздел **Administrator**

2. В левой навигационной панели выберите **Marketplace** > **Cluster Plugins**
3. В верхней панели навигации выберите кластер **global**
4. Найдите **Alauda Container Platform API Refiner** и кликните для просмотра деталей
5. Нажмите **Install** для развертывания плагина

Установка через YAML

1. Проверка доступных версий

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

```
# kubectl get moduleplugins apirefiner
NAME       AGE
apirefiner   4d20h

# kubectl get moduleconfigs -l cpaas.io/module-name=apirefiner
NAME              AGE
apirefiner-v4.0.4   4d21h
```

Это означает, что ModulePlugin `apirefiner` существует в кластере, а версия `v4.0.4` 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: apirefiner
    cpaas.io/module-name: '{"en": "Alauda Container Platform API Refine
r", "zh": "Alauda
      Container Platform API Refiner"}'
  labels:
    cpaas.io/cluster-name: global
    cpaas.io/module-name: apirefiner
    cpaas.io/module-type: plugin
    cpaas.io/product: Platform-Center
  name: apirefiner-global
spec:
  version: v4.2.0-default.1.g8f0543e4
```

Объяснение полей:

- `name`: Временное имя для кластерного плагина. Платформа переименует его после создания на основе содержимого в формате `<cluster-name>-<hash содержимого>`, например, `global-ee98c9991ea1464aaa8054bdacbab313`.
- `label cpaas.io/cluster-name`: API Refiner можно установить только в кластере `global`, оставьте это поле равным global.
- `label cpaas.io/module-name`: Имя плагина, должно совпадать с ресурсом ModulePlugin.
- `label cpaas.io/module-type`: Фиксированное поле, должно быть `plugin`; отсутствие этого поля приведёт к ошибке установки.
- `.spec.config`: Если соответствующий ModuleConfig пуст, это поле можно оставить пустым.
- `.spec.version`: Указывает версию плагина для установки, должна совпадать с `.spec.version` в ModuleConfig.

Процедура удаления

1. Выполните шаги 1-4 из процесса установки для поиска плагина

2. Нажмите **Uninstall** для удаления плагина

Конфигурация по умолчанию

Фильтруемые ресурсы

По умолчанию фильтруются следующие ресурсы:

Ресурс	API Version
namespaces	v1
projects	auth.alauda.io/v1
clustermodules	cluster.alauda.io/v1alpha2
clusters	clusterregistry.k8s.io/v1alpha1

Десенситизация полей

По умолчанию десенситизируется следующее поле:

- `metadata.annotations.cpaas.io/creator`

Обновление

INFO

В этом документе описаны принципы пути обновления и поддерживаемая совместимость версий для Alauda Container Platform API Refiner.

Содержание

Матрица совместимости

В таблице ниже перечислены поддерживаемые версии Alauda Container Platform API Refiner:

API Refiner Version	Supported Alauda Container Platform Version
4.3.x	4.2.x, 4.3.x

API Refiner Version	Supported Alauda Container Platform Version
4.2.x	4.2.x

Рекомендации по пути обновления

Обновление вместе с ACP

- **Description**: В выпусках ACP 4.1.x и более ранних версиях плагины API Refiner были согласованы по жизненному циклу с ACP и обновлялись как часть выпуска ACP. Для следующих путей обновления плагины обновляются вместе с ACP до версии 4.3.0 в процессе обновления ACP.
- **Supported ACP Upgrade Paths**:
  - `4.0.x -> 4.3.0`
  - `4.1.x -> 4.3.0`
- **Result**: После завершения обновления все плагины API Refiner будут обновлены до версии 4.3.0.

Обновление minor-версии

- **Description**: Плагины API Refiner поддерживают обновление minor-версии, если ACP находится на совместимой версии.
- **Prerequisite**: Версия ACP должна соответствовать приведенной выше матрице совместимости.
- **Example**: `4.2.x -> 4.3.x`

Обновление на уровне patch-версии

- **Description**: Обновления между любыми patch-версиями в пределах одной minor-версии полностью совместимы и могут выполняться напрямую.
- **Example**: `4.3.0 -> 4.3.x`

About Alauda Container Platform Compliance Service

Compliance Service is a platform module designed to support STIG compliance scanning and MicroOS operating system scanning. It provides out-of-the-box compliance scanning capabilities with support for scheduled scanning and comprehensive reporting.

Note

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

Конфигурации безопасности платформы

Изменение OIDC Client Secret платформы

ACP использует OIDC Client Secret для аутентификации между компонентами платформы. Может потребоваться выполнить ротацию этого секретного значения в целях соответствия требованиям безопасности, периодической ротации учетных данных или после возможной утечки учетных данных. После обновления секрета в global cluster платформа автоматически синхронизирует его со всеми member clusters.

Содержание

Предварительные требования

- Доступ к `kubectl` в **global cluster** с привилегиями cluster-admin.
- Все кластеры, управляемые платформой, должны быть обновлены до ACP v4.3.

- Следующие плагины с жизненным циклом `Agnostic` должны быть обновлены до версий, совместимых с ACP v4.3:
  - `Alauda Container Platform Gitops`
  - `Alauda Build of Kiali`
  - `Kubeflow Base`
  - `Alauda AI`
- Подготовлено новое значение client secret. Рекомендуется использовать криптографически случайную строку длиной не менее 32 символов.

> **WARNING**
>
> Изменение Client Secret приведет к перезапуску затронутых компонентов, что вызовет временные перебои в аутентификации. Планируйте эту операцию на окно технического обслуживания.

Обзор OIDC Secret

Учетные данные OIDC платформы хранятся в Kubernetes Secret со следующими параметрами:

Свойство	Значение
Secret Name	`cpaas-oidc-secret`
Namespace	`cpaas-system`

Secret содержит два ключа данных:

- `client-id` — идентификатор OIDC client. Значение по умолчанию — `alauda-auth`. **Не изменяйте это значение.**
- `client-secret` — OIDC client secret. Это значение вы будете ротировать.

Процедура

1 Создание резервной копии текущего secret

Перед внесением изменений создайте резервную копию манифеста текущего Secret, чтобы при необходимости можно было выполнить откат.

```
kubectl get secret cpaas-oidc-secret -n cpaas-system -o yaml > cpaas-oidc-secret-backup.yaml
```

> **WARNING**
>
> Резервная копия содержит конфиденциальные данные учетных данных. Храните ее в защищенном месте и удалите после завершения операции.

Зафиксируйте текущий хэш `client-secret` для проверки без использования открытого текста после обновления:

```
OLD_SECRET_SHA256="$(kubectl get secret cpaas-oidc-secret -n cpaas-system \
  -o jsonpath='{.data.client-secret}' | base64 -d | openssl dgst -sha256 -r | awk '{print $1}')"
echo "Current client-secret SHA256: ${OLD_SECRET_SHA256}"
```

2 Обновление Client Secret

> **DANGER**
>
> Не изменяйте значение `client-id` и не удаляйте Secret с последующим созданием заново. Это может привести к каскадным сбоям аутентификации во всей платформе.

Используйте один из следующих способов, чтобы обновить только поле данных `client-secret`.

Способ 1: использование `kubectl patch` (рекомендуется)

Считайте новый secret из защищенного интерактивного ввода:

```
read -rs NEW_CLIENT_SECRET
echo
kubectl patch secret cpaas-oidc-secret -n cpaas-system \
  --type merge \
  -p "{\"data\":{\"client-secret\":\"$(printf %s "$NEW_CLIENT_SECRET"
| base64 | tr -d '\n')\"}}"
unset NEW_CLIENT_SECRET
```

Способ 2: использование `kubectl edit`

Откройте Secret в редакторе по умолчанию и замените значение `client-secret` на новое значение, закодированное в base64.

```
kubectl edit secret cpaas-oidc-secret -n cpaas-system
```

TIP

Вы можете заранее сгенерировать значение в base64-кодировке:

```
read -rs NEW_CLIENT_SECRET
echo
printf %s "$NEW_CLIENT_SECRET" | base64
unset NEW_CLIENT_SECRET
```

Убедитесь, что Secret был обновлен в global cluster, не раскрывая открытый текст:

```
NEW_SECRET_SHA256="$(kubectl get secret cpaas-oidc-secret -n cpaas-sy
stem \
  -o jsonpath='{.data.client-secret}' | base64 -d | openssl dgst -sha
256 -r | awk '{print $1}')"
echo "Updated client-secret SHA256: ${NEW_SECRET_SHA256}"
test "${OLD_SECRET_SHA256}" != "${NEW_SECRET_SHA256}" && \
  echo "Verification passed: client-secret has changed."
unset OLD_SECRET_SHA256 NEW_SECRET_SHA256
```

3 Проверка состояния компонентов

После обновления Secret `base-operator` автоматически синхронизирует его со всеми member clusters. Некоторые компоненты, зависящие от OIDC credential, будут перезапущены для загрузки нового значения. В частности, убедитесь, что deployments `frontend` и `apollo` работают нормально:

```
kubectl get deploy -n cpaas-system frontend apollo
```

Убедитесь, что все Pod находятся в состоянии `READY` и `UP-TO-DATE`. Завершение перезапуска может занять несколько минут.

> **INFO**
>
> Если компонент не запускается, убедитесь, что Secret `cpaas-oidc-secret` существует в namespace `cpaas-system` и содержит корректные значения `client-id` и `client-secret`.

Откат

Если после изменения Secret возникнут проблемы, восстановите предыдущее значение из резервной копии, созданной на Step 1:

```
kubectl apply -f cpaas-oidc-secret-backup.yaml
```

В качестве альтернативы, если вам известно предыдущее значение secret:

```
read -rs PREVIOUS_CLIENT_SECRET
echo
kubectl patch secret cpaas-oidc-secret -n cpaas-system \
  --type merge \
  -p "{\"data\":{\"client-secret\":\"$(printf %s "$PREVIOUS_CLIENT_SECRE
T" | base64 | tr -d '\n')\"}}"
unset PREVIOUS_CLIENT_SECRET
```

После выполнения отката повторите verification steps, чтобы убедиться, что все компоненты работают нормально.

Отключение метода PKCE Plain

Начиная с ACP v4.3.0, проверка PKCE (Proof Key for Code Exchange) включена по умолчанию для авторизации входа. Для обеспечения обратной совместимости с затронутыми плагинами, использующими жизненный цикл `Agnostic` и еще не обновленными, платформа сохраняет метод `plain` кодового вызова наряду с безопасным методом `S256` после обновления.

После того как все затронутые плагины `Agnostic` будут обновлены до версий, совместимых с ACP v4.3.0, следует удалить метод `plain`, чтобы включить проверку PKCE только по `S256`.

Содержание

Предварительные требования

- Доступ `kubectl` к **глобальному кластеру** с привилегиями cluster-admin.
- Все кластеры, управляемые платформой, обновлены до ACP v4.3.

- Следующие затронутые плагины с жизненным циклом `Agnostic` обновлены до версий, совместимых с ACP v4.3:
  - `Alauda Container Platform Gitops`
  - `Alauda Build of Kiali`
  - `Kubeflow Base`
  - `Alauda AI`

> **WARNING**
>
> В данном контексте затронутыми плагинами являются перечисленные плагины `Agnostic`, которые используют поток авторизации платформы OIDC; удаление `plain` до того, как все они будут обновлены до версий, совместимых с ACP v4.3, приведет к сбоям аутентификации.

Процедура

Конфигурация клиента OIDC хранится как пользовательский ресурс `OAuth2Client` в пространстве имен `cpaas-system`. Определите имя целевого ресурса по client ID:

```
OIDC_CLIENT_NAME="$(kubectl get oauth2client -n cpaas-system \
  -o jsonpath='{range .items[?(@.id=="alauda-auth")]}{.metadata.name}
{"\n"}{end}' | head -n 1)"
test -n "${OIDC_CLIENT_NAME}" || { echo "Failed to find OAuth2Client for
id=alauda-auth"; exit 1; }
echo "Target OAuth2Client: ${OIDC_CLIENT_NAME}"
```

Отредактируйте ресурс `OAuth2Client`:

```
kubectl edit oauth2client "${OIDC_CLIENT_NAME}" -n cpaas-system
```

Найдите поле `codeChallengeMethods` и удалите запись `plain`, оставив только `S256`:

```
# Before
codeChallengeMethods:
- S256
- plain

# After
codeChallengeMethods:
- S256
```

Сохраните изменения и выйдите из редактора. Изменение вступает в силу немедленно.

Проверка

Убедитесь, что метод `plain` был удален:

```
kubectl get oauth2client "${OIDC_CLIENT_NAME}" -n cpaas-system \
  -o jsonpath='{.codeChallengeMethods}'
```

Вывод должен быть следующим:

```
["S256"]
```

Выполните как минимум одну проверку входа/авторизации для каждого затронутого плагина (`Alauda Container Platform Gitops`, `Alauda Build of Kiali`, `Kubeflow Base` и `Alauda AI`), чтобы убедиться, что отсутствуют сбои аутентификации, связанные с PKCE.

Откат

Если после удаления метода `plain` возникнут проблемы с аутентификацией (например, сбой входа или авторизации для любого затронутого плагина), добавьте его обратно:

```
kubectl patch oauth2client "${OIDC_CLIENT_NAME}" -n cpaas-system \
  --type merge \
  -p '{"codeChallengeMethods":["S256","plain"]}'
```

После отката еще раз проверьте входы в затронутые плагины, а затем выполните `unset OIDC_CLIENT_NAME`.

Пользователи и роли

Пользователь

Введение

- Источники пользователей
- Правила управления пользователями
- Представления назначения ролей
- Жизненный цикл пользователя

Руководства

Группа

Введение

- Введение в группы
- Типы групп

Руководства

Роль

Введение

Введение в роли

Системные роли

Пользовательские разрешения

Руководства

IDP

Введение

Overview

Supported Integration Methods

Руководства

Устранение

Пользовательская политика

Введение

Overview

Configure Security Policy

Available Policies

Пользователь

Введение

Введение

Источники пользователей

Правила управления пользователями

Представления назначения ролей

Жизненный цикл пользователя

Руководства

Управление ролями пользова

Назначение Platform Roles (системные

Привязка Kubernetes Roles (RoleBinding

Создание пользователя

Создание пользователя через консоль

Создание пользователя через YAML

Управлени

Сброс пароля

Обновление да

Активация пол

Отключение по

Добавление по

Просмотр роле

Удаление поль

Пакетные опер

Введение

Платформа поддерживает аутентификацию пользователей и проверку входа для всех пользователей.

Содержание

Источники пользователей

Локальные пользователи

- Администраторский аккаунт, созданный при развертывании платформы

- Аккаунты, созданные через интерфейс платформы
- Пользователи, добавленные через локальный конфигурационный файл dex

Пользователи третьих сторон

LDAP-пользователи

- Корпоративные пользователи, синхронизированные с LDAP-серверов
- Аккаунты импортируются через интеграцию с IDP (Identity Provider)
- Источник отображается как имя конфигурации IDP
- Интеграция настраивается через параметры IDP

OIDC-пользователи

- Пользователи сторонних платформ, аутентифицированные через протокол OIDC
- Источник отображается как имя конфигурации IDP
- Интеграция настраивается через параметры IDP

> **WARNING**
>
> Для OIDC-пользователей, добавленных в проект до их первого входа:
>
> - Источник отображается как "-" до успешного входа в платформу
> - После успешного входа источник меняется на имя конфигурации IDP

Другие пользователи третьих сторон

- Пользователи, аутентифицированные через поддерживаемые коннекторы dex (например, GitHub, Microsoft)
- Для дополнительной информации см. официальную документацию dex↗

Правила управления пользователями

WARNING

Обратите внимание на следующие важные правила:

- Локальные имена пользователей должны быть уникальны среди всех типов пользователей
- Пользователи третьих сторон (OIDC/LDAP) с совпадающими именами автоматически связываются
- Связанные пользователи наследуют права от существующих аккаунтов
- Пользователи могут входить через свои соответствующие источники
- В платформе отображается только одна запись пользователя на имя пользователя
- Источник пользователя определяется по последнему способу входа

Представления назначения ролей

На каждой странице с деталями пользователя теперь есть две отдельные вкладки для просмотра и управления ролями:

- **Platform Roles** (только для чтения): Список системных ролей платформы/проекта/пространства имён, связанных с пользователем. Эти роли нельзя редактировать или дублировать в UI, но при необходимости их можно отвязать.
- **Kubernetes Roles**: Отображает все объекты RoleBinding/ClusterRoleBinding, ссылающиеся на пользователя (во всех кластерах). Администраторы могут создавать или удалять привязки здесь для предоставления нативных прав Kubernetes.

Используйте вкладку Platform Roles для шаблонов доступа по умолчанию, а вкладку Kubernetes Roles — когда требуется тонкая настройка через нативные роли.

Жизненный цикл пользователя

В следующей таблице описаны различные статусы пользователей на платформе:

Статус	Описание
Normal	Аккаунт пользователя активен и может войти в платформу
Disabled	Аккаунт пользователя неактивен и не может войти. Обратитесь к администратору платформы для активации. Возможные причины: - Отсутствие входа более 90 дней подряд - Истечение срока действия аккаунта - Ручное отключение администратором
Locked	Аккаунт временно заблокирован из-за 5 неудачных попыток входа в течение 24 часов. Детали: - Длительность блокировки: 20 минут - Может быть разблокирован вручную администратором - Аккаунт становится доступен после окончания блокировки
Invalid	Аккаунт, синхронизированный с LDAP, который был удалён с LDAP-сервера. Примечание: недействительные аккаунты не могут войти в платформу

Руководства

Управление ролями пользова

Назначение Platform Roles (системные
Привязка Kubernetes Roles (RoleBinding

Создание пользователя

Создание пользователя через консоль
Создание пользователя через YAML

Управлени

Сброс пароля
Обновление да
Активация пол
Отключение по
Добавление по
Просмотр роле
Удаление поль
Пакетные опер

Управление ролями пользователей

Администраторы платформы могут управлять ролями других пользователей (не своей собственной учетной записи), чтобы предоставлять или отзывать разрешения. После рефакторинга RBAC назначение ролей разделено на **Platform Roles** (системные шаблоны) и **Kubernetes Roles** (родные объекты RoleBinding).

Содержание

Назначение Platform Roles (системные шаблоны)

Используйте эту вкладку для привязки или отвязки предопределённых ролей платформы/проекта/пространства имён, которые поставляются с продуктом.

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**.
2. Кликните по имени пользователя, для которого нужно изменить роли.
3. Откройте вкладку **Platform Roles**.
4. Нажмите **Add Role**.
5. В диалоговом окне:
   - Выберите роль из выпадающего списка **Role Name**.
   - При необходимости выберите область действия (Cluster / Project / Namespace).
   - Нажмите **Add**.

> **NOTE**
>
> Примечания по ролям платформы:
>
> - Здесь доступны только предопределённые системные роли; их нельзя редактировать или дублировать.
> - Роль может быть привязана только один раз в рамках одной области действия. Уже привязанные роли в списке отключены.
> - Встроенную роль Cluster Administrator нельзя переназначать для глобального кластера.

Удаление Platform Roles

1. Оставайтесь на вкладке **Platform Roles**.
2. Нажмите **Remove** рядом с ролью, которую хотите отвязать.
3. Подтвердите удаление.

Привязка Kubernetes Roles (RoleBinding / ClusterRoleBinding)

Используйте эту вкладку для предоставления детализированных разрешений через родные роли Kubernetes, существующие в конкретных кластерах.

Шаги

1. На странице с деталями пользователя переключитесь на вкладку **Kubernetes Roles**.
2. Нажмите **Add RoleBinding**.
3. Настройте привязку:
   - **Cluster**: Целевой кластер, в котором находится роль.
   - **Binding Type**: `RoleBinding` (область действия — namespace) или `ClusterRoleBinding`.
   - **Namespace**: Обязательно при выборе `RoleBinding`.
   - **Role Name**: Выберите существующий `Role` или `ClusterRole`.
   - **Subject**: Подтвердите текущего пользователя как субъект привязки.
4. Нажмите **Create**.

Удаление Kubernetes RoleBindings

1. Оставайтесь на вкладке **Kubernetes Roles**.
2. Найдите нужную привязку (при необходимости отфильтруйте по кластеру, пространству имён или роли).
3. Нажмите **Remove** и подтвердите.

> **WARNING**
>
> Разрешения на управление ролями:
>
> - Управлять ролями других пользователей могут только администраторы платформы.
> - Пользователи не могут изменять роли или привязки для своей собственной учетной записи.

Создание пользователя

Пользователи с ролями администратора платформы могут создавать локальных пользователей и назначать им роли через интерфейс платформы.

Содержание

Создание пользователя через консоль

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Нажмите **Create User**
3. Настройте следующие параметры:

Параметр	Описание
Password Type	Выберите способ генерации пароля: **Random**: Система генерирует безопасный случайный пароль **Custom**: Пользователь вводит пароль вручную
Password	Введите или сгенерируйте пароль в зависимости от выбранного типа.

Параметр	Описание
	Требования к паролю: - Длина: 8-32 символа - Должен содержать буквы и цифры - Должен содержать специальные символы (`~!@#$%^&*() -_=+?`) Особенности поля пароля: - Нажмите на иконку глаза, чтобы показать/скрыть пароль - Нажмите на иконку копирования, чтобы скопировать пароль
Mailbox	Электронная почта пользователя: - Должна быть уникальной - Может использоваться как имя пользователя для входа - Связана с именем пользователя
Validity Period	Установите период действия учетной записи пользователя: Варианты: - **Permanent**: Без ограничения по времени - **Custom**: Установите время начала и окончания с помощью выпадающего списка Time Range
Roles	Назначьте одну или несколько ролей пользователю
Continue Creating	Переключатель для управления поведением после создания: - **On**: Перенаправляет на страницу создания нового пользователя - **Off**: Отображает страницу с деталями созданного пользователя

4. Нажмите **Create**

NOTE

После успешного создания пользователя:

- Если включен "Continue Creating", вы будете перенаправлены на создание следующего пользователя
- Если выключен, отобразится страница с деталями созданного пользователя

Создание пользователя через YAML

Вы можете отправить следующий YAML в кластере `global` для создания пользователя.

```yaml
apiVersion: auth.alauda.io/v1
kind: User
metadata:
  labels:
    auth.cpaas.io/user.connector_id: ""                    # Connector ID for external authentication (leave empty for local users)
    auth.cpaas.io/user.connector_type: ""                  # Connector type for external authentication (leave empty for local users)
    auth.cpaas.io/user.email: c18c9911faaac4e1051a599b88c62af2  # MD5 hash of the username (spec.email)
    auth.cpaas.io/user.state: active                       # User state; must match spec.state
    auth.cpaas.io/user.username: ""                        # User display name; must match spec.username
    auth.cpaas.io/user.valid: "true"                       # Whether the user is valid; must match spec.valid
  name: c18c9911faaac4e1051a599b88c62af2                   # Name of the User resource; MD5 hash of spec.email
spec:
  connector_name: ""                                       # Name of the external authentication connector (leave empty for local users)
  connector_type: ""                                       # Type of the external authentication connector (leave empty for local users)
  email: leizhuaaa                                         # User identifier; can be an email address or any unique string
  is_admin: false                                          # Whether the user is an initial admin user; must be set to false
  state: active                                            # User account state: active or inactive
  username: ""                                             # Display name for the user
  valid: true                                              # Whether the user account is valid; should be set to true
```

Управление пользователями

Платформа предоставляет гибкие возможности управления пользователями, поддерживая как индивидуальное управление, так и пакетные операции для повышения эффективности в определённых сценариях (например, для выездных или удалённых команд).

WARNING

Важные ограничения:

- Системные аккаунты управлению не подлежат (роль администратора платформы, локальный источник)
- Текущие вошедшие в систему пользователи не могут управлять своими собственными аккаунтами
- Для изменения личных данных (отображаемое имя, пароль) используйте страницу личной информации

Содержание

Сброс пароля локального пользователя

Пользователи с правами управления платформой могут сбрасывать пароли других локальных пользователей.

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Нажмите на иконку рядом с записью нужного пользователя
3. Нажмите **Reset Password**
4. В диалоговом окне выберите тип пароля:
   - **Random**: Система сгенерирует безопасный случайный пароль
   - **Custom**: Введите новый пароль вручную

> **NOTE**
>
> Требования к паролю:
>
> - Длина: 8-32 символа
> - Должен содержать буквы и цифры

- Должен содержать специальные символы (`~!@#$%^&*() -_=+?`)

Особенности поля пароля:

- Нажмите на иконку глаза, чтобы показать/скрыть пароль
- Нажмите на иконку копирования, чтобы скопировать пароль

5. Нажмите **Reset**

Обновление даты истечения срока действия пользователя

Вы можете обновлять даты истечения срока действия для пользователей со статусом **normal**, **disabled** или **locked**. Пользователи, у которых срок действия истёк, будут автоматически отключены.

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Нажмите **Update Expiry Date** рядом с нужным пользователем
3. В диалоговом окне выберите опцию даты истечения срока:
   - **Permanent**: Без ограничения по времени
   - **Custom**: Установите время начала и окончания с помощью выпадающего списка Time Range
4. Нажмите **Update**

Активация пользователя

Вы можете активировать пользователей со статусом **disabled** или **locked**.

NOTE

Поведение при активации:

- Если пользователь находится в пределах срока действия: дата истечения остаётся без изменений
- Если срок действия пользователя истёк: дата истечения становится **Permanent**

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Нажмите **Activate** рядом с нужным пользователем
3. В диалоговом окне подтверждения нажмите **Activate**
4. Статус пользователя изменится на **normal**

Отключение пользователя

Вы можете отключить пользователей со статусом **normal** или **locked** в пределах срока действия. Отключённые пользователи не могут войти в систему, но могут быть повторно активированы.

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Нажмите на иконку рядом с нужным пользователем
3. Нажмите **Disable** и подтвердите

Добавление пользователя в локальную группу пользователей

Вы можете добавить пользователей с **Source** равным **Local** или `LDAP` в одну или несколько локальных групп пользователей.

WARNING

Поведение ролей групп:

- Пользователи автоматически наследуют роли своих групп
- Роли групп видны только на странице деталей группы (вкладка **Platform Roles**)
- Списки ролей отдельных пользователей показывают только напрямую назначенные роли

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Нажмите на иконку рядом с нужным пользователем
3. Нажмите **Add to User Group**
4. Выберите одну или несколько локальных групп пользователей
5. Нажмите **Add**

Просмотр ролей пользователя

После рефакторинга RBAC на странице деталей каждого пользователя отображаются две вкладки:

- **Platform Roles**: Показывает системные роли, привязанные к пользователю. Вы можете добавлять или удалять привязки, но не редактировать определения ролей.
- **Kubernetes Roles**: Перечисляет все RoleBinding/ClusterRoleBinding, ссылающиеся на пользователя в кластерах. Вы можете создавать или удалять привязки прямо на этой вкладке.

Подробные инструкции смотрите в разделе Manage User Roles.

Удаление пользователя

Администраторы платформы могут удалять любых пользователей, кроме текущего вошедшего в систему аккаунта, включая:

- Пользователей, настроенных через IDP
- Пользователей с источником `-`
- Локальных пользователей

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Нажмите на иконку рядом с нужным пользователем
3. Нажмите **Delete**
4. Нажмите **Confirm**

Пакетные операции

Вы можете выполнять пакетные операции для:

- Обновления сроков действия
- Активации пользователей
- Отключения пользователей
- Удаления пользователей

Шаги

1. В левой навигационной панели нажмите **Users** > **User Management**
2. Выберите одного или нескольких пользователей с помощью чекбоксов
3. Нажмите **Batch Operations** и выберите действие:
   - **Update Validity**
   - **Activate**
   - **Deactivate**

- **Delete**

> **NOTE**
>
> Детали пакетных операций:
>
> - **Update Validity**: Установить постоянный или пользовательский временной диапазон
> - **Activate**: Подтвердить активацию в диалоге
> - **Deactivate**: Подтвердить деактивацию в диалоге
> - **Delete**: Ввести пароль текущего аккаунта и подтвердить

Группа

Введение

Введение

Введение в группы

Типы групп

Руководства

Управление ролями групп по

Добавление роли в группу

Удаление роли из группы

Создание локальной пользов

Создание пользовательской группы

Управление пользовательскими группам

Управлени

Предварителы

Импорт участн

Удаление учас

Введение

Содержание

Введение в группы

Платформа поддерживает управление пользователями через пользовательские группы. Управляя ролями групп, вы можете эффективно:

- Предоставлять права на работу с платформой сразу нескольким пользователям
- Одновременно отзывать права у нескольких пользователей
- Реализовывать пакетный контроль доступа на основе ролей

Например, при кадровых изменениях в компании, когда необходимо предоставить права на работу с проектом или namespace сразу нескольким пользователям, вы можете:

1. Создать пользовательскую группу
2. Импортировать соответствующих пользователей в члены группы
3. Настроить роли проекта и namespace для группы

4. Применить единые права ко всем членам группы

Типы групп

Платформа поддерживает два типа групп:

Локальная пользовательская группа

- Создаётся непосредственно на платформе
- Источник отображается как **Local**
- Может быть обновлена или удалена
- Поддерживает:
  - Добавление или удаление пользователей из любого источника
  - Добавление или удаление ролей

Синхронизированная с IDP пользовательская группа

- Синхронизируется из подключённого IDP (LDAP, Azure AD)
- Источник отображается как имя подключённого ***IDP***
- Не может быть обновлена или удалена
- Поддерживает:
  - Добавление или удаление ролей
  - Нельзя управлять членами группы (добавлять или удалять)

Руководства

Управление ролями групп по

Добавление роли в группу

Удаление роли из группы

Создание локальной пользов

Создание пользовательской группы

Управление пользовательскими группам

Управлени

Предварительны

Импорт участн

Удаление учас

Управление ролями групп пользователей

Пользователи с правами управления платформой могут управлять ролями как для локальных групп пользователей, так и для групп пользователей, синхронизированных через IDP.

Содержание

Добавление роли в группу

Шаги

1. В левой навигационной панели нажмите **Users** > **User Group Management**
2. Нажмите на название целевой группы пользователей
3. На вкладке **Configure Role** нажмите **Add Role**
4. Нажмите, чтобы добавить роль

NOTE

Правила назначения ролей:

- В группу можно добавить несколько ролей
- Каждая роль может быть добавлена в одну группу только один раз

5. Выберите название роли из выпадающего списка
6. Выберите область действия роли (кластер, проект или namespace)
7. Нажмите **Add**

Удаление роли из группы

WARNING

При удалении роли из группы:

- Все разрешения, предоставленные этой ролью членам группы, будут отозваны
- Это действие нельзя отменить

Шаги

1. В левой навигационной панели нажмите **Users** > **User Group Management**
2. Нажмите на название целевой группы пользователей
3. На вкладке **Configure Role** нажмите **Remove** рядом с ролью
4. Нажмите **Confirm** для удаления роли

Создание локальной пользовательской группы

Локальные пользовательские группы позволяют реализовать управление доступом на основе ролей для нескольких пользователей из любого источника.

Содержание

Создание пользовательской группы

Шаги

1. В левой боковой панели нажмите **Users** > **User Group Management**
2. Нажмите **Create User Group**
3. Введите следующую информацию:

- **Name**: Название пользовательской группы
- **Description**: Описание назначения группы

4. Нажмите **Create**

Управление пользовательскими группами

Вы можете управлять пользовательскими группами, нажав на иконку на странице списка или нажав **Operations** в правом верхнем углу на странице с деталями.

Операция	Описание
Update User Group	Обновление информации о группе в зависимости от источника группы: - Для групп с **Source** `Local`: можно обновить как название, так и описание - Для групп с **Source** `IDP name`: можно обновить только описание
Delete Local User Group	Удаление пользовательских групп с **Source** `Local`

WARNING

При удалении группы:

- Все участники группы будут удалены
- Все роли, назначенные группе, будут удалены
- Это действие нельзя отменить

Управление членством в локальной группе пользователей

Управлять членством в локальных группах пользователей могут только пользователи с правами Platform Management.

Содержание

Предварительные требования

WARNING

Перед управлением членством в группах обратите внимание на следующие ограничения:

- Управлять группами и их участниками могут только пользователи с правами Platform Management

- Системные аккаунты и аккаунты, в данный момент вошедшие в систему, не могут управляться (импортироваться в группы или удаляться из них)
- В каждой локальной группе пользователей может быть не более 5000 участников
- При достижении лимита в 5000 участников дальнейший импорт невозможен

Импорт участников

Вы можете импортировать пользователей с платформы в локальные группы пользователей для централизованного управления правами доступа.

TIP

Пользователи, импортированные в группу, автоматически наследуют все операционные права, назначенные этой группе.

Шаги

1. В левой навигационной панели нажмите **Users** > **User Group Management**
2. Нажмите на название локальной группы пользователей, в которую хотите добавить участников
3. На вкладке **Group Member Management** нажмите **Import Member**
4. Выберите одного или нескольких пользователей с платформы, отметив галочками их имена пользователей/отображаемые имена
5. Нажмите **Import**

NOTE

- Вы можете выбрать только тех пользователей, которые в данный момент не являются членами группы
- Используйте кнопку **Import All**, чтобы импортировать всех пользователей из списка сразу

Удаление участников

При удалении пользователя из группы все операционные права, предоставленные этому пользователю через группу, будут автоматически отозваны.

Шаги

1. В левой навигационной панели нажмите **Users** > **User Group Management**
2. Нажмите на название локальной группы пользователей, из которой хотите удалить участников
3. На вкладке **Group Member Management** вы можете удалить участников двумя способами:
   - Нажмите **Remove** рядом с именем участника и подтвердите действие
   - Выберите одного или нескольких участников с помощью чекбоксов, затем нажмите **Batch Remove** и подтвердите действие

Роль

Введение

Введение

Введение в роли

Системные роли

Пользовательские разрешения

Руководства

Создание ролей Kubernetes

Предварительные требования

Создание Role или ClusterRole

Редактирование YAML роли (необязате

Создание RoleBindings

Проверка

Управление ролями

Просмотр ролей платформы (только дл

Назначение роли платформы пользоват

Назначение роли платформы пользоват

Просмотр и обновление роли Kubernete

Удаление роли Kubernetes

Управление RoleBindings

Рекомендации по лучшим практикам

Как создат

1. Обзор
2. Основные п
3. Технические
4. Поддержива
5. Конфигурац
6. Копировани

Введение

Содержание

Введение в роли

Управление ролями пользователей на платформе реализовано поверх Kubernetes RBAC (Role-Based Access Control). Начиная с ACP 4.2, модель разделена на два взаимодополняющих слоя:

- **Platform Roles**: Шаблоны, предоставляемые системой, которые гарантируют основные сценарии продукта. В UI они доступны только для чтения; вы можете только привязывать или отвязывать их для пользователей и групп.
- **Kubernetes Roles**: Нативные объекты Kubernetes `Role` и `ClusterRole`, которые можно создавать для каждого кластера, чтобы удовлетворить специфические требования к разрешениям. Управление этими ролями осуществляется на странице **Kubernetes Roles**, а привязка — через RoleBinding/ClusterRoleBinding.

Оба слоя в конечном итоге транслируются в разрешения Kubernetes. Назначение или удаление роли немедленно предоставляет или отзывает связанные операции

(создание, просмотр, обновление, удаление и т.д.) над целевыми ресурсами.

Системные роли

Для удовлетворения распространённых сценариев настройки разрешений платформа предоставляет следующие стандартные системные роли. Эти роли обеспечивают гибкий контроль доступа к ресурсам платформы и эффективное управление разрешениями пользователей.

Название роли	Описание	Уровень роли
Platform Administrator	Имеет полный доступ ко всем бизнес-ресурсам и ресурсам платформы	Platform
Platform Auditors	Может просматривать все ресурсы платформы и записи операций, но не имеет других прав	Platform
Cluster Administrator (Alpha)	Управляет и поддерживает ресурсы кластера с полным доступом ко всем ресурсам уровня кластера	Cluster
Project Administrator	Управляет администраторами namespace и квотами namespace	Project
namespace-admin-system	Управляет участниками namespace и назначениями ролей	Namespace
Developers	Разрабатывает, развёртывает и поддерживает кастомные приложения внутри namespace	Namespace

Пользовательские разрешения

Устаревший опыт создания кастомных ролей с помощью «чекбоксов» был удалён. Для реализации новых сценариев авторизации необходимо:

1. Использовать страницу **Kubernetes Roles** для создания нативных ролей (Role/ClusterRole) в нужном кластере, либо
2. Запросить шаблоны ролей у команды, владеющей соответствующим плагином, и привязать их через RoleBinding/ClusterRoleBinding.

> **WARNING**
>
> Удаление или редактирование нативной роли влияет на все RoleBinding/ClusterRoleBinding, которые на неё ссылаются. Рекомендуется проверить существующие привязки и по возможности протестировать изменения в staging-среде.

Руководства

Создание ролей Kubernetes

Управление ролями

Как создат

Создание ролей Kubernetes

Начиная с ACP 4.2, пользовательские разрешения предоставляются через нативные роли Kubernetes. Используйте страницу **Kubernetes Roles** (расположена в разделе **Users > Platform Roles > Kubernetes Roles**) для создания или управления объектами `Role` и `ClusterRole` в выбранном кластере.

Содержание

Предварительные требования

- Вам назначена платформа роль, которая предоставляет доступ к функции Kubernetes Roles.
- Вы выбрали целевой кластер в глобальном переключателе кластеров.
- В кластере уже существуют пространства имён, к которым будет ограничена ваша роль.

Создание Role или ClusterRole

1. В левой навигационной панели нажмите **Users** > **Platform Roles** > **Kubernetes Roles**.
2. Нажмите **Create Role**.
3. В открывшемся боковом меню:
   - Введите **Name** (должно соответствовать правилам именования Kubernetes).
   - Выберите **Type** (`Role` или `ClusterRole`).
   - Если выбран `Role` , укажите **Namespace**, в рамках которого будут действовать разрешения.
4. Настройте правила, добавив одну или несколько записей с указанием:
   - **API Groups**
   - **Resources**
   - **Resource Names** (необязательно)
   - **Verbs** (`get` , `list` , `watch` , `create` , `update` , `patch` , `delete`)
5. Нажмите **Create**.

Роль создаётся непосредственно в кластере и становится доступной для операций RoleBinding сразу же.

Редактирование YAML роли (необязательно)

1. Откройте вкладку **Kubernetes Roles** и нажмите на имя роли.
2. На вкладке **YAML** нажмите **Edit**.
3. Обновите поля, такие как метки, аннотации или правила.
4. Нажмите **Save** для применения изменений.

Создание RoleBindings

Чтобы назначить созданную роль пользователям или группам:

1. Просматривая роль, переключитесь на вкладку **RoleBindings**.
2. Нажмите **Create RoleBindings**.
3. Укажите:
   - **Name**
   - **Binding Type** (`RoleBinding` или `ClusterRoleBinding` — при исходной роли с областью действия namespace доступен только `RoleBinding`)
   - **Namespace** (для `RoleBinding`)
   - **Subjects** (User, Group или ServiceAccount с соответствующим именем)
4. Нажмите **Create**.

Альтернативно, откройте страницу **Users** или **User Groups**, переключитесь на вкладку **Kubernetes Roles** и создавайте привязки непосредственно с точки зрения пользователя.

Проверка

Используйте один из следующих способов, чтобы убедиться, что роль существует:

```
kubectl get role <role-name> -n <namespace>
kubectl get clusterrole <clusterrole-name>
```

Или обновите список **Kubernetes Roles** и воспользуйтесь встроенным поиском (по имени или метке) для поиска роли.

Управление ролями

Содержание

Просмотр ролей платформы (только для чтения)

Роли платформы остаются каноническими шаблонами для основной функциональности.

1. В левой навигационной панели нажмите **Users** > **Platform Roles**.
2. Используйте фильтры списка для поиска роли. В столбце **Role Type** теперь отображается `Platform`, `Project`, `Namespace` или `Cluster`.

3. Нажмите на имя роли, чтобы открыть страницу с деталями.
4. Перейдите на вкладку **YAML** для просмотра точного определения. Используйте **Download YAML**, если нужно сохранить спецификацию.

Назначение роли платформы пользователям через консоль

1. В левой навигационной панели нажмите **Users** > **Platform Roles**.
2. Нажмите на имя роли, чтобы открыть страницу с деталями.
3. Перейдите на вкладку **Members**.
4. Нажмите **Import Members**.
5. Вы можете выбрать пользователей из платформы и импортировать их в роль в качестве участников.

Назначение роли платформы пользователям через YAML

Вы можете отправить следующий YAML в кластере `global`, чтобы назначить конкретную роль платформы пользователю.

apiVersion: auth.alauda.io/v1 kind: UserBinding metadata: annotations: auth.cpaas.io/role.display-name: Platform Admin # Отображаемое имя назначаемой роли auth.cpaas.io/user.email: bxliu@alauda.io ↗ # Имя пользователя, которому назначается роль labels: auth.cpaas.io/role.display-name: "" # Отображаемое имя назначаемой роли auth.cpaas.io/role.level: platform # Область действия роли: platform, project, namespace или cluster auth.cpaas.io/role.name: acp-platform-admin # Имя назначаемой роли auth.cpaas.io/user.email: 569526aac97a17ce8c1c185d7544aae4 # MD5-хеш имени пользователя cpaas.io/cluster: "" # Имя кластера; требуется, если уровень роли namespace или cluster, оставьте пустым для platform или project cpaas.io/namespace: "" # Имя namespace; требуется, если уровень роли namespace, оставьте пустым для platform, project или cluster cpaas.io/project: "" # Имя проекта; требуется, если уровень

роли project или namespace, оставьте пустым для platform или cluster name: dc30204c17c7fe8b15383f4ed7798c88 # Имя ресурса UserBinding; можно настроить

Просмотр и обновление роли Kubernetes через YAML

1. Перейдите в **Users** > **Platform Roles** > **Kubernetes Roles**.
2. Выполните поиск по имени или метке.
3. Нажмите на имя роли, затем откройте вкладку **YAML**.
4. Нажмите **Edit**, измените манифест (метки, аннотации или `rules`) и нажмите **Save**.
5. Проверьте вкладку **RoleBindings**, чтобы убедиться, что существующие привязки соответствуют вашим ожиданиям.

Удаление роли Kubernetes

1. В списке **Kubernetes Roles** нажмите меню переполнения (…) рядом с ролью.
2. Выберите **Delete Role**.
3. Подтвердите имя роли для продолжения.

Удаление роли удаляет её из кластера. Также необходимо очистить все RoleBindings, которые ссылались на эту роль. В UI будет показано предупреждение, если привязки всё ещё существуют.

Управление RoleBindings

С точки зрения роли

1. Откройте роль (Role или ClusterRole) на вкладке **Kubernetes Roles**.
2. Перейдите на вкладку **RoleBindings**.

3. Используйте строку поиска (поддерживает фильтры по имени и меткам) для поиска существующих привязок.
4. Действия:
   - **Create RoleBindings**: запускает мастер создания.
   - **Update Role**: открывает YAML-редактор для самой роли.
   - **Delete Binding**: удаляет RoleBinding/ClusterRoleBinding после подтверждения.

С точки зрения пользователей или групп пользователей

1. Откройте **Users** (или **User Groups**) и выберите нужную запись.
2. Перейдите на вкладку **Kubernetes Roles**.
3. Просмотрите все RoleBindings, связанные с пользователем/группой по кластерам.
4. Нажмите **Add RoleBinding**, выберите:
   - Кластер
   - Тип привязки (RoleBinding/ClusterRoleBinding)
   - Роль/ClusterRole
   - Namespace (для RoleBinding)
   - Данные субъекта
5. Сохраните привязку.

Этот рабочий процесс дополняет существующую вкладку **Platform Roles**, которая по-прежнему используется для назначения системных ролей пользователям.

Рекомендации по лучшим практикам

- Используйте staging-кластеры для проверки изменений YAML перед применением в production.
- Храните определения ролей под управлением версий (например, экспортируйте их в Git), чтобы изменения были аудируемыми.

- В случае сомнений по необходимым разрешениям начните с YAML системной роли, скопируйте её локально и адаптируйте как роль Kubernetes через новый UI.

Как создать пользовательскую роль платформы

Содержание

1. Обзор

В этом документе объясняются основные концепции пользовательских ролей и как настроить RoleTemplate.

2. Основные понятия

- **RoleTemplate**: Шаблон пользовательской роли. Определяет семантику роли и наборы разрешений, которые контроллер преобразует в ClusterRoles.
- **FunctionResource**: Абстракция ресурсов K8s, используемых функциональностью продукта; ссылается через `functionResourceRef` в RoleTemplate.
- **ClusterRole**: Набор правил RBAC; может агрегироваться в системные роли с помощью меток.
- **UserBinding**: Связь между пользователями и ролями/областями; контроллер генерирует RoleBinding/ClusterRoleBinding на основе UserBinding для окончательной авторизации.

> **NOTE**
>
> - Значение `functionResourceRef` берётся из `metadata.name` FunctionResource.
> - Отображаемые имена обычно находятся в `metadata.annotations`, которые можно использовать для сопоставления с модулями UI.

3. Технические изменения

Начиная с ACP v4.3, права доступа функций постепенно мигрируют с FunctionResource на нативное управление ClusterRole K8s. Права модуля RoleTemplate будут агрегированы в системные роли через метки ClusterRole. Управление на основе FunctionResource будет снято с поддержки в v4.5.

4. Поддерживаемые методы конфигурации

- **Метод FunctionResource**: Выбор разрешений по модулям продукта и управление глаголами с тонкой детализацией.
- **Метод агрегации ClusterRole (aggregationRules)**: Централизованная агрегация через метки ClusterRole; рекомендуется для v4.2+.
- **customRules**: Настройка разрешений с использованием нативного синтаксиса RBAC K8s; формат правил такой же, как у ClusterRole `rules`.

5. Конфигурация

5.1 YAML RoleTemplate

Метод FunctionResource (поддерживается до v4.5)

```yaml
apiVersion: auth.alauda.io/v1beta1
kind: RoleTemplate
metadata:
  name: demo-funcrole
  annotations:
    cpaas.io/display-name: Пример функции роли
    cpaas.io/description: Пример FunctionResource
  labels:
    auth.cpaas.io/roletemplate.level: namespace
spec:
  rules:
    - functionResourceRef: acp-app
      verbs: [get, list, watch]
```

Пример FunctionResource:

```yaml
apiVersion: auth.alauda.io/v1beta1
kind: FunctionResource
metadata:
  name: acp-app
  annotations:
    cpaas.io/functionresource.module.display-name: Container Platform
    cpaas.io/functionresource.function.display-name: Application
  labels:
    auth.cpaas.io/functionresource.module: acp
    auth.cpaas.io/functionresource.function: app
    auth.cpaas.io/product: console-acp
spec:
  rules:
    - apiGroup: app.k8s.io
      resources: ["*"]
      bindScope: namespace
      bindCluster: unlimit
      bindNamespacePart: common
```

Метод агрегации ClusterRole (v4.2+)

```yaml
apiVersion: auth.alauda.io/v1beta1
kind: RoleTemplate
metadata:
  name: demo-aggrole
  annotations:
    cpaas.io/display-name: Пример роли агрегации
    cpaas.io/description: Пример агрегации ClusterRole
  labels:
    auth.cpaas.io/roletemplate.level: namespace
spec:
  aggregationRules:
    - clusterRoleSelectors:
        - matchLabels:
            rbac.cpaas.io/aggregate-to-namespace-developer: "true"
            rbac.cpaas.io/aggregate-to-scope-business-ns: "true"
      scope: business-ns
  rules: []
```

ClusterRole с метками агрегации:

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: cpaas:demo:business-ns:view
  labels:
    rbac.cpaas.io/aggregate-to-namespace-developer: "true"
    rbac.cpaas.io/aggregate-to-scope-business-ns: "true"
rules:
  - apiGroups: ["apps"]
    resources: ["deployments"]
    verbs: ["get", "list", "watch"]
```

Метод customRules

```yaml
apiVersion: auth.alauda.io/v1beta1
kind: RoleTemplate
metadata:
  name: demo-customrules
  annotations:
    cpaas.io/display-name: Пример роли с пользовательскими правилами
    cpaas.io/description: пример customRules
  labels:
    auth.cpaas.io/roletemplate.level: namespace
spec:
  customRules:
    - apiGroups: [""]
      resources: ["configmaps"]
      verbs: ["get", "list", "watch"]
```

Справочник полей RoleTemplate (с диапазонами):

Поле	Описание
`metadata.name`	Имя роли

Поле	Описание
`metadata.labels.auth.cpaas.io/roletemplate.level`	Уровень роли
`metadata.annotations.cpaas.io/(display-name/description)`	Отображаемое имя и описание
`spec.rules[].functionResourceRef`	Ссылка на FunctionResource
`spec.rules[].verbs`	Глаголы
`spec.aggregationRules[].scope`	Область агрегации
`spec.aggregationRules[].clusterRoleSelectors`	Селекторы меток
`spec.aggregationRules[].clusterRoleSelectors.matchLabels`	Ключ/значение метки
`spec.aggregationRules[].clusterRoleSelectors.matchExpressions`	Выражения
`spec.customRules[].apiGroups`	Группы API
`spec.customRules[].resources`	Ресурсы

Поле	Описание
`spec.customRules[].verbs`	Глаголы
`spec.customRules[].resourceNames`	Имена ресурсов
`spec.customRules[].nonResourceURLs`	URL без ресурсов

> **NOTE**
>
> `spec.rules` и `spec.aggregationRules` взаимоисключающие.

Системные метки для matchLabels

Метки агрегации системных ролей:

- `rbac.cpaas.io/aggregate-to-platform-admin: "true"`
- `rbac.cpaas.io/aggregate-to-platform-auditor: "true"`
- `rbac.cpaas.io/aggregate-to-cluster-admin: "true"`
- `rbac.cpaas.io/aggregate-to-project-admin: "true"`
- `rbac.cpaas.io/aggregate-to-namespace-admin: "true"`
- `rbac.cpaas.io/aggregate-to-namespace-developer: "true"`
- `rbac.cpaas.io/aggregate-to-basic-user: "true"`

Метки агрегации по областям:

- `rbac.cpaas.io/aggregate-to-scope-cluster: "true"`
- `rbac.cpaas.io/aggregate-to-scope-project-ns: "true"`
- `rbac.cpaas.io/aggregate-to-scope-business-ns: "true"`
- `rbac.cpaas.io/aggregate-to-scope-system-ns: "true"`
- `rbac.cpaas.io/aggregate-to-scope-kube-public: "true"`

NOTE

Поддерживаются также пользовательские метки для агрегации.

Справочник полей FunctionResource:

Поле	Описание
`metadata.name`	Идентификатор FunctionResource (для `functionResourceRef`)
`metadata.annotations`	Отображаемые имена (для сопоставления в UI)
`metadata.labels`	Метаданные модуля/функции (для индексирования)
`spec.rules[].apiGroup`	Группа API (`""` для core)
`spec.rules[].resources`	Типы ресурсов
`spec.rules[].bindScope`	Область (`cluster / namespace`)
`spec.rules[].bindCluster`	Область кластера (`global / unlimit / business`)
`spec.rules[].bindNamespacePart`	Часть namespace (`common / system / kube-public / project_ns / cluster / ""`)

NOTE

`acp-namespace-resource-manage` **не разрешён** в пользовательских ролях.

5.2 Получение ClusterRoles, сгенерированных из RoleTemplate

Запросить ClusterRoles, сгенерированные RoleTemplate, по меткам:

```
kubectl get clusterrole -l auth.cpaas.io/role.relative=<roletemplate-name
>
```

6. Копирование и сокращение (примеры)

Для пользовательских ролей рекомендуется копировать встроенный системный шаблон и сокращать его, чтобы не пропустить права модулей.

6.1 Сокращение системного RoleTemplate YAML

Шаги:

- Оставить необходимые FunctionResources.
- Сократить глаголы до только чтения (get/list/watch).
- Удалить неиспользуемые модули.

Пример роли: developer auditor без прав на секреты

> **NOTE**
>
> - В пользовательских ролях не настраивайте Управление ресурсами Namespace (FunctionResource: `acp-namespace-resource-manage`).
> - Удалите User Secret Dictionary (FunctionResource: `acp-user-secret`).
> - Установите все глаголы в get/list/watch.

Пример YAML:

```
apiVersion: auth.alauda.io/v1beta1
kind: RoleTemplate
metadata:
  annotations:
    cpaas.io/description: Ответственность за разработку, развертывание и
сопровождение в пределах namespace.
    cpaas.io/display-name: Копия разработчика
  labels:
    auth.cpaas.io/roletemplate.level: namespace
  name: namespace-developer-system-copy
spec:
  rules:
    - functionResourceRef: views-acp-userview
      verbs:
        - get
        - list
        - watch
    - functionResourceRef: infrastructure-clusters
      verbs:
        - get
        - list
        - watch
    - functionResourceRef: infrastructure-nodesmanage
      verbs:
        - get
        - list
        - watch
    - functionResourceRef: infrastructure-domains
      verbs:
        - get
        - list
        - watch
    - functionResourceRef: acp-subnet
      verbs:
        - get
        - list
        - watch
    - functionResourceRef: acp-networkpolicies
      verbs:
        - get
        - list
        - watch
    - functionResourceRef: infrastructure-storageclasses
```

```
    verbs:
      - get
      - list
      - watch
  - functionResourceRef: infrastructure-persistentvolumes
    verbs:
      - get
      - list
      - watch
  - functionResourceRef: acp-virtualmachineimagetemplates
    verbs:
      - get
      - list
      - watch
```

6.2 Сокращение namespace-developer-system до роли аудитора

Подход:

- Сохранить структуру областей (cluster / project-ns / business-ns / system-ns / kube-public).
- Заменить метки агрегации на метки аудитора (требуются помеченные ClusterRoles).

Пример (фрагмент):

```yaml
apiVersion: auth.alauda.io/v1beta1
kind: RoleTemplate
metadata:
  name: namespace-auditor
  annotations:
    cpaas.io/display-name: Аудитор namespace
    cpaas.io/description: Аудит только для чтения
  labels:
    auth.cpaas.io/roletemplate.level: namespace
spec:
  aggregationRules:
    - clusterRoleSelectors:
        - matchLabels:
            rbac.cpaas.io/aggregate-to-namespace-auditor: "true"
            rbac.cpaas.io/aggregate-to-scope-business-ns: "true"
      scope: business-ns
  rules: []
```

NOTE

`aggregate-to-namespace-auditor` должен совпадать с метками существующих ClusterRoles. Если нет, сначала создайте ClusterRole только для чтения и добавьте эту метку.

IDP

Введение

Введение

Overview

Supported Integration Methods

Руководства

Управление LDAP

Обзор LDAP

Поддерживаемые типы LDAP

Терминология LDAP

Добавление LDAP

Примеры конфигурации LDAP

Синхронизация пользователей LDAP

Соответствующие операции

Управление OIDC

Обзор OIDC

Добавление OIDC

Добавление OIDC через YAML

Соответствующие операции

Устранение неполадок

Удаление пользователя

Описание проблемы

Решение

Введение

Содержание

Overview

Платформа интегрируется с сервисом аутентификации Dex, что позволяет использовать предреализованные коннекторы Dex для аутентификации аккаунтов платформы через настройку IDP. Для получения дополнительной информации обратитесь к официальной документации Dex↗.

Supported Integration Methods

LDAP Integration

Если в вашей организации используется **LDAP** (Lightweight Directory Access Protocol) для управления пользователями, вы можете настроить LDAP на платформе для

подключения к LDAP-серверу вашей организации.

Преимущества интеграции LDAP:

- Обеспечивает связь между платформой и LDAP-сервером
- Позволяет пользователям организации входить с учетными данными LDAP
- Автоматически синхронизирует учетные записи пользователей организации с платформой

OIDC Integration

Платформа поддерживает интеграцию с IDP-сервисами, использующими протокол OpenID Connect (OIDC) для аутентификации пользователей третьих сторон.

Преимущества интеграции OIDC:

- Позволяет пользователям входить с аккаунтами третьих сторон
- Поддерживает корпоративные IDP-сервисы
- Обеспечивает безопасную аутентификацию через протокол OIDC

> **NOTE**
>
> Для аутентификации с использованием других коннекторов, не упомянутых выше, пожалуйста, обратитесь в техническую поддержку.

Руководства

Управление LDAP

Управление OIDC

Управление LDAP

Администраторы платформы могут добавлять, обновлять и удалять LDAP-сервисы на платформе.

Содержание

Обзор LDAP

LDAP (Lightweight Directory Access Protocol) — это зрелый, гибкий и хорошо поддерживаемый стандартный механизм взаимодействия с директориями. Он организует данные в иерархическую древовидную структуру для хранения информации о пользователях и организациях предприятия, преимущественно используется для реализации единого входа (SSO).

NOTE

Ключевые особенности LDAP:

- Обеспечивает связь между клиентами и LDAP-серверами
- Поддерживает операции хранения, извлечения и поиска данных
- Предоставляет возможности аутентификации клиентов
- Облегчает интеграцию с другими системами

Для получения дополнительной информации обратитесь к официальной документации LDAP ↗.

Поддерживаемые типы LDAP

OpenLDAP

OpenLDAP — это реализация LDAP с открытым исходным кодом. Если ваша организация использует open-source LDAP для аутентификации пользователей, вы можете настроить платформу для взаимодействия с LDAP-сервисом, добавив LDAP и настроив соответствующие параметры.

NOTE

Интеграция OpenLDAP:

- Обеспечивает аутентификацию пользователей LDAP на платформе
- Поддерживает стандартные протоколы LDAP
- Предоставляет гибкое управление пользователями

Для получения дополнительной информации об OpenLDAP обратитесь к официальной документации OpenLDAP ↗.

Active Directory

Active Directory — это программное обеспечение Microsoft на базе LDAP для предоставления сервисов хранения каталогов в системах Windows. Если ваша организация использует Microsoft Active Directory для управления пользователями, вы можете настроить платформу для взаимодействия с сервисом Active Directory.

NOTE

Интеграция Active Directory:

- Обеспечивает аутентификацию пользователей AD на платформе
- Поддерживает интеграцию с доменами Windows
- Предоставляет управление пользователями корпоративного уровня

Терминология LDAP

Общие термины OpenLDAP

Термин	Описание	Пример
dc (Domain Component)	Компонент домена	`dc=example,dc=com`
ou (Organizational Unit)	Организационная единица	`ou=People,dc=example,dc=com`
cn (Common Name)	Общее имя	`cn=admin,dc=example,dc=com`
uid (User ID)	Идентификатор пользователя	`uid=example`
objectClass (Object Class)	Класс объекта	`objectClass=inetOrgPerson`
mail (Mail)	Электронная почта	`mail=example@126.com`
givenName (Given Name)	Имя	`givenName=xq`
sn (Surname)	Фамилия	`sn=ren`
objectClass: groupOfNames	Группа пользователей	`objectClass: groupOfNames`
member (Member)	Атрибут участника группы	`member=cn=admin,dc=example,dc=com`
memberOf	Атрибут членства в группе	`memberOf=cn=users,dc=example,dc=com`

Общие термины Active Directory

Термин	Описание	Пример
dc (Domain Component)	Компонент домена	`dc=example,dc=com`

Термин	Описание	Пример
ou (Organizational Unit)	Организационная единица	`ou=People,dc=example,dc=co`
cn (Common Name)	Общее имя	`cn=admin,dc=example,dc=com`
sAMAccountName/userPrincipalName	Идентификатор пользователя	`userPrincipalName=example` `sAMAccountName=example`
objectClass: user	Класс объекта пользователя AD	`objectClass=user`
mail (Mail)	Электронная почта	`mail=example@126.com`
displayName	Отображаемое имя	`displayName=example`
givenName (Given Name)	Имя	`givenName=xq`
sn (Surname)	Фамилия	`sn=ren`
objectClass: group	Группа пользователей	`objectClass: group`
member (Member)	Атрибут участника группы	`member=CN=Admin,DC=example`
memberOf	Атрибут членства в группе	`memberOf=CN=Users,DC=examp`

Добавление LDAP

TIP

После успешной интеграции LDAP:

- Пользователи могут входить на платформу, используя свои корпоративные учетные записи
- Множественное добавление одного и того же LDAP перезапишет ранее синхронизированных пользователей

Предварительные требования

Перед добавлением LDAP подготовьте следующую информацию:

- Адрес LDAP-сервера
- Имя пользователя администратора
- Пароль администратора
- Другие необходимые параметры конфигурации

Шаги

1. В левой навигационной панели нажмите **Users** > **IDPs**
2. Нажмите **Add LDAP**
3. Настройте следующие параметры:

Основная информация

Параметр	Описание
Server Address	Адрес доступа к LDAP-серверу (например, `192.168.156.141:31758`)
Username	DN администратора LDAP (например, `cn=admin,dc=example,dc=com`)
Password	Пароль учетной записи администратора LDAP
Login Box Username Prompt	Сообщение-приглашение для ввода имени пользователя (например, "Please enter your username")

Настройки поиска

NOTE

Назначение настроек поиска:

- Соответствие LDAP-записям пользователей на основе заданных условий
- Извлечение ключевых атрибутов пользователей и групп
- Отображение атрибутов LDAP на атрибуты пользователей платформы

Параметр	Описание
Object Type	ObjectClass для пользователей: - OpenLDAP: `inetOrgPerson` - Active Directory: `organizationalPerson` - Группы: `posixGroup`
Login Field	Атрибут, используемый как имя пользователя для входа: - OpenLDAP: `mail` (адрес электронной почты) - Active Directory: `userPrincipalName`
Filter Conditions	Условия фильтра LDAP для фильтрации пользователей/групп Пример: `(&(cn=John*)(givenName=*xq*))`
Search Starting Point	Базовый DN для поиска пользователей/групп (например, `dc=example,dc=org`)
Search Scope	Область поиска: - `sub`: весь поддерево каталога - `one`: один уровень ниже начальной точки
Login Attribute	Уникальный идентификатор пользователя: - OpenLDAP: `uid` - Active Directory: `distinguishedName`
Name Attribute	Атрибут имени объекта (по умолчанию: `cn`)
Email Attribute	Атрибут электронной почты: - OpenLDAP: `mail`

Параметр	Описание
	- Active Directory: `userPrincipalName`
Group Member Attribute	Идентификатор участника группы (по умолчанию: `uid`)
Group Attribute	Атрибут связи с группой пользователей (по умолчанию: `memberuid`)

4. В разделе **IDP Service Configuration Validation**:

   - Введите действующее имя пользователя и пароль LDAP-аккаунта
   - Имя пользователя должно соответствовать настройке **Login Field**
   - Нажмите для проверки конфигурации

5. (Опционально) Настройте **LDAP Auto-Sync Policy**:

   - Включите переключатель **Auto-Sync Users**
   - Установите правила синхронизации
   - Используйте онлайн-инструмент↗ для проверки выражений CRON

6. Нажмите **Add**

> **NOTE**
>
> После добавления LDAP:
>
> - Пользователи могут входить до синхронизации
> - Информация о пользователях синхронизируется автоматически при первом входе
> - Автоматическая синхронизация происходит согласно заданным правилам

Примеры конфигурации LDAP

Конфигурация LDAP-коннектора

Ниже приведён пример настройки LDAP-коннектора:

```
apiVersion: dex.coreos.com/v1
kind: Connector
id: ldap        # Connector ID
name: ldap      # Connector display name
type: ldap      # Connector type is LDAP
metadata:
  name: ldap
  namespace: cpaas-system
spec:
  config:
    # LDAP server address and port
    host: ldap.example.com:636
    # DN and password for the service account used by the connector.
    # This DN is used to search for users and groups.
    bindDN: uid=serviceaccount,cn=users,dc=example,dc=com
    # Service account password, required when creating a connector.
    bindPW: password

    # Login account prompt. For example, username
    usernamePrompt: SSO Username

    # User search configuration
    userSearch:
      # Start searching from the base DN
      baseDN: cn=users,dc=example,dc=com
      # LDAP query statement, used to search for users.
      # For example: "(&(objectClass=person)(uid=<username>))"
      filter: (&(objectClass=organizationalPerson))

      # The following fields are direct mappings of user entry attribute
s.
      # User ID attribute
      idAttr: uid
      # Required. Attribute to map to email
      emailAttr: mail
      # Required. Attribute to map to username
      nameAttr: cn
      # Login username attribute
      # Filter condition will be converted to "(<attr>=<username>)", such
as (uid=example).
      username: uid

      # Extended attributes
```

```
    # phoneAttr: phone

  # Group search configuration
  groupSearch:
    # Start searching from the base DN
    baseDN: cn=groups,dc=freeipa,dc=example,dc=com
    # Group filter condition
    # "(&(objectClass=group)(member=<user uid>))".
    filter: "(objectClass=group)"
    # User group matching field
    # Group attribute
    groupAttr: member
    # User group member attribute
    userAttr: uid
    # 组显示名称
    nameAttr: cn
```

Примеры фильтров пользователей

```
# 1. Базовый фильтр: найти всех пользователей
(&(objectClass=person))

# 2. Комбинация нескольких условий: найти пользователей в определённом от
деле
(&(objectClass=person)(departmentNumber=1000))

# 3. Найти активных пользователей (Active Directory)
(&(objectClass=user)(!(userAccountControl:1.2.840.113556.1.4.803:=2)))

# 4. Найти пользователей с определённым доменом электронной почты
(&(objectClass=person)(mail=*@example.com))

# 5. Найти участников определённой группы
(&(objectClass=person)(memberOf=cn=developers,ou=groups,dc=example,dc=co
m))

# 6. Найти недавно вошедших пользователей (Active Directory)
(&(objectClass=user)(lastLogon>=20240101000000.0Z))

# 7. Исключить системные аккаунты
(&(objectClass=person)(!(uid=admin))(!(uid=system)))

# 8. Найти пользователей с определённым атрибутом
(&(objectClass=person)(mobile=*))

# 9. Найти пользователей в нескольких отделах
(&(objectClass=person)(|(ou=IT)(ou=HR)(ou=Finance)))

# 10. Пример сложной комбинации условий
(&
  (objectClass=person)
  (|(department=IT)(department=Engineering))
  (!(title=Intern))
  (manager=cn=John Doe,ou=People,dc=example,dc=com)
)
```

Примеры конфигурации поиска групп

```
# 1. Базовый фильтр: найти все группы
(objectClass=groupOfNames)

# 2. Найти группы с определённым префиксом
(&(objectClass=groupOfNames)(cn=dev-*))

# 3. Найти непустые группы
(&(objectClass=groupOfNames)(member=*))

# 4. Найти группы с определённым участником
(&(objectClass=groupOfNames)(member=uid=john,ou=People,dc=example,dc=co
m))

# 5. Найти вложенные группы (Active Directory)
(&(objectClass=group)(|(groupType=-2147483646)(groupType=-2147483644)))

# 6. Найти группы с определённым описанием
(&(objectClass=groupOfNames)(description=*admin*))

# 7. Исключить системные группы
(&(objectClass=groupOfNames)(!(cn=system*)))

# 8. Найти группы с определёнными участниками
(&(objectClass=groupOfNames)(|(cn=admins)(cn=developers)(cn=operators)))

# 9. Найти группы в определённом OU
(&(objectClass=groupOfNames)(ou=IT))

# 10. Пример сложной комбинации условий
(&
  (objectClass=groupOfNames)
  (|(cn=prod-*)(cn=dev-*))
  (!(cn=deprecated-*))
  (owner=cn=admin,dc=example,dc=com)
)
```

Примеры использования операторов AND(&) и OR(|) в LDAP-фильтрах

```
# Оператор AND (&) - должны выполняться все условия
# Синтаксис: (&(condition1)(condition2)(condition3)...)

# Пример AND с несколькими атрибутами
(&
  (objectClass=person)
  (mail=*@example.com)
  (title=Engineer)
  (manager=*)
)

# Оператор OR (|) - должно выполняться хотя бы одно условие
# Синтаксис: (|(condition1)(condition2)(condition3)...)

# Пример OR с несколькими атрибутами
(|
  (department=IT)
  (department=HR)
  (department=Finance)
)

# Комбинирование AND и OR
(&
  (objectClass=person)
  (|
    (department=IT)
    (department=R&D)
  )
  (employeeType=FullTime)
)

# Сложная комбинация условий
(&
  (objectClass=person)
  (|
    (&
      (department=IT)
      (title=*Engineer*)
    )
    (&
      (department=R&D)
      (title=*Developer*)
    )
```

```
  )
  (!(status=Inactive))
  (|(manager=*)(isManager=TRUE))
)
```

Синхронизация пользователей LDAP

После успешной синхронизации пользователей LDAP на платформу вы можете просмотреть синхронизированных пользователей в списке пользователей.

Вы можете настроить политику автоматической синхронизации при добавлении LDAP (которую можно обновить позже) или вручную запустить синхронизацию после успешного добавления LDAP. Ниже описано, как вручную запустить операцию синхронизации.

Примечания:

- Новые пользователи, добавленные в LDAP, интегрированный с платформой, могут войти на платформу до выполнения операции синхронизации пользователей. После успешного входа их информация автоматически синхронизируется с платформой.
- Пользователи, удалённые из LDAP, после синхронизации получат статус `Invalid`.
- По умолчанию срок действия вновь синхронизированных пользователей — **Постоянный**.
- Синхронизированные пользователи с тем же именем, что и существующие пользователи (локальные пользователи, пользователи IDP), автоматически связываются. Их права и срок действия будут соответствовать существующим пользователям. Они могут входить на платформу, используя метод входа, соответствующий их источнику.

Порядок действий

1. В левой навигационной панели нажмите **Users** > **IDPs**.
2. Нажмите на ***имя LDAP***, для которого хотите выполнить ручную синхронизацию.
3. В правом верхнем углу нажмите **Actions** > **Sync user**.
4. Нажмите **Sync**.

Примечания: Если вы вручную закроете диалоговое окно синхронизации, появится окно подтверждения закрытия. После закрытия диалога система продолжит синхронизацию пользователей. Если вы остаетесь на странице списка пользователей, получите обратную связь о результатах синхронизации. Если покинете страницу списка пользователей, результаты синхронизации не будут отображены.

Соответствующие операции

Вы можете нажать на
справа на странице списка или нажать **Actions** в правом верхнем углу на странице деталей, чтобы при необходимости обновить или удалить LDAP.

Операция	Описание
Обновить LDAP	Обновить конфигурацию добавленного LDAP или **LDAP Auto-Sync Policy**. **Примечание**: После обновления LDAP пользователи, синхронизированные с платформой через этот LDAP, также будут обновлены. Пользователи, удалённые из LDAP, станут недействительными в списке пользователей платформы. Для очистки мусорных данных можно выполнить операцию очистки недействительных пользователей.
Удалить LDAP	После удаления LDAP все пользователи, синхронизированные с платформой через этот LDAP, получат статус **Invalid** (связь между пользователями и ролями сохраняется), и они не смогут войти на платформу. После повторной интеграции необходимо повторно выполнить синхронизацию для активации пользователей. **Совет**: После удаления IDP, если необходимо удалить пользователей и группы пользователей, синхронизированные с платформой через LDAP, отметьте чекбокс **Clean IDP Users and User Groups** под окном подтверждения.

Управление OIDC

Платформа поддерживает протокол OIDC (OpenID Connect), позволяющий администраторам платформы входить в систему с использованием сторонних аккаунтов после добавления конфигурации OIDC. Администраторы платформы также могут обновлять и удалять настроенные службы OIDC.

Содержание

Обзор OIDC

OIDC (OpenID Connect) — это стандартный протокол аутентификации личности, основанный на протоколе OAuth 2.0. Он использует сервер авторизации OAuth 2.0 для предоставления аутентификации пользователя сторонним клиентам и передачи соответствующей информации об аутентификации личности клиенту.

OIDC позволяет всем типам клиентов (включая серверные, мобильные и JavaScript-клиенты) запрашивать и получать аутентифицированные сессии и информацию о конечном пользователе. Этот набор спецификаций расширяем, что позволяет участникам использовать дополнительные функции, такие как шифрование данных личности, обнаружение OpenID Provider и управление сессиями, когда это имеет смысл. Для получения дополнительной информации обратитесь к официальной документации OIDC↗.

Добавление OIDC

Добавив OIDC, вы сможете использовать сторонние аккаунты платформы для входа на платформу.

Примечание: После успешного входа пользователей OIDC на платформу, платформа будет использовать атрибут email пользователя в качестве уникального идентификатора. Пользователи сторонних платформ с поддержкой OIDC должны иметь атрибут **email**, иначе они не смогут войти на платформу.

Порядок действий

1. В левой навигационной панели нажмите **Users** > **IDPs**.
2. Нажмите **Add OIDC**.
3. Настройте параметры **Basic Information**.
4. Настройте параметры **OIDC Server Configuration**:

   - **Identity Provider URL**: URL издателя, который является адресом доступа к поставщику идентификации OIDC.
   - **Client ID**: идентификатор клиента для OIDC клиента.
   - **Client Secret**: секретный ключ для OIDC клиента.
   - **Redirect URI**: адрес обратного вызова после входа на стороннюю платформу, который представляет собой URL издателя dex + `/callback`.
   - **Logout URL**: адрес, на который пользователь будет перенаправлен после выполнения операции **Logout**. Если пусто, адрес выхода будет начальной страницей входа платформы.

5. В области **IDP Service Configuration Validation** введите **Username** и **Password** действительной учетной записи OIDC для проверки конфигурации.
   Совет: Если имя пользователя и пароль неверны, при добавлении будет выдана ошибка с указанием недействительных учетных данных, и OIDC не сможет быть добавлен.
6. Нажмите **Create**.

Добавление OIDC через YAML

Помимо конфигурации через форму, платформа также поддерживает добавление OIDC через YAML, что позволяет более гибко настраивать параметры аутентификации, сопоставления claims, синхронизацию групп пользователей и другие расширенные функции.

Пример: Конфигурация OIDC Connector

Следующий пример демонстрирует, как настроить OIDC connector для интеграции с сервисами аутентификации личности OIDC. Этот пример конфигурации подходит для следующих сценариев:

1. Необходимо интегрировать OIDC в качестве сервера аутентификации личности.
2. Необходимо поддерживать синхронизацию информации о группах пользователей.
3. Необходимо настроить адрес перенаправления после выхода из системы.
4. Необходимо настроить конкретные области (scopes) OIDC.
5. Необходимо настроить сопоставление claims.

```yaml
apiVersion: dex.coreos.com/v1
kind: Connector
# Connector basic information
id: oidc                # Connector unique identifier
name: oidc              # Connector display name
type: oidc              # Connector type is OIDC
metadata:
  annotations:
    cpaas.io/description: "11"  # Connector description
  name: oidc
  namespace: cpaas-system
spec:
  config:
    # OIDC server configuration
    # Configure server connection information, including server address, client credentials, and callback address
    issuer: http://auth.com/auth/realms/master              # OIDC server address
    clientID: dex                                           # Client ID
    # Service account secret key, valid when creating Connector resources for the first time
    clientSecret: xxxxxxx
    redirectURI: https://example.com/dex/callback           # Callback address, must match the address registered by the OIDC client

    # Security configuration
    # Configure SSL verification and user information acquisition method
    insecureSkipVerify: true                                # Whether to skip SSL verification, it is recommended to set to false in a production environment
    getUserInfo: false                                      # Whether to obtain additional user information through the UserInfo endpoint

    # Logout configuration
    # Configure the redirect address after user logout
    logoutURL: https://test.com                             # Logout redirect address, can be customized to the page jumped after user logout

    # Scope configuration
    # Configure the required authorization scope, ensure that the OIDC server supports these scopes
    scopes:
      - openid                                              # Required, us
```

```
ed for OIDC basic authentication
      - profile                                                 # Optional, us
ed to obtain user basic information
      - email                                                   # Optional, us
ed to obtain user email

    # Claim mapping configuration
    # Configure the mapping relationship between OIDC returned claims and
platform user attributes
    claimMapping:
      email: email                                              # Email mappin
g, used for user unique identification
      groups: groups                                            # User group m
apping, used for organization structure
      phone: ""                                                 # Phone mappin
g, optional
      preferred_username: preferred_username                    # Username map
ping, used for display name

    # Custom claimextra configuration
    # External custom fields will be dynamically added to the user object
spec.extra field
    claimExtra:
      - field: xxx            # Custom field name
        type: string          # Field type value is consistent with the d
efinition of golang language type. For example: string, int, bool, map[st
ring]string, []string, []int

    # User group configuration
    # Configure user group synchronization related parameters, ensure tha
t the token contains group information
    groupsKey: groups                                           # Specify the
key name of group information
    insecureEnableGroups: false                                 # Whether to en
able group synchronization function
```

Соответствующие операции

Вы можете нажать на
справа на странице списка или нажать **Actions** в правом верхнем углу на странице деталей, чтобы при необходимости обновить или удалить OIDC.

Операция	Описание
Обновить OIDC	Обновить добавленную конфигурацию OIDC. После обновления информации конфигурации OIDC исходные пользователи и методы аутентификации будут сброшены и синхронизированы в соответствии с текущей конфигурацией.
Удалить OIDC	Удалить OIDC, который больше не используется платформой. После удаления OIDC все пользователи, синхронизированные на платформу через этот OIDC, будут иметь статус **Invalid** (связь между пользователями и ролями сохраняется), и они не смогут войти на платформу. После повторной интеграции пользователи могут быть активированы успешным входом на платформу. **Совет**: После удаления IDP, если необходимо удалить пользователей и группы пользователей, синхронизированные на платформу через OIDC, отметьте флажок **Clean IDP Users and User Groups** под всплывающим окном.

Устранение неполадок

Удаление пользователя

Описание проблемы

Решение

Удаление пользователя

Содержание

Описание проблемы

Проблема: При создании или синхронизации нового пользователя система сообщает, что пользователь уже существует. Как с этим поступить?

По соображениям безопасности платформа не позволяет создавать новых пользователей (как локальных, так и IDP) с именами, совпадающими с ранее удалёнными пользователями. Это ограничение распространяется на:

- Создание новых локальных пользователей с именами, совпадающими с удалёнными пользователями
- Синхронизацию IDP пользователей с именами, совпадающими с удалёнными пользователями

После обновления до текущей версии вы можете столкнуться с этой проблемой при:

- Создании новых пользователей с именами, совпадающими с пользователями, удалёнными до обновления
- Синхронизации новых пользователей с именами, совпадающими с пользователями, удалёнными до обновления

Решение

Для решения этой проблемы необходимо очистить информацию об удалённых пользователях, выполнив определённые скрипты на узлах управления вашего глобального кластера.

Очистка удалённых IDP пользователей

Выполните следующую команду на любом узле управления вашего глобального кластера:

```
kubectl delete users -l 'auth.cpaas.io/user.connector_id=<IDP Name>,auth.cpaas.io/user.state=deleted'
```

Пример:

```
kubectl delete users -l 'auth.cpaas.io/user.connector_id=github,auth.cpaas.io/user.state=deleted'
```

Очистка удалённых локальных пользователей

Выполните последовательно эти два скрипта на любом узле управления вашего глобального кластера:

1. Очистка паролей пользователей:

   ```
   kubectl get users -l 'auth.cpaas.io/user.connector_id=local,auth.cpaas.io/user.state=deleted' | awk '{print $1}' | xargs kubectl delete password -n cpaas-system
   ```

2. Очистка пользователей:

```
kubectl delete users -l 'auth.cpaas.io/user.connector_id=local,auth.cpa
as.io/user.state=deleted'
```

Пользовательская политика

Введение

Введение

Платформа предоставляет комплексные политики безопасности пользователей для повышения безопасности входа и защиты от вредоносных атак.

Содержание

Overview

Платформа поддерживает следующие политики безопасности:

- Управление безопасностью паролей
- Отключение учетных записей пользователей
- Блокировка учетных записей пользователей
- Уведомления пользователей
- Контроль доступа

Configure Security Policy

Steps

1. В левой навигационной панели нажмите **User Role Management** > **User Security Policy**
2. Нажмите **Update** в правом верхнем углу
3. Настройте политики безопасности по необходимости
4. Нажмите **Update** для сохранения изменений

WARNING

Примечания по настройке политики:

- Отметьте галочкой политику для её включения
- Снимите галочку для отключения политики
- Отключённые политики сохраняют свои данные конфигурации
- При повторном включении политики восстанавливаются предыдущие настройки

Available Policies

Policy	Description
User Authentication Policy	Включает двойную аутентификацию для входа по паролю: - Пользователи получают коды подтверждения через указанные методы уведомлений - Поддерживает различные серверы уведомлений (например, Enterprise Communication Tool Server)
Password Security Policy	Управляет требованиями к паролям: Первый вход: - Обязывает сменить пароль при первом входе на платформу

Policy	Description
	Регулярные обновления: - Требует смены пароля после указанного периода (например, 90 дней) - Запрещает вход до обновления пароля
User Disablement Policy	Автоматически отключает неактивные учетные записи: - Срабатывает после указанного периода отсутствия входа
User Locking Policy	Защищает от атак методом перебора: Условия блокировки: - Срабатывает после указанного количества неудачных попыток входа в течение 24 часов Длительность блокировки: - Учетная запись остаётся заблокированной в течение указанного времени в минутах - Автоматически разблокируется после истечения периода блокировки
Notification Policy	Управляет уведомлениями пользователей: - Отправляет начальный пароль по электронной почте после создания пользователя
Access Control	Управляет сессиями пользователей и доступом: Управление сессиями: - Автоматически завершает неактивные сессии после указанного времени - Ограничивает максимальное количество одновременных онлайн-пользователей Контроль браузера: - Завершает сессию при закрытии всех вкладок продукта - Запрещает множественные входы с одного клиента :::note Важные замечания: - Контроль доступа влияет только на новые входы после обновления политики

Policy	Description
	- Восстановление вкладок браузера может не привести к завершению сессии - При запрете повторного входа разрешён только последний вход с клиента :::

Мультиарендность (Project)

Введение

Введение

Project

Namespaces

Relationship Between Clusters, Projects, and Namespaces

Руководства

Создание проекта

Процедура

Управление квотами проекта

Что такое ProjectQuota?

Как это работает

Управлени

Обновление оо

Удаление прое

Управление кластером проекта

Введение

Добавление кластера

Удаление кластера

Управлени

Импорт участн

Удаление учас

Введение

Содержание

Project

Проект — это единица изоляции ресурсов, которая обеспечивает сценарии многопользовательского использования в предприятиях. Он разделяет ресурсы одного или нескольких кластеров на изолированные среды, обеспечивая как изоляцию ресурсов, так и изоляцию персонала. Проекты могут представлять различные дочерние компании, отделы или проектные команды внутри предприятия. С помощью управления проектами можно достичь:

- Изоляции ресурсов между проектными командами
- Управления квотами внутри арендаторов
- Эффективного распределения и контроля ресурсов

Namespaces

Namespaces — это меньшие, взаимно изолированные пространства ресурсов внутри проекта. Они служат рабочими пространствами для пользователей для реализации их производственных нагрузок. Основные характеристики namespaces включают:

- Под проектом можно создать несколько namespaces
- Общая квота ресурсов всех namespaces не может превышать квоту проекта
- Квоты ресурсов выделяются более детально на уровне namespace
- Размеры контейнеров (CPU, память) ограничиваются на уровне namespace
- Повышение эффективности использования ресурсов за счёт тонкого контроля

Relationship Between Clusters, Projects, and Namespaces

Иерархия ресурсов платформы следует следующим правилам:

- Проект может использовать ресурсы (CPU, память, хранилище) из нескольких кластеров, а кластер может выделять ресурсы нескольким проектам.
- Под проектом можно создать несколько namespaces, при этом суммарные квоты ресурсов не должны превышать общие ресурсы проекта.
- Квота ресурсов namespace должна поступать из одного кластера, и namespace может принадлежать только одному проекту.

Руководства

Создание проекта

Процедура

Управление квотами проекта

Что такое ProjectQuota?

Как это работает

Управлени

Обновление оо

Удаление прое

Управление кластером проекта

Введение

Добавление кластера

Удаление кластера

Управлени

Импорт участн

Удаление учас

Создание проекта

Перед началом работы вашей проектной команды вы можете создать проект на основе существующих ресурсов кластера на платформе. Проект будет изолирован от других проектов (тенантов) как по ресурсам, так и по персоналу. При создании проекта вы можете выделить ресурсы в соответствии с масштабом проекта и реальными бизнес-потребностями. Проект может использовать ресурсы из нескольких кластеров на платформе.

WARNING

При создании проекта платформа автоматически создаст namespace с таким же именем, как у проекта, в связанных кластерах для изоляции ресурсов на уровне платформы. Пожалуйста, не изменяйте этот namespace и его ресурсы.

Содержание

Процедура

1. В представлении **Project Management** нажмите **Create Project**.
2. На странице **Basic information** настройте следующие параметры:

Параметр	Описание
Name	Имя проекта, которое не может совпадать с именем существующего проекта или любым именем из черного списка имен проектов. В противном случае проект создать нельзя. **Примечание**: Черный список имен проектов включает специальные имена namespace в кластерах платформы: `cpaas-system`, `cert-manager`, `default`, `global-credentials`, `kube-ovn`, `kube-public`, `kube-system`, `nsx-system`, `alauda-system`, `kube-federation-system`, `ALL-ALL` и `true`.
Cluster	Кластер(ы), связанные с проектом, в которых администратор может выделять квоты ресурсов. Нажмите на выпадающий список, чтобы выбрать один или несколько кластеров. **Примечание**: Кластеры в аномальном состоянии выбрать нельзя.

3. Нажмите **Next** и на шаге настройки квот проекта ознакомьтесь с разделом Manage Resource Quotas, чтобы задать квоты ресурсов, выделяемые текущему проекту для выбранных кластеров.
4. Нажмите **Create Project**.

Управление квотами проекта

В этом руководстве объясняется, как ACP расширяет Kubernetes ResourceQuota с помощью агрегированной квоты на уровне проекта (ProjectQuota). ProjectQuota позволяет ограничить сумму ResourceQuota по всем namespace в проекте, что даёт возможность планировать и контролировать ресурсы на уровне проекта, при этом делегируя лимиты отдельным namespace.

Содержание

Что такое ProjectQuota?

- ResourceQuota (родной для Kubernetes) ограничивает ресурсы на уровне namespace (CPU, память, количество объектов и т.д.). Для понятий, ключей и использования смотрите:
  - Resource Quotas

- ProjectQuota задаёт верхний предел для всего проекта: сумма всех ResourceQuota по namespace в проекте не должна превышать жёсткие лимиты проекта по тем же ключам.

Проще говоря: ResourceQuota ограничивает один namespace; ProjectQuota ограничивает сумму по всем namespace в проекте.

Как это работает

- Порядок работы: сначала определите или скорректируйте ProjectQuota, затем распределяйте ResourceQuota по namespace в рамках бюджета проекта.
- Область действия: ProjectQuota применяется к платформенному проекту и регулирует все namespace, которые к нему принадлежат.
- Агрегированное применение при приёме запроса:
  - При создании или обновлении ResourceQuota namespace платформа вычисляет сумму по тем же ключам (например, `limits.cpu`, `requests.memory`, `pods`) по всем namespace проекта, включая вносимое изменение.
  - Запрос разрешается только если новая сумма остаётся меньше или равна жёстким лимитам ProjectQuota. В противном случае изменение отклоняется с объяснением ошибки.
- Модель исполнения:
  - ProjectQuota ограничивает то, что можно выделить через ResourceQuota namespace (предварительное выделение), а не текущее мгновенное использование. Фактическое потребление регулируется ResourceQuota каждого namespace и планировщиком.

Когда использовать ProjectQuota

- Управление бюджетом/ёмкостью по проекту: выделить фиксированный бюджет CPU/памяти/объектов, затем распределить по namespace.
- Мультикомандные или мультиокруженческие проекты (например, `dev` / `staging` / `prod`), которые имеют общий верхний предел.

- Предотвращение размывания квот: поддерживать один «большой контейнер» на уровне проекта, чтобы квоты namespace не увеличивались незаметно со временем.

Ключи квот и единицы измерения

ProjectQuota поддерживает те же распространённые ключи, что и ResourceQuota (неполный список):

- Вычислительные ресурсы и память: `limits.cpu`, `limits.memory`, `requests.cpu`, `requests.memory`
- Количество рабочих нагрузок/объектов: `pods`, `services`, `configmaps`, `secrets`, `pvc` и другие

Единицы и правила подсчёта:

- CPU измеряется в ядрах (например, `2`, `500m`)
- Память измеряется в байтах (например, `8Gi`)
- Ключи, связанные с объектами, считаются целыми числами

Если сумма соответствующих ключей по всем namespace приближается к жёсткому лимиту ProjectQuota или превышает его, ACP блокирует создание или расширение ResourceQuota для этого ключа.

Советы по стратегии распределения

- Сначала определите «большой контейнер» проекта (ProjectQuota), затем разделите его на ResourceQuota по namespace для команд/окружений.
- Оставляйте запас 10%–30% для пиковых нагрузок и эластичного масштабирования.
- Регулярно пересматривайте: освобождайте неиспользуемые квоты и перераспределяйте; увеличивайте квоты для постоянно ограниченных namespace и соответственно корректируйте лимит проекта.

Лучшие практики и часто задаваемые вопросы

- В: При увеличении `limits.memory` в namespace возникает ошибка о превышении квоты проекта. Почему?
  - О: Жёсткий лимит ProjectQuota по этому ключу будет превышен запрашиваемым изменением. Уменьшите квоты в других namespace или сначала увеличьте лимит проекта, затем повторите изменение namespace.
- В: Я увеличил ProjectQuota, но рабочие нагрузки всё равно не запускаются.
  - О: Убедитесь, что ResourceQuota каждого namespace также увеличена соответствующим образом, и проверьте доступную ёмкость кластера/узлов.
- Рекомендация: Управляйте ProjectQuota в рамках обычного процесса контроля изменений, согласованного с планированием ёмкости (узлы/хранилище) и управлением бюджетом.

Управление проектом

В этом руководстве объясняется, как обновить основную информацию и квоты проекта для указанного проекта, а также как удалить проект.

Содержание

Обновление основной информации проекта

Обновите основную информацию для указанного проекта, такую как отображаемое имя и описание.

Порядок действий

1. В представлении **Project Management** нажмите на имя проекта, который нужно обновить.
2. В левой навигационной панели нажмите **Details**.
3. В правом верхнем углу нажмите **Actions** > **Update Basics**.

4. Измените или введите **Display name** и **Description**.
5. Нажмите **Update**.

Удаление проекта

Удалите проекты, которые больше не используются.

> **WARNING**
>
> После удаления проекта ресурсы, занятые проектом в кластере, будут освобождены.

Порядок действий

1. В представлении **Project Management** нажмите на имя проекта, который нужно удалить.
2. В левой навигационной панели нажмите **Details**.
3. В правом верхнем углу нажмите **Actions** > **Delete Project**.
4. Введите имя проекта и нажмите **Delete**.

Управление кластером проекта

В этом руководстве объясняется, как управлять ассоциациями кластеров для проекта. Вы можете добавить кластеры, чтобы выделить их ресурсы проекту, или удалить кластеры, чтобы вернуть выделенные ресурсы.

Содержание

Введение

Вы можете добавить кластеры в проект для выделения их ресурсов или удалить кластеры, чтобы вернуть выделенные ресурсы. Эта функциональность полезна в следующих сценариях:

- Когда ресурсов проекта недостаточно для бизнес-операций
- Когда необходимо выделить недавно созданный или добавленный кластер существующему проекту

- Когда нужно вернуть ресурсы кластера из проекта
- Когда определённому проекту требуется эксклюзивный доступ к кластеру

Добавление кластера

Добавьте кластер в проект и установите его квоту ресурсов.

Процедура

1. В представлении **Project Management** нажмите на имя проекта, в который хотите добавить кластер.
2. В левой навигационной панели нажмите **Details**.
3. В правом верхнем углу нажмите **Actions** > **Add Cluster**.
4. Выберите кластер и установите квоту ресурсов, которая будет выделена текущему проекту. Можно настроить следующие ресурсы:
   - CPU (ядра)
   - Память (Gi)
   - Хранилище (Gi)
   - Количество PVC (число)
   - Pods (число)
   - vGPU (виртуальный GPU)/MPS/pGPU (физический GPU, ядра)
   - Квота видеопамяти

NOTE

- Квота ресурсов GPU может быть настроена только при развертывании GPU-плагинов в кластере.

  Если ресурсы GPU — это **GPU-Manager или MPS GPU**, также можно настроить квоту **vMemory**.

  GPU Units: 100 единиц виртуальных ядер эквивалентны 1 физическому ядру (1 pGPU = 1

ядро = 100 ядер GPU-Manager = 100 ядер MPS), и единицы pGPU могут выделяться только целыми числами. 1 единица памяти GPU-Manager равна 256 Mi, 1 единица памяти MPS GPU равна 1 Gi, 1024 Mi = 1 Gi.

- Если для определённого типа ресурса квота не установлена, по умолчанию она считается **Неограниченной**. Это означает, что проект может использовать доступные ресурсы соответствующего типа в кластере по мере необходимости без максимального ограничения.
- Значение установленной квоты проекта должно находиться в пределах диапазона квот, отображаемого на странице. Под каждым полем ввода квоты ресурса отображаются выделенная квота и общий объём этого ресурса для справки.

5. Нажмите **Add**.

Удаление кластера

Удалите кластер, связанный с проектом.

WARNING

- После удаления кластера проект не сможет использовать бизнес-ресурсы, находящиеся под управлением удалённого кластера.
- Если удаляемый кластер находится в аномальном состоянии, ресурсы под этим кластером очистить не удастся. Рекомендуется исправить состояние кластера перед его удалением.

Процедура

1. В представлении **Project Management** нажмите на имя проекта, из которого хотите удалить кластер.
2. В левой навигационной панели нажмите **Details**.
3. В правом верхнем углу нажмите **Actions** > **Remove Cluster**.

4. В появившемся диалоговом окне **Remove Cluster** введите имя удаляемого кластера и нажмите кнопку **Remove** для успешного удаления кластера.

Управление участниками проекта

В этом руководстве объясняется, как управлять участниками проекта, включая импорт участников и назначение ролей, связанных с проектом.

Содержание

Импорт участников

Вы можете предоставить пользователям права на управление проектом и его пространствами имён, импортируя существующих пользователей платформы или добавляя пользователей OIDC. Назначайте роли, предоставляемые платформой, такие как администраторы проекта, администраторы пространства имён или разработчики. Если необходимы индивидуальные права, создайте собственную Kubernetes

Role/ClusterRole в разделе **Users > Platform Roles > Kubernetes Roles** и свяжите её с нужными пользователями через RoleBinding/ClusterRoleBinding.

Ограничения и условия

- Если на платформе не настроен OIDC IDP:
  - В качестве участников проекта можно импортировать только существующих пользователей платформы, включая:
    - пользователей OIDC, успешно вошедших в систему
    - пользователей, синхронизированных через LDAP
    - локальных пользователей
    - пользователей, добавленных в другие проекты как OIDC-пользователи (с источником, отмеченным как `-`)
- Если настроен OIDC IDP:
  - Можно добавлять действительные OIDC-аккаунты, соответствующие требованиям ввода
  - Проверка действительности аккаунта при добавлении не производится
  - Убедитесь, что аккаунт действителен, иначе вход будет невозможен
- Пользователи с правами системного администратора по умолчанию и текущий вошедший пользователь не могут быть импортированы

Процедура

1. В представлении **Project Management** нажмите на имя проекта, которым хотите управлять.
2. В левой навигационной панели выберите **Members**.
3. Нажмите **Import Member**.
4. Выберите либо **Member List**, либо **OIDC Users**.

Импорт из списка участников

Можно импортировать всех пользователей или выбрать конкретных из списка участников.

> **TIP**
>
> Используйте выпадающее меню групп пользователей в правом верхнем углу и поле поиска для фильтрации пользователей по имени.

Чтобы импортировать всех пользователей:

1. Выберите **Member List**.
2. Нажмите на выпадающее меню **Bind** и выберите роль, которую нужно назначить всем пользователям.
   Если роль требует указания пространства имён, выберите его в выпадающем меню **Namespaces**.
3. Нажмите **Import All**.

Чтобы импортировать конкретных пользователей:

1. Выберите **Member List**.
2. Отметьте одного или нескольких пользователей с помощью флажков.
3. Нажмите на выпадающее меню **Bind** и выберите роль для выбранных пользователей.
   Если роль требует указания пространства имён, выберите его в выпадающем меню **Namespaces**.
4. Нажмите **Import**.

Импорт пользователей OIDC

1. Выберите **OIDC Users**.
2. Нажмите **Add** для создания записи участника (повторите для нескольких участников).
3. Введите имя пользователя, аутентифицированного через OIDC, в поле **Name**.

> **WARNING**

Убедитесь, что имя пользователя соответствует аккаунту, который может быть аутентифицирован настроенной системой OIDC, иначе вход будет невозможен.

4. Выберите роль из выпадающего меню **Roles**.
   Если роль требует указания пространства имён, выберите его в выпадающем меню **Namespaces**.
5. Нажмите **Import**.

После успешного импорта вы сможете увидеть:

- участника в списке участников проекта
- пользователя в разделе **Platform Management > Users**
  - Источник отображается как "-" до первого входа/синхронизации
  - Источник обновляется после успешного входа/синхронизации

Удаление участников

Удалите участника проекта, чтобы отозвать его права.

Процедура

1. В представлении **Project Management** нажмите на имя проекта.
2. В левой навигационной панели выберите **Members**.

TIP

Используйте выпадающий список рядом с полем поиска для фильтрации участников по **Name**, **Display name** или **User Group**.

3. Нажмите **Remove** рядом с участником, которого хотите удалить.
4. Подтвердите удаление в появившемся диалоговом окне.

Аудит

Введение

Функция аудита платформы предоставляет упорядоченные по времени записи операций, связанных с пользователями и безопасностью системы. Это помогает анализировать конкретные проблемы и быстро решать возникающие в кластерах, пользовательских приложениях и других областях вопросы.

С помощью аудита вы можете отслеживать различные изменения в Kubernetes кластере, включая:

- Какие изменения произошли в кластере за определённый период времени
- Кто выполнил эти изменения (системные компоненты или пользователи)
- Детали важных событий изменений (например, обновления параметров POD)
- Результаты событий (успех или ошибка)
- Местоположение оператора (внутри или вне кластера)
- Записи операций пользователей (обновления, удаления, управленческие операции) и их результаты

Содержание

Требования

- Ваша учетная запись должна иметь права управления платформой или аудита платформы.
- Эта функция зависит от сервиса логирования. Пожалуйста, убедитесь, что в платформе предварительно установлены плагины ACP Log Essentails, ACP Log Collector и ACP Log Storage.

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

Процедура

1. В левой навигационной панели нажмите **Auditing**.
2. Выберите область аудита на вкладках:
   - **User Operations**: Просмотр записей операций пользователей, вошедших в платформу
   - **System Operations**: Просмотр записей системных операций (операторы начинаются с `system:`)
3. Настройте условия запроса для фильтрации событий аудита:

Условие запроса	Описание
Operator	Имя пользователя или системной учетной записи оператора (по умолчанию: `All`)
Actions	Тип операции (create, update, delete, manage, rollback, stop и др., по умолчанию: `All`)

Условие запроса	Описание
Clusters	Кластер, содержащий управляемый ресурс (по умолчанию: `All`)
Resource Type	Тип управляемого ресурса (по умолчанию: `All`)
Resource Name	Имя управляемого ресурса (поддерживается нечеткий поиск)

4. Нажмите **Search**.

TIP

- Используйте выпадающий список **Time Range** для установки временного диапазона аудита (по умолчанию: `Last 30 Minutes`). Можно выбрать предустановленный диапазон или задать свой.
- Нажмите на иконку обновления для обновления результатов поиска.
- Нажмите на иконку экспорта для скачивания результатов в формате `.csv`.

Результаты поиска

В результатах поиска отображается следующая информация:

Параметр	Описание
Operator	Имя пользователя или системной учетной записи оператора
Actions	Тип операции (create, update, delete, manage, rollback, stop и др.)
Resource Name/Type	Имя и тип управляемого ресурса
Clusters	Кластер, содержащий управляемый ресурс

Параметр	Описание
Namespaces	Namespace, содержащий управляемый ресурс
Client IP	IP-адрес клиента, с которого была выполнена операция
Operation Result	Результат операции на основе кода возврата API (2xx = успех, другие = ошибка)
Operation Time	Временная метка операции
Details	Нажмите кнопку **Details**, чтобы просмотреть полный аудиторский запись в формате JSON в диалоговом окне **Audit Details**

Телеметрия

Установка

Установка

ACP Telemetry — это сервис платформы, который собирает телеметрические данные с кластеров для онлайн-операций и обслуживания. Он собирает системные метрики и загружает их в Alauda Cloud для мониторинга и анализа.

Содержание

Предварительные требования

Перед установкой убедитесь, что:

- В Alauda Container Platform есть действующая лицензия
- Глобальный кластер имеет доступ в интернет

Шаги установки

1. Перейдите в **Administrator**

2. В левой навигационной панели нажмите **Marketplace** > **Cluster Plugins**
3. Выберите кластер **global** в верхней навигационной панели
4. Найдите **ACP Telemetry** и нажмите для просмотра деталей
5. Нажмите **Install** для развертывания плагина

Включение онлайн-операций

1. В левой навигационной панели нажмите **System Settings** > **Platform Maintenance**
2. Нажмите кнопку **On** для **Online Operations**

Шаги удаления

1. Выполните шаги 1-4 из процесса установки, чтобы найти плагин
2. Нажмите **Uninstall** для удаления плагина

Сертификаты

Автоматическая ротация серт cert-manager Сертифика

Мониторинг сертификатов Ротация TL

Automated Kubernetes Certificate Rotation

Это руководство поможет вам установить, понять и управлять Kubernetes Certificate Rotator в ACP для автоматизации ротации сертификатов Kubernetes в ваших кластерах.

Содержание

Installation

Смотрите Cluster Plugin для инструкций по установке.

> **Примечание:**
>
> - В настоящее время поддерживаются:
>
> - On-Premises кластеры
> - DCS кластеры

How it works

Этот плагин обрабатывает автоматическую ротацию следующих сертификатов.

Certificate file	Function	Node Type
apiserver.crt	Серверный сертификат для kube-apiserver	Control Plane Node
apiserver-etcd-client.crt	Клиентский сертификат для kube-apiserver для доступа к etcd	Control Plane Node
apiserver-kubelet-client.crt	Клиентский сертификат для kube-apiserver для доступа к kubelet	Control Plane Node
front-proxy-client.crt	Клиентский сертификат для kube-apiserver для доступа к агрегированным API серверам	Control Plane Node
etcd/server.crt	Серверный сертификат для etcd	Control Plane Node
etcd/peer.crt	Сертификат для взаимодействия между членами etcd	Control Plane Node
/root/.kube/config, admin.conf, super-admin.conf	Клиентский сертификат в kubeconfig для администрирования кластера	Control Plane Node
controller-manager.conf	Клиентский сертификат в kubeconfig для kube-controller-manager	Control Plane Node
scheduler.conf	Клиентский сертификат в kubeconfig для kube-scheduler	Control Plane Node

Certificate file	Function	Node Type
kubelet.crt	Серверный сертификат для kubelet	All Nodes
kubelet-client-current.pem	Клиентский сертификат для kubelet (ссылается в kubelet.conf)	All Nodes

Rotation Process

1. **Загрузка информации о сертификатах**
   Начальный этап включает сбор метаданных для всех целевых сертификатов. Поскольку эти сертификаты хранятся в разных путях на хосте, их содержимое должно быть прочитано из соответствующих файлов. Для этого создаётся временный Pod на целевом узле с примонтированными каталогами сертификатов, что позволяет Pod читать информацию. Информация о сертификатах собирается один раз в день. Детали сертификатов (пути, срок действия) хранятся в ConfigMap `cpaas-system/node-local-certs-<node-name>`. Зашифрованный CA сертификат хранится в Secret `cpaas-system/kubernetes-ca`.
2. **Условие запуска ротации**
   Поля `notBefore` и `notAfter` сертификата указывают период его действия. Ротация запускается, если оставшийся срок действия меньше 20% или 30 дней.
3. **Очередь ротации**
   Сертификаты, требующие ротации, помещаются в очередь на обработку. Программа ротации оценивает недавние действия по ротации и срочность ожидающих задач, чтобы решить, обрабатывать ли их немедленно. Это предотвращает возможные проблемы с состоянием кластера из-за одновременной ротации нескольких сертификатов.
4. **Генерация новых сертификатов**
   Программа ротации генерирует новые сертификаты на основе внутренне сохранённой информации CA. Процесс ротации создаёт временный Pod на целевом узле с примонтированными необходимыми каталогами сертификатов, что позволяет контролируемо изменять файлы.
5. **Перезапуск компонентов**
   Требуется перезапуск:

- `kube-apiserver`: необходимо перезапустить для загрузки новых сертификатов. Во время перезапуска он регенерирует внутренний loopback сертификат (действительный один год, используется только внутри и не может быть внешне ротирован).
- `kube-controller-manager`: необходимо перезапустить для перезагрузки kubeconfig файла.
- `kube-scheduler`: необходимо перезапустить для перезагрузки kubeconfig файла.
- `kubelet`: необходимо перезапустить для перезагрузки серверного сертификата.

Метод перезапуска: Добавить аннотации в YAML-файлы соответствующих статических Pod для запуска kubelet на пересоздание Pod. Для перезапуска kubelet примонтировать файловую систему хоста с `hostPID is true` и выполнить в контейнере команду "systemctl restart kubelet".
Автоматическая перезагрузка:

- Etcd может автоматически перезагружать сертификаты.

6. **Сроки ротации**

- Сертификаты `kubelet`: ротация через 61 день (срок действия 91 день)
- Сертификаты управляющей плоскости: ротация через 292 дня (срок действия 365 дней)

Operation Considerations

Если `kubelet` находится в ненормальном состоянии в окне ротации и не может автоматически ротировать сертификаты, требуется ручная ротация:

Операторы должны вручную обновить сертификаты.

Выполните следующие команды для ручного обновления сертификатов:

```
cert-renew --ca-cert <ca-cert-path> --ca-key <ca-key-path> --days <days>
<certificate or kubeconfig 1> <certificate or kubeconfig 2> ...
```

Например, чтобы обновить `kubelet.crt`:

```
cert-renew --ca-cert /etc/kubernetes/pki/ca.crt --ca-key /etc/kubernetes/
pki/ca.key --days 91 /etc/kubernetes/pki/kubelet.crt
```

Чтобы скачать и подготовить инструмент `cert-renew`, выполните:

```
curl "$(kubectl get services -n cpaas-system frontend -o jsonpath='{.spe
c.clusterIP}'):8080/cluster-cert-rotator/download/cert-renew" -o ./cert-r
enew && chmod +x ./cert-renew
```

Опционально, скачайте `renew-all.sh` для обновления всех сертификатов на узле:

```
curl "$(kubectl get services -n cpaas-system frontend -o jsonpath='{.spe
c.clusterIP}'):8080/cluster-cert-rotator/download/renew-all.sh" -o ./rene
w-all.sh
```

cert-manager

В каждом кластере автоматически разворачивается **Certificate для cert-manager**

cert-manager — это нативный контроллер управления сертификатами Kubernetes, который автоматически генерирует и управляет TLS-сертификатами на основе ресурсов `Certificate`. Многие компоненты в Kubernetes-кластерах используют cert-manager для управления своими TLS-сертификатами, обеспечивая безопасное взаимодействие.

Содержание

Обзор

Cert-manager управляет жизненным циклом сертификатов через Custom Resource Definitions (CRD) Kubernetes:

- **Certificate**: Определяет сертификаты, которые необходимо управлять
- **Issuer/ClusterIssuer**: Определяет издателей сертификатов

- **CertificateRequest**: Внутренний ресурс для обработки запросов на сертификаты

Как это работает

Когда создаётся ресурс `Certificate`, cert-manager автоматически:

1. Генерирует приватные ключи и запросы на подпись сертификатов
2. Получает подписанные сертификаты от указанного Issuer
3. Сохраняет сертификаты и приватные ключи в Kubernetes Secrets

Кроме того, cert-manager отслеживает срок действия сертификатов и обновляет их до истечения, чтобы обеспечить непрерывную доступность сервиса.

Определение сертификатов, управляемых cert-manager

Сертификаты, управляемые cert-manager, имеют соответствующие ресурсы `Secret` с типом `kubernetes.io/tls` и определёнными метками и аннотациями.

Общие метки и аннотации

Ресурсы `Secret`, управляемые cert-manager, обычно содержат следующие метки и аннотации:

Метки:

- `controller.cert-manager.io/fao: "true"`: Указывает, что этот Secret управляется cert-manager и включает фильтрованное кэширование Secret контроллером.

Аннотации:

- `cert-manager.io/certificate-name`: Имя сертификата
- `cert-manager.io/common-name`: Общее имя сертификата
- `cert-manager.io/alt-names`: Альтернативные имена сертификата

- `cert-manager.io/ip-sans`: IP-адреса сертификата
- `cert-manager.io/issuer-kind`: Тип издателя сертификата
- `cert-manager.io/issuer-name`: Имя издателя сертификата
- `cert-manager.io/issuer-group`: API-группа издателя
- `cert-manager.io/uri-sans`: URI Subject Alternative Names

Связанные ресурсы

- cert-manager Official Documentation ↗

Сертификаты OLM

Все сертификаты для компонентов **Operator Lifecycle Manager (OLM)** — включая `olm-operator`, `catalog-operator`, `packageserver` и `marketplace-operator` — автоматически управляются системой.

При установке операторов, которые определяют **webhooks** или **API services** в своем объекте **ClusterServiceVersion (CSV)**, OLM автоматически генерирует и обновляет необходимые сертификаты.

Мониторинг сертификатов

Cluster Enhancer предоставляет возможности мониторинга сертификатов, используемых в Kubernetes кластерах. Область мониторинга включает:

1. **Сертификаты компонентов Kubernetes**, включая сертификаты control plane и kubelet сервер/клиент (включая kubeconfig клиентские сертификаты)
2. **Сертификаты компонентов, работающих в кластере**, реализованные путем проверки всех Secrets с типом `kubernetes.io/tls`
3. **Сертификаты серверов, фактически используемые kube-apiserver** (включая внутренние loopback сертификаты для самодоступа) через доступ к Endpoints `kubernetes`

Пользователи могут найти и установить **Cluster Enhancer** во вкладке **Administrator**, перейдя в **Marketplace** > **Cluster Plugins** в левой навигации.

Содержание

Мониторинг статуса сертификатов

Статус истечения срока действия сертификатов можно просмотреть через метрику `certificate_expires_status`. Время истечения срока действия сертификатов можно просмотреть через метрику `certificate_expires_time`.

Текущий статус сертификата и время истечения срока действия можно увидеть во вкладке **Certificate Status**. Чтобы получить доступ к этой вкладке, перейдите во вкладку **Administrator**, затем в **Clusters > Clusters**, выберите конкретный кластер и перейдите на вкладку **Monitoring**.

Встроенные правила оповещений

Cluster Enhancer предоставляет встроенные правила оповещений `cpaas-certificates-rule` со следующими предупреждениями:

Оповещения по сертификатам Kubernetes

Правило	Уровень
Время истечения сертификата kubernetes приближается (меньше 30 дней) <= 30d и последние 1 минуты	Средний
Время истечения сертификата kubernetes приближается (меньше 10 дней) <= 10d и последние 1 минуты	Высокий
Сертификат kubernetes истек <= 0d и последние 1 минуты	Критический

Оповещения по сертификатам компонентов платформы

Правило	Уровень
Время истечения сертификата компонентов платформы приближается (меньше 30 дней) <= 30d и последние 1 минуты	Средний

Правило	Уровень
Время истечения сертификата компонентов платформы приближается (меньше 10 дней) <= 10d и последние 1 минуты	Высокий
Сертификат компонентов платформы истек <= 0d и последние 1 минуты	Критический

Ротация TLS сертификатов адресов доступа платформы

INFO

Для версии ACP v4.0.x применяйте ту же процедуру как к основному кластеру, так и к резервному кластеру (условия настройки Disaster Recovery), как описано здесь.

Содержание

Предварительные требования

- Пара TLS сертификатов и соответствующий закрытый ключ.

Процедуры

1. На любом узле control-plane в глобальном кластере экспортируйте резервные копии TLS сертификатов, используемых адресами доступа платформы ACP:

```
kubectl get certificate -n cpaas-system dex-serving-cert --ignore-not-found=true -o yaml > /cpaas/dex-serving-cert.yaml
kubectl get secret -n cpaas-system dex.tls -o yaml > /cpaas/dex.tls.yaml
```

2. Удалите текущие сертификаты:

```
kubectl delete certificate -n cpaas-system dex-serving-cert --ignore-not-found=true
kubectl delete secret -n cpaas-system dex.tls
```

3. Введите новый сертификат:

```
kubectl create secret tls dex.tls --cert=/path/to/tls.crt --key=/path/to/tls.key -n cpaas-system
```