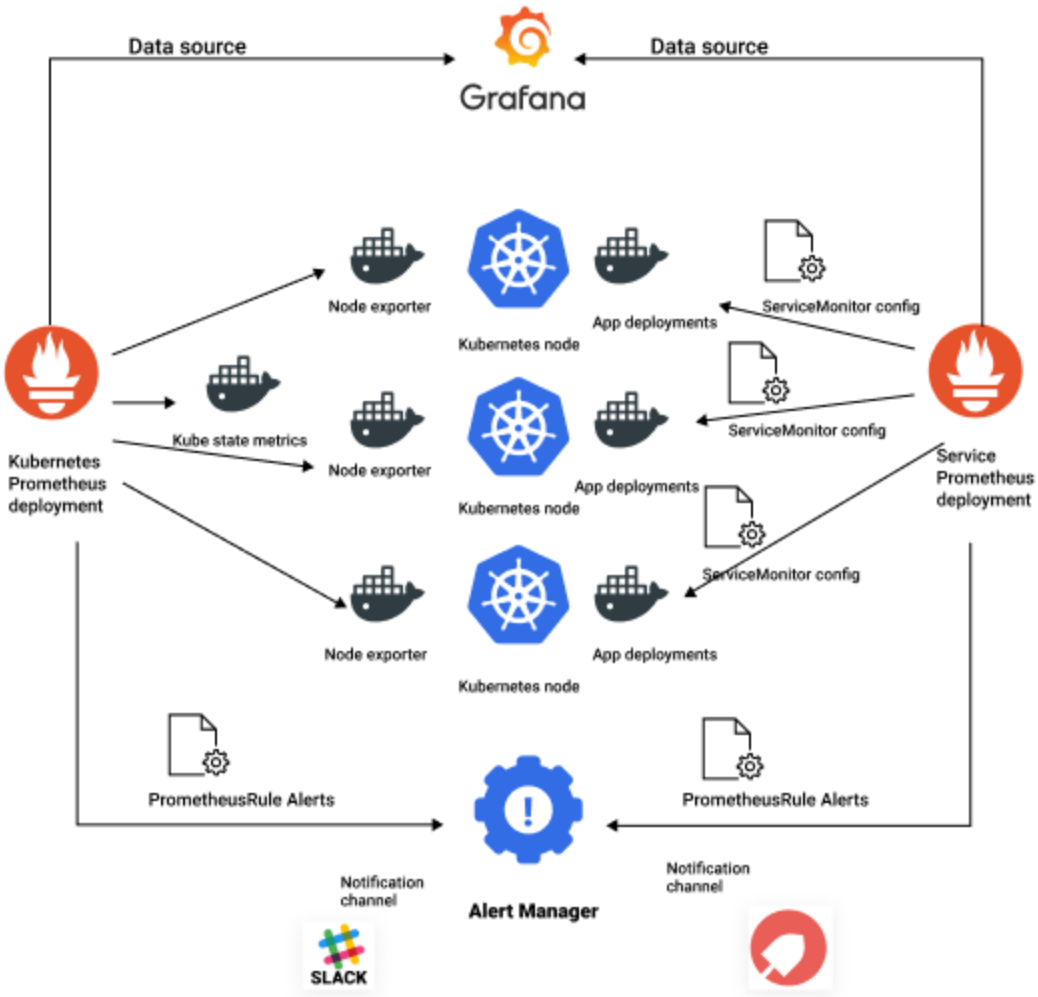
KUB 32: Мониторинг Kubernetes с помощью Kube Prometheus Stack

Описание:

Prometheus

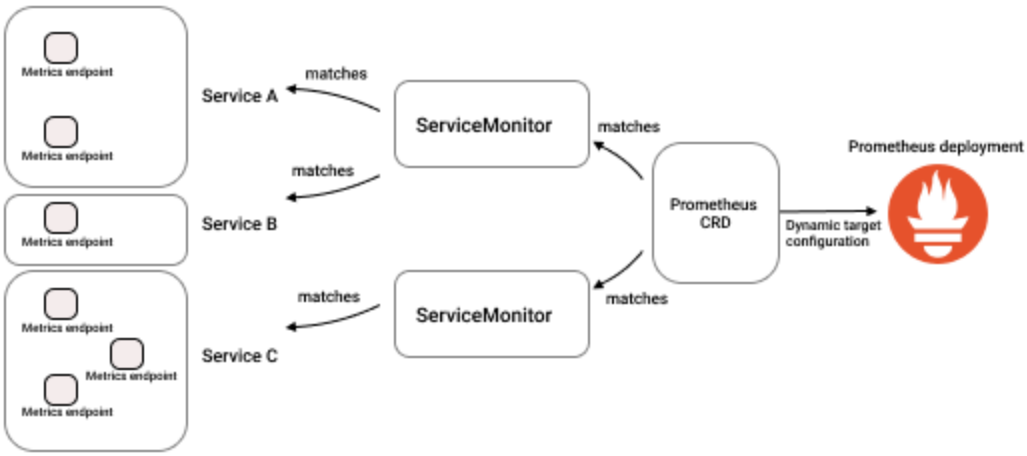
Под разный тип нужных ресурсов кастомный контроллер не сделаешь, да и управлять ими довольно тяжело. CoreOS выступила с инициативой фреймворка для подобных вещей. Вы устанавливаете определенный набор объектов в кластер и приобретаете новый функционал и CRD. Например, сможете разворачивать кластера mysql, kafka, postgres и многое другое простым созданием объекта в Kubernetes. Существует много реализаций operators для разных приложений. Одна из постоянно используемых — Prometheus.



После его установки вы сможете собирать метрики с разнообразных ресурсов в кластере. Основные типы ресурсов нашего оператора:

- Prometheus — управляет созданием сервера сбора метрик prometheus;
- PrometheusRule — создает правила мониторинга для Prometheus;
- AlertManager — управляет серверами для отправки уведомлений;
- ServiceMonitor — описывает объекты для сбора метрик.

Компоненты связаны между собой следующей схемой:



При необходимости при помощи ServiceMonitor получится подключать новые сервисы к нашему мониторингу, определяя в них, какие Service, порты и endpoint должны опрашиваться.
Работает вся эта магия благодаря опросам со стороны Prometheus Operator Kubernetes API и генерации конфигов и манифестов в кластере.
Давайте попробуем установить prometheus оператор в кластер. Для этого нам понадобится скачать чарт для prometheus operator и установить его. Здесь доступна детальная информация по этому чарту. Итак, поехали:

```
# Создаем namespace для мониторинга
$ kubectl create ns monitoring
namespace/monitoring created
# Скачиваем чарт локально
$ helm repo add prometheus-community
https://prometheus-community.github.io/helm-charts
"prometheus-community" has been added to your repositories

$ helm repo add stable https://charts.helm.sh/stable
"stable" has been added to your repositories

$ helm repo update
Hang tight while we grab the latest from your chart
repositories...
...Successfully got an update from the "ingress-nginx" chart
repository
...Successfully got an update from the "jetstack" chart
repository
...Successfully got an update from the "prometheus-community"
chart repository
...Successfully got an update from the "bitnami" chart
repository
```

```
...Successfully got an update from the "stable" chart repository
Update Complete. ⎈Happy Helming!⎈

$ helm  pull prometheus-community/kube-prometheus-stack

$ tar zxf kube-prometheus-stack-12.4.0.tgz

$ cd kube-prometheus-stack
```

Отлично, теперь мы можем подправить все необходимые переменные в файле values.yaml, предварительно его скопировав, и после этого задеплоить наш чарт. Настроек там действительно очень и очень много, но для начала достаточно только включить настройку ingress, чтобы мы могли получить доступ к нашему стеку (ingress.enabled: true и указать hosts). После этого можно деплоить чарт:

```
$ helm -n monitoring upgrade --install prometheus-stack -f
values.dev.yaml ./
Release "prometheus-stack" does not exist. Installing it now.
NAME: prometheus-stack
LAST DEPLOYED: Fri Dec  4 16:43:45 2020
NAMESPACE: monitoring
STATUS: deployed
REVISION: 1
NOTES:
kube-prometheus-stack has been installed. Check its status by
running:
  kubectl --namespace monitoring get pods -l
"release=prometheus-stack"

Visit https://github.com/prometheus-operator/kube-prometheus for
instructions on how to create & configure Alertmanager and
Prometheus instances using the Operator.
```

Теперь мы можем зайти через веб на ингресс и посмотреть статус его работы. Кстати, после установки prometheus operator создал много новых ресурсов, которые мы можем описывать:

```
$ kubectl get crd
NAME                                                  CREATED AT
alertmanagerconfigs.monitoring.coreos.com
2020-12-04T13:43:38Z
alertmanagers.monitoring.coreos.com
2020-12-04T13:43:38Z
```

```
podmonitors.monitoring.coreos.com
2020-12-04T13:43:38Z
probes.monitoring.coreos.com
2020-12-04T13:43:39Z
prometheuses.monitoring.coreos.com
2020-12-04T13:43:39Z
prometheusrules.monitoring.coreos.com
2020-12-04T13:43:39Z
servicemonitors.monitoring.coreos.com
2020-12-04T13:43:39Z
thanosrulers.monitoring.coreos.com
2020-12-04T13:43:40Z
volumesnapshotclasses.snapshot.storage.k8s.io
2020-12-02T12:20:27Z
volumesnapshotcontents.snapshot.storage.k8s.io
2020-12-02T12:20:27Z
volumesnapshots.snapshot.storage.k8s.io
2020-12-02T12:20:27Z
```

Давайте теперь попробуем задеплоить pod, с которого prometheus будет собирать метрики:

```
apiVersion: v1
kind: Pod
metadata:
  name: whoami
  labels:
    app: whoami
spec:
  containers:
    - name: whoami
      image: bee42/whoami:2.2.0
      ports:
      - containerPort: 80
        name: web
```

Наш под будет слушать на 80 порту и отдавать метрики по запросу /metrics. Давайте теперь создадим podmonitor, который научит prometheus опрашивать под:

```
apiVersion: monitoring.coreos.com/v1
kind: PodMonitor
metadata:
  name: whoami-monitor
```

```
  labels:
    release: prometheus-stack
spec:
  selector:
    matchLabels:
      app: whoami
  podMetricsEndpoints:
  - port: web
```

Также можем посмотреть, что наш podmonitor успешно создался:

```
$ kubectl get podmonitors
NAME              AGE
whoami-monitor    58s
```

Теперь можно зайти в prometheus или grafana и проверить, что данные с нового пода успешно собираются. Вообще PrometheusOperator создает достаточно много ресурсов, но основные из них — это ServiceMonitor & PodMonitor, которые позволяют мониторить endpoints сервисов и непосредственно поды. При добавлении этих ресурсов prometheus operator создаст и обновит конфигурацию prometheus, который начнет собирать данные с этих сервисов.

Задание:

1. Установите kube-prometheus-stack в ваш кластер в namespace monitoring.
2. Запустите pod redis-pod в namespace default с двумя контейнерами — redis & redis_exporter.
3. Создайте podmonitor redis-monitor в namespace default, чтобы prometheus собирал данные с redis_exporter.
4. Добавьте в графану dashboard, который будет выводить количество операций в секунду в редисе (вы можете использовать доменное имя, выданное вам в задании для настройки ingress для grafana)
5. Отправьте задание на проверку.