K8S 59: Kubernetes. Monitoring. Extended Metrics

Описание:

Одна из основных Metrics-server - это возможность использования выданных метрик, которые система может использовать для горизонтального автоматического расширения подов (Horizontal Pod Autoscaler или HPA). Пример - при высоком использовании CPU мы можем сделать подом правило для автоматического увеличения количества подов для уменьшения количества запросов на каждый экземпляр приложения, а позднее, когда нагрузка спадет (метрика станет ниже определенного значения), количество автоматически уменьшится, убив часть запущенных подов.
Но, скажем, вы знаете, что ваше приложение выдает метрику количества запросов в последнюю минуту и что при выходе из диапазона в 100-200 сервер начнет помирать, и хорошо бы иметь возможность использовать HPA для этого, но Metrics-server, естественно, эти метрики не выдает.
И вот тут вступает в силу возможность расширять список метрик, которые используются HPA, при помощи расширенных метрик (extended).
Требуется немного разобраться, как вся эта магия работает внутри на уровне Kubernetes API:

1. Metrics-server все собранные метрики отдает в Kubernetes Server API в виде ресурсов типа metrics.k8s.io, к которым, кстати, kubectl top и обращается.
2. Кастомные метрики могут храниться в подобной схеме в ресурсах типа custom.metrics.k8s.io или external.metrics.k8s.io, которые могут создавать другие приложения.

Есть 2 распространенных метода для передачи кастомных метрик:

1. Написание своего адаптера - для этого существует (официальный framework)[https://github.com/kubernetes-incubator/custom-metrics-apiserver].
2. Использовать существующий адаптер, который отдает метрики как объекты метрики, собранные где-то еще, к примеру, (из prometheus)[https://github.com/DirectXMan12/k8s-prometheus-adapter].

Но для того, чтобы вообще появилась возможность работы с кастомными метриками, требуется сначала активировать Aggregation Layer, который позволяет создавать новые API, которые обрабатывает другой сервис, определенный через ресурсы типа APIService, и на который направляет запросы Kubernetes API Server.

Полезные ссылки:

- metrics-server (github)
- Horizontal Pod Autoscaler (official docs)

- Horizontal Pod Autoscaler Walkthrough (official docs)
- Configure the Aggregation Layer (official docs)
- Resource metrics pipeline (official docs)
- Resource Metrics API
- Configure Kubernetes Autoscaling with Custom Metrics
- Horizontal Pod Autoscale with Custom Prometheus Metrics

Задание:

Выберите правильные варианты ответа на вопросы ниже и аргументируйте свой выбор:

1. Что из перечисленного требуется для работы Horizontal Pod Autoscaler?
- Созданный Custom Resource Definition
- Настроенный Custom API Server с определением данных для обращения к нему
- Специальный сервис, запущенный вне кластера

1. Какая команда используется для получения данных custom.metrics.k8s.io?
- kubectl describe
- kubectl get
- kubectl apply

1. Для чего используется ресурс типа APIService?
- Для описания новых ресурсов
- Для конфигурирования Kubernetes API Server
- Для определения дополнительного API Server