126Kubernetes 系列（一二零）KEDA：基于 Kubernetes 的事件驱动自动扩展

介绍

KEDA 它是一个开源项目，为 Kubernetes 工作负载提供事件驱动的自动扩展。它允许您根据从各种事件源接收到的事件数量自动扩展容器化应用程序。 KEDA 是一个云原生计算基金会 (CNCF) 沙箱项目，旨在补充 Kubernetes 的原生 Horizontal Pod Autoscaler (HPA)。

特点

1. 事件源集成：KEDA 支持多种事件源，例如 Apache Kafka、Azure 队列存储、RabbitMQ、Azure 事件中心等。它持续监视这些事件源中的传入事件。

2. 基于指标的扩展：KEDA 可以根据从事件源数据派生的自定义指标来扩展 Kubernetes Pod。它允许您根据事件数量、消息队列深度或其他相关指标定义自定义扩展规则。

3. 广泛的语言支持：KEDA 不依赖于特定的编程语言，使其适合用各种语言和框架编写的自动缩放应用程序。

4. 轻量级和非侵入性：KEDA 的设计是轻量级和非侵入性的，无需进行重大修改即可轻松与现有 Kubernetes 部署集成。

5. 可扩展性：通过创建自定义缩放器，KEDA 可以扩展以支持新的事件源和缩放行为。

6. 与外部缩放器集成：它还可以与 Prometheus 或 Azure Monitor 等外部系统集成，以利用其他指标进行缩放决策。

KEDA 在应用程序需要在事件驱动架构中处理事件的场景中特别有用，例如处理来自消息队列或事件流的消息的微服务。通过使用 KEDA，您可以确保 Kubernetes Pod 根据实时事件自动扩展和缩减，从而优化资源使用并提高应用程序响应能力。

配置

在 Kubernetes 中设置 KEDA 涉及部署 KEDA 控制器并为应用程序配置自定义扩展规则。以下是设置 KEDA 的步骤：

1. 安装 Helm （可选）

如果您的 Kubernetes 集群中没有安装 Helm，可以使用官方 Helm 安装说明进行安装：https://helm.sh/docs/intro/install/

2. 添加 KEDA Helm 仓库（可选）

如果尚未添加 KEDA Helm 存储库，您可以使用以下命令添加它：

```
helm repo add kedacore https://kedacore.github.io/charts
```

3. 安装 keda 控制器

```
helm install keda kedacore/keda --namespace keda
```

这将在 keda 命名空间中部署 KEDA 控制器。

4. 创建 Secret (可选)

如果您的事件源需要身份验证凭据，请创建包含必要凭据的 Kubernetes Secret。例如，如果您使用的是 Azure 存储队列，请使用存储帐户密钥创建一个 Secret。

```
kubectl create secret generic my-queue-secret --from-literal=storageAccountKey=<your-storage-account-key> --namespace <yo
```

5. 创建 keda 缩放对象

创建 KEDA 缩放对象来为您的应用程序配置自动缩放。缩放对象指定事件源、触发器元数据和缩放参数。

下面是 Kafka 主题、PostgreSQL、Prometheus 指标、CPU 和内存的 KEDA 缩放对象的示例 YAML 文件，您可以使用您需要的触发器并跳过其他触发器。

```
apiVersion: keda.k8s.io/v1alpha1
kind: ScaledObject
metadata:
  name: my-scaledobject
  namespace: <your-namespace>
spec:
  scaleTargetRef:
 deploymentName: <your-deployment-name>
  triggers:
  - type: kafka
    metadata:
      bootstrapServers: <kafka-broker-url>
      topic: <your-kafka-topic>
      consumerGroup: <your-consumer-group> .   # Make sure that this consumer group name is the same one as the one that
  triggers:
  - type: postgresql
    metadata:
      userName: "kedaUser"
      passwordFromEnv: PG_PASSWORD
      host: postgres-svc.namespace.cluster.local #use the cluster-wide namespace as KEDA
                                          #lives in a different namespace from your postgres
      port: "5432"
      dbName: test_db_name
      sslmode: disable
      query: "SELECT ceil(COUNT(*)::decimal / 16) FROM task_instance WHERE state='running' OR state='queued'"
      targetQueryValue: "2.2"
      metricName: backlog_process_count # DEPRECATED: This parameter is deprecated as of KEDA v2.10 and will be removed i
  triggers:
  - type: prometheus
    metadata:
      serverAddress: http://<prometheus-host>:9090
```

```
      metricName: http_requests_total # DEPRECATED: This parameter is deprecated as of KEDA v2.10 and will be removed in
      threshold: '100'
      query: sum(rate(http_requests_total{deployment="my-deployment"}[2m]))
  triggers:
  - type: cpu
    metricType: Utilization # Allowed types are 'Utilization' or 'AverageValue'
    metadata:
      type: Utilization # Deprecated in favor of trigger.metricType; allowed types are 'Utilization' or 'AverageValue'
      value: "60"
      containerName: "" # Optional. You can use this to target a specific container in a pod
  triggers:
  - type: memory
    metricType: Utilization # Allowed types are 'Utilization' or 'AverageValue'
    metadata:
      type: Utilization # Deprecated in favor of trigger.metricType; allowed types are 'Utilization' or 'AverageValue'
      value: "60"
      containerName: "" # Optional. You can use this to target a specific container in a pod
```

有关触发器 yaml 清单文件示例的参考，您可以使用 Keda 的官方文档：https://keda.sh/docs/2.10/scalers/

6. 应用缩放对象

应用缩放对象 YAML 文件来创建 KEDA ScaledObject：

```
kubectl apply -f path/to/scaledobject.yaml
```

7. 监控 KEDA 日志（可选）

您可以监控 KEDA 控制器日志以检查是否存在任何问题或扩展活动：

```
kubectl logs -n keda deployment/keda
```

8. 通过负载测试来验证 keda 功能

要在您的环境中测试 KEDA 设置，您需要执行负载测试，这将解释它在扩展时的行为方式。

您可以使用的工具如下所列：

Sr.no.	Request type	Tools
1.	APi load testing	K6 Tool , Gatling, Jmeter
2.	Database load testing (SQL and No-SQL)	Apache JMeter, Hammerdb
3.	Kafka Load testing.	Ktest , kafka Stress, Apache kafka Perf Test

就是这样！ KEDA 现在已在您的 Kubernetes 集群中设置，它将根据指定的事件源和 Scaled-object 中定义的缩放规则自动处理应用程序的自动缩放。确保监控应用程序的性能并根据需要调整扩展规则，以实现最佳资源利用率。