Health Monitoring Using Kubernetes

DESCRIPTION

Monitor cluster and application performance using Kubernetes.

**Background of the problem statement:**  
When working on a large scale dynamic applications, the number of requests/orders processed by the application per second are in lakhs. So, using Kubernetes cluster monitoring, determine if your application health and performance is up to date as expected.

**Features to be included:**

* Health Monitoring
* Resource Monitoring
* Scaling

**You must use the following:**

* **GKE Service Account with Valid Project**: to authenticate to Google Cloud Platform services
* **Google Cloud Shell**: a shell environment for managing resources hosted on Google Cloud Platform (GCP)
* **kubectl**: Kubernetes command-line tool, that allows you to run commands against Kubernetes clusters

**Following requirements should be met:**

* **Performance Monitoring** - Is my application up and running?
* **Resource Monitoring** - Does my application have the needed resources?
* **Scale up/Scale down**- Is my workload scaling up and down as necessary?
* Document the step-by-step process involved in completing this task.

This section will guide you to:

* Monitor the performance of your application and identify if it is up and running as expected.
* Monitor resources in terms of CPU/memory/disk so as to ensure that your application has enough resources required for its proper functioning.
* Scale up/scale down the application cluster size/no. of nodes.

**Step 1: Create a Kubernetes cluster**

**Step 2: Health/resource monitoring - extending infrastructure metrics**

In addition to application metrics that can be monitored using kubectl logs of running pods, Stackdriver custom metrics can also use measurements of your cluster's infrastructure not included in system metrics, such as container Disk I/O. You can deploy your own infrastructure monitoring agents to collect and push these metrics to Stackdriver.

**cAdvisor**

You can collect metrics using cAdvisor, the open source monitoring agent used in Kubernetes, to collect metrics. You can use prometheus-to-sd to push these metrics to Stackdriver.

To run cAdvisor on your own cluster, perform these steps:

a.) Clone cAdvisor

git clone <https://github.com/google/cadvisor.git>

cd cadvisor

b.) Install kustomize as shown below:

go get github.com/kubernetes-sigs/kustomize

c.) Create the example cAdvisor namespace and DaemonSet, which exports all container metrics which we are using to deploy our ecommerce site into that:

kustomize build deploy/kubernetes/overlays/examples | kubectl apply -f –

d.) Viewing metrics

**Step 3: Scale up/scale down application cluster size/number of nodes**

We will now resize clusters based on the number of nodes that increase as per the number of requests received from your ecommerce site. You can resize a cluster to increase or decrease the number of nodes in that cluster.

This will allow you to auto scale your application at runtime based upon the number of nodes a cluster requires to execute a particular number of requests/orders to handle large scale data applications.