6fusion Kubernetes Collector

version 1.0

Table of Contents

Overview

6fusion has a business engagement in the works with Redhat around metering their enterprise Platform-as-a-Service offering called **Openshift**, specifically the “Dedicated” version. This service relies on containers, which are managed by Kubernetes. The challenge that Redhat is currently facing is that they do not have a robust way to measure the actual consumption of these containers on Openshift. Redhat can utilize this metering technology to quantify this usage and allocate and charge customers accordingly.

The service is encapsulated in multiple Docker containers and Replica Controllers which they are managed by Kubernetes. This structure is integrated into the existing 6fusion meter architecture and the whole collector is an optional component of the meter.

When a Kubernetes collector is created it is automatically connected to cAdvisor to discover the following:

- Inventory of containers running
- Consumption of running containers

Once inventory and consumption information has been collected, it is then submitted to the appropriate API endpoints (on-prem API) in a 5-minute regular basis. The collector stores data for up to 20 minutes, providing for a variety of installation and integration scenarios.

Server Requirements

- A CoreOS operating system (recommended) or any Kubernetes compatible Linux distribution
- A Kubernetes cluster running properly with its containers, binaries, API Server and cAdvisor API

Local Computer Requirements

- Latest Kubernetes client kubectl
- Have the kubeconfig and all credential files required to access the Kubernetes instance of the server in any folder of your local computer
- Download this repository to your local computer

Collector Components

The Kubernetes collector is deployed as a virtual “appliance” (it is simple to deploy and configure). It consists of a series of sub-components and uses Kubernetes (http://kubernetes.io/) and container technologies to provide simplicity, flexibility, and scalability. Each collector component is a Kubernetes pod with one or more containers, depending on the component’s function. Each component is briefly described below.

Collector Datastore

The Kubernetes Collector datastore provides a cache for the inventory and consumption of running containers accepted via the API. The datastore is an instance of MongoDB. All data is stored for 20 minutes and can be configured.

Collector Pods

The 6fusion Kubernetes collector will create the following pods:

K8scollector Master Pod

This pod named 6fusion-collector-master will contain the following containers:

- **k8scollector-inventory**

  The container that collects the cluster inventory

- **k8scollector-onpremise**

  The container that connects to the 6fusion On Premise API to send the cluster data

- **k8scollector-cleancache**

  The container that performs a cache db clean of old and unused data in a regular basis

- **k8scollector-mongodb**

  The container that provides the cache db for the cluster data. The MongoDB cache in this pod will be exposed through a Kubernetes service called k8scollector-master on port 27017 so the separate Metrics container of this collector can connect to it and make the corresponding database operations required for the metrics collection.

K8scollector Metrics Pod

This pod named 6fusion-k8scollector-metrics will run as a Replication Controller so depending on the amount of containers running in the whole cluster, it can be scaled horizontally at any

time with the amount of replicas needed to satisfy the metrics collection in a short convenient amount of time. It contains the following container:

- **K8scollector-metrics**

  The container that collects the metrics of the machines in the cluster

On-premise API Data Model

The on-premise API is provided by a Rails application and is backed with a CouchDB instance as its data source. The WAC calculation is done by a specific, separate component written in GoLang. Each of these services lives in its own Docker container. This allows the meter to scale individual services as needed, without having to spin up multiple instances of the entire meter.

Note: *The organization will be created manually for each end-user (customer of Redhat) in on-premise API.*

Cache Data Model

The cache datastore is an instance of MongoDB and the Kubernetes collector uses the mongoid gem. Each end-user (customer of Redhat) will have an entire instance of the 6fusion collector. Within an end-user's meter, a single organization will be created manually in the 6fusion meter API and within each end-user's organization, a single infrastructure will be created automatically by the Kubernetes collector. All containers created by the end-user will be associated with that infrastructure.

Note: *There is no representation for Organizations, since it will be created manually for each end-user (customer of Redhat).*

This section provides a brief description of the objects that were created for kubernetes collector cache datastore:

Infrastructure

Infrastructures are defined as complete stacks of hardware that provide compute, network and storage services to applications. An infrastructure represents the smallest unit for which you want to calculate and report on capacity, utilization rate, and cost of production.

Within each end-user's organization, a **single** infrastructure will be created.

Machines

Machines are objects for which you send consumption samples. A machine could be a physical machine, virtual machine, cloud instance, or container. All containers created by the end-user will be associated with that infrastructure.

Disks

This will include every disk for which the inventory wants to report storage and disk IO consumption.

Nics

Likewise, every network that this machine utilizes should have a corresponding NIC entry.

Hosts

Hosts encapsulate the resources on which workloads are run. What is considered a host can vary, depending on the type of infrastructure. Hosts conform the kubernetes collector.

Pods

A pod is a group of one or more containers (such as Docker containers), the shared storage for those containers, and options about how to run the containers.

The kubernetes collector uses the official image of MongoDB (https://hub.docker.com/_/mongo/) and this will be contained in one pod with the k8scollector container.

Samples

Samples are collected on a 5-minute basis and consist of the cpu, memory, storage usage, and the disk, LAN, and WAN I/O. Samples encapsulate the metrics of consumption used by a machine:

- Machine Sample
- Disk Sample
- Nic Sample