Sizing of ONAP K8 Containers (Resource limit):-

Problem/Requirement:-

- As an ONAP Operator, We would like to have the control over designing the Capacity/Dimension of each container deployed over any Kubernetes host (Virtual or Physical), As Multiple onap component have different functionalities and therefore every component should have different resource requirements.
- By Default K8 assigns unlimited resources CPU/Memory for any specific container which is only hard limit with Host capabilities to handle it, Due to which any specific memory/High CPU Issues in any of the container can cause either complete Host to break down or scale out (if enabled) in a cloud environment,
- Kubernetes scheduler will also be unable to operate efficiently to put the containers over available hosts and also getting the root cause would also be tedious considering dynamic nature of resources available for all.

Solution:-

Concepts:-

- Resource Limit is the feature of Kubernetes which enables you to specify the CPU and memory of the container:-

  CPU and memory are collectively referred to as compute resources, or just resources. Compute resources are measurable quantities that can be requested, allocated, and consumed. They are distinct from API resources. API resources, such as Pods and Services are objects that can be read and modified through the Kubernetes API server

  CPU and memory are each a resource type. A resource type has a base unit. CPU is specified in units of cores, and memory is specified in units of bytes.

Meaning of CPU

Limits and requests for CPU resources are measured in *cpu* units. One cpu, in Kubernetes, is equivalent to:

- 1 AWS vCPU
- 1 GCP Core
- 1 Azure vCore
- 1 IBM vCPU
- 1 *Hyperthread* on a bare-metal Intel processor with Hyperthreading

Meaning of memory

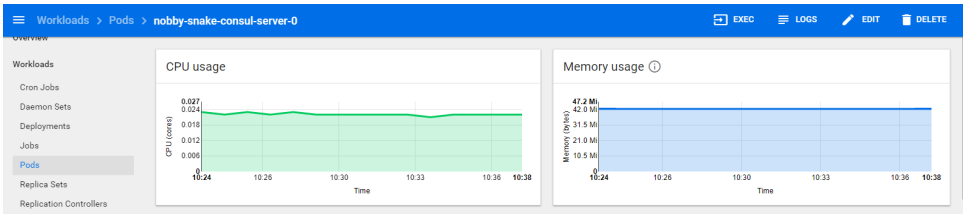
Limits and requests for `memory` are measured in bytes. You can express memory as a plain integer or as a fixed-point integer using one of these suffixes: E, P, T, G, M, K. You can also use the power-of-two equivalents: Ei, Pi, Ti, Gi, Mi, Ki. For example, the following represent roughly the same value:

KiB- ,KiloByte, MiB- MegaByte

Monitoring compute resource usage

By Default with Rancher installation, We can see the heapster installed under kube-system namespace

Heapster enables Container Cluster Monitoring and Performance Analysis for Kubernetes (versions v1.0.6 and higher), and platforms which include it.
Therefore, we can check the CPU/Memory stats over Kubernetes Dashboard as well as using kubectl top pod command. We can also check node resource capacity as well using kubectl top nodes as shown below:-



```
root@k8s-2:~# kubectl top pod nobby-snake-consul-server-0
NAME                           CPU(cores)   MEMORY(bytes)
nobby-snake-consul-server-0    22m          41Mi
root@k8s-2:~#
root@k8s-2:~#
root@k8s-2:~#
root@k8s-2:~# kubectl top nodes
NAME      CPU(cores)   CPU%      MEMORY(bytes)   MEMORY%
k8s-2     2951m        36%       37739Mi         84%
root@k8s-2:~#
```

- Ccoder, Weavescope are many other tools which can be used for testing the resource limit and performance of any running container in a realtime UI
based environment.

- When Containers have resource requests specified, the scheduler can make better decisions about which nodes(Hosts) to place Pods(Containers) on. And when Containers have their limits specified, contention for resources on a node(Host) can be handled in a specified manner such as Auto Scaling/replicas.
- This will enable us the Sizing of ONAP deployment even at high level(Currently its configurable for component level) That means we can deploy whatever we need with less and known resources capabilities.

Implementation:-

- We have implemented the Resource Limit Feature of Kubernetes with introducing the dynamics of flavors under it as well, such as small (dev) and Large (Prod) environment In Casablanca release, Same will enable any operator to use different flavors according to the requirement and also unlimited if no such resources restrictions needed.
- With Common flavor over-ride capabilities also introduced, we can overwrite any specific value for a container with a global values definition in common.
  Under values.yaml:-

```
# Resource Limit flavor -By Default using small
flavor: small
# Segregation for Different environment (Small and Large)
resources:
  small:
    limits:
      cpu: 2000m
      memory: 4Gi
    requests:
      cpu: 500m
      memory: 1Gi
  large:
    limits:
      cpu: 4000m
      memory: 8Gi
    requests:
      cpu: 1000m
      memory: 2Gi
  unlimited: {}
```

Outcome:-

- All of the ONAP Components were tested with their defined specific resources capacities that if Build is fine with that using various tools like weave scope and k8 to confirm the functionality, Same can be tested with any specific deployment and further can be configured/updated accordingly.

- So via this solution it also give us the pre known capacity values of all ONAP components deployed so that if we want to mix and match and cherry pick any of the components, we would be aware about the idea of resources capacity we need for that deployment.

References:-
https://kubernetes.io/docs/concepts/configuration/manage-compute-resources-container/

~~OOM-1145~~ **- Add Resource Limits to Helm Charts** CLOSED