CPU and Memory limits for pods/containers in AKS.

Deployment configuration –

resources:

          limits:

            cpu: 100m

            memory: 450Mi

          requests: Standard\_D2s\_v3

            cpu: 6m

            memory: 150Mi

In this configuration there are 2 major configurations. One is to define limits and another is to define requests.

Lilits- this defines the max usage capacity for the pods or for the containers. The given range is the maximum the resource can utilize.

Requests- this defines the minimum resource allocation for the resource.

In the above configuration we have defined that the minimum CPU allocated is 6m and the max can be used 100m.

What is use of this?

* If we don’t define max capacity for a pod or container. If that pod/ container have some issue that might consume more resources. So other pods may not have min requirement to run. That may create criticality and the application may not work as expected.
* This also may cause high resource utilization on node level and to crash the node.

We need to provide resource usage for each as required.

* For example, if you have multiple pods, you could distribute the CPU resources like this:

**Pod 1**:

Requests: 500m (0.5 CPU)

Limits: 1000m (1 CPU)

**Pod 2**:

Requests: 300m (0.3 CPU)

Limits: 600m (0.6 CPU)

**Pod 3**:

Requests: 200m (0.2 CPU)

Limits: 280m (0.28 CPU)

* This way, the total requests would be 1000m (1 CPU), and the total limits would be 1880m (1.88 CPUs), fitting within your available capacity.

What is 100 m?

* CPU resources in Kubernetes are measured in CPU units, not in MB or GB. Here's a quick breakdown:

**1 CPU** in Kubernetes is equivalent to 1 vCPU/Core for cloud providers or 1 hyperthread on bare-metal Intel processors.

**100 millicores (100m)** is equal to 0.1 CPU.

So, 100 millicores means the container can use up to 0.1 of a CPU core. This measurement is independent of memory, which is measured separately in MiB (Mebibytes) or GiB (Gibibytes)

How aks allocate cpu for a cluster?

* In Azure Kubernetes Service (AKS), a core or vCPU (virtual CPU) is equivalent to one physical CPU provided by azure.

AKS allocates CPUs to nodes based on the VM size you choose for your nodes. Each node in an AKS cluster is an Azure VM, and the number of vCPUs available depends on the VM size. For example, a Standard\_D2s\_v3 VM has 2 vCPUs and 8 GB of memory

A Standard\_D2s\_v3 VM in Azure has **2 vCPUs**. Here's how you can convert this to Kubernetes CPU units and set container usage limits:

* **1 vCPU** = 1 CPU in Kubernetes.
* **2 vCPUs** = 2 CPUs in Kubernetes.

However, remember that some CPU resources are reserved for the node's operating system and Kubernetes components. Typically, a small portion (e.g., 60 millicores for a 1-core node) is reserved.

**Example Calculation**

If you have 2 CPUs:

* Total available CPU for containers: 2 CPUs - reserved CPU.
* If 60 millicores are reserved per core, then for 2 cores, 120 millicores are reserved.
* Available CPU for containers: 2 CPUs - 0.12 CPUs = 1.88 CPUs (or 1880 millicores).

You can then distribute this available CPU among your containers based on their requests and limits.

To convert CPUs to millicores:

* **1 CPU** = 1000 millicores.

So, for 1.88 CPUs:

* (1.88 \text{ CPUs} \times 1000 \text{ millicores/CPU} = 1880 \text{ millicores}).

Therefore, 1.88 CPUs is equal to **1880 millicores**.

Memory allocation and usage is as usual it’s calculated by MB or by MIB (Mebibytes)

We can check total and divide as per requirement.

1 mib = 1.048576 MB

1 mb = 0.95367432