ResourceQuota is used to limit the usage of resources within a namespace. It helps administrators control and restrict the allocation of cluster resources to ensure fairness and controlled resource distribution among namespaces.

In Harvester, ResourceQuota can define usage limits for the following resources:

- **CPU:** Limits compute resource usage, including CPU cores and CPU time.
- **Memory:** Limits the usage of memory resources in bytes or other recognizable memory units.
- **Storage:** Limits the usage of storage resources.

Set ResourceQuota via Rancher

In the Rancher UI, administrators can configure resource quotas for namespaces through the following steps:

1. Click the hamburger menu and choose the **Virtualization Management** tab.
2. Choose one of the clusters and go to **Projects/Namespaces** > **Create Project**.
3. Specify the desired project **Name**. Next, go to the **Resource Quotas** tab and select the **Add Resource** option. Within the **Resource Type** field, select either **CPU Limit** or **Memory Limit** and define the **Project Limit** and **Namespace Default Limit** values.

You can configure the **Namespace** limits as follows:

1. Find the newly created project, and select **Create Namespace**.
2. Specify the desired namespace **Name**, and adjust the limits.
3. Complete the process by selecting **Create**.

:::note Attempts to provision VMs for guest clusters are blocked when the resource quotas are reached. Rancher responds by creating a new VM in a loop, in which each failed attempt to create a VM is immediately followed by another creation attempt. This results in a transient error state in the cluster that is not recorded as the VM is recreated. :::

:::important

Due to the Overhead Memory of Virtual Machine, each VM needs some additional memory to work. When setting **Memory Limit**, this should be taken into account. For example, when the project **Memory Limit** is `24 Gi`, it is not possible to run 3 VMs each has `8 Gi` memory.

:::

Overhead Memory of Virtual Machine

Upon creating a virtual machine (VM), the VM controller seamlessly incorporates overhead resources into the VM's configuration. These additional resources intend to guarantee the consistent and uninterrupted functioning of the VM. It's important to note that configuring memory limits requires a higher memory reservation due to the inclusion of these overhead resources.

For example, consider the creation of a new VM with the following configuration:

- CPU: 8 cores
- Memory: 16Gi

:::note The operating system, either Linux or Windows, does not affect overhead calculations. :::

Memory Overhead is calculated in the following sections:

- **Memory PageTables Overhead:** This accounts for one bit for every 512b RAM size. For instance, a memory of 16Gi requires an overhead of 32Mi.
- **VM Fixed Overhead:** This consists of several components:
  - `VirtLauncherMonitorOverhead` : 25Mi (the `ps` RSS for virt-launcher-monitor)
  - `VirtLauncherOverhead` : 100Mi (the `ps` RSS for the virt-launcher process)
  - `VirtlogdOverhead` : 20Mi (the `ps` RSS for virtlogd)
  - `VirtqemudOverhead` : 35Mi (the `ps` RSS for virtqemud)
  - `QemuOverhead` : 30Mi (the `ps` RSS for qemu, minus the RAM of its (stressed) guest, minus the virtual page table)
- **8Mi per CPU (vCPU) Overhead:** Additionally, 8Mi of overhead per vCPU is added, along with a fixed 8Mi overhead for IOThread.
- **Extra Added Overhead:** This encompasses various factors like video RAM overhead and architecture overhead. Refer to Additional Overhead for further details.
- **additional-guest-memory-overhead-ratio** User can further tune the `Memory Overhead` by the Harvester setting additional-guest-memory-overhead-ratio, which defaults to `"1.5"` . This setting is important for VM to eliminate the chance to hit OOM(Out of Memory).

This calculation demonstrates that the VM instance necessitates an additional memory overhead of approximately 380Mi.

For more information, see Memory Overhead.

For more information on how the memory overhead is calculated in Kubevirt, refer to the source code GetMemoryOverhead.

:::note

The `Overhead Memory` varies between different Harvester releases (with different Kubevirt releases) because all those backing components are keeping adding new features and fixing bugs, they need more memory.

:::

Automatic adjustment of ResourceQuota during migration

When the allocated resource quota controlled by the `ResourceQuota` object reaches its limit, migrating a VM becomes unfeasible. The migration process automatically creates a new pod mirroring the resource requirements of the source VM. If these pod creation prerequisites surpass the defined quota, the migration operation cannot proceed.

Available as of v1.2.0

In Harvester, the `ResourceQuota` values will dynamically expand ahead of migration to accommodate the resource needs of the target virtual machine. After migration, the ResourceQuotas will be reinstated to their prior configurations.

Please be aware of the following constrains of the automatic resizing of `ResourceQuota` :

- `ResourceQuota` cannot be changed during VM migration.
- When raising the `ResourceQuota` value, if you create, start, or restore other VMs, Harvester will verify if the resources are sufficient based on the original `ResourceQuota` . If the conditions are not met, the system will alert that the migration process is not feasible.
- After expanding `ResourceQuota` , potential resource contention may occur between non-VM pods and VM pods, leading to migration failures. Therefore, deploying custom container workloads and

VMs to the same namespace is not recommended.

- Due to the concurrent limitation of the webhook validator, the VM controller will execute a secondary validation to confirm resource sufficiency. If the resource is insufficient, it will auto config the VM's `RunStrategy` to `Halted`, and a new annotation `harvesterhci.io/insufficient-resource-quota` will be added to the VM object, informing

  you that the VM was shut down due to insufficient resources.

Disable automatic adjustment of ResourceQuota during migration

Available as of v1.4.2

When a `ResourceQuota` object has the annotation `harvesterhci.io/skipResourceQuotaAutoScaling: "true"`, Harvester does not automatically adjust the values of that object. This feature is useful for debugging, troubleshooting and other tasks.

:::info important

You must set the annotation before the migration starts. If the annotation is set while the values are already being adjusted, Harvester is unable to automatically restore the previous configuration.

:::