Live migration means moving a virtual machine to a different host without downtime.

:::note

- Live migration is not allowed when the virtual machine is using a management network of bridge interface type.
- Live migration is not allowed when the virtual machine has any volume of the `CD-ROM` type. Such volumes should be ejected before live migration.
- Live migration is not allowed when the virtual machine has any volume of the `Container Disk` type. Such volumes should be removed before live migration.
- Live migration is not allowed when the virtual machine has any `PCIDevice` passthrough enabled. Such devices need to be removed before live migration.
- Live migration is not allowed when the volumeAccessMode of any volume in the virtual machine is `ReadWriteOnce`. Such volumes should be removed before live migration.

:::

How Migration Works

Each node has multiple CPU models that are labeled with different keys.

- Primary CPU model: `host-model-cpu.node.kubevirt.io/{cpu-model}`
- Supported CPU models: `cpu-model.node.kubevirt.io/{cpu-model}`
- Supported CPU models for migration: `cpu-model-migration.node.kubevirt.io/{cpu-model}`

During live migration, the system checks the value of `spec.domain.cpu.model` in the VirtualMachineInstance (VMI) CR, which is derived from `spec.template.spec.domain.cpu.model` in the VirtualMachine (VM) CR. If the value of `spec.template.spec.domain.cpu.model` is not set, the system uses the default value `host-model`.

When `host-model` is used, the process fetches the value of the primary CPU model and fills `spec.NodeSelectors` of the newly created pod with the label `cpu-model-migration.node.kubevirt.io/{cpu-model}`.

Alternatively, you can customize the CPU model in `spec.domain.cpu.model`. For example, if the CPU model is `XYZ`, the process fills `spec.NodeSelectors` of the newly created pod with the label `cpu-model.node.kubevirt.io/XYZ`.

However, `host-model` only allows migration of the VM to a node with same CPU model. For more information, see Limitations.

Starting a Migration

1. Go to the **Virtual Machines** page.
2. Find the virtual machine that you want to migrate and select ⋮ **> Migrate**.
3. Choose the node to which you want to migrate the virtual machine. Click **Apply**.

When you have node scheduling rules configured for a VM, you must ensure that the target nodes you are migrating to meet the VM's runtime requirements. The list of nodes you get to search and select from will be generated based on:

- VM scheduling rules.

- Possibly node rules from the network configuration.

Aborting a Migration

1. Go to the **Virtual Machines** page.
2. Find the virtual machine in migrating status that you want to abort. Select ⋮ **> Abort Migration**.

Migration Timeouts

Completion Timeout

The live migration process will copy virtual machine memory pages and disk blocks to the destination. In some cases, the virtual machine can write to different memory pages or disk blocks at a higher rate than these can be copied. As a result, the migration process is prevented from being completed in a reasonable amount of time.

Live migration will be aborted if it exceeds the completion timeout of 800s per GiB of data. For example, a virtual machine with 8 GiB of memory will time out after 6400 seconds.

Progress Timeout

Live migration will also be aborted when copying memory doesn't make any progress in 150s.

Limitation

`host-model` only allows migration of the VM to a node with same CPU model. However, specifying a CPU model is not always required. When no CPU model is specified, you must shut down the VM, assign a CPU model that is supported by all nodes, and then restart the VM.

Example:

- A node: `host-model-cpu.node.kubevirt.io/XYZ` `cpu-model-migration.node.kubevirt.io/XYZ` `cpu-model.node.kubevirt.io/123`
- B node: `host-model-cpu.node.kubevirt.io/ABC` `cpu-model-migration.node.kubevirt.io/ABC` `cpu-model.node.kubevirt.io/123`

Migrating a VM with `host-model` is not possible because the values of `host-model-cpu.node.kubevirt.io` are not identical. However, both nodes support the `123` CPU model, so you can migrate any VM with the `123` CPU model using either of the following methods:

- Cluster level: Run `kubectl edit kubevirts.kubevirt.io -n harvester-system` and add `spec.configuration.cpuModel: "123"`. This change also affects newly created VMs.
- Individual VMs: Modify the VM configuration to include `spec.template.spec.domain.cpu.model: "123"`.

Both methods require the restarting the VMs. If you are certain that all nodes in the cluster support a specific CPU model, you can define this at the cluster level before creating any VMs. In doing so, you eliminate the need to restart the VMs (to assign the CPU model) during live migration.