Replace Ethernet NICs

You may want to replace the Ethernet NICs of a bare-metal node in a Harvester cluster for various reasons, including the following:

- Malfunction or damage
- Insufficent hardware capacity
- Missing features

You can follow the steps below and run them in each node step by step.

Pre-Replacement Checks

1. Verify that the installed Harvester version supports the new NICs.
2. Test the new NICs in non-production environment.
3. On the **Virtual Machines** screen of the Harvester UI, verify that the status of all VMs is either *Running* or *Stopped*.
4. On the embedded Longhorn dashboard, verify that the status of all Longhorn volumes is *Healthy*.
5. (Optional) On the **Harvester Support** screen, generate a support bundle for comparison purposes.

Collect Information

Before any action is taken, it is important to collect the current network information and status.

- Harvester network configuration: By default, Harvester creates a bond interface named `mgmt-bo` for the management network and one new bond interface for each cluster network. Harvester saves network configuration details in the file `/oem/90_custom.yaml`.

  Example: A NIC named `ens3` was added to the `mgmt-bo` bond interface.

  ```
  - path: /etc/sysconfig/network/ifcfg-mgmt-bo
    permissions: 384
    owner: 0
    group: 0
    content: |+
      STARTMODE='onboot'
      BONDING_MASTER='yes'
      BOOTPROTO='none'
      POST_UP_SCRIPT="wicked:setup_bond.sh"
      BONDING_SLAVE_0='ens3'
      BONDING_MODULE_OPTS='miimon=100 mode=active-backup '
      DHCLIENT_SET_DEFAULT_ROUTE='no'
    encoding: ""
    ownerstring: ""

  - path: /etc/sysconfig/network/ifcfg-ens3
    permissions: 384
    owner: 0
    group: 0
  ```

```
  content: |
    STARTMODE='hotplug'
    BOOTPROTO='none'
  encoding: ""
  ownerstring: ""
```

- Physical NICs: You can use the command `ip link` to retrieve related information, including the state of each NIC and the corresponding master (if applicable).

  Example:

  ```
  $ ip link | grep master -1

  2: ens3: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
  master mgmt-bo state UP mode DEFAULT group default qlen 1000
      link/ether 52:54:00:03:3a:e4 brd ff:ff:ff:ff:ff:ff
  --

  4: mgmt-bo: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc noqueue
  master mgmt-br state UP mode DEFAULT group default qlen 1000
      link/ether 52:54:00:03:3a:e4 brd ff:ff:ff:ff:ff:ff
  ```

- PCI devices: You can use the command `lspci` to retrieve a list of devices, which allows you to quickly identify the network NICs. To retrieve detailed information about each device, use the command `lspci -v`.

  Example (`lspci`):

  ```
  $ lspci
  00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet
  Controller (rev 03)
  ```

  Example (`lspci -v`):

  ```
  $ lspci -v
  00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet
  Controller (rev 03)
    Subsystem: Red Hat, Inc. QEMU Virtual Machine
    Physical Slot: 3
    Flags: bus master, fast devsel, latency 0, IRQ 11
    Memory at fc080000 (32-bit, non-prefetchable) [size=128K]
    I/O ports at c000 [size=64]
    Expansion ROM at fc000000 [disabled] [size=512K]
    Kernel driver in use: e1000
    Kernel modules: e1000
  ```

- Linux kernel log: You can use the command `dmesg` to display kernel messages, which include most of the required information. If you save the messages to `kernel.log`, you can check the driver and link status.

  Harvester places sub-NICs into the bond interfaces. In the following example, an additional bond interface named `data-bo` is created in the cluster.

```
$ grep "(slave" kernel.log  (or: dmesg | grep "(slave")

Jan 08 00:35:00 localhost kernel: mgmt-bo: (slave eno5): Enslaving as a backup interface with an up link
Jan 08 00:35:00 localhost kernel: mgmt-bo: (slave ens4f0): Enslaving as a backup interface with an up link
Jan 08 00:37:34 localhost kernel: data-bo: (slave eno6): Enslaving as a backup interface with an up link
Jan 08 00:37:35 localhost kernel: data-bo: (slave ens4f1): Enslaving as a backup interface with an up link
```

The NICs are renamed.

```
$ grep "renamed" kernel.log

Jan 08 00:34:48 localhost kernel: tg3 0000:02:00.0 eno1: renamed from eth2 // eth2 / eno1 is not used by Harvester
Jan 08 00:34:48 localhost kernel: tg3 0000:02:00.3 eno4: renamed from eth6 // eth6 / eno4 is not used by Harvester
Jan 08 00:34:48 localhost kernel: tg3 0000:02:00.2 eno3: renamed from eth5 // eth5 / eno3 is not used by Harvester
Jan 08 00:34:48 localhost kernel: tg3 0000:02:00.1 eno2: renamed from eth3 // eth3 / eno2 is not used by Harvester
Jan 08 00:34:49 localhost kernel: i40e 0000:5d:00.0 eno5: renamed from eth0
Jan 08 00:34:49 localhost kernel: i40e 0000:af:00.0 ens4f0: renamed from eth4
Jan 08 00:34:49 localhost kernel: i40e 0000:5d:00.1 eno6: renamed from eth1
Jan 08 00:34:49 localhost kernel: i40e 0000:af:00.1 ens4f1: renamed from eth2
```

The NIC driver of `eno5(0000:5d:00.0)` is `(intel) i40e 10Gbps Full Duplex`.

```
$ grep "0000:5d:00.0" kernel.log

Jan 08 00:34:47 localhost kernel: i40e 0000:5d:00.0: fw 8.71.63306 api 1.11 nvm 10.54.7 [8086:1572] [103c:22fc]
Jan 08 00:34:47 localhost kernel: i40e 0000:5d:00.0: MAC address: 48:df:37:24:c2:00
Jan 08 00:34:47 localhost kernel: i40e 0000:5d:00.0: FW LLDP is enabled
Jan 08 00:34:47 localhost kernel: i40e 0000:5d:00.0 eth0: NIC Link is Up, 10 Gbps Full Duplex, Flow Control: None
Jan 08 00:34:47 localhost kernel: i40e 0000:5d:00.0: PCI-Express: Speed 8.0GT/s Width x8
Jan 08 00:34:47 localhost kernel: i40e 0000:5d:00.0: Features: PF-id[0] VFs: 64 VSIs: 66 QP: 112 RSS FD_ATR FD_SB NTUPLE DCB VxLAN Geneve PTP VEPA
Jan 08 00:34:49 localhost kernel: i40e 0000:5d:00.0 eno5: renamed from eth0
```

The enabled NICs are detected.

```
$ grep "is Up" kernel.log

Jan 08 00:34:47 localhost kernel: i40e 0000:5d:00.0 eth0: NIC Link is Up, 10 Gbps Full Duplex, Flow Control: None
Jan 08 00:34:48 localhost kernel: i40e 0000:5d:00.1 eth1: NIC Link is Up, 10
```

```
Gbps Full Duplex, Flow Control: None
Jan 08 00:34:48 localhost kernel: i40e 0000:af:00.0 eth4: NIC Link is Up, 10
Gbps Full Duplex, Flow Control: None
Jan 08 00:34:49 localhost kernel: i40e 0000:af:00.1 eth2: NIC Link is Up, 10
Gbps Full Duplex, Flow Control: None
```

Enable Maintenance Mode

1. (Optional) Stop VMs that cannot or must not be migrated.
2. Enable maintenance mode on the target node to automatically migrate all VMs to other nodes.

- Wait for everything to become ready, and then repeat the steps in the Pre-check section.
- Manually stop a VM in the following situations:
    - The VM fails to migrate.
    - The VM has selectors that prevent it from migrating to other nodes.
    - The VM has special hardware (for example, PCI passthrough or vGPUs) that prevent it from migrating to other nodes.

(Optional) Update the Network Config

There are one or more Network Config under every Cluster Network on Harvester. Each `Network Config` is backed by a `VlanConfig` CRD object.

:::info important

Updating the `Network Config` is **required** if the new NICs will be placed in different physical slots or will have different uplink parameters.

:::

1. Check the node.

   When a Harvester cluster node belongs to a `Network Config`, the `Node` object has a label with the key `network.harvesterhci.io/vlanconfig`.

   Example:

   ```
   apiVersion: v1
   kind: Node
   metadata:
     labels:
       ...
       network.harvesterhci.io/vlanconfig: vlan123
   ```

2. Remove this node from the `Network Config`.

   When the new NICs are placed in different slots, you must change the `Network Config` to exclude this node. You can delete the VlanConfig if the `Network Config` object selects only this node from `nodeSelector`.

   Example:

```
apiVersion: network.harvesterhci.io/v1beta1
kind: VlanConfig
spec:
  clusterNetwork: data
  nodeSelector:
    kubernetes.io/hostname: node123  // select one or more nodes
  uplink:
    bondOptions:
      miimon: 100
      mode: 802.3ad
    linkAttributes:
      mtu: 1500
      txQLen: -1
    nics:
    - enp0s1
    - enp0s2
```

When VMs are still running on an affected node, the network webhook returns an error.

3. Check the `Node` object.

   Depending on the situation, either the label `network.harvesterhci.io/vlanconfig` changes or is removed.

4. Check the `VlanStatus` object.

   Depending on the situation, either the status of the `VlanStatus` object's `ready` condition changes to `"True"` or the object is deleted.

   Example:

   ```
   apiVersion: network.harvesterhci.io/v1beta1
   kind: VlanStatus
   metadata:
   ...
   status:
     clusterNetwork: data
     conditions:
     - lastUpdateTime: "2024-02-03T18:32:41Z"
       status: "True"
       type: ready
     linkMonitor: public
     localAreas:
     - cidr: 10.190.186.0/24
       vlanID: 2013
     node: node123
     vlanConfig: vlan123
   ```

(Optional) Drain the Node

You may find that some Longhorn replicas remain active on the node even after completing the previously outlined procedures.

1. Drain the node. (This is optional in Harvester.)

- Scenario 1: The `numReplicas` value of all volumes is `3`, which means that each Longhorn volume has three active replicas.

  The Longhorn Engine recognizes that it can no longer communicate with the replica on the drained node, and then marks that replica as failed. None of the replicas hold any special significance to Longhorn so it functions as long as it can communicate with at least one replica.

- Scenario 2: Some Longhorn volumes have *fewer* than three active replicas, or you manually attached volumes using the Harvester UI or Longhorn UI.

  You must manually detach the replicas or move them to other nodes, and then drain the node using the command `kubectl drain --ignore-daemonsets <node name>`. The option `--ignore-daemonsets` is required because Longhorn deploys daemonsets such as Longhorn Manager, Longhorn CSI plugin, and Longhorn Engine image.

  Replicas running on the node are stopped and marked as `Failed`. Longhorn Engine processes running on the node are migrated with the pod to other nodes. Once the node is fully drained, no replicas and engine processes should remain running on the node.

2. Replenish replicas.

   After a node is shut down, Longhorn does not start rebuilding the replicas on other nodes until the `replica-replenishment-wait-interval` (default value: 600 seconds) is reached. If the node comes back online before the wait interval value is reached, Longhorn reuses the replicas. Otherwise, Longhorn rebuilds the replicas on another node.

   During system maintenance, you can modify the `replica-replenishment-wait-interval` value using the embedded Longhorn UI to enable faster replica rebuilding.

   Harvester v1.3.0 uses Longhorn v1.6.0, while Harvester v1.2.1 uses Longhorn v1.4.3.

Replace the Nics

1. Shut the node down.

2. Replace the NICs.

3. Restart the node.

4. Collect information about the current network configuration and status.

If you observe any abnormalities, generate a support bundle for troubleshooting purposes.

(Optional) Update the Network Config Again

:::info important

Updating the `Network Config` is **required** if the new NICs will be placed in different physical slots.

:::

1. Add the node to the `Network Config`.

   You must create a new `Network Config` or change the `Network Config` to include this node.

2. Check the `Node` object.

   The label `network.harvesterhci.io/vlanconfig` reflects the specific `Network Config` used.

3. Check the `VlanStatus` object.

   The status of the `VlanStatus` object's `ready` condition changes to `"True"` .

Disable Maintenance Mode

1. Wait for the node to be moved back to the cluster.

2. Disable maintenance mode.

3. (Optional) Start the VMs that you manually stopped.

4. (Optional) Manually migrate VMs to this node.

Troubleshooting

Harvester uses multiple network-related pods and CRDs. When troubleshooting, check the pod logs and the status of CRD objects.

Pods:

```
$ kubectl get pods -n harvester-system
NAME                                                    READY   STATUS    RESTARTS
AGE
harvester-network-controller-cnf22                      1/1     Running   2 (60m ago)
3d22h  // Network controller agent daemonSet, deployed in each node
harvester-network-controller-manager-859c4bd874-xcllf   1/1     Running   2 (60m ago)
3d22h  // Network controller
harvester-network-webhook-56b877d5d5-z42dp              1/1     Running   2 (60m ago)
3d22h
```

CRDs:

```
clusternetworks.network.harvesterhci.io
linkmonitors.network.harvesterhci.io
vlanconfigs.network.harvesterhci.io
vlanstatuses.network.harvesterhci.io
```