Red Hat Enterprise Linux 9

Deploying RHEL 9 on Microsoft Azure

Obtaining RHEL system images and creating RHEL instances on Azure

Last Updated: 2023-05-10

Red Hat Enterprise Linux 9 Deploying RHEL 9 on Microsoft Azure

Obtaining RHEL system images and creating RHEL instances on Azure

Legal Notice

Abstract

To use Red Hat Enterprise Linux (RHEL) in a public cloud environment, you can create and deploy RHEL system images on various cloud platforms, including Microsoft Azure. You can also create and configure a Red Hat High Availability (HA) cluster on Azure. The following chapters provide instructions for creating cloud RHEL instances and HA clusters on Azure. These processes include installing the required packages and agents, configuring fencing, and installing network resource agents.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE 3

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION 4

CHAPTER 1. PUSHING VHD IMAGES TO MICROSOFT AZURE CLOUD USING THE GUI IMAGE BUILDER TOOL 5

CHAPTER 2. DEPLOYING A RED HAT ENTERPRISE LINUX IMAGE AS A VIRTUAL MACHINE ON MICROSOFT AZURE 8
2.1. RED HAT ENTERPRISE LINUX IMAGE OPTIONS ON AZURE 8
2.2. UNDERSTANDING BASE IMAGES 10
2.2.1. Using a custom base image 10
2.2.2. Required system packages 10
2.2.3. Azure VM configuration settings 11
2.2.4. Creating a base image from an ISO image 11
2.3. CONFIGURING A CUSTOM BASE IMAGE FOR MICROSOFT AZURE 12
2.3.1. Installing Hyper-V device drivers 13
2.3.2. Making configuration changes required for a Microsoft Azure deployment 14
2.4. CONVERTING THE IMAGE TO A FIXED VHD FORMAT 16
2.5. INSTALLING THE AZURE CLI 17
2.6. CREATING RESOURCES IN AZURE 18
2.7. UPLOADING AND CREATING AN AZURE IMAGE 21
2.8. CREATING AND STARTING THE VM IN AZURE 22
2.9. OTHER AUTHENTICATION METHODS 24
2.10. ATTACHING RED HAT SUBSCRIPTIONS 24
2.11. SETTING UP AUTOMATIC REGISTRATION ON AZURE GOLD IMAGES 25
2.12. ADDITIONAL RESOURCES 26

CHAPTER 3. CONFIGURING A RED HAT HIGH AVAILABILITY CLUSTER ON MICROSOFT AZURE 27
3.1. CREATING RESOURCES IN AZURE 27
3.2. REQUIRED SYSTEM PACKAGES FOR HIGH AVAILABILITY 31
3.3. AZURE VM CONFIGURATION SETTINGS 32
3.4. INSTALLING HYPER-V DEVICE DRIVERS 32
3.5. MAKING CONFIGURATION CHANGES REQUIRED FOR A MICROSOFT AZURE DEPLOYMENT 33
3.6. CREATING AN AZURE ACTIVE DIRECTORY APPLICATION 36
3.7. CONVERTING THE IMAGE TO A FIXED VHD FORMAT 37
3.8. UPLOADING AND CREATING AN AZURE IMAGE 38
3.9. INSTALLING RED HAT HA PACKAGES AND AGENTS 39
3.10. CREATING A CLUSTER 41
3.11. FENCING OVERVIEW 42
3.12. CREATING A FENCING DEVICE 42
3.13. CREATING AN AZURE INTERNAL LOAD BALANCER 44
3.14. CONFIGURING THE LOAD BALANCER RESOURCE AGENT 45
3.15. CONFIGURING SHARED BLOCK STORAGE 46
3.16. ADDITIONAL RESOURCES 51

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Submitting comments on specific passages

1. View the documentation in the **Multi-page HTML** format and ensure that you see the **Feedback** button in the upper right corner after the page fully loads.
2. Use your cursor to highlight the part of the text that you want to comment on.
3. Click the **Add Feedback** button that appears near the highlighted text.
4. Add your feedback and click **Submit**.

Submitting feedback through Bugzilla (account required)

1. Log in to the Bugzilla website.
2. Select the correct version from the **Version** menu.
3. Enter a descriptive title in the **Summary** field.
4. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
5. Click **Submit Bug**.

CHAPTER 1. PUSHING VHD IMAGES TO MICROSOFT AZURE CLOUD USING THE GUI IMAGE BUILDER TOOL

You can create **.vhd** images using image builder. Then, you can push the **.vhd** images to a Blob Storage of the Microsoft Azure Cloud service provider.

Prerequisites

- You have root access to the system.
- You have opened the image builder interface of the RHEL web console in a browser.
- You created a blueprint. See Creating an image builder blueprint in the web console interface .
- You have a Microsoft Storage Account created.
- You have a writable Blob Storage prepared.

Procedure

1. For the **blueprint name**, click the **Images** tab .
2. Click **Create Image** to create your customized image.
   A pop-up window opens.
   a. From the **"Type** drop-down menu list, select the **Azure Disk Image (.vhd)** image.
   b. Check the **Upload to Microsoft Azure** check box to upload your image to the Microsoft Azure Cloud and click **Next**.
   c. To authenticate your access to Microsoft Azure, type your "Storage account" and "Storage access key" in the corresponding fields. Click **Next**.
   You can find your Microsoft Storage account details in the Settings→Access Key menu list.
   d. Type a **"Image name"** to be used for the image file that will be uploaded and the Blob "Storage container" in which the image file you want to push the image into. Click **Next**.
   e. Review the information you provided and click **Finish**.
   Optionally, you can click **Back** to modify any incorrect detail.
3. When the image creation process starts, a small pop-up on the upper right side displays with the message: **Image creation has been added to the queue**.
   After the image process creation is complete, click the blueprint you created the image from. In the **Images** tab, you can see the **Image build complete** status for the image you created.
4. To access the image you pushed into **Microsoft Azure Cloud**, access the **Microsoft Azure portal**.
5. On the search bar, type **Images** and select the first entry under **Services**. You are redirected to the **Image dashboard**.
6. Click **+Add**. You are redirected to the **Create an Image** dashboard.
   Insert the below details:
   a. **Name**: Choose a name for your new image.

b. **Resource Group**: Select a **resource group**.

c. **Location**: Select the **location** that matches the regions assigned to your storage account. Otherwise you will not be able to select a blob.

d. **OS Type**: Set the operating system type to **Linux**.

e. **VM Generation**: Keep the VM generation set on **Gen 1**.

f. **Storage Blob**: Click **Browse** on the right of **Storage blob input**. Use the dialog to find the image you uploaded earlier.
Keep the remaining fields as in the default choice.

7. Click **Create** to create the image. After the image is created, you can see the message **Successfully created image** in the upper right corner.

8. Click **Refresh** to see your newly created image and open it.

9. Click **+ Create VM**. You are redirected to the **Create a virtual machine** dashboard.

10. In the **Basic** tab, under **Project Details**, your **Subscription** and the **Resource Group** are already pre-set.
If you want to create a new **Resource Group**:

a. Click **Create new**.
A pop-up prompts you to create the **Resource Group Name** container.

b. Insert a name and click **OK**.
If you want to keep the already pre-set **Resource Group**:

11. Under **Instance Details**, enter:

a. **Virtual machine name**

b. **Region**

c. **Image**: The image you created is pre-selected by default.

d. **Size**: Choose a VM size that better suits your needs.
Keep the remaining fields default.

12. Under **Administrator account**, enter the below details:

a. **Username**: the name of the account administrator.

b. **SSH public key source**: from the drop-down menu, select **Generate new key pair**.
You can either use the key pair you already have or you can create a new key pair.
Alternatively, you can use **image builder** to add a user to the image with a preset public key.
See Creating a user account with SSH key for more details.

c. **Key pair name**: insert a name for the key pair.

13. Under **Inbound port rules**, select values for each of the fields:

a. **Public inbound ports**: **Allow selected ports**.

b. **Select inbound ports**: Use the default set **SSH (22)**.

14. Click **Review + Create**. You are redirected to the **Review + create** tab and receive a confirmation that the validation passed.

15. Review the details and click **Create**.
Optionally, you can click **Previous** to fix previous options selected.

16. A **generates new key pair** window opens. Click **Download private key and create resources**. Save the key file as "*yourKey*.pem".

17. After the deployment is complete, click **Go to resource**.

18. You are redirected to a new window with your VM details. Select the public IP address on the upper right side of the page and copy it to your clipboard.

Now, to create an SSH connection with the VM to connect to the Virtual Machine.

1. Open a terminal.

2. At your prompt, open an SSH connection to your VM. Replace the IP address with the one from your VM, and replace the path to the .pem with the path to where the key file was downloaded.

```
# ssh -i ./Downloads/yourKey.pem azureuser@10.111.12.123
```

3. You are required to confirm if you want to continue to connect. Type **yes** to continue.

As a result, the output image you pushed to the Microsoft Azure Storage Blob is ready to be provisioned.

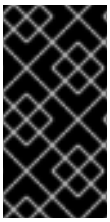
Additional resources

- Microsoft Azure Storage Documentation .
- Create a Microsoft Azure Storage account .
- Open a case on Red Hat Customer Portal .
- Help + support.
- Contacting Red Hat.

CHAPTER 2. DEPLOYING A RED HAT ENTERPRISE LINUX IMAGE AS A VIRTUAL MACHINE ON MICROSOFT AZURE

To deploy a Red Hat Enterprise Linux 9 (RHEL 9) image on Microsoft Azure, follow the information below. This chapter:

- Discusses your options for choosing an image
- Lists or refers to system requirements for your host system and virtual machine (VM)
- Provides procedures for creating a custom VM from an ISO image, uploading it to Azure, and launching an Azure VM instance



IMPORTANT

You can create a custom VM from an ISO image, but Red Hat recommends that you use the *Red Hat Image Builder* product to create customized images for use on specific cloud providers. With Image Builder, you can create and upload an Azure Disk Image (VHD format). See Composing a Customized RHEL System Image for more information.

For a list of Red Hat products that you can use securely on Azure, refer to Red Hat on Microsoft Azure.

Prerequisites

- Sign up for a Red Hat Customer Portal account.
- Sign up for a Microsoft Azure account.
- Enable your subscriptions in the Red Hat Cloud Access program. The Red Hat Cloud Access program allows you to move your Red Hat subscriptions from physical or on-premise systems to Azure with full support from Red Hat.

2.1. RED HAT ENTERPRISE LINUX IMAGE OPTIONS ON AZURE

The following table lists image choices for RHEL 9 on Microsoft Azure, and notes the differences in the image options.

Table 2.1. Image options

Image option	Subscriptions	Sample scenario	Considerations

Image option	Subscriptions	Sample scenario	Considerations
Choose to deploy a Red Hat Gold Image.	Use your existing Red Hat subscriptions.	Enable subscriptions through the Red Hat Cloud Access program, and then choose a Red Hat Gold Image on Azure. See the Red Hat Cloud Access Reference Guide for details on Gold Images and how to access them on Azure.	The subscription includes the Red Hat product cost; you pay Microsoft for all other instance costs. Red Hat Gold Images are called "Cloud Access" images because you use your existing Red Hat subscriptions. Red Hat provides support directly for Cloud Access images.
Choose to deploy a custom image that you move to Azure.	Use your existing Red Hat subscriptions.	Enable subscriptions through the Red Hat Cloud Access program, upload your custom image, and attach your subscriptions.	The subscription includes the Red Hat product cost; you pay Microsoft for all other instance costs. Custom images that you move to Azure are "Cloud Access" images because you use your existing Red Hat subscriptions. Red Hat provides support directly for Cloud Access images.
Choose to deploy an existing Azure image that includes RHEL.	The Azure images include a Red Hat product.	Choose a RHEL image when you create a VM using the Azure console, or choose a VM from the Azure Marketplace.	You pay Microsoft hourly on a pay-as-you-go model. Such images are called "on-demand." Azure provides support for on-demand images through a support agreement. Red Hat provides updates to the images. Azure makes the updates available through the Red Hat Update Infrastructure (RHUI).

NOTE

You can create a custom image for Azure using Red Hat Image Builder. See Composing a Customized RHEL System Image for more information.

The following sections provide information and procedures related to Red Hat Enterprise Linux custom images.

Additional resources

- Using Red Hat Gold Images on Microsoft Azure
- Red Hat Cloud Access program
- Azure Marketplace
- Billing options in the Azure Marketplace
- Red Hat Enterprise Linux Bring-Your-Own-Subscription Gold Images in Azure

2.2. UNDERSTANDING BASE IMAGES

This section includes information on using preconfigured base images and their configuration settings.

2.2.1. Using a custom base image

To manually configure a virtual machine (VM), first create a base (starter) VM image. Then, you can modify configuration settings and add the packages the VM requires to operate on the cloud. You can make additional configuration changes for your specific application after you upload the image.

To prepare a cloud image of RHEL, follow the instructions in the sections below. To prepare a Hyper-V cloud image of RHEL, see the Prepare a Red Hat-based virtual machine from Hyper-V Manager .

2.2.2. Required system packages

To create and configure a base image of RHEL, your host system must have the following packages installed.

Table 2.2. System packages

Package	Repository	Description
libvirt	rhel-9-for-x86_64-appstream-rpms	Open source API, daemon, and management tool for managing platform virtualization
virt-install	rhel-9-for-x86_64-appstream-rpms	A command-line utility for building VMs
libguestfs	rhel-9-for-x86_64-appstream-rpms	A library for accessing and modifying VM file systems

Package	Repository	Description
guestfs-tools	rhel-9-for-x86_64-appstream-rpms	System administration tools for VMs; includes the **virt-customize** utility

2.2.3. Azure VM configuration settings

Azure VMs must have the following configuration settings. Some of these settings are enabled during the initial VM creation. Other settings are set when provisioning the VM image for Azure. Keep these settings in mind as you move through the procedures. Refer to them as necessary.

Table 2.3. VM configuration settings

Setting	Recommendation
ssh	ssh must be enabled to provide remote access to your Azure VMs.
dhcp	The primary virtual adapter should be configured for dhcp (IPv4 only).
Swap Space	Do not create a dedicated swap file or swap partition. You can configure swap space with the Windows Azure Linux Agent (WALinuxAgent).
NIC	Choose **virtio** for the primary virtual network adapter.
encryption	For custom images, use Network Bound Disk Encryption (NBDE) for full disk encryption on Azure.

2.2.4. Creating a base image from an ISO image

The following procedure lists the steps and initial configuration requirements for creating a custom ISO image. Once you have configured the image, you can use the image as a template for creating additional VM instances.

Prerequisites

- Ensure that you have enabled your host machine for virtualization. See Enabling virtualization in RHEL 9 for information and procedures.

Procedure

1. Download the latest Red Hat Enterprise Linux 9 DVD ISO image from the Red Hat Customer Portal.

2. Create and start a basic Red Hat Enterprise Linux VM. For instructions, see Creating virtual machines.

   a. If you use the command line to create your VM, ensure that you set the default memory and CPUs to the capacity you want for the VM. Set your virtual network interface to **virtio**. For example, the following command creates a **kvmtest** VM using the **rhel-9.0-x86_64-kvm.qcow2** image:

   ```
   # virt-install \
      --name kvmtest --memory 2048 --vcpus 2 \
      --disk rhel-9.0-x86_64-kvm.qcow2,bus=virtio \
      --import --os-variant=rhel9.0
   ```

   b. If you use the web console to create your VM, follow the procedure in Creating virtual machines using the web console, with these caveats:

      - Do not check **Immediately Start VM**.
      - Change your **Memory** size to your preferred settings.
      - Before you start the installation, ensure that you have changed **Model** under **Virtual Network Interface Settings** to **virtio** and change your **vCPUs** to the capacity settings you want for the VM.

3. Review the following additional installation selection and modifications.

   - Select **Minimal Install** with the **standard RHEL** option.
   - For **Installation Destination**, select **Custom Storage Configuration**. Use the following configuration information to make your selections.
     - Verify at least 500 MB for **/boot**.
     - For file system, use xfs, ext4, or ext3 for both **boot** and **root** partitions.
     - Remove swap space. Swap space is configured on the physical blade server in Azure by the WALinuxAgent.
   - On the **Installation Summary** screen, select **Network and Host Name**. Switch **Ethernet** to **On**.

4. When the install starts:

   - Create a **root** password.
   - Create an administrative user account.

5. When installation is complete, reboot the VM and log in to the root account.

6. Once you are logged in as **root**, you can configure the image.

2.3. CONFIGURING A CUSTOM BASE IMAGE FOR MICROSOFT AZURE

To deploy a RHEL 9 virtual machine (VM) with specific settings in Azure, you can create a custom base image for the VM. The following sections describe additional configuration changes that Azure requires.

2.3.1. Installing Hyper-V device drivers

Microsoft provides network and storage device drivers as part of their Linux Integration Services (LIS) for Hyper-V package. You may need to install Hyper-V device drivers on the VM image prior to provisioning it as an Azure virtual machine (VM). Use the **lsinitrd | grep hv** command to verify that the drivers are installed.

Procedure

1. Enter the following **grep** command to determine if the required Hyper-V device drivers are installed.

```
# lsinitrd | grep hv
```

In the example below, all required drivers are installed.

```
# lsinitrd | grep hv
drwxr-xr-x   2 root     root            0 Aug 12 14:21 usr/lib/modules/3.10.0-
932.el9.x86_64/kernel/drivers/hv
-rw-r--r--   1 root     root        31272 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el9.x86_64/kernel/drivers/hv/hv_vmbus.ko.xz
-rw-r--r--   1 root     root        25132 Aug 11 08:46 usr/lib/modules/3.10.0-
932.el9.x86_64/kernel/drivers/net/hyperv/hv_netvsc.ko.xz
-rw-r--r--   1 root     root         9796 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el9.x86_64/kernel/drivers/scsi/hv_storvsc.ko.xz
```

If all the drivers are not installed, complete the remaining steps.

NOTE

An **hv_vmbus** driver may exist in the environment. Even if this driver is present, complete the following steps.

2. Create a file named **hv.conf** in **/etc/dracut.conf.d**.

3. Add the following driver parameters to the **hv.conf** file.

```
add_drivers+=" hv_vmbus "
add_drivers+=" hv_netvsc "
add_drivers+=" hv_storvsc "
add_drivers+=" nvme "
```

NOTE

Note the spaces before and after the quotes, for example, **add_drivers+=" hv_vmbus "**. This ensures that unique drivers are loaded in the event that other Hyper-V drivers already exist in the environment.

4. Regenerate the **initramfs** image.

```
# dracut -f -v --regenerate-all
```

Verification

1. Reboot the machine.
2. Run the **lsinitrd | grep hv** command to verify that the drivers are installed.

2.3.2. Making configuration changes required for a Microsoft Azure deployment

Before you deploy your custom base image to Azure, you must perform additional configuration changes to ensure that the virtual machine (VM) can properly operate in Azure.

Procedure

1. Log in to the VM.
2. Register the VM, and enable the Red Hat Enterprise Linux 9 repository.

```
# subscription-manager register --auto-attach
Installed Product Current Status:
Product Name: Red Hat Enterprise Linux for x86_64
Status: Subscribed
```

3. Ensure that the **cloud-init** and **hyperv-daemons** packages are installed.

```
# dnf install cloud-init hyperv-daemons -y
```

4. Create **cloud-init** configuration files that are needed for integration with Azure services:
   a. To enable logging to the Hyper-V Data Exchange Service (KVP), create the **/etc/cloud/cloud.cfg.d/10-azure-kvp.cfg** configuration file and add the following lines to that file.

```
reporting:
    logging:
        type: log
    telemetry:
        type: hyperv
```

   b. To add Azure as a datasource, create the **/etc/cloud/cloud.cfg.d/91-azure_datasource.cfg** configuration file, and add the following lines to that file.

```
datasource_list: [ Azure ]
datasource:
    Azure:
        apply_network_config: False
```

5. To ensure that specific kernel modules are blocked from loading automatically, edit or create the **/etc/modprobe.d/blocklist.conf** file and add the following lines to that file.

```
blacklist nouveau
blacklist lbm-nouveau
blacklist floppy
```

```
blacklist amdgpu
blacklist skx_edac
blacklist intel_cstate
```

6. Modify **udev** network device rules:

   a. Remove the following persistent network device rules if present.

   ```
   # rm -f /etc/udev/rules.d/70-persistent-net.rules
   # rm -f /etc/udev/rules.d/75-persistent-net-generator.rules
   # rm -f /etc/udev/rules.d/80-net-name-slot-rules
   ```

   b. To ensure that Accelerated Networking on Azure works as intended, create a new network device rule **/etc/udev/rules.d/68-azure-sriov-nm-unmanaged.rules** and add the following line to it.

   ```
   SUBSYSTEM=="net", DRIVERS=="hv_pci", ACTION=="add",
   ENV{NM_UNMANAGED}="1"
   ```

7. Set the **sshd** service to start automatically.

   ```
   # systemctl enable sshd
   # systemctl is-enabled sshd
   ```

8. Modify kernel boot parameters:

   a. Open the **/etc/default/grub** file, and ensure the **GRUB_TIMEOUT** line has the following value.

   ```
   GRUB_TIMEOUT=10
   ```

   b. Remove the following options from the end of the **GRUB_CMDLINE_LINUX** line if present.

   ```
   rhgb quiet
   ```

   c. Ensure the **/etc/default/grub** file contains the following lines with all the specified options.

   ```
   GRUB_CMDLINE_LINUX="loglevel=3 crashkernel=auto console=tty1 console=ttyS0
   earlyprintk=ttyS0 rootdelay=300"
   GRUB_TIMEOUT_STYLE=countdown
   GRUB_TERMINAL="serial console"
   GRUB_SERIAL_COMMAND="serial --speed=115200 --unit=0 --word=8 --parity=no --
   stop=1"
   ```

   d. Regenerate the **grub.cfg** file.
   On a BIOS-based machine:

   ```
   # grub2-mkconfig -o /boot/grub2/grub.cfg
   ```

   On a UEFI-based machine:

   ```
   # grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
   ```

If your system uses a non-default location for **grub.cfg**, adjust the command accordingly.

9. Configure the Windows Azure Linux Agent (**WALinuxAgent**):

   a. Install and enable the **WALinuxAgent** package.

   ```
   # dnf install WALinuxAgent -y
   # systemctl enable waagent
   ```

   b. To ensure that a swap partition is not used in provisioned VMs, edit the following lines in the **/etc/waagent.conf** file.

   ```
   Provisioning.DeleteRootPassword=y
   ResourceDisk.Format=n
   ResourceDisk.EnableSwap=n
   ```

10. Prepare the VM for Azure provisioning:

    a. Unregister the VM from Red Hat Subscription Manager.

    ```
    # subscription-manager unregister
    ```

    b. Clean up the existing provisioning details.

    ```
    # waagent -force -deprovision
    ```

    NOTE

    This command generates warnings, which are expected because Azure handles the provisioning of VMs automatically.

    c. Clean the shell history and shut down the VM.

    ```
    # export HISTSIZE=0
    # poweroff
    ```

2.4. CONVERTING THE IMAGE TO A FIXED VHD FORMAT

All Microsoft Azure VM images must be in a fixed **VHD** format. The image must be aligned on a 1 MB boundary before it is converted to VHD. To convert the image from **qcow2** to a fixed **VHD** format and align the image, see the following procedure. Once you have converted the image, you can upload it to Azure.

Procedure

1. Convert the image from **qcow2** to **raw** format.

   ```
   $ qemu-img convert -f qcow2 -O raw <image-name>.qcow2 <image-name>.raw
   ```

2. Create a shell script using the contents below.

   ```
   #!/bin/bash
   ```

```
MB=$((1024 * 1024))
size=$(qemu-img info -f raw --output json "$1" | gawk 'match($0, /"virtual-size": ([0-9]+),/, val)
{print val[1]}')
rounded_size=$((($size/$MB + 1) * $MB))
if [ $(($size % $MB)) -eq  0 ]
then
 echo "Your image is already aligned. You do not need to resize."
 exit 1
fi
echo "rounded size = $rounded_size"
export rounded_size
```

3. Run the script. This example uses the name **align.sh**.

```
$ sh align.sh <image-xxx>.raw
```

- If the message *"Your image is already aligned. You do not need to resize."* displays, proceed to the following step.
- If a value displays, your image is not aligned.

4. Use the following command to convert the file to a fixed **VHD** format.
The sample uses qemu-img version 2.12.0.

```
$ qemu-img convert -f raw -o subformat=fixed,force_size -O vpc <image-xxx>.raw
<image.xxx>.vhd
```

Once converted, the **VHD** file is ready to upload to Azure.

5. If the **raw** image is not aligned, complete the following steps to align it.

a. Resize the **raw** file using the rounded value displayed when you ran the verification script.

```
$ qemu-img resize -f raw <image-xxx>.raw <rounded-value>
```

b. Convert the **raw** image file to a **VHD** format.
The sample uses qemu-img version 2.12.0.

```
$ qemu-img convert -f raw -o subformat=fixed,force_size -O vpc <image-xxx>.raw
<image.xxx>.vhd
```

Once converted, the **VHD** file is ready to upload to Azure.

2.5. INSTALLING THE AZURE CLI

Complete the following steps to install the Azure command line interface (Azure CLI 2.1). Azure CLI 2.1 is a Python-based utility that creates and manages VMs in Azure.

Prerequisites

- You need to have an account with Microsoft Azure before you can use the Azure CLI.
- The Azure CLI installation requires Python 3.x.

Procedure

1. Import the Microsoft repository key.

```
$ sudo rpm --import https://packages.microsoft.com/keys/microsoft.asc
```

2. Create a local Azure CLI repository entry.

```
$ sudo sh -c 'echo -e "[azure-cli]\nname=Azure
CLI\nbaseurl=https://packages.microsoft.com/yumrepos/azure-
cli\nenabled=1\ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" >
/etc/yum.repos.d/azure-cli.repo'
```

3. Update the **dnf** package index.

```
$ dnf check-update
```

4. Check your Python version (**python --version**) and install Python 3.x, if necessary.

```
$ sudo dnf install python3
```

5. Install the Azure CLI.

```
$ sudo dnf install -y azure-cli
```

6. Run the Azure CLI.

```
$ az
```

Additional resources

- Azure CLI
- Azure CLI command reference

2.6. CREATING RESOURCES IN AZURE

Complete the following procedure to create the Azure resources that you need before you can upload the **VHD** file and create the Azure image.

Procedure

1. Authenticate your system with Azure and log in.

```
$ az login
```

NOTE

If a browser is available in your environment, the CLI opens your browser to the Azure sign-in page. See Sign in with Azure CLI for more information and options.

2. Create a resource group in an Azure region.

```
$ az group create --name <resource-group> --location <azure-region>
```

Example:

```
[clouduser@localhost]$ az group create --name azrhelclirsgrp --location southcentralus
{
  "id": "/subscriptions//resourceGroups/azrhelclirsgrp",
  "location": "southcentralus",
  "managedBy": null,
  "name": "azrhelclirsgrp",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
```

3. Create a storage account. See SKU Types for more information about valid SKU values.

```
$ az storage account create -l <azure-region> -n <storage-account-name> -g
<resource-group> --sku <sku_type>
```

Example:

```
[clouduser@localhost]$ az storage account create -l southcentralus -n azrhelclistact -g
azrhelclirsgrp --sku Standard_LRS
{
  "accessTier": null,
  "creationTime": "2017-04-05T19:10:29.855470+00:00",
  "customDomain": null,
  "encryption": null,
  "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Storage/storageAccounts/azr
helclistact",
  "kind": "StorageV2",
  "lastGeoFailoverTime": null,
  "location": "southcentralus",
  "name": "azrhelclistact",
  "primaryEndpoints": {
    "blob": "https://azrhelclistact.blob.core.windows.net/",
    "file": "https://azrhelclistact.file.core.windows.net/",
    "queue": "https://azrhelclistact.queue.core.windows.net/",
    "table": "https://azrhelclistact.table.core.windows.net/"
},
"primaryLocation": "southcentralus",
"provisioningState": "Succeeded",
"resourceGroup": "azrhelclirsgrp",
"secondaryEndpoints": null,
"secondaryLocation": null,
"sku": {
  "name": "Standard_LRS",
  "tier": "Standard"
},
"statusOfPrimary": "available",
```

```
"statusOfSecondary": null,
"tags": {},
  "type": "Microsoft.Storage/storageAccounts"
}
```

4. Get the storage account connection string.

```
$ az storage account show-connection-string -n <storage-account-name> -g
<resource-group>
```

Example:

```
[clouduser@localhost]$ az storage account show-connection-string -n azrhelclistact -g
azrhelclirsgrp
{
  "connectionString":
"DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=azrhelclistact
AccountKey=NreGk...=="
}
```

5. Export the connection string by copying the connection string and pasting it into the following command. This string connects your system to the storage account.

```
$ export AZURE_STORAGE_CONNECTION_STRING="<storage-connection-string>"
```

Example:

```
[clouduser@localhost]$ export
AZURE_STORAGE_CONNECTION_STRING="DefaultEndpointsProtocol=https;Endpoi
ntSuffix=core.windows.net;AccountName=azrhelclistact;AccountKey=NreGk...=="
```

6. Create the storage container.

```
$ az storage container create -n <container-name>
```

Example:

```
[clouduser@localhost]$ az storage container create -n azrhelclistcont
{
  "created": true
}
```

7. Create a virtual network.

```
$ az network vnet create -g <resource group> --name <vnet-name> --subnet-name
<subnet-name>
```

Example:

```
[clouduser@localhost]$ az network vnet create --resource-group azrhelclirsgrp --name
azrhelclivnet1 --subnet-name azrhelclisubnet1
{
  "newVNet": {
```

```
    "addressSpace": {
      "addressPrefixes": [
      "10.0.0.0/16"
      ]
    },
    "dhcpOptions": {
      "dnsServers": []
    },
    "etag": "W/\"\"",
    "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Network/virtualNetworks/azr
helclivnet1",
    "location": "southcentralus",
    "name": "azrhelclivnet1",
    "provisioningState": "Succeeded",
    "resourceGroup": "azrhelclirsgrp",
    "resourceGuid": "0f25efee-e2a6-4abe-a4e9-817061ee1e79",
    "subnets": [
      {
        "addressPrefix": "10.0.0.0/24",
        "etag": "W/\"\"",
        "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Network/virtualNetworks/azr
helclivnet1/subnets/azrhelclisubnet1",
        "ipConfigurations": null,
        "name": "azrhelclisubnet1",
        "networkSecurityGroup": null,
        "provisioningState": "Succeeded",
        "resourceGroup": "azrhelclirsgrp",
        "resourceNavigationLinks": null,
        "routeTable": null
      }
    ],
    "tags": {},
    "type": "Microsoft.Network/virtualNetworks",
    "virtualNetworkPeerings": null
    }
}
```


Additional resources

- Azure Managed Disks Overview
- SKU Types

2.7. UPLOADING AND CREATING AN AZURE IMAGE

Complete the following steps to upload the **VHD** file to your container and create an Azure custom image.

NOTE

The exported storage connection string does not persist after a system reboot. If any of the commands in the following steps fail, export the connection string again.

Procedure

1. Upload the **VHD** file to the storage container. It may take several minutes. To get a list of storage containers, enter the **az storage container list** command.

```
$ az storage blob upload \
    --account-name <storage-account-name> --container-name <container-name> \
    --type page --file <path-to-vhd> --name <image-name>.vhd
```

Example:

```
[clouduser@localhost]$ az storage blob upload \
--account-name azrhelclistact --container-name azrhelclistcont \
--type page --file rhel-image-9.vhd --name rhel-image-9.vhd

Percent complete: %100.0
```

2. Get the URL for the uploaded **VHD** file to use in the following step.

```
$ az storage blob url -c <container-name> -n <image-name>.vhd
```

Example:

```
$ az storage blob url -c azrhelclistcont -n rhel-image-9.vhd
"https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.vhd"
```

3. Create the Azure custom image.

```
$ az image create -n <image-name> -g <resource-group> -l <azure-region> --source
<URL> --os-type linux
```

NOTE

The default hypervisor generation of the VM is V1. You can optionally specify a V2 hypervisor generation by including the option **--hyper-v-generation V2**. Generation 2 VMs use a UEFI-based boot architecture. See Support for generation 2 VMs on Azure for information on generation 2 VMs.

The command may return the error "Only blobs formatted as VHDs can be imported." This error may mean that the image was not aligned to the nearest 1 MB boundary before it was converted to **VHD**.

Example:

```
$ az image create -n rhel9 -g azrhelclirsgrp2 -l southcentralus --source
https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.vhd --os-type
linux
```

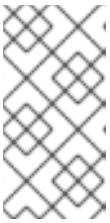
2.8. CREATING AND STARTING THE VM IN AZURE

The following steps provide the minimum command options to create a managed-disk Azure VM from the image. See az vm create for additional options.

Procedure

1. Enter the following command to create the VM.

```
$ az vm create \
   -g <resource-group> -l <azure-region> -n <vm-name> \
   --vnet-name <vnet-name> --subnet <subnet-name> --size Standard_A2 \
   --os-disk-name <simple-name> --admin-username <administrator-name> \
   --generate-ssh-keys --image <path-to-image>
```



NOTE

The option **--generate-ssh-keys** creates a private/public key pair. Private and public key files are created in ~/**.ssh** on your system. The public key is added to the **authorized_keys** file on the VM for the user specified by the **--admin-username** option. See Other authentication methods for additional information.

Example:

```
[clouduser@localhost]$ az vm create \
-g azrhelclirsgrp2 -l southcentralus -n rhel-azure-vm-1 \
--vnet-name azrhelclivnet1 --subnet azrhelclisubnet1 --size Standard_A2 \
--os-disk-name vm-1-osdisk --admin-username clouduser \
--generate-ssh-keys --image rhel9

{
  "fqdns": "",
  "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Compute/virtualMachines/rhe
l-azure-vm-1",
  "location": "southcentralus",
  "macAddress": "",
  "powerState": "VM running",
  "privateIpAddress": "10.0.0.4",
  "publicIpAddress": "<public-IP-address>",
  "resourceGroup": "azrhelclirsgrp2"
```

Note the **publicIpAddress**. You need this address to log in to the VM in the following step.

2. Start an SSH session and log in to the VM.

```
[clouduser@localhost]$ ssh -i /home/clouduser/.ssh/id_rsa clouduser@<public-IP-address>.
The authenticity of host ',<public-IP-address>' can't be established.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '<public-IP-address>' (ECDSA) to the list of known hosts.

[clouduser@rhel-azure-vm-1 ~]$
```

If you see a user prompt, you have successfully deployed your Azure VM.

You can now go to the Microsoft Azure portal and check the audit logs and properties of your resources. You can manage your VMs directly in this portal. If you are managing multiple VMs, you should use the Azure CLI. The Azure CLI provides a powerful interface to your resources in Azure. Enter **az --help** in

the CLI or see the Azure CLI command reference to learn more about the commands you use to manage your VMs in Microsoft Azure.

2.9. OTHER AUTHENTICATION METHODS

While recommended for increased security, using the Azure-generated key pair is not required. The following examples show two methods for SSH authentication.

Example 1: These command options provision a new VM without generating a public key file. They allow SSH authentication using a password.

```
$ az vm create \
   -g <resource-group> -l <azure-region> -n <vm-name> \
   --vnet-name <vnet-name> --subnet <subnet-name> --size Standard_A2 \
   --os-disk-name <simple-name> --authentication-type password \
   --admin-username <administrator-name> --admin-password <ssh-password> --image <path-to-image>
```

```
$ ssh <admin-username>@<public-ip-address>
```

Example 2: These command options provision a new Azure VM and allow SSH authentication using an existing public key file.

```
$ az vm create \
   -g <resource-group> -l <azure-region> -n <vm-name> \
   --vnet-name <vnet-name> --subnet <subnet-name> --size Standard_A2 \
   --os-disk-name <simple-name> --admin-username <administrator-name> \
   --ssh-key-value <path-to-existing-ssh-key> --image <path-to-image>
```

```
$ ssh -i <path-to-existing-ssh-key> <admin-username>@<public-ip-address>
```

2.10. ATTACHING RED HAT SUBSCRIPTIONS

Complete the following steps to attach the subscriptions you previously enabled through the Red Hat Cloud Access program.

Prerequisites

- You must have enabled your subscriptions.

Procedure

1. Register your system.

```
# subscription-manager register --auto-attach
```

2. Attach your subscriptions.
   - You can use an activation key to attach subscriptions. See Creating Red Hat Customer Portal Activation Keys for more information.

- Alternatively, you can manually attach a subscription using the ID of the subscription pool (Pool ID). See Attaching and Removing Subscriptions Through the Command Line .

Additional resources

- Creating Red Hat Customer Portal Activation Keys
- Attaching and Removing Subscriptions Through the Command Line
- Using and Configuring Red Hat Subscription Manager

2.11. SETTING UP AUTOMATIC REGISTRATION ON AZURE GOLD IMAGES

To make deploying RHEL 9 virtual machines (VM) on Micorsoft Azure faster and more comfortable, you can set up Gold Images of RHEL 9 to be automatically registered to the Red Hat Subscription Manager (RHSM).

Prerequisites

- You have enabled an eligible Red Hat product subscription for Cloud Access on Azure, and RHEL 9 Gold Images are therefore available to you in Microsoft Azure. For instructions, see Using Gold Images on Azure.

NOTE

A Microsoft Azure account can only be attached to a single Red Hat account at a time. Therefore, ensure no other users require access to the Azure account before attaching it to your Red Hat one.

Procedure

1. Use the Gold Image to create a RHEL 9 VM in your Azure instance. For instructions, see Creating and starting the VM in Azure .
2. Start the created VM.
3. In the RHEL 9 VM, enable automatic registration.

```
# subscription-manager config --rhsmcertd.auto_registration=1
```

4. Enable the **rhsmcertd** service.

```
# systemctl enable rhsmcertd.service
```

5. Disable the **redhat.repo** repository.

```
# subscription-manager config --rhsm.manage_repos=0
```

6. Power off the VM, and save it as a managed image on Azure. For instructions, see How to create a managed image of a virtual machine or VHD.
7. Create VMs using the managed image. They will be automatically subscribed to RHSM.

Verification

- In a RHEL 9 VM created using the above instructions, verify the system is registered to RHSM by executing the **subscription-manager identity** command. On a successfully registered system, this displays the UUID of the system. For example:

```
# subscription-manager identity
system identity: fdc46662-c536-43fb-a18a-bbcb283102b7
name: 192.168.122.222
org name: 6340056
org ID: 6340056
```

Additional resources

- Red Hat Gold Images in Azure
- Overview of RHEL images in Azure
- Configuring cloud sources for Red Hat services

2.12. ADDITIONAL RESOURCES

- Red Hat in the Public Cloud
- Red Hat Cloud Access Reference Guide
- Frequently Asked Questions and Recommended Practices for Microsoft Azure

CHAPTER 3. CONFIGURING A RED HAT HIGH AVAILABILITY CLUSTER ON MICROSOFT AZURE

To configure a Red Hat High Availability (HA) cluster on Azure using Azure virtual machine (VM) instances as cluster nodes, see the following sections. The procedures in these sections assume that you are creating a custom image for Azure. You have a number of options for obtaining the RHEL 9 images you use for your cluster. See Red Hat Enterprise Linux Image Options on Azure for information on image options for Azure.

The following sections provide:

- Prerequisite procedures for setting up your environment for Azure. After you set up your environment, you can create and configure Azure VM instances.
- Procedures specific to the creation of HA clusters, which transform individual nodes into a cluster of HA nodes on Azure. These include procedures for installing the High Availability packages and agents on each cluster node, configuring fencing, and installing Azure network resource agents.

Prerequisites

- Sign up for a Red Hat Customer Portal account .
- Sign up for a Microsoft Azure account with administrator privileges.
- You need to install Azure command line interface (CLI). For more information, see Installing the Azure CLI.
- Enable your subscriptions in the Red Hat Cloud Access program . The Red Hat Cloud Access program allows you to move your Red Hat subscriptions from physical or on-premise systems onto Azure with full support from Red Hat.
  - Alternatively, RHEL images can be obtained on-demand using Azure Marketplace.

3.1. CREATING RESOURCES IN AZURE

Complete the following procedure to create a region, resource group, storage account, virtual network, and availability set. You need these resources to set up a cluster on Microsoft Azure.

Procedure

1. Authenticate your system with Azure and log in.

```
$ az login
```

NOTE

If a browser is available in your environment, the CLI opens your browser to the Azure sign-in page.

Example:

```
[clouduser@localhost]$ az login
```

```
To sign in, use a web browser to open the page https://aka.ms/devicelogin and enter the code
FDMSCMETZ to authenticate.
 [
  {
   "cloudName": "AzureCloud",
   "id": "Subscription ID",
   "isDefault": true,
   "name": "MySubscriptionName",
   "state": "Enabled",
   "tenantId": "Tenant ID",
   "user": {
    "name": "clouduser@company.com",
    "type": "user"
   }
  }
 ]
```

2. Create a resource group in an Azure region.

```
$ az group create --name resource-group --location azure-region
```

Example:

```
[clouduser@localhost]$ az group create --name azrhelclirsgrp --location southcentralus

{
  "id": "/subscriptions//resourceGroups/azrhelclirsgrp",
  "location": "southcentralus",
  "managedBy": null,
  "name": "azrhelclirsgrp",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
```

3. Create a storage account.

```
$ az storage account create -l azure-region -n storage-account-name -g resource-
group --sku sku_type --kind StorageV2
```

Example:

```
[clouduser@localhost]$ az storage account create -l southcentralus -n azrhelclistact -g
azrhelclirsgrp --sku Standard_LRS --kind StorageV2

{
  "accessTier": null,
  "creationTime": "2017-04-05T19:10:29.855470+00:00",
  "customDomain": null,
  "encryption": null,
  "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Storage/storageAccounts/azr
```

```
helclistact",
  "kind": "StorageV2",
  "lastGeoFailoverTime": null,
  "location": "southcentralus",
  "name": "azrhelclistact",
  "primaryEndpoints": {
    "blob": "https://azrhelclistact.blob.core.windows.net/",
    "file": "https://azrhelclistact.file.core.windows.net/",
    "queue": "https://azrhelclistact.queue.core.windows.net/",
    "table": "https://azrhelclistact.table.core.windows.net/"
},
"primaryLocation": "southcentralus",
"provisioningState": "Succeeded",
"resourceGroup": "azrhelclirsgrp",
"secondaryEndpoints": null,
"secondaryLocation": null,
"sku": {
  "name": "Standard_LRS",
  "tier": "Standard"
},
"statusOfPrimary": "available",
"statusOfSecondary": null,
"tags": {},
  "type": "Microsoft.Storage/storageAccounts"
}
```

4. Get the storage account connection string.

```
$ az storage account show-connection-string -n storage-account-name -g resource-group
```

Example:

```
[clouduser@localhost]$ az storage account show-connection-string -n azrhelclistact -g azrhelclirsgrp
{
  "connectionString":
"DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=azrhelclistact
AccountKey=NreGk...=="
}
```

5. Export the connection string by copying the connection string and pasting it into the following command. This string connects your system to the storage account.

```
$ export AZURE_STORAGE_CONNECTION_STRING="storage-connection-string"
```

Example:

```
[clouduser@localhost]$ export
AZURE_STORAGE_CONNECTION_STRING="DefaultEndpointsProtocol=https;Endpoi
ntSuffix=core.windows.net;AccountName=azrhelclistact;AccountKey=NreGk...=="
```

6. Create the storage container.

```
$ az storage container create -n container-name
```

Example:

```
[clouduser@localhost]$ az storage container create -n azrhelclistcont

{
  "created": true
}
```

7. Create a virtual network. All cluster nodes must be in the same virtual network.

```
$ az network vnet create -g resource group --name vnet-name --subnet-name subnet-name
```

Example:

```
[clouduser@localhost]$ az network vnet create --resource-group azrhelclirsgrp --name azrhelclivnet1 --subnet-name azrhelclisubnet1
{
  "newVNet": {
    "addressSpace": {
      "addressPrefixes": [
      "10.0.0.0/16"
      ]
  },
  "dhcpOptions": {
    "dnsServers": []
  },
  "etag": "W/\"\"",
  "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Network/virtualNetworks/azrhelclivnet1",
  "location": "southcentralus",
  "name": "azrhelclivnet1",
  "provisioningState": "Succeeded",
  "resourceGroup": "azrhelclirsgrp",
  "resourceGuid": "0f25efee-e2a6-4abe-a4e9-817061ee1e79",
  "subnets": [
    {
      "addressPrefix": "10.0.0.0/24",
      "etag": "W/\"\"",
      "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Network/virtualNetworks/azrhelclivnet1/subnets/azrhelclisubnet1",
      "ipConfigurations": null,
      "name": "azrhelclisubnet1",
      "networkSecurityGroup": null,
      "provisioningState": "Succeeded",
      "resourceGroup": "azrhelclirsgrp",
      "resourceNavigationLinks": null,
      "routeTable": null
    }
  ],
  "tags": {},
```

```
  "type": "Microsoft.Network/virtualNetworks",
  "virtualNetworkPeerings": null
  }
}
```

8. Create an availability set. All cluster nodes must be in the same availability set.

```
$ az vm availability-set create --name MyAvailabilitySet --resource-group
MyResourceGroup
```

Example:

```
[clouduser@localhost]$ az vm availability-set create --name rhelha-avset1 --resource-
group azrhelclirsgrp
{
  "additionalProperties": {},
    "id":
"/subscriptions/.../resourceGroups/azrhelclirsgrp/providers/Microsoft.Compute/availabilitySets/rh
elha-avset1",
    "location": "southcentralus",
    "name": “rhelha-avset1",
    "platformFaultDomainCount": 2,
    "platformUpdateDomainCount": 5,

[omitted]
```

Additional resources

- Sign in with Azure CLI
- SKU Types
- Azure Managed Disks Overview

3.2. REQUIRED SYSTEM PACKAGES FOR HIGH AVAILABILITY

The procedure assumes you are creating a VM image for Azure HA using Red Hat Enterprise Linux. To successfully complete the procedure, the following packages must be installed.

Table 3.1. System packages

Package	Repository	Description
libvirt	rhel-9-for-x86_64-appstream-rpms	Open source API, daemon, and management tool for managing platform virtualization
virt-install	rhel-9-for-x86_64-appstream-rpms	A command-line utility for building VMs
libguestfs	rhel-9-for-x86_64-appstream-rpms	A library for accessing and modifying VM file systems

Package	Repository	Description
guestfs-tools	rhel-9-for-x86_64-appstream-rpms	System administration tools for VMs; includes the **virt-customize** utility

3.3. AZURE VM CONFIGURATION SETTINGS

Azure VMs must have the following configuration settings. Some of these settings are enabled during the initial VM creation. Other settings are set when provisioning the VM image for Azure. Keep these settings in mind as you move through the procedures. Refer to them as necessary.

Table 3.2. VM configuration settings

Setting	Recommendation
ssh	ssh must be enabled to provide remote access to your Azure VMs.
dhcp	The primary virtual adapter should be configured for dhcp (IPv4 only).
Swap Space	Do not create a dedicated swap file or swap partition. You can configure swap space with the Windows Azure Linux Agent (WALinuxAgent).
NIC	Choose **virtio** for the primary virtual network adapter.
encryption	For custom images, use Network Bound Disk Encryption (NBDE) for full disk encryption on Azure.

3.4. INSTALLING HYPER-V DEVICE DRIVERS

Microsoft provides network and storage device drivers as part of their Linux Integration Services (LIS) for Hyper-V package. You may need to install Hyper-V device drivers on the VM image prior to provisioning it as an Azure virtual machine (VM). Use the **lsinitrd | grep hv** command to verify that the drivers are installed.

Procedure

1. Enter the following **grep** command to determine if the required Hyper-V device drivers are installed.

```
# lsinitrd | grep hv
```

In the example below, all required drivers are installed.

```
# lsinitrd | grep hv
drwxr-xr-x   2 root     root            0 Aug 12 14:21 usr/lib/modules/3.10.0-
```

```
932.el9.x86_64/kernel/drivers/hv
-rw-r--r--   1 root     root        31272 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el9.x86_64/kernel/drivers/hv/hv_vmbus.ko.xz
-rw-r--r--   1 root     root        25132 Aug 11 08:46 usr/lib/modules/3.10.0-
932.el9.x86_64/kernel/drivers/net/hyperv/hv_netvsc.ko.xz
-rw-r--r--   1 root     root         9796 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el9.x86_64/kernel/drivers/scsi/hv_storvsc.ko.xz
```

If all the drivers are not installed, complete the remaining steps.

NOTE

An **hv_vmbus** driver may exist in the environment. Even if this driver is present, complete the following steps.

2. Create a file named **hv.conf** in **/etc/dracut.conf.d**.
3. Add the following driver parameters to the **hv.conf** file.

```
add_drivers+=" hv_vmbus "
add_drivers+=" hv_netvsc "
add_drivers+=" hv_storvsc "
add_drivers+=" nvme "
```

NOTE

Note the spaces before and after the quotes, for example, **add_drivers+=" hv_vmbus "**. This ensures that unique drivers are loaded in the event that other Hyper-V drivers already exist in the environment.

4. Regenerate the **initramfs** image.

```
# dracut -f -v --regenerate-all
```

Verification

1. Reboot the machine.
2. Run the **lsinitrd | grep hv** command to verify that the drivers are installed.

3.5. MAKING CONFIGURATION CHANGES REQUIRED FOR A MICROSOFT AZURE DEPLOYMENT

Before you deploy your custom base image to Azure, you must perform additional configuration changes to ensure that the virtual machine (VM) can properly operate in Azure.

Procedure

1. Log in to the VM.
2. Register the VM, and enable the Red Hat Enterprise Linux 9 repository.

```
# subscription-manager register --auto-attach
Installed Product Current Status:
Product Name: Red Hat Enterprise Linux for x86_64
Status: Subscribed
```

3. Ensure that the **cloud-init** and **hyperv-daemons** packages are installed.

```
# dnf install cloud-init hyperv-daemons -y
```

4. Create **cloud-init** configuration files that are needed for integration with Azure services:

   a. To enable logging to the Hyper-V Data Exchange Service (KVP), create the **/etc/cloud/cloud.cfg.d/10-azure-kvp.cfg** configuration file and add the following lines to that file.

```
reporting:
    logging:
        type: log
    telemetry:
        type: hyperv
```

   b. To add Azure as a datasource, create the **/etc/cloud/cloud.cfg.d/91-azure_datasource.cfg** configuration file, and add the following lines to that file.

```
datasource_list: [ Azure ]
datasource:
    Azure:
        apply_network_config: False
```

5. To ensure that specific kernel modules are blocked from loading automatically, edit or create the **/etc/modprobe.d/blocklist.conf** file and add the following lines to that file.

```
blacklist nouveau
blacklist lbm-nouveau
blacklist floppy
blacklist amdgpu
blacklist skx_edac
blacklist intel_cstate
```

6. Modify **udev** network device rules:

   a. Remove the following persistent network device rules if present.

```
# rm -f /etc/udev/rules.d/70-persistent-net.rules
# rm -f /etc/udev/rules.d/75-persistent-net-generator.rules
# rm -f /etc/udev/rules.d/80-net-name-slot-rules
```

   b. To ensure that Accelerated Networking on Azure works as intended, create a new network device rule **/etc/udev/rules.d/68-azure-sriov-nm-unmanaged.rules** and add the following line to it.

```
SUBSYSTEM=="net", DRIVERS=="hv_pci", ACTION=="add",
ENV{NM_UNMANAGED}="1"
```

7. Set the **sshd** service to start automatically.

```
# systemctl enable sshd
# systemctl is-enabled sshd
```

8. Modify kernel boot parameters:

   a. Open the **/etc/default/grub** file, and ensure the **GRUB_TIMEOUT** line has the following value.

```
GRUB_TIMEOUT=10
```

   b. Remove the following options from the end of the **GRUB_CMDLINE_LINUX** line if present.

```
rhgb quiet
```

   c. Ensure the **/etc/default/grub** file contains the following lines with all the specified options.

```
GRUB_CMDLINE_LINUX="loglevel=3 crashkernel=auto console=tty1 console=ttyS0
earlyprintk=ttyS0 rootdelay=300"
GRUB_TIMEOUT_STYLE=countdown
GRUB_TERMINAL="serial console"
GRUB_SERIAL_COMMAND="serial --speed=115200 --unit=0 --word=8 --parity=no --
stop=1"
```

   d. Regenerate the **grub.cfg** file.
   On a BIOS-based machine:

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

   On a UEFI-based machine:

```
# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```

   If your system uses a non-default location for **grub.cfg**, adjust the command accordingly.

9. Configure the Windows Azure Linux Agent (**WALinuxAgent**):

   a. Install and enable the **WALinuxAgent** package.

```
# dnf install WALinuxAgent -y
# systemctl enable waagent
```

   b. To ensure that a swap partition is not used in provisioned VMs, edit the following lines in the **/etc/waagent.conf** file.

```
Provisioning.DeleteRootPassword=y
ResourceDisk.Format=n
ResourceDisk.EnableSwap=n
```

10. Prepare the VM for Azure provisioning:

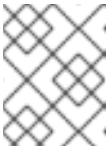
    a. Unregister the VM from Red Hat Subscription Manager.

    –

```
# subscription-manager unregister
```

b. Clean up the existing provisioning details.

```
# waagent -force -deprovision
```



NOTE

This command generates warnings, which are expected because Azure handles the provisioning of VMs automatically.

c. Clean the shell history and shut down the VM.

```
# export HISTSIZE=0
# poweroff
```

3.6. CREATING AN AZURE ACTIVE DIRECTORY APPLICATION

Complete the following procedure to create an Azure Active Directory AD Application. The Azure AD Application authorizes and automates access for HA operations for all nodes in the cluster.

Prerequisites

Install the Azure Command Line Interface (CLI).

Procedure

1. Ensure you are an Administrator or Owner for the Microsoft Azure subscription. You need this authorization to create an Azure AD application.
2. Log in to your Azure account.

```
$ az login
```

3. Enter the following command to create the Azure AD Application. To use your own password, add the **--password** option to the command. Ensure that you create a strong password.

```
$ az ad sp create-for-rbac --name FencingApplicationName --role owner --scopes
"/subscriptions/SubscriptionID/resourceGroups/MyResourseGroup"
```

Example:

```
[clouduser@localhost ~] $ az ad sp create-for-rbac --name FencingApp --role owner --
scopes "/subscriptions/2586c64b-xxxxxx-xxxxxxx-
xxxxxxx/resourceGroups/azrhelclirsgrp"
Retrying role assignment creation: 1/36
Retrying role assignment creation: 2/36
Retrying role assignment creation: 3/36
{
  "appId": "1a3dfe06-df55-42ad-937b-326d1c211739",
  "displayName": "FencingApp",
  "name": "http://FencingApp",
```

```
  "password": "43a603f0-64bb-482e-800d-402efe5f3d47",
  "tenant": "77ecefb6-xxxxxxxxxx-xxxxxxx-757a69cb9485"
}
```

4. Save the following information before proceeding. You need this information to set up the fencing agent.
   - Azure AD Application ID
   - Azure AD Application Password
   - Tenant ID
   - Microsoft Azure Subscription ID

Additional resources

- View the access a user has to Azure resources

3.7. CONVERTING THE IMAGE TO A FIXED VHD FORMAT

All Microsoft Azure VM images must be in a fixed **VHD** format. The image must be aligned on a 1 MB boundary before it is converted to VHD. To convert the image from **qcow2** to a fixed **VHD** format and align the image, see the following procedure. Once you have converted the image, you can upload it to Azure.

Procedure

1. Convert the image from **qcow2** to **raw** format.

```
$ qemu-img convert -f qcow2 -O raw <image-name>.qcow2 <image-name>.raw
```

2. Create a shell script using the contents below.

```
#!/bin/bash
MB=$((1024 * 1024))
size=$(qemu-img info -f raw --output json "$1" | gawk 'match($0, /"virtual-size": ([0-9]+),/, val)
{print val[1]}')
rounded_size=$((($size/$MB + 1) * $MB))
if [ $(($size % $MB)) -eq  0 ]
then
 echo "Your image is already aligned. You do not need to resize."
 exit 1
fi
echo "rounded size = $rounded_size"
export rounded_size
```

3. Run the script. This example uses the name **align.sh**.

```
$ sh align.sh <image-xxx>.raw
```

   - If the message *"Your image is already aligned. You do not need to resize."* displays, proceed to the following step.

- If a value displays, your image is not aligned.

4. Use the following command to convert the file to a fixed **VHD** format.
   The sample uses qemu-img version 2.12.0.

```
$ qemu-img convert -f raw -o subformat=fixed,force_size -O vpc <image-xxx>.raw
<image.xxx>.vhd
```

   Once converted, the **VHD** file is ready to upload to Azure.

5. If the **raw** image is not aligned, complete the following steps to align it.

   a. Resize the **raw** file using the rounded value displayed when you ran the verification script.

```
$ qemu-img resize -f raw <image-xxx>.raw <rounded-value>
```

   b. Convert the **raw** image file to a **VHD** format.
   The sample uses qemu-img version 2.12.0.

```
$ qemu-img convert -f raw -o subformat=fixed,force_size -O vpc <image-xxx>.raw
<image.xxx>.vhd
```

   Once converted, the **VHD** file is ready to upload to Azure.

3.8. UPLOADING AND CREATING AN AZURE IMAGE

Complete the following steps to upload the **VHD** file to your container and create an Azure custom image.

NOTE

The exported storage connection string does not persist after a system reboot. If any of the commands in the following steps fail, export the connection string again.

Procedure

1. Upload the **VHD** file to the storage container. It may take several minutes. To get a list of storage containers, enter the **az storage container list** command.

```
$ az storage blob upload \
    --account-name <storage-account-name> --container-name <container-name> \
    --type page --file <path-to-vhd> --name <image-name>.vhd
```

   Example:

```
[clouduser@localhost]$ az storage blob upload \
--account-name azrhelclistact --container-name azrhelclistcont \
--type page --file rhel-image-9.vhd --name rhel-image-9.vhd

Percent complete: %100.0
```

2. Get the URL for the uploaded **VHD** file to use in the following step.

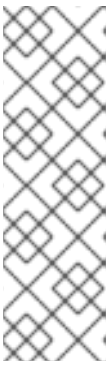
```
$ az storage blob url -c <container-name> -n <image-name>.vhd
```

Example:

```
$ az storage blob url -c azrhelclistcont -n rhel-image-9.vhd
"https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.vhd"
```

3. Create the Azure custom image.

```
$ az image create -n <image-name> -g <resource-group> -l <azure-region> --source
<URL> --os-type linux
```



NOTE

The default hypervisor generation of the VM is V1. You can optionally specify a V2 hypervisor generation by including the option **--hyper-v-generation V2**. Generation 2 VMs use a UEFI-based boot architecture. See Support for generation 2 VMs on Azure for information on generation 2 VMs.

The command may return the error "Only blobs formatted as VHDs can be imported." This error may mean that the image was not aligned to the nearest 1 MB boundary before it was converted to **VHD**.

Example:

```
$ az image create -n rhel9 -g azrhelclirsgrp2 -l southcentralus --source
https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.vhd --os-type
linux
```

3.9. INSTALLING RED HAT HA PACKAGES AND AGENTS

Complete the following steps on all nodes.

Procedure

1. Launch an SSH terminal session and connect to the VM using the administrator name and public IP address.

```
$ ssh administrator@PublicIP
```

To get the public IP address for an Azure VM, open the VM properties in the Azure Portal or enter the following Azure CLI command.

```
$ az vm list -g <resource-group> -d --output table
```

Example:

```
[clouduser@localhost ~] $ az vm list -g azrhelclirsgrp -d --output table
Name    ResourceGroup          PowerState      PublicIps       Location
------  ---------------------  --------------  -------------   --------------
node01  azrhelclirsgrp         VM running      192.98.152.251  southcentralus
```

2. Register the VM with Red Hat.

```
$ sudo -i
# subscription-manager register --auto-attach
```

NOTE

If the **--auto-attach** command fails, manually register the VM to your subscription.

3. Disable all repositories.

```
# subscription-manager repos --disable=*
```

4. Enable the RHEL 9 Server HA repositories.

```
# subscription-manager repos --enable=rhel-9-for-x86_64-highavailability-rpms
```

5. Update all packages.

```
# dnf update -y
```

6. Install the Red Hat High Availability Add-On software packages, along with all available fencing agents from the High Availability channel.

```
# dnf install pcs pacemaker fence-agents-azure-arm
```

7. The user **hacluster** was created during the pcs and pacemaker installation in the previous step. Create a password for **hacluster** on all cluster nodes. Use the same password for all nodes.

```
# passwd hacluster
```

8. Add the **high availability** service to the RHEL Firewall if **firewalld.service** is installed.

```
# firewall-cmd --permanent --add-service=high-availability
# firewall-cmd --reload
```

9. Start the **pcs** service and enable it to start on boot.

```
# systemctl start pcsd.service
# systemctl enable pcsd.service

Created symlink from /etc/systemd/system/multi-user.target.wants/pcsd.service to
/usr/lib/systemd/system/pcsd.service.
```

Verification

- Ensure the **pcs** service is running.

```
# systemctl status pcsd.service
pcsd.service - PCS GUI and remote configuration interface
```

```
Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset: disabled)
Active: active (running) since Fri 2018-02-23 11:00:58 EST; 1min 23s ago
Docs: man:pcsd(8)
        man:pcs(8)
Main PID: 46235 (pcsd)
  CGroup: /system.slice/pcsd.service
          └─46235 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &
```

3.10. CREATING A CLUSTER

Complete the following steps to create the cluster of nodes.

Procedure

1. On one of the nodes, enter the following command to authenticate the pcs user **hacluster**. In the command, specify the name of each node in the cluster.

   ```
   # pcs host auth <hostname1> <hostname2> <hostname3>
   ```

   Example:

   ```
   [root@node01 clouduser]# pcs host auth node01 node02 node03
   Username: hacluster
   Password:
   node01: Authorized
   node02: Authorized
   node03: Authorized
   ```

2. Create the cluster.

   ```
   # pcs cluster setup <cluster_name> <hostname1> <hostname2> <hostname3>
   ```

   Example:

   ```
   [root@node01 clouduser]# pcs cluster setup new_cluster node01 node02 node03

   [...]

   Synchronizing pcsd certificates on nodes node01, node02, node03...
   node02: Success
   node03: Success
   node01: Success
   Restarting pcsd on the nodes in order to reload the certificates...
   node02: Success
   node03: Success
   node01: Success
   ```

Verification

1. Enable the cluster.

   ```
   [root@node01 clouduser]# pcs cluster enable --all
   node02: Cluster Enabled
   ```

```
node03: Cluster Enabled
node01: Cluster Enabled
```

2. Start the cluster.

```
[root@node01 clouduser]# pcs cluster start --all
node02: Starting Cluster...
node03: Starting Cluster...
node01: Starting Cluster...
```

3.11. FENCING OVERVIEW

If communication with a single node in the cluster fails, then other nodes in the cluster must be able to restrict or release access to resources that the failed cluster node may have access to. This cannot be accomplished by contacting the cluster node itself as the cluster node may not be responsive. Instead, you must provide an external method, which is called fencing with a fence agent.

A node that is unresponsive may still be accessing data. The only way to be certain that your data is safe is to fence the node using STONITH. STONITH is an acronym for "Shoot The Other Node In The Head," and it protects your data from being corrupted by rogue nodes or concurrent access. Using STONITH, you can be certain that a node is truly offline before allowing the data to be accessed from another node.

Additional resources

- Fencing in Red Hat High Availability Cluster

3.12. CREATING A FENCING DEVICE

Complete the following steps to configure fencing. Complete these commands from any node in the cluster

Prerequisites

You need to set the cluster property **stonith-enabled** to **true**.

Procedure

1. Identify the Azure node name for each RHEL VM. You use the Azure node names to configure the fence device.

```
# fence_azure_arm \
   -l <AD-Application-ID> -p <AD-Password> \
   --resourceGroup <MyResourceGroup> --tenantId <Tenant-ID> \
   --subscriptionId <Subscription-ID> -o list
```

Example:

```
[root@node01 clouduser]# fence_azure_arm \
-l e04a6a49-9f00-xxxx-xxxx-a8bdda4af447 -p
z/a05AwCN0IzAjVwXXXXXXXEWIoeVp0xg7QT//JE=
--resourceGroup azrhelclirsgrp --tenantId 77ecefb6-cff0-XXXX-XXXX-757XXXX9485
--subscriptionId XXXXXXXX-38b4-4527-XXXX-012d49dfc02c -o list
```

```
node01,
node02,
node03,
```

2. View the options for the Azure ARM STONITH agent.

```
# pcs stonith describe fence_azure_arm
```

Example:

```
# pcs stonith describe fence_apc
Stonith options:
password: Authentication key
password_script: Script to run to retrieve password
```

WARNING

For fence agents that provide a method option, do not specify a value of cycle as it is not supported and can cause data corruption.

Some fence devices can fence only a single node, while other devices can fence multiple nodes. The parameters you specify when you create a fencing device depend on what your fencing device supports and requires.

You can use the **pcmk_host_list** parameter when creating a fencing device to specify all of the machines that are controlled by that fencing device.

You can use **pcmk_host_map** parameter when creating a fencing device to map host names to the specifications that comprehends the fence device.

3. Create a fence device.

```
# pcs stonith create clusterfence fence_azure_arm
```

Verification

1. Test the fencing agent for one of the other nodes.

```
# pcs stonith fence azurenodename
```

Example:

```
[root@node01 clouduser]# pcs status
Cluster name: newcluster
Stack: corosync
Current DC: node01 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Fri Feb 23 11:44:35 2018
Last change: Fri Feb 23 11:21:01 2018 by root via cibadmin on node01
```

```
3 nodes configured
1 resource configured

Online: [ node01 node03 ]
OFFLINE: [ node02 ]

Full list of resources:

 clusterfence  (stonith:fence_azure_arm):  Started node01

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

2. Start the node that was fenced in the previous step.

```
# pcs cluster start <hostname>
```

3. Check the status to verify the node started.

```
# pcs status
```

Example:

```
[root@node01 clouduser]# pcs status
Cluster name: newcluster
Stack: corosync
Current DC: node01 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Fri Feb 23 11:34:59 2018
Last change: Fri Feb 23 11:21:01 2018 by root via cibadmin on node01

3 nodes configured
1 resource configured

Online: [ node01 node02 node03 ]

Full list of resources:

clusterfence    (stonith:fence_azure_arm):  Started node01

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

Additional resources

- Fencing in a Red Hat High Availability Cluster
- General properties of fencing devices

3.13. CREATING AN AZURE INTERNAL LOAD BALANCER

The Azure internal load balancer removes cluster nodes that do not answer health probe requests.

Perform the following procedure to create an Azure internal load balancer. Each step references a specific Microsoft procedure and includes the settings for customizing the load balancer for HA.

Prerequisites

Azure control panel

Procedure

1. Create a Basic load balancer . Select **Internal load balancer**, the **Basic SKU**, and **Dynamic** for the type of IP address assignment.
2. Create a back-end address pool . Associate the backend pool to the availability set created while creating Azure resources in HA. Do not set any target network IP configurations.
3. Create a health probe . For the health probe, select **TCP** and enter port **61000**. You can use TCP port number that does not interfere with another service. For certain HA product applications (for example, SAP HANA and SQL Server), you may need to work with Microsoft to identify the correct port to use.
4. Create a load balancer rule . To create the load balancing rule, the default values are prepopulated. Ensure to set **Floating IP (direct server return)** to **Enabled**.

3.14. CONFIGURING THE LOAD BALANCER RESOURCE AGENT

After you have created the health probe, you must configure the **load balancer** resource agent. This resource agent runs a service that answers health probe requests from the Azure load balancer and removes cluster nodes that do not answer requests.

Procedure

1. Install the **nmap-ncat** resource agents on all nodes.

   ```
   # dnf install nmap-ncat resource-agents
   ```

   Perform the following steps on a single node.

2. Create the **pcs** resources and group. Use your load balancer FrontendIP for the IPaddr2 address.

   ```
   # pcs resource create resource-name IPaddr2 ip="10.0.0.7" --group cluster-resources-group
   ```

3. Configure the **load balancer** resource agent.

   ```
   # pcs resource create resource-loadbalancer-name azure-lb port=port-number --group cluster-resources-group
   ```

Verification

- Run **pcs status** to see the results.

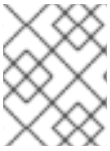
```
[root@node01 clouduser]# pcs status
```

Example output:

```
Cluster name: clusterfence01
Stack: corosync
Current DC: node02 (version 1.1.16-12.el7_4.7-94ff4df) - partition with quorum
Last updated: Tue Jan 30 12:42:35 2018
Last change: Tue Jan 30 12:26:42 2018 by root via cibadmin on node01

3 nodes configured
3 resources configured

Online: [ node01 node02 node03 ]

Full list of resources:

clusterfence (stonith:fence_azure_arm):      Started node01
Resource Group: g_azure
    vip_azure  (ocf::heartbeat:IPaddr2):      Started node02
    lb_azure   (ocf::heartbeat:azure-lb):     Started node02

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

3.15. CONFIGURING SHARED BLOCK STORAGE

To configure shared block storage for a Red Hat High Availability cluster with Microsoft Azure Shared Disks, use the following procedure. Note that this procedure is optional, and the steps below assume three Azure VMs (a three-node cluster) with a 1 TB shared disk.



NOTE

This is a stand-alone sample procedure for configuring block storage. The procedure assumes that you have not yet created your cluster.

Prerequisites

- You must have installed the Azure CLI on your host system and created your SSH key(s).
- You must have created your cluster environment in Azure, which includes creating the following resources. Links are to the Microsoft Azure documentation.
  - Resource group
  - Virtual network
  - Network security group(s)
  - Network security group rules
  - Subnet(s)

- Load balancer (optional)
- Storage account
- Proximity placement group
- Availability set

Procedure

1. Create a shared block volume using the Azure command **az disk create**.

```
$ az disk create -g <resource_group> -n <shared_block_volume_name> --size-gb
<disk_size> --max-shares <number_vms> -l <location>
```

For example, the following command creates a shared block volume named **shared-block-volume.vhd** in the resource group **sharedblock** within the Azure Availability Zone **westcentralus**.

```
$ az disk create -g sharedblock-rg -n shared-block-volume.vhd --size-gb 1024 --max-
shares 3 -l westcentralus

{
  "creationData": {
    "createOption": "Empty",
    "galleryImageReference": null,
    "imageReference": null,
    "sourceResourceId": null,
    "sourceUniqueId": null,
    "sourceUri": null,
    "storageAccountId": null,
    "uploadSizeBytes": null
  },
  "diskAccessId": null,
  "diskIopsReadOnly": null,
  "diskIopsReadWrite": 5000,
  "diskMbpsReadOnly": null,
  "diskMbpsReadWrite": 200,
  "diskSizeBytes": 1099511627776,
  "diskSizeGb": 1024,
  "diskState": "Unattached",
  "encryption": {
    "diskEncryptionSetId": null,
    "type": "EncryptionAtRestWithPlatformKey"
  },
  "encryptionSettingsCollection": null,
  "hyperVgeneration": "V1",
  "id": "/subscriptions/12345678910-12345678910/resourceGroups/sharedblock-
rg/providers/Microsoft.Compute/disks/shared-block-volume.vhd",
  "location": "westcentralus",
  "managedBy": null,
  "managedByExtended": null,
  "maxShares": 3,
  "name": "shared-block-volume.vhd",
  "networkAccessPolicy": "AllowAll",
  "osType": null,
```

```
  "provisioningState": "Succeeded",
  "resourceGroup": "sharedblock-rg",
  "shareInfo": null,
  "sku": {
    "name": "Premium_LRS",
    "tier": "Premium"
  },
  "tags": {},
  "timeCreated": "2020-08-27T15:36:56.263382+00:00",
  "type": "Microsoft.Compute/disks",
  "uniqueId": "cd8b0a25-6fbe-4779-9312-8d9cbb89b6f2",
  "zones": null
}
```

2. Verify that you have created the shared block volume using the Azure command **az disk show**.

```
$ az disk show -g <resource_group> -n <shared_block_volume_name>
```

For example, the following command shows details for the shared block volume **shared-block-volume.vhd** within the resource group **sharedblock-rg**.

```
$ az disk show -g sharedblock-rg -n shared-block-volume.vhd

{
  "creationData": {
    "createOption": "Empty",
    "galleryImageReference": null,
    "imageReference": null,
    "sourceResourceId": null,
    "sourceUniqueId": null,
    "sourceUri": null,
    "storageAccountId": null,
    "uploadSizeBytes": null
  },
  "diskAccessId": null,
  "diskIopsReadOnly": null,
  "diskIopsReadWrite": 5000,
  "diskMbpsReadOnly": null,
  "diskMbpsReadWrite": 200,
  "diskSizeBytes": 1099511627776,
  "diskSizeGb": 1024,
  "diskState": "Unattached",
  "encryption": {
    "diskEncryptionSetId": null,
    "type": "EncryptionAtRestWithPlatformKey"
  },
  "encryptionSettingsCollection": null,
  "hyperVgeneration": "V1",
  "id": "/subscriptions/12345678910-12345678910/resourceGroups/sharedblock-
rg/providers/Microsoft.Compute/disks/shared-block-volume.vhd",
  "location": "westcentralus",
  "managedBy": null,
  "managedByExtended": null,
  "maxShares": 3,
  "name": "shared-block-volume.vhd",
```

```
  "networkAccessPolicy": "AllowAll",
  "osType": null,
  "provisioningState": "Succeeded",
  "resourceGroup": "sharedblock-rg",
  "shareInfo": null,
  "sku": {
    "name": "Premium_LRS",
    "tier": "Premium"
  },
  "tags": {},
  "timeCreated": "2020-08-27T15:36:56.263382+00:00",
  "type": "Microsoft.Compute/disks",
  "uniqueId": "cd8b0a25-6fbe-4779-9312-8d9cbb89b6f2",
  "zones": null
}
```

3. Create three network interfaces using the Azure command **az network nic create**. Run the following command three times using a different **<nic_name>** for each.

```
$ az network nic create \
    -g <resource_group> -n <nic_name> --subnet <subnet_name> \
    --vnet-name <virtual_network> --location <location> \
    --network-security-group <network_security_group> --private-ip-address-version
IPv4
```

For example, the following command creates a network interface with the name **shareblock-nodea-vm-nic-protected**.

```
$ az network nic create \
    -g sharedblock-rg -n sharedblock-nodea-vm-nic-protected --subnet sharedblock-
subnet-protected \
    --vnet-name sharedblock-vn --location westcentralus \
    --network-security-group sharedblock-nsg --private-ip-address-version IPv4
```

4. Create three VMs and attach the shared block volume using the Azure command **az vm create**. Option values are the same for each VM except that each VM has its own **<vm_name>**, **<new_vm_disk_name>**, and **<nic_name>**.

```
$ az vm create \
    -n <vm_name> -g <resource_group> --attach-data-disks
<shared_block_volume_name> \
    --data-disk-caching None --os-disk-caching ReadWrite --os-disk-name <new-vm-
disk-name> \
    --os-disk-size-gb <disk_size> --location <location> --size <virtual_machine_size> \
    --image <image_name> --admin-username <vm_username> --authentication-type
ssh \
    --ssh-key-values <ssh_key> --nics <nic_name> --availability-set <availability_set> --
ppg <proximity_placement_group>
```

For example, the following command creates a VM named **sharedblock-nodea-vm**.

```
$ az vm create \
-n sharedblock-nodea-vm -g sharedblock-rg --attach-data-disks shared-block-
volume.vhd \
```

```
--data-disk-caching None --os-disk-caching ReadWrite --os-disk-name sharedblock-nodea-vm.vhd \
--os-disk-size-gb 64 --location westcentralus --size Standard_D2s_v3 \
--image /subscriptions/12345678910-12345678910/resourceGroups/sample-azureimagesgroupwestcentralus/providers/Microsoft.Compute/images/sample-azure-rhel-9.3.0-20200713.n.0.x86_64 --admin-username sharedblock-user --authentication-type ssh \
--ssh-key-values @sharedblock-key.pub --nics sharedblock-nodea-vm-nic-protected --availability-set sharedblock-as --ppg sharedblock-ppg

{
  "fqdns": "",
  "id": "/subscriptions/12345678910-12345678910/resourceGroups/sharedblock-rg/providers/Microsoft.Compute/virtualMachines/sharedblock-nodea-vm",
  "location": "westcentralus",
  "macAddress": "00-22-48-5D-EE-FB",
  "powerState": "VM running",
  "privateIpAddress": "198.51.100.3",
  "publicIpAddress": "",
  "resourceGroup": "sharedblock-rg",
  "zones": ""
}
```

Verification

1. For each VM in your cluster, verify that the block device is available by using the **ssh** command with your VM's IP address.

```
# ssh <ip_address> "hostname ; lsblk -d | grep ' 1T '"
```

For example, the following command lists details including the host name and block device for the VM IP **198.51.100.3**.

```
# ssh 198.51.100.3 "hostname ; lsblk -d | grep ' 1T '"

nodea
sdb    8:16   0    1T  0 disk
```

2. Use the **ssh** command to verify that each VM in your cluster uses the same shared disk.

```
# ssh <ip_address> "hostname ; lsblk -d | grep ' 1T ' | awk '{print \$1}' | xargs -i udevadm info --query=all --name=/dev/{} | grep '^E: ID_SERIAL='"
```

For example, the following command lists details including the host name and shared disk volume ID for the instance IP address **198.51.100.3**.

```
# ssh 198.51.100.3 "hostname ; lsblk -d | grep ' 1T ' | awk '{print \$1}' | xargs -i udevadm info --query=all --name=/dev/{} | grep '^E: ID_SERIAL='"

nodea
E: ID_SERIAL=3600224808dd8eb102f6ffc5822c41d89
```

After you have verified that the shared disk is attached to each VM, you can configure resilient storage for the cluster.

Additional resources

- Configuring a GFS2 file system in a cluster
- Configuring GFS2 file systems

3.16. ADDITIONAL RESOURCES

- Support Policies for RHEL High Availability Clusters - Microsoft Azure Virtual Machines as Cluster Members
- Configuring and Managing High Availability Clusters