



Deployment Guide | Technical Doc

OPENSIFT VIRTUALIZATION

Deployment Overview

In this demonstration, I will walk you through the process of creating a custom golden image based on the VyOS 1.4.3 / 1.5 operating system and publishing it to OpenShift Virtualization. This allows customers to easily leverage custom templates for their deployments.

The workflow consists of the following key steps:

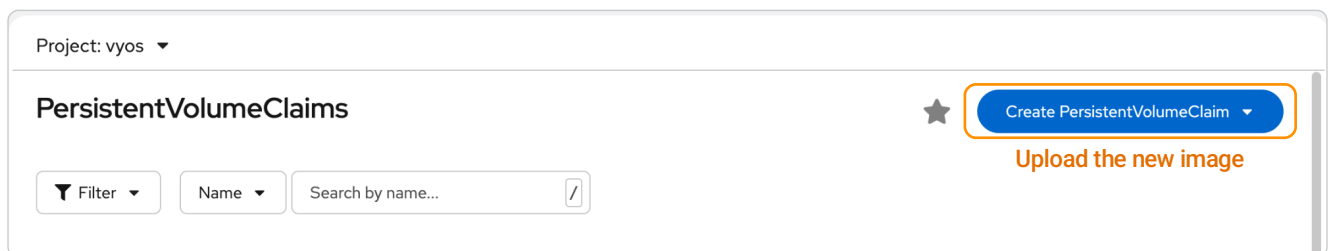
- Start with a custom **QCOW2 image** that includes predefined configurations such as users, packages, and hostnames.
- Publish this image to OpenShift Virtualization as a **Data Volume**, which will serve as the foundation for a new VM template.
- Create a template from the Data Volume.
- Deploy a virtual machine (VM) from the template and validate that all predefined configurations are applied correctly.

Uploading Your QCOW2 Image as a Data Volume

Before we can create a VM template, the QCOW2 image must first be uploaded as a Data Volume. This step ensures that the image can be referenced and reused by templates in OpenShift Virtualization.

Follow these steps:

1. **Navigate to Storage and select the option that supports Persistent Volume Claims (PVCs).**
2. **Upload your QCOW2 image into the chosen Persistent Volume Claim.**



Once the image is uploaded and available as a Data Volume, it can be linked to a template and used to spin up VMs with your custom VyOS configurations.

3. **Add this new format, which need to be chose as access mode RWX / Volumen: Filesystem**



Project: default ▼

Upload data to Persistent Volume Claim

Persistent Volume Claim creation
 This Persistent Volume Claim will be created using a DataVolume through Containerized Data Importer (CDI)

Upload data *

vyos-1.5-20250925131009-openshift-amd64.qcow2 Upload Clear

Persistent Volume Claim Name *

vyos-1.5-openshift

A unique name for the storage claim within the project

StorageClass *

SC nfs-csi ▼

☒ **Apply optimized StorageProfile settings**
 Use optimized access mode & volume mode settings from StorageProfile resource.

Size *

- 30 + GiB ▼

Ensure your PVC size covers the requirements of the uncompressed image and any other space requirements.

Access mode: ReadWriteMany / Volume mode: Filesystem

Upload Cancel

4. When the image is uploaded to our PVC, a template can be created using the data module. When the VM is started up, it will use the provided data volume as the boot volume for the VM with all the installation packages we requested

Create a template from the Data Volume

```
kind: Template
apiVersion: template.openshift.io/v1
metadata:
  name: vyos-1.5-devops-openshift
  namespace: vyos
  labels:
    flavor.template.kubevirt.io/medium: 'true'
    os.template.kubevirt.io/debian: 'true'
    template.kubevirt.io/type: vm
    vm.kubevirt.io/template: server.medium
    vm.kubevirt.io/template.namespace: vyos
    workload.template.kubevirt.io/server: 'true'
  annotations:
    iconClass: icon-debian
    name.os.template.kubevirt.io/vyos-1.5: VyOS 1.5 For DevOps
objects:
  - apiVersion: kubevirt.io/v1
```

```

kind: VirtualMachine
metadata:
  annotations:
    vm.kubevirt.io/validations: |
      [
        {
          "name": "minimal-required-memory",
          "path": "jsonpath:..spec.domain.resources.requests.memory",
          "rule": "integer",
          "message": "This VM requires more memory.",
          "min": 1073741824
        }
      ]
  labels:
    app: '${NAME}'
    vm.kubevirt.io/template: vyos-1.5-medium
    vm.kubevirt.io/template.revision: '1'
    vm.kubevirt.io/template.version: v0.19.4
  name: '${NAME}'
spec:
  dataVolumeTemplates:
    - metadata:
        name: '${NAME}-rootdisk'
      spec:
        storage:
          resources:
            requests:
              storage: 30Gi
          accessModes: []
          storageClassName: nfs-csi
        source:
          pvc:
            name: vyos-1.5-openshift
            namespace: vyos
  running: false
  template:
    metadata:
      annotations:
        vm.kubevirt.io/flavor: medium
        vm.kubevirt.io/os: debian
        vm.kubevirt.io/workload: server
      labels:
        kubevirt.io/domain: '${NAME}'
        kubevirt.io/size: medium
    spec:
      domain:
        cpu:
          cores: 2
          sockets: 1
          threads: 1
        devices:
          disks:
            - name: cloudinitdisk
              disk:
                bus: virtio
            - name: rootdisk
              bootOrder: 1
              disk:

```



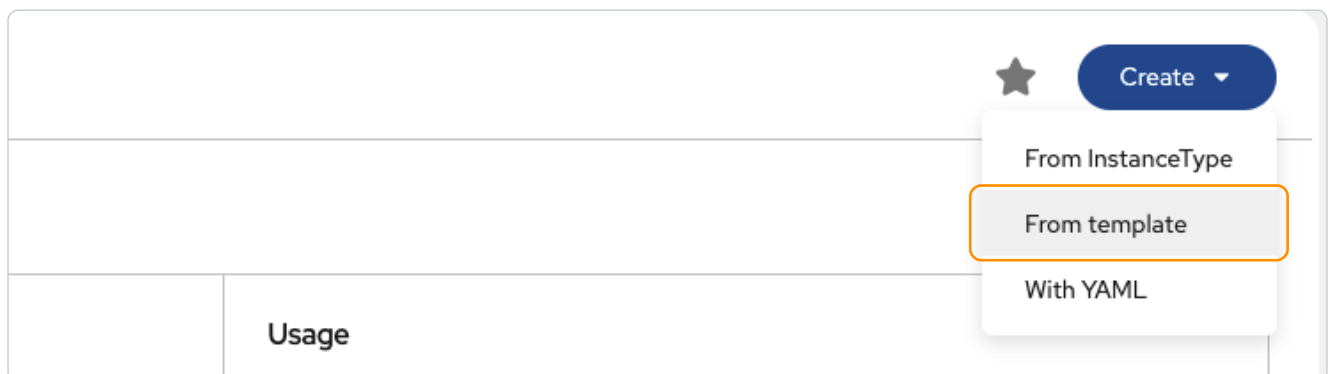
```

        bus: virtio
    interfaces:
      - name: nic-0
        model: virtio
        masquerade: {}
    networkInterfaceMultiqueue: true
    rng: {}
    gpus: []
    hostDevices: []
    resources:
      requests:
        memory: 4Gi
    evictionStrategy: LiveMigrate
    networks:
      - name: nic-0
        pod: {}
    terminationGracePeriodSeconds: 180
    volumes:
      - name: cloudinitdisk
        cloudInitNoCloud:
          userData: |
            #cloud-config
            user: vyos
            password: Vy0S-0p3nshift
            chpasswd: { expire: False }
      - name: rootdisk
        dataVolume:
          name: '${NAME}-rootdisk'
    hostname: '${NAME}'
parameters:
  - name: NAME
    description: Name for the new VM
    required: true
  - name: CLOUD_USER_PASSWORD
    description: Randomized password for the cloud-init user
    generate: expression
    from: '[a-z0-9]{4}-[a-z0-9]{4}-[a-z0-9]{4}'

```

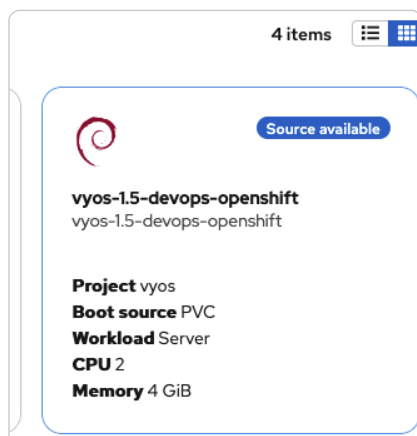
Deploy a VyOS VM from the template and validate that all predefined configurations are applied correctly.

5. Deploy a VM from our template with cloud-init, so, we go to Virtual-Machine and create from template



The screenshot shows the OpenShift Virtualization console interface. On the right side, there is a 'Create' button with a dropdown arrow. A dropdown menu is open, showing three options: 'From InstanceType', 'From template' (which is highlighted with an orange border), and 'With YAML'. Below the dropdown, there is a table with a header 'Usage'.

6. Choose our template, it will create the VM with our requirements



7. It shows the VM parameter, if we want to change something or leave it as default.

vyos-1.5-devops-openshift
vyos-1.5-devops-openshift

Server false

Description
N/A

Documentation
N/A

CPU | Memory
[2 CPU | 4 GiB Memory](#)

Network interfaces (1)

Name	Network	Type
nic-0	Pod networking	Masquerade

Disks (2)

Name	Drive	Size
cloudinitdisk	Disk	-
rootdisk	Disk	30 GiB

Disk size *

-

30

+

GiB

Drivers
☐ Mount Windows drivers disk

Optional parameters

Quick create VirtualMachine

VirtualMachine name *

vyos-1-5-new

Project
vyos

Public SSH key
[Not configured](#)

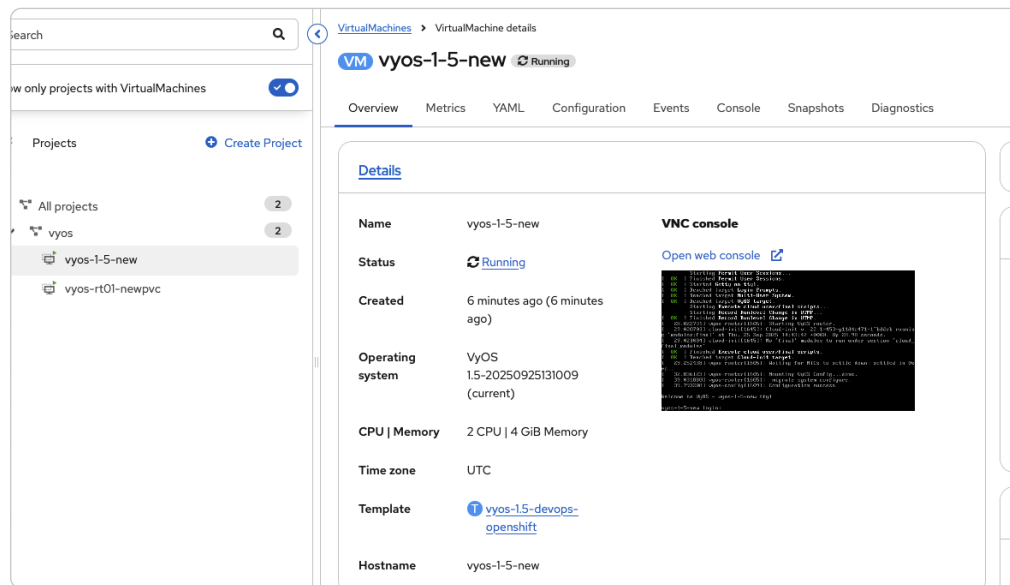
☐ Start this VirtualMachine after creation

Quick create VirtualMachine

Customize VirtualMachine

Cancel

8. After it, the provision of our VM is done, when it finished we can start the VM



9. Access with our credentials defined in the template

```
Welcome to VyOS!

┌───┐
└─ VyOS 1.5-20250925131009
└─ current

* Documentation: https://docs.vyos.io/en/latest
* Project news:  https://blog.vyos.io
* Bug reports:   https://vyos.dev

You can change this banner using "set system login banner post-login" command.

VyOS is a free software distribution that includes multiple components,
you can check individual component licenses under /usr/share/doc/*/copyright

Last login: Thu Sep 25 14:53:52 2025 from 10.128.0.209
vyos@vyos-1-5-new:~$
vyos@vyos-1-5-new:~$
vyos@vyos-1-5-new:~$
vyos@vyos-1-5-new:~$ show version
Version:      VyOS 1.5-20250925131009
Release train: current
Release flavor: openshift

Built by:     support@vyos.io
Built on:     Thu 25 Sep 2025 13:10 UTC
Build UUID:   9c0be458-153a-499e-ae1f-7bf6200d6151
Build commit ID: 12dbf5156a2a42

Architecture: x86_64
Boot via:     installed image
System type:  KVM guest
Secure Boot:  n/a (BIOS)

Hardware vendor: Red Hat
Hardware model: OpenShift Virtualization
Hardware S/N:
Hardware UUID: 6b68a81b-6eb4-50ea-86d2-5aa37aa1c3fe

Copyright:    VyOS maintainers and contributors
vyos@vyos-1-5-new:~$
```