# Verifying OpenShift Installations

After the OpenShift installation finishes successfully, administrators must ensure that the installed cluster is healthy and ready for day-2 tasks to onboard users and applications.

## OpenShift Cluster Health

From the bastion host, you can perform a basic health check using the `oc` command.

- Configure the `KUBECONFIG` environment variable to authenticate against the Kubernetes API with `cluster-admin` permissions.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

- Verify that all the cluster nodes have their system clock synchronized with a Network Time Protocol (NTP) server.

```
[user@demo ~]$ oc debug node/master01
...output omitted...
sh-4.4# chroot /host
sh-4.4# cat /etc/chrony.conf
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
pool 2.rhel.pool.ntp.org iburst
...output omitted...

sh-4.4# sudo chronyc tracking
Reference ID    : 8AEC8070 (time.gac.edu)
Stratum         : 3
Ref time (UTC)  : Thu Feb 11 13:06:57 2021
System time     : 0.000034756 seconds fast of NTP time
Last offset     : -0.000001187 seconds
RMS offset      : 0.004707427 seconds
Frequency       : 28.194 ppm fast
Residual freq   : -0.000 ppm
Skew            : 0.136 ppm
Root delay      : 0.052070152 seconds
Root dispersion : 0.018801220 seconds
Update interval : 64.9 seconds
Leap status     : Normal
```

The `chronyd` systemd service running on the cluster node uses the NTP pool `2.rhel.pool.ntp.org`. The system clock is synchronized with the NTP server `time.gac.edu`.

Repeat this procedure on all the cluster nodes.

- Verify that all the cluster nodes are in a `Ready` status.

If a cluster node is not in a `Ready` status, it cannot communicate with the OpenShift control plane and is unavailable to the cluster.

```
[user@demo ~]$ oc get nodes
NAME       STATUS   ROLES    AGE   VERSION
master01   Ready    master   15h   v1.19.0+9f84db3
master02   Ready    master   15h   v1.19.0+9f84db3
master03   Ready    master   15h   v1.19.0+9f84db3
worker01   Ready    worker   15h   v1.19.0+9f84db3
worker02   Ready    worker   15h   v1.19.0+9f84db3
```

- Check that all the cluster nodes are reporting usage metrics.

```
[user@demo ~]$ oc adm top node
NAME       CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
master01   677m         19%    4747Mi          31%
master02   391m         11%    3300Mi          22%
master03   519m         14%    4037Mi          27%
worker01   273m         7%     2435Mi          35%
worker02   313m         8%     2906Mi          42%
```

- Ensure that there are no certificate signing requests (CSRs) pending approval.

```
[user@demo ~]$ oc get csr | grep Pending
```

- Confirm that the cluster version operator report shows that the OpenShift cluster is available and ready.

```
[user@demo ~]$ oc get clusterversion
NAME      VERSION  AVAILABLE   PROGRESSING   SINCE   STATUS
version   4.6.4    True        False         22h     Cluster version is 4.6.4
```

- Check that all the cluster operators are available and ready.

  If the cluster is healthy, all the cluster operators should be available and not progressing unless one or more operators are still applying the configuration.

```
[user@demo ~]$ oc get clusteroperators
NAME                           VERSION AVAILABLE PROGRESSING DEGRADED   SINCE
authentication                 4.6.4   True      False       False      22h
cloud-credential               4.6.4   True      False       False      22h
cluster-autoscaler             4.6.4   True      False       False      22h
config-operator                4.6.4   True      False       False      22h
console                        4.6.4   True      False       False      22h
...output omitted...
```

- Verify that there are not any pods with scheduling or execution issues in the cluster.

```
[user@demo ~]$ oc get pods --all-namespaces | grep -v -E 'Running|Completed'
NAMESPACE   NAME   READY   STATUS   RESTARTS   AGE
```

## OpenShift Etcd Health

- Ensure that all the etcd cluster members are healthy.

```
[user@demo ~]$ oc get pods -n openshift-etcd | grep etcd-master
etcd-master01  3/3 Running  0  22h
etcd-master02  3/3 Running  0  22h
etcd-master03  3/3 Running  0  22h
```

```
[user@demo ~]$ oc rsh -n openshift-etcd etcd-master01
sh-4.4# etcdctl endpoint health --cluster
https://192.168.50.10:2379 is healthy: successfully committed proposal:
 took=10.8ms
https://192.168.50.12:2379 is healthy: successfully committed proposal:
 took=11.8ms
https://192.168.50.11:2379 is healthy: successfully committed proposal:
 took=12.1ms
```

## OpenShift API and Console Health

- Verify that the OpenShift API DNS record `api.ocp4.example.com` is configured to use the external load balancer IP address `192.168.50.254`.

```
[user@demo ~]$ dig api.ocp4.example.com
...output omitted...
;; QUESTION SECTION:
;api.ocp4.example.com. IN A

;; ANSWER SECTION:
api.ocp4.example.com. 85333 IN A 192.168.50.254

;; Query time: 0 msec
;; SERVER: 172.25.250.254#53(172.25.250.254)
;; WHEN: Thu Jan 28 05:11:46 EST 2021
;; MSG SIZE  rcvd: 71
```

- Verify that the OpenShift API is available by requesting the Kubernetes version.

```
[user@demo ~]$ curl -k https://api.ocp4.example.com:6443/version
...output omitted...
  "gitVersion": "v1.19.0+9f84db3",
...output omitted...
```

- Check that you can connect to the OpenShift Console.

```
[user@demo ~]$ curl -kIs \
> https://console-openshift-console.apps.ocp4.example.com
...output omitted...
HTTP/1.1 200 OK
...output omitted...
```

```
[user@demo ~]$ firefox https://console-openshift-console.apps.ocp4.example.com
```

## OpenShift Registry Health

- Ensure that the number of internal registry pods running on the OpenShift cluster matches its deployment configuration.

```
[user@demo ~]$ oc -n openshift-image-registry get deployment.apps/image-registry
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
image-registry   2/2     2            2           24h
```

- If there are multiple compute nodes, verify that each registry pod is running on a different compute node.

```
[user@demo ~]$ oc -n openshift-image-registry  get pods -o wide
...output omitted...
NAME                              READY STATUS   RESTARTS AGE IP          NODE
image-registry-69d88984fb-tjpkl 1/1   Running  0        16m 10.128.2.32 worker02
image-registry-59b67d44f6-n7wkq 1/1   Running  0        16m 10.131.2.12 worker01
...output omitted...
```

- From any cluster node, check the internal registry health.

```
[user@demo ~]$ oc debug node/master01
...output omitted...
sh-4.4# chroot /host
sh-4.4# curl -kIs \
> https://image-registry.openshift-image-registry.svc:5000/healthz
...output omitted...
HTTP/2 200
...output omitted...
```

- Verify that the internal registry deployment is using persistent storage. Also, ensure that the image registry operator is in the `Managed` management state.

```
[user@demo ~]$ oc get configs.imageregistry.operator.openshift.io cluster -o yaml
...output omitted...
spec:
  managementState: Managed
...output omitted...
  storage:
    pvc:
      claim: registry-claim
...output omitted...
```

## OpenShift Ingress Health

- Verify that the wildcard DNS record for applications, `*.apps.ocp4.example.com`, is configured to use the external load balancer IP address `192.168.50.254`.

```
[user@demo ~]$ dig test.apps.ocp4.example.com
...output omitted...
;; QUESTION SECTION:
;test.apps.ocp4.example.com. IN A
```

```
;; ANSWER SECTION:
test.apps.ocp4.example.com. 86358 IN A 192.168.50.254

;; Query time: 0 msec
;; SERVER: 172.25.250.254#53(172.25.250.254)
;; WHEN: Thu Jan 28 05:11:46 EST 2021
;; MSG SIZE  rcvd: 71
```

- Check that you can access an application exposed by an OpenShift Ingress route.

```
[user@demo ~]$ oc get routes -A | grep downloads
openshift-console downloads downloads-openshift-console.apps.ocp4.example.com
```

```
[user@demo ~]$ curl -kIs \
> https://downloads-openshift-console.apps.ocp4.example.com
...output omitted...
HTTP/1.0 200 OK
...output omitted...
```

- Ensure that the number of router pods running on the OpenShift cluster matches its deployment configuration.

```
[user@demo ~]$ oc -n openshift-ingress get deployment.apps/router-default
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
router-default   2/2     2            2           24h
```

- If there are multiple compute nodes, verify that each router pod is running on a different compute node.

```
[user@demo ~]$ oc -n openshift-ingress  get pods -o wide | grep router
NAME                          READY STATUS  RESTARTS AGE IP            NODE
router-default-b7567-l2z4d 1/1   Running 1        16h 192.168.50.13 worker01
router-default-b7567-qf8x4 1/1   Running 1        16h 192.168.50.14 worker02
```

## OpenShift Dynamic Storage Provider Health

When installing OpenShift in a supported cloud provider, the installer configures a dynamic storage provider. In this case, you must verify the status of the dynamic storage provider.

During the OpenShift installation on AWS using the full-stack automation method, the installer configures an AWS EBS dynamic storage provider. This dynamic storage provider uses the `aws-ebs` storage provisioner.

The OpenShift installation process creates a storage class named `gp2` that uses the AWS EBS dynamic storage provider as the back end. The `gp2` storage class dynamically provisions persistent storage for the containerized applications running on the OpenShift cluster. The OpenShift installation process configures the `gp2` storage class as the default storage class. Unless you specify a different storage class in the PVC definition, any PVC request will use the `gp2` storage class to create and bound the PV dynamically.

- Check the AWS EBS `gp2` storage class status.

```
[user@demo ~]$ oc get sc
NAME            PROVISIONER          RECLAIMPOLICY BINDINGMODE      EXPANSION AGE
gp2 (default)   kubernetes.io/aws-ebs   Delete  WaitForFirstConsumer true    32m
gp2-csi         ebs.csi.aws.com         Delete  WaitForFirstConsumer true     32m
```

The gp2 storage class uses the `WaitForFirstConsumer` volume binding mode. This volume binding mode delays the binding and provisioning of a `PersistentVolume` until a pod using the `PersistentVolumeClaim` is created. This configuration is immutable for this storage class.

- Verify that the `gp2` storage class works as expected.

  Create a simple httpd application that uses persistent storage for its `DocumentRoot` directory at `/var/www/html`.

```
[user@demo ~]$ oc new-project httpd-persistent
...output omitted...
[user@demo ~]$ cat /tmp/httpd-persistent.yaml
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: httpd-claim
  namespace: httpd-persistent
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
---
apiVersion: v1
kind: Pod
metadata:
  name: httpd
  namespace: httpd-persistent
spec:
  containers:
  - image: registry.redhat.io/rhel8/httpd-24:latest
    name: httpd
    ports:
    - containerPort: 8080
      name: http
      protocol: TCP
    volumeMounts:
    - mountPath: /var/www/html
      name: httpd-claim
  volumes:
  - name: httpd-claim
    persistentVolumeClaim:
      claimName: httpd-claim
```

```
[user@demo ~]$ oc create -f /tmp/httpd-persistent.yaml
persistentvolumeclaim/httpd-claim created
pod/httpd created
```

Verify that the PVC creation automatically triggers the PV provisioning and bounding through the gp2 default storage class.

```
[user@demo ~]$ oc get pvc
NAME          STATUS   VOLUME       CAPACITY   ACCESS MODES   STORAGECLASS   AGE
httpd-claim   Bound    pvc-d965cb1f  3Gi        RWO            gp2            29s
```

```
[user@demo ~]$ oc rsh httpd
sh-4.4$ df -h
Filesystem      Size  Used Avail Use% Mounted on
...output omitted...
/dev/nvme2n1   2.9G  9.0M  2.9G   1% /var/www/html
...output omitted...
```

## OpenShift Application Build and Deployment Test

- Build and deploy a test application to verify the OpenShift application build cycle.

```
[user@demo ~]$ oc new-project validate
...output omitted...
[user@demo ~]$ oc new-app django-psql-example
...output omitted...
```

```
[user@demo ~]$ oc get pods -n validate
NAME                          READY   STATUS      RESTARTS   AGE
django-psql-example-1-build   0/1     Completed   0          11m
django-psql-example-1-deploy  0/1     Completed   0          10m
django-psql-example-1-vfb5l   1/1     Running     0          10m
postgresql-1-bdgkk            1/1     Running     0          11m
postgresql-1-deploy           0/1     Completed   0          11m
```

```
[user@demo ~]$ oc logs -f django-psql-example-1-build
...output omitted...
Successfully pushed image-registry.openshift-image-registry.svc:5000/validate/
django-psql-example@sha256:b97b...ff82
Push successful
```

```
[user@demo ~]$ oc logs -f django-psql-example-1-deploy
...output omitted...
--> Scaling django-psql-example-1 to 1
--> Success
```

```
[user@demo ~]$ oc get routes -n validate
NAME                HOST/PORT                                                  PATH
 SERVICES            PORT  TERMINATION WILDCARD
django-psql-example django-psql-example-validate.apps.ocp4.example.com
 django-psql-example <all>             None
```

```
[user@demo ~]$ curl -Is \
> django-psql-example-validate.apps.ocp4.example.com
...output omitted...
HTTP/1.1 200 OK
...output omitted...
```

```
[user@demo ~]$ firefox http://django-psql-example-validate.apps.ocp4-
aws.example.com
```

## OpenShift Cluster Network

- Verify the OpenShift cluster network configuration.

```
[user@demo ~]$ oc get network.config/cluster -o yaml
apiVersion: config.openshift.io/v1
kind: Network
metadata:
...output omitted...
status:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  clusterNetworkMTU: 8142
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
```

## OpenShift Etcd Storage Performance

- Verify the etcd storage performance.

```
[user@demo ~]$ oc get pods -n openshift-etcd | grep etcd-master
etcd-master01  3/3 Running  0  22h
etcd-master02  3/3 Running  0  22h
etcd-master03  3/3 Running  0  22h
```

```
[user@demo ~]$ oc rsh -n openshift-etcd etcd-master01
...output omitted...
sh-4.4# etcdctl check perf --load="s"
 60 / 60 Booooooooooooooooooooooooooooooooooooooooooooooooooo! 100.00% 1m0s
PASS: Throughput is 150 writes/s
PASS: Slowest request took 0.220329s
PASS: Stddev is 0.018010s
PASS

sh-4.4# etcdctl check perf --load="m"
 60 / 60 Booooooooooooooooooooooooooooooooooooooooooooooooooo! 100.00% 1m0s
PASS: Throughput is 964 writes/s
PASS: Slowest request took 0.379547s
PASS: Stddev is 0.022218s
PASS

sh-4.4# etcdctl check perf --load="l"
 60 / 60 Booooooooooooooooooooooooooooooooooooooooooooooooooo! 100.00% 1m0s
```

```
FAIL: Throughput too low: 4586 writes/s
PASS: Slowest request took 0.258474s
PASS: Stddev is 0.032695s
FAIL
```

From the test result, the etcd cluster performs well for a medium cluster (`--load="m"`) and fails for a large cluster (`--load="l"`). For more information about etcd performance, visit the etcd documentation page at https://etcd.io/docs/current/op-guide/hardware/

For more detailed etcd storage performance information, use the `fio` tool from the `etcd-perf` container to run a performance test on the control plane nodes. The performance test output reports whether the disk is fast enough to host etcd by comparing the 99th percentile of the fsync metric captured from the run to see if it is less than **10 ms**.

```
[user@demo ~]$ oc debug node/master01
...output omitted...
sh-4.4# chroot /host
sh-4.4# podman run --volume /var/lib/etcd:/var/lib/etcd:Z
 quay.io/openshift-scale/etcd-perf
...output omitted...
{
  "fio version" : "fio-3.7",
...output omitted...
  "global options" : {
    "rw" : "write",
    "ioengine" : "sync",
    "fdatasync" : "1",
    "directory" : "/var/lib/etcd",
    "size" : "22m",
    "bs" : "2300"
  },
...output omitted...
      "write" : {
        "iops" : 328.808892,
...output omitted..
        },
...output omitted...
    }
  ]
}
---------------------------------------------
99th percentile of fsync is 6193152 ns
99th percentile of the fsync is within the recommended threshold - 10 ms, the disk
 can be used to host etcd
```

The `fio` performance test produces the following result:

1. This test writes 22 MiB of data in blocks of 2300 bytes on the `/var/lib/etcd` directory.
2. The 99th percentile of the fsync is 6193152 ns, which is equivalent to **6 ms** of write latency.
3. The operating system has achieved an average of **328 IOPS** during the test.

## OpenShift Machine API

When installing OpenShift in a supported cloud provider, the installer configures the compute node autoscaling using the OpenShift Machine API component. In this case, you must verify the status of the compute node autoscaling. One of the essential advantages of using the full-stack automation installation method on AWS is that the OpenShift installation process configures the compute node autoscaling.

The OpenShift Machine API is the component that defines and manages the OpenShift `Machines` resource. The OpenShift `Machines` resource represents the OpenShift cluster nodes. The OpenShift Machine API:

- Creates, updates, and deletes `Machines`
- Creates the infrastructure (instance or VM) for the node

You can use the OpenShift `MachineSets` resource to control sets of `Machines`. A `Machineset` represents:

- A set of `Machines`
- An abstraction of the underlying infrastructure

When installing OpenShift on AWS using the full-stack automation method, the OpenShift installer creates and configures a `MachineSet` for each availability zone in the selected region.

```
[user@demo ~]$ oc get machines -n openshift-machine-api
NAME                                 PHASE     TYPE         REGION      ZONE
       AGE
ocp4-aws-9r678-master-0              Running   m5.2xlarge   us-east-2   us-
east-2a   16h
ocp4-aws-9r678-master-1              Running   m5.2xlarge   us-east-2   us-
east-2b   16h
ocp4-aws-9r678-master-2              Running   m5.2xlarge   us-east-2   us-
east-2c   16h
ocp4-aws-9r678-worker-us-east-2a-gq2ps   Running   m5.4xlarge   us-east-2   us-
east-2a   16h
ocp4-aws-9r678-worker-us-east-2b-slp7l   Running   m5.4xlarge   us-east-2   us-
east-2b   16h
ocp4-aws-9r678-worker-us-east-2c-vj7pj   Running   m5.4xlarge   us-east-2   us-
east-2c   16h
```

```
[user@demo ~]$ oc get machinesets -n openshift-machine-api
NAME                            DESIRED   CURRENT   READY   AVAILABLE   AGE
ocp4-aws-9r678-worker-us-east-2a   1         1         1       1           16h
ocp4-aws-9r678-worker-us-east-2b   1         1         1       1           16h
ocp4-aws-9r678-worker-us-east-2c   1         1         1       1           16h
```

```
[user@demo ~]$ oc get nodes --label-columns \
> failure-domain.beta.kubernetes.io/region,failure-domain.beta.kubernetes.io/zone
NAME                                         STATUS   ROLES    AGE   VERSION
     REGION      ZONE
ip-10-0-151-177.us-east-2.compute.internal   Ready    master   16h
 v1.19.0+9f84db3   us-east-2   us-east-2a
```

```
ip-10-0-157-4.us-east-2.compute.internal     Ready    worker   16h
 v1.19.0+9f84db3   us-east-2   us-east-2a
ip-10-0-166-182.us-east-2.compute.internal   Ready    worker   16h
 v1.19.0+9f84db3   us-east-2   us-east-2b
ip-10-0-180-27.us-east-2.compute.internal    Ready    master   17h
 v1.19.0+9f84db3   us-east-2   us-east-2b
ip-10-0-205-233.us-east-2.compute.internal   Ready    master   17h
 v1.19.0+9f84db3   us-east-2   us-east-2c
ip-10-0-217-153.us-east-2.compute.internal   Ready    worker   16h
 v1.19.0+9f84db3   us-east-2   us-east-2c
```

Using the worker `MachineSets`, you can scale up (or down) the number of compute nodes running on the cluster. When scaling up a worker `MachineSet`:

- The OpenShift Machine API automatically provisions and starts an AWS EC2 instance for the new compute node.
- The new compute node gets its ignition configuration file and installs RHCOS.
- The new compute node joins the OpenShift cluster automatically.

```
[user@demo ~]$ oc scale machineset  ocp4-aws-9r678-worker-us-east-2c \
> --replicas=2 -n openshift-machine-api
machineset.machine.openshift.io/ocp4-aws-9r678-worker-us-east-2c scaled
```

```
[user@demo ~]$ oc get machinesets -n openshift-machine-api
NAME                              DESIRED   CURRENT   READY   AVAILABLE   AGE
ocp4-aws-9r678-worker-us-east-2a  1         1         1       1           17h
ocp4-aws-9r678-worker-us-east-2b  1         1         1       1           17h
ocp4-aws-9r678-worker-us-east-2c  2         2         1       1           17h
```

After a few minutes, the new compute node must be in `Ready` status.

```
[user@demo ~]$ oc get machinesets -n openshift-machine-api
NAME                              DESIRED   CURRENT   READY   AVAILABLE   AGE
ocp4-aws-9r678-worker-us-east-2a  1         1         1       1           17h
ocp4-aws-9r678-worker-us-east-2b  1         1         1       1           17h
ocp4-aws-9r678-worker-us-east-2c  2         2         2       2           17h
```

```
[user@demo ~]$ oc get machines -n openshift-machine-api
...output omitted..
ocp4-aws-9r678-worker-us-east-2c-65hln   Running   m5.4xlarge   us-east-2
 us-east-2c   3m41s
ocp4-aws-9r678-worker-us-east-2c-vj7pj   Running   m5.4xlarge   us-east-2   us-
east-2c   17h
```

```
[user@demo ~]$ oc get nodes --label-columns \
> failure-domain.beta.kubernetes.io/region,failure-domain.beta.kubernetes.io/zone
...output omitted..
ip-10-0-196-32.us-east-2.compute.internal    Ready    worker   2m31s
 v1.19.0+9f84db3   us-east-2   us-east-2c
ip-10-0-205-233.us-east-2.compute.internal   Ready    master   17h
 v1.19.0+9f84db3   us-east-2   us-east-2c
```

As you can see in the preceding example, the OpenShift Machine API automatically provisioned and added a new compute node to the cluster (`ip-10-0-196-32.us-east-2.compute.internal`) on the desired `us-east-2c` AWS AZ.

# Gathering OpenShift Data

When interacting with Red Hat Support to solve any OpenShift issue, administrators are asked to provide cluster debug information. Depending on the OpenShift component to troubleshoot, administrators can use different debug mechanisms.

## OpenShift Cluster Data

You can gather cluster debug information using the `oc adm must-gather` CLI command as the `cluster-admin` user. This CLI command collects the information from your cluster, such as:

- Resource definitions
- Audit logs
- Service logs

The execution of the `oc adm must-gather` command creates a new pod on the cluster. That pod collects the cluster data and stores it in a new directory. The new directory name starts with `must-gather.local`. The `oc adm must-gather` command creates this directory in the current working directory.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

```
[user@demo ~]$ cd ${HOME}
[user@demo ~]$ oc adm must-gather
[must-gather      ] OUT Using must-gather plugin-in image: quay.io/openshift-
release-dev/ocp-v4.0-art-dev@sha256:47..86
[must-gather      ] OUT namespace/openshift-must-gather-ndwcz created
[must-gather      ] OUT clusterrolebinding.rbac.authorization.k8s.io/must-
gather-5ptk2 created
[must-gather      ] OUT pod for plug-in image quay.io/openshift-release-dev/ocp-
v4.0-art-dev@sha256:47..86 created
[must-gather-276db] POD Wrote inspect data to must-gather.
...output omitted...
```

```
[user@demo ~]$ ls
install-config.yaml  must-gather.local.1227184995617480385/  ocp4-cluster/
```

```
[user@demo ~]$ find must-gather.local.1227184995617480385/
must-gather.local.1227184995617480385/
must-gather.local.1227184995617480385/timestamp
...output omitted...
```

As you can see, this directory contains the OpenShift resources definitions, services logs, and audit logs.

**Note**

When opening an OpenShift support case in the Red Hat Customer Portal, create a tar file with the output generated by the `oc adm must-gather` execution and attach it to the support case.

```
[user@demo ~]$ tar cvaf must-gather.tar.gz \
> must-gather.local.1227184995617480385/
...output omitted...
```

You can gather debug information about specific features using the `oc adm must-gather` CLI command with the `--image` or `--image-stream` argument.

```
[user@demo ~]$ oc adm must-gather \
> --image-stream=openshift/must-gather \
> --image=registry.redhat.io/container-native-virtualization/\
> cnv-must-gather-rhel8:v2.5.2
...output omitted...
```

For instance, using the `cnv-must-gather-rhel8` image the `oc adm must-gather` command collects OpenShift Virtualization specific data.

**Most Commonly Used Must-gather Images**

| Image | Purpose |
|---|---|
| registry.redhat.io/container-native-virtualization/cnv-must-gather-rhel8:v2.5.2 | Data collection for OpenShift Virtualization |
| registry.redhat.io/openshift-serverless-1/svls-must-gather-rhel8 | Data collection for OpenShift Serverless |
| registry.redhat.io/openshift-service-mesh/istio-must-gather-rhel7 | Data collection for Red Hat OpenShift Service Mesh |
| registry.redhat.io/rhcam-1-2/openshift-migration-must-gather-rhel8 | Data collection for migration-related information |
| registry.redhat.io/ocs4/ocs-must-gather-rhel8 | Data collection for Red Hat OpenShift Container Storage |
| registry.redhat.io/openshift4/ose-cluster-logging-operator | Data collection for Red Hat OpenShift cluster logging |

## OpenShift Node Data

In some scenarios, Red Hat Support will ask you to collect a `sosreport` file from a specific OpenShift cluster node. The `sosreport` command is a tool that collects configuration details, system information, and diagnostic data from Red Hat Enterprise Linux (RHEL) and Red Hat Enterprise Linux CoreOS (RHCOS) systems.

Red Hat recommends using a debug pod to generate a `sosreport` from an OpenShift cluster node.

```
[user@demo ~]$ oc debug node/ip-10-0-151-177.us-east-2.compute.internal
...output omitted...
sh-4.4# chroot /host
sh-4.4# toolbox
Trying to pull registry.redhat.io/rhel8/support-tools...
...output omitted...

[root@ip-10-0-151-177 /]# sosreport -k crio.all=on -k crio.logs=on
...output omitted...
Your sosreport has been generated and saved in:
  /host/var/tmp/sosreport-ip-10-0-151-177-1234-2021-01-11-crlsmlr.tar.xz

 Size   31.07MiB
 Owner  root
 md5    3ddfb7a774002fc8fb18da9c9c1534bc

Please send this file to your support representative.

[root@ip-10-0-151-177 /]# exit
sh-4.4# exit
sh-4.4# ls -lrt \
> /host/var/tmp/sosreport-ip-10-0-151-177-1234-2021-01-11-crlsmlr.tar.xz
-rw-------. 1 root root 32578896 Jan 11 13:42 /host/var/tmp/sosreport-
ip-10-0-151-177-1234-2021-01-11-crlsmlr.tar.xz
```

## OpenShift Remote Health Monitoring

OpenShift collects anonymized aggregated information about the health, usage, and size of the clusters. With this information, Red Hat can proactively react to issues that can impact customers.

The OpenShift cluster reports this information to Red Hat using two components:

- Telemetry
- Insights Operator

The Telemetry component sends a chosen subset of the cluster monitoring metrics to Red Hat. These metrics are sent continuously and describe:

- OpenShift cluster size
- OpenShift components health
- OpenShift upgrades health
- Limited OpenShift usage information
- OpenShift alerts summary info

The Insights Operator periodically gathers the cluster configuration and component failure status and reports that data to Red Hat. Using this information, the Red Hat OpenShift Cluster Manager proactively identifies potential cluster issues and provides solutions and preventive actions.

> **Note**
> For more information, refer to the *Remote health monitoring with connected clusters* guide in the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/support

You can access your cluster information using the Red Hat OpenShift Cluster Manager Console [https://cloud.redhat.com/openshift]. If you manage several OpenShift clusters, you will need your `cluster id` to identify your cluster in the Red Hat OpenShift Cluster Manager Console.

```
[user@demo ~]$ oc get clusterversion \
> -o jsonpath='{.items[].spec.clusterID}{"\n"}'
9d1c5e73-9deb-452b-b327-376d04315246
```

After retrieving the `cluster id`, open your web browser and navigate to Red Hat OpenShift Cluster Manager Console [https://cloud.redhat.com/openshift] using your Red Hat account. Then click your `cluster id` link and review your cluster information under the Overview, Monitoring, Insights, and Support navigation tabs.

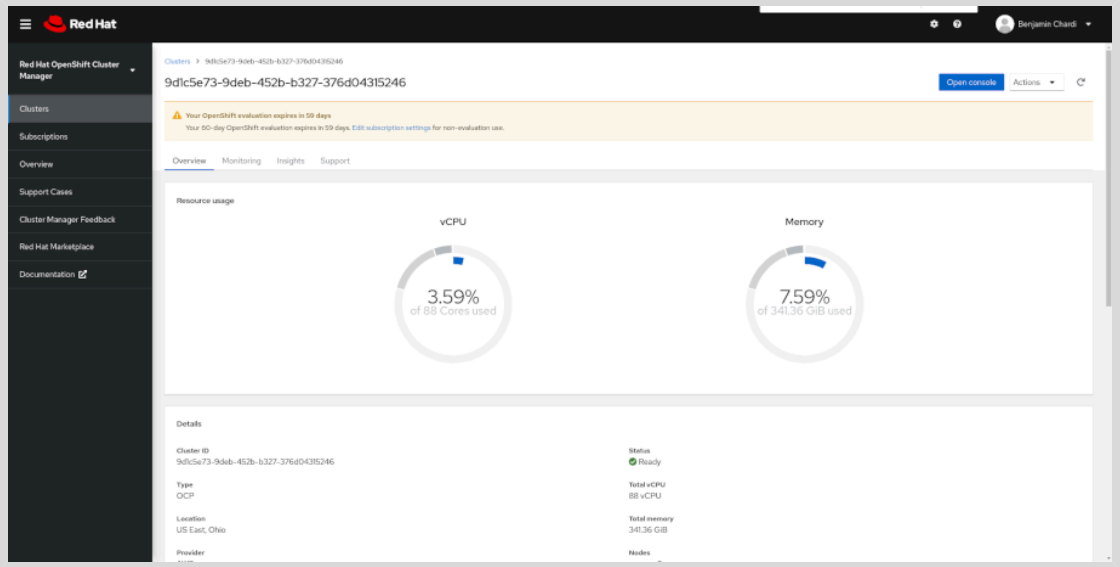


**Figure 1.9: Red Hat OpenShift Cluster Manager Console**

# Deleting an OpenShift Cluster

If you need to remove your OpenShift cluster, then you can run the OpenShift installer using the `destroy cluster` option.

```
[user@demo ~]$ openshift-install destroy cluster \
> --dir=${HOME}/ocp4-cluster
...output omitted...
```

**References**

- For more information, refer to the *Installation configuration* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing

- For more information, refer to the *Installing a cluster on AWS with customizations* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_aws