A translation for your language preference does not exist.

Infrastructure Nodes in OpenShift 4

SOLUTION VERIFICATO - Aggiornato 28 Giugno 2022 alle 13:31 - English

Ambiente

- Red Hat OpenShift Container Platform (RHOCP) - 4

Problema

Infrastructure nodes allow customers to isolate infrastructure workloads for two primary purposes:

1. to prevent incurring billing costs against subscription counts and
2. to separate maintenance and management.

This solution is meant to complement the official documentation on creating Infrastructure nodes in OpenShift 4. In addition there is a great OpenShift Commons video describing this whole process: OpenShift Commons: Everything about Infra nodes

To resolve the first problem, all that is needed is a node label added to a particular node, set of nodes, or machines and machineset. Red Hat subscription vCPU counts omit any vCPU reported by a node labeled `node-role.kubernetes.io/infra: ""` and you will not be charged for these resources from Red Hat. Please see How to confirm infra nodes not included in subscription cost in OpenShift Cluster Manager? to confirm your vCPU reports correctly after applying the configuration changes in this article.

To resolve the second problem we need to schedule infrastructure workloads specifically to infrastructure nodes and also to prevent other workloads from being scheduled on infrastructure nodes. There are two strategies for accomplishing this that we will go into later.

You may ask why infrastructure workloads are different from those workloads running on the control plane. At a minimum, an OpenShift cluster contains 2 worker nodes in addition to 3 control plane nodes. While control plane components critical to the cluster operability are isolated on the masters, there are still some infrastructure workloads that by default run on the worker nodes - the same nodes on which cluster users deploy their applications.

> **Note:** To know the workloads that can be executed in infrastructure nodes, check the "Red Hat OpenShift control plane and infrastructure nodes" section in OpenShift sizing and subscription guide for enterprise Kubernetes.

Planning node changes around any nodes hosting these infrastructure components should not be addressed lightly, and in general should be addressed separately from nodes specifically running normal application workloads.

Risoluzione

> **Note:** in OSD and ROSA it's not possible to create MachineSets/MachineConfigPools. Also, the infra nodes in OSD and ROSA are managed by Red Hat, and customer workloads cannot be executed there. Please refer to Create and configure MachineSets/MachinePools in OSD and ROSA to manage MachineSets using the OCM MachinePools.

Isolating Infrastructure Nodes

Applying a specific node selector to all infrastructure components will guarantee that they will be scheduled on nodes with that label. See more details on node selectors in placing pods on specific nodes using node selectors, and about node labels in understanding how to update labels on nodes.

Our node label and matching selector for infrastructure components will be `node-role.kubernetes.io/infra: ""`.

To prevent other workloads from also being scheduled on those infrastructure nodes, we need one of two solutions:

- Apply a taint to the infrastructure nodes and tolerations to the desired infrastructure workloads.
  OR
- Apply a completely separate label to your other nodes and matching node selector to your other workloads such that they are mutually exclusive from infrastructure nodes.

TIP: To ensure High Availability (HA) each cluster should have three Infrastructure nodes, ideally across availability zones. See more details about rebooting nodes running critical infrastructure.

TIP: Review the infrastructure node sizing suggestions

About the "worker" role and the MachineConfigPool

By default all nodes except for masters will be labeled with `node-role.kubernetes.io/worker: ""` . We will be adding `node-role.kubernetes.io/infra: ""` to infrastructure nodes.

The MachineConfigOperator reconciles any available MachineConfigs defined to match a specific selector in a MachineConfigPool. All custom MCP objects will descend from the parent worker pool as documented in custom pools. You do *not* need a custom pool unless you actually need to change a specific node to have a different set of MachineConfigs applied to it. You do *not* need a custom machineconfigpool or machineconfig for Infrastructure nodes to work correctly, as it is not a strict requirement for any of the particular problems here (node labeling, scheduling specific workloads, isolation, preventing other workload scheduling). You can, however, use one if you find it useful, and you can find an example of how to create one in the above link. For example, if you decide for some other reason all your infrastructure nodes should have a particular MachineConfig that does something specifically different from your worker nodes, you would use a MachineConfigPool to ensure that MachineConfig applies to those nodes particular to your pool.

However, if you want to *remove* the existing worker role from your infra nodes, you *will need* an MCP to ensure that all the nodes upgrade correctly. This is because the worker MCP is responsible for updating and upgrading the nodes, and it finds them by looking for this node-role label. If you remove the label, you must have a MachineConfigPool that can find your infra nodes by the infra node-role label instead. Previously this was not the case and removing the worker label could have caused issues in OCP <= 4.3.

This infra MCP definition below will find all MachineConfigs labeled both "worker" and "infra" and it will apply them to any Machines or Nodes that have the "infra" role label. In this manner, you will ensure that your infra nodes can upgrade without the "worker" role label.

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfigPool
metadata:
  name: infra
spec:
  machineConfigSelector:
    matchExpressions:
      - {key: machineconfiguration.openshift.io/role, operator: In, values:
[worker,infra]}
  nodeSelector:
    matchLabels:
      node-role.kubernetes.io/infra: ""
```

Configuring Infrastructure Nodes Using Node Selectors and Taints and Tolerations

Applying a taint to the infrastructure nodes and a toleration for that taint to all infrastructure components will guarantee that only those resources will be scheduled on the Infrastructure nodes. Taints can prevent workloads that do not have a matching toleration from running on particular nodes. However, some workloads such as daemonsets still need to be scheduled on these particular nodes. In this case, those workloads need a universal toleration. There *was* an outstanding issue where taints were causing issues with some infrastructure daemonset components that may not have had a universal toleration, but it has been resolved since RHBA-2020:3180 with 4.3.31, in RHBA-2020:2786 with 4.4.11 and RHBA-2020:240 with 4.5.1. See Critical DaemonSets Missing Universal Toleration for further information.

With MachineSets

If your cluster was installed using MachineSets to manage your Machines and Nodes, then you can use MachineSets to define your infrastructure nodes as well. See the official documentation: Creating Infrastructure Machinesets.

An example MachineSet with the required `nodeSelector` and `taints` applied might look like this:

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructureID>
  name: <infrastructureID>-infra-<zone>
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructureID>
      machine.openshift.io/cluster-api-machineset: <infrastructureID>-infra-<zone>
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructureID>
        machine.openshift.io/cluster-api-machine-role: infra
        machine.openshift.io/cluster-api-machine-type: infra
        machine.openshift.io/cluster-api-machineset: <infrastructureID>-infra-<zone>
    spec:
      metadata:
        labels:
          node-role.kubernetes.io/infra: ""
          node-role.kubernetes.io: infra
      taints:
      - effect: NoSchedule
        key: node-role.kubernetes.io/infra
        value: reserved
      - effect: NoExecute
        key: node-role.kubernetes.io/infra
        value: reserved
```

Without MachineSets

If you are not using the MachineSet API to manage your nodes, labels and taints are applied manually to each node:

Label it:

```
oc label node <node-name> node-role.kubernetes.io/infra=
oc label node <node-name> node-role.kubernetes.io=infra
```

Taint it:

```
oc adm taint nodes -l node-role.kubernetes.io/infra node-
role.kubernetes.io/infra=reserved:NoSchedule node-
role.kubernetes.io/infra=reserved:NoExecute
```

Moving Components to the Infrastructure Nodes

To move components to the infrastructure nodes, they must now have the infra Node Selector and a Toleration for the Taint assigned to the infrastructure nodes.
The following is an example, taken from the IngressController default, of what should be included in each respective resource spec to apply the node selector and toleration:

```
spec:
  nodePlacement:
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/infra: ""
    tolerations:
    - effect: NoSchedule
      key: node-role.kubernetes.io/infra
      value: reserved
    - effect: NoExecute
      key: node-role.kubernetes.io/infra
      value: reserved
```

Router

To move the router, the following patch on the `IngressController` will add both the node selector and Toleration:

```
oc patch ingresscontroller/default -n  openshift-ingress-operator  --type=merge -p
'{"spec":{"nodePlacement": {"nodeSelector": {"matchLabels": {"node-
role.kubernetes.io/infra": ""}},"tolerations": [{"effect":"NoSchedule","key": "node-
role.kubernetes.io/infra","value": "reserved"},{"effect":"NoExecute","key": "node-
role.kubernetes.io/infra","value": "reserved"}]}}}'
```

TIP: The router is configured by default to have only 2 replicas, but with 3 infrastructure nodes the following patch is required to scale to 3 routers.

```
oc patch ingresscontroller/default -n openshift-ingress-operator --type=merge -p
'{"spec":{"replicas": 3}}'
```

Registry

To move the registry, apply the following patch to the `config/cluster` object.

```
oc patch configs.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"nodeSelector": {"node-role.kubernetes.io/infra": ""},"tolerations":
[{"effect":"NoSchedule","key": "node-role.kubernetes.io/infra","value": "reserved"},
{"effect":"NoExecute","key": "node-role.kubernetes.io/infra","value": "reserved"}]}}'
```

Monitoring

Prometheus, Grafana and AlertManager comprise the default monitoring stack. To move these components create a config map with the required Node Selectors and Tolerations.

Define the ConfigMap as the `cluster-monitoring-configmap.yaml` file with the following:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: cluster-monitoring-config
  namespace: openshift-monitoring
data:
  config.yaml: |+
    alertmanagerMain:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
    prometheusK8s:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
    prometheusOperator:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
    grafana:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
    k8sPrometheusAdapter:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
```

```
        value: reserved
        effect: NoExecute
    kubeStateMetrics:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
    telemeterClient:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
    openshiftStateMetrics:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
    thanosQuerier:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoSchedule
      - key: node-role.kubernetes.io/infra
        value: reserved
        effect: NoExecute
```

Then apply it to the cluster::

```
oc create -f cluster-monitoring-configmap.yaml
```

Logging

Logging components can also be moved to the infrastructure nodes. Additional information for moving the logging resources can be found in the OCP Documentation.

Configuring Infrastructure Nodes Using only Node Selectors

If you prefer to avoid taints entirely, this is also possible through a specified scheduling behavior. You can schedule your own workloads to run on specific non-infrastructure nodes by applying a distinct label to these other nodes and either updating your default scheduler to choose this particular node label or specifically request this node label by project namespace.

With MachineSets

To add app nodes specific label in MachineSet enabled installation and we need to modify the MachineSet and label existing nodes manually. You modify the MachineSet so that any new Machines receive this particular label, but you also modify the current Nodes to add the label to them as well because the Machine API does not dynamically update Machines or Nodes from changes to the MachineSet that originally created them. By default the cluster may already have a "worker" MachineSet that can be repurposed for your app label.

```
$ oc patch machineset $MACHINESET_APP --type=merge -p '{"spec":{"template":{"spec":
{"metadata":{"labels":{"node-role.kubernetes.io/app":""}}}}}}'
$ oc patch machineset $MACHINESET_INFRA  --type=merge -p '{"spec":{"template":{"spec":
{"metadata":{"labels":{"node-role.kubernetes.io/infra":""}}}}}}'
$ oc patch machineset $MACHINESET_INFRA  --type=merge -p '{"spec":{"template":{"spec":
{"metadata":{"labels":{"node-role.kubernetes.io":"infra"}}}}}}'
$ oc label node <node-name> node-role.kubernetes.io/app=""
$ oc label node <node-name> node-role.kubernetes.io/infra=""
$ oc label node <node-name> node-role.kubernetes.io=infra
```

Without MachineSets

Worker nodes can be designated as `infra` nodes or `app` nodes through labeling.

1. Add a label to the worker node(s) you wish to act as app node(s):

   ```
   $ oc label node <node-name> node-role.kubernetes.io/app=""
   ```

2. Add a label to the worker node(s) you wish to act as infra node(s):

   ```
   $ oc label node <node-name> node-role.kubernetes.io/infra=""
   $ oc label node <node-name> node-role.kubernetes.io=infra
   ```

3. Check to see if applicable nodes now have the `infra` role and `app` roles. Not that the worker roles should remain.

   ```
   $ oc get nodes
   ```

Change the Scheduling Behavior for App Workloads

Create a default node selector, so pods without a nodeSelector will be assigned a subset of nodes to be deployed on, for example by default deploy in worker nodes.
As an example, the defaultNodeSelector to deploy pods on worker nodes by default would look like:

```
defaultNodeSelector: node-role.kubernetes.io/app=
```

This could also be performed with the following patch command:

```
$ oc patch scheduler cluster --type=merge -p '{"spec":{"defaultNodeSelector":"node-
role.kubernetes.io/app="}}'
```

Note: Prior to a bugfix in 4.6.1, `oc debug node` would not work on `master` or `infra` nodes after changing the `defaultNodeSelector` from the installation default. This issue is described in RHBZ #1812813 and oc debug node Fails When a Default nodeSelector is Defined

Note: When changing the default node selector on the scheduler, some projects need an explicit, empty project node selector for DaemonSets. For example, `openshift-logging` project. Review any other daemonsets you may have and the behavior you want.

```
oc annotate namespace openshift-logging openshift.io/node-selector=
```

Alternatively to changing the default scheduler for all workloads on the cluster, you could include this annotation on your namespace to control the scheduling behavior of all resources in that namespace. See How to configure project node selector in OpenShift 4 for project node selector details.

```
$ oc annotate namespace $PROJECT openshift.io/node-selector=node-
role.kubernetes.io/app=
```

Change the Scheduling Behavior for Infra Workloads

Move infrastructure resources to the newly labeled infra nodes. This would look exactly the same as above without the spec for tolerations. See further documentation about moving resources to infrastructure machine sets.

Causa Principale

Infrastructure nodes allow customers to isolate infrastructure workloads, but are not included in the OCP 4 default installation.

Operazioni di diagnostica

In a default OCP 4 installation, only `master` and `worker` MCP exists, and also only `master` and `worker` roles:

```
$ oc get mcp
NAME     CONFIG                                             UPDATED   UPDATING
DEGRADED   MACHINECOUNT   READYMACHINECOUNT   UPDATEDMACHINECOUNT
DEGRADEDMACHINECOUNT   AGE
master   rendered-master-xxxxxxxxxx   True      False      False      3              3
3                    0                    8h
worker   rendered-worker-yyyyyyyyyy   True      False      False      3              3
3                    0                    8h

$ oc get nodes
NAME                                          STATUS   ROLES    AGE    VERSION
master-0.lab.example.com   Ready    master   8h   v1.19.0+b00ba52
master-1.lab.example.com   Ready    master   8h   v1.19.0+b00ba52
master-2.lab.example.com   Ready    master   8h   v1.19.0+b00ba52
worker-0.lab.example.com   Ready    worker   8h   v1.19.0+b00ba52
worker-1.lab.example.com   Ready    worker   8h   v1.19.0+b00ba52
worker-2.lab.example.com   Ready    worker   8h   v1.19.0+b00ba52
```

Prodotto(s) **Red Hat OpenShift Container Platform** **Componente** **openshift-node**

Categoria **Configure**

Tag **billing** **configuration** **ocp_4** **openshift** **security** **shift_usability** **subscription**

This solution is part of Red Hat's fast-track publication program, providing a huge library of solutions that Red Hat engineers have created while supporting our customers. To give you the knowledge you need the instant it becomes available, these articles may be presented in a raw and unedited form.

People who viewed this solution also viewed

Unable to apply taints on infra node in RHOCP4

Solution - May 27, 2021

Node labels keeps changing in an OpenShift's node

Solution - Jan 10, 2020

Node feature discovery is only working on nodes with 'node-role.kubernetes.io/worker' label set

Solution - Aug 13, 2020

20 Comments

RED HAT

ACTIVE CONTRIBUTOR

110 Points

26 May 2020 8:49 AM

Carsten Lichy-Bittendorf

please read also the chapter Creating infrastructure MachineSets in the product documentation.

Rispondere

COMMUNITY MEMBER

51 Points

11 June 2020 8:00 AM

Jorge Martinez Garcia

I have applied preset guide and it broke machine-config-operator functionality on infra nodes. This is described in https://access.redhat.com/solutions/5061861, included in the "known issues". So, don't miss to read that part.

Rispondere

RED HAT

COMMUNITY MEMBER

28 Points

11 June 2020 1:42 PM

Ava Shulman

Hi Jorge, That is correct, there is currently a known issue affecting the machine config and dns daemon. There is a workaround however for the machine-config daemon and the following patch can be applied:
oc patch ds machine-config-daemon -n openshift-machine-config-operator --

type=merge -p '{"spec": {"template": { "spec": {"tolerations": [{"operator":"Exists"}]}}}}'
https://access.redhat.com/solutions/5061861 for more information

Rispondere

COMMUNITY MEMBER

50 Points

12 June 2020 2:38 PM

Thomas Müller

This document seems to conflict with this one: Creating an Infra Node in OpenShift v4 (4287111)

Rispondere

RED HAT

PRO

494 Points

21 July 2020 4:55 AM

Takayoshi Kimura RHCOE

Another, potentially cleaner node selector solution for the infra scheduling issue:

Applications scheduled to infra nodes in OpenShift 4:
https://access.redhat.com/solutions/5238051

Rispondere

PRO

417 Points

4 March 2021 12:26 PM

Sam Morris

Scratching my head that URL now redirects to *this* article...

Rispondere

RED HAT

COMMUNITY MEMBER

84 Points

5 March 2021 5:40 PM

Brian Ward RHCSA RHCOE

Sam, that article was consolidated into this one, and its content can be found under the section "Configuring Infrastructure Nodes Using only Node Selectors".

Rispondere

NEWBIE

19 Points

21 August 2020 8:51 AM

Mike Hulsman

Created an infra machineset on an vsphere OCP 4.5.4 with nodeselector and taints. Currently testing with only 3 masters and 3 infra machines. Got the problem that update's are not working because the monitoring is is stuck becauase of taint problems with the thanos-querier pods. Followed above procedures, modified the monitoring configmap with tolerations and taints, but the thanos-querier does not have the updated nodeselector and tolerations. because of thanos-querier is part of the monitoring stack, it should also be set in the monitoring configmap. The same is for kube-storage-version-migrator (in my case with only 3 masters and 3 infra nodes)

Rispondere

RED HAT

COMMUNITY MEMBER

28 Points

18 December 2020 5:39 PM

Ava Shulman

You should not taint infra nodes if your cluster is consists of only masters + infras. The only new behaviour it will cause is failed scheduling of any components without the correct toleration for infra nodes (or masters)

Rispondere

NEWBIE

12 Points

10 December 2020 10:00 AM

Apostolos Polymenakos

There's still contradicting information in this solution. The parts "The worker label should never be removed" and "You do not need a custom machineconfigpool or machineconfig for Infrastructure nodes to work correctly" refer to the GitHub MCO custom pools doc which actually describes the opposite! So which is it?

Rispondere

RED HAT

GURU

2618 Points

15 December 2020 9:56 AM

Oscar Arribas Arribas

If you use the MCP approach, you can remove the worker label from the nodes if your MCP inherits from the worker MCP:

```
spec:
  machineConfigSelector:
    matchExpressions:
      - {key: machineconfiguration.openshift.io/role, operator: In, values:
[worker,infra]}
[...]
```

Rispondere

RED HAT

COMMUNITY MEMBER

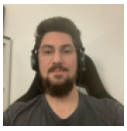
84 Points

15 December 2020 1:53 PM

Brian Ward RHCSA RHCOE

Thanks Oscar and Apostolos. The solution was updated to provide the MCP approach as well. The infra MCP is only required if you desire to remove the worker label. Originally the MCO team had built things such that removing the worker label was untested and may have caused indeterminate behavior. This is no longer the case, and as Apostolos has pointed out in the GitHub MCO link, we can see it has changed. Here is the specific commit to the docs where this was noted by the MCO team.

Rispondere



RED HAT

ACTIVE CONTRIBUTOR

152 Points

27 May 2021 9:28 AM

Mauro Oddi

The monitoring configmap is missing the thanos querier config which is mentioned in the docs [1]. i.e.:

```
[...]
    thanosQuerier:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: infra
        value: reserved
        effect: NoSchedule
      - key: infra
        value: reserved
        effect: NoExecute
```

[1] https://docs.openshift.com/container-platform/4.7/machine_management/creating-infrastructure-machinesets.html#binding-infra-node-workloads-using-taints-tolerations_creating-infrastructure-machinesets

Rispondere

RED HAT

COMMUNITY MEMBER

12 August 2021 12:16 PM

Andreas Furbach

54 Points

The MachineconfigSelector for the infra nodes will prevent applying different machineconfig to the infra nodes as those will be overwritten with the worker configs (lexicographic ordering of merging those configs) in some cases. One example is due to applying a CR for the kubelet.conf to match the systemReserved memory to the node sizing (see https://access.redhat.com/solutions/5843241). Even though `99-worker-generated-kubelet` and `99-infra-generated-kubelet-1` are both selected for the infra nodes, the infra mc is overwritten with the worker mc and all nodes get the same systemReserved memory.

I'm afraid this may also happen with other configurations.

Rispondere

EXPERT

1042 Points

13 August 2021 10:58 AM

Alexandros Phinikarides

I've bumped into this issue as well, because, as you mentioned, the machineConfigs are processed in natural sort order. There isn't any clear workaround, as the two KCSs below mention different things:

1. https://access.redhat.com/solutions/6145432 (creates a zz-infra MCP)
2. https://access.redhat.com/solutions/6119621 (apply the KubeletConfig by the creation time order)

Rispondere

RED HAT

COMMUNITY MEMBER

69 Points

3 November 2021 8:00 PM

Camilo Alberto Méndez León RHCA RHCE

Hi, I appied the "Configuring Infrastructure Nodes Using Node Selectors and Taints and Tolerations without MachineSets" instructions for a bare metal cluster, and I noticed the following:

1. The ConfigMap used for moving the monitoring stack does not mention the thanosQuerier pods. I'd add to the configmap this:

```
...
    thanosQuerier:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
      - key: infra
        value: reserved
        effect: NoSchedule
      - key: infra
        value: reserved
        effect: NoExecute
```

2. Using taints, the fluentd pods cannot be scheduled in the infra nodes. For solving this it is necesary add the following to the ClusterLogging custom resource, in tthe spec.collection.logs.fluentd section:

```
...
        tolerations:
        - key: infra
          value: reserved
          effect: NoSchedule
        - key: infra
          value: reserved
          effect: NoExecute
```

Rispondere

NEWBIE

10 Points

5 November 2021 1:01 PM

Daniel Wong

We are planning to have Azure Red Hat OpenShift cluster for the comming project, with nodes as below: - Master (Standard_D8s_v3) * 3 (total 24 vcores, 96Gi memory) - Worker (Standard_D8s_v3) * 3 (total 24 vcores, 96Gi memory)

As there will be **no Infra nodes**, and some Infra nodes workloads will run on the Worker nodes.

I find some of these workloads are listed in this website, such as ingress router, monitoring, log management, registry, etc: https://www.redhat.com/en/resources/openshift-subscription-sizing-guide

My questions are: 1. Can these workloads (or some of) to be run on Master nodes instead of Worker nodes by configuration? 2. For some workload cannot run on Master nodes, how many vcores will be consumed on Worker nodes?

Rispondere

RED HAT

ACTIVE CONTRIBUTOR

110 Points

5 November 2021 1:13 PM

Carsten Lichy-Bittendorf

Please discuss these kind of questions with a RH engineer via a support case

Rispondere

RC

COMMUNITY MEMBER

26 Points

16 February 2022 1:32 AM

Ryan Castle

Using either taints or defaultNodeSelector will result in a setup where end users can intentionally or unintentionally deploy workloads to infra nodes - either by adding tolerations or by overriding the defaultNodeSelector in a PodSpec.

Ensuring that *all* user namespaces have a `openshift.io/node-selector` constraint appears to be the only method of entirely preventing this. Updating the `oc new-project` template is a good start (https://docs.openshift.com/container-platform/4.8/applications/projects/configuring-project-creation.html).

Rispondere

KR

NEWBIE

9 Points

26 June 2022 10:42 AM

Khaled Raad RHCSA

After I applied it I had error in image-registry and elasticsearch MountVolume.MountDevice failed for volume "pvc-d668e08d-a169-4ee2-96e2-c602b2b45543" : kubernetes.io/csi: attacher.MountDevice failed to create newCsiDriverClient: driver name openshift-storage.cephfs.csi.ceph.com not found in the list of registered CSI drivers

Unable to attach or mount volumes: unmounted volumes=[registry-storage], unattached volumes=[kube-api-access-z8spd registry-storage registry-tls ca-trust-extracted registry-certificates trusted-ca installation-pull-secrets bound-sa-token]: timed out waiting for the condition

I fix it by following below articles suggested by Red Hat support

https://access.redhat.com/solutions/6047841
https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/4.6/html-single/managing_and_allocating_storage_resources/index#managing-container-storage-interface-component-placements_rhocs

Rispondere