Troubleshooting Cisco Container Platform

This appendix describes the problems that may occur during the installation and operation of Cisco Container Platform and the possible ways of resolving these problems.

It contains the following topics:

- Unable to Upgrade Cisco Container Platform due to Network Misconfiguration , on page 1
- Unable to Deploy NGINX Ingress Controller Using Helm, on page 1
- Unable to Start NGINX Ingress Controller Pod, on page 2
- Unable to Power on Worker VMs after a Shutdown, on page 2
- Application Pods Crash When Using Contiv CNI in Tenant Clusters, on page 2
- How to Create Sosreports, on page 4

Unable to Upgrade Cisco Container Platform due to Network Misconfiguration

When you enter a wrong IP address range for the Control Plane in the **Verify Network** screen of the **Upgrade** wizard, the following error message is appears:

```
Cannot patch address pool <uuid> with data: <some-data>
```

Recommended Solution

You must go back to the **Verify Network** screen of the **Upgrade** wizard and configure the IP address range for the Control Plane again.

For more information, see Upgrading Cisco Container Platform Control Plane.

Unable to Deploy NGINX Ingress Controller Using Helm

When deploying the NGINX Ingress controller using Helm fails as RBAC is not configured in Helm, the following error message appears:

```
It seems the cluster it is running with Authorization enabled (like RBAC) and there is no
permissions for the ingress controller. Please check the configuration
```

Recommended Solution

As Cisco Container Platform uses RBAC for authentication, Helm also needs to be configured to use RBAC.

Enable the RBAC parameter in Helm using the following command:

```
--set rbac.create=true
```

Unable to Start NGINX Ingress Controller Pod

When kube-proxy is used, setting both the `controller.service.externalIPs` and `controller.hostNetwork` variables to `true` for the NGINX-Ingress chart results in an invalid configuration.

Both kube-proxy and NGINX uses port 80 for communication, causing a port conflict, and the NGINX Ingress controller pod is set to the `CrashLoopBackOff` state.

The following error message appears:

```
Port 80 is already in use. Please check the flag --http-port
```

Recommended Solution

Ensure that both the `controller.service.externalIPs` and `controller.hostNetwork` variables are not set to `true` at the same time.

Unable to Power on Worker VMs after a Shutdown

Worker VMs may fail to power on after a shutdown and the following error message appears:

```
File system specific implementation of LookupAndOpen[file] failed.
```

Recommended Solution

Follow these steps to resolve the problem:

1. From the left pane, click the VM that you want to power on.
2. From the right pane, from the **Actions** drop-down list, choose **Edit Settings**.

   The **Edit Settings** window displays the multiple hard disks of the VM.
3. Except for the primary hard disk (Hard disk 1), click each hard disk, and then click the **Remove** icon.

   Ensure that the **Delete files from datastore** check box is not checked.
4. Click **OK**.

Application Pods Crash When Using Contiv CNI in Tenant Clusters

When you use Contiv as the CNI for a tenant cluster, you need to ensure that the application pods that need HugePages must have the following section in the pod manifest. Otherwise, the pods may crash.

```
resources:
   limits:
     hugepages-2Mi: 512Mi
     memory: 512Mi
```

The preceeding section in the pod manifest limits 512 MB in memory for HugePages for the pod. It allocates 256 HugePages, with each HugePage having 2MB size.

HugePages are allocated to the pods only if you have enabled HugePages on the host. Otherwise, the HugePage allocation in the pod manifest is ignored by Kubernetes. The following table shows the Cisco Container Platform CNIs that use HugePages.

Cisco Container Platform CNI	Use HugePages
Contiv	Yes
ACI	No
Calico	No

Example of Allocating HugePages for Applications

Step 1 Check the total and free HugePages on the worker nodes. Each HugePage is 2048 KB in size.

```
$ grep -i huge /proc/meminfo
AnonHugePages: 0 kB
ShmemHugePages: 0 kB
HugePages_Total: 1024
HugePages_Free: 972
HugePages_Rsvd: 0
HugePages_Surp: 0
Hugepagesize: 2048 kB

$ sudo sysctl -a | grep -i huge
vm.hugepages_treat_as_movable = 0
vm.hugetlb_shm_group = 0
vm.nr_hugepages = 1024
vm.nr_hugepages_mempolicy = 1024
vm.nr_overcommit_hugepages = 0
```

Step 2 If the host has less HugePages, increase the HugePages allocation.

```
sudo su
echo 2048 > /proc/sys/vm/nr_hugepages

# Check the increased number of HugePages
cat /proc/sys/vm/nr_hugepages
grep -i huge /proc/meminfo
sudo sysctl -a | grep -i huge
```

Note You need to perform these steps on all the hosts.

Step 3 Create the `bookinfo.yaml` file that allocates HugePages to the `reviews-v1` pod.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
name: reviews-v1
spec:
template:
    metadata:
    labels:
        app: reviews
        version: v1
```

```
    spec:
    containers:
    - name: reviews
        image: istio/examples-bookinfo-reviews-v1:1.5.0
        imagePullPolicy: IfNotPresent
        resources:
        limits:
            hugepages-2Mi: 512Mi
            memory: 512Mi
        ports:
        - containerPort: 9080
```

Step 4 Deploy `bookinfo.yaml` and check usage of HugePages.

```
$ kubectl create -f istio-$ISTIO_VERSION/samples/bookinfo/kube/bookinfo.yaml
deployment.extensions "reviews-v1" created

$ kubectl get pods | grep reviews
reviews-v1-6f56455f68-t6phs                                  1/1       Running   0          3m

# Check usage of HugePages by the pods
$ kubectl describe pod reviews-v1-6f56455f68-t6phs | grep -i '^Name:\|Image:\|huge\|mem'
Name:           reviews-v1-6f56455f68-t6phs
    Image:          istio/examples-bookinfo-reviews-v1:1.5.0
    hugepages-2Mi:  512Mi
    memory:         512Mi
    hugepages-2Mi:  512Mi
    memory:         512Mi

# Check usage of HugePages on each host
$ grep -i huge /proc/meminfo
AnonHugePages:         0 kB
ShmemHugePages:        0 kB
HugePages_Total:    1024
HugePages_Free:      972
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB

$ sudo sysctl -a | grep -i huge
vm.hugepages_treat_as_movable = 0
vm.hugetlb_shm_group = 0
vm.nr_hugepages = 1024
vm.nr_hugepages_mempolicy = 1024
vm.nr_overcommit_hugepages = 0
```

Step 5 Check the decrease of the `HugePages_Free` field in the output when the `reviews-v1` pod is using HugePages.

```
grep  -i huge /proc/meminfo
```

How to Create Sosreports

Sosreports are used by support engineers for troubleshooting customer support issues. They contain system log files, configuration details, and system information from your Cisco Container Platform environment.

Note

- For Control Plane issues, you need to run the sosreport from the Control Plane master VM, if available.
- For tenant cluster issues, you need to run the sosreport from the Control Plane master VM and the tenant plane master VM.
- For network issues impacting pods on a particular worker, you need to run the sosreport from the impacted tenant worker node.

Follow these steps to create an sosreport:

Step 1 ssh to the VM.

Step 2 Run sosreport on the node of your choice.

```
sudo sosreport
```

The sosreport is created and saved in the following location:

```
/tmp/sosreport-xxxxxx.tar.xz
```

Step 3 Validate the sosreport file using the following checksum:

```
xxxxxxxxx
```

Step 4 Securely transfer the sosreport file to your customer representative.
The file transfer method can vary depending on your deployment environment. For example, you can use Secure Copy (SCP) for Portable Operating System Interface systems (POSIX) and Windows Secure Copy (WinSCP) for windows clients. For more information, refer to Uploading Files to Cisco Technical Assistance Center (TAC).

Troubleshooting Cisco Container Platform