Troubleshooting Information

Feature Summary and Revision History

Summary Data

Table 1: Summary Data

Applicable Product(s) or Functional Area	SMI
Applicable Platform(s)	PCF
Feature Default Setting	Enabled – Configuration required to disable
Related Documentation	Not Applicable

Revision History

Table 2: Revision History

Revision Details	Release
First introduced.	2020.01.0

Debugging the PCF Deployment Issues

This section describes how to debug the issues that may occur when you deploy PCF through the SMI Deployer.

To debug the deployment issues, use the following checklist. If the checklist does not assist you in resolving the issue, analyze the diagnostic data that is available in the form of logs.

Table 3: Troubleshooting Checklist

Task	Resolution
Verify if the Ops Center is refreshing with the latest configurations	Manually verify if the configurations are refreshed. If the Ops Center is not refreshing or displaying the recent changes, then reinstall the helm charts. For information on reinstalling the charts, see Issue with Refreshing the PCF Ops Center, on page 3.
Validate if the external IPs and ports are accessible.	Use Telnet or any other application protocol and access the external IP address. This is to confirm that the IP address is accessible. If you are unsure of the IP address, run the following in the Kubernetes service to view the configured external IP addresses and port number: **`kubectl get services -n`** *`namespace`*
Ensure that the IP addresses and ports that are configured for PCF are open in the firewall.	Use the following command to open the ports: **`firewall-cmd –zone=public –add-port=`** *`port`***`/tcp –permanent`**
Confirm if PCF connects with the other NFs.	Use the following command on the master node to verify that a healthy connection is available between the NFs: **`nc -v`** Alternatively, from the proto VM, run the **nc -v** command on the Telnet CLI.

Task	Resolution
Validate that the successfully deployed helm chart is listed in the helm list.	Use the following steps to determine which helm chart is not listed in the helm list: 1. Run the following on the master node to view the list of deployed helm charts: **`helm list`** 2. If the helm chart is not found, run the following in the operational mode to view the charts irrespective of their deployment status. **`show helm charts`** 3. Review the pcf-ops-center logs to identify the helm chart which has the issue. Depending on the issue, take the appropriate action. Alternatively, you can review the consolidated set of logs, using the following command: **`kubectl logs -n namespace consolidated-logging-0`** For information about the event logs, see Event Logs.

Issue with Refreshing the PCF Ops Center

This section describes how to refresh the PCF Ops Center to display the latest configurations.

Problem

The PCF Ops Center is not considering the recent configurations due to which you may observe stale data or not get the expected response.

Resolution

You can refresh the PCF Ops Center using the basic and advanced steps. Perform the advanced steps only when the basic steps do not resolve the issue.

Basic

1. Run the following to undeploy PCF from the Ops Center:

   `system mode shutdown`

2. Use the following to manually purge any pending deployments from the helm:

   `helm delete --purge` *`helm_chart_name`*

3. From the master node, run the following to delete the configMaps from the namespace where PCF is installed:

   `kubectl delete cm` *`config_map_name`* **`-n`** *`namespace`*

4. Run the following to delete the product-specific configMaps from the CNEE namespace.
   a. Use the following to list the available configMaps:

      `kubectl get configmaps -n` *`namespace`*

From the list, determine the configMap that you want to delete.

b. Run the following to delete the configMap:

Kubectl delete configmap *configmap_name* **-n** *namespace*

5. Use the following commands to reinstall the helm chart. Once the chart is installed, a new instance of the PCF Ops Center is available.

helm upgrade –install *release name addR/chart_name* **-f** *filenames* **--namespace** *namespace*

Advanced

1. Remove the cnee-ops-center.
2. Delete the configMaps from the namespace.

   For more information on step 1 and 2, see the **Basic** steps.
3. Install the PCF Ops Center. For information on how to PCF Ops Center, see Deploying and Accessing PCF.

The recent configuration is not rendered because the responsible pods are not in a healthy state to process the refresh request. To investigate the issue at the pod level, review the pod's state.

Use the following command to view the pod's logs:

kubectl describe pod *pod_name* **-n** *namespace*

Alternatively, you can review the consolidated set of logs, using the following command:

kubectl logs -n namespace consolidated-logging-0

For information about the event logs, see Event Logs.

In the logs, the values in the Status and Ready columns indicate the following:

- If the Status column displays the state as Running, and the Ready column has the same number of containers on both sides of the forward-slash (/), then the pod is healthy and operational. This implies that the issue is at the application level. To investigate the application issue, check the logs of all the containers residing within the pods to detect the issue. Or, log into the container and review the logs.
- If the Status column displays the state as Pending, Waiting, or CrashLoopBackOff, then run the following to review the details such as the messages, reasons, and other relevant information:

  kubectl describe pod *pod_name* **-n** *namespace*
- If the Status is init or ContainerCreating, it signifies that the pod is in the process of starting up.
- If the Status is Running, and in the Ready column the number of containers on both sides of forward-slash (/) are different, then the containers have issues.

  Run the following to view the details:

  kubectl describe pod *pod_name* **-n** *namespace*

  When reviewing the details, if the Ready column has the value as false then it indicates that the corresponding container has issues. Review the associated logs to understand the issue.

- If the Status and Ready columns, and logs of the container do not indicate any issue, then verify that the required ingress or the service that is required to reach the application is up and running.

Subscriber Not Found or Primary Key Not Found

This section describes how to resolve the issues that report the Subscriber Not Found or Primary Key Not Found messages.

Problem

When the NFs cannot find the subscriber details, they send the Subscriber Not Found or Primary Key Not Found to PCF.

Resolution

1. Analyse the logs of the PCF Engine and REST endpoint pod for the subscriber or primary key related issues.

   On the master node, run the following command to determine the engine and rest-ep pod.

   kubectl logs -n *namespace pod_name*

2. Navigate to the pods and review the subscriber availability status and the subscriber count in the database. Based on the subscriber's status, take the appropriate action to resolve the issue.

   cdl show session *count/summary*

Message Routing Issues

This section describes how to troubleshoot the message routing issues.

Problem

You may observe a message routing failure when a message from the PCF endpoint incorrectly routes a message from Canary to the PCF Engine. The issue occurs when the message is sent to an incorrect PCF group.

Resolution

The following conditions might be causing the message routing failure. Check for these conditions and correct them, if necessary.

- From the PCF Ops Center, manually verify that the routing rules are configured correctly and they match the incoming traffic.
- Ensure that the Istio proxy is injected in the pcf-rest-ep pod.
- Verify that the virtual services are generated using the **istioctl** command. For more information on the traffic routing logs, see Collecting the Troubleshooting Information, on page 6.
- Enable the DEBUG level for com.cisco.pcf.endpoint.routing and review the pcf-rest-ep logs for any issues. Use the following command to enable the DEBUG level:

```
debug logging logger com.cisco.pcf.endpoint.routing level debug
```

Collecting the Troubleshooting Information

If you encounter issues in your PCF environment, gather and analyse the information associated to the failed action or process. Having this information enables you to detect the component that experiences the failure and resolve the issue faster.

The following table covers the components which might experience an issue, and the logs that contain the information corresponding to the issue.

Table 4: Issues

Issue	Logs
Deployment errors	Review the following logs to determine the issue. These logs assist you in identifying the component that may be the source of the error. Use the following commands on the master node: • View the available pods and review the pod status: **kubectl logs -n** *namespace pod_name* Depending on the pod's state, perform the appropriate remediation actions. To understand the pod's states, see States. • View the configured helm charts and their status: **helm list** • View the helm chart details for the REST endpoint: **helm get** *namespace* **-pcf-rest-ep**
Communication issues between the NFs	1. On the master node, run the following command to identify the pod that is responsible for the communication: **kubectl logs -n** *namespace pod_name* 2. Use the tcpdump utility to trace the packets.
Registration and deregistration issues	Use the following command to review the PCF REST endpoint logs: **helm get** *namespace* **-pcf-rest-ep**
Ops Center issues	Review the pod's log that hosts the Ops Center to determine the issue. **kubectl logs -n** *namespace pod_name* To resolve the issue, if you require the configuration information, then run one of the following commands: **show full-configuration** Or, **show running-config**

Issue	Logs
Traffic routing issues	To view the traffic routing-specific logs, use the following configuration: **kubectl get pod -o yaml -n** *namespace pcf-rest-ep pod_name* **istioctl get virtualservice -n** *namespace* **-o yaml** **istioctl get destinationrules -n** *namespace* **-o yaml** Also, review the logs of the following pods: • Pcf-rest-ep instance • Pcf-engine instance • Datastore or Session DB
Subscriber issues	Review the logs associated to the PCF Engine and REST endpoint to determine the issue. For additional information about the subscriber availability status and the subscriber count in the database, run the following command: **cdl show session** *count/summary*

Alerts

Alerts are notification messages that are generated when incidents requiring your attention or response occur. Review the historical and active alerts to determine the issue.

Alerts for PCF are generated through the CEE utility. To view these alerts, run the following command in the CEE Ops Center:

For active alerts:

```
show alerts active
```

For historical alerts:

```
show alerts history
```

Note You must have appropriate permission to view the alert details.

For information on application-based alerts, see PCF Application-Based Alerts.

Interface Error Codes

This section describes the codes that PCF reports for the interface errors.

Interface codes are generated as part of the logs or captured in the statistics.

The following tables describes the error and the corresponding codes:

Table 5: N7 Error Codes

Error	Error Code	Description
USER_UNKNOWN	400 Bad Request	The HTTP request is rejected because the end user who is specified in the request is unknown to the PCF.
ERROR_INITIAL_PARAMETERS	400 Bad Request	The HTTP request is rejected. This error is reported when the set of session or subscriber information which PCF requires for a rule selection is incomplete, erroneous, or unavailable for decision making. For example, QoS, RAT type, and subscriber information.
ERROR_TRIGGER_EVENT	400 Bad Request	The HTTP request is rejected because the set of session information sends a message that originated due to a trigger is incoherent with the previous set of session information for the same session. For example, trigger met was RAT changed, and the RAT notified is the same as before.
TRAFFIC_MAPPING_ INFO_REJECTED	403 Forbidden	The HTTP request is rejected because the PCF doesn't accept one or more of the traffic mappings filters provided by the SMF in a PCC Request.
ERROR_CONFLICTING_REQUEST	403 Forbidden	The HTTP request is rejected because the PCF can't accept the UE-initiated resource request as a network initiated resource allocation is already in-progress. This resource allocation has packet filters that cover the packet filters in the received UE-initiated resource request. The SMF rejects the attempt for a UE-initiated resource request.
POLICY_CONTEXT_DENIED	403 Forbidden	The HTTP request is rejected because the PCF doesn't accept the SMF request due to operator policies and local configuration.

Table 6: N28 Error Codes

Error	Error Code	Description
USER_UNKNOWN	400 Bad Request	The subscriber that is specified in the request isn't known at the CHF and the subscription can't be created.

Error	Error Code	Description
NO_AVAILABLE_POLICY _COUNTERS	400 Bad Request	There are no policy counters available for the subscriber at the CHF.

Note The generic error codes are applicable for all the network interfaces.

Table 7: Generic Error Codes

Error	Error Code	Description
TIMEOUT	408 Request Timeout	The HTTP request to the server took longer than the period the server is configured to wait.
OVERLOAD	429 Too Many Requests	The server has received too many consecutive requests to process within a short interval.
INTERNAL_ERROR	500 Internal Server Error	The server has encountered an unprecedented condition, which does not have an appropriate message.
SERVICE_UNAVAILABLE	503 Service Unavailable	The server cannot process the request because it is either, overloaded or is unavailable due to scheduled maintenance. This is a transient state.

Forwarding logs to the Splunk Server

This section describes how to enable PCF to forward the logs to the Splunk server.

Splunk is a third-party monitoring application that stores the log files and provides index-based search capability. You can configure PCF to send the logs securely to a Splunk server which could be an external server.

Important The Splunk server is a third-party component. Cisco does not take the responsibility of installing, configuring, or maintaining this server.

Use the following configuration to forward the logs to the Splunk server.

```
config
  debug splunk
    batch-count no_events_batch
```

batch-interval-ms*batch_interval_ms*
batch-size-bytes *batch_size*
hec-token*hec_token*
hec-url *hec_url*
end

The following is an example configuration:

```
configure
 debug splunk hec-url https://splunk.10.86.73.80.nip.io:8088
 debug splunk hec-token 68a81ab4-eae9-4361-92ea-b948f31d26ef
 debug splunk batch-interval-ms 100
 debug splunk batch-count 10
 debug splunk batch-size-bytes 102400
 end
```

NOTES:

- **debug splunk**—Enters the configuration debug mode.
- **batch-count** *no_events_batch*—Specify the maximum number of events to be sent in each batch.
- **batch-interval-ms** *batch_interval_ms*—Specify the interval in milliseconds at which a batch event is sent.
- **batch-size-bytes** *batch_size*—Specify the maximum size in bytes of each batch of events.
- **hec-token** *hec_token*—Specify the HTTP Event Collector (HEC) token for the Splunk server.
- **hec-url** *hec_url*—Specify the protocol, hostname, and HTTP Event Collector port of the Splunk server. The default port is 8088.

Pods stop running when PCF is upgraded through the Rolling Upgrade process

This section describes how to ensure that the pods are running when PCF is upgraded.

Problem

When the PCF version is upgraded to the subsequent available version, some pods such as CRD and Policy Engine stop running.

Resolution

Whenever you configure PCF ensure that you configure the following parameters:

- **db global-settings db-replica** *replica_count*
- **db spr shard-count** *shard_count*
- **rest-endpoint ips** *ip_address1, ip_address2, ip_address3*
- **rest-endpoint port** *port_number*
- **engine** *engine_name*

  replicas *replica_count*

unified-api-replicas *api_replica_count*

subversion-run-url *repository_url*

subversion-config-url *configuration_url*

tracing-service-name *service_name*

- **service-registration profile locality** *profile_name*
- **service-registration profile plmn-list [mcc mnc]**
- **service-registration profile snssais [sst sd]**