Persistent storage for policy configuration

Persistent storage for policy configuration

Persistent storage is a storage solution that:

- retains configuration data after power or network resources are disconnected,
- The CPC provides various storage technologies for managing the configuration data. The CPC has pre-defined storage such as the OpenStack Cinder volume used for storing the CRD data. CPC optionally stores the CRD data in shared storage such as OpenStack Cinder (default) or local storage. In the case of deployment on bare metal servers, CPC uses the local storage class along with the default storage layer as the persistent storage.
- uses Kubernetes Persistent Volume (PV) and Persistent Volume Claim (PVC) abstractions to decouple storage from individual pods.

Restore Capability

The Subversion repository stores the policy-specific configuration data in the XMI format. This repository resides in an SVN pod. If the SVN pod is restarted, the repository experiences a data loss. In such scenarios, you must reimport the configuration files to the SVN pod.

A new restore mechanism is introduced to protect the configuration data and maintain its integrity when the SVN pod restarts.

How persistent storage works

Restore Capability

The restore capability maintains the continuity of the policy configuration files in conditions where the SVN pod is restarted.

The policy configuration files are in the XMI format. Each SVN repository contains XMI files that are represented in a configMap. The configMap is updated whenever a policy configuration is modified and committed into an SVN repository. When the SVN pod is restarted, it verifies if the configMap is available and the corresponding XMI files are loaded to the repository.

The restore capability is managed through these configMaps:

- Monitor-svn-configmap-pcf: Contains configuration data in key-value pairs that represent the repository name and policy hash.
- Policy-svn-persistence-configmap: Contains the configured value of the policy-configuration-restore configMap.

Configure persistent storage

Enable persistent volume claim

This section describes how to enable persistent volume claim to configure persistent storage.

Procedure

To enable persistent volume claim, use the following configuration:

Example:

```
config
  k8s
  use-volume-claims [ true | false ]
  end
```

Command parameters:

- **config** — Enters the configuration terminal.
- **k8s** — Enters the Kubernetes configuration mode.
- **use-volume-claims [true | false]** — Configures whether volume claims are used during NF deployment.
  - When set to **true**, the default storage class, such as OpenStack Cinder, is enabled.
  - When set to **false**, data is stored in memory and is lost on a pod restart.

Assign persistent storage

Assign the storage class so that CPC Converged Policy knows which persistent volume to use when storing Custom Resource Definition (CRD) data.

Before you begin

Ensure that **use-volume-claims** is set to **true** before assigning a storage volume. See the Enable persistent volume claim section.

Procedure

To assign the persistent storage volume class, enter the following configuration in the CPC Converged Policy CLI:

Example:

```
config
 db
   global-settings
      volume-storage-class [ default | local ]
      end
```

Command parameters:

- **volume-storage-class [default | local]** — Configures the storage that gets assigned as the persistent storage.
  - **default** — Uses the OpenStack Cinder shared storage class.
  - **db** — Enters the database configuration mode.
  - **local** — Uses the local storage class, which is required for bare metal server deployments.

Configure the restore capability

Procedure

To configure the restore capability, enter the following configuration in the Policy Ops Center console:

Example:

```
config
   engine engine_name
     pcf policy-configuration-restore [ true | false ]
     end
```

Command parameters:

- **engine** *engine_name* — Specifies the engine for which the restore capability must be configured.
- **pcf policy-configuration-restore [true | false]** — Configures the capability that restores the configMap when the SVN pod restarts. The default value is **true**.

Configure MongoDB storage size

Feature description

Table 1: Feature description

Feature Name	Release Information	Description
Configure MongoDB storage size	2026.03.0	This enhancement allows to define custom Persistent Volume Claim (PVC) storage sizes for individual MongoDB replica sets.

The MongoDB storage configuration feature enables you to define custom Persistent Volume (PV) storage sizes for individual MongoDB replica sets. Previously, the PCF enforced a fixed 5 Gi limit for all MongoDB PVCs, which prevented operators from scaling storage to meet specific Subscriber Profile Repository (SPR) data volumes and retention policies. This enhancement introduces a two-level configuration hierarchy, a global default and a per-replica-set override that allows you to optimize storage for specific environments, reducing under-provisioning risks and costs.

Configure the storage parameter

Before you begin

Ensure that you meet these requirements:

- Ensure you have administrative access to the PCF Ops-Center CLI.
- Verify that the required storage values are in specified range (5- 250).
- Understand that storage configuration changes apply only to new PVC provisioning. You cannot resize existing PVCs through this configuration.

Define custom persistent volume storage sizes for MongoDB replica sets.

Procedure

Step 1 Log in to the PCF Ops-Center CLI and enter `config` mode.

Step 2 Configure the global default storage size for all Session Control Database (SCDB) instances.

```
scdb storage <value>
```

Step 3 Configure the storage size for a specific MongoDB replica set. This step overrides the global default.

```
scdb replica-name <replica-name> storage <value>
```

Step 4 `commit` the configuration changes.

Example:

Sample configuration

```
scdb:
  storage: "5Gi"
  replicaName:
    - name: session
      storage: "10Gi"
    - name: balance
      storage: "8Gi"
```

Step 5 After the system reaches 100% operational status, confirm that Kubernetes PVC resources reflect the update.

```
kubectl get pvc -n <namespace>
```

Example:

```
NAME                          STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS   AGE
db-local-data-session-0       Bound    pvc-12345678-abcd-1234-5678-1234567890ab   10Gi       RWO
        local-storage  2m
db-local-data-balance-0       Bound    pvc-87654321-dcba-4321-8765-0987654321ba   8Gi        RWO
        local-storage  2m
```

Note
Interpretation of results:

- **Success**: The system applies the new storage size to the PVC resources.
- **Syntax error**: Storage must be between 5Gi and 250 Gi.

Troubleshoot storage configuration

Use this section to troubleshoot and resolve common MongoDB storage configuration errors, including instances where the configuration fails to apply or the PVCs do not reflect the expected changes.

Table 2: Troubleshoot storage configuration

Issue	Corrective action
PVC resize failure	If the new size is not reflected by existing PVCs, check whether the Kubernetes cluster's storage class supports resizing.