Cisco Common Data Layer

Feature Summary and Revision History

Summary Data

Table 1: Summary Data

Applicable Product(s) or Functional Area	SMF
Applicable Platform(s)	SMI
Feature Default Setting	Disabled – Configuration Required
Related Changes in this Release	Not Applicable
Related Documentation	Not Applicable

Revision History

Table 2: Revision History

Revision Details	Release
Added the failure handling support.	2023.03.0
Added the procedures for configuration and verification of the event trace data in the CDL database record.	2021.02.0

Revision Details	Release
First introduced.	Pre-2020.02.0

Geo Redundancy of Cisco Common Data Layer in SMF

The Common Data Layer (CDL) is a high-performance next-generation Key-value (KV) data store layer for all the Cloud Native applications. These applications use the CDL as a state management with High Availability (HA) and Geo HA functions.

This feature enables SMF to support the Geo Redundant (GR) version of the CDL. The CDL endpoint pod is the front end for receiving requests from SMF application towards CDL. When the primary CDL endpoint fails, the SMF sends the database service requests to the next highly rated secondary endpoint. If the next rated endpoint is unavailable, then the SMF reattempts the operation on the subsequent endpoint that has the highest rating and so on.

In case of specific errors or failures from CDL to SMF, the retry mechanism is used for the Create, Find, Update, and Delete requests. The number of retries is based on the SLA.

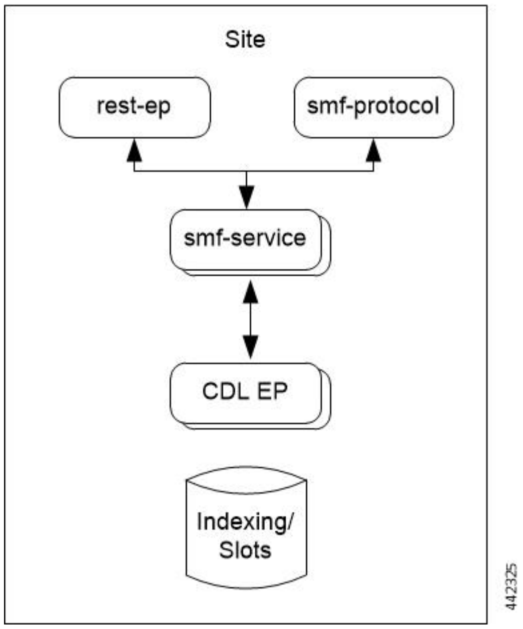
When the SMF receives the successful response during the retries, then the retries are stopped and the response is sent to the application.

With this functionality, SMF provides a non-disrupted N7 or Diameter message handling.

Architecture

The following figure depicts the failover that happens when the SMF service is unable to access the CDL datastore endpoint.

Figure 1: CDL Datastore Architecture



With relevance to this architecture, you can configure CDL through SMF Ops Center. When the SMF connects to the CDL, it uses the local endpoints.

How CDL works in SMF

When CDL is configured in SMF through the SMF Ops Center, SMF gets enabled to support multiple CDL datastore endpoints. You can configure the endpoints by specifying the IP addresses, ports, and assigning ratings to each endpoint. By default, SMF considers the local endpoint as the primary endpoint, which has the maximum rating. SMF performs CDL API operations on the primary endpoint. If this endpoint is unavailable, then SMF routes the operations to the next maximum rated endpoint. SMF keeps failing over to the accessible secondary endpoint or until all the configured secondaries are exhausted. It does not reattempt a query on the next rated endpoint if the endpoint is reachable but responds with error or timeout.

If SMF is unable to access any of the endpoints in the cluster, then CDL operation fails with the "Datastore Unavailable" error.

The following table lists the scenarios, expected behavior, and the related error responses from CDL.

Table 3: Scenarios, Expected Behavior, and Error Responses from CDL

Scenario	Expected Behavior	Error Response from CDL
All pods are active and healthy and trigger the CDL operations.	CDL Create, Read, Update, Delete (CRUD) operations are not retried.	Not applicable
CDL endpoint pod restarts and triggers CDL operations.	CDL CRUD operations are retried.	CDL sends the following error responses with the error code: • ERROR_FROM_INDEXING [501] • ERROR_FROM_SLOT [502] • DATASTORE_EP_QUEUE_FULL [503] • MAX_CAPACITY_REACHED [507

Call Flows

This section describes the call flow that is associated with this feature.

- CDL Endpoint Failure Call Flow, on page 3

CDL Endpoint Failure Call Flow

This section describes the SMF local data store endpoint failure call flow.

Figure 2: CDL Endpoint Failure Call Flow

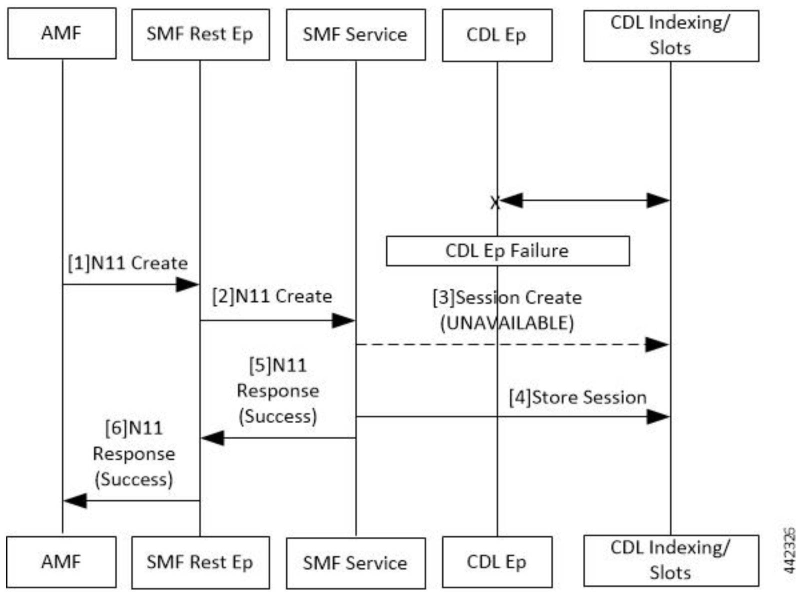


Table 4: CDL Endpoint Failure Call Flow Description

Step	Description
1	The AMF sends a Create Request to SMF REST endpoint over the N11 interface.
2	After receiving the request, the SMF REST endpoint forwards the Create Request to the SMF service.
3	The SMF service attempts to reach the CDL endpoint to send the session creation request. However, the CDL endpoint is unreachable.
4	The Create Request is evaluated in the stored session and the SMF service forwards the request to the CDL endpoint.
5	After the call request is successful, the SMF service notifies the Success Message to the SMF REST endpoint.
6	The SMF REST endpoint forwards the Success Message to the AMF.

Limitations

The CDL configuration in SMF has the following limitations:

- The SMF service attempts to reroute the calls only when it encounters gRPC errors, such as UNAVAILABLE. It does not acknowledge errors that the datastore endpoint returns and actual gRPC timeouts, such as DEADLINE_EXCEEDED gRPC status code.
- The SMF service does not resolve failures occurring with the datastore such as indexing and slot failures. The CDL layer must resolve these failures and if necessary, send an API call on the remote.
- By default, the maximum retry limit for a request is three. After reaching this limit, the retry is stopped and the response is sent to the application.

- On receiving the response message as success during the retries, the further retry is stopped and the response is sent to the application.

Configure the CDL through SMF Ops Center

The configuration of the CDL using SMF Ops Center involves the following steps:

1. Configure the CDL session database and define the base configuration, on page 5
2. Configure the Zookeeper in CDL, on page 7

Configure the CDL session database and define the base configuration

Follow these steps to configure the CDL session database in SMF:

Procedure

Step 1 Define the system ID, node type, and zookeeper replica server ID in the global configuration mode to set up the CDL session database.

CLI Command	Description	Range	Default Value
cdl system-id *system_id*	Specifies the system or Kubernetes cluster identity.	NA	1
cdl node-type *node_type*	Specifies the Kubernetes node label to configure the node affinity. The value must be a string.	0–64	session
cdl zookeeper replica *zookeeper_replica_id*	Specifies the zookeeper replica server's ID.	NA	NA

Step 2 Set the log level, cluster, replica, slot, index, and slice parameters to define the base configuration for the CDL datastore session and save the configuration.

CLI Command	Description	Range	Default Value
cdl logging default-log-level *debug_level*	**cdl logging default-log-level** *debug_level*—Specifies the default log level.	NA	NA
cluster-id *cluster-id*	**cluster-id** *cluster-id*—Specifies the Kubernetes cluster identity.	NA	1

CLI Command	Description	Range	Default Value
endpoint replica *num_replica*	**endpoint replica** *num_replica*—Specifies the number of replicas to be created. The value must be an integer.	*num_replica*—The value must be an integer in the range of 1 to 16.	*num_replica*—The default value is 1.
slot { replica *num_replica* | **map** *num_map/shards*| **write-factor** *write_factor*	• **replica** *num_replica*—Specifies the number of replicas to be created. • **map** *num_map/shards*—Specifies the number of partitions in a slot. • **write-factor** *write_factor*—Specifies the number of copies to be written before a successful response. **Note** Make sure that the value is lower than or equal to the number of replicas.	• *num_replica*—The value must be an integer in the range of 1 to 16. • *num_map/shards*—The value must be an integer in the range of 1 to 1024. • *write_factor*—The value must be an integer in the range of 0 to 16.	• *num_replica*—The default value is 1. • *num_map/shards*—The default value is 1. • *write_factor*—The default value is 1.
slot notification { host *host* | **port** *port* | **limit** *tps*	• **notification host** *host*—Specifies the notification server hostname or IP address. • **slot notification port** *port*—Specifies the notification server port number. • **slot notification limit** *tps* —Specifies the notification limit per second.	NA	• *host*—The default value is datastore-notification-ep . • *port*—The default value is 8890. • *tps* —The default value is 2000

CLI Command	Description	Range	Default Value
index { replica *num_replica* | **map** *num_map/shards* | **write-factor** *write_factor*	• **replica** *num_replica*—Specifies the number of replicas to be created. • **map** *num_map/shards*—Specifies the number of partitions in a slot. Avoid modifying this value after deploying the CDL. • **write-factor** *write_factor*—Specifies the number of copies to be written before a successful response.	• *num_replica*—The value must be an integer in the range of 1 to 16. • **map** *num_map/shards*—The value must be an integer in the range of 1 to 1024. • **write-factor** *write_factor* —The value must be an integer in the range of 0 to 16.	• *num_replica*—The default value is 2. • *num_map/shards*—The default value is 1. • *write_factor* —The default value is 1.
slice-names *cdl_slice_name*	**slice-names** *cdl_slice_name*—Specify the CDL slice names. The value must be an alphanumeric string.	*cdl_slice_name*—The value must be an integer in the range of 0 to 16.	NA

What to do next

Configure the Zookeeper in CDL

Configure the Zookeeper in CDL

Use these steps to configure the Zookeeper in CDL:

Procedure

Step 1 Specify the size of Zookeeper data storage in gigabytes in the global configuration mode.

cdl zookeeper data-storage-size *data_storage_size_in_gb*

Example:

```
[smf] smf(config)# cdl zookeeper data-storage-size 20 GB
```

Note
The default value is 20 GB. The value must be an integer in the range of 1 to 64.

Step 2 Specify the size of the Zookeeper data log storage in gigabytes.

log-storage-size *log_storage_size_in_gb*

Example:

```
[smf] smf(config-cdl-zookeeper)# log-storage-size 20 GB
```

Note
The default value is 20 GB. The value must be an integer in the range of 1 to 64.

Step 3 Specify the number of replicas to be created in the zookeeper.

replica *num_replicas*

Example:

```
[smf] smf(config-cdl-zookeeper)# replica 3
```

Note
The default value is 3. The value must be an integer in the range of 1 to 16.

Step 4 Specify the status of the JMX metrics.

enable-JMX-metrics *boolean_value*

Example:

```
[smf] smf(config-cdl-zookeeper-replica)# enable-JMX-metrics true
```

Note
The default value is true. The value must be boolean.

Step 5 Specify the status of the persistent storage for Zookeeper data and save the configuration.

enable-persistence *boolean_value*

Example:

```
[smf] smf(config-cdl-zookeeper-replica)# enable-persistence false
```

Note
The default value is false. The value must be boolean.

Sample configuration of CDL session database

This section shows a sample configuration of CDL in a HA environment.

```
config
cdl system-id system_id
cdl zookeeper replica num_zk_replica
cdl datastore session
 endpoint replica ep_replica
 index map index_shard_count
 slot replica slot_replica
 slot map slot_shard_count
 slice-names cdl_slice_name
exit
```

Configure event trace data

This section describes how to configure the SMF to store event trace data in a CDL database record. Enable or disable the event trace data storage in the CDL database record for system diagnostics. Event trace data shows the call flow event for subscribers.

Procedure

Step 1 Enable the storage of event trace data in the global configuration mode for system diagnostics.

system-diagnostics event-trace *[enable | disable]*

Example:

```
[smf] smf(config)# system-diagnostics event-trace  enable
```

Note
If you disable the *event_trace_state*, you save approximately 1 KB of database storage for each SMF database record.

Save and commit the configuration.

Step 2 [*Optional*] Use the **show running-config system-diagnostics event-trace** command to verify if the event trace is enabled.

Example:

```
 [smf] smf# show running-config system-diagnostics event-trace
 system-diagnostics event-trace enabled
```