Deploying and Configuring SMF through Ops Center

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
SMF deployment on bare metal server is supported and fully qualified in this release.	2021.01.0
First introduced.	Pre-2020.02.0

Feature Description

The SMF deployment and configuration procedure involves deploying the SMF through the Subscriber Microservices Infrastructure (SMI) Cluster Deployer and configuring the settings or customizations through the SMF Operations (Ops) Center. The Ops Center is based on the ConfD CLI. The SMF configuration includes the NRF profile data configuration and the externally visible IP addresses and ports.
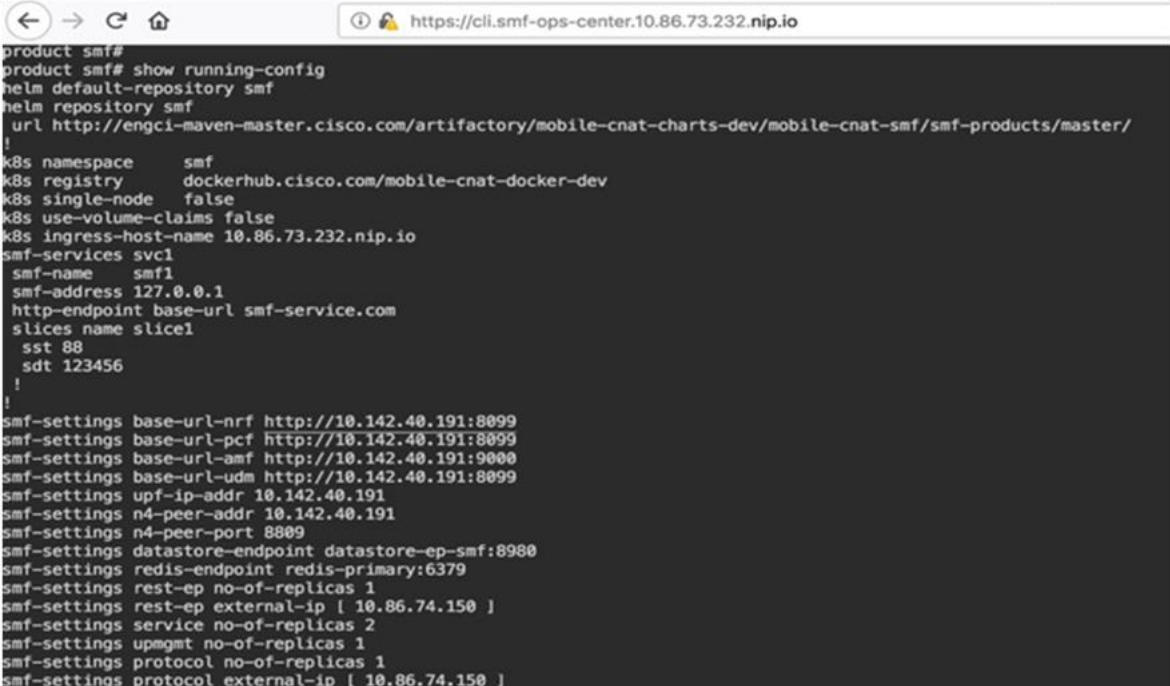
SMF Ops Center

The Ops Center is a system-level infrastructure that provides the following functionality:

- A user interface to trigger a deployment of microservices with the flexibility of providing variable helm chart parameters to control the scale and properties of Kubernetes objects (deployment, pod, services, and so on) associated with the deployment.

- A user interface to push application-specific configuration to one or more microservices through Kubernetes configuration maps.

- A user interface to issue application-specific execution commands (such as show and clear commands). These commands:

    - Invoke some APIs in application-specific pods

    - Display the information returned on the user interface application

The following screenshot is a sample of the web-based command line interface presented to the user.

Figure 1: Web-based CLI of Ops Center

The SMF Ops Center allows you to configure the features such as licensing, SMF engine, REST Endpoint, and CDL.

Prerequisites

Before deploying SMF on the SMI layer:

- Ensure that all the virtual network functions (VNFs) are deployed.

- Run the SMI synchronization operation for the SMF Ops Center and Cloud Native Common Execution Environment (CN-CEE)

Converged Core Refactoring Changes

This section describes the changes related to converged core refactoring in this chapter.

The Day1 SMF configuration is updated to include the **s11** and **sxa** interfaces in the GTP endpoint and Protocol endpoint configuration respectively.

Deploying and Accessing SMF

This section describes how to deploy SMF and access the SMF Ops Center.

Deploying SMF

The Subscriber Microservices Infrastructure (SMI) platform is responsible for deploying and managing the Cloud Native 5G SMF application and other network functions.

For information on how to deploy SMF Ops Center on a vCenter environment, see *Deploying and Upgrading the Product* section in the *Ultra Cloud Core Subscriber Microservices Infrastructure — Operations Guide*.

For deploying SMF Ops Center on a OpenStack environment, see *UAME-based VNF Deployment* section in the *UAME-based 4G and 5G VNF Deployment Automation Guide*.

For information on how to deploy SMF Ops Center on bare metal servers (currently Cisco UCS-C servers) environment, see *Operating the SMI Cluster Manager on Bare Metal* section in *Ultra Cloud Core Subscriber Microservices Infrastructure — Operations Guide*.

Accessing the SMF Ops Center

You can connect to the SMF Ops Center through SSH or the web-based CLI console.

- SSH:

  ssh admin@*ops_center_pod_ip* **-p 2024**

- Web-based console:

  1. Log in to the Kubernetes master node.

  2. Run the following command:

     kubectl get ingress *<namespace>*

The available ingress connections get listed.

3. Select the appropriate ingress and access the SMF Ops Center.

4. Access the following URL from your web browser:

   cli.*<namespace>*-**ops-center**.*<ip_address>*.**nip.io**

By default, the Day 0 configuration is loaded into the SMF.

Day 0 Configuration

To view the Day 0 configuration, run the following command.

show running-config

The following is a sample Day 0 configuration:

```
# show running-config
helm default-repository base-repos
helm repository base-repos
 url https://charts.10.192.1.111.nip.io/ccg.2021.01.0.i60
exit
k8s name           2nd-a18-kub-cluster
k8s namespace      cn-cn3
k8s nf-name        smf
k8s registry       docker.10.192.1.111.nip.io/ccg.2021.01.0.i60
k8s single-node    false
k8s use-volume-claims false
k8s ingress-host-name 10.84.104.34.nip.io
k8s nodes 2nd-a18-kub-cluster-master-11
 node-type    master
 worker-type master
exit
k8s nodes 2nd-a18-kub-cluster-master-22
 node-type    master
 worker-type master
exit
k8s nodes 2nd-a18-kub-cluster-master-33
 node-type    master
 worker-type master
exit
aaa authentication users user admin
 uid         1117
 gid         1117
 password    $1$XNGJOr.C$iZZvQbNfmPN15qG4GpQa8/
 ssh_keydir /tmp/admin/.ssh
 homedir     /tmp/admin
exit
aaa ios level 0
 prompt "\h> "
exit
aaa ios level 15
 prompt "\h# "
exit
aaa ios privilege exec
 level 0
  command action
  exit
  command autowizard
  exit
  command enable
  exit
```

```
           command exit
           exit
           command help
           exit
           command startup
           exit
         exit
         level 15
           command configure
           exit
         exit
       exit
      nacm write-default deny
      nacm groups group LI
       user-name [ liadmin ]
      exit
      nacm groups group admin
       user-name [ admin ]
      exit
      nacm rule-list admin
       group [ admin ]
       rule li-deny-tap
        module-name       lawful-intercept
        path              /lawful-intercept
        access-operations *
        action            deny
       exit
       rule li-deny-clear
        module-name       tailf-mobile-smf
        path              /clear/lawful-intercept
        access-operations *
        action            deny
       exit
       rule any-access
        action permit
       exit
      exit
      nacm rule-list confd-api-manager
       group [ confd-api-manager ]
       rule any-access
        action permit
       exit
      exit
      nacm rule-list ops-center-security
       group [ * ]
       rule change-self-password
        module-name       ops-center-security
        path              /smiuser/change-self-password
        access-operations exec
        action            permit
       exit
       rule smiuser
        module-name       ops-center-security
        path              /smiuser
        access-operations exec
        action            deny
       exit
      exit
      nacm rule-list lawful-intercept
       group [ LI ]
       rule li-accept-tap
        module-name       lawful-intercept
        path              /lawful-intercept
        access-operations *
```

```
 action           permit
 exit
 rule li-accept-clear
  module-name      tailf-mobile-smf
  path             /clear/lawful-intercept
  access-operations *
  action           permit
 exit
exit
nacm rule-list any-group
 group [ * ]
 rule li-deny-tap
  module-name      lawful-intercept
  path             /lawful-intercept
  access-operations *
  action           deny
 exit
 rule li-deny-clear
  module-name      tailf-mobile-smf
  path             /clear/lawful-intercept
  access-operations *
  action           deny
 exit
exit
```

SMF Service Configuration

The SMF service requires the basic configuration to process PDU Session Management API calls.

Configuring Pod-level Labels

☞

Important The pod-level labelling configuration is applicable only when the SMF is deployed on a bare metal server.

Use the following sample configuration to configure the SMF pod layout when the virtual machine is short of CPU and memory resources.

config

    endpoint protocol
      labels key *label_key* **value** *label_value*
      cpu { max-process *process_thread_count* **| request** *resource_request_number***}**

       memory { limit *max_resource_limit* **| request** *resource_request_number***}**
       end

NOTES:

- **labels key** *label_key* **value** *label_value*: Specify the K8 node affinity label key and value.

  label_key and *label_value* accept alphanumeric characters. For example, the key can be smi.cisco.com/protocol.

👉

Important   The pod-level configuration takes precedence over the layered node-level configuration, that is, at the protocol, service, or session level configuration.

- **cpu { max-process** *process_thread_count* | **request** *resource_request_number* **}**: Enables the K8 pod CPU configuration.

  - **max-process** *process_thread_count*: Specify the maximum number of parallel OS threads to use. *process_thread_count* must be an integer in the range of 1-32.

  - **request** *resource_request_number*: Specify the CPU resource request in millicores. *resource_request_number* must be an integer in the range of 100-1000000.

- **memory { limit** *max_resource_limit* | **request** *resource_request_number* **}**: Enables the K8 pod memory configuration.

  - **limit** *max_resource_limit* : Specify the maximum number of used memory resources in megabytes. *max_limit* must be an integer in the range of 100-200000.

  - **request** *resource_request_number*: Specify the memory resource request in megabytes. *request_number* must be an integer in the range of 100-200000.

Use the following table for node-level labelling.

Node	OAM	Protocol	CDL	SMF
Node 1	Yes	Yes	Yes	No
Node 2	Yes	Yes	Yes	No
Node 3	Yes	No	No	Yes
Node 4	No	No	No	Yes

Loading Day 1 Configuration

To load the Day 1 configuration for SMF, run the following command:

ssh admin@*ops_center_pod_ip* **-p 2024**   < *Day1config.cli*

✎

Note   The Day1config.cli file contains the necessary parameters required for the Day 1 configuration.

Alternatively, you can copy the configuration and paste it in the SMF Ops Center CLI to load the Day 1 configuration.

```
configure
  <Paste the Day 1 configuration here>
  commit
  exit
```

A sample *Day1config.cli* file, which contains the Day 1 configuration for SMF is shown below.

Day1config.cli

The following is a sample Day1config.cli file, which contains the Day 1 configuration for the SMF.