IP Address Management

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	• IPAM: Enabled – Always-on • Unique IP Pools for UPF: Disabled – Configuration required to enable • Auto-Reclamation of Under-Utilized IP Chunks: Disabled- Configuration required to enable • Identification of Corrupted Chunks: Disabled- Configuration required to enable. • Reconciliation of IP Chunks between SMF and UPF: Disabled- Configuration required to enable. • IP Chunk Auto-Throttle and ToD Chunk Clearance: Disabled- Configuration required to enable. • Route Aggregation to Handle Switch Limit: Disabled- Configuration required to enable.
Related Changes in this Release	Not Applicable
Related Documentation	Not Applicable

Revision History

Table 2: Revision History

Revision Details	Release
Added support for following features: • Auto-Reclamation of Under-Utilized IP Chunks. • Identification of Corrupted Chunks. • Reconciliation of IP Chunks between SMF and UPF. • IP Chunk Auto-Throttle and ToD Chunk Clearance. • Route Aggregation to Handle Switch Limit.	2023.03.0
Nexthop forwarding address configuration added to IPv6 address range and prefix range.	2023.01.4
Added support for the following features: • IPAM Periodic Reconciliation • UPF Fallback functionality	2023.01.0

Revision Details	Release
Added support for the following features: • IPAM reconciliation CLI commands for IPAM hardening. • IP pool allocation per slice and DNN feature. • SMF to allocate UPFs with unique IP pools.	2022.04.0
Added support for the following features: • New calls with static IP address. • Quarantine queue size. • IP address validation with CDL Configuration and statistics.	2021.02.0
IP Address Validation with CDL Configuration introduced.	2021.02.0
Updated quarantine time range to 3600 seconds.	2021.02.0
VRF Support introduced.	2020.02.5
First introduced.	Pre-2020.02.0

Feature Description

IP Address Management (IPAM) is a method of tracking and managing IP addresses of a network. IPAM is one of the core components of the subscriber management system. Traditional IPAM functionalities are insufficient in Cloud-Native network deployments. Hence, IPAM requires additional functionalities to work with the Cloud-Native subscriber management system. The Cloud-Native IPAM system is used in various network functions, such as SMF and PCF.

The IPAM system includes the following functionalities to serve the Cloud Native and Control and User Plane Separation (CUPS) architecture:

- **Centralized IP Resource Management**—Based on the needs of the Internet Service Provider (ISP), the Control Plane (CP) is deployed either on a single (centralized) cluster or multiple (distributed) clusters. For multiple cluster deployments, the IPAM automatically manages the single IP address space across the multiple CPs that are deployed in the distributed environment.
- **IP Address Range Reservation per User Plane**—For subscribers connecting to the Internet core, the User Plane (UP) provides the physical connectivity. The UP uses the summary routes to advertise subscriber routes to the Internet core. For CPs that are managing multiple UPs, the CP reserves a converged IP subnet to the UPs. In such a scenario, the IPAM splits the available address space into smaller address ranges and assigns it to different UPs.
- **IP Address Assignment from Pre-Reserved Address Ranges**—When subscribers request for an IP address, the IPAM assigns addresses from the pre-reserved address range of their respective UP.

- **IPv4 and IPv6 Pool Next Hop Address Ranges**—SMF supports next hop configuration for IPv4 and IPv6 pools along with address ranges and prefix ranges.

Note For uniform compatibility, the **nexthop-forwarding-address** configuration option is available in both the Internet Assigned Numbers Authority (IANA) and Identity Association for Prefix Delegation (IAPD) IPv6 configuration profiles. SMF does not use the IANA configuration but uses only the IAPD configuration. BNG uses the IANA IPv6 configuration..

How it Works

IPAM uses the following sub-modules for the Cloud-Native subscriber management system:

- IPAM Server—This module manages the complete list of pools and address space configurations. The IPAM server splits the configured address ranges into smaller address ranges statically or dynamically to distribute them to IPAM cache modules. The IPAM server is deployed as a centralized entity to serve group of Cloud-Native clusters or can be an integrated entity within a single cluster.
- IPAM Cache—This module receives the free address ranges from the IPAM server and allocates the individual IP addresses to the IPAM clients. Usually, the IPAM cache is deployed in a distributed mode running within each cluster to communicate with the co-located or remotely-located IPAM server. The IPAM cache also handles address range reservation per UP and pool threshold monitoring. The IPAM server and cache modules can run as an integrated mode.
- IPAM Client—This module handles the request and release of an individual IP address from the IPAM cache for each IP managed end-device. The IPAM client is tightly coupled with a respective network function.

IPAM Integration in SMF

Feature Description

The IP Address Management (IPAM) is a technique for tracking and managing the IP address space of a network. A core component of the subscriber management system, the IPAM, provides all the functionalities necessary for working with the Cloud-Native subscriber management system. Also, the IPAM acts as a generic IP address management system for the different network functions, such as the SMF and Policy Control Function (PCF).

Architecture

This section describes the IPAM integration in the SMF architecture.

IPAM Integration

The IPAM and SMF reside in the Application Services layer.

- SMF Node Manager Application—The SMF Node Manager application handles the UPF, ID resource, and IP address management. Hence, the SMF Node Manager application integrates IPAM cache and IPAM client modules. The UPF manager uses the IPAM client module for address range reservation per UPF.
- SMF Service Application— The SMF Service application provides PDU session services. During session establishment and termination, the IP addresses are requested and released back. The SMF Service application invokes the IPC to Resource Manager (RMGR) in Node Manager, which receives (free) the IP from the IPAM module.
- IPAM Server Application—Based on the deployment model, the IPAM Server application can run as an independent microservice, as a part of the same cluster, or in a remote cluster. For standalone deployments, the IPAM Servers are an integral part of the IPAM cache.

Components

This section describes the different components of the IPAM system.

IPAM Sub-Modules

The IPAM system includes the following sub-modules:

- **IPAM Server** – The IPAM server module manages the complete list of pools and address space configuration. It splits the configured address ranges into smaller address ranges (statically and dynamically) and distributes it to the IPAM cache modules. You can deploy the IPAM server either as a centralized entity to serve a group of cloud native clusters or as an integrated entity within a single cluster.
- **IPAM Cache** – The IPAM cache acquires free address ranges from the IPAM server and allocates individual IP addresses to the IPAM clients. Deployed in a distributed mode running within each cluster, the IPAM cache communicates with co-located and remotely located IPAM servers. Additionally, the IPAM cache takes care of the address range reservation per data plane and pool threshold monitoring.
- **IPAM Client** – The IPAM client module handles the request and release of the individual IP addresses from the IPAM cache for each IP managed end-device. Based on the use cases, the IPAM client module caters the needs of specific network functions (such as SMF, PCF, and so on).

How it Works

This section describes the call flows pertaining to the integration of the IPAM in the SMF.

Call Flows

The following call flow depicts the integration of the IPAM in the SMF.

Figure 1: IPAM Integration Call Flow

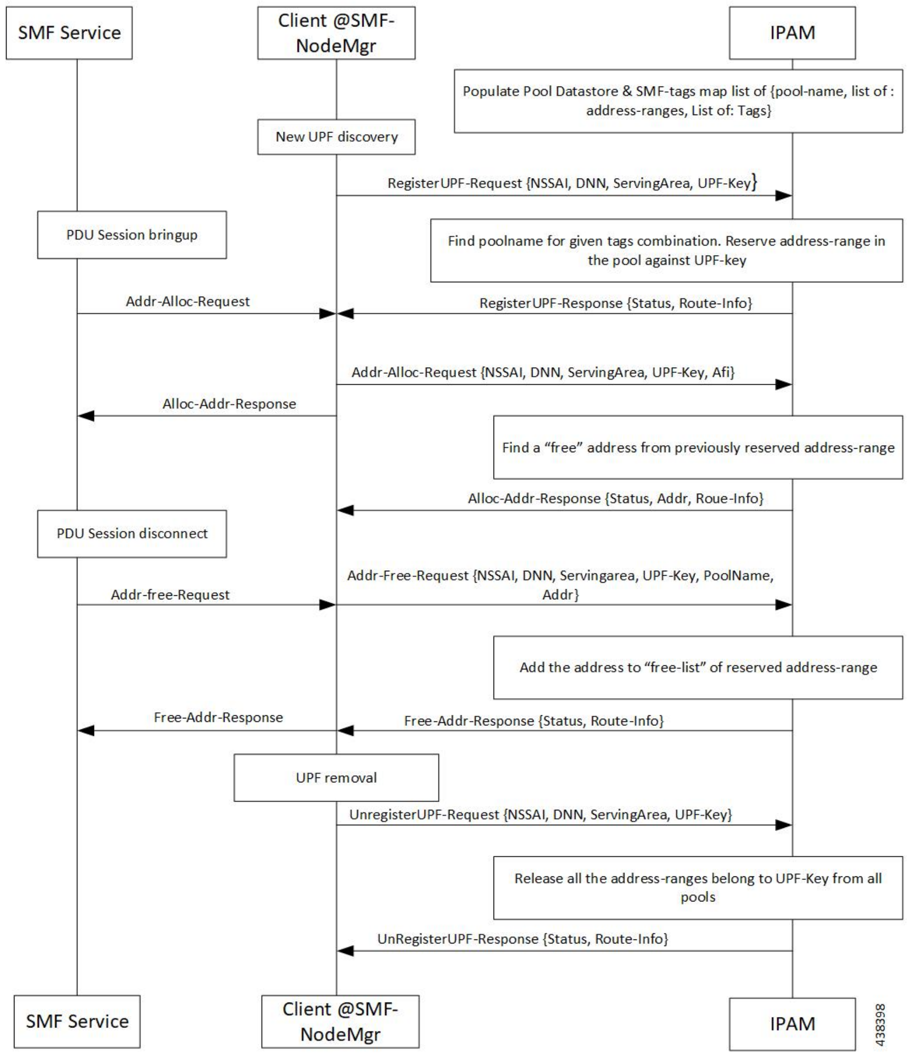

SMF Service
Client @SMF-NodeMgr
IPAM
Populate Pool Datastore & SMF-tags map list of {pool-name, list of : address-ranges, List of: Tags}
New UPF discovery
RegisterUPF-Request {NSSAI, DNN, ServingArea, UPF-Key}
PDU Session bringup
Find poolname for given tags combination. Reserve address-range in the pool against UPF-key
Addr-Alloc-Request
RegisterUPF-Response {Status, Route-Info}
Addr-Alloc-Request {NSSAI, DNN, ServingArea, UPF-Key, Afi}
Alloc-Addr-Response
Find a "free" address from previously reserved address-range
Alloc-Addr-Response {Status, Addr, Roue-Info}
PDU Session disconnect
Addr-Free-Request {NSSAI, DNN, Servingarea, UPF-Key, PoolName, Addr}
Addr-free-Request
Add the address to "free-list" of reserved address-range
Free-Addr-Response
Free-Addr-Response {Status, Route-Info}
UPF removal
UnregisterUPF-Request {NSSAI, DNN, ServingArea, UPF-Key}
Release all the address-ranges belong to UPF-Key from all pools
UnRegisterUPF-Response {Status, Route-Info}
SMF Service
Client @SMF-NodeMgr
IPAM
438398

Table 3: IPAM Integration Call Flow Description

Step	Description
1	IPAM populates the local cache and cache pod with the data configured under IPAM pool configuration. Split the address ranges according to the split size configured under address range.
2	The Node Manager (NM) receives UPF discovery or registration request.
3	The NM forwards the UPF registration request to IPAM for a given DNN or address type.
4	IPAM finds the pool for the given tag and address type and allocates a free address range against the given UPF key.
5	Register the UPF response, status, and route information.
6	The SMF service performs bring up of PDU session. The NM forwards the request to IPAM for the address allocation request.

Step	Description
7	IPAM receives the request for address allocation for tag, UPF key, Authority and Format Identifier (AFI), and Group Identifier (GrID).
8	IPAM allocates free address from the previously allocated address range and responds with the status and allocated address, and route information.
9	The SMF service performs bring down of PDU session. The NM forwards the request to IPAM for address release request.
10	IPAM receives the request for address release for pool name, UPF key, AFI, and GrID.
11	IPAM adds the address to free list of the reserved address range and responds with the status and route information.
12	IPAM receives UPF deregistration request with tag, UPF key, and AFI.
13	Release all the address ranges from the pool associated to the tag, UPF key, and AFI. Then, move the address range to the free list.
14	IPAM sends the UPF deregistration response along with the status and route information.

Configuring IPAM

This section describes how to configure the IPAM in the SMF.

Configuring the IPAM in the SMF involves the following steps:

1. Configuring IPv4 Address Ranges, on page 8
2. Configuring IPv6 Address Ranges, on page 9
3. Configuring IPv6 Prefix Ranges, on page 9
4. Configuring SMF Tags, on page 13
5. Configuring IPv4 Address Range Threshold, on page 14
6. Configuring IPv6 Address Range Threshold, on page 15
7. Configuring IPv6 Prefix Range Threshold, on page 15
8. Configuring IPv4 Address Range Split, on page 16
9. Configuring IPv6 Address and Prefix Address Range Split, on page 17
10. Configuring Global Threshold, on page 18
11. Configuring IPAM Source, on page 19

Note In release 2021.02 and later, IPAM pools must be associated to a Geographic Redundancy (GR) instance. That is, you must configure GR instance ID in the IPAM Configuration mode. This configuration is not backward compatible. If you are upgrading SMF to 2021.02 or a later release from a release prior to 2021.02, make sure you first remove the old IPAM configuration and apply the new configuration after the Ops center is accessible.

Configuring IPv4 Address Ranges

To configure the IPv4 address ranges, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            vrf-name vrf_name
            ipv4
               address-range start_ipv4_address end_ipv4_address
               commit
```

NOTES:

- **ipam**: Enter the IPAM configuration mode.
- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **vrf-name** *vrf_name*: Specify the virtual routing and forwarding (VRF) name of the pool.
- **ipv4**: Enter the IPv4 mode of the pool.
- **address-range** *start_ipv4_address end_ipv4_address*: Specify the start address and end address of IPv4 address range in dotted-decimal notation.

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p1
                  vrf-name one
                  ipv4
                     address-range 209.165.200.225 209.165.200.253
                     address-range 209.165.201.1 209.165.201.30
                     end
```

Verifying the IPv4 Address Range of a Pool

Use the **show ipam** *pool_name* **ipv4-addr** command to view the IPv4 address ranges for the given pool name. Based on the configuration, the address ranges are dynamically split. You can also view whether the address range is free or allocated to a data plane (user plane) using this command.

The following is an example output of the **show ipam** *pool_name* **ipv4-addr** command.

```
show ipam pool p1 ipv4-addr
=================================================================
Flag Indication: S(Static) O(Offline)
=================================================================
StartAddress          EndAddress    AllocContext     Flag
====================================================================
209.165.200.225    209.165.200.253  Upf-100
209.165.201.1      209.165.201.30   Upf-200
209.165.202.129    209.165.202.158  Free:NM1





====================================================================
```

Configuring IPv6 Address Ranges

To configure the IPv6 address ranges, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            vrf-name vrf_name
            ipv6
               address-range start_ipv6_address end_ipv6_address
               commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **vrf-name** *vrf_name*: Specify the VRF name of the pool.
- **ipv6**: Enter the IPv6 mode of the pool.
- **address-range** *start_ipv6_address end_ipv6_address*: Specify the start address and end address of IPv6 address range in colon-separated hexadecimal notation.

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p1
                  vrf-name one
                  ipv6
                     address-range 1::1 1::1000
                     address-range 2::1 2::1000
                     end
```

Configuring IPv6 Prefix Ranges

To configure the IPv6 prefix ranges, use the following sample configuration:

```
config
   ipam
      instance instance_id
         address-pool pool_name
            vrf-name vrf_name
            ipv6
            prefix-ranges
               prefix-range prefix_value prefix-length length
               commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of address pool. *pool_name* must be a string.
- **vrf-name** *vrf_name*: Specify the VRF name of the pool. *vrf_name* must be a string.
- **ipv6**: Enter the IPv6 mode of the pool.
- **prefix-ranges**: Enter the prefix ranges mode.
- **prefix-range** *prefix_value* **prefix-length** *length* : Specify the IPv6 prefix range and the IPv6 prefix length.

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p3
                  vrf-name three
                  ipv6
                     prefix-ranges
                        prefix-range 1:1:: prefix-length 48
                        prefix-range 2:1:: prefix-length 48
                        end
```

Verifying the IPv6 Address Prefix Range of a Pool

Use the **show ipam pool** *pool_name* **ipv6-prefix** command to view the prefix ranges for the given pool name. Based on the configuration, the address ranges are dynamically split. You can also view whether the address range is free or allocated to a data plane (user plane) using this command.

The following is an example output of the **show ipam pool** *pool_name* **ipv6-prefix** command.

```
show ipam pool p1 ipv6-prefix
=====================================================================================
Flag Indication: S(Static) O(Offline)
=====================================================================================
StartAddress                EndAddress                   AllocContext         Flag
=====================================================================================
aaaa:bbbb:ccc0::/64    aaaa:bbbb:ccc4::/64         Upf-100
aaaa:bbbb:dd00::/64    aaaa:bbbb:dd12::/64         Upf-200
bbbb:cccc:ee00::/64    bbbb:cccc:ee12::/64         Free:NM1
bbbb:cccc:ff00::/64    bbbb:cccc:ff12::/64         Free:NM0
xxxx:yyyy:zz00::/64    xxxx:yyyy:zz12::/64         Free:CP
```

Configuring IPv4 Address and Prefix Ranges with Next Hop Forwarding Address

To configure the IPv4 address with the next hop configuration for IPv4 pools/address ranges, use the following sample configuration:

```
configure
   ipam
      instance nstance_id
         address-pool pool_name
            ipv4
            address-ranges
               address-range start_ipv4_address end_ipv4_address
nexthop-forwarding-address nexthop_forwarding_address
               prefix-range prefix_value length  prefix_length
nexthop-forwarding-address nexthop_forwarding_address
               split-size per-cache number_of_addresses
               split-size per-dp number_of_addresses
               commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **ipv4**: Enter the IPv4 mode of the pool.
- **address-ranges**: Specify the starting address of the IPv4 address range. Enter the IPv4 address range and prefix range addresses with the next hop forwarding address.
  - **address-range** *start_ipv4_address end_ipv4_address* **nexthop-forwarding-address** *nexthop_forwarding_address*: Specify the starting and the ending addresses of the IPv4 address range with the next hop forwarding address.
  - **prefix-range** *prefix_value* **length** *prefix_length*: Specify the prefix value and the length within the IPv4 address.
  - **nexthop-forwarding-address** *nexthop_forwarding_address*: Specify the next hop forwarding address.
- **split-size per-cache** *number_of_addresses*: Specify the number of IPv4 addresses per chunk for IPAM cache allocation. Specify in the power of 2. The IPAM server consumes this configuration. *number_of_addresses* must be an integer in the range of 2-262144.
- **split-size-per-dp** *number_of_addresses*: Specify the number of IPv4 addresses per chunk for data plane allocation. Specify in the power of 2. The IPAM cache consumes this configuration.

  number_of_addresses must be an integer in the range of 2-262144.

Configuration Example

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p1
                  ipv4
                     split-size per-cache 1024
                     split-size per-dp 256
                     end
```

Configuring IPv6 Address Ranges with Next Hop Forwarding Address

To configure the IPv6 address with the next hop configuration for IPv6 pools and address ranges, use the following sample configuration:

```
configure
   ipam
      instance instance_id
```

```
        address-pool pool_name
          ipv6
          address-ranges
            address-range start_ipv6_address end_ipv6_address
nexthop-forwarding-address nexthop_forwarding_address
            prefix-range prefix_value length prefix_length
nexthop-forwarding-address nexthop_forwarding_address
              split-size per-cache number_of_addresses
              split-size per-dp number_of_addresses
              exit
              prefix-range prefix_value length prefix_length
nexthop-forwarding-address nexthop_forwarding_address
            commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **ipv6**: Enter the IPv6 mode of the pool.
- **address-ranges**: Specify the IPv6 address ranges and prefix range addresses with the next hop forwarding address.

  Note IANA IPv6 configuration is used by BNG.

  - **address-range** *start_ipv6_address end_ipv6_address* : Specify the starting and the ending addresses of the IPv6 address range.
  - **nexthop-forwarding-address** *nexthop_forwarding_address*: Specify the nexthop forwarding address.
  - **prefix-range** *prefix_value* **length** *prefix_length*: Specify the prefix value and length within the IPv6 address.
  - **nexthop-forwarding-address** *nexthop_forwarding_address*: Specify the next hop forwarding address.

- **prefix-ranges** : Specify the prefix ranges of an IPv6 address.

  Note SMF supports only IAPD IPv6 configuration.

  - **split-size per-cache** *number_of_addresses*: Specify the number of IPv6 addresses per chunk for IPAM cache allocation.
  - **split-size-per-dp** *number_of_addresses*: Specify the number of IPv6 addresses per chunk for the Data plane allocation.
  - **prefix-range** *prefix_value* **length** *prefix_length* **nexthop-forwarding-address** *nexthop_forwardng_address*: Specify the prefix value and the length within the IPv6 address with the next hop forwarding address.

Configuration Example

The following is an example configuration.

```
ipam
 instance 1
  address-pool ISE-Pool1
   vrf-name ISP
   tags
    dnn cisco_vlan400.com
   exit
   ipv6
    address-ranges
      address-range 1000::1 1000::ffff nexthop-forwarding-address :9001::3
      prefix-range 2607:fc20:1010:: length 98 nexthop-forwarding-address :9001::3
    prefix-ranges
       split-size
         per-cache 32768
         per-dp    32768
       exit
       prefix-range 2607:fc20:1010:: length 44 nexthop-forwarding-address :9001::3
   exit
  exit
```

Configuring SMF Tags

To configure the SMF tags, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            tags
               nssai nssai_value
               dnn dnn_name
               serving-area serving_area_value
               commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **tags**: Specify the pool tags to set additional properties for a pool in generic manner.
  - **nssai** *nssai_value*: Specify the NSSAI tag for the pool. *nssai_value* must be a string.
  - **dnn** *dnn_value*: Specify the location DNN or DNN tag for the pool. *dnn_value* must be a string.

Note

- Based on **pool-selection nssai** configuration, the SMF sends the "slice + dnn" as tag to IPAM.
- The NSSAI value must match the SMF slice configuration name.

  - **serving-area** *serving_area_value*: Specify the serving area tag for the pool. *serving_area_value* must be a string.

Configuration Example for SMF tags

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool
                  tags
                     nssai one
                     dnn two
                     serving-area three
                     end
```

Configuration Example of IPAM Tag with same SMF slice configuration name

```
ipam
 instance 1
  address-pool p1
   tags
    nssai slice1
    dnn   dnn1

  address-pool p2
   tags
    nssai slice1
    dnn   dnn2
```

Configuring IPv4 Address Range Threshold

IPAM keeps monitoring the pool usage threshold. Based on the configured threshold value, IPAM requests for next free address range or releases the address range.

To configure the IPv4 threshold, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            ipv4
               threshold
                  upper-threshold percentage
                  commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **ipv4**: Enter the IPv4 mode of the pool.
- **threshold**: Enter the threshold sub-mode.
- **upper-threshold** *percentage*: Specify the IPv4 upper threshold value in percentage.

The following is a sample configuration.

```
config
   ipam
      instance 1
         address-pool p1
                  ipv4
                     threshold
```

```
                        upper-threshold 80
                        end
```

Verifying the Threshold of a Pool

Use the **show ipam pool** command to view the summary of current threshold of each pool.

The following is an example output of the **show ipam pool** command.

```
show ipam pool
=================================================================
PoolName   Ipv4Utilization  Ipv6AddrUtilization  Ipv6PrefixUtilization
=================================================================
  p1               80%            80%               0%
  p2               75%            0%                70%
=================================================================
```

Configuring IPv6 Address Range Threshold

To configure the IPv6 address range threshold, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            ipv6
               address-ranges
                  threshold
                     upper-threshold percentage
                     commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **ipv6**: Enter the IPv6 mode of the pool.
- **address-ranges**: Enter the IPv6 address ranges sub-mode.
- **threshold**: Enter the threshold sub-mode.
- **upper-threshold** *percentage*: Specify the IPv6 upper threshold value in percentage.

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p2
                  ipv6
                     address-ranges
                        threshold
                           upper-threshold 75
                           end
```

Configuring IPv6 Prefix Range Threshold

To configure the IPv6 prefix range threshold, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            ipv6
               prefix-ranges
                  threshold
                     upper-threshold percentage
                     commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **ipv6**: Enter the IPv6 mode of the pool.
- **prefix-ranges**: Enter the IPv6 prefix ranges sub-mode.
- **threshold**: Enter the threshold sub-mode.
- **upper-threshold** *percentage*: Specify the IPv6 upper threshold value in percentage.

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p3
                  ipv6
                     prefix-ranges
                        threshold
                           upper-threshold 78
                           end
```

Configuring IPv4 Address Range Split

To configure the IPv4 address range split, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            ipv4
               split-size per-cache number_of_addresses
               split-size per-dp number_of_addresses
               commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **ipv4**: Enter the IPv4 mode of the pool.
- **split-size per-cache** *number_of_addresses*: Specify the number of IPv4 addresses per chunk for IPAM cache allocation. Specify in the power of 2. The IPAM server consumes this configuration.

  number_of_addresses must be an integer in the range of 2-262144.

- **split-size-per-dp** *number_of_addresses*: Specify the number of IPv4 addresses per chunk for data plane allocation. Specify in the power of 2. The IPAM cache consumes this configuration.

  number_of_addresses must be an integer in the range of 2-262144.

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p1
                  ipv4
                     split-size per-cache 1024
                     split-size per-dp 256
                     end
```

Configuring IPv6 Address and Prefix Address Range Split

To configure the IPv6 address and prefix address range split, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            ipv6
               address-ranges
                  split-size per-cache number_of_addresses
                  split-size per-dp number_of_addresses
                  exit
               prefix-ranges
                  split-size per-cache number_of_addresses
                  split-size per-dp number_of_addresses
                  commit
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the address pool. *pool_name* must be a string.
- **ipv6**: Enter the IPv6 mode of the pool.
- **address-ranges**: Enter the IPv6 address-ranges sub-mode.
- **split-size per-cache** *number_of_addresses*: Specify the number of IPv4 addresses per chunk for IPAM cache allocation. Specify in the power of 2. The IPAM server consumes this configuration.

  number_of_addresses must be an integer in the range of 2-262144.
- **split-size-per-dp** *number_of_addresses*: Specify the number of IPv4 addresses per chunk for data plane allocation. Specify in the power of 2. The IPAM cache consumes this configuration.

  number_of_addresses must be an integer in the range of 2-262144.
- **prefix-ranges**: Enter the IPv6 prefix ranges sub-mode.

The following is an example configuration.

```
config
   ipam
      instance 1
         address-pool p1
```

```
                  ipv6
                     address-ranges
                        split-size per-cache 4096
                        split-size per-dp 1024
                        exit
                     prefix-ranges
                        split-size per-cache 8192
                        split-size per-dp 2048
                        end
```

Configuring Global Threshold

To configure the global threshold, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         threshold
            ipv4-addr percentage
            ipv6-addr percentage
            ipv6-prefix percentage
            commit
```

NOTES:

- **threshold**: Enter the threshold sub-mode.
- **ipv4-addr** *percentage*: Specify the IPv4 threshold value in percentage.
- **ipv6-addr** *percentage*: Specify the IPv6 threshold value in percentage.
- **ipv6-prefix** *percentage*: Specify the IPv6 prefix threshold value in percentage.

The following is an example configuration.

```
config
   ipam
      instance 1
         threshold
                  ipv4-addr 80
                  ipv6-addr 75
                  ipv6-prefix 70
                  end
```

Verifying the Details of a Pool

This section describes how to verify the integration of IPAM in the SMF.

Use the **show ipam pool** *pool_name* command to view more details of a specific pool name.

The following is an example output of the **show ipam pool** *pool_name* command.

```
show ipam pool p1
--------------------------------------------------------
Ipv4Addr   [Total/Used/Threshold] = 7680 / 7680 / 80%
Ipv6Addr   [Total/Used/Threshold] = 0 / 0 / 0.00%
Ipv6Prefix [Total/Used/Threshold] = 512 / 512 / 80%
Instance ID = 1
--------------------------------------------------------
```

Configuring IPAM Source

To configure the IPAM source, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         source local
            source external ipam
               host ip_address
               port port_number
               vendor type
               commit
```

NOTES:

- **source local**: Enter the local data store as the pool source.
- **source external ipam** : Enter the external IPAM server as the pool source.
- **host** *ip_address* : Specify the host name of the external IPAM server.
- **port** *port_number* : Specify the port of the external IPAM server.
- **vendor** *type*: Specify the vendor type of the external IPAM server.

The following is an example configuration.

```
config
   ipam
      instance 1
         source external ipam
            host 209.165.200.225
            port 10000
            vendor cisco
            end
```

Verifying the IPAM Integration Configuration

This section describes how to verify the integration of IPAM in the SMF.

Verifying the Details of a Data Plane

Use the **show ipam dp** *data_plane_name* command to view details of a specific data plane (user plane).

The following is an example output of the **show ipam dp** *data_plane_name* command.

```
show ipam dp UPF-100
--------------------------------------------------------
Ipv4Addr   [Total/Used/Threshold] = 512 / 100 / 20%
Ipv6Addr   [Total/Used/Threshold] = 0 / 0 / 0.00%
Ipv6Prefix [Total/Used/Threshold] = 512 / 300 / 70%
Instance ID = 1
--------------------------------------------------------
```

Verifying the Threshold for Data Plane

Use the **show ipam dp** command to view the summary of the current threshold for each data plane (User Plane).

The following is an example output of the **show ipam dp** command.

```
show ipam dp
===============================================================
DpName     Ipv4Utilization  Ipv6AddrUtilization  Ipv6PrefixUtilization
===============================================================
UPF-100          20%             40%              70%
UPF-200          40%             20%              20%
===============================================================
```

Verifying the IPv4 Address Range Assigned to a Data Plane

Use the **show ipam dp** *data_plane_name* **ipv4-addr** command to view the IPv4 address ranges assigned to a data plane.

The following is an example output of the **show ipam dp** *data_plane_name* **ipv4-addr** command.

```
show ipam dp UPF-100 ipv4-addr
==========================================================================================
Flag  Indication: S(Static) O(Offline) R(For Remote Instance)
G:N/P Indication: G(Cluster InstId) N(Native NM InstId) P(Peer NM InstId)
==========================================================================================
StartAddress         EndAddress       AllocContext   Route            G:N/P   Utilization
   Flag
==========================================================================================
209.165.200.225  209.165.200.253  Pool-1        209.165.200.224/27 1:1/0     99.60%

209.165.201.1    209.165.201.30   Pool-2        209.165.201.0/27   1:1/0     99.60%
 R
==========================================================================================
```

Verifying the IPv6 Address Range Assigned to a Data Plane

Use the **show ipam dp** *data_plane_name* **ipv6-prefix** command to view the IPv6 address ranges assigned to a data plane.

The following is an example output of the **show ipam dp** *data_plane_name* **ipv6-prefix** command.

```
show ipam dp UPF-100 ipv6-prefix
==========================================================================================
Flag  Indication: S(Static) O(Offline) R(For Remote Instance)
G:N/P Indication: G(Cluster InstId) N(Native NM InstId) P(Peer NM InstId)
==========================================================================================
StartAddress                  EndAddress                AllocContext       Route
         G:N/P    Utilization   Flag
==========================================================================================
2001:DB80:8f20::            2001:fc20:8f20:ffff::    ims-ipv6-pool1(n6) 2001:fc20:8f20::/48
      1:1/0    99.60%
2001:fc20:8f21::            2001:fc20:8f21:ffff::    ims-ipv6-pool1(n6) 2001:fc20:8f21::/48
      1:0/1    99.80%
2001:fc20:8f22::            2001:fc20:8f22:ffff::    ims-ipv6-pool1(n6) 2001:fc20:8f22::/48
      1:0/1    0.00%         R
2001:fc20:8f23::            2001:fc20:8f23:ffff::    ims-ipv6-pool1(n6) 2001:fc20:8f23::/48
      1:1/0    0.00%         R
2001:fc20:8f49::            2001:fc20:8f49:ffff::    ims-ipv6-pool1(n6) 2001:fc20:8f49::/48
      1:1/0    34.42%
2001:fc20:8f4f::            2001:fc20:8f4f:ffff::    ims-ipv6-pool1(n6) 2001:fc20:8f4f::/48
      1:0/1    33.58%
==========================================================================================
```

Configuring IP Pool Selection Method

Use the following configuration to configure an IP pool selection method.

```
config
nssai name nssai_name
   dnn dnn
   pool-selection [ pool_selection_method  ]
   sdt sdt_value
   sst sst_value
   tai-group-list tai_group_list
   end
```

NOTES:

- **pool-selection [** *pool_selection_method* **]**: Configure the IP pool selection method as DNN or NSSAI. The default value of **pool-selection** is *dnn*. If you configure **pool-selection [** *nssai* **]** for a slice, then in IPAM configuration for all the DNN for that UPF, "slice1+dnn" is to be configured.

Note The slice-based pool selection is not supported.

Configuration Example

The following is an example configuration of the IP pool selection method.

```
nssai name slice1
pool-selection [nssai]
exit

nssai name slice2
pool-selection [nssai dnn]
exit

nssai name slice3
exit
```

Note If no pool selection method is configured, then the default value of **pool-selection [** *dnn* **]** is used.

Configuring UPF Group Profile for IP Pool Selection

To configure the UPF group profile for IP pool selection, use the following sample configuration.

Note This configuration is required to support the slice-based IP pool.

```
config
  profile network-element upf upf_name
    upf-group-profile upf_group_profile_name
    dnn-list dnn_list_value
    end
```

NOTES:

- **profile network-element upf** *upf_name*: Specify a profile name for the UPF.
- **upf-group-profile** *upf_group_profile_name*: Specify the name of the UPF group configuration. The *upf_group_profile_name* value must be a string.
- **dnn-list** *dnn_list_value*: Specify the list of DNNs that the UPF node supports. The *dnn_list_value* value must be a string with a range of DNN list values.

Configuration Example

The following is an example configuration.

```
profile network-element upf upf1
   upf-group-profile group1
   dnn-list [dnn1, dnn2]
```

Configuring Slice Group List for IP Pool Selection

To configure the slice group list for IP pool selection, use the following sample configuration.

Note This configuration is required to support the slice-based IP pool.

```
config
   profile upf-group upf_group_profile_name
      slice-group-list slice_group_list_name
      end
```

NOTES:

- **profile upf-group** *upf_group_profile_name*: Specify the UPF group name that must be associated to the specified UPF network configuration. The *upf_group_profile_name* value must be an alphanumeric string.
- **slice-group-list** *slice_group_list_name*: Specify the list of slice groups that the UPF node supports. The *slice_group_list_name* value must be a string with a range of slice groups.

Configuration Example

The following is an example configuration.

```
profile upf-group group1
 slice-group-list [ slice1 ]
exit
```

Note Based on the NSSAI configuration of the IP pool selection, the SMF sends the "slice + dnn" as tag to IPAM.

Example

```
profile network-element upf upf1
upf-group-profile group1
 dnn-list [dnn1, dnn2, dnn3]

profile upf-group group1
 slice-group-list [slice1, slice2 slice3]
exit
```

Static IP Support

Feature Description

IPAM is the core component of the subscriber management system. Traditional IPAM functionalities prove insufficient in the Cloud Native network deployments. Hence, IPAM requires more functionalities to work with the Cloud Native subscriber management system.

The Static IP Support feature enables the support of static IP on the SMF using IPAM. This feature supports the following functionalities:

- Static pool configuration—dynamic addition and deletion of static IP pool or static IP address range when the system is running.
- Splits static address ranges into smaller chunks and associates them with the configured UPFs
- Enables program routes according to static address range reservation during UPF association
- Enables secondary authentication under the DNN profile
- Selects UPF based on reserved address range and Framed-IP received from the Authentication response
- Handle UPF addition, deletion, and Sx path failure
- Add a DNN to an existing UPF

Calls with Static IP Address

The SMF supports calls with static IP address and validates if the IP address belongs to the static pool.

The SMF supports Create Session Request with static IP address and also handles Create Session Request received with PAA. The SMF validates if the requested IP address is configured under static pool and assigns the same IP address for the session. If the IP address is not configured under static pool, then SMF rejects the session.

Important In Release 2021.02, the SMF does not support fallback to dynamic IP allocation.

The following behavior is applicable only to sessions with static IP address.

- If the SMF receives static IP in Subscription Response from UDM during the 5G Session Create procedure, it assigns the same IP address to the UE session if the IP is configured under static pool. If the IP address is not configured under static pool, then SMF rejects the session.
- If the RADIUS interface is enabled and if the RADIUS server returns the static IP address, then SMF ignores the IP address received in Create Session Request or Subscription Response.

How it Works

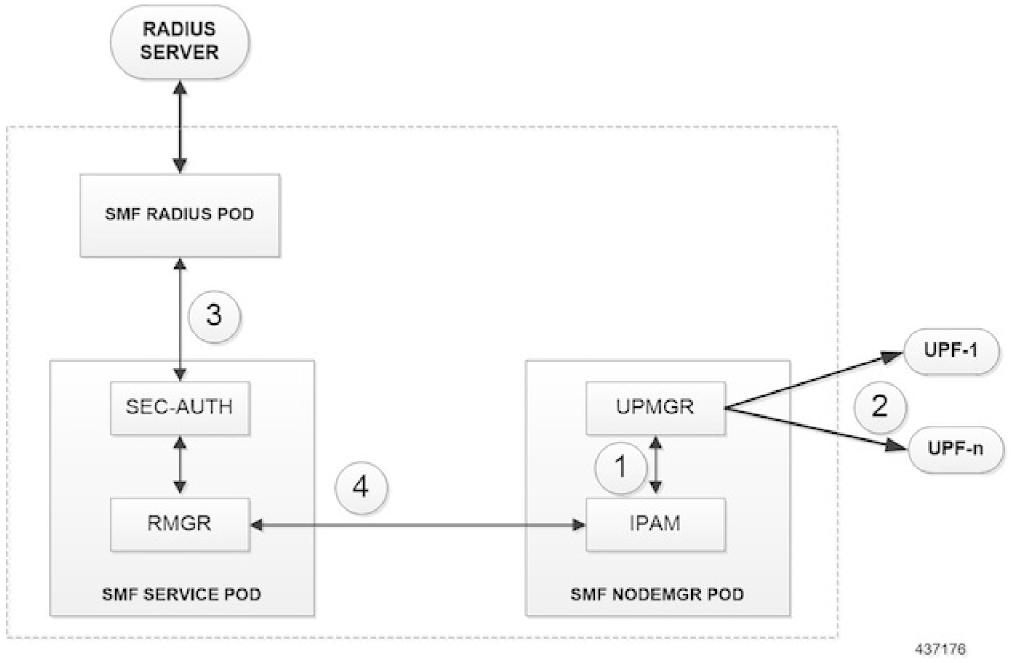
This section provides a brief of how the Static IP Support feature works.

The SMF receives a framed IP address of the subscriber from external AAA servers, such as RADIUS. While IPAM is not involved in individual IP address management in this scenario, it still handles the route management and UPF management for static address ranges.

IPAM splits the 'static' address ranges equally according to number of UPFs present in the SMF configuration. Unlike dynamic IP, IPAM splits all static IP address ranges and assigns them for all configured UPFs. IPAM involves and selects an UPF when the external AAA server returns the framed IP of the subscriber. IPAM looks for the route which includes this static IP and then selects the UPF where the route is already configured.

The following figure shows how the static IP address is assigned to the configured UPFs.

Figure 2: Static IP Address Management Procedure



1. IPAM splits the static ranges into equal number of address ranges based on the number of configured UPFs.
2. The UPMGR programs the corresponding static routes on the associated UPFs.
3. Subscribers get static IP from RADIUS server authorization response.
4. SMF service selects the right UPF based on address ranges and UPF map allocation from the Node Manager.

Address Range Split

Splitting a given address range into smaller address ranges is a key functionality of the IPAM server and IPAM cache. The following guidelines determine address range split:

- Size of a split address range depends on the configured value or the default value as per the Authority and Format Identifier (AFI) type.
- Size of a split address range must be a power of 2 or at least to the closest of it. That is, it should be able to represent the split range in subnet/mask notation such that a route can be added in the data plane (user plane) if required.
- Configured or default address range size must be at the power of 2.

The address range must be split into smaller ranges immediately on configuration or initial start-up. This helps in better sorting of address ranges based on size and faster allocation during actual address range allocation requests. The address range exchange between modules is always in the mentioned size.

Table 4: Examples of IPv4 Address Range Split

Address Range	Split Size (number of addresses per range)	Split Ranges (* Odd sized ranges)	Route Notation
209.165.200.225 - 209.165.200.254	128	[1] 209.165.200.225 – 209.165.200.254 [2] 209.165.202.129 – 209.165.202.158	[1] 209.165.200.224/27 [2] 209.165.202.128/27
209.165.201.1 – 209.165.201.30	256	[1] 209.165.200.224 – 209.165.200.254 [2] 209.165.201.0 – 209.165.201.30 [3] 209.165.202.128 – 209.165.202.158 ... [n] 209.165.200.225 – 209.165.200.253	[1] 209.165.201.1/27 [2] 209.165.200.224/27 [3] 209.165.202.128/27 ... [n] 209.165.201.0/27
209.165.200.229 – 209.165.200.253	256	[1] 209.165.201.1 – 209.165.201.30 * [2] 209.165.202.129 – 209.165.202.158 [3] 209.165.200.225 – 209.165.200.253 *	[1] 209.165.201.0/27 [2] 209.165.200.224/27 [3] 209.165.202.128/27

Table 5: Examples of IPv6 Address Range Split

Address Range	Split Size (number of addresses per range)	Split Ranges (* Odd sized ranges)	Route Notation

1:: - 1::1000	1024	[1] 1:: – 1::3FF [2] 1::400 – 1::7FF [3] 1::800 – 1::BFF [4] 1::C00 – 1::FFF	[1] 1::/118 [2] 1::400/118 [3] 1::800/118 [4] 1::C00/118
1::3 - 1::1DEF	1024	[1] 1::3 – 1::3FF * [2] 1::400 – 1::7FF [3] 1::800 – 1::BFF … [n] 1::1C00 – 1::1DEF *	[1] 1::/118 [2] 1::400/118 [3] 1::800/118 … [n] 1::1C00/118

Examples of IPv6 Address Range Split

Prefix split needs two length fields for performing the split.

- Network length
- Host length

Prefixes are split between these two length fields and a new route is calculated.

Example 1: network-length = 48, prefix-length = 64

Total (64-48) = 16 bits (that is, 65536 prefixes are available for the split)

Example 2: network-length = 32, prefix-length = 56

Total (56-32) = 24 bits (that is, 16 million prefixes available for the split)

Note For SMF, the host-length is hard-coded as ‘64’. Only network-length can be configured using the CLI.

Table 6: Examples of IPv6 Address Range Split

Prefix Range	Split Size (number of addresses per range)	Split Ranges (* Odd sized ranges)	Route Notation

1:2:3::
Nw-len = 48
Host-len = 64
	8192	[1]1:2:3:: ... 1:2:3:1fff
		[2]1:2:3:2000:: ... 1:2:3:2fff::
		[3]1:2:3:3000:: ... 1:2:3:3fff::
		...
			[1]1:2:3::/51
			[2]1:2:3:2000/51
			[3]1:2:3:3000/51
			...

Call Flows

This section describes the static IP call flow.

The following figure shows the static IP address allocation call flow.

Figure 3: Static IP Call Flow

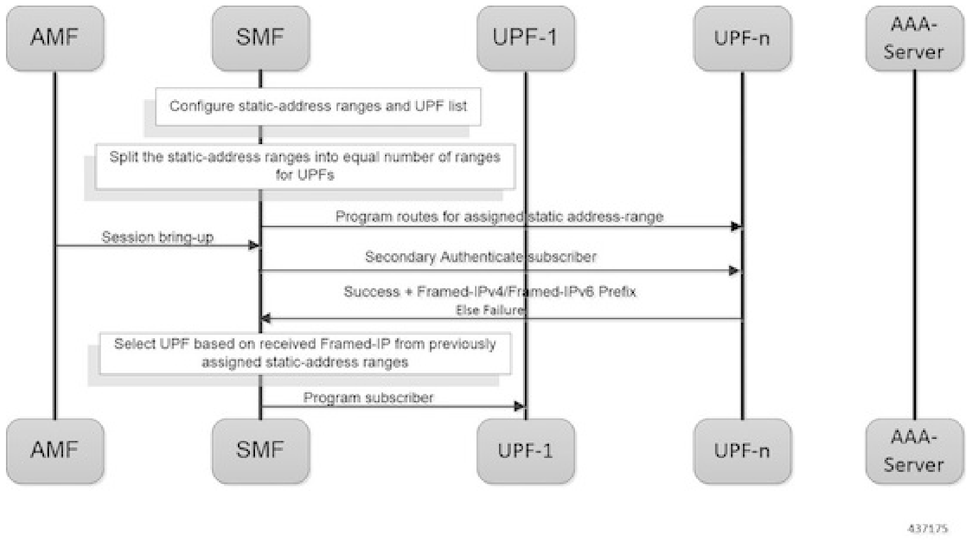

AMF
SMF
UPF-1
UPF-n
AAA-Server
Configure static-address ranges and UPF list
Split the static-address ranges into equal number of ranges for UPFs
Program routes for assigned static address-range
Session bring-up
Secondary Authenticate subscriber
Success + Framed-IPv4/Framed-IPv6 Prefix
Else Failure
Select UPF based on received Framed-IP from previously assigned static-address ranges
Program subscriber
437175

Table 7: Static IP Call Flow Description

Step	Description
1	Configure the static address ranges and UPF list.
2	Split the static address ranges into equal number of ranges for UPFs.
3	Enable program routes for the assigned static address range.
4	Bring up the session.
5	Enable secondary authentication under the DNN profile.
6	The SMF sends the Authentication Request to the RADIUS server. The RADIUS server sends an Authentication Response with the static IP of the subscriber. The SMF selects the UPF based on the static IP and continues with the programming.
7	Complete the subscriber programming.

Adding a DNN

This section describes the sequence of operations for adding a DNN.

1. Create a static IP pool in IPAM with the corresponding DNN.
2. Add a DNN Profile.
3. If applicable, add the UPFs.
4. Associate the IP address ranges of the DNN to the available UPFs.

Note The route is added as part of RegisterUpf requests during explicit Sx association.

Adding a Static IP Address Range

This section describes the sequence of operations for adding a static IP address range in SMF.

- If new static IP address range is added to a single stack IP pool, the IP address ranges are split according to the configuration and associated with available UPFs in load sharing manner.
    1. Similar to initial association, intermediate association is also done based on the number of IP addresses against the number of configured UPFs.
    2. If UPF is already registered with IPAM:
        - Route addition is triggered, or else
        - No immediate action is taken
- If a dual stack pool is configured, all IP address ranges, both IPv4 and IPv6 are associated with the UPF, which is the least loaded.
    - If UPF is already registered with IPAM:
        - Route addition is triggered, or else
        - No immediate action is taken

Adding a Static IP Pool

This section describes the sequence of operations for adding a static IP pool in SMF.

- If a single stack IP pool is configured, the IP address ranges are split according to the configuration and associated with available UPFs in load-sharing manner.
    1. Similar to initial association, intermediate association is also done based on the number of IP addresses against the number of configured UPFs.
    2. If UPF is already registered with IPAM:
        - Route addition is triggered, or else
        - No immediate action is taken
- If a dual stack pool is configured, all IP address ranges, both IPv4 and IPv6 are associated with the UPF, which is the least loaded.
    - If UPF is already registered with IPAM:
        - Route addition is triggered, or else
        - No immediate action is taken

Adding the UPF

This section describes the sequence of operations for adding the UPF.

1. When a UPF is added, NodeMgr sends the list of IPs to IPAM.
2. When new static IP pool or static IP address range is configured, this feature enables route association for UPFs based on load balancing model.

Note The same procedure is applicable when a new or existing DNN is added to a new or existing UPF respectively.

3. To redistribute existing static IP pools or ranges to the new UPF, use the following procedure:
    - Mark a pool/range offline
    - Clear the subscribers
    - Delete IP pool or range
    - Add the IP pool or range again.

      This step allocates the chunks to the new UPF.

Deleting the UPF

This section describes the sequence of operations for deleting an existing UPF.

1. To delete an existing UPF, it is first marked "offline".

   Run the appropriate CLI commands to manually clear the sessions.
2. The NodeMgr notifies IPAM about the UPF removal.
3. IPAM moves the static address ranges from all DNNs of the removed UPF to other available UPFs.
4. The Nodemgr initiates ReleaseUpf to IPAM. IPAM releases dynamic address ranges to the free list.
5. The Nodemgr sends an N4 Association Release message to UPF and to clean up UPF from the cache.

Note If the UPF is not marked offline and a manual clean-up is not performed before its removal, the system behavior might be erratic.

Deleting a Static IP Address Range

This section describes the sequence of operations for deleting a static IP address range in SMF.

1. To delete an IP address range from a static IP pool, it is first marked "offline".
2. Reject new calls, which have the IP address assigned from the offline IP address range.
3. Remove the existing subscribers. To remove the existing subscribers, run the following CLI commands:

```
clear subscriber ipv4-range { pool_name | start_of_range }
clear subscriber ipv6-range { pool_name | start_of_range }
```

4. Remove the static IP address range configuration and trigger route deletion to the registered UPFs.

Deleting a Static IP Pool

This section describes the sequence of operations for deleting a static IP pool in SMF.

1. To delete a static IP pool, it is first marked "offline".
2. Reject new calls, which have the IP address assigned from the offline IP pool.
3. Remove the existing subscribers. To remove the existing subscribers, run the following CLI commands:

   ```
   clear subscriber ipv4-pool pool_name
   clear subscriber ipv6-pool pool_name
   ```

4. After all the subscribers are deleted, remove the IP pool configuration and trigger route deletion to the registered UPFs.

Removing Sx Association with an Offline UPF

This section describes the sequence of operations for removing association with an offline UPF.

1. Set UPF as offline in **profile-network-element-upf** configuration.

   SMF stops selecting and associating dynamic IPs to the specific UPF for new sessions.

2. NodeMgr receives configuration change notification about an offline UPF.

   SMF stops selecting and associating static IPs to the specific UPF for new sessions or associations.

3. NodeMgr acknowledges the heart-beat messages for an already associated UPF.
4. NodeMgr acknowledges the N4 association update from the UPF with release indication.

   This step does not impact the static and dynamic chunk allocations for IPAM.

   The IPAM module is unaware of the offline status for the UPF. It might include the offline UPF to add new IP pool or address ranges.

Sx Path Failure on UPF

This section describes the sequence of operations for Sx path failure on UPF.

1. The NodeMgr initiates the **clear subscriber** command.
2. The NodeMgr sends UnRegisterUpf to IPAM.
3. IPAM releases any dynamic IP address ranges and moves it to free range list.
4. IPAM retains any static IP address ranges for the UPF. Sx path failure does not impact static IP address mappings.

Limitations

The Static IP Support feature has the following limitations:

- Change of a pool from dynamic to static, and from static to dynamic is not supported when the system is in running mode.
- Addition or removal of UPF is not supported when the system is in running mode.
- The address range split must be optimal based on the number of UPFs and number of addresses in the ranges.

  For example:

  If there are 2 UPFs and 1024 addresses specified in the range, then specify the per-dp-split-size as 512.

  If there are 3 UPFs and 1024 addresses, then specify the per-dp-split-size as 256.
- When the system is running, the DNN cannot be removed from a UPF.
- Changing dual-stack IPAM pool to single-stack or changing single-stack IPAM pool to dual-stack is not supported.

Configuring Static IP Support

To configure the Static IP Support feature, use the following sample configuration:

```
config
  ipam
    instance gr_instance_id
      address-pool pool_name
        static
        end
```

NOTES:

- **ipam**: Enter the IPAM configuration mode.
- **address-pool** *pool_name*: Specify the name of the address pool to enter the pool configuration. *pool_name* must be a string.
- **static**: Enable the static IP mode.

Statistics Support

The smf_service_resource_mgmt_stats and smf_service_node_mgr_stats provide details on static IP allocation type information.

The ip_req_type attribute in these statistics supports the following labels:

- ip-static-subscription—Static IP allocation information based on subscription
- ip-static-radius—Static IP allocation information based on RADIUS

Dual-stack Static IP Support Through IPAM

Feature Description

The SMF supports dual-stack static IP using IPAM. For dual-stack sessions, the AAA server sends both the IPv4 and IPv6 address prefixes as part of the Access-Accept message. In the SMF-IPAM configuration, both the IPv4 and IPv6 address prefixes are added in the same pool. The IPAM assigns both the IPv4 and IPv6 routes to a single UPF.

During the UPF selection, the Node Manager application uses the UPF for both the IPv4 and IPv6 addresses from the IPAM to handle them accordingly.

How it Works

The SMF supports dual-stack static IP through IPAM in the following ways:

- Pool to UPF mapping—Based on the number of UPFs available, the IPv4 address ranges and IPv6 prefix ranges are split into smaller chunks. Then, the pair (chunk) is configured into the same IPAM pool.

  IPAM assigns all the addresses and prefixes that are configured in one dual-stack pool to a UPF in the manner they are received. The AAA server returns the dual-stack addresses from the same pair. From these addresses, SMF selects one UPF for dual-stack programming.

  The load-balancing of number of addresses and prefixes are managed. IPAM performs only the dual-stack static-pool to UPF mapping.

- Address range no-split configuration—IPAM uses the "no-split" configuration to prevent the splitting of address ranges into smaller chunks. This configuration helps to prevent having multiple routes programming for a specific range.

The following table lists the errors or exceptions and how to handle them:

Table 8: Error and Exception Handling

Error or Exception	Exception Handling
IPv4 UPF and IPv6 UPF are configured incorrectly	**1.** Select an active UPF. In case both the UPFs are active, select the UPF with the IPv4 address. **2.** Reset the IP information of the other stack and update the PDU session type accordingly.
IPv4 address is invalid or null	Select the UPF with IPv4 address and update the PDU session type accordingly.
IPv6 prefix is invalid or null	Select the UPF with IPv6 address and update the PDU session type accordingly.
IPv4 address and IPv6 prefix are invalid	Reject both the IPv4 address and IPv6 prefix.

Limitations

The Dual-stack Static IP Support feature has the following limitation:

- The change in ‘no-split' configuration is not supported when the system is in running mode.

Configuring Dual-stack Static IP

This section describes how to configure the dual-stack static IP support using IPAM.

Configuring IPAM No-Split

To configure the IPAM no-split, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
         ipv4
            split-size no-split
            exit
         ipv6 prefix_ranges
            split-size no-split
            exit
         exit
```

NOTES:

- **split-size no-split**: Prevent the IPv4 address ranges or IPv6 prefix ranges from splitting into smaller chunks.

IPAM Offline Mode Support

Feature Description

The SMF supports the addition of a dynamic pool, IPv4, or IPv6 address-range to a dynamic pool by default. The new chunks are added to the respective tags, such as DNN, and are assigned from the same pool.

To delete a dynamic pool or an IPv4 or IPv6 address range from a dynamic pool:

1. Configure the pool or address range as offline. The IPAM then stops assigning addresses from the respective pool or address range.
2. Use the following **clear subscriber** CLI commands to delete the subscribers based on respective pool or address range that are configured to offline mode:
   - **clear subscriber ipv4-pool** *pool_name*
   - **clear subscriber ipv4-range** *pool_name/start_of_range*
   - **clear subscriber ipv6-pool** *pool_name*

- **clear subscriber ipv6-range** *pool_name/start_of_range*

3. Use the following **cdl show** CLI commands and wait until all the subscribers are deleted:
   - **cdl show sessions count summary filter { key ipv4-pool:** *pool_name* **condition match }**
   - **cdl show sessions count summary filter { key ipv4-range:** *pool_name/start_of_range* **condition match }**
   - **cdl show sessions count summary filter { key ipv6-pool:** *pool_name* **condition match }**
   - **cdl show sessions count summary filter { key ipv6-range:** *pool_name/start_of_range* **condition match }**
   - **cdl show sessions count summary slice-name** *slice_name*
4. After all the subscribers are deleted, delete the pool or address range from the IPAM configuration.

Configuring the IPAM Offline Mode

This section describes how to configure the IPAM offline feature for pool, IPv4 address range, and IPv6 prefix ranges.

Configuring Pool to Offline Mode

To configure the entire pool to offline mode, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            offline
            end
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the pool to enter the pool configuration. *pool_name* must be a string.
- **offline**: Configure the pool to offline mode.

Setting IPv4 Address Range to Offline Mode

To configure the IPv4 address range to offline mode, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            vrf-name vrf_name
            ipv4
               address-range start_ipv4_address end_ipv4_address offline
               end
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the pool to enter the pool configuration. *pool_name* must be a string.
- **ipv4**: Enter the IPv4 mode.
- **address-range** *start_ipv4_address end_ipv4_address* **offline**: Specify the IP addresses for the start and end IPv4 address range.
  - **offline**: Set the selected address range to offline mode.

Setting IPv6 Prefix Ranges to Offline Mode

To configure IPv6 prefix range to offline mode, use the following sample configuration:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
         vrf-name vrf_name
            ipv6
               prefix-ranges
                  prefix-range prefix_valuelength prefix_lengthoffline
                  end
```

NOTES:

- **address-pool** *pool_name*: Specify the name of the pool to enter the pool configuration. *pool_name* must be a string.
- **ipv6**: Enter the IPv6 mode.
- **prefix-ranges**: Enter the prefix ranges mode.
- **prefix-range** *prefix_value* **length** *prefix_length* **offline**: Specify the prefix range and prefix length of the IPv6 prefix range.
  - **offline**: Set the selected address range to offline mode.

IPAM Redundancy Support Per UPF

Feature Description

The SMF supports IPAM redundancy and load balancing for each UPF. The IPAM running in the Node Manager microservice has two IPAM instances that are associated to each UPF. When one IPAM instance is inactive, the other IPAM instance manages the address allocation requests for the UPF.

How it Works

This section provides a brief of how the IPAM redundancy support per UPF feature works.

- Peer Selection—The Node Manager peer is selected during the UPF association.

- UPF Registration with Peer IPAM—IPAM is notified with the instance ID of the peer for the UPF during the registration of the UPF call. IPAM allocates routers from the local data for the specific DNN and checks if the peer IPAM instance is in active or inactive state.

  If the peer IPAM instance is active, a REST call is sent to it to register to the same UPF in the local instance and to receive the routes as response.

  If the peer IPAM instance is inactive, the local instance takes over the IPAM context of the remote instance. Then, the local instance registers to the UPF, receives the routes, and keeps the data back in the cache pod. After the peer instance is active, it restores the same data from the cache pod.

  Routes from both the instances are sent to UPF for load-balanced address allocations from both the instances.

- Address Allocation in Load-Balanced Model—As one UPF is registered to two IPAM servers, SMF sends the address allocation requests to any peer that is load-balanced. Respective IPAM instances assign new addresses from their local address bitmap. If one peer instance is inactive, the other peer instance handles all the requests.

- Address Release Request Handling—In IPAM, the Address Release request is sent to the instance that had allocated the IP the first time. If that peer is inactive, the Address Release request is sent to the peer IPAM.

  The IPAM instance that receives the address releases for remote instances, keeps buffering these instances locally and updates the cache pod periodically. After the remote peers are active, they handle the buffered address release requests.

- Release of the UPF—When a peer IPAM is active during the release of a UPF, a REST call is sent to clear the data. If the peer IPAM is inactive, the existing IPAM instance takes over the operational data of the remote IPAM, clears the UPF information, and updates the cache pod.

IPAM Quarantine Timer

Feature Description

The IPAM Quarantine Timer Support feature supports the IPAM quarantine timer for the IP pool address. This feature keeps the released IP address busy until the quarantine timer expires to prevent the reuse of that IP address. Each IP pool must be configured with a timer value. This value determines the duration of a recently released address to be in the quarantine state before it is available for allocation. After the timer expires, the IP address is available in the list of free addresses for allocation by the subscriber. A released IP address with no address quarantine timer is considered to be in use for allocation. If a subscriber attempts to reconnect when the address quarantine timer is armed even if it is the same subscriber ID, the subscriber does not receive the same IP address.

Configuring IPAM Quarantine Timer

This section describes how to configure the IPAM quarantine timer.

Configuring IPAM Quarantine Timer

This section describes how to configure the IPAM quarantine timer.

```
config
   ipam instanceinstance_id
      address-pool pool_name
         address-quarantine-timer quarantine_timer_value



         end
```

NOTES:

- **address-pool** *pool_name*—Specifies the name of the pool to enter the pool configuration. *pool_name* must be the name of the address pool.
- **address-quarantine-timer** *quarantine_timer_value*—Specifies the value of the quarantine timer in seconds. *quarantine_timer_value* must be in the range of 4-3600 seconds. The default value is 4.

IP Address Validation with CDL Configuration

This section describes how to validate IP Address with CDL configurtation.

System Diagnostics IP Validation

This section describes how to enable/disable System Diagnostics IP Validation.

```
config
   system-diagnostics ip-validation enable ignore-mismatch-responses
   exit
```

NOTES:

system-diagnostics ip-validation ignore-mismatch-responses — Ignores any CDL inconsistencies during address validation.

IP validation ignore mismatch responses is meant for avoiding duplicate IPs. If this feature is enabled, SMF Nodemgr checks if the current IP is already used by any other records in CDL. If no records are found, then IP address is assigned to the UE. If CDL record is found, then a new IP is assigned to the UE.

Important Enabling validation ignore mismatch responses may have certain performance impact.

Statistics

nodemgr_diag_ip_verify

Description: Display Nodemgr to CDL IP-Validation query related statistics

Metrics-Type: Counter

Query: sum(nodemgr_diag_ip_verify{namespace="$namespace"}) by (status)

Labels:

Label: status

Value: success | duplicate_record_found | cdl_ipc_failure | ipv4_alloc_failed | ipv6_alloc_failed | unknown

- success Record not found in CDL
- duplicate_record_found Duplicate record found in CDL
- cdl_ipc_failure Search IPC request to CDL failed
- ipv4_alloc_failed IPV4 address-request failed, unable to get free-IP, twice
- ipv6_alloc_failed IPV6 prefix-request failed, unable to get free-IP, twice
- unknown IPC request to CDL failed twice, give-up and return the IP to smf-service

IPAM_Quarantine_Statistics

Description: Display IPAM Quarantine IP Batch related statistics

Metrics-Type: Counter

Query: sum(IPAM_Quarantine_Statistics{namespace="$namespace"}) by (addressType, type)

Labels:

Label: pool

Value: <name-of-pool>

Label: upf

Value: <name-of-upf>

Label: addressType

Value: IPv4 | IPv6PD

Label: type

Value: start_batch_qsize | end_batch_qsize | pop_count_qtime | pop_count_qsize | avg_qtime_secs

- start_batch_qsize - Number of IPs in QT-queue at the start of batch processing
- end_batch_qsize - Number of IPs in QT-queue at end of batch processing
- pop_count_qsize - Number of IPs removed from QT-queue due to qsize limit
- pop_count_qtime - Number of IPs removed from QT-queue due to qtime limit
- avg_qtime_secs - Average time-in-seconds the IPs were in QT-queue before removing

IPAM Data Reconciliation

Feature Description

The SMF supports the IPAM data reconciliation feature to reconcile IPAM data with the CDL records. This feature is triggered through the EXEC mode CLI. IPAM reconciliation is triggered at instance level, pool level, and chunk level.

Triggering IPAM Reconciliation

This sections describes how to trigger the IPAM reconciliation on instance level, pool level, and chunk level.

Triggering IPAM Reconciliation at Instance Level

To trigger IPAM reconciliation at an instance level, use the following CLI command:

```
reconcile ipam instance instance_id
```

NOTES:

- **reconcile ipam instance** *instance_id*: Trigger IPAM reconciliation for a specific GR instance ID.

Triggering IPAM Reconciliation at Pool Level

To trigger IPAM reconciliation at a pool level, use the following CLI command:

```
reconcile ipam instance instance_id pool-name pool_name
```

NOTES:

- **pool-name** *pool_name* : Trigger IPAM reconciliation for a specific address pool.

Triggering IPAM Reconciliation at Chunk Level

To trigger IPAM reconciliation at a chunk level, use the following CLI command:

```
reconcile ipam instance instance_id pool-name pool_name chunk-start-ip
chunk_start_ip_address
```

NOTES:

- **chunk-start-ip** *chunk_start_ip_address* : Specify the IPAM reconciliation chunk starting IP address.

IPAM Periodic Reconciliation

The IPAM reconciliation can be triggered manually through the CLI. It also gets triggered on the nodemgr startup or after the GR role-switchover.

This process needs upgradation to a system or a software-dependent procedure. It requires a support to provide the IPAM reconciliation configuration, to run a time-driven activity, periodically in the background.

You can schedule to run a daily IPAM reconciliation activity, using the CLI configuration framework for the following:

- Specific GR instances ID
- Specific address pool under a GR instance ID

The IPAM reconciliation process performs multiple queries to the CDL and fetches subscriber sessions to sync or to update the IPAM cache-data data.

Limitations

This feature has the following limitations:

- Schedule the periodic IPAM reconciliation, during the time when the system has less traffic load management.
- Scheduling multiple reconciliations at the same time of the day isn't supported.
- The nodemgr can trigger only one instance of the reconciliation process at a time.
- Multiple reconciliation schedules can be set across pools, by ensuring at least with a gap of five minutes between two triggers.
- The IPAM reconciliation is supported only for non-static pools.

Feature Configuration

The updated IPAM configuration CLI framework supports the following:

- Scheduling of the reconciliation for GR Instance ID
- Specific address pool support

To configure this feature, use the following configuration:

```
config
   ipam
      instance <gr_instance_id>
         reconcile-schedule
            tod-hour <time_of_day_hour_value>
            tod-minute <time_of_day_minute_value>
…
      address-pool <pool_name>
         reconcile-schedule
            tod-hour <time_of_day_hour_value>
            tod-minute <time_of_day_minute_value>
            end
```

NOTES:

- **ipam**—Enter the IPAM configuration.
- **instance** <*gr_instance_id*>—Specify the IPAM reconciliation for a specific GR instance ID.
- **address-pool** <*pool_name*>—Specify the name of the pool to enter the pool configuration. The <*pool_name*> must be the name of the address pool.
- **reconcile-schedule**—Specify the required schedule for reconciliation. You can configure the time-of-day value in hours and minutes, to set the time for triggering the daily reconciliation at that specified time.
- **tod-hour** <*time_of_day_hour_value*>—Specify your required time of the day in hours. You can configure the specified hour in a 24-hour format 0–23.
- **tod-minute** <*time_of_day_minute_value*>—Specify your required time of the day in minutes. You can configure the specified minute in a 60 minute format 0–59.

Configuration Example

The following example configuration allows the IPAM to set a daily schedule to trigger reconciliation for gr-instance-id 1 at midnight 00:00.

```
config
 ipam
  instance 1
   reconcile-schedule
    tod-hour   0
    tod-minute 0
   exit
```

The following example configuration allows the IPAM to set a daily schedule to trigger reconciliation for testPool1 for gr-instance-id 1 at 10:30 p.m. daily.

```
config
 ipam
  instance 1
   address-pool testPool1
    reconcile-schedule
     tod-hour   22
     tod-minute 30
    exit
    ipv4
     split-size
      per-cache 8192
      per-dp    1024
     exit
     address-range 209.165.200.225 209.165.200.254
    exit
   exit
  exit
```

Configuring IPAM Quarantine Qsize

This section describes how to configure the IPAM quarantine queue size support feature.

Configuring IPAM Quarantine Queue Size

This section describes how to configure the IPAM quarantine timer.

```
config
   ipam instanceinstance_id
      address-pool pool_name
         address-quarantine-qsize quarantine_queue_size
         exit
        exit
```

NOTES:

- **ipam**—Enter the IPAM configuration.
- **address-pool** *pool_name*—Specifies the name of the pool to enter the pool configuration. *pool_name* must be the name of the address pool.
- **address-quarantine-qsize** *quarantine_queue_size*—Specifies the value of the quarantine queue size. The default value is 0.

  During QT processing, excess IP addresses in quarantine-queue are released to Free-list irrespective of quarantine-timer expiry by force.

Overlapping IP Address Pools

Feature Description

The Overlapping IP Address Pools feature improves flexibility in assigning IP addresses dynamically. This feature allows you to configure overlapping IP address pool groups to create different address spaces and concurrently use the same IP addresses in different address spaces.

You can configure overlapping IP address range across different pools with unique DNN and VRF type.

Configuring Overlapping IP Address Pools

Use the following example configuration to configure overlapping static IP address pools.

```
config
ipam instance instance_id 1
source local
address-pool pool1
  static
  vrf-name vrf1@ISP
  tags
   dnn dnn1
  exit
  ipv4
   split-size
    per-cache 256
    per-dp    256
   exit
   address-range 209.165.200.225 209.165.200.254
  exit
exit
address-pool pool2
  static
  vrf-name vrf2@ISP
  tags
   dnn dnn2
  exit
  ipv4
   split-size
    per-cache 256
    per-dp    256
   exit
   address-range 209.165.200.225 209.165.200.254
  exit
exit
exit
```

The following is an example configuration for overlapping IP address pools.

```
config
ipam instance instance_id 1
source local
address-pool pool1
  vrf-name vrf1@ISP1
  tags
   dnn dnn1
  exit
  ipv4
```

```
   split-size
    per-cache 256
    per-dp    256
   exit
   address-range 209.165.200.225 209.165.200.254
  exit
exit
address-pool pool2
  vrf-name vrf2@ISP2
  tags
   dnn dnn2
  exit
  ipv4
   split-size
    per-cache 256
    per-dp    256
   exit
   address-range 209.165.200.225 209.165.200.254
  exit
exit
exit
```

Auto-Reclamation of Under-Utilized IP Chunks

Feature Description

SMF supports auto-reclamation of under-utilized IP chunks that are allocated to various UPF nodes. By configuring the Utilization Threshold and Inactivity Threshold, SMF can trigger the auto-reclamation process periodically or on an instant basis.

The under-utilization and inactivity of IP chunks are detected based on the inactivity timer configuration. When inactivity and under-utlization are detected, SMF triggers IP chunk reclamation by clearing all the subscribers using the IP chunk. SMF also triggers route deletion to UPF.

Limitations

Following are the known limitations of auto-reclamation of under-utilized IP chunks in SMF:

- During reclamation of under-utilized IP chunk, all released IPs must undergo configured address pool level Quarantine Time (QT time).
- If an IP from the reclaimed IP chunk is associated with an active subscriber session, SMF tears down the subscriber session and is expected to see session discontinuity.
- Rolling update must be avoided during scheduled reclamation procedure.

Configuring Instant Reclamation Process for Under-Utilized IP Chunks

Use the following configuration to execute instant reclamation of under-utilization of IP chunks:

exec-ipam reclaim-chunk { utilization-threshold *utilization_threshold* **inactivity-threshold** *inactivity_threshold* **[instance** *grInstance* **] [pool-name** *poolName* **] [chunk-start-ip** *ip* **] }**

NOTES:

- **exec-ipam**—This command executes IPAM commands.
- **reclaim-chunk**—This CLI executes IP chunk reclamation procedure.
- **utilization-threshold** *utilization_threshold*—This CLI is used to configure the utilization threshold for reclamation.
- **inactivity-threshold** *inactivity_threshold*—This CLI is used to configure the inactivity threshold for reclamation.
- **instance** *grInstance*—This CLI allows configuring GR instance ID.
- **pool-name** *poolName*—This CLI allows defining the pool name.
- **chunk-start-ip** *ip*—This CLI allows defining the Chunk Start IP of a specific pool.

Configuration Example

Following is the sample configuration for instant IP chunk reclamation process:

```
exec-ipam reclaim-chunk utilization-threshold 2 inactivity-threshold 180 instance 1 pool-name
 poolv4 chunk-start-ip 209.165.201.1
```

Configuring Periodic Reclamation Process for Under-Utilized IP Chunks

Use the following configuration to trigger periodic reclamation of under-utilization of IP chunks:

```
config
   ipam
      instance instance_id
         chunk-reclamation { schedule tod-hour tod_hour_value schedule
tod-minute tod_min_value utilization-threshold utilization_threshold
inactivity-threshold inactivity_threshold }
         exit
      exit
   exit
```

NOTES:

- **chunk-reclamation**—This CLI is used to configure periodic IP chunk reclamation.
- **schedule**—This CLI is used to configure Time of Day values for the chunk-reclamation process.
- **tod-hour** *tod_hour_value*—This CLI is used to configure the Time-of-day hour value for the chunk-reclamation process. The value range for **tod-hour** is <0-23>.
- **tod-minute** *tod_min_value*—This CLI is used to configure the Time-of-day minute value for the chunk-reclamation process. The value range for **tod-minute** is <0-59>.
- **utilization-threshold** *utilization_threshold*—This CLI is used to configure the utilization threshold for reclamation. The value range for **utilization-threshold** is <0-20>. The default value is 2.
- **inactivity-threshold** *inactivity_threshold*—This CLI is used to configure the inactivity threshold for reclamation. The value range for **inactivity-threshold** is <0-3600>. The default value is 1800.

Note It is suggested to configure Time-of-day when traffic is expected to be low.

Configuration Example

Following is the sample configuration for periodic IP chunk reclamation process:

```
config
  ipam
    instance <instance>
      chunk-reclamation
        schedule tod-hour <0..23> tod-minute <0..59>
        utilization-threshold <0..20>
        inactivity-threshold  <0..3600>
      exit
    exit
  exit
exit
```

Configuration Verification

To verify the confiuration, following show command should be used:

```
[smf] smf# show running-config ipam instance 1 chunk-reclamation
Tue May  16 20:36:58.887 UTC+00:00
ipam
instance 1
  chunk-reclamation
   schedule tod-hour 22
   schedule tod-minute 10
   utilization-threshold 5
   inactivity-threshold  600
  exit
exit
exit
```

OAM Support

This section discusses the metrics supported in this feature.

Bulk Statistics

Following counters are supported as part of chunk reclamation process:

IPAM_chunk_reclamation_count: It is a counter that specifies the total IP chunks reclaimed. It is triggered when an IP chunk gets reclaimed. This metric shall be maintained at GR-Instance, IP pool, address-type, UPF, remote instance, and trigger state levels.

disc_ip_chunk_reclamation: This counter is added as a reason under **smf_active_call_disconnect_stats** and **smf_disconnect_stats** statistics. This metric is introduced for sessions cleared due to chunk reclamation.

Unique IP Pools for UPFs

Feature Description

With this feature, SMF enables you to perform the following tasks:

- Allocate specific set of IP pools for edge UPFs in such a way that the UPFs do not share the same IP pool
- Fall back to centrally located UPF when the edge UPF is down

For unique IP pool assignment to UPFs, SMF uses tag with IP address pools in the IPAM configuration based on the location DNN. Then, the SMF associates this tag name while configuring UPF selection for each DNN.

To implement the fall back to central UPF, SMF provides option to configure a central UPF if edge UPF is down.

Configuring SMF for Unique IP Pools

This section provides the configurations that are required to ensure unqiue IP address allocation to the UPFs.

Configuring this feature involves the following steps:

- Configuring Tags Based on Location DNN, on page 46
- Enabling UPF Fallback, on page 46

Configuring Tags Based on Location DNN

To define the location-based DNN profile, use the following sample configuration:

```
config
   profile location-dnn location_dnn_name
      location-area-group la_group_name profile dnn_profile_name
      end
```

NOTES:

- **profile location-dnn** *location_dnn_name*—Specify the name of the location-based DNN profile.
- **location-area-group** *la_group_name* **profile** *dnn_profile_name*—Specify the name of location area group where the subscriber belongs to and the DNN profile.

  Based on the location defined in this profile, SMF tags the IP pools and selects the UPF for each DNN.

Enabling UPF Fallback

To enable the UPF fallback functionality with unique IP pools, use the following sample configuration:

```
config
   profile dnn dnn_profile_name
      dnn rmgr dnn_name fallback secondary_dnn_name
      end
```

NOTES:

- **profile dnn** *dnn_profile_name*—Specify the name of the DNN profile.
- **dnn rmgr** *dnn_name* **fallback** *secondary_dnn_name*—Specify the name of primary and secondary DNNs.

  SMF enables the fallback to centrally located UPF based on the DNN when any of the following conditions are fulfilled:

  - IP pool and UPF selected based on location fails.
  - UPF of the configured DNN is down.
  - Location of the UE isn't configured.

Configuration Example

The following is an example of the configuration used for unique IP pool allocation.

```
config
profile location-area-group lag1
 tai-group tai-grp
exit
profile location-area-group lag2
 tai-group tai-grp2
exit
profile location-dnn dnnloc-1
 location-area-group  lag1 profile dnnprof-ims-1
 location-area-group  lag2 profile dnnprof-ims-2
exit
policy dnn polDnn
 dnn ims profile dnnprof-ims //fallback dnn profile
 dnn ims location-dnn-profile  dnnloc-1 //location-based dnn profile
exit
profile upf-group upf-group1
 location-area-group-list [ lag1 ] //grouping upf based on location
 failure-profile FHUP
exit
profile upf-group upf-group2
 location-area-group-list [ lag2 ] //grouping upf based on location
 failure-profile FHUP
exit
profile upf-group upf-group3 // central upf group - no location tag
 failure-profile FHUP
exit

profile network-element upf nfprf-upf1
 node-id           n4-peer-DAUI0301
 n4-peer-address ipv4 209.165.201.3
 n4-peer-port      8805
 upf-group-profile upf-group1//ims-lag1 picks upf-group1, based on location
 dnn-list          [ ims-lag1 magenta-ims-dnn sos-pool-ipv6 ]
 capacity          10
 priority          1
exit
profile network-element upf nfprf-upf3
 node-id           n4-peer-DAUI0303
 n4-peer-address ipv4 209.165.201.4
 n4-peer-port      8805
 upf-group-profile upf-group1//ims-lag1 picks upf-group1, based on location
 dnn-list          [ ims-lag1 magenta-ims-dnn sos-pool-ipv6 ]
 capacity          10
 priority          1
exit
profile network-element upf nfprf-upf5
 node-id           n4-peer-DAUI0305
 n4-peer-address ipv4 209.165.201.5
 n4-peer-port      8805
 upf-group-profile upf-group2//ims-lag2 picks upf-group2, based on location
 dnn-list          [ ims-lag2 magenta-ims-dnn sos-pool-ipv6 ]
 capacity          10
 priority          1
exit
profile network-element upf nfprf-upf7
 node-id           n4-peer-DAUI0307
 n4-peer-address ipv4 209.165.201.6
 n4-peer-port      8805
 upf-group-profile upf-group2//ims-lag2 picks upf-group2, based on location
 dnn-list          [ ims-lag2 magenta-ims-dnn sos-pool-ipv6 ]
 capacity          10
 priority          1
exit
profile network-element upf nfprf-upf8
 node-id           n4-peer-DAUI0308
 n4-peer-address ipv4 209.165.201.7
 n4-peer-port      8805
 upf-group-profile upf-group3//ims-central picks upf-group3, if location is not available
 dnn-list          [ ims-central magenta-ims-dnn sos-pool-ipv6 ]
 capacity          10
 priority          1
exit

profile dnn dnnprof-ims-1//dnn profile, where ip pool and upf is selected based on location
dnn ims-lag1 network-function-list [ upf ]
dnn rmgr ims-lag1 fallback ims-central
timeout up-idle 3600 cp-idle 7320
.
.
.
session skip-ind false
upf apn ims-lag1
qos-profile 5qi-to-dscp-mapping-table-IMS
.
.
.

profile dnn dnnprof-ims-2//dnn profile, where ip pool and upf is selected based on location
dns primary ipv4 209.165.200.225
dns primary ipv6 fd00:976a::9
dns secondary ipv4 209.165.200.226
dns secondary ipv6 fd00:976a::10
dnn ims-lag1 network-function-list [ upf ]
dnn rmgr ims-lag1 fallback ims-central
timeout up-idle 3600 cp-idle 7320
.
.
.
profile dnn dnnprof-ims//dnn profile, where ip pool and upf selected based on location fails
 but falls back based on dnn based on precedence
dns primary ipv4 209.165.200.227
dns primary ipv6 fd00:976a::9
dns secondary ipv4 209.165.200.228
dns secondary ipv6 fd00:976a::10
dnn ims-central network-function-list [upf ]
dnn rmgr ims-central
```

```
timeout up-idle 3600 cp-idle 7320
.
.
.

config
 ipam
  instance 1
   source local
   address-pool ims-ipv6-pool1
   address-quarantine-timer 3600
    vrf-name                 n6
    tags
     dnn ims-lag1//ip pool for upf-group1 and dnn profile dnnprof-ims-1
    exit
    ipv4
    address-range 1.1.1.0 1.1.10.254
    exit
    ipv6
    prefix-ranges
     split-size
      per-cache 65536
      per-dp    65536
     exit
    exit
   exit
   address-pool ims-ipv6-pool2
   address-quarantine-timer 3600
    vrf-name                 n6
    tags
     dnn ims-lag2//ip pool for upf-group2 and dnn profile dnnprof-ims-2
    exit
    ipv4
    address-range 2.1.1.0 2.1.10.254
    exit
   ipv6
    prefix-ranges
    split-size
    per-cache 65536
    per-dp    65536
   exit
    prefix-range 2607:fc20:8aa0:: length 44
   exit
   exit
   address-pool ims-ipv6-pool3
   address-quarantine-timer 3600
    vrf-name                 n6
    tags
    dnn ims-central//ip pool for upf-group3 and dnn profile dnnprof-ims
    exit
   ipv4
   address-range 3.1.1.0 3.1.10.254
   exit
   ipv6
   prefix-ranges
    split-size
    per-cache 65536
    per-dp    65536
   exit
   prefix-range 3607:fc20:8aa0:: length 44
   exit
   exit
   exit
```

Identification of Corrupted IP Chunks

Feature Description

SMF supports identification of IP chunks that are missing or corrupted, by auditing the configured IP pools and their data in cache-pod and IPAM internal states.

SMF identifies the corrupted IP chunks in the following scenarios:

1. IP chunk is free and also allocated to a UPF.
2. IP chunk is locked but not allocated to any UPF.
3. Corrupted due to geo replication issues.

These audits can be performed either from the primary or secondary node to know about any discrepancies between the current visible IPAM configuration and the available IPAM cache data (cache-pod and internal IPAM cache state).

How It Works

This feature allows capturing the chunk status under the following scenarios:

- The chunk entry is missing from the IPAM cache-pod data.
- The chunk entry is present, however, has some missing essential non-unique-keys such as pool-name, address-type, gr-instance-id, and so on.
- The chunk is showing as allocated (locked) in the cache-pod, but it is not allocated by the node manager to any UPF.
- The chunk is showing as free in the cache-pod, but the node manager has allocated the IP chunk to a UPF (DP).

OAM Support

This section discusses the metrics and statistics supported in this feature.

Bulk Statistics

Following metric is supported under the new category **SMF IPAM Pool Total Chunks Counter Category**:

IPAM_pool_total_chunks: It captures the total number of chunks for the pool.

- **Metrics-Type**: Gauge

Following labels are supported as part of this metric:

- **grInstId**: GR Instance ID
- **pool**: Address pool name
- **addressType**: Address-Type (ipv4/ipv6)

- **IPAM_Chunk_Alloc_Type**: Chunk allocation status

  It has the following possible values:

  - **FreeCP**: Chunks free in cache-pod
  - **FreeNM**: Chunks free with node manager instance
  - **AllocatedUPF**: Chunk allocated to UPF

Note The parameter FreeCP label captures the total number of free chunks available in the cache-pod. As this value is common to the node manager instances, it is captured only by one of the node manager instances (leader). This way the Grafana sum query shows the correct total chunk count.

Reconciliation of IP Chunks between SMF and UPF

Feature Description

In some unforeseen scenarios, there is a possibility of mismatch between the list of DP chunks allocated at SMF and UPF. This can lead to the allocation of duplicate IPs in the network.

SMF supports reconciliation of IP chunks that allows to avoid duplicate IP allocation.

How It Works

As part of the new DP chunk allocation to UPF, the following mechanism must be followed:

1. Allocate a free DP chunk to the UPF.
2. Clear the subscribers from the node manager to CDL to free subscriber sessions matching the following criteria:
   - Newly allocated DP chunk (ipv4-startrange / ipv6-startrange as the non-unique keys for freeing the sessions).
   - Sessions created before the current system time.
3. Trigger N4 Association update with Route Deletion Request to all other UPFs serving the specific DNN, except for the UPF selected as part of the new chunk allocation.

   Note All these steps must be triggered in-parallel.

4. During **clear subscriber** trigger, use the new correlation Id as CorrelationIdDpChunkAuditLocal (0xFF0D), which must be used as part of the metrics during session deletion.
5. The DNN can be Resource Manager DNN /Incoming DNN.

Note In a sunny day scenario, **clear subscriber** from the node manager to CDL should not match any sessions in CDL. Also, the N4 Association update with the Route Deletion request should not have any effect on the UPFs as they are not using this chunk or route.

Configuration to Enable or Disable Reconciliation of IP Chunks

Use following configuration to enable or disable the feature Reconciliation of IP Chunks between SMF and UPF:

```
config
   ipam
      instance gr_instance_id
         audit chunk [ local | none ]
           end
```

NOTES:

- **audit chunk** *[local | none]* —Configures audit activity on IPAM. **audit chunk** has two possible values:
  - *local* —Enables the feature.
  - *none* —Disables the feature.

  Default value is *none*.

Configuration Example

The following is an example configuration:

```
[smf] smf(config)# ipam instance 1 audit chunk
Possible completions:
  local   Enable local audit
  none    Disable audit
[smf] smf(config)# ipam instance 1 audit chunk local
```

OAM Support

This feature supports following metrics and statistics:

Bulk Statistics

Following label is added in the reasons in the existing bulk statistics **smf_disconnect_stats**:

- **dp_chunk_audit_local**—Session disconnected due to IPAM local audit.

IP Chunk Auto-Throttle and ToD Chunk Clearance

Feature Description

SMF supports the auto-throttling feature to better utilize the IP pools and manage the IP address chunk allocation across multiple UPFs. Enabling this feature helps balance the load across UPFs by allowing the SMF to throttle additional or new IP address chunk allocation based on UPF capacity as advertised by the UPF, or local configuration on SMF.

In this feature, when the throttle hits, then no further chunk allocation happens. However, if the chunks are already allocated, it continues to facilitate the calls until all the chunks are exhausted.

Maximum supported sessions can be configured:

1. Per network element (UPF) level
2. Per vDNN basis (also applicable per UPF level)

If both are configured, then the minimum of two values gets applied for that UPF. Both these configurations are done using the new introduced CLI **max-upf-sessions** .

How It Works

The process of auto-throttling happens through the following steps:

1. SMF supports a new proprietary IE in PFCP association setup request coming from a UPF, which carries the maximum session capacity of a UPF.

   For more information on the support of this IE from the UPF side, see the *UCC 5G UPF Configuration and Administration Guide, Release2023.03*.

2. Maximum session capacity per UPF can be configured on SMF as well. This configuration can be done per UPF configuration or per UPF, per DNN.
3. As part of the PFCP association, the SMF allocates IP chunk to UPF per node manager.
4. When IP usage per UPF reaches the threshold of 80% of total IP allocated, SMF tries to allocate further chunks. The allocation for a new chunk can also happen based on internal changes like pool config changes, pool threshold hit, and so on.
5. Using this feature, SMF now compares the IP usage of the UPF, which includes current IP usage by the active sessions and also the IPs which are in quarantine, with the maximum session capacity of the UPF.
6. SMF throttles chunk allocation, if the current usage above the maximum session capacity.

Limitations

Following are the known limitations of this feature:

- Throttling is not applicable when the initial chunk is allocated to each node manager instance on a UPF association.

- The split-size configuration must be less than the session limits configured in SMF/UPF. Ideally, it must be less than half considering that initial chunks are allocated to both the native node manager instances and also to the remote nodemgr instance (for GR deployment).
- In cases where the session limits are configured with lower values, the actual minimum value applicable will be in relation to the chunk size (cache-split & dp-split) configuration.

Configuring Maximum Supported Session for UPF on SMF

Following CLI configures the maximum session supported per UPF:

```
config
  profile network-element upf upf_profile_name
    max-upf-sessions max_upf_sessions_count
    n4-peer-address {  [ ipv4-address  ipv4_address ] [ ipv6-address 
ipv6_address ] }
    n4-peer-port  port_number
    dnn-list dnn_list_value
    capacity lb_capacity
    priority lb_priority
    end
```

NOTES:

- **max-upf-sessions** *max_upf_sessions_count*—Maximum sessions supported for a particular UPF.

Configuration Example

Following is the sample configuration for defining Maximum Supported Sessions for UPF:

```
profile network-element upf upf1
 max-upf-sessions 2000
 n4-peer-address ipv4 10.1.8.48
 n4-peer-port 8805
 dnn-list     [ intershat intershat1 intershat2 intershat3 intershat4 intershat5 intershat6
 intershat7 intershat_hrt intershatipex spectrum ]
 capacity     65535
 priority     65535
exit
```

Configuring Maximum Sessions Supported Per vDNN Per UPF

SMF supports the configuration of maximum sessions supported per vDNN per UPF. Initial DNN profile selection happens based on UE DNN. SMF selects the vDNN based on the DNN RMGR configuration under the initial DNN profile. Additional DNN profiles can be configured for this vDNN and the CLI **max-upf-sessions** can be configured under this vDNN based DNN profile.

Note

- The **max-upf-sessions** can be configured under the initial DNN profile also based on the use case.
- The DNN RMGR must be configured as a DNN profile where the **max-upf-sessions** can be configured.

```
config
  profile dnn dnn_profile_name
    network-element-profiles { amf | chf | pcf | sepp | udm } profile_name
 | ipv6 ipv6_address }
    dnn  ims network-function-list [ chf | pcf | udm | upf ]
    timeout  up-idle up_idle_duration cp-idle cp_idle_duration
    charging-profile  charging_profile_name
    pcscf-profile  profile_name
    ppd-profile  profile_name
    ssc-mode  [ 1 | 2 | 3 ] allowed [ 1 | 2 | 3 ]
    session type  default_session_type
    max-upf-sessions max_upf_sessions_count
    end
```

Configuration Example

Following is the sample configuration for defining Maximum Supported Session per vDnn per UPF on SMF:

```
profile dnn dnnprof-ims
 dns primary ipv4 10.177.0.34
 dns primary ipv6 fd00:976a::9
 dns secondary ipv4 10.177.0.210
 dns secondary ipv6 fd00:976a::10
 dnn rmgr ims-pool-ipv6
 wps-profile dynamic-wps
 upf apn ims
exit

profile dnn dnnprof-ims.epdg.prod
 dns primary ipv4 10.177.0.35
 dns primary ipv6 fd00:976a::8
 dns secondary ipv4 10.177.0.211
 dns secondary ipv6 fd00:976a::11
 dnn rmgr ims-pool-ipv6
 upf apn ims.epdg.prod
exit

profile dnn ims-pool-ipv6
 network-element-profiles chf nfprf-chf1
 network-element-profiles amf nfprf-amf1
 network-element-profiles pcf nfprf-pcf1
 network-element-profiles udm nfprf-udm1
 dnn ims network-function-list [ chf pcf udm upf ]
 timeout up-idle 3600 cp-idle 7320
 charging-profile          chgprof-1
 pcscf-profile             pcscf1
 ppd-profile               ppd-prof1
 ssc-mode 1 allowed [ 2 ]
 session type IPV6
 max-upf-sessions 10000
exit
```

OAM Support

This section covers the metrics supported in this feature.

Metrics

Following new metric was introduced to capture the event when new chunk allocation is throttled:

Metric Name: IPAM_DP_chunk_allocation_throttled

Description: This metric captures the event when a new chunk allocation request is throttled for a UPF for a specific address-type and DNN. The new chunk allocation request is normally triggered on every address allocation request after hitting the upper threshold (80%) for the specific address-type and DNN.

Following labels are added in this metric:

- **grInstId:** GR Instance ID
- **upf:** UPF Key
- **addressType:** Address Type (IPv4/IPv6-PD/IPv6
- **dnn:** DNN Tag name
- **chunkAllocTrigger:** Trigger for new chunk allocation
    - **Threshold_Hit_Addr_Alloc**: Threshold hit after new address allocation.
    - **Threshold_Hit_DP_Monitoring**: Threshold hit after pool monitoring for that UPF.
    - **Insufficient_Addr_Space**: If sufficient addresses are not available during IP allocation.

Route Aggregation to Handle Switch Limit

Feature Description

For every IP chunk that SMF allocates to a UPF, the UPF publishes a route record to upstream routers for advertising the IP address subnets toward the data network. However, some of the routers have limitation with the number route records that it can allow.

It limits the number of IP chunks that SMF can allocate to a UPF and is forced to use bigger chunks (32K / 16K). But, using bigger IP chunks causes under-utilization of the allocated IP chunks, therefore, those chunks cannot be used by native instances.

SMF allows defining smaller chunk sizes and hence, limiting the number of route records that are published using some route record optimizations.

How It Works

Chunk size defining and route aggregation is done by allocating continuous IP chunks to a UPF and publishing a single or aggregated route records as per the subnet of the continuous chunk. This functionality works in the following process:

- Smaller chunk sizes, such as 4K, in consecutive fashion are defined.
- During allocation, IPAM allocates a complete group to a UPF and single route record, as per chunk group range, is sent to UPF.

- The chunk groups have 2, 4, or 8 chunks within it. This way, one group consists of 8K, 16K, or 32K size, and accordingly the route record of that size is sent to the UPF.
- Node manager instances work on the smaller IP chunks only.
- Upon an initial UPF registration, chunks for the remote node manager are also allocated from the same group.
- When the chunks start getting freed up, they return to the free pool only when all the chunks within the group are freed.
- To further optimize the routes sent to the UPF, **reserve-contiguous-groups** option can also be enabled. With this SMF upfront reserves multiple continuous chunks-groups (as per the **max-session-size** configured for the UPF/vDNN) for the UPF and push only one route record toward the UPF. The option **reserve-contiguous-groups** configuration must be enabled along with the complete address pool configuration. When enabled, the option to disable it, is not supported. If the user wishes to change the pool behavior, they must clear the subscribers associated with the pool, delete the address pool configuration, and re-configure it without the option.
- The cache-split and dp-split values must be the same when the chunk-groups are to be enabled for the address-pool. In case the dp-split and cache-split values are different then the node manager pulls more chunks.

Migration of Old Pools Without Chunk-Group into Pools with Chunk-Group

Migration of IP pools is carried-out in the following process:

1. Add new pools with chunkGroupSize configuration for the same vDNN (with IP ranges unique across the pools within the system and also the subnets for the address-ranges of the new pool must be as per standard IP-subnet). Here, **reserve-contiguous-groups** can be enabled or disabled based on the usage or planning.
2. Check if the new pools are showing from **show ipam pool** output.

   Sample config:

   ```
   [smf-m6-cndp-rack2/data] smf# show ipam pool Mon Jul 24 08:34:36.617 UTC+00:00
   ==========================================================================================
   PoolName Ipv4Utilization Ipv6AddrUtilization Ipv6PrefixUtilization
   ==========================================================================================
   data-ipv6-pool2 92.23% 0.00% 97.66%
   data-ipv6-pool1 59.42% 0.00% 34.40%
   data-ipv6-pool4 0.00% 0.00% 0.00%
   data-ipv6-pool3 1.70% 0.00% 1.10%
   ==========================================================================================
   [smf-m6-cndp-rack2/data] smf#
   ```

3. When the newly added pool is visible from **show ipam pool** output, check if the new pool is active with chunk-group-wise allocation as per chunks-per-group configuration. It can be done using the command **show ipam pool <newly added pool> ipv4-addr/ ipv6-prefix**.

   Sample config:

   ```
   show ipam pool data-ipv6-pool4 ipv4-addr/ ipv6-prefix
   ```

4. Mark the old pools without chunk-group as offline. Expectation is New IPs must not be allocated from the old pool further.

5. For the ongoing traffic, chunks from the new pool will be allocated as per chunk-group configuration. It can be checked using the same CLI.

   show ipam dp <DP association> show ipam pool <new pool> ipv4-addr/ ipv6-prefix
6. As sessions associated with IPs from the old pools are terminated, they release the IPs from the old pools, and further release their chunks.
7. Wait until all the chunks from the old pools are released.

Note If there are further IPs still stuck/stale with old pools, then the subscribers can be cleared with admin-clear CLIs. For example, **show subscriber count nf-service smf dnn <dnn> show subscriber count nf-service smf ipv4-pool/ ipv6-pool <old pool>**, and **clear subscriber nf-service smf ipv4-pool/ ipv6-pool <old pool>**. Wait for all the subscribers to get cleared from the old pools with this CLI and the corresponding old routes toward the UPF must also get deleted.

8. Check if all the subscribers are moved back in the new pool with chunk-group.

   show subscriber count nf-service smf ipv4-pool/ ipv6-pool <new pool>
9. When all the subscriber move back in the new pool, delete the configuration for the old pool from IPAM.

```
[smf-m6-cndp-rack2/data] smf(config)# ipam instance 1
Mon Jul 24 08:43:19.875 UTC+00:00
[smf-m6-cndp-rack2/data] smf(config-instance-1)# no address-pool data-ipv6-pool2
```

Note It is recommended to carry-out the migration of old pools without chunk-group into pools with chunk-group in low-traffic hours/maintenance window.

Limitations

Following are the known limitations of this feature:

- Chunk Groups must be used for managing larger address pools and not for smaller address pools.
- The chunk-size (dp-split) must be 2K, 4K, 8K, and so on. It is not suitable in cases where the dp-split is less than 256.
- The values of cache-split and dp-split must be the same when chunk-groups are to be enabled for the address-pool.

Configuring IPv4 Address Range to Define Chunk Group Size

To define the small chunk-group size for IPv4 address range use the following configuration:

```
config
  ipam
    instance gr_instance_id
      address-pool pool_name
          ipv4
```

```
chunk-group chunks-per-group { 2 | 4| 8 }
exit
```

NOTES:

- **chunk-group**—This CLI configures chunk-groups for a pool.
- **chunks-per-group** *{ 2 | 4| 8 }*— This CLI defines the number of chunks in a chunk group. The values can be either 2, 4, or 8. Therefore, the chunk-group can be calculated as:

  Chunk Group= Chunk Size * 2K or 4K or 8K

Note This feature does not support or recommend the configuration of odd-numbered chunks-per-group.

Configuration Example

Following is the sample configuration for defining the chunk group size for the IPv4 address range:

```
config
ipam
 instance 1
  address-pool poolv4
   vrf-name ISP
   tags
    dnn intershat
   exit
   ipv4
    split-size
     per-cache 256
     per-dp    256
    exit
    chunk-group
     chunks-per-group 4
    exit
    address-range 10.0.0.1 10.0.10.254
   exit
  exit
 exit
exit
```

Configuration Verification

Following is the output of the configuration of IPv4 address range to define chunk group size:

```
[smf] smf# show ipam pool poolv4 ipv4-addr

======================================================================================
Flag Indication: S(Static) O(Offline)
======================================================================================
StartAddress          EndAddress         AllocContext                   Flag  GroupId
======================================================================================
10.0.0.1              10.0.0.255         Free:CP                              1
10.0.1.0              10.0.1.255         Free:CP                              1
10.0.2.0              10.0.2.255         Free:CP                              1
10.0.3.0              10.0.3.255         Free:CP                              1
10.0.4.0              10.0.4.255         10.1.13.52:10.1.8.184                2
10.0.5.0              10.0.5.255         10.1.13.52:10.1.8.184                2
10.0.6.0              10.0.6.255         Free:CP                              2
```

```
10.0.7.0              10.0.7.255         Free:CP                              2
10.0.8.0              10.0.8.255         Free:CP                              3
10.0.9.0              10.0.9.255         Free:CP                              3
10.0.10.0             10.0.10.254        Free:CP                              3
```

Note The CLI **show ipam pool <pool-name> <ipv4-addr|ipv6-prefix|ipv6-addr>** output is updated to have an additional column *GroupId* to show the chunk Group ID. Chunks that belong to the same group reflect in the same chunk Group ID.

Configuring IPv6 Prefix Range to Define Chunk Group

Use the following configuration to define the small chunk-group size for the IPv6 address range:

```
config
   ipam
      instance instance_id
         address-pool pool_name
            ipv6
               prefix-ranges
                  prefix-range Prefix-range
                  chunk-group chunks-per-group { 2 | 4| 8 }
                  exit
```

Configuration Example

Following is the sample configuration for defining chunk group size for the IPv6 prefix range:

```
ipam
 instance 1
  address-pool poolv6
   vrf-name ISP
   tags
    dnn intershat
   exit
   ipv6
    prefix-ranges
     split-size
      per-cache 8192
      per-dp    1024
     exit
     chunk-group
      chunks-per-group 4
     exit
     prefix-range 2001:db0:: length 48
    exit
   exit
  exit
 exit
exit
```

Configuring IPv6 Address Range to Define Chunk Group

Use the following configuration to define the small chunk-group size for the IPv6 address range:

```
config
   ipam
      instance gr_instance_id
         address-pool pool_name
            vrf-name vrf_name
            ipv6
               address-range start_ipv6_address end_ipv6_address
               chunk-group chunks-per-group { 2 | 4| 8 }
               exit
```

Configuration Example

Following is the sample configuration for defining chunk group size for the IPv6 address range:

```
ipam
 instance 1
  address-pool poolv6DNN2
   vrf-name ISP
   tags
    dnn intershat1
   exit
   ipv6
    address-ranges
     split-size
      per-cache 1024
      per-dp    1024
     exit
     chunk-group
      chunks-per-group 4
     exit
     address-range 64:ff9b:: 64:ff9b::ffff:ffff
    exit
   exit
  exit
 exit
exit
```

Configuration Verification for IPv6 Prefix Range and Address Range

The configuration for IPv6 prefix range and address range can be verified using the following show command:

```
[smf-m6-cndp-rack2/data] smf# show ipam dp 100.105.64.10:10.210.184.49 ipv6-p
Tue Jul  11 12:08:36.232 UTC+00:00

==========================================================================================
Flag  Indication: S(Static) O(Offline) R(For Remote Instance) RF(Route Sync Failed)
G:N/P Indication: G(Cluster InstId) N(Native NM InstId) P(Peer NM InstId)
==========================================================================================
StartAddress                 EndAddress               Route                    G:N/P
Utilization   Flag      AllocContext
==========================================================================================
2607:fb90:8700::             2607:fb90:8700:fff::     2607:fb90:8700::/52      1:0/1
100.00%                 data-ipv6-pool1(ISP)
2607:fb90:8700:1000::        2607:fb90:8700:1fff::    2607:fb90:8700:1000::/52 1:1/0
100.00%                 data-ipv6-pool1(ISP)
==========================================================================================
```

The allocated context per chunk-group for IPv6 can be verified using the following command:

show ipam pool data-ipv6-pool2 ipv6-prefix | include (DP-Name)

A sample output/format is given here:

```
[smf-m6-cndp-rack2/data] smf# show ipam pool data-ipv6-pool2 ipv6-prefix | include
100.105.64.80:10.210.184.49
Mon Jul 17 07:43:20.510 UTC+00:00
2607:fb90:8732:400::/64 2607:fb90:8732:7ff::/64 100.105.64.80:10.210.184.49 33
2607:fb90:8732:800::/64 2607:fb90:8732:bff::/64 100.105.64.80:10.210.184.49 33
2607:fb90:8732:1000::/64 2607:fb90:8732:13ff::/64 100.105.64.80:10.210.184.49 34
2607:fb90:8732:1800::/64 2607:fb90:8732:1bff::/64 100.105.64.80:10.210.184.49 34
2607:fb90:8732:2000::/64 2607:fb90:8732:23ff::/64 100.105.64.80:10.210.184.49 35
2607:fb90:8732:2800::/64 2607:fb90:8732:2bff::/64 100.105.64.80:10.210.184.49 35
2607:fb90:8732:3000::/64 2607:fb90:8732:33ff::/64 100.105.64.80:10.210.184.49 36
2607:fb90:8732:3800::/64 2607:fb90:8732:3bff::/64 100.105.64.80:10.210.184.49 36
2607:fb90:8732:4000::/64 2607:fb90:8732:43ff::/64 100.105.64.80:10.210.184.49 37
2607:fb90:8732:4800::/64 2607:fb90:8732:4bff::/64 100.105.64.80:10.210.184.49 37
2607:fb90:8732:5000::/64 2607:fb90:8732:53ff::/64 100.105.64.80:10.210.184.49 38
2607:fb90:8732:5800::/64 2607:fb90:8732:5bff::/64 100.105.64.80:10.210.184.49 38
2607:fb90:8732:6000::/64 2607:fb90:8732:63ff::/64 100.105.64.80:10.210.184.49 39
2607:fb90:8732:6800::/64 2607:fb90:8732:6bff::/64 100.105.64.80:10.210.184.49 39
2607:fb90:8732:7000::/64 2607:fb90:8732:73ff::/64 100.105.64.80:10.210.184.49 40
2607:fb90:8732:7800::/64 2607:fb90:8732:7bff::/64 100.105.64.80:10.210.184.49 40
2607:fb90:8732::/64 2607:fb90:8732:3ff::/64 100.105.64.80:10.210.184.49 33
2607:fb90:8732:c00::/64 2607:fb90:8732:fff::/64 100.105.64.80:10.210.184.49 33
2607:fb90:8732:1400::/64 2607:fb90:8732:17ff::/64 100.105.64.80:10.210.184.49 34
2607:fb90:8732:1c00::/64 2607:fb90:8732:1fff::/64 100.105.64.80:10.210.184.49 34
2607:fb90:8732:2400::/64 2607:fb90:8732:27ff::/64 100.105.64.80:10.210.184.49 35
2607:fb90:8732:2c00::/64 2607:fb90:8732:2fff::/64 100.105.64.80:10.210.184.49 35
2607:fb90:8732:3400::/64 2607:fb90:8732:37ff::/64 100.105.64.80:10.210.184.49 36
2607:fb90:8732:3c00::/64 2607:fb90:8732:3fff::/64 100.105.64.80:10.210.184.49 36
2607:fb90:8732:4400::/64 2607:fb90:8732:47ff::/64 100.105.64.80:10.210.184.49 37
2607:fb90:8732:4c00::/64 2607:fb90:8732:4fff::/64 100.105.64.80:10.210.184.49 37
2607:fb90:8732:5400::/64 2607:fb90:8732:57ff::/64 100.105.64.80:10.210.184.49 38
2607:fb90:8732:5c00::/64 2607:fb90:8732:5fff::/64 100.105.64.80:10.210.184.49 38
2607:fb90:8732:6400::/64 2607:fb90:8732:67ff::/64 100.105.64.80:10.210.184.49 39
2607:fb90:8732:6c00::/64 2607:fb90:8732:6fff::/64 100.105.64.80:10.210.184.49 39
2607:fb90:8732:7400::/64 2607:fb90:8732:77ff::/64 100.105.64.80:10.210.184.49 40
```

Pre-Allocation of IP Chunks Based on Maximum Session Size

Use the following configuration to define the small chunk-group size for IPv6 address range:

```
config
   ipam
      instance instance_id
         address-pool pool_name
            ipv4
               chunk-group chunks-per-group { 2 | 4| 8 }
               reserve-contiguous-groups
                  exit
```

NOTES:

- **reserve-contiguous-groups**—If **reserve-contiguous-groups** is enabled and **max-upf-sessions** is configured, then IPAM will reserve the chunk-group and chunks in the power of 2 to cover the value of **max-upf-sessions**.

For example, if the **max-upf-sessions** is 60000 and **reserve-contiguous-groups** is enabled with DP-size 8192 and **chunks-per-group** is 4, then IPAM reserves two chunk-groups in this case with 65536 (2 to the power of 16) IPs and single-route is published for this toward the UPF.

Configuration Example

Following is the sample configuration for pre-allocating multiple continuous chunks as per the max-session-size configured for the UPF/vDNN:

```
[smf] smf# show running-config ipam instance 1 address-pool poolv4
Sun Jun  18 20:00:19.226 UTC+00:00
ipam
 instance 1
  address-pool poolv4
   vrf-name ISP
   tags
    dnn intershat
   exit
   ipv4
    split-size
     per-cache 256
     per-dp    256
    exit
    chunk-group
     chunks-per-group          4
     reserve-contiguous-groups
    exit
    address-range 10.0.0.1 10.0.10.254
   exit
  exit
 exit
exit
```

OAM Support

This section covers the metrics details supported in this feature.

Metrics

This feature adds a new label **groupID** in the following two metrics:

1. **IPAM_chunk_events_total**
2. **IPAM_chunk_allocations_current**

groupID: This label captures the additional information about the chunk-group ID from which the chunk is allocated/released for an UPF. Following configuration helps cature this lable:

```
config
   infra metrics verbose application metrics IPAM_chunk_allocations_current level production
 granular-labels [ groupID ]
   infra metrics verbose application metrics IPAM_chunk_events_total level production
granular-labels [ groupID ]
```

Note **groupID** is a granular label and is not enabled by default due to high cardinality for this label's value.

Troubleshooting Information

This section provides information on using the command line interface (CLI) commands, alerts, logs, and metrics for troubleshooting issues that may arise during system operation.

Range of IPv6 Allocated to UPF

The **show ipam dp** *dp_name* **ipv6-prefix** CLI command displays the IP pool chunks allocated to UPF. This pool chunk includes the VRF tag information and details, such as whether the pool defined is a static or dynamic pool.

[unknown] smf# show ipam dp 198.18.1.3 ipv6-prefix

```
=================================================================================
Flag Indication: S(Static) O(Offline)
N/P  Indication: N(Native InstId) P(Peer InstId)
===========================================================================================
StartAddress          EndAddress             AllocContext       Route
N/P    Utilization   Flag
=================================================================================
3001:db0::             3001:db0:0:3fff::      v6pool4(vrf4@ISP)  3001:db0::/50            -
                     S
3001:db0::             3001:db0:0:3fff::      v6pool3(vrf3@ISP)  3001:db0::/50            -
                     S
3001:db0:0:4000::      3001:db0:0:7fff::      v6pool4(vrf4@ISP)  3001:db0:0:4000::/50     -
                     S
3001:db0:0:4000::      3001:db0:0:7fff::      v6pool3(vrf3@ISP)  3001:db0:0:4000::/50     -
                     S
=================================================================================
[unknown] smf#
```

Range of IPv4 Allocated to UPF

The **show ipam dp** *dp_name* **ipv4-addr** CLI command displays the IP pool chunks allocated to UPF. This pool chunk includes the VRF tag information and details, such as whether the pool defined is a static or dynamic pool.

[unknown] smf# show ipam dp 209.165.201.3 ipv4-addr

```
==========================================================================================
Flag Indication: S(Static) O(Offline)
N/P  Indication: N(Native InstId) P(Peer InstId)
==========================================================================================
StartAddress     EndAddress         AllocContext       Route              N/P   Utilization
   Flag
==========================================================================================
209.165.200.129  209.165.202.131    v4pool3(vrf3@ISP)209.165.200.129/27      -
      -S
209.165.200.129  209.165.202.131    v4pool4(vrf4@ISP)209.165.200.129/27      -
      -S
209.165.200.253  209.165.202.153    v4pool3(vrf3@ISP)209.165.200.253/27      -
      -S
209.165.200.253  209.165.202.153    v4pool4(vrf4@ISP)209.165.200.253/27      -
      -S
209.165.202.154  209.165.202.155    v4pool3(vrf3@ISP)209.165.202.154/27      -
      -S
209.165.202.154  209.165.202.155    v4pool4(vrf4@ISP)209.165.202.154/27      -
```

```
       -S
209.165.202.156  209.165.202.156    v4pool3(vrf3@ISP)209.165.202.156/27      -
       -S
209.165.202.156  209.165.202.156    v4pool4(vrf4@ISP)209.165.202.156/27      -
       -S
209.165.202.128  209.165.202.158    v4pool4(vrf4@ISP)209.165.202.128/27      -
       -S
209.165.202.129  209.165.202.158    v4pool4(vrf4@ISP)209.165.202.129/27      -
       -S
209.165.200.225  209.165.200.253    v4pool4(vrf4@ISP)209.165.200.225/27      -
       -S
209.165.201.134  209.165.201.30     v4pool4(vrf4@ISP)209.165.201.134/27      -
       -S
==========================================================================================
```

IP Pool Mapping Error Logs

The following is a sample error log for incorrect static IP to pool mapping or if static IP received from RADIUS is not found with any UPF.

```
[smf-service-n0-0] 2020/09/23 07:42:25.969 smf-service [DEBUG] [rmgrutil.go:501]
[smf-service.smf-app.resource] [imsi-123456789012345:5] [imsi-123456789012345:5] [16]
response received for message NmgrRersourceMgmtResponse
[smf-service-n0-0] 2020/09/23 07:42:25.969 smf-service [INFO] [upmgrCacheApi.go:450]
[misc-lib.upmgrcache.gen] Cache doesnot have entry for UpfEpKey:
[smf-service-n0-0] 2020/09/23 07:42:25.969 smf-service [ERROR] [rmgrutil.go:73]
[smf-service.smf-app.resource] [imsi-123456789012345:5] [imsi-123456789012345:5] [16] Both
 the associated nodemgr instances for upfEpKey:  is down
[smf-service-n0-0] *errors.errorString Both the associated nodemgr instances for upfEpKey:
  is down
[smf-service-n0-0] /opt/workspace/smf-service/src/smf-service/vendor/wwwin-github.cisco.com/
mobile-cnat-golang-lib/app-infra.git/src/app-infra/infra/Transaction.go:621 (0xd8b29e)
[smf-service-n0-0]
/opt/workspace/smf-service/src/smf-service/procedures/generic/rmgrutil.go:73 (0x14dbd61)
```