

IP Address Management

- Feature Summary and Revision History, on page 2
- Feature Description, on page 3
- How it Works, on page 4
- IPAM Integration in SMF, on page 5
- Static IP Support, on page 24
- Dual-stack Static IP Support Through IPAM, on page 37
- IPAM Offline Mode Support, on page 38
- IPAM Redundancy Support Per UPF, on page 41
- IPAM Quarantine Timer, on page 42
- IP Address Validation with CDL Configuration, on page 43
- IPAM Data Reconciliation, on page 44
- Failure handling for IPAM chunk operations, on page 47
- IP allocation through DHCP server, on page 52
- IPv6 Prefix Delegation, on page 62
- Configuring IPAM Quarantine Qsize, on page 68
- Overlapping IP Address Pools, on page 68
- Auto-Reclamation of Under-Utilized IP Chunks, on page 70
- Unique IP Pools for UPFs, on page 72
- Reconciliation of IP Chunks between SMF and UPF, on page 76
- IP Chunk Throttling, on page 78
- IPAM Cache Data Sharding, on page 83
- IP Allocation and Reuse, on page 84
- Route Aggregation to Handle Switch Limit, on page 92
- NAT Support, on page 100
- Troubleshooting Information, on page 110

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. • NAT Support: 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 the following features: • Releasing N4 resources using a CLI • Network Address Translation (NAT) along with sending NAT binding updates in SMF.	2023.04.0

Revision Details	Release
Added support for the 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
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 a 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.

Sources of IP Allocation

Whenever a UE requests an SMF for an IP address, SMF allocates the IP address to the UE from two sources:

If the IPAM source is configured ...	The IP allocation happenes from ...
as local,	the local IPAM pool. This is the default behavior. To learn more about local IPAM-based IP allocation, see the IPAM Integration in SMF topic.
as DHCP,	the DHCP Server. Learn more about DHCP-based IP allocation, see the IP Allocation through DHCP Server topic.

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

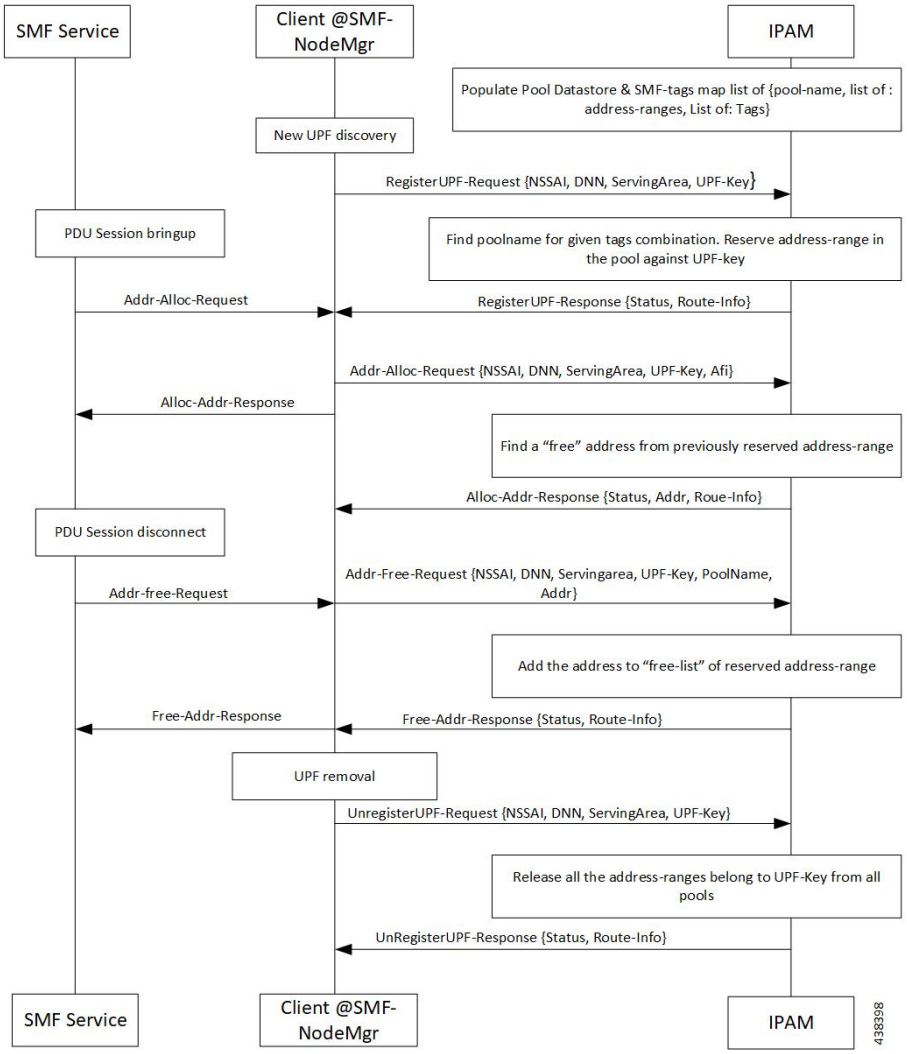


Table 3: IPAM Integration Call Flow Description

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 9
2. Configuring IPv6 Address Ranges, on page 10
3. Configuring IPv6 Prefix Ranges, on page 10
4. Configuring SMF Tags, on page 14
5. Configuring IPv4 Address Range Threshold, on page 15
6. Configuring IPv6 Address Range Threshold, on page 16
7. Configuring IPv6 Prefix Range Threshold, on page 16
8. Configuring IPv4 Address Range Split, on page 17
9. Configuring IPv6 Address and Prefix Address Range Split, on page 18
10. Configuring Global Threshold, on page 19
11. Configuring IPAM Source, on page 20

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 instance_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

Table 4: Feature History

Feature	Release	Description
Duplicate Static IP Detection and Resolution	2024.03.4	This feature provides a mechanism to handle error scenarios where the same static IP gets allocated to two different UEs or two different PDU sessions belonging to the same UE or DNN. This feature ensures that such duplicate IP allocations are detected and appropriate actions are taken to prevent conflicts. Command introduced: **condition duplicate-ip** — Use this command in policy rule management configuration to detect the duplicate static IP allocations and reject or terminate such session requests.

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.

Due to erroneous configurations, multiple UEs could be allocated the same static IP address. This can also happen with two PDU sessions belonging to the same UE or DNN, resulting in network conflicts that may lead to unpredictable behaviour.

If duplicate IP address is detected during the subsequent attach, SMF terminates the subscriber session to prevent network conflicts.

In standard 5G network implementations, the UDM holds the highest precedence in allocating static IP addresses. However, if secondary authentication is enabled and the subscriber reattaches to the network, the AAA server takes precedence over the UDM.

The benefits of detecting duplicate static IP allocation are:

- Prevents IP Conflicts—Ensures that static IPs are uniquely allocated, preventing network issues caused by IP conflicts.
- Improves Network Reliability—Enhances the network's overall reliability by detecting and handling duplicate IP allocations.
- Automates Error Handling—Automatically terminates sessions with duplicate IPs, reducing the need for manual intervention.

How it Works

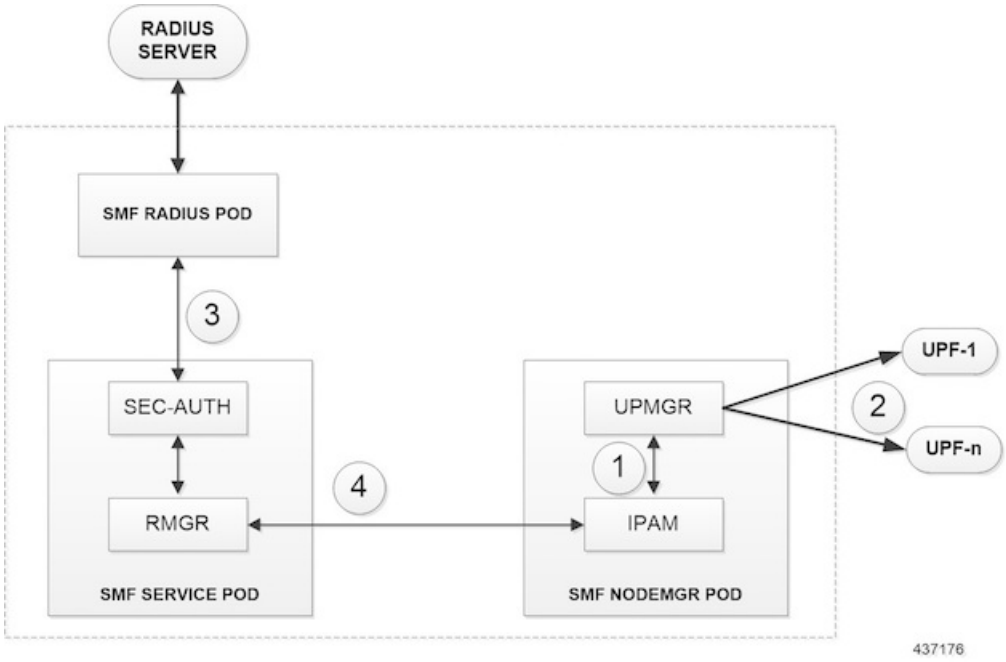
This section provides a brief of how the Static IP Support feature works.

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 5: 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 6: 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 7: 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

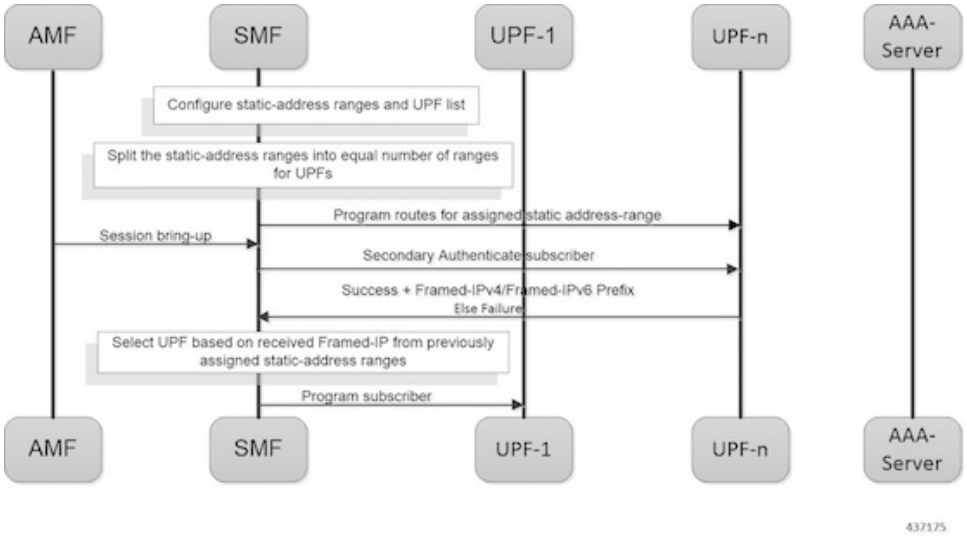


Table 8: Static IP Call Flow Description

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:

- When the system is running,
  - change of a pool from dynamic to static, and from static to dynamic is not supported.
  - addition or removal of UPF is not supported.
  - the DNN cannot be removed from a UPF.
- SMF does not support IPv6 addresses with the prefix length appended.
- 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 two UPFs and 1024 addresses specified in the range, then specify the per-dp-split-size as 512.

  If there are three UPFs and 1024 addresses, then specify the per-dp-split-size as 256.
- Changing dual-stack IPAM pool to single-stack or changing single-stack IPAM pool to dual-stack is not supported.
- SMF cannot
  - detect duplicate IP allocation to different UEs hosted on different SMFs due to misconfiguration.
  - handle scenarios where DHCP IPAM is enabled, and duplicate IPs are allocated.

Configuration for Static IP

Configure Static IP

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.

Configure Duplicate Static IP Detection

Follow these steps to detect and resolve the duplicate static IP addresses allocated by UDM and AAA server.

Before you begin

- Associate the DNN profiles with event management policies. For more details, see the Configuration for Associating QoS Profile and Event Management Policy to the DNN Profile section.

Procedure

Step 1 Configure Event Management Policy, on page 34

Step 2 Define Rules and Condition for Handling Session Termination, on page 34

Step 3 Configure Termination Action in Action Definition Policies, on page 35

Configure Event Management Policy

Follow these steps to configure the priority-based event handling. This configuration sets up a policy for managing events related to a DNN profile.

Procedure

Step 1 Enter the event management policy configuration mode.

policy eventnmgmt *policy_eventmgmt_name*

Example:

```
[smf] smf# config
[smf] smf(config)# policy eventmgmt emp
```

Step 2 Define the priority of the event management policy and then configure the event type as new call, rule, and action name to be executed.

priority *priority_number* **event new-call ruledef** *ruledef_name* **actiondef** *actiondef_name*

Example:

```
[smf] smf(config-eventmgmt-emp)# priority 2 event new-call ruledef rd1 actiondef ad1
```

The valid priority range is from 1 to 65535. Both *ruledef_name* and *actiondef_name* are alphanumeric strings that can be between 1 and 63 characters long.

Step 3 Save and commit the configuration.

Example:

```
[smf] smf(config-eventmgmt-emp))# end
```

Define Rules and Condition for Handling Session Termination

Follow these steps to define rules and condition for handling session termination.

Procedure

Step 1 Enter the rule definition policy configuration mode.

policy rulemgmt *policy_rulemgmt_name*

Example:

```
[smf] smf# config
[smf] smf(config)# policy rulemgmt rm1
```

Step 2 Specify the name of the ruledef to add to the policy.

ruledef *ruledef_name*

Example:

```
[smf] smf(config-rulemgmt-rm1)# ruledef rd1
```

Step 3 Define the condition as **duplicate-ip is true ip-type is static**.

Example:

```
[smf] smf(config-ruledef-rd1)# condition duplicate-ip is true ip-type is static
```

Configuring this condition allows the SMF to detect duplicate static IP addresses during the reattach scenario.

Step 4 Save and commit the configuration.

Example:

```
[smf] smf(config-ruledef-rd1))# end
```

Configure Termination Action in Action Definition Policies

Follow these steps to configure session termination action for new calls with duplicate static IP addresses.

Procedure

Step 1 Enter the action management policy configuration mode.

policy actionmgmt *policy_actionmgmt_name*

Example:

```
[smf] smf# config
[smf] smf(config)# policy actionmgmt act1
```

Step 2 Specify the action definition policy.

actiondef *actiondef_name*

Example:

```
[smf] smf(config-actionmgmt-act1)# actiondef ad1
```

Step 3 Specify the priority in which the actions are to be executed. Then, configure the action to terminate the session.

priority *priority_number* **action terminate-session**

Example:

```
[smf] smf(config-actiondef-ad1)# priority 1 action terminate-session
```

Step 4 Save and commit the configuration.

Example:

```
[smf] smf(config-actiondef-ad1)# end
```

Step 5 [Optional] Use **show running-config policy actionmgmt** command to verify if the PDU session termination is enabled for new calls with duplicate static IP addresses.

Example:

```
[[smf] smf# show running-config policy actionmgmt act1
Thu Jun 20 18:50:12.888 UTC+00:00
policy actionmgmt act1
actiondef ad1
priority 1 action terminate-session
```

The text in **bold** indicates that session termination is enabled on the SMF.

Sample Configuration for Duplicate Static IP Detection

This section provides the sample configuration to enable duplicate static IP detection.

```
profile dnn internet
    eventmgmt-policy em1
exit

policy eventmgmt em1
   priority 2 event new-call ruledef rd1 actiondef ad1
exit

policy rulemgmt rm1
    ruledef rd1
        condition duplicate-ip is true ip-type is static
        exit
exit

policy actionmgmt am1
    actiondef ad1
        priority 1 action terminate-session
    exit
exit
```

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.

Use these statistics for monitoring sessions disconnected due to duplicate static IP allocations:

- disc_pdusetup_static_duplicate_ip—The total number of 5G sessions that are rejected by SMF due to detection of duplicate static IP.
- disc_pdnsetup_static_duplicate_ip—The total number of 4G or Wi-Fi sessions that are rejected by SMF due to detection of duplicate static IP.

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 9: 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.

Error or Exception	Exception Handling
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

SMF supports 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.

How it Works

Prerequisite

It is essential to make pool or IPv4/IPv6 address-range offline and also ensure that their address ranges have zero utilization before deleting them from the configuration.

Following is the process 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. Use following **show ipam** commands to confirm that all address-ranges/chunks corresponding to the deleted pool or address-ranges are not showing any utilizations:
   - **show ipam pool** *pool-name*
   - **show ipam pool** *pool-name* **ipv4-addr | ipv6-prefix**
   - **show ipam dp** *dp-key* **ipv4-addr | ipv6-prefix**



Note If the subscribers are all cleared for the matching pool or address-range to be deleted, but the corresponding chunks are showing utilization in the **show ipam** commands, then use the IPAM reconciliation to trigger release of such stale IPs. The stale IPs would be released after expiring the quarantine period only.

For more details on IPAM reconciliation, see the section *IPAM Data Reconciliation*.

5. 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 instance instance_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
```

Failure handling for IPAM chunk operations

The Feature History Table provides details about the IPAM chunk operation failure handling feature.

Table 10: Feature History Table

Feature Name	Release	Feature Description
Failure handling for IPAM chunk operations	2025.03.0	IP Address Management (IPAM) failure handling prevents IP mismatch between SMF and UPF, which reduces call failures and improves operational efficiency. This feature enables IPAM to trigger failure handling when SMF fails to notify or apply chunk operations on UPF. The nodemgr retries the update request based on the failure handling template (FHT). If the update fails after retries, the nodemgr notifies IPAM, which then initiates the cleanup process and places the chunk in quarantine. When the quarantine period ends, IPAM triggers nodemgr to initiate Sx Association Update to release those chunks towards UPF and then releases the chunk internally. The failure state can be viewed using a show ipam command. Commands introduced: **failure-handling route-updates** *deactivate* and **no failure-handling route-updates deactivate** in the ipam configuration. Commands enhanced: • **N4AssociationUpdateReq** message type in profile failure-handling interface pfcp message type configuration. • *assoc-setup* and *route-update* conditions and *continue*, *retry-ignore*, and *retry-continue* actions for **N4AssociationUpdateReq** message type. • **I(Chunk state Init)** and **F(Chunk state Failed)** flags are displayed in the output of show ipam dp command. **Default Setting**: Enabled

Key features of IPAM failure handling

The failure handling for IPAM chunk operations feature enables IPAM with these features:

- Failure notifications—IPAM receives alerts from the node manager and UPF about failures in sending or applying route update messages.

- Corrective actions—IPAM processes these failure notifications and takes appropriate actions on the IP chunks allocated to the affected UPF.
- Data consistency—maintaining consistency in chunk information between SMF and UPF in the event of failures.

How IPAM failure handling works

Workflow

These stages describe how IPAM failure handling works:

1. SMF initiates the Sx Association update procedure after a successful exchange of Sx Association Setup Request and Response between SMF and UPF.
2. As part of initial Sx Association update procedure, nodemgr triggers IPAM to allocate the IP chunks for the UPF profile.
3. If the Sx Association update Request fails, nodemgr triggers release of IP chunks towards IPAM based on the FHT. It also triggers the Sx Association release towards UPF.
4. For any subsequent chunk allocation towards UPF, IPAM allocates the chunks and triggers nodemgr to update the UPF.
5. If the chunk allocation or release operation from IPAM fails to update the UPF, nodemgr checks the FHT based on the cause of failure. If the max-retry value is configured for N4AssociationUpdateReq in the FHT, the update request is retried till it receives a success response or retry counter is exhausted.



Note While retrying the request for the respective cause code, If a response is received with a different cause code, then the FHT that corresponds to the new cause code is applied, and the max-retry value is updated to the configured value of the respective cause code. Nodemgr sends the chunk update status to IPAM.

6. If the nodemgr is unable to update the chunk operation in the UPF even after retrying the messages for the configured number of times, then the nodemgr notifies that failure to IPAM.
7. On receiving the failure notification from nodemgr, IPAM initiates the cleanup process and places the chunk in quarantine. When the quarantine period ends, IPAM triggers nodemgr to initiate Sx Association Update to release those chunks towards UPF and then releases the chunk internally.



Note As the quarantine map expiry timer runs every minute, the chunks will remain in QT map for 1 to 2 minutes. The quarantine expiry for failed chunks is 1 minute. This timer differs from the IP quarantine timer configured on the address pool. The chunk operation failure handling functionality is not applicable for static address pools.

Configure failure handling in IPAM

The failure handling configuration in IPAM is enabled by default. Use this procedure to deactivate it.

Procedure

Step 1 Enter the instance ID in IPAM configuration mode.

ipam instance ID

Example:

```
[smf] smf(config)# ipam instance 1
```

Step 2 Deactivate the failure-handling route-updates in IPAM and save the configuration.

failure-handling route-updates *deactivate*

Example:

```
[smf] smf(config-instance-1)# failure-handling route-updates deactivate
```

Step 3 Use **no failure-handling route-updates deactivate** to activate the failure-handling route-updates in IPAM explicitly.

Step 4 [Optional] Use **show running-config ipam instance 1 failure-handling route-updates** to verify if the failure handling is deactivated.

Example:

```
[smf] smf# show running-config ipam instance 1 failure-handling route-updates
ipam
 instance 1
  failure-handling route-updates deactivate
 exit
exit
```

Display the state of IP chunks operation

You can view the state of IP chunk operation using a **show ipam dp-tag** <*dp_name*> <*addr-type*> command.

The I and F flags in the output of this command indicate the states of IP chunk operations between SMF and UPF.

```
[smf] smf# show ipam dp-tag 10.1.9.190:10.1.12.152::dnn@fwa5g ipv4-addr
Fri May  23 12:20:39.146 UTC+00:00
================================================================================
Flag  Indication: S(Static) O(Offline) R(Remote Instance) RF(Route Sync Failed) I(Chunk
state Init) F(Chunk state 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
================================================================================
17.2.2.0              17.2.3.255      17.2.2.0/23        1:1/0       0.00%         F
    poolv4DNN10(ISP)QT*
17.2.4.0              17.2.5.255      17.2.4.0/23        1:0/1       0.00%         I
     poolv4DNN10(ISP)
```

Table 11: States of IP chunk operation

Flag	Description
I (Chunk state Init)	The chunk is in the init state. SMF waits for confirmation from UPF on whether the route update is successful or failed.
F (Chunk state Failed)	The chunk is in a failure state. SMF receives a failure response from UPF and serves in quarantine. A chunk that is in quarantine state is tagged with **QT***.
No flag	The chunk receives a success response from UPF and is in a usable state.

Statistics for IPAM failure handling

This section describes the statistics for monitoring the route update status on IPAM.

Use the **IPAM_Route_Update_Statistics** metric to monitor the route update status on IPAM. This metric is enabled at production level. You can disable the metric by setting the verbosity level to off.

The **IPAM_Route_Update_Statistics** metrics include these labels:

- **grInstId**: Gr instance ID
- **addressType**: Address type, such as IPv4 or IPv6.
- **upf**: IPAM dp-key for the UPF.
- **operationType**: Operation for the route update such as request, response, or timeout.
- **chunkState**: Status for the route-update such as success or failure.
- **routeUpdateTrigger**: Internal IPAM operation during which this route update was triggered.

IP allocation through DHCP server

Table 12: Feature history

Feature name	Release information	Feature description
External DHCP-based IP Address Allocation	2025.02.0	The enterprises with large and complex 5G networks require centralized IP address allocation and management. This feature helps achieve centralization of IPAM by allowing the SMF to allocate IPs through a DHCP server. **Commands Introduced:** • **ipam dhcp**: This CLI is configured under the DNN profile to indicate the IPAM as DHCP. • **dhcp**: This CLI is configured under the IPAM profile to tag the address pool as DHCP. • **vlan** *vlan_id*: This CLI is configured under the IPv4 and IPv6 modes to attach the VLAN IDs. **Default Settings:** Disabled—Configuration Required to Enable

In a 5G network, SMF handles the dynamic IP address allocation from the local IPAM pool. However, the complex 5G enterprise networks require centralization of IP address allocation and management.

To facilitate this, SMF supports using a DHCP server to allocate the IP addresses to the subscriber UEs. It allows the network operator to have a centralized IP management that aligns with the needs of complex and large-scale enterprise environments.

How DHCP-based IP allocation works

Pre-requisite: For this process to work, the UPF must support DHCP client or DHCP proxy.

If the DHCP is enabled for the DNN, SMF requests UPF to allocate the UE IP address during PDU Session Establishment. UPF interacts with external DHCP server(s) to get an IP address and sends the allocated IP address in the N4 Session Establishment Response.

The DHCP servers depend upon the VLAN ID to identify the pool to allocate the IP address. SMF sends the VLAN ID to UPF in UE IP Address Pool Identity IE in the N4 Session Establishment Request message.



Note This feature requires 1:1 mapping between vDNN, VLAN (one each for IPv4 and IPv6), and UPF.

These are the detailed steps of DHCP server-based IP allocation:

1. During session creation, SMF triggers Radius Authentication Request toward the AAA (Authentication, Authorization, Accounting) server based on configuration.

   The AAA server may respond back with

- VLAN ID with which the DHCP message has to be sent,
- Virtual DNN.

If secondary authentication is not configured or if the AAA server does not return VLAN ID or the VDNN, SMF picks the VLAN ID from the local configuration.

2. The SMF sends N7 SM Policy Create Request to PCF without UE IP address. Similarly, the SMF sends the initial N40 Charging Data Request to the CHF without including the UE IP address in the message.
3. SMF requests UPF to allocate the UE IP address, if DHCP is enabled for the DNN. SMF includes the VLAN ID in the N4 Session Establishment Request message.

   To know more about the criteria used for selecting VLAN ID, see the VLAN Selection Criteria topic.

Note

- In case the VLAN ID is neither configured locally nor received from the AAA server, SMF sends the N4 Session Establishment Request without UE IP Address Pool Identity IE. In this case, the UPF rejects the request.
- The DHCPv6 is not supported by the DHCP client in UPF in this release.

4. The DHCP client in UPF goes to DHCP server to obtain the IP address and sends it to the SMF in N4 Session Establishment Response, upon receiving the N4 Session Establishment Request from SMF.
5. SMF sends N7 SM Policy Update Request including the IP address to PCF after receiving the N4 Session Establishment Response. SMF sends the N40 Charging Data Request Update message to CHF to update the IP address.

   Radius Accounting Request Start is sent to AAA server at this stage.

Note The Radius Accounting Request START is deferred and done after N4 Establishment.

6. SMF sends this IP address in create response message to the UE in case of v4 PDN and in router advertisement to the UE in case of v6 PDN. Any failure in retrieving the IP address results in SMF rejecting the Create Request by default.

VLAN selection criteria

SMF sends VLAN ID over N4 interface to UPF to indicate the VLAN with which the DHCP message has to be sent. SMF can receive the VLAN ID from AAA server, or it can be configured locally in IP Pool.

If the VLAN ID is available from both AAA server and local configuration, SMF selects the VLAN ID received from the AAA server. The VLAN ID configured in the local IP Pool configuration should match with that from the AAA server in such cases.

Note The VLAN selection criteria assumes that the configuration to select AAA server vDNN is enabled.

vDNN from AAA Server	VLAN ID from AAA Server	VLAN ID from IP Pool Configuration	Selected VLAN ID for N4	vDNN Used for IP Pool Selection
Yes	Yes	Yes	VLAN ID from AAA Server	vDNN from AAA Server
Yes	Yes	No	VLAN ID from AAA Server	vDNN from AAA Server
Yes	No	Yes	VLAN ID from IP Pool	vDNN from AAA Server
Yes	No	No	If VLAN ID is neither configured locally nor received from the Radius server, SMF sends the N4 Session Establishment Request without UE IP Address Pool Identity IE.	vDNN from AAA Server
No	Yes	Yes	VLAN ID from AAA Server	vDNN from **dnn rmgr** configuration
No	Yes	No	VLAN ID from AAA Server	vDNN from **dnn rmgr** configuration
No	No	Yes	VLAN ID from IP Pool	vDNN from **dnn rmgr** configuration
No	No	No	If VLAN ID is neither configured locally nor received from the Radius server, SMF sends the N4 Session Establishment Request without UE IP Address Pool Identity IE.	vDNN from **dnn rmgr** configuration

Failure scenarios during DHCP-based IP allocation

While allocating the IPs from the external DHCP, the errors can occur during:

- IP allocation
- DHCP renewal

Failures during IP allocation

When UPF connects with the DHCP server for IP allocation, it may experience these failures.

- **DHCP Server Not Responding:** In case the DHCP server does not respond, UPF sends a new proprietary cause **211: “DHCP Timeout”** to SMF.
- **IP Allocation Failure from DHCP Server:** In case there is a failure on DHCP server in allocating the IP address, UPF returns **Cause 79: “All dynamic addresses are occupied"** to SMF. When both IPv4 and IPv6 IP addresses are requested, this cause should be returned only when both IPv4 and IPv6 allocations fail.

  In case of IP allocation failure due to any other issue, UPF sends **Cause 213: PFCP_CAUSE_IP_ALLOCATION_FAILURE** in the N4 Establishment Response.

- **Partial IP Allocation Failure for An IPv4v6 Session:** The DHCP server may fail to allocate the IP partially, that is failure in allocating either IPv4 or IPv6 in a IPv4v6 session. In this case, the UPF sends a new proprietary cause **212: "Dynamic Address Allocation Partially Accepted"** to SMF.

Failures during DHCP renewal

Upon DHCP renewal or rebind failure, UPF-initiated session release is triggered. SMF then, releases the session.

Note

- EPFAR configuration is mandatory to handle DHCP renewal failure. If not configured, UPF will do a local purge leading to a mismatch between SMF and UPF.
- In case of IPv4v6 session, the session is released even if the DHCP renewal fails for one of the IP addresses, that is, IPv4 or IPv6.

DHCP fallback and retry

In case of failure in allocating IP due to issues such as IP exhaust, the N4 failure handling can be configured to retry.

Know more about configure DHCP fallback in the Enabling UPF Fallback topic.

Note

- Currently only one fallback DNN is supported and hence retry is done only once.
- The failure handling template needs to be configured with appropriate N4 cause codes for fallback or retry to work.

Limitations

These are the known limitations of this feature:

- When DHCP is enabled, the Session hold timer for IP/ID reuse is not applicable. These two CLIs are restricted so that only of these features can be enabled.
- When DHCP is enabled, Prefix Delegation feature is ignored even if configured.
- When DHCP is enabled, N7 Policy Optimization feature is ignored even if configured.
- When DHCP is enabled, SMF ignores static IP address received from UDM or Radius Server.
- DHCP feature is not supported for roaming subscribers except visitor-LBO. This behaviour is undefined in case of misconfiguration.
- With this feature, the N40 Charging Data Request (Update) message is sent to CHF to update the UE IP address, even if there is no usage reports or charging triggers.
- This feature is not supported with Gx or Gz interfaces.
- NextHopIP in the IPAM pool configuration is not applicable to DHCP.

- For IPAM pools configured as DHCP, the IPAM features like chunk reclamation, audits, and so on, are not applicable.
- The P5G feature for UE to VLAN mapping on N6 interface is not applicable for DHCP sessions. Since Radius sends a single vlan, both features cannot be enabled at the same time.

Enable DHCP-based IP allocation

Configuring DHCP involves these steps:

Procedure

Step 1 Enable DHCP under IPAM profile

Step 2 Configure the address pool and VLAN

Enable DHCP under DNN profile

Follow these steps to configure DHCP under the DNN profile:

Procedure

Step 1 Create an instance of DNN profile using the command **profile dnn** *dnn_profile_name*.

Example:

```
[smf] smf# config
[smf] smf(config)# profile dnn dnnprof
[smf] smf(config-dnn-dnnprof)#
```

Step 2 Indicate IPAM as DHCP using the command **ipam dhcp**.

Example:

```
[smf] smf(config-dnn-dnnprof)# ipam dhcp
[smf] smf(config-dnn-dnnprof)#
```

Note
It is mandatory to have one to one mapping between IP pool and UPF for DHCP-based IPAM.

Step 3 Exit the DNN profile using the command **end**.

Example:

```
[smf] smf(config-dnn-dnnprof)# end
[smf] smf(config)#
```

The DHCP is enabled under DNN profile.

What to do next

To configure DHCP to enable centralization of IP allocation, tag the address pool as DHCP and configure VLAN ID.

Configure the address pool and VLAN

This task helps you tag the address pool as DHCP and configure the VLAN.

Before you begin

Before tagging the address pool as DHCP and configuring VLAN, enable DHCP under DNN profile.

Procedure

Step 1 Create an instance of the Instance profile using the command **instance** *gr_instance_id*.

Example:

```
[smf] smf# config
[smf] smf(config)# ipam
[smf] smf(config-ipam)# instance instance1
[smf] smf(config-instance-instance1)#
```

Step 2 Configure source as local using the commnd **source local**.

Example:

```
[smf] smf(config-instance-instance1)# source local
```

Step 3 Specify the name of the pool to enter the Address Pool configuration mode. Also, configure the address pool as DHCP, and define the DNN under which the DHCP is configured using the command **address-pool** *pool_name* **dhcp** under the defined address pool.

Example:

```
[smf] smf(config-instance-instance1)# address-pool addr1
[smf] smf(config-address-pool-addr1)# dhcp
[smf] smf(config-address-pool-addr1)# exit
```

Note
It is mandatory to configure a single pool per vDNN (either one pool with both IPv4 and IPv6, or one each for IPv4 and IPv6). In case of misconfiguration, where there are multiple IPv4 or IPv6 pools for a single vDNN, the behaviour is undefined, that is, the SMF can fetch any random pool.

Step 4 Configure the VLAN for either or both IPv4 and IPv6 addresses under the IPv4 and IPv6 command modes using the command **ipv4 vlan** *vlan_id* and **ipv6 vlan** *vlan_id*.

Example:

For IPv4 addresses:

```
[smf] smf(config-address-pool-addr1)# ipv4
[smf] smf(config-ipv4)# vlan [1001]
```

Example:

For IPv6 addresses:

```
[smf] smf(config-address-pool-addr1)# ipv6
[smf] smf(config-ipv6)# vlan [1001]
```

Note
Cofiguring VLAN under the IP pool is required, in case the AAA server does not provide the VLAN ID.

Step 5 Configure the split-size for either or both of IPv4 and IPv6 addresses under the configured VLAN using the command **{ split-size [no-split]** .

Example:

For IPv4 addresses:

```
[smf] smf(config-ipv4)# split-size no-split
[smf] smf(config-ipv4)# exit
```

For IPv6 Addresses:

```
[smf] smf(config-ipv6)# prefix ranges split-size no-split
[smf] smf(config-ipv6)# exit
```

Note

- If the address or prefix range are configured, the **split-size** must be configured as **no-split**. If the address or prefix range are not configured under IP pool, SMF does not advertise routes to UPF.
- Configuring IP address or prefix-range is optional and is needed, if SMF has to advertise routes to UPF. A maximum of two address-ranges can be configured per pool or vDNN marked as dhcp. The routes or chunks would not be distributed beyond this.

Step 6 Use the command **exit** to save and exit the current command mode.

Example:

```
[smf] smf(config-ipv6)# exit
[smf] smf(config-address-pool-addr1)#
```

Step 7 Verify the configuration using the command **show running-config ipam instance** *gr_instance_id* **address-pool** *pool_name*.

Example:

```
[smf] smf# show running-config ipam instance instance1 address-pool addr1
ipam
 instance 1
  address-pool addr1
   vrf-name ISP
   dhcp
   tags
    dnn dnnprof
   exit
   ipv4
    vlan [ 1001 ]
    split-size
     no-split
    exit
   ipv6
    vlan [ 1001 ]
    prefix-ranges
     split-size
      no-split
     exit
    exit
   exit
```

```
  exit
 exit
exit
```

This configuration tags the address pool as DHCP configures the VLAN ID to receive IP address from the DHCP server.

Monitoring and troubleshooting

This topic discusses the statistics and metrics used for monitoring and troubleshooting this feature.

Bulkstats

These existing statistics are enhanced to support this feature:

- The statistics **smf_disconnect_stats** is enhanced with new failure reasons:
  - **5G call disconnected due to corresponding N4 failures: disc_pdusetup_dhcp_timeout**, **disc_pdusetup_dhcp_ip_alloc_failure** , and **disc_pdusetup_up_ip_alloc_failure** to indicate 5G call is disconnected due to corresponding N4 failures.
  - **4G call disconnected due to corresponding N4 failures: disc_pdnsetup_dhcp_timeout**, **disc_pdnsetup_dhcp_ip_alloc_failure**, and **disc_pdnsetup_up_ip_alloc_failure** to indicate 4G call is disconnected due to corresponding N4 failures.
  - **5G or 4G call disconnect due to DHCP renewal failure**: **disc_pdurel_dhcp_renewal_failure** and **disc_pdnrel_dhcp_renewal_failure** to indicate 5G or 4G call is disconnected due to DHCP renewal failure.
- Existing SMF service stat **smf_service_udp_req_msg_total** is pegged with the new N4 failure cause codes, such as **Cause 211: DHCP Timeout**, **Cause 79: All dynamic addresses are occupied**, or **Cause 212: Dynamic Address Allocation Partially Accepted** upon receiving N4 Session Establishment Response with these failures.
- The Session type label in existing procedure stats is enhanced to support new value as **dhcp**.
- The existing statistics categoty **smf_service_node_mgr_stats** consisting of the label **ip_req_type** is enhanced with two new values:
  - ip-dhcp-id-alloc
  - ip-dhcp-id-dealloc

  These stats are pegged when the SMF service sends request to the node manager for ID allocation or ID deallocation for DHCP sessions.



Note The statistics **smf_service_resource_mgmt_stats** is not applicable for DHCP sessions.

show subscriber

These **show subscriber** commands are extended to handle the DHCP scenarios:

- **show subscriber nf-service smf dhcpv4-addr { ipv4 | all }**
- **show subscriber nf-service smf dhcpv6-addr { ipv4 | all }**
- **show subscriber nf-service smf dhcp all**
- **show subscriber nf-service smf dhcpv4-vlan { vlan-id | all }**
- **show subscriber nf-service smf dhcpv6-vlan { vlan-id | all }**
- **show subscriber nf-service smf count [dhcpv4-addr | dhcpv6-addr | dhcp] [ip | all]**
- **show subscriber nf-service smf count [dhcpv4-vlan | dhcpv6-vlan] [vlan-id]**

These non-unique keys are added for DHCP sessions:

- dhcp:v4-$ipv4/dhcpv4:vlan-$vlanId
- dhcp:v6-$ipv6/dhcpv6:vlan-$vlanId

Note Existing sessions keys “**ipv4-addr:**", "**ipv4-pool:**", "**ipv6-pfx:**" & "**ipv6-pool:**" are not applicable for DHCP sessions.

show subscriber dhcpv4-addr all displays all the DHCPv4 subscriber status.

```
[smf] smf# show subscriber dhcpv4-addr all
subscriber-details
{
  "subResponses": [
    [
      "roaming-status:homer",
      "ue-type:nr-capable",
      "supi:imsi-123456789012345",
      "gpsi:msisdn-223310101010101",
      "pei:imei-123456786666660",
      "psid:5",
      "snssai:002abf123",
      "dnn:intershat",
      "emergency:false",
      "rat:e-utran",
      "access:3gpp access",
      "connectivity:4g",
      "udm-sdm:10.1.11.18",
      "auth-status:authenticated",
      "pcfGroupId:PCF-dnn=intershat_dhcp;",
      "policy:2",
      "pcf:10.1.11.18",
      "rulebase:rba_charging_StaticDynamic_online",
      "upf:10.1.11.18",
      "upfEpKey:10.1.11.18:10.1.13.155",
      "id-index:1:0:32768",
      "id-index-key:1:0:0:32768",
      "id-value:0/1",
      "chfGroupId:CHF-dnn=intershat_dhcp;",
      "chf:10.1.11.18",
```

```
      "data-tunnel:IPV4",
      "dhcp:v4-12.0.5.1/dhcpv4-vlan-2011",
      "gtp-peer:10.1.11.18",
      "wps:false",
      "peerGtpuEpKey:10.1.11.18:10.1.11.18",
      "namespace:smf",
      "nf-service:smf"
    ]
  ]
}
```

clear subscriber nf-service smf

These **clear subscriber nf-service smf** commands are enhanced to clear the DHCP subscribers:

- **clear subscriber nf-service smf dhcpv4-addr { ipv4 | all }**
- **clear subscriber nf-service smf dhcpv6-addr { ipv6 | all }**
- **clear subscriber nf-service smf dhcp all**
- **clear subscriber nf-service smf dhcpv4-vlan { vlan-id | all }**
- **clear subscriber nf-service smf dhcpv6-vlan { vlan-id | all }**

IPv6 Prefix Delegation

Table 13: Feature History

Feature Name	Release Information	Description
IPv6 Prefix Delegation from Local Pool	2024.04.0	The SMF supports IPv6 Prefix Delegation, allowing User Equipment (UE) and Customer Premises Equipment (CPE) to request additional IPv6 prefixes for configuring routers or cellular gateway devices. The delegated prefix length range allowed is from 48 to 62. This IPv6 prefix delegation is especially beneficial for service providers as it helps manage subscriber networks more effectively by assigning a prefix to CPE devices acting as routers between the subscriber's internal network and the core network **Command Introduced**: Following command is introduced under Profile DNN: **ipv6-prefix-delegation prefix length** *prefix_length* **dnn prefix-delegation** *vdnn* **Default Setting**: Disabled – Configuration Required to Enable

When setting up your network through SMF, you have the flexibility to delegate prefixes to your routers. SMF assigns IPv6 address range to routers or cellular gateway devices using 3GPP defined procedures and use prefix delegation to assign IP addresses to many devices behind the router.

The IPv6 Prefix Delegation feature allows User Equipment (UE) and Customer Premises Equipment (CPE) to request additional IPv6 prefixes for configuring routers or cellular gateway devices.

These routers or cellular gateways should create PDU session with unique APN/DNN towards 4g/5G network.

Key Functionalities of IPv6 Prefix Delegation

The following are the functionalities of IPv6 Prefix Delegation feature:

- **Single Prefix Length per DNN**: SMF supports only a single prefix length per Data Network Name (DNN). The prefix length range allowed is between 48 and 62.

- **Delegated Prefix from Local Pool (IPAM)**: SMF fetches the delegated prefix from the local pool (IP Address Management).
- **Allocation of Delegated prefix**: SMF allocates the delegated prefix as per the DNN configuration, and sends it to UPF.

You can verify and clear sessions with delegated prefixes using the following commands:

- Show subscriber commands:
  - show subscriber nf-service smf delg-prefix { starts-with dp }
  - show subscriber nf-service smf delg-prefix { starts-with dp-$prefixLength }
  - show subscriber count nf-service smf delg-prefix { starts-with dp-$prefixLength }
- Clear subscriber commands:
  - clear subscriber nf-service smf delg-prefix { starts-with dp }
  - clear subscriber nf-service smf delg-prefix { starts-with dp-$prefixLength }



Note The value provided next to the **starts-with** in the preceding commands is matched with the delegated prefix session key **dp:dp-$prefixLength/$ipPool/$delegatedPrefix** and includes all sessions with **dp:<starts-with-val>**.

For example, if the sessions are present with prefix-length 61 and 62, **{starts-with dp-6}** includes all the sessions with key **dp:dp-61* & dp:dp-62***.

Prerequisite for IPv6 Prefix Delegation in SMF

The support for prefix delegation is required in UPF.



Note With older UPF versions that doesn't support IPv6 Delegation feature, the implementation changes cause a decode error. Hence, it is mandatory to upgrade the UPF before enabling this feature in SMF.

How IPv6 Prefix Delegation Works

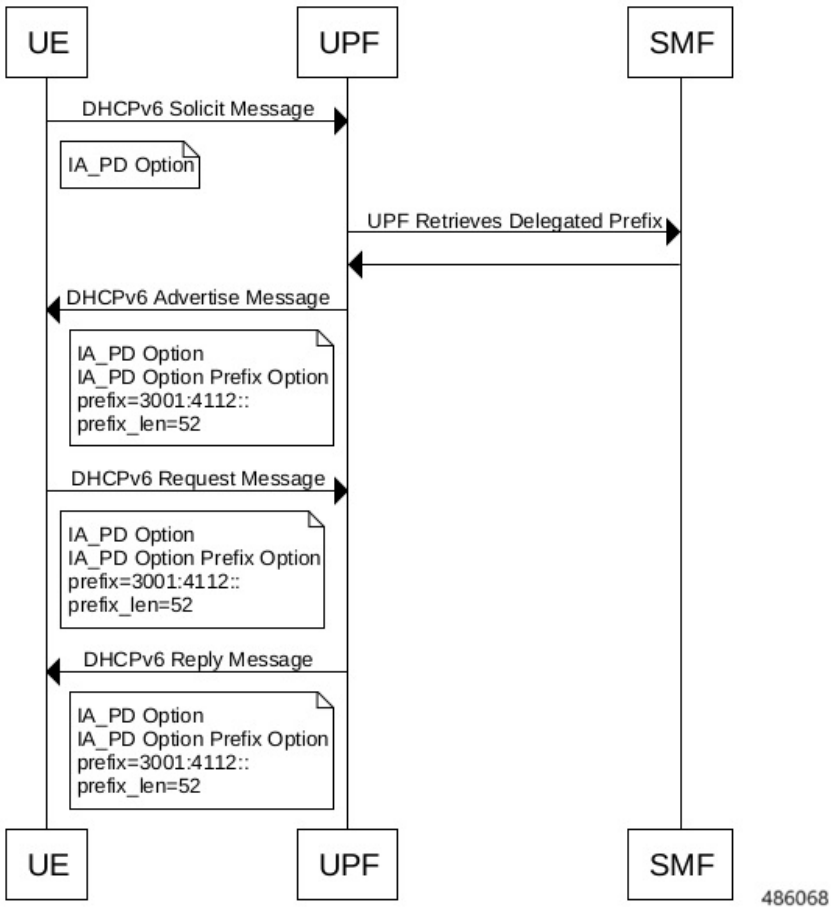
These are the stages in the IPv6 Prefix delegation:

- Configure Profile DNN and Prefix length to enable the prefix delegation from the local pool. For configuration steps, see the Configure Profile DNN and Configure Prefix Length sections.

Allocation of IPv6 Prefix Delegation

This section describes the call flow and stages in allocation of IPv6 prefix delegation.

Figure 4: Allocation of IPv6 Prefix



The following stages describe interactions between SMF (Session Management Function) and UPF (User Plane Function) for allocating and managing delegated prefixes.

1. The UE/CPE sends DHCPv6 Solicit messages to the UPF. The UE/CPE can include a prefix length in IA_PD Prefix option.
2. UPF receives the requested messages over N3 or S1U/S5U interface on the GTPU tunnel that was created when the PDN session for the UE was established. Since UPF does not manage IP pool locally, UPF acts as a proxy DHCP server and receives the prefix from SMF. SMF provides delegated prefix as per the DNN configuration. SMF rejects the request, if DHCP solicit is received with a prefix length different from the configured prefix length value. If prefix length is not received, SMF allocates as per the configured prefix length.
3. UPF sends response to the DHCPv6 Solicit message with a DHCP advertise message containing the delegated prefix in the IA_PD option.

   SMF allocates delegated prefix from the local IPAM pool.

Configure Profile DNN

Use the following procedure to configure the DNN profile for enabling Prefix Delegation from the local pool:

Procedure

Step 1 Log in to the Global Configuration mode.

Step 2 Enter the **profile dnn** *dnn_profile_name* command. The **ipv6-prefix-delegation** enables Prefix Delegation feature for this DNN. A separate DNN profile is mandatory for Prefix Delegation. Only one prefix length per DNN is supported for prefix delegation.

Example:

```
config
      profile dnn dnn_profile_name
              ipv6-prefix-delegation prefix length prefix_length
              dnn prefix-delegation vdnn
      exit
```

Step 3 Enter the **ipv6-prefix-delegation prefix length** *prefix_ length* command. This indicates the delegated-prefix length supported for this dnn and suppored range is 48 to 62. When UE sends DHCP Solicit message without the requested prefix-length, SMF uses this as a prefix length.

Example:

```
config
      profile dnn dnn_profile_name
              ipv6-prefix-delegation prefix length prefix_length
              dnn prefix-delegation vdnn
      exit
```

Step 4 Enter the **dnn prefix-delegation** *vdnn* command to configure Prefix Delegation (PD) vDNN. This configuration is mandatory since separate IP Pools are needed for delegated prefix.

Example:

```
config
      profile dnn dnn_profile_name
              ipv6-prefix-delegation prefix length prefix_length
              dnn prefix-delegation vdnn
      exit
```

Note
PD vDNN is associated to the dedicated UPF. If it is not associated, then the behavior is undefined. PD DNN is not used for UPF Selection; the expectation is that SMF and UPF are configured properly.

It is recommended to associate dedicated UPF for PD DNN, to avoid splitting the PD IP pool into multiple chunks in case of different UPFs.

SMF does not support usage of the same UE IP Pools configure in Control Plane to assign the delegated prefixes and non-delegated prefixes.

Step 5 Enter the **exit** command to end the current configuration mode.

Configure IPAM

Use this procedure to configure the SMF IPAM for Prefix delegation.

Procedure

Step 1 Log in to the IPAM Configuration mode.

```
ipam
 instance 1
  address-pool poolv6_pfx62
   vrf-name ISP1
   tags
    dnn pd_vdnn
   exit
   ipv6
    prefix-ranges
     split-size
      per-cache 256
      per-dp    256
     exit
     prefix-length 62
     prefix-range 2607:fb91:8a00:: length 52
    exit
   exit
  exit
 exi
```

Step 2 Enter the **prefix-length** *prefix_value* command to specify the prefix length for IP allocation. Must be an integer from 2 to 64. IPs allocated from the pool will always be only with the specified prefix. The DNN tag should match with the **pd_vdnn** configured under the DNN profile configuration.

Example:

```
ipam
 instance 1
  address-pool poolv6_pfx62
   vrf-name ISP1
   tags
    dnn pd_vdnn
   exit
   ipv6
    prefix-ranges
     split-size
      per-cache 256
      per-dp    256
     exit
     prefix-length 62
     prefix-range 2607:fb91:8a00:: length 52
    exit
   exit
  exit
 exi
```

Note
It is mandatory to upgrade the UPF before enabling the IPv6 Prefix Delegation feature in the SMF to avoid decode errors in the UPF.

Note
The prefix length in IPAM should match the prefix length configured under profile DNN.

The IP pool size and configuration for PD pools must be planned considering the number of chunks required per UPF and the chunk-size restrictions.

What to do next

Verify Prefix Delegation from the Local Pool .

Verify Prefix Delegation from the Local Pool

To verify the chunk creation from the local pool, use the following **show ipam pool** command:

show ipam pool

The following output displays the chunk creation in the pool.

```
[smf] smf# show ipam pool poolv6_pfx62
-------------------------------------------------------
 Ipv4Addr   [Total/Used/Utilization] = 0 / 0 / 0.00%
 Ipv6Addr   [Total/Used/Utilization] = 0 / 0 / 0.00%
 Ipv6Prefix [Total/Used/Utilization] = 1024 / 0 / 0.00%
 Instance ID                         = 1
-------------------------------------------------------

[smf] smf# show ipam pool poolv6_pfx62 ipv6-prefix
================================================================================
Flag Indication: S(Static) O(Offline)
================================================================================
StartAddress               EndAddress               AllocContext   Flag  GroupId
================================================================================
2607:fb91:8a00::/62        2607:fb91:8a00:3fc::/62   Free:CP
2607:fb91:8a00:400::/62    2607:fb91:8a00:7fc::/62   Free:CP
2607:fb91:8a00:800::/62    2607:fb91:8a00:bfc::/62   Free:CP
2607:fb91:8a00:c00::/62    2607:fb91:8a00:ffc::/62   Free:CP
=======================
```

Metrics

This section describes the metrics that are collected and reported for the IPv6 Prefix delegation. These metrics are required to show the count of prefix delegation (per prefix length).

Following Metrics are enhanced with the following labels:

- **smf_service_resource_mgmt_stats**: This counter tracks the total number SMF Service Resource Management Stats. The existing “ip_req_type” label is enhanced with “pd_alloc” and “pd_dealloc.
- **nodemgr_resource_mgmt_resp_stats** : This counter tracks the total number Resource management response Stats in Node Manager. The existing "req_type" label enhanced with "pd_req_alloc" and "pd_req_rel".

- **smf_sess_report_stats** is enhanced for the new report types.

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 instance instance_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 73
- Enabling UPF Fallback, on page 73

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
```

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 Throttling

Feature Description

Table 14: Feature History

Feature Name	Release	Feature Description
IPv6 transport on N4 interface	2025.03.0	SMF now supports IPv6 addresses over N4 interface.
Optimize IPAM Performance and Scalability	2024.03.0	The IP chunk allocation throttle feature triggers throttling checks based on the maximum UPF sessions supported. However, throttling checks occur even if the maximum UPF sessions are not configured, causing unnecessary processing overhead. To address this, SMF introduces a new command, **chunk-throttling false**, at the gr-instance level to disable IP chunk throttling explicitly.

The SMF supports the IP chunk throttling feature to optimize IP pool utilization and manage IP address chunk allocation across multiple UPFs. This feature helps balance the load by throttling new IP chunk allocations based on UPF capacity as advertised by the UPF, or local configuration on SMF. When throttling is triggered, no new chunks are allocated, but existing chunks continue to be used until exhausted.

To determine the UPF capacity, SMF evaluates the following configurations:

- The **max-upf-sessions** command configured at the network element (UPF) level.
- The **max-upf-sessions** command configured at the vDNN level (also applicable per UPF level).
- The **load-control capacity** command configured at the user plane service level in UPF.

If all three configurations are present, the SMF compares the values and selects the lowest one to control IP chunk allocation for the UPF. However, the SMF initiates throttling checks even when the maximum number of UPF sessions is not configured in the UPF, leading to unnecessary processing overhead.

To mitigate unintended throttling checks, SMF introduces a new configuration command, **chunk-throttling false**, at the gr-instance level to explicitly disable IP chunk throttling.

How It Works

The IP chunk throttling process involves 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. Configure the maximum session capacity per UPF on SMF. This configuration can be applied 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 80% of the total IP allocated, SMF attempts to allocate additional chunks. Further allocation can also occur based on internal changes, such as pool configuration modifications or pool threshold hits.
5. SMF compares the IP usage of the UPF, which includes current IP usage by active sessions and IPs in quarantine, with the maximum session capacity of the UPF.
6. SMF throttles chunk allocation if the current usage exceeds 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.

Note The CLI **n4-peer-address** can either configure IPv4 address or IPv6 address at a time. If both are configured at the same, it may lead to unexpected behavior.

Configuration Example

Following is the sample configuration for defining the 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 the maximum supported sessions 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
```

Disable IP Chunk Throttling

Use this procedure to disable IP chunk allocation throttling.

Procedure

Step 1 Enter the IPAM configuration mode.

Example:

```
[smf] smf# config
[smf] smf(config)# ipam
```

Step 2 Specify the GR instance number. The *instance_id* ranges from 1 to 8.

instance *instance_id*

Example:

```
[smf] smf(config-ipam)# instance 1
```

Step 3 Disable IP chunk allocation throttling by setting the value to false. The default value is true.

chunk-throttling { false | true }

Example:

```
[smf] smf(config-instance-instance_id)# chunk-throttling false
```

Step 4 Save and commit the configuration.

Example:

```
[smf] smf(config-instance-instance_id)# end
```

This command lets you to either commit or ignore the configurations. Entering **yes** allows you to save or modify the configurations.

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.

IPAM Cache Data Sharding

Table 15: Feature History

Feature Name	Release	Feature Description
Optimize IPAM Performance and Scalability	2024.03.0	SMF enables you to configure IPAM data sharding with an appropriate number of shards to overcome processing and storage capacity bottlenecks. Data sharding enhances IPAM performance and scalability, enabling support for high TPS and a large number of IP pools, pool-tags, and UPFs. Command introduced: **data-sharding num-shards** *ipam_data_shards*

With the demand for DNNs, IP pools, and UPFs in network deployments growing, the IPAM cache data size increases significantly. The current IPAM system faces challenges with high transactions per second (TPS) and large-scale deployments as it operates on a single cache context structure during IP address allocation. This leads to longer response times and backpressure. Sharding the IPAM cache data addresses these issues by allowing parallel processing of IP allocation and release requests.

Sharding is the process of fragmenting the IPAM cache data into multiple parts (shards) based on a shard key, which is a combination of UPF ID, pool-tag string or pool name. Each shard operates independently and controls its own cache context fragments. Sharding can be enabled or disabled through CLI command **data-sharding** at GR instance level. The number of shards is configurable, allowing scalability based on the number of IP pools, UPFs, and address types deployed.

Data sharding helps in

- scaling the performance to support higher TPS for IPAM operations.
- managing the larger cache data more efficiently.

Limitations with Data Sharding

Consider these limitations when implementing IPAM cache data sharding.

- There is no support for migrating existing IPAM cache data when enabling or disabling sharding.
- A fresh start of SMF is required for any changes to the sharding configuration.

Enable IPAM Data Sharding

Use this procedure to enable sharding of IPAM cache data.

Procedure

Step 1 Enter the IPAM configuration mode.

Example:

```
[smf] smf# config
[smf] smf(config)# ipam
```

Step 2 Specify the GR instance number. The *instance_id* ranges from 1 to 8.

instance *instance_id*

Example:

```
[smf] smf(config-ipam)# instance 1
```

Step 3 Enable IPAM cache data sharding and specify the number of shards. The value of *shards_number* must be an integer between 2 and 16. To disable data sharding, use the **no data-sharding** command.

By default, the data sharding is disabled.

data-sharding num-shards *shards_number*

Example:

```
[smf] smf(config-instance-instance_id)#data-sharding num-shards 4
```

Important
You must restart SMF for the configuration to take effect.

Step 4 Save and commit the configuration.

Example:

```
[smf] smf(config-instance-instance_id)# end
```

This command lets you to either commit or ignore the configurations. Entering **yes** allows you to save or modify the configurations.

IP Allocation and Reuse

The initial step for IP allocation involves the UE sending a PDU Session Establishment Request to the network to obtain an IP address. SMF receives this IP allocation request from the UE. Subsequently, the network allocates the necessary resources to other network components and communicates the allocated IP address to the UE through the PDU Session Establishment Accept message.

To retain the same IP address after disconnection from the network, the SMF offers configuration options to enable the hold timer functionality for IP reuse. The hold timer duration can be set at either the converged-core profile level or the DNN level.

When the **session hold-duration** CLI command is configured, and an active subscriber is disconnected, the IP address is held and considered still in use. The IP address does not return to the free state until the session hold timer expires.

This functionality allows subscribers who reconnect within the specified duration to retain the same IP address from the IP pool, ensuring optimal reuse of IP addresses.

Table 16: Feature History

Feature Name	Release	Feature Description
Support for UE IP Address Hold and Reuse	2024.03.1	This feature allows SMF to allocate the same IP address and session ID to a user session if the user reattaches within a configurable session hold timer after session release. This feature improves the user experience and reduces the IP address exhaustion in scenarios where users frequently attach and detach. **New commands**: **session-hold duration** in Converged Core profile and DNN profile configurations

Benefits of Holding and Reusing the UE IP Address

This section lists the positive outcomes that users can experience from using this feature.

- **Optimized IP Address Management**: This feature allows for the efficient reuse of IP addresses, which is crucial for maintaining a sustainable and scalable network infrastructure.
- **Improved Session Continuity**: The Hold Timer functionality ensures that user sessions can be reattached within a specified period after network disruptions or upgrades, providing a seamless user experience.
- **Comprehensive Coverage**: The feature is designed to handle a wide range of scenarios, including user session release, reattachment within and after hold timer expiry, and various failure scenarios. This comprehensive coverage ensures that the network can handle various situations, providing a versatile solution that can adapt to different operational needs.

Guidelines for IP Reuse During Rolling Upgrade and Downgrade

Remember to check these important guidelines while performing the rolling upgrade procedure.

- Enable the **session-hold** CLI only after the rolling upgrade is complete.
- Clear the user sessions that are in hold state using **clear subscriber session-state hold** command.

Before starting the rolling downgrade, make sure you

- disable the IP reuse feature
- clear the user sessions in IP reuse state

Limitations with UE IP Address Hold and Reuse

Note these potential limitations when requesting the same IP address during a reattach scenario.

- You must reuse both the ID and IP address together, as reusing the ID alone is not supported.
- This feature neither supports static IP allocation and DHCP IP allocations, nor transitions between dynamic and static IP allocations. In such cases, the SMF will release the old IP and allocate a new one.
- For converged core sessions, the cnSGWc initially selects the UPF, and the cnPGW attempts to reuse the same UPF. To maintain UPF convergence during reattach, the cnSGW must provide the same UPF used in the previous session. If the cnSGW does not provide the same UPF, UPF session convergence cannot be maintained.
- IP reuse may not occur for new calls if the previous call was not converged. This is because the priority is given to the UPF selected by the cnSGW for the new converged call.
- If there is a transition between an unauthenticated IMSI and an authenticated IMSI (or vice versa) during an emergency call, the IP and ID will not be reused.
- IP addresses will not be reused when
  - initiating create over half-create
  - 4G-only UE performs a Wi-Fi over 4G attach or vice versa
  - 4G-only UE reattaches as a 5G-capable UE
- This feature does not work if the PDN type, slice, DNN, UPF, or vDNN mismatch between the previous and the current session.

Handling Attach or PDU Setup Failures During Hold State

In the event of an attach or PDU setup processing failure while in a hold state, the SMF executes specific actions as outlined in the table.

Hold Timer Expired During Attach or PDU Setup Processing	Attach Failure Occurs in N10 or N7 Create Processing	Attach Failure Occurs in N7 Update, N40, N11 or N4 Processing	Action
No	Yes	N/A	• Do not stop hold timer • Remain in hold state
No	N/A	Yes (IP Not reused)	• Release the IP • Send delete/release to peer NFs • Delete the session from CDL
No	N/A	Yes (IP Reused)	• Do not stop hold timer • Remain in hold state

Hold Timer Expired During Attach or PDU Setup Processing	Attach Failure Occurs in N10 or N7 Create Processing	Attach Failure Occurs in N7 Update, N40, N11 or N4 Processing	Action
Yes	Yes	N/A	• Release the IP • Send delete/release to peer NFs • Delete the session from CDL
Yes	N/A	Yes	

Criteria for IP Reuse

The following scenarios will prevent the IP reuse of a session in hold state:

1. Inconsistencies in pdnType
2. Discrepancies in Slice
3. Variations in dnn and Vdnn
4. UPF disparities originating from SGW
5. UPF mismatches
6. UPF outages
7. Activation of location-based UPF selection

These conditions will be validated during the initial processing of creation requests and on reuse failure, except for scenario 4. If reuse fails, an IP release will be triggered to the Resource Manager (R-MGR), and the UPF will be allocated after IP allocation.

The UPF address from the previous PDU setup or attach will be used as the preferred UPF, increasing the chance of the same UPF being selected for future calls.

A UPF mismatch from SGW (scenario 4) can occur if the SGW does not support this feature or if the call originates from a new SGW. This can also happen if the session was in NR or Wi-Fi before release.

For new converged calls, priority is given to the UPF selected by the cnSGW. Therefore, IP reuse may not occur if the previous call was not converged. However, if the previous call was also converged and the cnSGWc supports this feature, this scenario may not arise. If the new call is not converged, then the IP address, ID, and UPF will be reused.

How does SMF Allow UE IP Address Reuse

SMF checks if the hold timer functionality is enabled in the converged-core profile or the DNN profile, and if the hold timer value is valid. SMF then takes appropriate action based on these conditions.

Table 17: Scenarios for IP Reuse

Condition 1	Condition 2	Action
If user session exists with the same PDU session ID and same SMF instance	If user reattaches before hold timer expiry and if the reuse criteria match	SMF reuses the same IP and ID for the session.
	If the user does not reattach within the hold timer expiry	SMF releases the IP and ID and deletes the session or PDU context details.
	If the reuse criteria do not match	SMF allocates new IP and ID for the session and releases the old IP and ID.
If session do not exist with the same PDU session ID and same SMF instance		SMF allocates new IP and ID for the session.

SMF moves the session to hold state and retains the IP and ID reuse data.

Prerequisites for UE IP Address Hold and Reuse

SMF allows you to reuse the IP address only when these conditions are met.

- The user session must have the same PDU session ID for the reattach scenario. This feature relies on the mapping of the user session to the IP and ID allocation details.
- The IMSI and EBI must match for the 4G sessions during re-attach scenario.
- The same UPF must be selected for the user session. If UPF selection is based on location or determined by cnSGW, this feature may not function as intended.

Enable IP Reuse Feature

You can enable the IP reuse feature by setting the hold timer duration at converged core profile, DNN profile or at both the profiles.

Procedure

Follow the steps as outlined in these sections.

- Configure Hold Timer at Converged Core
- Configure Hold Timer at DNN

Configure Session Hold Timer at Converged Core

Use this procedure to set the session hold timer at the converged-core profile level.

Before you begin

- Review the Guidelines and Limitations sections in this document.
- Ensure that you are connected to the SMF Ops Center to run the configuration.

Procedure

Step 1 Enter the converged-core profile configuration mode.

profile converged-core *ccg_profile_name*

Example:

```
[smf] smf# config
[smf] smf(config)# profile converged-core ccg1
```

Step 2 Specify the duration, in seconds, for holding the user's IP address for reuse. The valid range for the hold timer is 30 to 300 seconds.

session-hold duration *hold_timer*

Example:

```
[smf] smf(config-converged-core-ccg1)# session-hold duration 120
```

If the session hold timer is configured both in converged core profile and DNN profile, the timer value configured in DNN overrides the timer value in converged core profile.

Note
SMF does not support dynamic change in configuration of session hold timer for ongoing sessions. Any change in the timer value will be applicable only for new sessions.

Use **no session-hold** command to disable the hold timer functionality.

Step 3 Save and commit the configuration.

Example:

```
[smf] smf(config-converged-core-ccg1)# end
```

This command lets you to either commit or ignore the configurations. Entering **yes** allows you to save or modify the configurations.

Configure Session Hold Timer at DNN

Use this task to configure the session hold timer duration at the DNN level.

Before you begin

- Review the Guidelines and Limitations sections in this document.

- Ensure that you are connected to the SMF Ops Center to run the configuration.

Procedure

Step 1 Enter the DNN profile configuration mode.

profile converged-core *dnn_profile_name*

Example:

```
[smf] smf# config
[smf] smf(config)# profile dnn dnn_test
```

Step 2 Specify the duration, in seconds, for holding the user's IP address for reuse. The valid range for the hold timer is 30 to 300 seconds.

session-hold duration *hold_timer*

Example:

```
[smf] smf(config-dnn-dnn_test)# session-hold duration 140
```

If the session hold timer is configured both in converged core profile and DNN profile, the timer value configured in DNN overrides the value in converged core profile. SMF holds the UE IP address for 140 seconds after the session release.

Note
SMF does not support dynamic change in configuration of session hold timer for ongoing sessions. Any change in the timer value will be applicable only for new sessions.

Use **no session-hold** command to disable the hold timer functionality.

Step 3 Save and commit the configuration.

Example:

```
[smf] smf(config-dnn-dnn_test)# end
```

This command lets you to either commit or ignore the configurations. Entering **yes** allows you to save or modify the configurations.

Troubleshooting Information for IP Reuse

This section provides output of show commands, clear commands, and bulk statistics that can be used to troubleshoot the issues.

Show Commands

To view the state of user sessions, use the **show subscriber** command.

```
[smf] smf# show subscriber supi imsi-123456789012345 nf-service smf full
subscriber-details
{
"subResponses": [
{
"status": true,
"genericInfo": {
```

```
"pduState": "pduState_hold",
"pduSesstype": "UnknownPduSessionType",
"alwaysOn": "None",
"dcnr": "None",
"wps": "Non-Wps Session",
"ueType": "NR Capable UE",
"authStatus": "Unauthenticated",
"ipAllocationBy": "Dynamic",
"anType": "3GPP_ACCESS",
"holdTime": "2024-08-19T18:50:53Z"
},
"policySubData": {
"supportedFeatureList": {}
},
"upfServData": {},
"access4GSubData": {}
}
]
}
```


To determine the number of user sessions in active state, use the following commands:

- **show subscriber count nf-service smf all**
- **cdl show sessions count summary slice-name 1 filter { condition not match key "1#/#sessState:Hold" }**

To determine the number of user sessions in hold state, use the following commands:

- **show subscriber count nf-service smf session-state hold**
- **cdl show sessions count summary slice-name 1 filter { condition match key "1#/#sessState:Hold" }**

Note The show commands will display the number of active and hold sessions only when this feature is enabled.

Clear Command

By default, subscriber sessions will be moved to hold state during all call clearing or release actions, such as IP pool offline, pool removal, and GTP-U or GTP-C path failure cases. In these scenarios, use the **clear subscriber** commands, except **clear subscriber upf | upf-list**, to clear the sessions in the hold state.

Subscriber sessions will be fully cleaned up without being moved to the hold state during IP chunk reclamation or UPF-related releases, such as path failure, re-association, or restart/recovery cases. In these instances, use the new CLI keyword **no-session-hold** along with the existing **clear subscriber** command options to completely clear the session.

Bulk Statistics Output

This feature introduces the following new bulk statistics:

- **smf_disconnect_session_hold_stats**: This statistic is introduced for the cleanup of calls in hold state due to hold timer expiry, and also for the admin to clear the sessions in hold state.
- **smf_ip_reuse_stats**: This statistic is introduced to determine the IP reuse count. It includes the following labels:

- reason
- rat_type

This feature introduces the following updates to the bulk statistics:

- **db_records_total**: This statistic includes a new label “session-hold” to identify if the CDL record is in hold state.
- **smf_disconnect_stats**: This statistic includes a new label “hold=true/false” to indicate if the sessions are moved to hold state.
- **smf_service_stats**: This statistic includes a new label “ipreuse=true/false” to indicate if the IP reuse feature is enabled.

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
           address-range start_ipv4_address end_ipv4_address
             chunk-group chunks-per-group { 2 | 4| 8 }
             exit
```

NOTES:

- **address-range** *start_ipv4_address end_ipv4_address*—This CLI specifies the range of IPv4 address range consisting of start address and end addresses. The address range in dotted-decimal notation.

When **chunk-group** is enabled, the number of IPs in the address range should be a power of 2, i.e., the number of IPs should be even. If the address range has odd number of IPs, it displays following error message:

"address range from " + startAddr +" to "+ endAddr + " is odd"

Note This check is applicable only when the **chunk-group** is enabled and ony applicable for IPv4 addresses.

- **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.

NAT Support

Table 18: Feature History

Feature Name	Release Information	Description
Network Address Translation Binding Updates	2023.04	SMF supports NAT Binding Record functionality where, N4 Session Report Request Message is used for providing NAT Binding Record Information from UPF to SMF. After SMF receives NAT Binding Record Information from UPF, it sends the corresponding NAT Binding Updates to the AAA Server over the RADIUS interface. **Default Setting**: Disabled – Configuration required to enable

Feature Description

Network Address Translation (NAT) is a service that enables private IP networks to use the internet and cloud. NAT translates private IP addresses in an internal network to a public IP address before packets are sent to an external network. This helps to save IP addresses and improve security.

When NAT is enabled on the UPF, external platforms require the real-time NAT binding updates. This allows the mapping of private IP address to public IP address used on the Gi interface to be provided to external platforms.

SMF supports NAT Binding Record functionality where, N4 Session Report Request Message is used for providing NAT Binding Record Information from UPF to SMF. After SMF receives NAT Binding Record Information from UPF, it sends the corresponding NAT Binding Updates to the AAA Server over the RADIUS interface.

For more information, refer to the UCC 5G SMF Configuration and Administration Guide > IP Address Management chapter.

How it Works

This section describes the call flows and procedures pertaining to the Network Address Translation binding updates in the SMF.

Network Address Translation Binding Updates feature supports the following functionalities:

- NAT Binding updates of Port Chunk Allocation and Release

- Session deletion with Valid NAT IP
- NAT Binding Record handling at SMF

Call Flows

The following call flow depicts sending NAT Binding Updates in the SMF.

Port Chunk Allocation

NAT Binding Updates for Port Chunk Allocation occurs when subscribers share Network Address Translation IP addresses. This is performed either at a call setup or during data flow based on the allocation mode. NBUs supports both one-to-one and many-to-one Network Address Translation IP Modes.

This section describes the call flows pertaining to the Network Address Translation binding updates for Port chunk allocation in the SMF.

Figure 5: Call Flow for Port Chunk Allocation

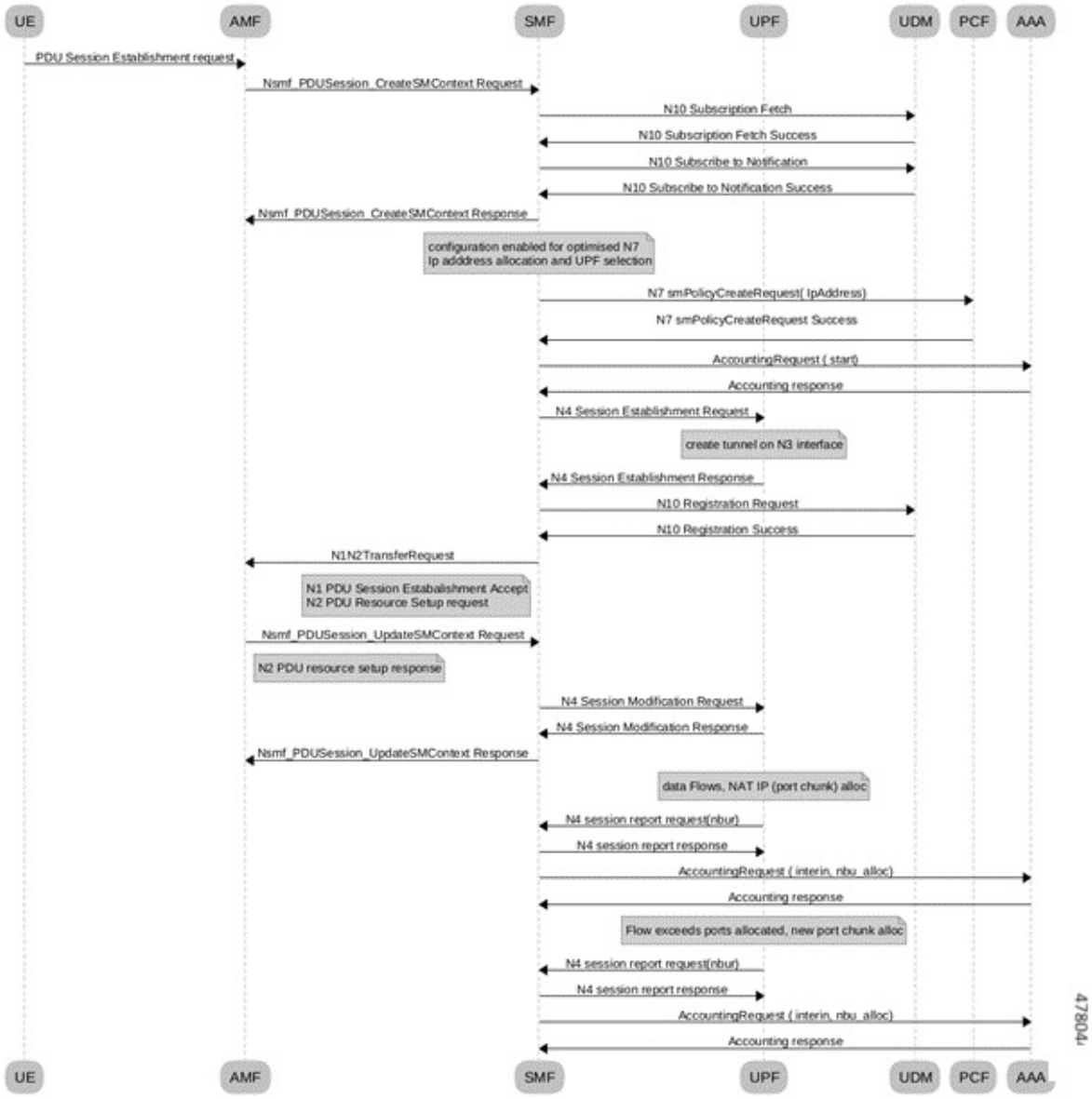


Table 19: Port Chunk Allocation Call Flow Description

Step	Description
1	During 5G session creation, RADIUS Accounting Request Start message is sent to the AAA server. .
2	When data flow happens for the subscriber, NAT IP and Port chunk gets allocated. Then UPF sends new N4 Session Report Request with NBR Information to the SMF.

Step	Description
3	SMF sends NAT Binding Update as RADIUS Accounting Request interim update to the AAA along with NAT related AVPs.
4	When number of flows crosses the already allocated ports, UPF allocates a new port chunk. Every time when Port chunk gets allocated, UPF sends a new N4 Session Report Request to the SMF and SMF in turn sends NAT Binding Update to the AAA server as RADIUS Accounting Interim update.

Port Chunk Release Procedure

This section describes the call flows pertaining to the Network Address Translation binding updates for Port chunk release in the SMF.

Figure 6: Call Flow for Port Chunk Release

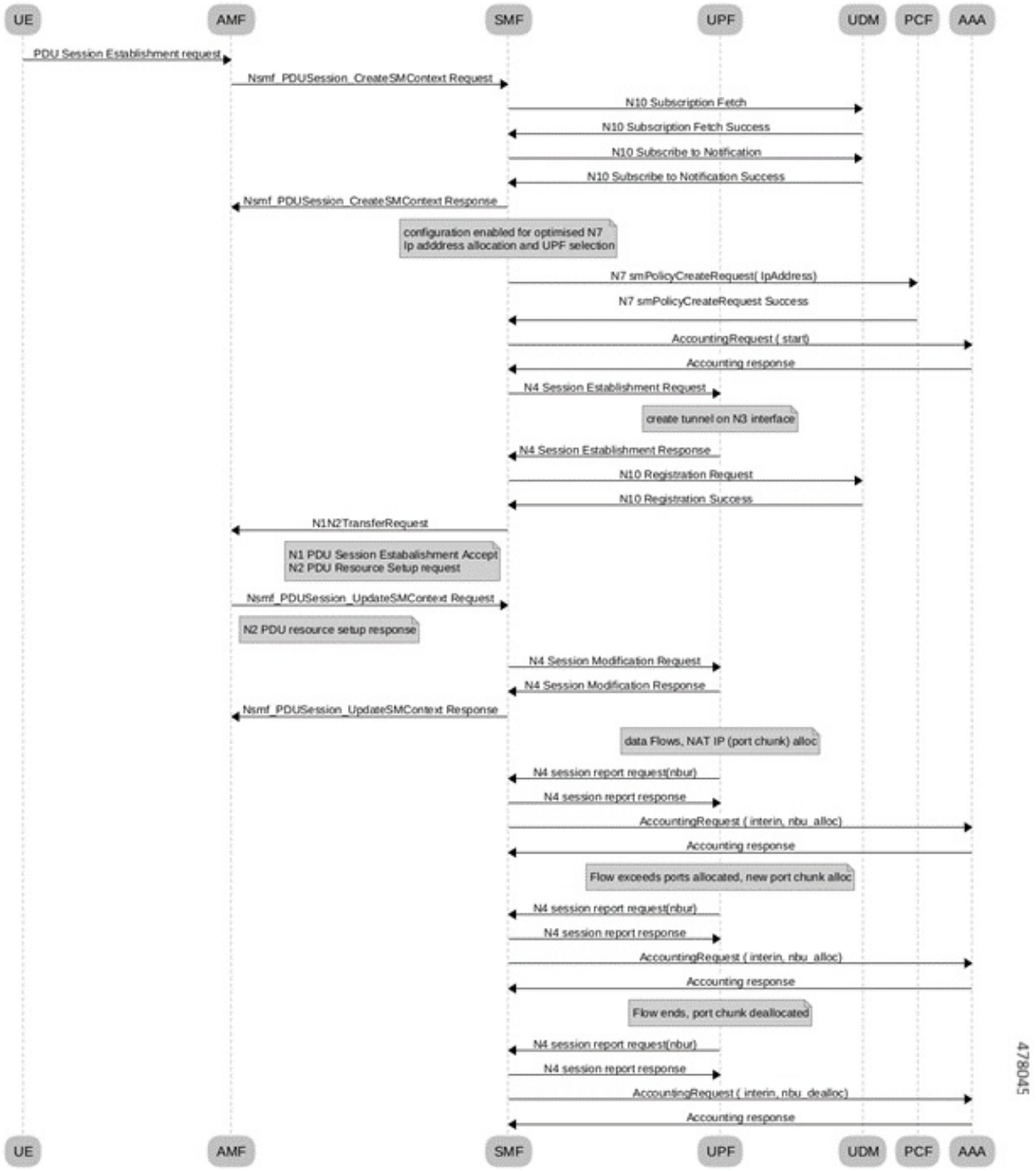


Table 20: Port Chunk Release Call Flow Description

Step	Description
1	During 5G session creation, RADIUS Accounting Request Start message is sent to the AAA server. .

Step	Description
2	When data flow happens for the subscriber, NAT IP and Port chunk gets allocated. Then UPF sends new N4 Session Report Request with NBR Information to the SMF.
3	SMF sends the NAT Binding Update as RADIUS Accounting Request interim update to the AAA server with NAT related AVPs.
4	When number of flows crosses the already allocated ports, UPF allocates a new port chunk. Every time when Port chunk gets allocated, UPF sends new N4 Session Report Request to the SMF and SMF in turn sends NAT Binding Update to the AAA server as RADIUS Accounting Interim update.
5	UPF releases the port chunk henever port chunk is not used. For each Port Chunk Release N4 Session Report Request is sent to the SMF. SMF in turn sends NAT Binding Update to the AAA server as RADIUS Accounting Interim update.
Note Every time Port chunk gets allocated or released, NAT Binding Update is sent to the AAA server as RADIUS Accounting Interim update for Port Chunk Allocation or Port Chunk Release.	

Session Deletion with Valid NAT IP

The following call flow depicts the session deletion with Valid NAT IP in the SMF.

Figure 7: Call Flow for Session Deletion

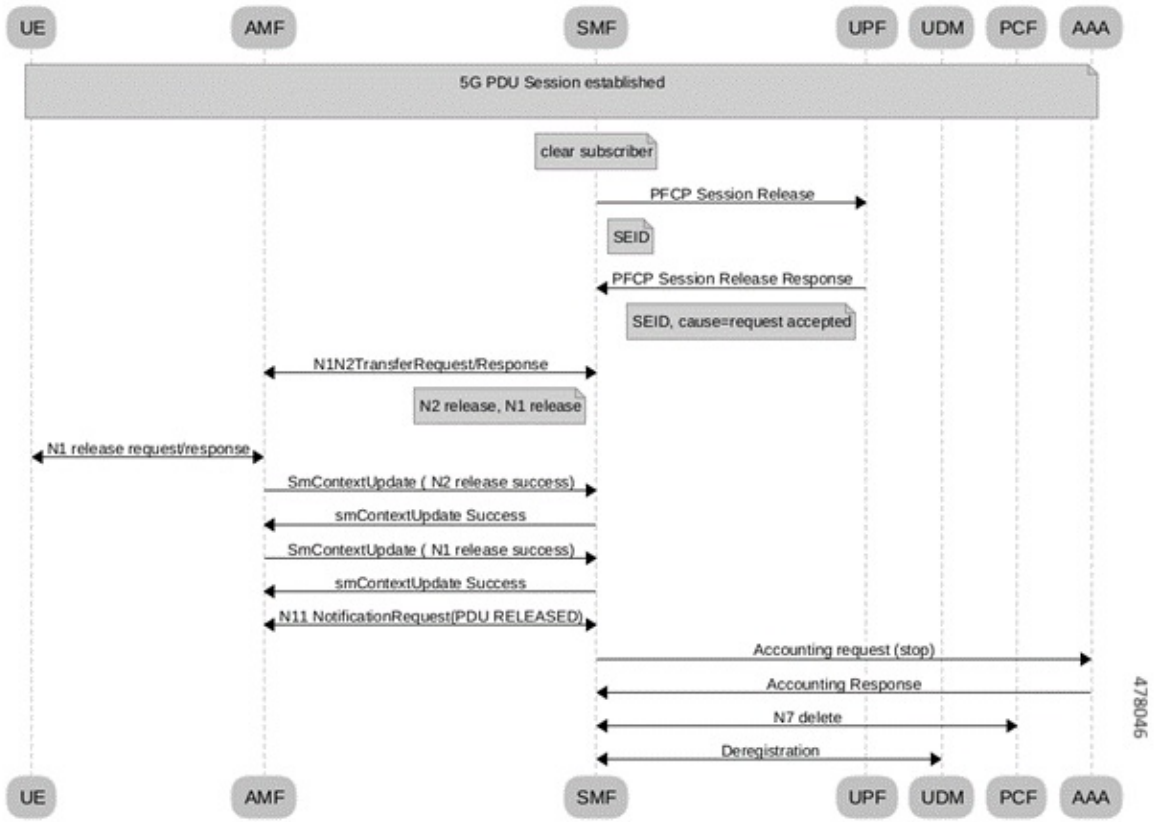


Table 21: Session Deletion Call Flow Description

Step	Description
1	During 5G session creation, RADIUS Accounting Request Start message is sent to the AAA server.
2	When data flow happens for the subscriber, NAT IP and Port chunk gets allocated. Then UPF sends new N4 Session Report Request with NBR Information to the SMF.
3	SMF sends the NAT Binding Update as RADIUS Accounting Request interim update to the AAA server.

There is no explicit messages sent to indicate Port Chunk Release, during Session Deletion Procedure. As part of Normal Session Deletion procedure, SMF sends Radius Accounting STOP message to the AAA Server. and this Radius STOP message should clear the previously sent NAT Binding Updates for Port Chunk Allocation for the Subscriber.

PFCP IEs for NAT Binding Procedure

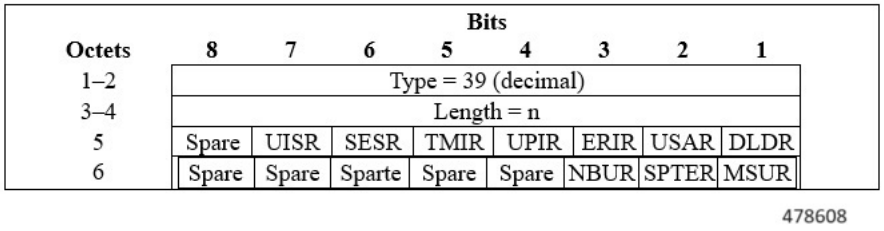
The following tables shows IEs in PFCP Session Report Message that carries NAT Binding Information.

Table 22: Session Report Request for Nat Bind Update

InformationElements	Priority	Condition / Comment	IE Type
Report Type	M	This IE shall indicate the type of the report that is NBUR	Report Type
NBR Info	M	NBR Info IE shall carry Information for NAT Bind Updates	NBR Info

The following table describes the encoding of the Report Type IE. It indicates the type of the report the UPF function sends SMF. To send the NBR Information, the report type is set to NBUR.

Figure 8: Report Type



The following table describes the NBR Information grouped IE.

Figure 9: NBR Information

Octet 1 and	NBR Info IE Type = 247 (decimal)						
Octets 3 and 4	Length = n						
Information elements	**P**	**Condition / Comment**	**Appl.**				**IE Type**
			Sxa	**Sxb**	**Sxc**	**N4**	
NAT IP	M	This IE shall identify NAT IP allocated or released	-	X	-	X	NAT IP
Port Chunk Info	M	Port Chunk Info IE carries Port Chunk that is allocated or released. Several IEs within the same IE type may be present to represent a list of Port Chunks.	-	X	-	X	Port Chunk Info
Allocation Flag	M	Allocation Flag shall indicate that NBR Info is for Allocation or for Release	-	X	-	X	Allocation Flag
NAPT Number of Users Per IP	M	This IE carries Number of Users allowed to share the same NAT IP	-	X	-	X	NAPT Number of Users Per IP
Release Timer	M	Release Timer is the Release Timer configure for the realm from which NAT IP was allocated or released	-	X	-	X	Release Timer

The following table describes the encoding of the NAT IP IE.

Figure 10: NAT IP

	Bits							
Octets	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
1–2	Type = 248 (decimal)							
3–4	Length = 4							
5–8	IPv4 IP Address Value							

The following table describes the encoding of the Port Chunk Information IE.

Figure 11: Port Chunk Information

	Bits							
Octets	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
1–2	Type = 249 (decimal)							
3–4	Length = 4							
5–6	Start Port Value							
7–8	End Port Value							

The following table describes the encoding of the Allocation Flag IE.

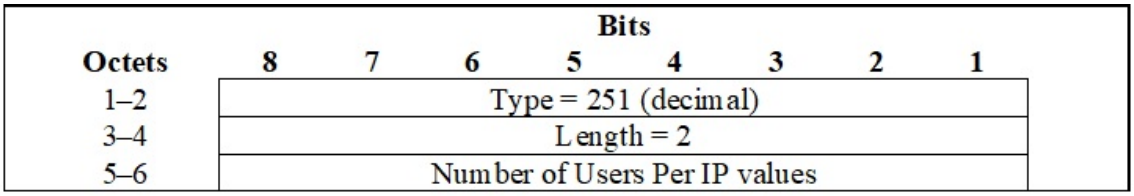
Figure 12: Allocation Flag

	Bits							
Octets	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
1–2	Type = 250 (decimal)							
3–4	Length = 1							
5	Flag							

Flag value 1 indicates the Allocation and Flag Value 0 indicates the Release.

The following table describes the encoding of the NAPT number of users per IP IE.

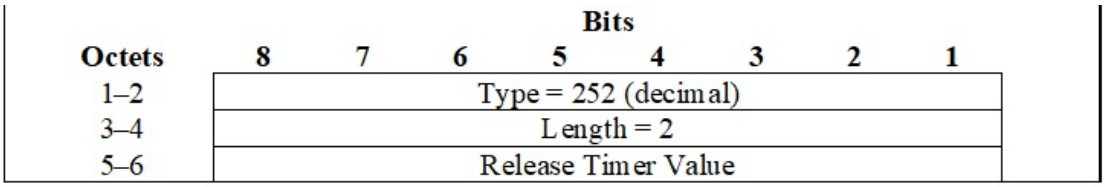
Figure 13: NAPT Number of Users Per IP



Octets	Bits 8	7	6	5	4	3	2	1
1–2	Type = 251 (decimal)							
3–4	Length = 2							
5–6	Number of Users Per IP values							

The following table describes the encoding of the Release Timer IE.

Figure 14: Release Timer



Octets	Bits 8	7	6	5	4	3	2	1
1–2	Type = 252 (decimal)							
3–4	Length = 2							
5–6	Release Timer Value							

The value of the release timer is in seconds.

Handling Network Address Translation Binding Record at SMF

SMF handles a Session report in a Non-State Handler process. A response is sent back immediately and an internal event is posted to handle the request.

Typically, NAT Binding Reports are considered as high priority and adding them to the Usage report procedure results in delay. This is because the Usage report procedure has lower priority compared to other procedures such as handover, modify, idle mode procedures.

To handle with appropriate priortiy, SMF handles the NIntSelfTxnN4SessRptReq for NBU as part of non-state handler. This allows messages to get handled even when other procedures are ongoing. This ensures that NAT Binding Update handling need not wait until the completion of the current procedure.

There are no changes on the Handling of regular Session Reports (non-NBU).

Note During Create and Release procedures, NAT Binding Report is not handled fully. Only the N4 Success Response is sent and the message gets ignored.

Radius AVPs for NAT Binding Procedure

NBUs are sent as part of RADIUS Accounting Interim update messages. Following new RADIUS attributes (AVPs) are sent as part of NBU messages, in addition to standard RADIUS Accounting Attributes.

- NAT-IP-Address: Network Address Translation IP Address is allocated or released to the Subscriber flow.
- NAT-Port-Block-Start: Starting Port of Port chunk that is allocated or released.
- NAT-Port-Block-End: Ending Port of the Port Chunk that is allocated or released.
- Alloc-Flag: Flag indicates that NBU is for Allocation or Release. 1 is for allocation and 0 for release.

- Loading-Factor: Number of users per IP
- Binding-Timer: Network Address Translation Binding Release Timer

Limitations

Following are the limitations:

- When a call gets cleared due to ungraceful termination, for example, battery that is drained before the IP release, Network Address Translation bind update is not performed.
- When the Network Address Translation binding updates feature is enabled, minimal performance impact to SMF might occur because of additional signaling between UPF to SMF and SMF to the AAA Server.
- As there is an extra hop (UPF to SMF update) for sending an NBU to the AAA server, some delay can occur before the NBU is sent. Hence, it is not guaranteed to achieve real-time delivery of those updates.

Configuring NAT Binding Update

Following CLI configures NAT binding updates in SMF:

```
config
  profile dnn intershat dnn_intershat
    supported-features   [ inter-plmn-ho nat-binding-update  ]
  exit
```

NOTES

- **supported-features** —List of supported features which can be enabled/disabled in SMF at DNN level. *This flag enables inter plmn roaming handover*
- **inter-plmn-ho** —Enables inter plmn roaming handover.
- **nat-binding-update** —Enables support for NAT Binding Update procedure

.

OAM Support

Bulk Statistics

The following metrics details are modified in support of NAT Binding Updates (NBU) feature.

- smf_sess_report_stats - This existing metrics includes a label sess_report_type for which a new label value sess_report_type_nbur is added. Use the metrics in handling PFCP Session Report with NBU.

  Sample Metrics:

  ```
  smf_sess_report_stats{app_name="SMF",cluster="SMF",data_center="DC",gr_instance_id="1",
  instance_id="0",rat_type="NR",reason="success",
  service_name="smf-service",sess_report_type="sess_report_type_nbur"} 3
  ```

- radius_accounting_message_stats - This existing metrics includes a new label trigger_reason and new label value nat_binding_update. Use the metrics in handling PFCP Session Report with NBU.

  Sample Metrics:

```
Attempt:
radius_accounting_message_stats{app_name="SMF",cluster="SMF",data_center="DC",
dnn="",gr_instance_id="1",instance_id="0",procedure_type="radius_update",
rat_type="NR",reason="",service_name="smf-service",status="attempted",
trigger_reason="nat_binding_update"} 3

Successful Response:
radius_accounting_message_stats{app_name="SMF",cluster="SMF",data_center="DC",
dnn="",gr_instance_id="1",instance_id="0",procedure_type="radius_update",
rat_type="NR",reason="",service_name="smf-service",status="success",
trigger_reason="nat_binding_update"} 2

Failure Response:
radius_accounting_message_stats{app_name="SMF",cluster="SMF",data_center="DC",
dnn="",gr_instance_id="1",instance_id="0",procedure_type="radius_update",
rat_type="NR",reason="",service_name="smf-service",status="failures",
trigger_reason="nat_binding_update"} 1
```

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)
```

Releasing N4 Resources or Association

Table 23: Feature History

Feature Name	Release Information	Description
N4 Resource or Association Release	2023.04	SMF supports clearing resources with respect to a particular UPF from the SMF using a CLI **release-resource**.

Feature Description

Whenever the network is in an erroneous state with respect to the state of IP pools, such as pool mismatch between SMF and UPF, there may be a need to clear resources like subscriber sessions or IP pools/chunks with respect to a particular UPF from the SMF.

This feature introduces a CLI that triggers and automates the resource releasing process from SMF. When the resource releasing process is triggered, SMF does the following two things:

1. It clears all the IP pools associated to the peer. This in turn clears the calls using the pool and also deletes the routes.

Note Static IP chunks will not be cleared as part of the release resource operation.

2. A new call disconnect reason is added to the SMF Disconnect Statistics. Also, the routes are re-registered with UPF post a quarantine time of 2 minutes.

Caution This feature to be executed only when the SMF is in an unhealthy state. If executed on a healthy SMF, it will impact the active subscribers as it involves clearing the calls. Please exercise with caution.

How It Works

The N4 resource or association releasing process follows the given steps:

1. Set the priority of the UPF to 0. This prevents new calls from picking up this UPF.
2. Execute the command for the N4 interface to clean up the resources.
3. Ensure to check if the calls are cleared.
4. Set the priority of UPF back.

Configuration for Releasing N4 Resources or Associations

The following configuration is used to release N4 resources or associations between SMF and UPF:

```
mode debug exec  action release-resource attributes { interface n4 peer {
 ip upf_ip } } condition { dnn matches dnn-name }
```

NOTES:

- **debug**—This keyword indicates that this is a debug mode operational CLI.
- **exec**—This command executes the command to trigger N4 release per UPF from SMF.
- **action** —This command indicates the action to perform.
- **release-resource**—Upon executing **release-resource**, the SMF triggers clean-up of subscriber/pool resources to the peer provided.
- **attributes**—This command defines the attributes to perform an action.
- **peer**—The UPF peer can be defined using the UPF IP address.
- **interface** —This command specifies the interface name to trigger the operation for.
- **condition**—This CLI specifies the additional condition to filter the data for the selected operation. This is an optional filter.
- **dnn**—Pools associated to this dnn-name will be cleared based on the given operator. The DNN name for which the IP pools are assigned, has to be used here.
- **matches**—This command verifies the DNN that matches the output modifiers.

Configuration Example

Following is the sample configuration for releasing the N4 resources or associations:

```
mode debug exec action release-resource attributes { interface n4 peer { ip <upf_ip> } }
condition { dnn matches dnn1 }
```

Configuration Verification

Following command verifies the configuration for releasing the N4 resources and associations per UPF and SMF:

```
[smf] smf# show ipam pool poolv4 ipv4-addr
Wed Aug  23 07:04:48.511 UTC+00:00

================================================================================
Flag Indication: S(Static) O(Offline)
================================================================================
StartAddress          EndAddress         AllocContext                  Flag  GroupId
================================================================================
209.165.200.225       209.165.200.231    QT:NM1
209.165.200.232       209.165.200.239    Free:CP
209.165.200.240       209.165.200.247    Free:CP
209.165.200.248       209.165.200.254    Free:NM1
================================================================================
```

OAM Support

This section details the metrics supported in this feature.

Metrics

Following reason is added in the **smf_disconnect_stats** counter:

disc_rel_upf_admin_clear: The session terminates as the IP chunk is cleared for a particular UPF triggered by the admin.

Identification of Corrupted IP Chunks

Feature Description

SMF allows identifying the missing or corrupted IP chunks by auditing the configured IP pools and their data in cache-pod and IPAM internal states.



Note Please contact your Cisco Account or support representative for any questions about configuring SMF to identify the missing or corrpupted IP chunks.

How It Works

SMF can capture the debug information that can check for any corruption in IPAM data due to following scenario:

- The chunk entry is missing from the IPAM cache-pod data.
- The chunk entry is present, but 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.