IP Address Management

Feature Summary and Revision History

Summary Data

Table 1: Summary Data

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

Revision History

Table 2: Revision History

Revision Details	Release
Introduced support for allocating IANA and IAPD from same IP pool.	2025.01.0

Revision Details	Release
Introduced support for IPAM Asynchronous route programming.	2024.04.0
Introduced support for pre-allocation of Gateway IP and address chunks.	2024.04.0
Introduced enhancements in IPAM chunk allocation with NM pair for SRG.	2024.02.0
Introduced support for dynamic chunk allocation in a CP-GR setup.	2024.02.0
Introduced support for Variable Chunk Size for an IPAM Data Plane.	2024.01.0
Introduced support for the Static IP Mapping Database.	2022.02.0
Enhancement Introduced: The IPAM feature is NSO-integrated.	2021.04.0
The following sections were added: • Configuration Recommendations • IPAM Enhancements • Static IP Pool Enhancements	2021.03.0
First introduced.	2021.01.0

Feature Description

Note This feature is Network Services Orchestrator (NSO) integrated.

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 Session Management function (SMF), Policy Charging function (PCF), and Broadband Network Gateway (BNG).

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.

IPAM Components

This section describes the different components of the IPAM system.

IPAM Sub-Modules

The IPAM functionalities are categorized in the following sub-modules:

IPAM Server

This module manages the complete list of pools and address-space configurations. It splits the configured address-ranges into smaller address-ranges (statically or dynamically) to distribute it to the IPAM Cache modules. The IPAM server can be deployed as a centralized entity to serve a group of CN clusters or as an integrated entity within a single cluster.

IPAM Cache

This module acquires the free address-ranges from the IPAM server and allocates individual IP addresses to the IPAM clients. The IPAM cache is generally deployed in the Distributed mode running within each cluster, to communicate with the co-located or remotely located IPAM server. It is also responsible for address-range reservation per UP and pool threshold monitoring. The IPAM server and cache modules can also run in an integrated mode.

IPAM Client

This module is tightly coupled with its respective network-function, responsible for handling request and release of individual IP address from the IPAM cache for each IP managed end-device.

Unlike the IPAM server and cache module, the IPAM client caters to use-cases specific to network-functions such as BNG, SMF, PCF, and so on.

IPAM Integration in cnBNG

The Cloud-Native Broadband Network Gateway (cnBNG) function comprises of loosely coupled microservices that provide the functionality of the BNG. The decomposition of these microservices is based on the following three-layered architecture:

1. Layer 1: Protocol and Load Balancer Services (Stateless)
2. Layer 2: Application services (Stateless)
3. Layer 3: Database Services (Stateful)

The IPAM and cnBNG integration occurs in the Application Services layer.

BNG Node Manager Application—The BNG Node Manager application is responsible for the User Plane function (UPF) management, ID and resource management, and IP address management. Therefore, the IPAM Cache is integrated as part of this microservice.

Also, the UPF uses the IPAM Client module for address-range-reservation per UPF.

BNG DHCP and PPPOE Application—The BNG-DHCP and BNG-PPOE pods are responsible for providing IP addresses to the BNG subscriber session. During session bring-up, the IP address is requested and during session bring-down, the IP address is released back. These First Sign of Life (FSOL) applications send the inter-process communications (IPC) to the Resource Manager (RMGR) component in the NodeMgr. The NodeMgr receives the IPC and invokes the IPAM component.

IPAM Server Application—Based on the deployment model, the IPAM Server runs as an independent microservice as part of the same cluster or in a remote cluster.

In standalone deployments, the IPAM Server functionality is an integral part of the IPAM Cache, that is, it runs as part of the Node Manager microservice itself.

Configuration Recommendations

This section provides the following configuration recommendations.

Pool-Size Configuration

While configuring the pool-size, the recommendation is to keep additional buffer of IPs for undisturbed subscriber churn. Consider the following recommendations:

- Released IPs become part of the quarantine queue depending on the quarantine configuration. These IPs cannot be reused until they are free.
- Due to the threshold logic, the CP automatically reserves additional chunks to the User Plane function (UPF) when required. It is based on a first-come-first-server basis.

  In a multiple UPF sharing a pool use case, whichever UPF hits the threshold first is given a chunk. Therefore, free chunks may not be available for other UPFs. That is, 100% pool utilization may not occur for a given pool in certain conditions. In this scenario, configure additional IPs accordingly or split into multiple pools per UPF.

Split-Size Configuration

While configuring the split-size of a dynamic pool, consider the following recommendations:

- Number of IPs in pool
- Total number of subscribers. That is, the number of IPs versus the percentage of the number of sessions.
- Number of UPFs sharing the pool
- Number of sessions per UPF. That is, the number of sessions versus the percentage of the number of UPFs.
- Quarantine time of the pool
- Churn rate. That is CPS of IP allocation, CPS of IP release.
- Max routes a UPF can support (example: ASR9k supports 32 routes per loopback in Release 7.4.x/7.5.x)

Each Node Manager (nodemgr) uses per-cache as the split-size of the chunk.

Each UPF is assigned per-dp as the split-size of a chunk.

Each Node Manager reserves chunk of per-cache size and further splits (if-applicable) and assigns them to a UPF.

A chunk can be in one of the following states:

- <upf-name> (means, this chunk is assigned to an UPF)
- Free:CP (means, it is free in the cache-pod, any Node Manager can use it)
- Free:NM0 (means, it is free in the Node Manager instance-0. Only this Node Manager can use it)
- Free:NM1 (means, it is free in the Node Manager instance-1. Only this Node Manager can use it)
- QT:NM0 (means, it is in quarantine in the Node Manage instance-0. This is moved to free after quarantine timer expiry)
- QT:NM1 (means, it is in quarantine the Node Manage instance-1. This is moved to free after quarantine timer expiry)

Threshold Configuration

Changing threshold configuration on-the-fly takes effect only after the next event, such as address-allocation or address-delete. On an idle-system, there may not be any immediate action.

Variable Chunk Size Support for an IPAM Data Plane

You can now configure multiple pools of IP addresses, each with different chunk sizes for a given IPAM Data Plane (SRG group in SRG deployment, or an UPF in non-SRG deployment). This configuration would require an administrator to divide the available IP ranges into different pools, which might pose challenges in effectively using the IP addresses. To overcome this issue, you can initially configure a reasonable number of IP addresses in the pools as per the deployment plan. By monitoring the IP chunk usage per pool, you can then provide additional IP ranges as required. You can also use a metric to monitor the chunk usage for each pool.

To associate multiple pools to a DHCP profile, you can now attach a pool-tag, which in turn can be associated with one or more pools. In tag-based IP allocation for a given DP, the first chunk is allocated from the highest priority pool (for example, priority 0) for the specified tag, and the next chunk is allocated from the second highest priority pool (priority 1). If one chunk gets allocated from both the pools that are associated to a tag, any subsequent chunk requirement utilizes the least priority pool.

For configuration details, see Configuring Variable Chunk Size Support for an IPAM Data Plane section.

Limitations for Variable Chunk Size Support for an IPAM Data Plane

- This feature is supported only for IPoE subscribers.
- Presently, only two chunk sizes (2 pools) are supported for a given Data Plane.
- Configuring same priority for different pools that are associated with the same pool tag is not supported.
- In a DHCP profile, you can either configure pool-name or pool-group-tag. Configuring both together is not supported.
- You cannot change the Pool priority dynamically. Priority must be specified when the pool is configured. The default value is 0.

- You cannot change the Tag configuration under a pool dynamically. Tag information must be configured when the pool is configured.

How it Works

This section describes the call flow pertaining to the integration of the IPAM in the cnBNG.

Call Flows

This section describes the following IPAM call flows in cnBNG:

- IPAM initial sequence call flow
- IPAM call flow
- IPAM static-pool call flow

IPAM Initial Sequence Call Flow

This section describes the cnBNG initial sequence call-flow.

Figure 1: IPAM Initial Sequence Call Flow

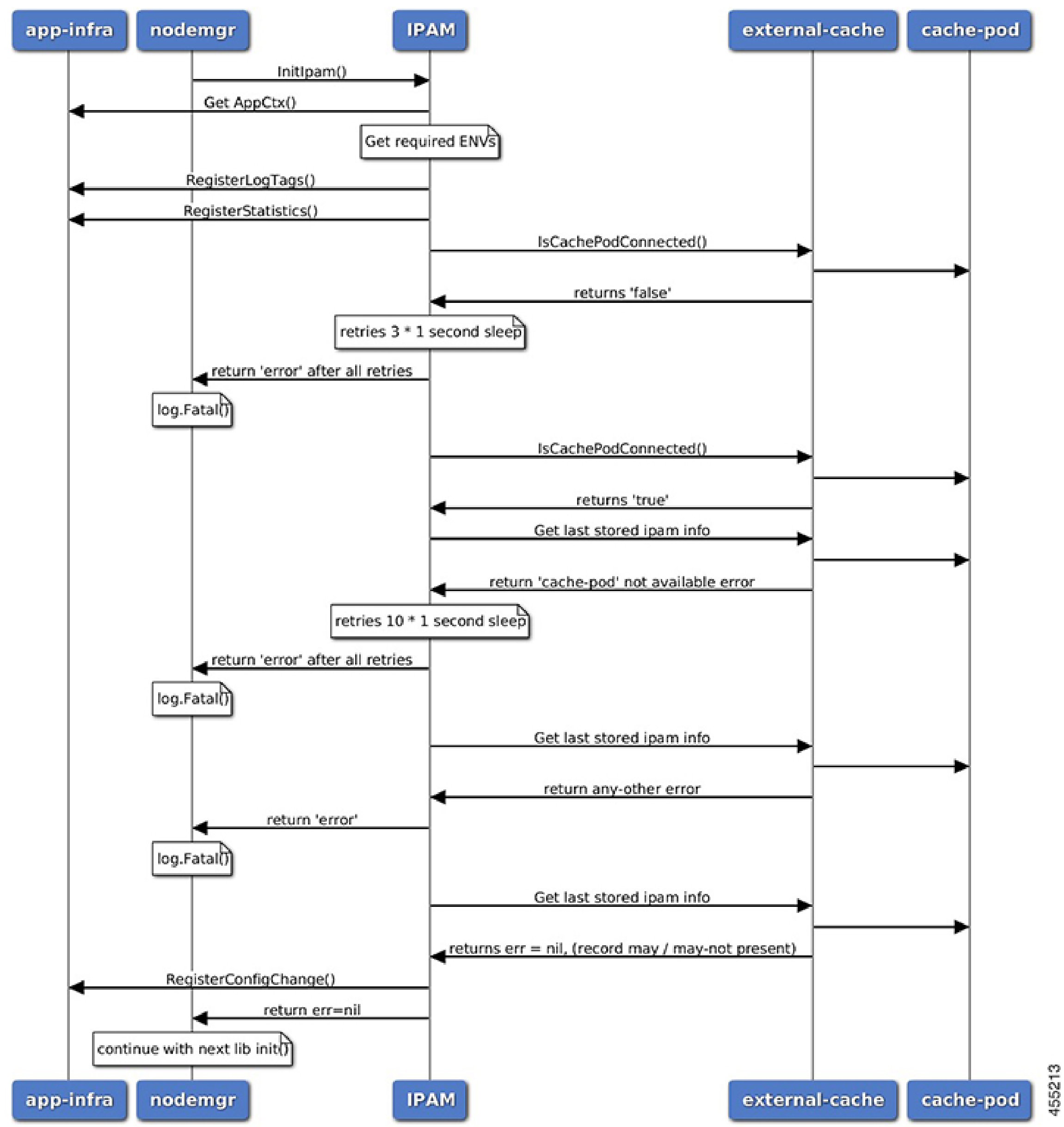


Table 3: IPAM Initial Sequence Call Flow Description

Step	Description
1	IPAM reads the required environments, registers with the application infrastructure for log-tags, metrics, and database connection.
2	IPAM restores the previous state from the cache-pod, if present.
3	IPAM registers for configuration change and applies the new configuration change, if any. -change, apply new config-changes if any

IPAM Call Flow

This section describes the cnBNG IPAM call-flow.

Figure 2: IPAM Call Flow

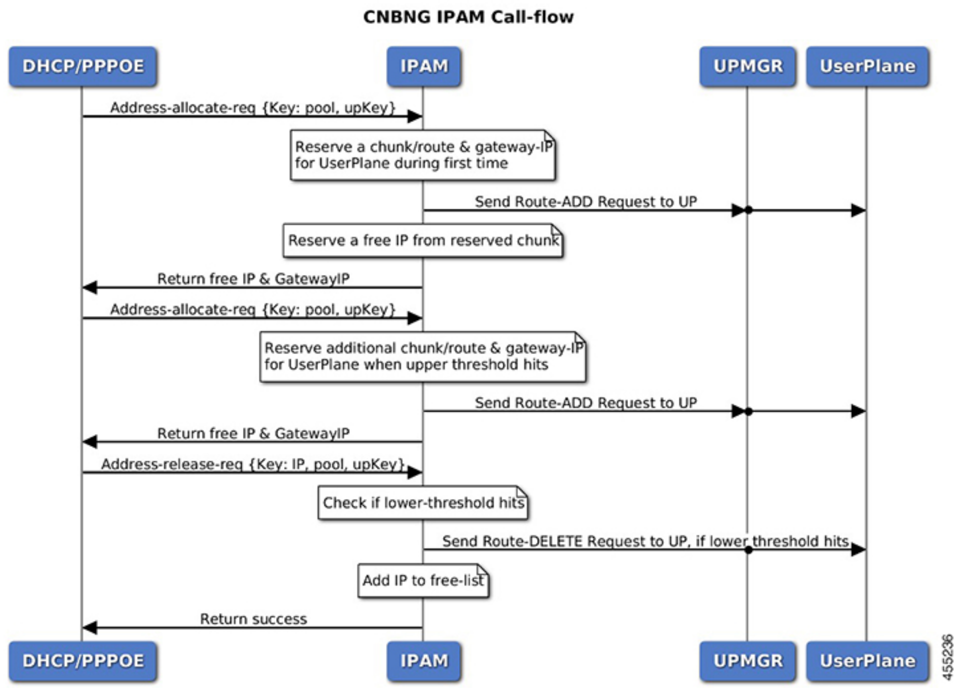


Table 4: IPAM Call Flow Description

Step	Description
1	IPAM receives the 'addr-alloc' request from the DHCP or PPPoE pod with pool-name, addr-type and user plane function (UPF) as input.
2	IPAM reserves a new address-range (if not already present for UPF) and sends a ROUTE-ADD message to the UPF. It waits for a success or failure response. If the receives a failure response, it removes the chunk and repeats this step.
3	IPAM reserves a free-IP from the assigned address-range and returns to the DHCP or PPPoE.
4	IPAM monitors the 'upper-threshold' for each UPF during each IP address-allocation and also has a background thread that monitors. It then assigns new address-ranges to the UPF and repeats the ROUTE-ADD flow.
5	IPAM receives the 'addr-free' request from the DHCP or PPPoE pod with pool-name, addr-type, addr or pfx, and UPF as input.
6	IPAM moves the addr or pfx first to the quarantine-list until the quarantine timer and later moves it to the free-list.

Step	Description
7	IPAM monitors the ‘lower-threshold’ (currently 0%) of the address-range of each UPF, removes the address-range from the UPF, and sends the ROUTE-DELETE message.

IPAM Static-Pool Call Flow

This section describes the IPAM static-pool call flow.

Figure 3: IPAM Static Pool Call Flow

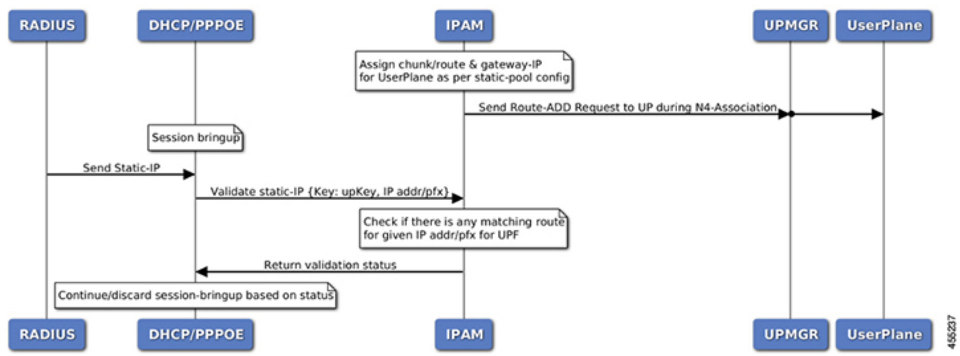


Table 5: IPAM Call Flow Description

Step	Description
1	IPAM receives the ‘addr-alloc’ request from the DHCP or PPPoE pod with pool-name, addr-type and user plane function (UPF) as input.
2	IPAM reserves a new address-range (if not already present for UPF) and sends a ROUTE-ADD message to the UPF. It waits for a success or failure response. If the receives a failure response, it removes the chunk and repeats this step.
3	IPAM reserves a free-IP from the assigned address-range and returns to the DHCP or PPPoE.
4	IPAM monitors the ‘upper-threshold’ for each UPF during each IP address-allocation and also has a background thread that monitors. It then assigns new address-ranges to the UPF and repeats the ROUTE-ADD flow.
5	IPAM receives the ‘addr-free’ request from the DHCP or PPPoE pod with pool-name, addr-type, addr or pfx, and UPF as input.
6	IPAM moves the addr or pfx first to the quarantine-list until the quarantine timer and later moves it to the free-list.
7	IPAM monitors the ‘lower-threshold’ (currently 0%) of the address-range of each UPF, removes the address-range from the UPF, and sends the ROUTE-DELETE message.

Limitations

The IPAM feature has the following limitations:

- Duplicate IP address is not supported within a pool.
- Duplicate IP address is not supported across pools, that belong to same VRF.
- Removal of ‘pool’ is not supported while addresses are already assigned.
- Removal or modification of IP-address-ranges is not supported while addresses are already assigned.
- Change of ‘source’ field is not supported while address or prefixes are already assigned.
- Change of ‘vrf-name’ of pool is not supported while address or prefixes are already assigned.
- Start-address should be less than the End-address.
- Configuring addr-range split-size in wrong manner, that is, size of address-range < size-of-per-cache < size-of-dp, is not supported.
- Configuring IPv6 Address (IANA) and Prefix (IAPD) values interchangeably is not supported.
- Configuring invalid ‘prefix-length’ for Prefix (IAPD) range is not supported.

Configuring IPAM Feature

This section describes how to configure the IPAM feature.

Configuring the IPAM feature involves the following steps:

1. Configuring IPAM source
2. Configuring the global threshold
3. Configure IPAM address pool
4. Configuring IPv4 address ranges
5. Configuring IPv6 address ranges
6. Configuring IPv6 prefix ranges
7. Configuring the IPv4 threshold
8. Configuring the IPv6 threshold
9. Configuring IPv4 address range split
10. Configuring IPv6 address and prefix address-range split

Configuring IPAM Source

Use the following configuration to configure the IPAM source.

```
config
   ipam
```

```
    source local
     threshold { ipv4-add percentage | ipv6-address percentage | ipv6-prefix
percentage }
      commit
```

NOTES:

- **ipam**: Enters the IPAM Configuration mode.
- **source local**: Enters the local datastore as the pool source.
- **threshold { ipv4-add** *percentage* **| ipv4-address** *percentage* **| ipv6-prefix** *percentage* **}**: Specifies the threshold in percentage for the following:
  - **ipv4-add** *percentage*: Specifies the IPv4 threshold. The valid values range from 1 to 100. The default value is 80.
  - **ipv6-add** *percentage*: Specifies the IPv4 threshold. The valid values range from 1 to 100. The default value is 80.
  - **ipv6-prefix** *percentage*: Specifies the IPv6 threshold prefix. The valid values range from 1 to 100. The default value is 80.

Configuring Global Threshold

Use the following configuration to configure the global threshold.

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

NOTES:

- **ipam**: Enters the IPAM Configuration mode.
- **threshold**: Enters the threshold sub-mode.
- **ipv4-add** *percentage*: Specifies the IPv4 threshold. The valid values range from 1 to 100. The default value is 80.
- **ipv6-add** *percentage*: Specifies the IPv4 threshold. The valid values range from 1 to 100. The default value is 80.
- **ipv6-prefix** *percentage*: Specifies the IPv6 threshold prefix. The valid values range from 1 to 100. The default value is 80.

Configuring IPAM Address Pool

Use the following configuration to configure the IPAM address pool.

```
config
  ipam
    address-pool pool_name [ address-quarantine-timer ] [offline ] [ static
 user_plane_name ] [ vrf-name string ]
     commit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **address-pool** *pool_name* **[address-quarantine-timer] [offline] [static** *user_plane_name* **] [vrf-name** *string* **]**: Configures the address pool configuration. *pool_name* must be the name of the address pool.

  This command configures the following parameters:

  - **address-quarantine-timer**: Specifies the address quarantine time in seconds. The valid values range from 4 to 3600. The default value is 4.
  - **offline**: Sets the address pool to offline mode.
  - **static** *user_plane_name*: Specifies the 'user-plane' name associated to this static-pool.
  - **vrf-name** *string*: Configures the Virtual routing and forwarding (VRF) name of the pool.

Configuring IPAM Quarantine Timer

This section describes how to configure the IPAM quarantine timer.

```
config
   ipam
      address-pool pool_name
         address-quarantine-timer quarantine_timer_value
         vrf-name vrf_name_value
         ip4
            address-range start_ipv4_address end_ipv4_address
            address-range start_ipv4_address end_ipv4_address
         !
       !
     !
```

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-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.
- **vrf-name** *vrf_name_value*—Specifies the name of the VPN routing and forwarding (VRF) for the pool.
- **ipv4**—Enters the IPv4 mode.
- **address-range** *start_ipv4_address end_ipv4_address*—Specifies the IP addresses for start and end IPv4 address-range.

Configuring IP Address Reservation

Use the following configuration to reserve the minimum number of addresses for each User Plane function (UPF), Node Manager, and pool.

pool.

```
config
   ipam
      instance instance_id
         min-dp-addr-size { ipv4-addr | ipv6-addr | ipv6-prefix }
         commit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **instance** *instance_id*: Specifies the IPAM instance and enters the instance sub-mode. *instance_id* must be an integer. The valid value ranges from 1 to 8.
- **min-dp-addr-size { ipv4-addr | ipv6-addr | ipv6-prefix }**: Specifies the minimum number of addresses to reserve for each UPF, Node Manager, and pool.
  - **ipv4-addr**: Specifies the minimum number of IPv4 addresses to reserve.
  - **ipv6-addr**: Specifies the minimum number of IPv6 addresses to reserve.
  - **ipv6-prefix**: Specifies the minimum number of IPv6 prefixes to reserve.

Configuring IPv4 Address Ranges

Use the following configuration to configure the IPv4 address ranges.

```
config
   ipam
      address-pool pool_name
         ipv4
            address-range start_ipv4_address end_ipv4_address [ default-gateway
ipv4_address ] [ offline ]
            commit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **address-pool** *pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **ipv4**: Enters the IPv4 mode of the pool.
- **address-range** *start_ipv4_address end_ipv4_address* **[default-gateway** *ipv4_address* **] [offline]**: Configures the IPv4 address range with the starting and ending IPv4 address.
  - **default-gateway** *ipv4_address*: Specifies the IPv4 address of the default gateway.
  - **offline**: Sets the address pool to offline mode.

Configuring IPv6 Address Ranges

Use the following configuration to configure the IPv6 address ranges:

```
config
  ipam
    address-pool pool_name
      ipv6
        address-range start_ipv6_address end_ipv6_address [ offline ]
        commit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **address-pool** *pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **ipv6**: Enters the IPv6 mode of the pool.
- **address-range** *start_ipv6_address end_ipv6_address* **[offline]**: Configures the IPv6 address range with the starting and ending IPv6 address.

  [offline]: Sets the address pool to offline mode.

Configuring IPv6 Prefix Ranges

Use the following configuration to configure the IPv6 prefix ranges:

```
config
  ipam
    address-pool pool_name
      ipv6
        prefix-ranges
          prefix-range prefix_value prefix-length prefix_length
          commit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **address-pool***pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **ipv6**: Enters the IPv6 mode of the pool.
- **prefix-ranges**: Enters the prefix ranges mode.
- **prefix-range** *prefix_value* **prefix-length** *length*: Configures the IPv6 prefix range. *prefix_value* specifies the IPv6 prefix range.

  prefix-length *length* specifies the IPv6 prefix length.

Configuring IPv4 Threshold

Use the following configuration to configure the IPv4 threshold:

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

NOTES:

- **ipam**: Enters the IPAM Configuration mode.
- **address-pool** *pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **ipv4**: Enters the IPv4 mode of the pool.
- **threshold**: Enters the threshold sub-mode.
- **upper-threshold** *percentage*: Specifies the IPv4 upper threshold value in percentage. The valid values range from 1 to 100. The default value is 80.

The following is a sample configuration:

```
config
   ipam
      address-pool p1
         ipv4
            threshold
               upper-threshold 80
```

Configuring IPv6 Prefix-Range Threshold

Use the following configuration to configure the IPv6 prefix-range threshold.

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

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **address-pool** *pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **ipv6**: Enters the IPv6 mode of the pool.
- **prefix-ranges**: Enters the IPv6 prefix ranges sub-mode.

- **threshold**: Enters the threshold sub-mode.
- **upper-threshold** *percentage*: Specifies the IPv6 upper-threshold value in percentage.

The following is an example configuration:

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

Configuring IPv4 Address Range Split

Use the following configuration to configure the IPv4 address range split.

```
config
   ipam
      address-pool pool_name
         ipv4
            [ no ] split-size { per-cache value | per-dp value }
            commit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- -**address-pool** *pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **ipv4**: Enters the IPv4 mode of the pool.
- **[no] split-size { per-cache** *value* **| per-dp** *value* **}**: Specifies the size of the IPv4 range to be split for each IPAM cache allocation. The IPAM server consumes this configuration. The **no** form of this command disables the splitting of the address-ranges into smaller chunks.

  per-cache *value*: Specifies the size of the IPv4 range to be split for each Data-Plane (User-Plane) allocation. The valid values range from 2 to 262144. The default value is 1024.

  The IPAM cache consumes this configuration.

- **per-dp** *value*: Specifies the size of the IPv4 range to be split for each Data-Plane (User-Plane) allocation. The valid values range from 2 to 262144 The default value is 256.

  The IPAM cache consumes this configuration.

Configuring IPv6 Address and Prefix Address-Range-Spilt

Use the following configuration to configure the IPv6 address and prefix address range spilt.

```
config
   ipam
      address-pool pool_name
         ipv6
            address-ranges
```

```
        [ no ] spilt-size { per-cache value | per-dp value }
        commit
    prefix-ranges
        [ no ] spilt-size { per-cache value | per-dp value }
        commit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **address-pool** *pool_name*: Configures the address pool. *pool_name* must be the name of the address pool.
- **ipv6**: Enters the IPv6 mode of the pool.
- **[no] spilt-size { per-cache** *value* **| per-dp** *value* **}**: Specifies the size of the IPv6 range to be split for each IPAM cache allocation. The IPAM server consumes this configuration. The **no** form of this command disables the splitting of the address-ranges into smaller chunks.

  per-cache *value*: Specifies the size of the IPv6 range to be spilt for each Data-Plane (User-Plane) allocation. The valid values range from 2 to 262144. The default value is 1024.

  The IPAM cache consumes this configuration.
- **per-dp** *value*: Specifies the size of the IPv6 range to be spilt for each Data-Plane (User-Plane) allocation. The valid values range from 2 to 262144 The default value is 256.

  The IPAM cache consumes this configuration.

Configuring Variable Chunk Size Support for an IPAM Data Plane

Use the following commands to configure the IPAM tags:

```
ipam
 instance instance_id
  source local
  address-pool pool_name
   tags
    group tag_value
   exit
   pool-priority
    priority <0>
   exit
```

NOTES:

- **ipam**: Enters the IPAM configuration.
- **instance** *instance_id*: Specifies the IPAM instance and enters the instance sub-mode. *instance_id* must be an integer. The valid value ranges from 1 to 8.
- **source local**: Enters the local datastore as the pool source.
- **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

- **tags group** *tag_value*: Specifies the tag group value of the pool. All pools carrying the same **tag group** value can be associated to a DHCP profile using the keyword **pool-group-tag** and the corresponding **tag group** value. The value must be a string.
- **pool-priority priority** *value*: Specifies the order of IP chunk and IP allocation among pools with the same tag value. *value* must be the integer **0** or **1**. **0** is the default value, and has the highest priority..

Use the following commands to configure DHCP for pool association:

```
profile dhcp dhcp_profile_name
 ipv4
  server
   pool-group-tag tag_value
   lease hours hours_value
  exit
 exit
exit
```

NOTES:

- **profile dhcp** *dhcp_profile_name*: Specifies the DHCP profile name.
- **ipv4**: Enters IPv4 configuration mode.
- **server**: Enters server configuration mode.
- **pool-group-tag** *tag_value*: Specifies the group tag value that is used to associate the profile with the group tag defined in the pool.

IPAM Enhancements

This section lists the following IPAM enhancements.

IPAM Quarantine Timer

The IP quarantine logic enhancements are as follows:

- The maximum quarantine configuration is increased to 1 hour (Range: 4 to 3600 seconds).
- If the configured quarantine time is <= 15min, additional buffer of 60 seconds is added to the configured quarantine time.
- If the configured quarantine time is > 15min, additional buffer of 5 minutes is added to the configured quarantine time.
- Default quarantine time processing thread interval is changed from 5 to 60 seconds.
- The IP is moved to the free-list after ~(configured-qTime + buffer + delay-from-qt-thread-processing).
- Upon Node Manager pod restart, quarantine time of all older IPs in the quarantine time-queue is reset and will restart from beginning.
- After Node Manager pod restart, all IPs released as part of reconciliation are moved to the quarantine-queue before moving to the free-bitmap (this includes pre-reserved IPs).

Address-Range Level Quarantine

If an address-range is removed from the UPF after releasing all the IPs in a proper manner (that is, each released IP went through quarantine time) then the address-range is moved directly to free-list.

If an address-range is removed from the UPF due to the UPF-release with some of the addresses allocated, then the complete address-range is put under quarantine for the configured time and then moved to free-list.

The **show ipam pool** command displays quarantine-chunks with a special 'alloc-context'.

Pool and UPF Threshold Monitoring

The UPF threshold monitoring enhancements are as follows:

- **Upper threshold:** Default = 80%, configurable. This is used to add new chunks to the pool or UPF.
- **SafeCutOff:** Default = (upper-threshold-5%), not-configurable. After hitting upper-threshold, new chunks are allocated to the pool or UPF to bring down the current-utilization to safecutoff level , that is, upper-threshold – 5%.
- **Lower threshold:** Default = 90% of upper-threshold, not-configurable. This is used to remove a chunk from the pool or UPF.

Each Node Manager runs a pool level threshold monitoring. When a chunk is assigned to the UPF, the Node Manager checks the pool-threshold hit and reserves additional chunks from the cache-pod for future use.

For pool threshold calculation, the total number of IPs left in free-chunks are considered; not the actual number of allocated IPs on an assigned chunk. That is, after a chunk is assigned to the UPF, it is considered as fully used for pool-threshold monitoring purpose. A complete free address-range can be released back to the cache-pod based on lower-threshold calculation.

For UPF threshold monitoring, the actual number of total IPs and allocated IPs are considered; more chunks are reserved for the UPF when the upper-threshold hits. The Node Manager adds the route to the UPF whenever a new chunk is assigned to it due to the threshold hit. For performance reasons, the route is not deleted if it was added in at the last minute.

The upper threshold is configurable (default=80%), when this threshold hits, new chunks are added until the current-utilization falls back to the safe-cutoff level. That is, 75% is safe cutoff if the upper-threshold is 80%.

Lower threshold is 90% of the upper-threshold. Thati is, if the upper-threshold is 80%, then the lower-threshold is 72%, a chunk can be removed from the UPF only when the remaining threshold is below 72%. Otherwise, the chunk remains in the UPF assigned list. This logic is applied to avoid frequent route-add and route-delete operations around boundary condition. The UPF threshold monitoring is triggered during events such as address-allocate, address-release, and config-change. On idle-system, the behavior may differ, however, in a running system, the threshold calculation occurs regularly.

Marking a pool or address-range as offline overrides the lower-threshold logic. That is, if an offline chunk is completely free, it is removed from the UPF irrespective of the lower-threshold calculation.

Multiple Replica Handling

IPAM is part of the Node Manager (nodemgr) pod. A maximum of two nodemgr pods are supported per BNG cluster.

During UPF-registration, one of the nodemgr pod gets all the static-pool-routes for the UPF and all the dynamic-pool-routes from both the nodemgr pod if anything is allocated earlier and programs it.

During IP-allocation, the IPC request goes to one of the nodemgr pods. If no routes were assigned earlier, a new route is assigned and if successful, an IP is returned to FSOL. Even if one nodemgr pod goes down, the other nodemgr can handle IP-allocations, provided enough IPs are available. Chunks that are reserved by one nodemgr cannot be used by the other nodemgr for address-allocations.

During IP-release, the IPC request should go to the IP owner nodemgr as best-effort. If the IPC fails due, then the IP become stale on the IPAM. During nodemgr bring-up, the CDL reconciliation occcurs, which recovers the stale IPs. In addition, a new CLI is added **reconcile-ipam** to manually trigger IPAM-CDL reconciliation on a need basis. This command should be executed only during maintenance because it is a heavy operation.

During the UPF release, the N4 release comes to one of the nodemgrs. It sends an internal-IPC to the other nodemgr and both clean-up all the routes assigned to the respective UPF. If one of the nodemgr is down during that time, the other nodemgr takes over the content and releases the chunks on behalf of its peer.

IPAM Chunk-Optimization with Node Manager Pair for SRG

Table 6: Feature History

Feature Name	Release Information	Description
IPAM Chunk-Optimization with Node Manager Pair for SRG	2024.02.0	You can now optimize the allocation of chunks and hence avoid underutilization or wastage of chunks. With this feature, a Subscriber Redundancy Group is persistently associated with a designated Node Manager (NM), and chunks are allocated only to the associated NM. Earlier, subscribers across different groups were load-balanced between both NMs, and chunks were allocated for both NMs. This sometimes resulted in underutilization or wastage of chunks.

In large-scale clusters, chunks are distributed across groups and Node Managers (NMs) in a balanced manner. However, this can lead to inefficiencies, including underutilization and wastage of chunks.

With this feature, each group is persistently associated with a single NM. Chunks are then allocated only to the NM that is sticky to the associated group. This allocation strategy prevents the dispersion of chunks across NMs for a given SRG group, thereby addressing the issue of chunk underutilization and wastage.

When group configurations change, such as addition or deletion of groups, SRG updates the distribution of new groups across the NMs, while maintaining existing associations. If a group is deleted, its stickiness association is removed from the corresponding NM. However, existing mappings for other groups remain intact. When new groups are added, they are balanced among the available NMs, but existing group-NM associations remain unchanged.

Node Manager Responsibility and Failover Handling

The owner NM (pod) is the primary entity responsible for managing all Data Plane (DP) and IP-related activities for its associated groups. These critical functions include:

- Data Plane registration and deregistration
- IP address allocation and release
- Threshold monitoring

- Reconciliation processes
- Monitoring of quarantined chunks

When the active NM instance becomes unavailable, the peer NM takes over and handles requests on behalf of the owner NM instance.

Note The peer NM does not accept the very first subscriber for a new group/pool/AFI combination. Such requests are rejected, as initial subscriber handling is reserved for the owner NM.

To prevent data corruption and avoid rapid toggling of NM instance roles, a buffer period has been implemented before it becomes active on the peer NM. The peer NM resumes active responsibilities only if the owner NM has been down for more than 3 minutes. Upon becoming active for a remote NM that is down, the peer NM loads the IPAM state to manage any pending requests for the affected NM groups.

Limitations

Following are the known limitations of IPAM Chunk-Optimization with Node Manager Pair for SRG feature:

- The peer NM becomes active for other NM groups if the owner NM has been down for more than three minutes. This delay helps prevent data corruption during consecutive NM reloads.
- Only address allocation and new chunk reservation are supported.
- Reconciliation is not performed on the peer NM on behalf of the active NM if the active NM goes down. This may result in the IPAM system showing a higher number of IP addresses in use compared to the actual subscriber count. Discrepancies are resolved once the owner NM is back online and the IPAM performs reconciliation, syncing the IPAM counter with the session count.
- The peer NM cannot support the allocation of the very first IP address for a new subscriber under a specific combination of group, pool, and AFI. First-time IP allocations must be performed by the owner NM.
- Configuration changes to the IPAM are not supported when another NM is active on the peer NM. Any accidental configuration changes made during this period will only be processed once the owner NM is Up.
- If the route synchronisation is executed when one NM is down or its NM is active on peer NM, there might be inconsistencies in chunk programming towards the User Plane Function (UPF). The likelihood of this occurring is low, but if such a discrepancy is noticed, it can be corrected by executing the **route-sync** command after the owner NM is fully active.

Verification

Use the **show ipam dp** *peerid* **ipv4-address** to view if the groups and chunks are associated with any one NM.

```
bng# show ipam dp peer-x1 ipv4-addr
Wed Oct  11 19:13:22.877 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
==========================================================================================
33.0.0.0              33.0.1.255      33.0.0.0/23      1:0/-1   100.00%
automation-poolv4(automation-vrf)(asr9k-ACTIVE,asr9k-STANDBY)
33.0.6.0              33.0.7.255      33.0.6.0/23      1:0/-1   95.90%
automation-poolv4(automation-vrf)(asr9k-ACTIVE,asr9k-STANDBY)
33.0.10.0             33.0.11.255     33.0.10.0/23     1:0/-1   0.59%
automation-poolv4(automation-vrf)(asr9k-ACTIVE,asr9k-STANDBY)
44.44.1.0             44.44.1.255     44.44.1.0/24     1:N/A                  S
automation-poolv4(automation-vrf)(asr9k-ACTIVE,asr9k-STANDBY)
```

In the above example, the peer-x1 is associated with NM-0, and all chunks are associated only with NM-0.

Dynamic Chunk Allocation for CP-GR

Table 7: Feature History

Feature Name	Release Information	Description
Dynamic Chunk Allocation for CP-GR	2024.02.0	This feature provides you greater control over how chunks are managed across active and remote clusters. This feature introduces an optimized mechanism that eliminates the need for pre-reserving chunks on the remote cluster, thus conserving resources and improving overall system efficiency.

In the current CP-GR setup, a large number of chunks are reserved in advance for a remote cluster to ensure a smooth switchover in case of failover events. These chunks are used only when there is a switchover or when all chunks in the active cluster are exhausted, causing inefficient chunk reservation.

The Dynamic Chunk Allocation for CP-GR feature cancels the need to reserve additional chunks on the remote cluster. Instead, during a switchover event, the same chunks initially assigned to Network Modules (NMs) in the primary cluster are seamlessly reused in the remote cluster, ensuring a fluid transition without the need for pre-allocated chunk reserves. IPAM automatically performs reconciliation of all chunks during a switchover.

You can use the CLI command **reserve-chunk-for-remote-site { enable | disable }** to either select the traditional method of reserving chunks for the remote server or to adopt the new approach where local chunks are dynamically utilized by the remote site.

Configuration Example

The following is a sample configuration:

```
config
 ipam
  instance 1
   source local
   reserve-chunk-for-remote-site { enable | disable }
   address-pool SAMP_POOL_2048
    vrf-name FTTX-SUB-EAST
    ipv4
     split-size
      per-cache 32768
      per-dp    2048
     exit
     address-range 209.165.200.225 209.165.255.254
```

```
      address-range 209.165.201.1 209.165.201.30
     exit
    exit
   exit
  exit
```

NOTES:

- **reserve-chunk-for-remote-site enable**: Reserves chunks on the remote cluster.
- **reserve-chunk-for-remote-site disable**: Chunks are not reserved on the remote cluster. This is the default behavior.

Note The value of the **reserve-chunk-on-remote-site** CLI must not be altered dynamically. To enable or disable the **reserve-chunk-on-remote-site** command post initial IPAM configuration, shut down and restart the system. In case of a GR setup, both the clusters must be shut down and then restarted.

IPAM Route Programming Enhancements

Table 8: Feature History

Feature Name	Release Information	Description
IPAM Route Programming Enhancements	2024.04.0	IPAM now programs routes asynchronously, improving system stability and performance. IPAM sends route update requests and handles responses in separate routines, allowing continuous address allocation without delays.

Prior to Release 2024.04.0, IPAM programs routes toward UP in a synchronous manner. This means IPAM waits for a response from UP before processing any other requests for that AFI. If there is a delay in the route update response from UP, goroutines on the Node Manager (NM) start to pile up. Significant delays can cause the NM to crash. Also, new address allocations on IPAM remain blocked until the response is received.

Asynchronous Route Programming Implementation

Starting Release 2024.04.0, the IPAM is enhanced to program routes asynchronously. IPAM sends the route update request and waits for the response in a separate routine. Based on the response, IPAM moves the chunks to the proper state.

Scenarios for Route Programming

IPAM programs routes toward UP in two scenarios:

- When the first subscriber comes to IPAM for a given AFI on an NM.
- When there is a threshold hit for the UP or SRG-Group for that AFI on NM.

Scenario 1: First Subscriber

When the first subscriber comes to IPAM for a given AFI, the behavior remains unchanged. IPAM programs the route for that UP in a synchronous manner. This usually happens during the initial system setup, where delays are not expected. Since IPAM handles UP or SRG-Group registration during this flow, it remains synchronous for simplicity.

Scenario 2: Threshold Hit

When there is a threshold hit for a given UP or SRG-Group for an AFI, IPAM fetches new chunks from NM or Cachepod, and programs the route toward the UP or SRG-Group asynchronously. This ensures no impact on existing or new IP allocations.

No explicit configuration is required for this functionality. This feature becomes applicable as new routes are programmed for the UP. The system checks the PFCP retransmission timeout configuration, and if this configuration is set, the route retry is done based on the configured timeout. Otherwise, by default, the retry occurs after 15 seconds plus a 5-second buffer.

The following is a sample PFCP retransmission timeout configuration:

```
instance instance-id 1
 endpoint n4-protocol
  retransmission timeout 15 max-retry 1
 exit
exit
```

Pre-Allocation of Gateway IP and Address Chunks

Table 9: Feature History

Feature Name	Release Information	Description
Pre-Allocation of Gateway IP and Address Chunks	2024.04.0	This feature ensures a smoother and more efficient onboarding process for new subscribers by reserving the first IP address in the allocated chunk for gateway functionalities.

Gateway IP Allocation

In a routed network configuration, the subscriber's gateway IP resides at the first Layer 3 hop on the access side router. The gateway IP is the first IP address within the allocated chunk or address range for the access side router. This first IP address is reserved and will not be assigned to any subscribers. Reserving this IP ensures that it is consistently available for gateway functionalities.

Address Chunk Pre-Allocation

Before onboarding the first subscriber, it is necessary to pre-allocate the address chunk and the first IP address within this chunk. This pre-allocation allows the first IP to be programmed on the access side router in advance, ensuring that the network infrastructure is prepared before any subscribers connect. Pre-allocating the chunk is mandatory for session bring up on routed SRG groups.

Administrative Command for Pre-Allocation

To support this pre-allocation process, we have introduced a new administrator-triggered command, **ipam-address-chunk allocate**. This command enables the pre-allocation of the address chunk and the gateway IP. Administrators must use this command whenever a new access interface or port is onboarded. If there are multiple BNG access interfaces on the same access router, pre-allocation must be performed multiple times.

Address Chunk Release

You can also release an IP address chunk when the data plane (DP) is released, particularly after bringing down subscribers. During the removal of an SRG group, it is mandatory to follow the Method of Procedure (MOP), which includes bringing down all subscribers in the group and executing an action command (**ipam-address-chunk release**) to release the DP and its associated chunks. If the action command is used to release an address-chunk without bringing down subscribers, the DP is unregistered, and the chunk is moved to quarantine. The **ipam-address-chunk allocate** command is intended for the initial IP reservation; subsequent chunk allocations for the DP occurs only when the IP usage threshold is reached. If you execute the action command a second time, the command execution will succeed, but no additional chunk will be allocated.

Configure Pre-Allocation of Gateway IP and Address Chunks

Procedure

Step 1 Use the **ipam-address-chunk** action command to configure pre-allocation of Gateway IP and address chunks.

Example:

```
ipam-address-chunk { allocate | release { [ pool-group-tag value | pool-name name
]}
     { address-type [ ipv6-addr | ipv6-prefix | ipv4 | ipv6 ] }
      [ ipam-dp-key dp_key ]
    [ gr-instance  gr_instance ]
      [ srg-peer-id  srg_peer_id ]
```

NOTES:

- **allocate**: Enables pre-allocation of IP address chunk and the gateway IP.
- **release**: Releases an IP address chunk.
- **pool-group-tag / pool-name**: Provide either the pool-name or the pool-group-tag, and this should match the pool information configured in the DHCP profile.
- **address-type**: Specify one of the four possible address types:
  - ipv6-addr
  - ipv6-prefix
  - ipv4
  - ipv6

  Note
  When the **ipam-address-chunk** action command is executed with the **ipv6** address type, IPAM checks for a pool configured with **split-prefix-iana-first** enabled and allocates both IANA and IAPD from the same prefix. If no such pool is found, an error is returned.

  This is a mandatory parameter.
- **ipam-dp-key**: Specifies the data plane key for IP management. This is a mandatory parameter.

- **Routed SRG case**: Indicates the value of **ipam-dp-key** specified in the DHCP pool. Currently, circuit-id is supported. Essentially, use the value of circuit-id set on the access-side OLT.
- **Non-Routed SRG case**: The **ipam-dp-key** can either be the same as the srg-peer-id or it can be different.
- **Non-Routed Non SRG case**: This scenario is not supported currently.
- **Routed Non SRG case**: This scenario is not supported currently.

- **gr-instance**: Specifies the GR instance information. If not provided, the local gr-instance is used as the default value. This is a mandatory parameter.
- **srg-peer-id**:The SRG group peer-id as specified in the configuration. This is a mandatory parameter.

The output of this action command provides information about the chunk and the first IP address that were reserved. For example,

```
bng# ipam-address-chunk allocate instance-id 1 pool-name dhcp-ipv6-iapd ipv6-prefix ipam-dp-key
INGJRJKTMDHRTW6001ENBESR001 srg-peer-id Peer1
Sat Aug  24 06:27:29.200 UTC+00:00
result
Gateway Address: 2001:DB8::1/50
```

Step 2 Use the **show ipam { dp | dp-tag }** *value***{ ipv6-addr | ipv6-prefix | ipv4-addr }** to view the reserved IP address and the summary route of the allocated chunks. This information is useful for identifying the first IP address that needs to be configured on the access side router.

Example:

```
 show ipam dp INGJRJKTMDHRTW600TB2DEVICE11101 ipv6-addr

==================================================================================================


Flag  Indication: S(Static) O(Offline) R(For Remote Instance) RF(Route Sync Failed) F(Fixed Chunk
for DP)

Other Indication: A+(Waiting for route update response) QT*(Quarantined due to route delete failure)


                  QT+(Waiting for route update response post timeout)

G:N/P Indication: G(Cluster InstId) N(Native NM InstId) P(Peer NM InstId)

==================================================================================================


StartAddress                  EndAddress                 Route                    GatewayAddress
       G:N/P    Utilization   Flag      AllocContext

==================================================================================================

2001:DB8::8000                    2001:DB8::bfff               2001:DB8::8000/114
2001:DB8:8001/114           1:1/-1   0.01%         F         dhcp-ipv6-iana-11(FTTX_SUB)

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

IANA and IAPD Allocation from Same IP Range

Table 10: Feature History

Feature Name	Release Information	Description
IANA and IAPD Allocation from Same IP Range	2025.01.0	You can now reduce the route count by using a single summary route for both Internet Assigned Numbers Authority (IANA) and Internet Address Prefix Delegation (IAPD) from a single IP pool. This feature optimizes the use of IPv6 prefix ranges and offers flexibility in address allocation. By introducing a virtual address range within the IPAM data structure, you can reserve the first prefix for IANA and use subsequent prefixes for IAPD with the **split-prefix-iana-first** command.

The IANA and IAPD Allocation from Same IP Range feature allows the allocation of both Internet Assigned Numbers Authority (IANA) and Internet Address Prefix Delegation (IAPD) from a single IP pool, specifically an IPv6 prefix range. The primary objective is to optimize the use of IP address spaces and reduce the route count by using a single address scope for both IANA and IAPD in a subscriber redundancy group (SRG). This feature introduces a virtual address range within the IPAM data structure to facilitate this allocation.

The feature enables the use of a single IPv6 subnet or summary route for both IANA and IAPD, which assists in reducing the route count. When the IPAM configuration includes the **split-prefix-iana-first** command, the first prefix in the allocated range is reserved for IANA, while the subsequent prefixes are used for IAPD.

For IANA, the number of addresses assigned to a dpKey is fixed at 65,536. Even if the usage threshold is reached, no additional IANA addresses can be allocated.

Restrictions for IANA and IAPD Allocation from Same IP Range

These restrictions apply to the IANA and IAPD Allocation from Same IP Range feature:

- Once **split-prefix-iana-first** is configured for a pool, it cannot be removed unless the entire pool configuration is deleted.
- When an IPv6 address range or prefix range is configured, you cannot dynamically enable **split-prefix-iana-first**. To change the pool behavior to **split-prefix-iana-first**, you must mark all IPv6 address ranges or prefix ranges offline, delete them, and then make the configuration change.
- IPAM dpKey registration or release using the action command for the address type **ipv6** is only possible for pools configured with **split-prefix-iana-first** using either a pool name or group name. Conversely, you cannot use the address types **ipv6-addr** or **ipv6-prefix** for pools that have **split-prefix-iana-first** configured.
- The option to mark a pool as offline is disabled when **split-prefix-iana-first** is configured. Currently, you cannot mark an address range as offline with this setting enabled. If an incorrect address range is configured, it must be deleted directly, although this is not the recommended method of operation.

Note If an address range, whose first prefix is assigned to IANA, is marked offline, IPAM will not be able to assign a new chunk to IANA.

Configure IANA and IAPD Allocation from Same IP Range

Procedure

Step 1 Use the **split-prefix-iana-first** command under IPv6 settings to enable the feature.

Example:

```
config
  ipam
   instance instance_id
    address-pool pool_name
     vrf-name vrf_name
     ipv4
      split-size
       per-cache value
       per-dp value
      exit
       address-range start_ipv4_address end_ipv4_address
     exit
     ipv6
      split-prefix-iana-first
      prefix-ranges
       split-size
        per-cache value
        per-dp value
      exit
      prefix-range prefix_value prefix-length length
     exit
    exit
   exit
  exit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **instance** *instance_id*: Specifies the IPAM instance and enters the instance sub-mode. instance_id must be an integer. The valid value ranges from 1 to 8.
- **address-pool***pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **vrf-name** *vrf_name*: Specifies the name of the VPN routing and forwarding (VRF) for the pool.

- **ipv4**: Enters the IPv4 mode of the pool.
- **split-size { per-cache** *value* **| per-dp** *value* **}**: Specifies the size of the IPv4 range to be split for each IPAM cache allocation. The IPAM server consumes this configuration. The **no** form of this command disables the splitting of the address-ranges into smaller chunks.

  per-cache *value*: Specifies the size of the IPv4 range to be split for each Data-Plane (User-Plane) allocation. The valid values range from 2 to 262144. The default value is 1024.

  The IPAM cache consumes this configuration.
- **per-dp** *value*: Specifies the size of the IPv4 range to be split for each Data-Plane (User-Plane) allocation. The valid values range from 2 to 262144 The default value is 256.
- **address-range** *start_ipv4_address end_ipv4_address*: Configures the IPv4 address range with the starting and ending IPv4 address.
- **ipv6**: Enters the IPv6 mode of the pool.
- **split-prefix-iana-first**: Enables allocation of both IANA and IAPD from the same IP pool.
- **prefix-ranges**: Enters the prefix ranges mode.
- **prefix-range** *prefix_value* **prefix-length** *length*: Configures the IPv6 prefix range. *prefix_value* specifies the IPv6 prefix range.

  prefix-length *length* specifies the IPv6 prefix length.

Step 2 Use the IPAM show commands to check the allocation status and utilization of IPv6 addresses and prefixes.

Example:

1

```
bng# show ipam dp INMUNVMBGNSMNB0044FSAOLT001

Thu Dec  5  07:27:53.820 UTC+00:00
--------------------------------------------------------
 Ipv4Addr   [Total/Used/Reserved/Utilization] = 0 / 0 / 0 / 0.00%
 Ipv6Addr   [Total/Used/Reserved/Utilization] = 65536 / 0 / 2 / 0.00%
 Ipv6Prefix [Total/Used/Reserved/Utilization] = 256 / 0 / 1 / 0.39%
 Instance ID                         = 1
 SRG PeerID                          = Peer1
 L3Routed                            = true
--------------------------------------------------------
```

In this example, **65536** indicates the total address count for virtual IANA.

Example:

2

```
bng# show ipam dp INMUNVMBGNSMNB0044FSAOLT001 ipv6-addr

Thu Dec  5  07:27:57.929 UTC+00:00
====================================================================================================
Flag  Indication: S(Static) O(Offline) R(For Remote Instance) RF(Route Sync Failed) F(Fixed Chunk
for DP) V(Virtual)
Other Indication: A+(Waiting for route update response) QT*(Quarantined due to route delete failure)

                  QT+(Waiting for route update response post timeout)
G:N/P Indication: G(Cluster InstId) N(Native NM InstId) P(Peer NM InstId)
====================================================================================================
```

```
StartAddress         EndAddress      Route                 GatewayAddress     G:N/P    Utilization
  Flag      AllocContext
=================================================================================================
2001:DB8::8000      2001:DB8::bfff  2001:DB8::8000/114    2001:DB8:8001/114   1:0/-1     0.01%
   V    group-naoverpd5(automation-vrf)
=================================================================================================
```

Flag **V** - indicates a virtual address range for IANA.

Static IP Support

Feature Summary and Revision History

Summary Data

Table 11: Summary Data

Applicable Product(s) or Functional Area	cnBNG
Applicable Platform(s)	SMI
Feature Default Setting	Disabled - Configuration Required
Related Changes in this Release	Not Applicable
Related Documentation	*Cloud Native BNG Control Plane Command Reference Guide*

Revision History

Table 12: Revision History

Revision Details	Release
Multiple UPFs support with a static IP-pool.	2024.01.0
First introduced.	2021.03.0

Feature Description

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

The Static Pool Enhancements provide more functionalities on the cnBNG using IPAM. These functionalities are as follows:

- Supports addition of a new static-pool or static-addr-range dynamically when system is running.

- Send routes when User Plane function (UPF) is already associated.

- Supports removal of a new static-pool or static-addr-range dynamically when system is running
  - Mark the static-pool or static-addr-range offline.
  - Clear subscribers, manually.
  - Delete the configuration. IPAM sends the route-delete command to the UPF if it is already associated.

- The cnBNG Control Plane now supports sharing a single static IP-pool with multiple UPFs.
  - To enable sharing of the same static IP-pool across multiple UPFs, you must configure the static IP-pool as a Global Static pool. Global Static pool here refers to a static pool that is used by the IPAM module for sessions with IP addresses set from an external entity such as RADIUS, and not bound to any UPF.

    For configuration details, see Configuring a Static IP-pool to Support Multiple UPFs.

Note Changing the UPF name directly in the static IP pool is not supported. To change the UPF name, gracefully delete the static IP pool and readd the new UPF.

Note the following guidelines while configuring a static IP pool:

- An address pool is marked as ‘static’ during configuration. A given address pool can be either configured as 'dynamic' or 'static' mode only. It cannot be modified while the system is running.
- Each static pool is associated to an User Plane function (UPF). The IPAM configures all the address-ranges as routes on the UPF. The recommendation is to use the ‘no-split’ configuration to avoid having too many splits and routes.
- The IPV4 address ranges must be configured with the ‘default-gateway’ value, because unlike dynamic address-range, IPAM cannot assign ‘default-gateway’ on its own. Also, the static address-ranges must not be split further. Therefore, use the ‘no-split’ configuration.

Limitations

Support of multiple UPFs with a Static IP-pool feature has the following limitations:

- This feature is supported only for PPPoE subscribers.
- You must not dynamically change Global Static-Pool to non-Global Static-Pool (UPF specific static pool), or the reverse.
- You may encounter the following scenario when you want to delete an IPv4 address range in a static pool:

  Consider the following sample configuration.

```
config
  ipam
    instance 1
      address-pool old-static-pool-ft
        vrf-name vrf_lps
        static enable
```

```
          ipv4
            split-size
              no-split
            exit
            address-range 209.165.200.225 209.165.200.254 offline default-gateway
209.165.200.225
            address-range 209.165.201.1 209.165.201.30 default-gateway 209.165.201.1
           address-range 209.165.202.129 209.165.202.158 default-gateway 209.165.202.129
```

If you want to delete the address range that is marked offline, and if you run the command `bng(config-ipv4)#` **`no address-range 209.165.200.225 209.165.200.254 default-gateway 209.165.200.225`**, only the default gateway address is deleted, and the address range is not deleted.

To delete the address range, you must run the command as follows:

`bng(config-ipv4)#` **`no address-range 209.165.200.225 209.165.200.254`**

Note To delete the address range, ensure that **default-gateway** *ip_address* is not included in the command.

Configuring Static IP Support

This section describes how to configure the Static IP support feature.

Configuring the Static IP Support involves the following step.

Configuring Static IPv4 Address Ranges, on page 32

Configuring Static IPv4 Address Ranges

Use the following configuration to configure the Static IPv4 address ranges.

```
config
   ipam
      instance instance_id
         source { local | external }
         address-pool pool_name
         vrf-name vrf_name
         static enable user_plane_name
         ipv4
            split-size no-split
            exit
         ipv4
         address-range start_ipv4_address end_ipv4_address
            default-gateway ipv4_address
            exit
         exit
      exit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.

- **address-pool***pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **vrf-name** *vrf_name*: Specifies the name of the VPN routing and forwarding (VRF) for the pool.
- **static enable** *user_plane_name*: Configures static IP details. Sets the specified User Plane (UP) as static. *user_plane_name* is the specified UP for this static pool.
- **ipv4**: Enters the IPv4 mode of the pool.
- **split-size no-split**: Specifies that the address-ranges should not be into smaller chunks.
- **address-range** *start_ipv4_address end_ipv4_address*: Configures the IPv4 address range with the starting and ending IPv4 address.
- **default-gateway** *ipv4_address*: Specifies the IPv4 address of the default gateway.
- IPAM does only route-validation for Static IP. Validate if UPF has a route for a given static IP address or prefix. Individual address-management is not performed.

Configuring a Static IP-pool to Support Multiple UPFs

Use the following configuration to configure a static IP-pool to support multiple UPFs.

```
config
  ipam
    instance instance_id
      address-pool pool_name
        vrf-name string
        static enable
        exit
```

NOTES:

- **ipam**: Enters the IPAM configuration mode.
- **address-pool** *pool_name*: Configures the address pool configuration. *pool_name* must be the name of the address pool.
- **vrf-name** *vrf_name_value*: Specifies the name of the VPN routing and forwarding (VRF) for the pool.
- **static enable** : Enables sharing of the same static IP pool across multiple UPFs (global static pool) and SRGs.

Static IP Mapping Database

Feature Summary and Revision History

Summary Data

Table 13: Summary Data

Applicable Product(s) or Functional Area	cnBNG
Applicable Platform(s)	SMI
Feature Default Setting	Disabled - Configuration Required
Related Documentation	Not Applicable

Revision History

Table 14: Revision History

Revision Details	**Release**
First introduced.	2022.02.0

Feature Description

The centralized IPAM component of the cnBNG CP enables mapping of static IPs for priority or VIP subscribers. The Static IP Mapping Database feature supports this requirement. It enables mapping of the MAC address of the subscriber to the static IP address. The static IP mapping database is maintained in the cnBNG CP, which ensures that the particular VIP subscriber receives the mapped static IP address every time the subscriber connects.

Maintaining the static IP mapping database is the key function of this feature.

- Each static IP mapping entry is pushed via the CLI, NETCONF, or RESTCONF interfaces.
- Given the number of entries, the static IP mapping entry cannot be stored as a configuration in CONFD. Instead, mapping values are directly stored in IPAM. Therefore, the **show running config** CLI command does not display these entries.
- If a subscriber receives a framed IP address from RADIUS, this IP address is given priority over the static IP mapped in IPAM and is assigned to the subscriber. However, if RADIUS does not allocate any address and the DHCP profile has a static IP mapping configuration enabled (see, Configuring Static Database Key in DHCP Profile, on page 39), then it attempts to fetch the IP for this subscriber from the static IP mapping database.

> **Note**
> - A copy of static IP address mapping entries is stored in volume-mounted ETCD database for handling pod restart and image upgrades.
> - The static IP mapping database remains persistent across the system shutdown and running mode.

How it Works

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

The DHCP profile is configured with the *subscriber-key-template* that is used for IP mapping search. The DHCP forms the *subscriber-key* based on the DHCP profile configuration and incoming packet, and then forwards the request to the IPAM.

The IPAM searches the static database based on the *subscriber-key* input, finds the mapped IP address, and responds to the DHCP.

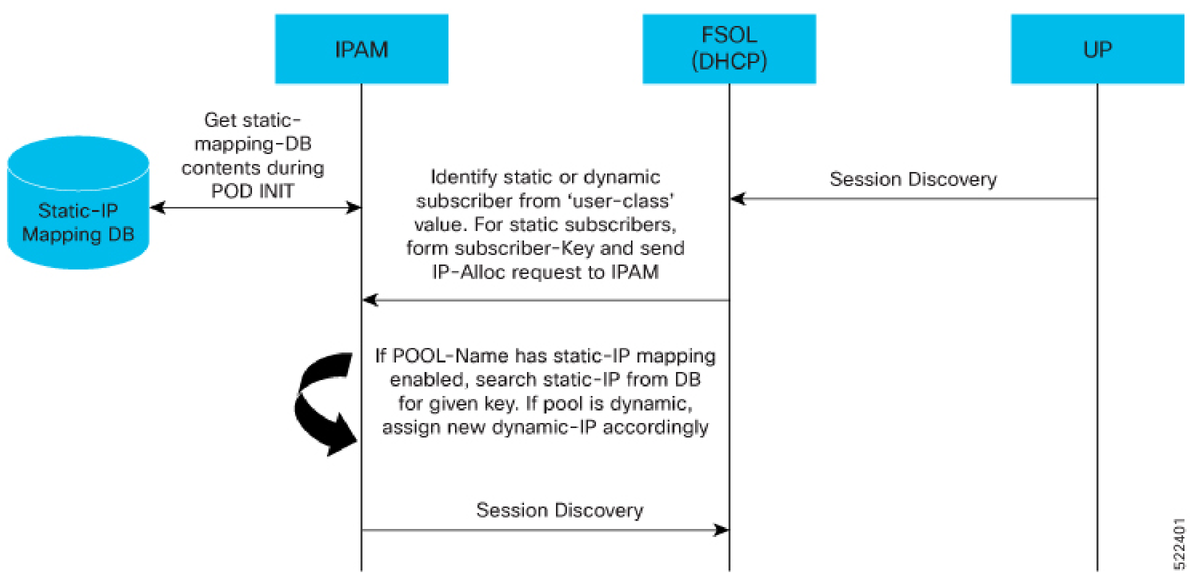
If the IP static database does not have an entry for the *subscriber-key* in the database, the IPAM returns a failure to the DHCP. This causes the DHCP to fail the session.

Call Flows

This section includes the following call flow.

> **Note** The Static IP Mapping Database feature is supported only for IPoE (DHCP) sessions.

Figure 4: Static IP Mapping Call Flow



Configuring the Static IP Mapping Database

This section describes how to configure the Static IP Mapping Database feature.

Configuring the Static IP Mapping Database feature involves the following procedures.

- Configuring Static IP Mapping Database in IPAM, on page 36
- Configuring Static Database Key in DHCP Profile, on page 39

Configuring Static IP Mapping Database in IPAM

Use the following CLI commands to add or delete entries in the static IP mapping database.

```
ipam-staticdb mapping { pool pool_name
{ ipv4 { address ipv4_address subkey subkey }
| ipv6-addr { address ipv6_address subkey subkey }
| ipv6-prefix { address ipv6_prefix_address subkey subkey }
| delete }
```

NOTES:

- **pool** *pool_name*: Specifies the pool name.
- **ipv4 address** *ipv4_address* **subkey** *subkey*: Specifies the IPv4 address and subkey.
- **ipv6-addr address** *ipv6_address* **subkey** *subkey*: Specifies the IPv6 address and subkey.
- **ipv6-prefix address** *ipv6_prefix_address* **subkey** *subkey*: Specifies the IPv6 prefix address and subkey.
- **delete**: Deletes the static IP mapping database.

The static IP mapping database in CP can be in following format.

```
staticMapping:
  - poolName: <pool-name1>
    ipv4List:
     - <subscriber-key1>
       <static-ip-address1>
     - <subscriber-key2>
       <static-ip-address2>
 - poolName: <pool-name2>
    ipv4List:
     - <subscriber-key1>
       <static-ip-address1>
     - <subscriber-key2>
       <static-ip-address2>
```

Note

- The subscriber-key is defined based on the deployment need. For example, the subscriber-key can be the MAC address of the subscriber or a combination of multiple identifiers with some delimiters.
- For static IP key mapping, the only supported identifier is the MAC address.
- The IP address can be IPv4, IPv6-NA, or IPv6-PD.

Example

The following are the CLI and RESTCONF configuration examples.

Sample Output: CLI

The following is a sample CLI of the static IP mapping database.

```
bng# ipam-staticdb mapping pool { pool-name static-pool ipv4 { subkey aa11.0000.0001 address
 20.20.0.5 } ipv4 { subkey aa11.0000.0002 address 20.20.0.6 } ipv6-addr { subkey
aa11.0000.0001 address 2001:cb0::0:1 } ipv6-prefix { subkey aa11.0000.0001 address
2001:ef0::/48 } }
mapping-details
Static IP Mapping Success
```

Sample Output: RESTCONF

The following is a sample RESTCONF of the static IP mapping database.

```
curl -i -k -X "POST"
"https://restconf.bng-ops-center.10.84.102.254.nip.io/restconf/data/cnbng-static-map:ipam-static-db/mapping"
 -H 'Content-Type: application/yang-data+json' -H 'Accept: application/yang-data+json' -u
'admin:*****' \
-d $'{"pool":{"pool-name":"test1",
"ipv4": [
{"subkey": "0000.1111.2221","address": "1.1.1.1"}
{"subkey": "0000.1111.2222","address": "1.1.1.2"}
]}}'
```

Note The following caveats are applicable when configuring the static IP mapping database:

- Data is not validated for admininstrator-configured static database. However, input data is accepted as-is and stored. It is the responsibility of the adminstrator to configure and map valid IP addresses to the subscribers.

  For example, validation of address formats and duplicate IP addresses across subscribers is not supported.

- The following soft limits are applicable on the number of entries per request via the CLI or RESTCONF.
  - Number of static IP mapping entries per CLI request is 100.
  - Number of static IP mapping entries per RESTCONF request is 1000.

  Any number that exceeds more than the specified limits can be time consuming or lead to indeterministic behaviour.

Verifying Static IP Mapping Database in IPAM

Use the following **show** commands to verify the administrator-specified IP to MAC mapping.

- Number of mappings per AFI (IPv4, IANA, IAPD) level.

```
bng# show ipam-staticdb
staticdb-info
{
"staticdbInfo": [
{
"poolName": "static-pool",
"ipv4Total": 7,
"ipv6AddrTotal": 7,
```

```
"ipv6PfxTotal": 4
}
]
}
```

- Number of mappings per pool, per AFI level.

```
bng# show ipam-staticdb pool-name static-pool
staticdb-info
{
"staticdbInfo": [
{
"poolName": "static-pool",
"ipv4Total": 7,
"ipv6AddrTotal": 7,
"ipv6PfxTotal": 4
}
]
}
```

- MAC to IP mapping information.

```
bng# show ipam-staticdb pool-name static-pool ipv4-addr start-index 3
staticdb-info
{
"staticdbInfo": [
{
"poolName": "static-pool",
"ipv4Total": 7,
"ipv6AddrTotal": 7,
"ipv6PfxTotal": 4,
"ipv4Range": [
{
"subkey": "aa11.0000.0003",
"subip": "20.20.0.3"
},
{
"subkey": "aa11.0000.0004",
"subip": "20.20.0.4"
},
{
"subkey": "aa11.0000.0005",
"subip": "20.20.0.5"
},
{
"subkey": "aa11.0000.0006",
"subip": "20.20.0.6"
},
{
"subkey": "aa11.0000.0007",
"subip": "20.20.0.7"
}
]
}
]
}
```

Note

By default only 1000 entries are displayed per index-based **show** command.

For example:

- When start-index is 100, entries from 100 to 1099 are displayed.
- When start-index is 1000, entries from 1000 to 1999 are displayed and so on.

Configuring Static Database Key in DHCP Profile

Use the following CLI commands to configure the static database key in the DHCP profile.

```
config
   profile dhcp dhcp_profile_name
      ipv4
         { class class_name | mode server | server }
          server static-ip-key identifier client mac_address client_mac_address

            exit
         exit
      exit
   exit
```

NOTES:

- **profile dhcp** *dhcp_profile_name*: Specifies the DHCP profile name.
- **ipv4**: Enters IPv4 configuration mode.
- **{ class** *class_name* **| mode | server }**: Enters Class, Mode Server, or Server configuration mode.
- **server static-ip-key identifier client mac_address** *client_mac_address*: Specifies the static-ip-key identifier and client MAC address. *client_mac_address* must be in Client MAC Address in AABB.CCDD.EEFF format.

The static database key in the DHCP profile can be in the following format:

```
profile dhcp dhcp-server1
 ipv4
  mode server
  server
   pool-name   pool-ISP
   dns-servers [ 8.8.8.8 ]
   lease days 10
   lease hours 0
   lease minutes 3
   static-ip-key identifier client-mac-address
  exit
  class c1
   server
    pool-name static-pool
    static-ip-key identifier client-mac-address
   exit
  exit
 exit
 ipv6
  mode server
```

```
  server
   iana-pool-name static-pool
   lease days 10
   lease hours 0
   lease minutes 4
   static-ip-key identifier client-mac-address
  exit
  class c2
   server
    iapd-pool-name static-pool
    static-ip-key identifier client-mac-address
   exit
  exit
 exit
exit
```