Subscriber Manager

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	*Cloud Native BNG Control Plane Command Reference Guide*

Revision History

Table 2: Revision History

Revision Details	Release
Introduced support for Radius-Based Policy (RaBaPol)	2025.02.0
Introduced support for Shared Policy Instance (SPI)	2025.02.0
Introduced support for session disconnect history.	2025.01.0
Enhancement Introduced: The Subscriber Manager feature is NSO-integrated.	2021.04.0
The following features are supported: • Automatic Session Reconciliation • Framed Route Support • Subscriber QoS Policy	2021.03.0
First introduced.	2021.01.0

Feature Description

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

In the Subscriber Manager (SM) context, a subscriber is a binding between the cnBNG Control Plane (CP) and a single subscriber end device. The SM is designed to provide a generic mechanism to connect edge subscribers to services enabling features. Subscribers are identified, authenticated, authorized, and accounted for in the SM.

Note The Subscriber Manager is also referred to as the Session Manager.

The following is a high-level list of the SM functionalities:

- Provides a generic mechanism for different Broadband Access Protocols such as DHCP and PPPoE.
- Provides an interface with off-box Radius servers using policy-plane to meet protocol and network provisioning requirements.
- Supports different subscriber lifecycle events such as CoA, idle timeout processing, and periodic reauthorization.

- Provides support for configuring subscriber lifecycle events that help customer define the subscriber behavior for the different subscriber lifecycle events.
- Derives per subscriber configuration from multiple sources.
- Maintains the subscriber state and subscriber configuration in a centralized session database.
- Interacts with the User Plane (UP) for subscriber session creation and subscriber feature configurations.

Subscriber features that are configured on cnBNG enable service providers to deploy certain specific functionalities like restricting the use of certain network resources, allowing Law Enforcement Agencies (LEAs) to conduct electronic surveillance, and so on.

Subscriber Features

The cnBNG supports the following subscriber features on the UP. For details, see the latest version of the Broadband Network Gateway Configuration Guide for Cisco ASR 9000 Series Routers listed here: https://www.cisco.com/c/en/us/support/routers/asr-9000-series-aggregation-services-routers/products-installation-and-configuration-guides-list.html.

- IPv4 or IPv6
  - Maximum Transmission Unit (MTU)
  - Unicast Reverse Path Forwarding (URPF)
  - Internet Control Message Protocol (ICMP)
- Access Control List (ACL)
  - Input ACL (IPv4 or IPv6)
  - Output ACL (IPv4 or IPv6)
- QoS (Quality of Service)
  - Input (policing)
  - Output (policing, shaping)
  - Policy merging (up to 6 policy maps and 10 class maps, including the default)
- Policy-based Routing (PBR)
  - Input policy (HTTP redirect)
- Accounting
  - Session Accounting
    - Periodic accounting
  - Service Accounting
    - Periodic accounting

To configure subscriber features, see Configuring Subscriber Manager Features, on page 4.

How it Works

This section provides a brief about how the Subscriber Manager works.

The SM functionality is hosted in a SM pod having one container in it. The SM pod communicates with the BNG Ops Center, policy-plane, and PFCP-EP pods using the APP infrastructure inter-pod communication (IPC).

The Subscriber Microservices Infrastructure (SMI) instantiates the SM pod. There can be more than one SM pod in the cluster. Each SM pod instance is independent. The per subscriber data is stored in a centralized database such that any SM pod can access this data.

Configuring Subscriber Manager Features

This section describes how to configure Subscriber Manager features on the CP.

The configuration of the Subscriber Manager features involves the following procedures:

- Configuring the HTTPR Policy Name, on page 4
- Configuring IPv4 Options, on page 5
- Configuring IPv6 Options, on page 5
- Configuring QoS Parameters
- Configuring the VRF Name, on page 6
- Configuring a Subscriber Profile, on page 7

Note

- To configure PPP feature options, see Creating the PPP Feature Template
- To configure service accounting, see Configuring Service Accounting, on page 18
- To configure session accounting, see Configuring Session Accounting, on page 18

Configuring the HTTPR Policy Name

Use the following commands to configure the Policy Based Routing (PBR) HTTP Redirect (HTTPR) policy name.

```
config
  profile feature-template feature_template_name
  httpr-policy httpr_policy_name
  exit
```

NOTES:

- **profile feature-template** *feature_template_name*: Specifies the profile feature template name.
- **httpr-policy** *httpr_policy_name*: Specifies the PBR HTTPR policy name. The *httpr_policy_name* value can range from 1 to 128 characters.

Configuring IPv4 Options

Use the following commands to configure IPv4 options.

```
config
  profile feature-template feature_template_name
  ipv4
    disable-unreachables
    egress-acl string
    ingress-acl string
    mtu mtu_bytes
    verify-unicast-source reachable-via-rx
    exit
```

NOTES:

- **profile feature-template** *feature_template_name*: Specifies the profile feature template name.
- **ipv4**: Enters the IPv4 Configuration mode to configure the IPv4 features.
- **disable-unreachables**: Disables sending the Internet Control Message Protocol (ICMP) Unreachable messages.
- **egress-acl** *string*: Specifies the IPv4-based egress Access Control List (ACL) list. The supported length of the *string* ranges from 1 to 128 characters.
- **ingress-acl** *string*: Specifies the IPv4-based ingress ACL list. The supported length of the *string* ranges from 1 to 128 characters.
- **mtu** *mtu_bytes*: Specifies the maximum transmission unit (MTU). The supported *mtu_bytes* value can range from 68 to 65535 bytes.
- **verify-unicast-source reachable-via-rx**: Enables per packet validation for unicast. The source is reachable via the interface on which packet is received.

Configuring IPv6 Options

Use the following commands to configure IPv6 options.

```
config
  profile feature-template feature_template_name
  ipv6
    disable-unreachables
    egress-acl string
    ingress-acl string
    mtu mtu_bytes
    verify-unicast-source reachable-via-rx
    exit
```

NOTES:

- **profile feature-template** *feature_template_name*: Specifies the profile feature template name.
- **ipv6**: Enters the IPv6 Configuration mode to configure the IPv6 features.

- **disable-unreachables**: Disables sending the Internet Control Message Protocol (ICMP) Unreachable messages.
- **egress-acl** *string*: Specifies the IPv6-based egress Access Control List (ACL) list. The supported length of the *string* ranges from 1 to 128 characters.
- **ingress-acl** *string*: Specifies the IPv6-based ingress ACL list. The supported length of the *string* ranges from 1 to 128 characters.
- **mtu** *mtu_bytes*: Specifies the maximum transmission unit (MTU). The supported *mtu_bytes* value can range from 68 to 65535 bytes.
- **verify-unicast-source reachable-via-rx**: Enables per packet validation for unicast. The source is reachable via the interface on which packet is received.

Configuring QoS Parameters

Use the following commands to configure the Quality of Service (QoS) parameters.

```
config
  profile feature-template feature_template_name
  qos
    in-policy qos_input_policy_name
    merge-level integer
    out-policy qos_output_policy_name
    exit
```

NOTES:

- **profile feature-template** *feature_template_name*: Specifies the profile feature template name.
- **qos**: Enters the QoS Configuration mode to configure the parameters.
- **in-policy** *qos_input_policy_name*: Specifies the QoS input policy name. The supported length of the *qos_input_policy_name* ranges from 1 to 128 characters.
- **merge-level** *integer*: Enables or disables the merge level. A merge value of 0 disables the merge-level. Any value greater than 0, enables the merge level.
- **out-policy** *qos_output_policy_name*: Specifies the QoS output policy name. The supported length of the *qos_output_policy_name* ranges from 1 to 128 characters.

Configuring the VRF Name

Use the following commands to configure the virtual routing and forwarding (VRF) name.

```
config
  profile feature-template feature_template_name
  vrf-name vrf_name
  exit
```

NOTES:

- **profile feature-template** *feature_template_name*: Specifies the profile feature template name.

- **vrf-name** *vrf_name*: Specifies the VRF name. The supported length of the *vrf_name* ranges from 1 to 128 characters.

Configuring a Subscriber Profile

Use the following commands to create a subscriber profile.

```
configure
  profile subscriber subscriber_profile
    aaa { authenticate aaa_profile_for_authentication |
        authorize aaa_profile_for_authorization }
    activate-feature-template feature_template_name
    apply-all-class
    class class_name
      aaa aaa_profile_for_authentication | authorize aaa_profile_for_authorization
      activate-feature-template  feature_template_name
      matches
        match { protocol { dhcp | ppp } } | username { ascii
              ascii_string  | regex reg-exp string}
              | source-mac { ascii ascii_string
              | regex reg-exp string } |
              circuit-id { ascii  ascii_string
              | regex reg-exp string } |
              remote-id { ascii ascii_string
              | regex reg-exp string }
              match-type { all match { protocol | username |
              source-mac | circuit-id | remote-id } | any match {
              protocol | username | source-mac | circuit-id
              | remote-id } }
        exit
      dhcp-profile  dhcp_profile_name
      event session-activate { aaa { authenticate | authorize } |
                             activate-feature-templates
                             feature_templates_list
                             | apply-all-class | class class_name
                             | deactivate-feature-templates
                             feature_templates_list
      pppoe-profile pppoe_profile_name
      session-type { ipv4 | ipv4v6 | ipv6 }
      exit
```

NOTES:

- **profile subscriber** *subscriber_profile_name*: Specifies the profile subscriber name and enters the Profile Subscriber Configuraton mode.
- **aaa { authenticate** *aaa_profile_for_authentication* **| authorize** *aaa_profile_for_authorization* **}**: Specifies the AAA profile to associate for authentication and authorization.
- **activate-feature-templates** *feature_template_name*: Specifies the list of feature-templates in sequence for activation.
- **apply-all-class**: Applies all classes that are enabled.

- **class** *class_name* : Specifies the subscriber class name.
- **matches**: Enters the matches Configuration sub-mode to specify the match values.
  - **match { protocol { dhcp | ppp } | username { ascii** *ascii_string* **| regex** *reg-exp string* **} | source-mac { ascii** *ascii_string* **| regex** *reg-exp string* **} | { circuit-id { ascii** *ascii_string* **|regex** *reg-exp string* **} | remote-id { ascii** *ascii_string* **| regex** *reg-exp string* **} }**: Specifies the list of match values.
    - **match { protocol { dhcp | ppp }**: Specifies the match protocol as DHCP or PPP.
    - **username { ascii** *ascii_string* **| regex** *reg-exp string* **}**: Specifies the username in ascii format or regular express (reg-exp) string.
    - **source-mac { ascii** *ascii_string* **| regex** *reg-exp string* **}**: Specifies the source MAC address in ascii format or regular express (reg-exp) string.
    - **remote-id { ascii** *ascii_string* **| regex** *reg-exp string* **}**: Specifies the remote identifier in ascii format or regular express (reg-exp) string.
    - **circuit-id { ascii** *ascii_string* **| regex** *reg-exp string* **}**: Specifies the circuit identifier in ascii format or regular express (reg-exp) string.
    - **match-type { all match { protocol | username | source-mac | circuit-id | remote-id } | any match { protocol | username | source-mac | circuit-id | remote-id } }**: Specifies the match key and value for matching any or all of the options: protocol, username, source-mac, circuit-id, and remote-id.

Note By default **aaa, activate-feature-templates, apply-all-class**, and **class** are executed as part of the session bring-up. The PPPoE and DHCP access protocols use these events to create a subscriber in the SM. The operator may configure the AAA actions and activate-feature-templates, suitable for a subscriber.

- **dhcp-profile** *dhcp_profile_name*: Associates the DHCP first sign of life (FSOL) profile.
- **event session-activate**: Specifies the subscriber event to activate.

  Some Access-Protocols require a two-stage session bring up. For example with PPPoE subscribers, the PPPoE Access protocol will call the Session-Start event for FSOL followed by Session-Activate during PPP negotiation and authentication. The IPoE subscribers created by DHCP will not use this event. The operator may configure authenticate, authorize AAA actions and feature templates as suitable for a subscriber.
- **pppoe-profile** *pppoe_profile_name*: Associates the PPPoE FSOL profile.
- **session-type { ipv4 | ipv4v6 | ipv6 }**: Specifies the allowed session-types as IPv4, IPv4v6, and IPv6.

Automatic Session Reconciliation

Feature Description

The Automatic Session Reconciliation feature enables reconciliation of sessions that are out of synchronization between the Control Plane (CP) and User Plane (UP).

Desynchronization of a session occurs when the transaction is successful in the UP but times out before receiving a response from the UP.

The existing transaction-id increments by 1 in every request initiated from the CP to the UP. The CDL record stores the transaction-id per session when the UP conveys a successful response to the CP. The UP also stores this transaction-id when the transaction is successful in the UP.

How it Works

This section briefly describes how the Automatic Session Reconciliation feature works.

The UP validates the transaction-id received in every request from the CP. When a received transaction-id is not incremental to the transaction-id present in UP, the UP discards the transaction and responds to the CP with a transaction-id mismatch error.

On receiving the transaction-id mismatch error, the CP discards the current transaction and initiates a new transaction to replay the complete session data to the UP. After this session replay, the session reconciles and synchronizes automatically in the CP and UP.

Framed Route Support

Feature Description

The Framed Route Support on subscriber sessions enables framed (dynamic) routes to be added for individual subscribers. Framed route per subscriber support is provided through RADIUS or Change of Authorization (CoA).

A framed route is pushed from the Control Plane (CP) to the User Plane (UP) only when the IP address is allocated for the respective address family indicator (AFI). The UP withdraws the framed route when the respective AFI goes down (for example, when an IP address is deallocated).

The configuration format of the framed route is as follows:

- **IPv4**

```
Framed-Route = "[vrf <prefix VRF>] {<prefix>/<prefix_length>} [vrf <next hop vrf>]
                [<next hop prefix>] [<metric>] [tag <tag-value>] "
Framed-Route = "[vrf <vrf-name>] {<prefix> <netmask>} [vrf <next hop vrf>]
                [<next hop prefix>]
                [<metric>] [tag <tag-value>] "


cisco-avpair = "[vrf <prefix VRF>] {<prefix>/<prefix_length>} [vrf <next hop vrf>]
```

```
                [<next hop prefix>] [<metric>] [tag <tag-value>]
cisco-avpair = "[vrf <vrf-name>] {<prefix> <netmask>} [vrf <next hop vrf>]
                [<next hop prefix>]
                [<metric>] [tag <tag-value>] "
```

- **IPv6**

```
Framed-IPv6-Route = "[vrf <prefix VRF>] {<prefix>/<prefix_length>}
                     [vrf <next hop vrf>]
                     [<next-hop prefix>] [<metric>] [tag <tag-value>] "
cisco-avpair = "[vrf <prefix VRF>] {<prefix>/<prefix_length>} [vrf <next hop vrf>]
                [<next hop prefix>][<metric>] [tag <tag-value>] "
```

The description of the format of the framed route is as follows:

- **[vrf <prefix VRF>]**: This is an optional parameter. Specfies the virtual routing and forwarding (VRF) prefix.
- **{<prefix>/<prefix_length>} or {<prefix> <netmask>}**: This is a mandatory parameter. Specifies the prefix and prefix mask or prefix length for the destination.
- **[vrf <next hop vrf>]**: This is an optional parameter. Specifies the next hop VRF name.
- **[<next hop prefix>]**: This is an optional parameter. Specifies that when the next hop is specified as "0.0.0.0" for IPv4 or "::" for IPv6, the IP address of the session must be used as the next hop prefix.
- **[<metric>]**: This is an optional parameter. Specifies the route metric.
- **[tag <tag-value>]**: This is an optional parameter. Specifies a tag value that can be used as a match for controlling redistribution using route policies.

For information about the framed-route attributes, see Table 1 and Table 2 in the RADIUS Attributes chapter.

Implementing the framed (dynamic) route support depends on the UP. Therfore, check the *UP Cloud Native BNG User Plane Configuration Guide for Cisco ASR 9000 Series Router* for the following before enabling the framed route.

- IPv4 and IPv6 framed route support for PPP Termination and Aggregation (PTA) and IPoE
- VRF and next hop VRF support for PTA and IPoE
- CoA support for framed route for PTA and IPoE
- Maximum routes supported per subscriber per AFI for PTA and IPoE

Session disconnect history

A session disconnect history is a troubleshooting feature that

- maintains a cache of disconnected subscriber sessions
- enables operators to clear disconnect history records as needed, and
- provides metrics for monitoring session disconnects across User Planes.

You can customize the number of disconnect history records stored per user plane and clear records globally or by specific identifiers.

Table 3: Feature history

Feature Name	Release Information	Description
Session disconnect history enhancements	2025.04.0	You can now clear session disconnect history and increase the storage limit to 168,000 entries for up to two User Planes. New CLI commands allow clearing records by User Plane and setting custom limits. Disconnect history metrics are also available in Grafana for improved monitoring and visibility.
Session disconnect history	2025.01.0	This feature enhances troubleshooting by providing detailed records of past session disconnections in cnBNGs. This feature is crucial for understanding why sessions have been disconnected in the past, allowing for effective problem resolution and network management.

Benefits of session disconnect history enhancements

The enhancements to session disconnect history provide several benefits:

- You can clear disconnect history records globally or for specific User Planes.
- The default limit of 1,000 records per user plane can be increased up to 168,000, but this higher limit is supported for only two User Planes due to performance considerations. Not all User Planes can be set to this maximum value.
- Metrics are exposed for monitoring disconnects, providing visibility through tools such as Grafana and the Ops-center.
- Backward compatibility is maintained, so previous configurations and data remain accessible.

Restrictions for session disconnect history

These restrictions apply to the session disconnect history feature:

- This feature cannot be enabled or disabled via the CLI.
- Display is limited to per UPF or per SRG-peer-id only.

Configure the disconnect history record limit per User Plane

Set a custom limit for the number of disconnect history records stored for each User Plane.

By default, each user plane stores up to 1,000 disconnect history records. You can increase this limit up to 168,000, but this higher limit is supported only for two User Planes. If you attempt to set the higher limit for a third user plane, Ops Center will reject the configuration.

Before you begin

- Ensure you have access privileges to configure User Plane settings.

Procedure

Enter the user-plane configuration mode, and configure the desired record limit.

Example:

```
user-plane
 instance 1
  user-plane automation-userplane
   disconnect-history-records 161000
   peer-address ipv4 10.1.33.94
   subscriber-profile automation-subsprofile
  exit
 exit
exit
```

The user plane will retain up to the specified number of disconnect history records. If the configuration is removed or reset, the default limit of 1,000 records per user plane applies.

Clear session disconnect history records

Remove disconnect history records from the system for maintenance or troubleshooting.

Before you begin

- Confirm which records (all, or specific User Plane) you wish to clear.

Procedure

Step 1 Use the **clear subscriber session disconnect history** command to clear all disconnect history records.

Step 2 Use the **clear subscriber session disconnect-history upf** *user-plane-name* command to clear records for a specific User Plane.

Monitor disconnect history metrics

Track and visualize disconnect history statistics to gain operational insight.

Monitoring platforms such as Grafana provide metrics that help visualize disconnect events across User Planes.

Before you begin

Ensure your monitoring system is configured to collect and display the relevant metrics.

Procedure

Step 1 Access your monitoring system (for example, Grafana).

Step 2 Locate the **Session_Disconnect_History_Total** metric, and review labels such as disconnect_reason, upf, and srg_peer_id.

NOTES:

- **disconnect_reason**: reason for the subscriber call disconnect
- **upf**: User Plane name
- **srg_peer_id**: SRG peer ID

The **srg_peer_id** label is only included if you enable it through a specific configuration in Ops Center. By default, this configuration is not set.

a) Use this sample configuration to configure srg_peer_id.

Example:

```
bng#
config
 infra metrics verbose application
  level trace
  metrics Session_Disconnect_History_Total
   level production
   granular-labels [ srg_peer_id ]
  exit
 exit
```

Step 3 Use these metrics to identify trends and troubleshoot disconnect issues.

Verify session disconnect history

Use the **show subscriber session disconnect-history** command to view the disconnected session details.

UPF based CLIs

- `bng# show subscriber session disconnect-history upf up1 unique`

```
Tue Dec  17 03:32:08.430 UTC+00:00
subscriber-details
-------------------------------------------------------------------------------------------------------
[Disconnect Reason]           [Last Disconnect Time]   [Mac-Address]  [Sublabel]
[Srg-Peer-Id] [Count]
-------------------------------------------------------------------------------------------------------
UPF: [up1]
-------------------------------------------------------------------------------------------------------
Dhcp admin delete              2024/12/16 14:34:36.080   aa11.0000.0001   16777223
 Peer1        1
PPPoE admin delete             2024/12/16 14:44:27.079   cc11.0000.0001   16777229
 Peer1        1
PPPoE received PADT
from the client                2024/12/16 14:43:59.983   cc11.0000.0001   16777228
 Peer1        1
SessionDisconnect              2024/12/16 14:31:59.338   aa11.0000.0001   16777222
 Peer4        6
admin triggered subscriber
session-synchronize-cp failed  2024/12/16 14:38:44.085   aa11.0000.0001   16777226
 Peer1        1
session timeout                2024/12/16 14:35:45.055   aa11.0000.0001   16777224
 Peer1        1
```

```
CoA Session-Disconnect         2024/12/16 14:42:46.001   aa11.0000.0001   16777227
 Peer1        1
```

This command displays the time of the last disconnected call and the total number of calls for each recorded disconnect reason for the UPF.

- bng# **show subscriber session disconnect-history upf** *up1* **last** *1*

```
Tue Dec  17 03:32:56.705 UTC+00:00
subscriber-details
-------------------------------------------------------------------------------------------
[Disconnect Reason]    [Last Disconnect Time]   [Mac-Address]  [Sublabel] [Srg-Peer-Id]
-------------------------------------------------------------------------------------------
UPF: [up1]
-------------------------------------------------------------------------------------------
PPPoE admin delete     2024/12/16 14:44:27.079  cc11.0000.0001   16777229  Peer1
```

This command displays the most recent disconnected calls for the selected number, covering all disconnect reasons, in reverse chronological order for the UPF.

- bng# **show subscriber session disconnect-history upf up1 filter mac** *aa11.0000.0064*

```
Mon Nov  25 03:49:26.734 UTC+00:00
subscriber-details
{
  "subResponses": [
    {
      "subLabel": "16777514",
      "srgPeerId": "Peer1",
      "srgGroupId": "Group1",
      "srgIntfId": "1",
      "mac": "aa11.0000.0064",
      "acct-sess-id": "0100012a",
      "sesstype": "ipoe",
      "state": "established",
      "subCreateTime": "Mon, 25 Nov 2024 03:40:57 UTC",
      "dhcpAuditId": 2,
      "transId": "1",
      "subsAttr": {
        "attrs": {
<snip>
      "upfsInfo": {
        "up1": {
          "portName": "GigabitEthernet0/0/0/1",
          "upId": 293,
          "transId": 1,
          "smupState": "smUpSessionCreated"
        },
        "up1-stby": {
          "portName": "GigabitEthernet0/0/0/3",
          "upId": 296,
          "transId": 1,
          "smupState": "smUpSessionCreated",
          "lastUpdateTime": "Mon, 25 Nov 2024 03:40:57 UTC"
        }
      },
      "sess-events": [
        "Time, Event, Status",
        "2024-11-25 03:40:57.85041449 +0000 UTC, SessionCreate, success",
        "2024-11-25 03:40:57.875228277 +0000 UTC, N4-Create:up1, PASS",
        "2024-11-25 03:40:57.876039904 +0000 UTC, SessionUpdate, success",
        "2024-11-25 03:40:57.887317627 +0000 UTC, N4CreateToStdby:up1-stby, PASS",
        "2024-11-25 03:41:08.735558746 +0000 UTC, SessionTimerExpiry:up1, PASS"
```

```
      ]
    }
```

This command displays all CDL lines in the disconnect history cache for the given MAC address. It displays the complete session context.

- bng# **show subscriber session disconnect-history upf** *up1* **filter sublabel** *16777514*

```
Mon Nov  25 03:50:02.691 UTC+00:00
subscriber-details
{
  "subResponses": [
    {
      "subLabel": "16777514",
      "srgPeerId": "Peer1",
      "srgGroupId": "Group1",
      "srgIntfId": "1",
      "mac": "aa11.0000.0064",
      "acct-sess-id": "0100012a",
      "sesstype": "ipoe",
      "state": "established",
      "subCreateTime": "Mon, 25 Nov 2024 03:40:57 UTC",
      "dhcpAuditId": 2,
      "transId": "1",
      "subsAttr": {
        "attrs": {
<snip>

      "upfsInfo": {
        "up1": {
          "portName": "GigabitEthernet0/0/0/1",
          "upId": 293,
          "transId": 1,
          "smupState": "smUpSessionCreated"
        },
        "up1-stby": {
          "portName": "GigabitEthernet0/0/0/3",
          "upId": 296,
          "transId": 1,
          "smupState": "smUpSessionCreated",
          "lastUpdateTime": "Mon, 25 Nov 2024 03:40:57 UTC"
        }
      },
      "sess-events": [
        "Time, Event, Status",
        "2024-11-25 03:40:57.85041449 +0000 UTC, SessionCreate, success",
        "2024-11-25 03:40:57.875228277 +0000 UTC, N4-Create:up1, PASS",
        "2024-11-25 03:40:57.876039904 +0000 UTC, SessionUpdate, success",
        "2024-11-25 03:40:57.887317627 +0000 UTC, N4CreateToStdby:up1-stby, PASS",
        "2024-11-25 03:41:08.735558746 +0000 UTC, SessionTimerExpiry:up1, PASS"
      ]
    }
  ]
}
```

This command displays all CDL lines in the disconnect history cache for the given sublabel. It displays the complete session context.

SRG peer-id based CLIs

- bng# **show subscriber session disconnect-history srg-peer-id** *Peer4* **last** *5*

```
Tue Dec  17 03:36:39.161 UTC+00:00
subscriber-details
-------------------------------------------------------------------------------------
[Disconnect Reason] [Last Disconnect Time]   [Mac-Address]  [Sublabel] [UserPlane]
-------------------------------------------------------------------------------------
PeerID: [Peer4]
-------------------------------------------------------------------------------------
SessionDisconnect   2024/12/16 14:31:59.338   aa11.0000.0001   16777222  up1
SessionDisconnect   2024/12/16 14:29:56.763   aa11.0000.0001   16777221  up1
SessionDisconnect   2024/12/16 14:29:08.779   aa11.0000.0004   16777220  up1
SessionDisconnect   2024/12/16 14:29:07.783   aa11.0000.0003   16777219  up1
SessionDisconnect   2024/12/16 14:29:06.784   aa11.0000.0002   16777218  up1
```

This command displays the most recent disconnected calls for the selected number, covering all disconnect reasons, in reverse chronological order for the specific SRG peer-id.

- `bng# show subscriber session disconnect-history srg-peer-id Peer1 unique`

```
Tue Dec  17 03:37:36.656 UTC+00:00
subscriber-details
-----------------------------------------------------------------------------------------------------------
[Disconnect Reason]           [Last Disconnect Time]    [Mac-Address]  [Sublabel]
[UserPlane] [Count]
-----------------------------------------------------------------------------------------------------------
PeerID: [Peer1]
-----------------------------------------------------------------------------------------------------------
Dhcp admin delete              2024/12/16 14:34:36.080   aa11.0000.0001  16777223
 up1       1
PPPoE admin delete             2024/12/16 14:44:27.079   cc11.0000.0001  16777229
 up1       1
PPPoE received PADT from the
 client                        2024/12/16 14:43:59.983   cc11.0000.0001  16777228
 up1       1
admin triggered subscriber
session-synchronize-cp failed  2024/12/16 14:38:44.085   aa11.0000.0001  16777226
 up1       1
session timeout                2024/12/16 14:35:45.055   aa11.0000.0001  16777224
 up1       1
CoA Session-Disconnect         2024/12/16 14:42:46.001   aa11.0000.0001  16777227
 up1       1
```

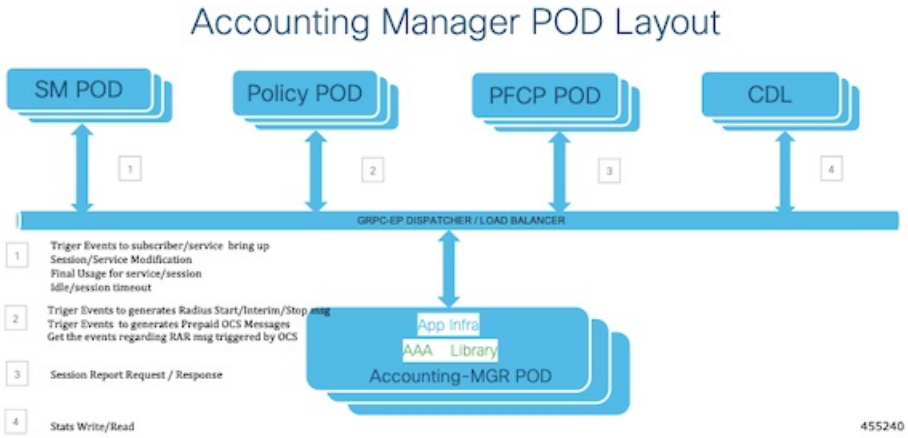
This command displays the time of the last disconnected call and the total number of calls for each recorded disconnect reason for the specific SRG peer-id.

Subscriber Accounting Functions

Feature Description

The Accounting Manager handles the Subscriber Accounting functions in the cnBNG CP. The Accounting function includes features that track traffic either in volume or duration. It provides accounting information for subscribers on a session or per service. The Accounting function determines the length and duration of a given service that a subscriber has used. Certain regulations require service providers to account for services they provide to the subscriber.

The following figure illustrates the Accounting Manager external interfaces.



The Accounting Manager in cnBNG supports the following forms of accounting:

Service Accounting

ISPs can offer different tiered services to their subscribers with the ability to move between different tiers. Different tiers could correspond to different bandwidths offered to the subscriber. A subscriber can enable a new service that corresponds to temporarily moving from one tier of service to another. ISPs need to keep track of when a new service is enabled and how long it is active for each subscriber. Often there might be a need to count the number of packets and bytes associated with a service. Both of these forms of accountng are referred to as service accounting. When service accounting is enabled, BNG sends a Service-Start request when service is activated and a Service-Stop request when the service is deactivated. A timestamp is sent with both the actions. Service-Stop can also contain statistics associated with the service.

To configure Service Accounting, see Configuring Service Accounting, on page 18.

Session Accounting

When Session Accounting is activated, an Accounting-Start request is sent to AAA when the session is started. When the session is terminated, an Accounting-Stop request is sent. The Accounting-Stop request contains the final session accounting statistics (packets, bytes in, bytes out). An “interim” session accounting can be optionally activated that sends Interim-Updates periodically while the session is active. These updates provide the current session statistics accumulated since the start of the session.

Session Accounting is configured directly on the template.

To configure Session Accounting, see Configuring Session Accounting, on page 18.

Limitations and Restrictions

The Subscriber Accounting Function has the following limitation in this release:

- An interim Interval of zero is not supported.
- AAA profile change at service level is not supported.
- Service-level attributes changes are not supported after service bring-up.
- Session accounting is mandatory to enable Service accounting due to User Plane (UP) (asr9k) limitation.
- Session and Service Accounting enable or disable is not supported after session or service is up because of UP limitations. Session Accounting must be enabled only during session bring-up.

Configuring Subscriber Accounting Functions

This section describes how to configure the Subscriber Accounting Functions.

The configuration of the Subscriber Accounting Functions involve the following procedures:

- Configuring Service Accounting
- Configuring Session Accounting

Configuring Service Accounting

Use the following commands to configure service accounting.

```
config
  profile feature-template feature-template
  service accounting
    aaa-profile aaa_profile_name
    enable
    periodic-interval interval_in_seconds
    exit
```

NOTES:

- **profile feature-template** *feature-template*: Specifies the profile feature template name and enters Feature-Template Configuration mode.
- **service accounting**: Enters the Service Configuration mode to configure service accounting for a AAA profile.
- **aaa-profile** *aaa_profile_name*: Specifies the AAA profile to use for service accounting.
- **enable**: Enables service accounting for the specified AAA profile.
- **periodic-interval** *interval_in_seconds*: Specifies the interim interval in seconds. The valid values range from 60 to 4320000 seconds.

Configuring Session Accounting

Use the following commands to configure session accounting.

```
config
  profile feature-template feature-template
  session accounting
    aaa-profile aaa_profile_name
    dual-stack-delay delay_in_seconds
    enable
    periodic-interval interval_in_seconds
    exit
```

NOTES:

- **profile feature-template** *feature-template*: Specifies the profile feature template name and enters Feature-Template Configuration mode.

- **session accounting**: Enters the Session Configuration mode to configure session accounting for a AAA profile.
- **aaa-profile** *aaa_profile_name*: Specifies the AAA profile to use for session accounting.
- **dual-stack-delay** *delay_in_seconds*: Specifies the dual stack set delay time in seconds. The valid values range from 1 to 30 seconds.
- **enable**: Enables session accounting for the specified AAA profile.
- **periodic-interval** *interval_in_seconds*: Specifies the interim interval in seconds. The valid values range from 60 to 4320000 seconds.

Subscriber QoS Policy

Feature Description

The Subscriber Quality of Service (QoS) Policy feature uses the following Cisco AVPs to apply the subscriber QOS policy through RADIUS.

```
cisco-avpair = "subscriber:sub-qos-policy-in=<ingress qos policy name>"
cisco-avpair = "subscriber:sub-qos-policy-out=<egress qos policy name>",
```

Example:

```
radius profile
cisco-avpair = "subscriber:sub-qos-policy-in=qos_in_100mbps",
cisco-avpair = "subscriber:sub-qos-policy-out=qos_out_100mbps",
```

"qos_in_100mbps" and "qos_out_100mbps" are the QoS policy maps that are configured in the User Plane (UP). The merge-level and accounting features are not supported through RADIUS. If unsupported features are passed from RADIUS, behaviour is undefined.

Applying QoS from profile feature-template and through RADIUS using sub-qos-policy-in or sub-qos-policy-out is not supporetd for the same subscriber. When applied, behaviour is undefined.

For information about the sub-qos-policy-in or sub-qos-policy-out attributes, see Table 2 in the RADIUS Attributes chapter.

RADIUS-Based Policing - QoS Shape-Rate parameterization

RADIUS-Based Policing (RaBaPol) is a network management approach that

- enables the use of customized parameters instead of default parameters to activate cnBNG subscriber services.
- allows for greater flexibility and control over service configurations.

Table 4: Feature History

Feature Name	Release Information	Description
RADIUS-Based Policing - QoS Shape-Rate parameterization	2025.02.0	You can now dynamically manage your cnBNG subscriber services through RADIUS-based activation. With RADIUS-Based Policing (RaBaPol), you can customize service parameters, such as the QoS shape-rate, according to your requirements, giving you greater control over service management.

Parameterization of QoS shape-rate

RaBaPol supports the customization of the QoS shape-rate parameter. This parameter can be sent to the cnBNG Control Plane (CP) by the RADIUS server either during the initial connection setup as Cisco VSAs in an Access Accept message, or through Change of Authorization (CoA) messages.

Configuring QoS shape-rate parameterization

To establish QoS shape-rate parameterization, use the **shape average $var_name = value** command in the policy-map class configuration mode in the cnBNG User Plane (UP). This customization is feature-dependent and requires specific syntax and semantics. For QoS, a dollar sign ($) is added as a prefix to the **shape-rate** variable, and the default value, along with the variables, is configured in the policy-map definition.

Handling service changes and errors

If a service associated with a subscriber needs a change in the variable list, deactivate the current service using CoA Session-Disconnect and activate the updated service using CoA Session-Activate process. If an error occurs during feature activation, the cnBNG UP reverts all features and associated variable lists to their previous states.

Policy merging support

You can merge QoS policies from multiple dynamic templates. Configure these templates through CLI or download them from an AAA server for comprehensive policy integration.

Benefits of RADIUS-Based Policing

The RADIUS-Based Policing feature provides these benefits.

- **Dynamic activation**: Enables dynamic and flexible service activation based on RADIUS messages.
- **QoS customization**: Allows for the customization of QoS parameters to meet specific subscriber needs.
- **Policy merging**: Supports the merging of QoS policies from multiple dynamic templates for a subscriber.
- **Error rollback**: Provides rollback capabilities to previous states in case of errors during service activation.

Use case for QoS shape-rate parameterization

This use case illustrates how to manage and customize network QoS settings when a subscriber starts a session.

1. **Subscriber session initiation**: A user starts a session with specific credentials and settings, such as a username, password, and protocol type. For example,

```
user-cpe@abc.com         Password="abc"
            Framed-Protocol=PPP,
            Service-Type=Framed-User
```

```
.....
 Cisco-avpair = "subscriber:sa=DEFAULT-QOS(shape-rate=120000)
```

2. **AAA server communication**: The Authentication, Authorization, and Accounting (AAA) server sends an Access-Accept message to the cnBNG. This message specifies the service name, action type, and a list of variables with their values, like the QoS shape-rate.
3. **Policy configuration**: The service name from the AAA message maps to a feature-template on the cnBNG's control plane, and the specified QoS shape-rate is used to override the default settings on the cnBNG's user plane. The policy merges these custom values with default values, retaining defaults where no specific values are provided.
4. **Service activation via CoA**: Alternatively, service activation can be achieved using CoA, which involves removing the old policy and configuring a new, merged policy in the hardware.

Limitations of configuring RADIUS-Based Policy

This limitation applies to the RADIUS-Based Policy feature:

- Service modifications with different RaBaPol configurations are not supported.

Configure QoS shape-rate parameterization

Follow these steps to configure QoS shape-rate parameterization.

Procedure

Step 1 Define a feature template with the desired QoS configuration on the cnBNG CP.

Example:

```
config
  profile feature-template feature_template_name
    qos
        in-policy qos_input_policy_name
        out-policy qos_output_policy_name
        merge-level integer
     exit
  exit
```

NOTES:

- **profile feature-template** *feature_template_name*: Specifies the profile feature template name.
- **qos**: Enters the QoS configuration mode to configure the parameters.
- **in-policy** *qos_input_policy_name*: Specifies the QoS input policy name. The supported length of the *qos_input_policy_name* ranges from 1 to 128 characters.
- **out-policy** *qos_output_policy_name*: Specifies the QoS output policy name. The supported length of the *qos_output_policy_name* ranges from 1 to 128 characters.

- **merge-level** *integer*: Enables or disables the merge level. A merge value of 0 disables the merge-level. Any value greater than 0, enables the merge level.

This is a sample configuration.

```
config
  profile feature-template DEFAULT-QOS
    qos
     in-policy hqos-policy1
     out-policy hqos-policy2
     merge-level 10
   exit
  exit
```

Step 2 Configure the policy map with a shape-rate value, on the cnBNG UP.

Example:

```
config
  policy-map policy_map_name
      class class-default
          shape average $shape-rate =  rate (units)
      exit
      end-policy-map
  exit
```

NOTES:

- **policy-map** *policy_map_name*: Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
- **class class-default**: Configures a traffic policy for the default class of the traffic policy.
- **shape average $shape-rate =** *rate (units)*: Average shaping rate in the specified units. Values can be from 1 to 4294967295. *Units* can be one of the following:
  - **bps**
  - **gbps**
  - **kbps**
  - **mbps**

This is a sample configuration.

```
config
  policy-map hqos-policy2
      class class-default
          shape average $shape-rate = 100000 kbps
      exit
      end-policy-map
  exit
```

In this example, the service named DEFAULT-QOS has QoS features enabled. The associated feature template is configured with outgoing QoS policies. The default value of shape-rate (the rate at which traffic is shaped) is set to 100000 kbps.

Step 3 Add the user profile to the USER file in RADIUS.

Example:

```
user-cpe@abc.com         Password="abc"
               Framed-Protocol=PPP,
               Service-Type=Framed-User
                        .....
               Cisco-avpair = "subscriber:sa=DEFAULT-QOS(shape-rate=120000)”
```

This specified QoS shape-rate value (for example, 120000) overrides the default value configured on the cnBNG UP.

Step 4 Use the **show subscriber session detail** command to verify the configuration, on the cnBNG CP.

Example:

```
show subscriber session detail
```

```
subscriber-details
{
  "subResponses": [
    {
      "subLabel": "16777218",
      "mac": "cc11.0000.0001",
      "acct-sess-id": "01000002",
      "upf": "asr9k-1",
      "port-id": "Bundle-Ether1",
      "up-subs-id": "1",
      "sesstype": "ppp",
      "state": "established",
      "subCreateTime": "Fri, 15 Nov 2024 03:34:47 UTC",
      "pppAuditId": 3,
      "transId": "2",
      "subcfgInfo": {
      "activatedServices": [
          {
            "serviceName": "DEFAULT-QOS",
            "serviceAttrs": {
              "attrs": {
                "accounting-list": "automation-aaaprofile",
                "acct-interval": "900",
                "service-acct-enabled": "true",
                "service-parameters": "shape-rate=120000",
                "sub-qos-policy-in": "hqos-policy1",
                "sub-qos-policy-out": "hqos-policy2"
              }
} } ] } } ] }
```

Step 5 Use the **show policy-map applied interface** command to view sessions configured with RaBaPol, on the cnBNG UP.

Example:

```
bng# show policy-map applied interface Bundle-Ether1.1.pppoe100

Input policy-map applied to Bundle-Ether1.1.pppoe100:

  policy-map hqos-policy1
   class class-default
    police rate 200 kbps
    !
   !

Output policy-map applied to Bundle-Ether1.1.pppoe100:

  policy-map hqos-policy2
   class class-default
```

```
 shape average $shape-rate = 100000 kbps
!
```

Shared Policy Instance

Shared Policy Instance (SPI) is a mechanism that enables

- allocation of a single set of QoS resources among groups of cnBNG sub-interfaces and bundle sub-interfaces
- sharing of these resources across multiple Ethernet flow points, bundle interfaces, or groups of sub-interfaces.

Table 5: Feature History

Feature Name	Release Information	Description
Shared Policy Instance	2025.02.0	You can now allocate and share a single set of QoS resources across multiple cnBNG sub-interfaces and bundle sub-interfaces. By using a single QoS policy instance across multiple sub-interfaces, you can achieve aggregate shaping across your sub-interfaces, promoting streamlined bandwidth management.

Efficient QoS policy sharing across sub-interfaces: SPI allows you to share a single QoS policy instance among multiple sub-interfaces to maintain a unified rate through aggregate shaping. Sub-interfaces sharing the QoS policy must belong to the same physical interface, with the number ranging from 2 to the maximum supported by the port.

Configuration and application of policies: To implement SPI, you must configure a complete hierarchical policy-map that includes both parent and child policies. The SPI name can be defined and linked to a feature template or downloaded from a RADIUS server.

There are two main ways to configure these policies:

- **CLI and Feature Template**: Policy is configured through a Command Line Interface (CLI) and applied through a feature-template.
- **CLI and AAA Server**: Policy is configured through CLI and applied through an AAA server.

Limitations of configuring Shared Policy Instance

Session consistency within S-VLAN interface

Sessions sharing the same SPI must remain within the same S-VLAN interface.

Service accounting

Service accounting is not supported for services configured with an SPI.

SPI name change requirements

- If you modify the policy-map associated with an SPI, you must also change the SPI name.
- Avoid the following scenarios:
  - Applying a new policy with the same policy-map name but a different SPI name to a subscriber who already has an SPI policy applied. The system will reject this configuration.
  - Applying a new policy with a different policy-map name but the same SPI name. The system will reject this configuration as well.

CoA service-update request limitation

When a service policy with a user profile configuration that includes an SPI is enabled, you cannot simultaneously use an SPI in a CoA service-update request.

Configure a policy with SPI using feature template

Perform this task to configure a policy with shared policy instance in the input and output direction using feature template.

Procedure

Step 1 Define a feature template on the Control Plane (CP) that includes the SPI configuration.

Example:

```
config
  profile feature-template feature_template_name
    qos
        in-policy qos_input_policy_name
          in-shared-policy-instance spi_name
        out-policy qos_output_policy_name
          out-shared-policy-instance spi_name
     exit
  exit
```

NOTES:

- **profile feature-template** *feature_template_name*: Specifies the profile feature template name.
- **qos**: Enters the QoS configuration mode to configure the parameters.
- **in-policy** *qos_input_policy_name*: Specifies the QoS input policy associated with SPI. The supported length of the *qos_input_policy_name* ranges from 1 to 128 characters.
- **in-shared-policy-instance** *input_spi_name*: Specifies the input SPI name for the QoS policy. This command applies a shared traffic policy to inbound traffic across multiple interfaces.
- **out-policy** *qos_output_policy_name*: Specifies the QoS output policy associated with SPI. The supported length of the *qos_output_policy_name* ranges from 1 to 128 characters.

- **out-shared-policy-instance** *output_spi_name*: Specifies the output SPI name for the QoS Policy. This command applies a shared traffic policy to outbound traffic across multiple interfaces.

This is a sample configuration.

```
config
   profile feature-template DEFAULT-QOS
      qos
         in-policy hqos-policy1
           in-shared-policy-instance spi1
         out-policy hqos-policy2
           out-shared-policy-instance spi2
     exit
  exit
```

Step 2 Configure traffic policing on the cnBNG UP to monitor the traffic rate and apply actions (such as dropping or remarking packets) when the traffic exceeds the allowed limit.

Example:

```
config
  policy-map policy_map_name
     class class-default
        police rate value
     exit
     end-policy-map
  exit
```

NOTES:

- **policy-map** *policy_map_name*: Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
- **class class-default**: Configures a traffic policy for the default class of the traffic policy.
- **police rate** *value*: Configures traffic policing. The *value* indicates Committed information rate (CIR). Range is from 1 to 4294967295.

Step 3 Configure traffic shaping for a specific interface on the cnBNG UP.

Example:

```
config
  policy-map policy_map_name
     class class-default
        shape average value
     exit
     end-policy-map
  exit
```

NOTES:

- **shape average** *value*: Specifies the average shaping rate in the specified units. This command limits the average rate of outgoing traffic to a predefined value. Values can be from 1 to 4294967295.

Step 4 Use the **show subscriber session detail** command to verify the configuration.

Example:

```
bng# show subscriber session detail
subscriber-details
{
  "subResponses": [
    {
      "subLabel": "16777220",
      "mac": "0011.9400.0001",
      "acct-sess-id": "01000004",
      "upf": "asr9k-1",
      "port-id": "Bundle-Ether1.1",
      "up-subs-id": "3",
      "sesstype": "ppp",
      "state": "established",
      "subCreateTime": "Fri, 15 Nov 2024 04:18:51 UTC",
      "pppAuditId": 3,
      "transId": "2",
      "subcfgInfo": {
        "committedAttrs": {
        "activatedServices": [
          {
            "serviceName": "DEFAULT-QOS",
            "serviceAttrs": {
              "attrs": {
                "accounting-list": "aaaprofile",
                "acct-interval": "900",
                "service-acct-enabled": "true",
                "sub-qos-policy-in": "hqos-policy1",
                "sub-qos-policy-out": "hqos-policy2",
                "sub-qos-spi-in": "spi1",
                "sub-qos-spi-out": "spi2"
              }
            } } ] } ] }
```

Configure a Policy with SPI using RADIUS

Follow these steps to configure a policy with shared policy instance using RADIUS.

Procedure

Step 1 Configure a policy map that can be shared to one or more interfaces to specify a service policy, on the cnBNG UP.

Example:

```
config
  policy-map policy_map_name1
      class class-default
         police rate value
      exit
      end-policy-map
  exit

policy-map policy_map_name2
      class class-default
         shape average value
      exit
```

```
    end-policy-map
  exit
```

This is a sample configuration.

```
config
  policy-map hqos-policy1
      class class-default
          police rate 1024 kbps
      !
      end-policy-map
  !
  policy-map hqos-policy2
      class class-default
          shape average 4096 kbps
      !
      end-policy-map
  !
```

NOTES:

- **police rate** *value*: Specifies the policing rate for the policy-map. The value represents the committed information rate and ranges from 1 to 4294967295.
- **shape average** *value*: Specifies the average shaping rate in specified units. Values can be from 1 to 4294967295.

Step 2 Add the QoS policy with the SPI name to the USER file in RADIUS.

Example:

```
abc@example.com Cleartext-Password:= "xyz"
cisco-avpair += "sub-qos-policy-in=hqos-policy1 shared-policy-instance spi1",
cisco-avpair += "sub-qos-policy-out=hqos-policy2 shared-policy-instance spi2",
```

Step 3 Use the **show subscriber session detail** command to verify the configuration of a subscriber with a user-profile that includes both QoS and SPI settings, on the cnBNG CP.

Example:

```
bng# show subscriber session detail
subscriber-details
      {
      "subResponses": [
      {
      "subLabel": "16777221",
      "mac": "cc11.0000.0001",
      "acct-sess-id": "01000005",
      "upf": "asr9k-1",
      "port-id": "Bundle-Ether1",
      "up-subs-id": "4",
      "sesstype": "ppp",
      "state": "established",
      "subCreateTime": "Fri, 15 Nov 2024 04:35:15 UTC",
      "pppAuditId": 3,
      "transId": "2",
      "subcfgInfo": {
        "committedAttrs": {
          "attrs": {
            "accounting-list": "aaaprofile",
            "acct-interval": "900",
            "addr-pool": "pool-ISP",
            "ppp-authentication": "pap,chap",
            "ppp-ipcp-reneg-ignore": "true",
```

```
            "ppp-ipv6cp-reneg-ignore": "true",
            "ppp-lcp-delay-seconds": "1",
            "ppp-lcp-reneg-ignore": "true",
            "service-type": "Framed(2)",
            "session-acct-enabled": "true",
            "sub-qos-policy-in": "hqos-policy1 shared-policy-instance spi1",
            "sub-qos-policy-out": "hqos-policy2 shared-policy-instance spi2",
            "vrf": "default"
          }
        } } } ] }
```

Step 4 Use the **show cnbng-nal subscriber all detail** command to display sessions with user-profile having QoS and SPI, on the cnBNG UP.

Example:

```
show cnbng-nal subscriber all detail
Interface:                 Bundle-Ether1.1.pppoe4
UPID:                      0x00000004
CPID:                      0x01000005
Type:                      PPPoE
PPPOE Session Id:          00000006

Attribute List: 0x175d470
1:  ipv4-unnumbered len=  9  value= Loopback0
2:  sub-qos-policy-in len= 59  value= hqos-policy1 shared-policy-instance spi1
3:  sub-qos-policy-out len= 63  value= hqos-policy2 shared-policy-instance spi2
```