Authentication, Authorization, and Accounting Functions

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 to balance transaction load on the RADIUS server based on least outstanding transactions.	2024.03.0
Introduced Asynchronous mode for sending RADIUS Accounting messages.	2024.03.0

Revision Details	Release
Introduced support for different interim intervals for service and session accounting.	2024.02.0
Introduced a conditional approach to Change of Authorization (CoA) retries based on the Error-Cause AVPs.	2024.02.0
Support is added for IPv6 transport between cnBNG and RADIUS AAA server, and also between cnBNG endpoint and CoA client.	2024.01.0
The RADIUS Automated Testing feature is introduced.	2024.01.0
The **user-plane-ip** keyword is added to the **nas-ip** attribute under **radius accounting**, and **radius attribute** configurations.	2023.04.0
Enhancement Introduced: The AAA feature is NSO-integrated.	2021.04.0

• This release introduces support for the Multi-Action Change of Authorization.
• Updated the following sections:
  • Configuring AAA Attributes: The **user-plane** is added as the one of the **format-order** identifiers while formatting AAA attributes.
  The **format-string** keyword was added to the AAframe format of the attribute.
  • Configuring RADIUS Accounting Options: The *nas_port* variable is added to explicitly specify the **nas-port** value.
  The **{ format-e** *format_e* **{ nas-port-type** *nas_port_type* **} }** options are added to the **nas-port** keyword.
  • Configuring RADIUS Attribute Format: The *nas_port* variable is added to explicitly specify the **nas-port** value.
  The **{ format-e** *format_e* **{ nas-port-type** *nas_port_type* **} }** options are added to the **nas-port** keyword.
	2021.03.0

First introduced.	2021.01.0

Feature Description

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

Note: All references to BNG in this chapter refer to the Cloud-Native Broadband Network Gateway (cnBNG).

This chapter provides information about configuring authentication, authorization, and accounting (AAA) functions on the BNG. BNG interacts with the RADIUS server to perform AAA functions. A group of RADIUS servers form a server group that is assigned specific AAA tasks. A method list defined on a server or server group lists methods by which authorization is performed. Some of the RADIUS features include creating specific AAA attribute formats, load balancing of RADIUS servers, throttling of RADIUS records, Change of Authorization (CoA), Session Accounting, and Service Accounting for QoS.

AAA Overview

AAA acts as a framework for effective network management and security. It helps in managing network resources, enforcing policies, auditing network usage, and providing bill-related information. BNG connects to an external RADIUS server that provides the AAA functions.

The RADIUS server performs the three independent security functions (authentication, authorization, and accounting) to secure networks against unauthorized access. The RADIUS server runs the Remote Authentication Dial-In User Service (RADIUS) protocol. (For details about RADIUS protocol, refer to RFC 2865). The RADIUS server manages the AAA process by interacting with BNG, and databases and directories containing user information.

The RADIUS protocol runs on a distributed client-server system. The RADIUS client runs on BNG (Cisco ASR 9000 Series Router) that sends authentication requests to a central RADIUS server. The RADIUS server contains all user authentication and network service access information.

The AAA processes, the role of RADIUS server during these processes, and some BNG restrictions, are explained in these sections:

Authentication

The authentication process identifies a subscriber on the network, before granting access to the network and network services. The process of authentication works on a unique set of criteria that each subscriber has for gaining access to the network. Typically, the RADIUS server performs authentication by matching the credentials (user name and password) the subscriber enters with those present in the database for that subscriber. If the credentials match, the subscriber is granted access to the network. Otherwise, the authentication process fails, and network access is denied.

Authorization

After the authentication process, the subscriber is authorized for performing certain activity. Authorization is the process that determines what type of activities, resources, or services a subscriber is permitted to use. For example, after logging into the network, the subscriber may try to access a database, or a restricted website. The authorization process determines whether the subscriber has the authority to access these network resources.

AAA authorization works by assembling a set of attributes based on the authentication credentials provided by the subscriber. The RADIUS server compares these attributes, for a given username, with information

contained in a database. The result is returned to BNG to determine the actual capabilities and restrictions that are to be applied for that subscriber.

Accounting

The accounting keeps track of resources used by the subscriber during network access. Accounting is used for billing, trend analysis, tracking resource utilization, and capacity planning activities. During the accounting process, a log is maintained for network usage statistics. The information monitored include, but are not limited to - subscriber identities, applied configurations on the subscriber, the start and stop times of network connections, and the number of packets and bytes transferred to, and from, the network.

BNG reports subscriber activity to the RADIUS server in the form of accounting records. Each accounting record comprises of an accounting attribute value. This value is analyzed and used by the RADIUS server for network management, client billing, auditing, etc.

The accounting records of the subscriber sessions may timeout if the BNG does not receive acknowledgments from the RADIUS server. This timeout can be due to RADIUS server being unreachable or due to network connectivity issues leading to slow performance of the RADIUS server. It is therefore recommended that a RADIUS server **deadtime** be configured on the BNG, to avoid loss of sessions. Once this value is configured, and if a particular session is not receiving an accounting response even after retries, then that particular RADIUS server is considered to be non-working and further requests are not sent to that server.

Restrictions

- On session disconnect, transmission of the Accounting-Stop request to RADIUS may be delayed for a few seconds while the system waits for the "final" session statistics to be collected from the hardware. The Event-Timestamp attribute in that Accounting-Stop request should, however, reflect the time the client disconnects, and not the transmission time.

Using RADIUS Server Group

A RADIUS server group is a named group of one or more RADIUS servers. Each server group is used for a particular service. For example, in an AAA network configuration having two RADIUS server groups, the first server group can be assigned the authentication and authorization task, while the second group can be assigned the accounting task.

Server groups can include multiple host entries for the same server. Each entry, however, must have a unique identifier. This unique identifier is created by combining an IP address and a UDP port number. Different ports of the server, therefore, can be separately defined as individual RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to different UDP ports on the same server. Further, if two different host entries on the same RADIUS server are configured for the same service (like the authentication process), then the second host entry acts as a fail-over backup for the first one. That is, if the first host entry fails to provide authentication services, BNG tries with the second host entry. (The RADIUS host entries are tried in the order in which they are created.)

For assigning specific actions to the server group, see Configuring RADIUS Server Group, on page 34.

Specifying Method Order

Method order for AAA defines the methods using which authorization is performed, and the sequence in which these methods are executed. Before any defined authentication method is performed, the method order must be applied to the configuration mechanism responsible for validating user-access credentials.

On BNG, you have to specify the method order and the server group that will be used for AAA services. For specifying method order, see Configuring Method Order for AAA, on page 23.

Defining AAA Attributes

The AAA attribute is an element of RADIUS packet. A RADIUS packet transfers data between a RADIUS server and a RADIUS client. The AAA attribute parameter, and its value - form a Attribute Value Pair (AVP). The AVP carries data for both requests and responses for the AAA transaction.

The AAA attributes either can be predefined as in Internet Engineering Task Force (IETF) attributes or vendor defined as in vendor-specific attributes (VSAs). For more information about the list of BNG supported attributes, see RADIUS Attributes.

The RADIUS server provides configuration updates to BNG in the form of attributes in RADIUS messages. The configuration updates can be applied on a subscriber during session setup through two typical methods—per-user attributes, which applies configuration on a subscriber as part of the subscriber's authentication Access Accept, or through explicit domain, port, or service authorization Access Accepts. This is all controlled by the Policy Rule Engine's configuration on the subscriber.

When BNG sends an authentication or an authorization request to an external RADIUS server as an Access Request, the server sends back configuration updates to BNG as part of the Access Accept. In addition to RADIUS configuring a subscriber during setup, the server can send a change of authorization (CoA) message autonomously to the BNG during the subscriber's active session life cycle, even when the BNG did not send a request. These RADIUS CoA updates act as dynamic updates, referencing configured elements in the BNG and instructing the BNG to update a particular control policy or service policy.

BNG supports the concept of a "service", which is a group of configured features acting together to represent that service. Services can be represented as either features configured on dynamic-templates through CLI, or as features configured as RADIUS attributes inside Radius Servers. Services are activated either directly from CLI or RADIUS through configured "activate" actions on the Policy Rule Engine, or through CoA "activate-service" requests. Services can also be deactivated directly (removing all the involved features within the named service) through configured "deactivate" action on the Policy Rule Engine or through CoA "deactivate-service" requests.

The attribute values received from RADIUS interact with the subscriber session in this way:

- BNG merges the values received in the RADIUS update with the existing values that were provisioned statically by means of CLI commands, or from prior RADIUS updates.
- In all cases, values received in a RADIUS update take precedence over any corresponding CLI provisioned values or prior RADIUS updates. Even if you reconfigured the CLI provisioned values, the system does not override session attributes or features that were received in a RADIUS update.
- Changes made to CLI provision values on the dynamic template take effect immediately on all sessions using that template, assuming the template features have not already been overridden by RADIUS. Same applies to service updates made through CoA "service-update" requests.

AAA Attribute List

An attribute list is named list that contains a set of attributes. You can configure the RADIUS server to use a particular attribute list to perform the AAA function.

To create an attribute list, see Configuring RADIUS Attributes, on page 28.

AAA Attribute Format

It is possible to define a customized format for some attributes. For the configuration syntax for creating a new format, see Configuring AAA Attributes, on page 20.

Once the format is defined, the FORMAT-NAME can be applied to various AAA attributes such as username, nas-port-ID, calling-station-ID, and called-station-ID. The configurable AAA attributes that use the format capability are explained in the section Creating Attributes of Specific Format, on page 6.

To create a customized nas-port attribute and apply a predefined format to nas-port-ID attribute , see Configuring RADIUS Attribute Format, on page 29.

Specific functions can be defined for an attribute format for specific purposes. For example, if the input username is "text@abc.com", and only the portion after "@" is required as the username, a function can be defined to retain only the portion after "@" as the username. Then, "text" is dropped from the input, and the new username is "abc.com". To apply username truncation function to a named-attribute format, see Configuring AAA Attributes, on page 20.

Creating Attributes of Specific Format

BNG supports the use of configurable AAA attributes. The configurable AAA attributes have specific user-defined formats. The following sections list some of the configurable AAA attributes used by BNG.

Username

BNG has the ability to construct AAA username and other format-supported attributes for subscribers using MAC address, circuit-ID, remote-ID, and DHCP Option-60 (and a larger set of values available in CLI). The DHCP option-60 is one of the newer options that is communicated by the DHCP client to the DHCP server in its requests; it carries Vendor Class Identifier (VCI) of the DHCP client's hardware.

The MAC address attribute is specified in the CLI format in either of these forms:

- mac-address: for example, 0000.4096.3e4a
- mac-address-ietf: for example, 00-00-40-96-3E-4A
- mac-address-raw: for example, 000040963e4a
- mac-address-custom1: for example, 01.23.45.67.89.AB

  (This particular MAC address format is available only from Release 6.2.1 and later).

NAS-Port-ID

The NAS-Port-ID is constructed by combining BNG port information and access-node information. The BNG port information consists of a string in this form:

```
"eth phy_slot/phy_subslot/phy_port:XPI.XCI"
```

For 802.1Q tunneling (QinQ), XPI is the outer VLAN tag and XCI is the inner VLAN tag.

If the interface is QinQ, the default format of nas-port-ID includes both the VLAN tags; if the interface is single tag, it includes a single VLAN tag.

In the case of a single VLAN, only the outer VLAN is configured, using this syntax:

```
<slot>/<subslot>/<port>/<outer_vlan>
```

In the case of QinQ, the VLAN is configured using this syntax:

```
<slot>/<subslot>/<port>/<inner_vlan>.<outer_vlan>
```

In the case of a bundle-interface, the phy_slot and the phy_subslot are set to zero (0); whereas the phy_port number is the bundle number. For example, 0/0/10/30 is the NAS-Port-ID for a Bundle-Ether10.41 with an outer VLAN value 30.

The nas-port-ID command is extended to use the 'nas-port-type' option so that the customized format (configured with the command shown above) can be used on a specific interface type (nas-port-type).

If 'type' option is not specified, then the nas-port-ID for all interface types is constructed according to the format name specified in the command.

Calling-Station-ID and Called-Station-ID

BNG supports the use of configurable calling-station-ID and called-station-ID. The calling-station-ID is a RADIUS attribute that uses Automatic Number Identification (ANI), or similar technology. It allows the network access server (NAS) to send to the Access-Request packet, the phone number from which the call came from. The called-station-ID is a RADIUS attribute that uses Dialed Number Identification (DNIS), or similar technology. It allows the NAS to send to the Access-Request packet, the phone number that the user called from.

NAS-Port Format

NAS-Port is a 4-byte value that has the physical port information of the Broadband Remote Access Server (BRAS), which connects the Access Aggregation network to BNG. It is used both by Access-Request packets and Accounting-Request packets. To uniquely identify a physical port on BRAS, multiple pieces of information such as shelf, slot, adapter, and so on is used along with the port number. A configurable format called format-e is defined to allow individual bits or group of bits in 32 bits of NAS-Port to represent or encode various pieces that constitute port information.

Individual bits in NAS-Port can be encoded with these characters:

- Zero: 0
- One: 1
- PPPoX slot: S
- PPPoX adapter: A
- PPPoX port: P
- PPPoX VLAN Id: V
- PPPoX VPI: I
- PPPoX VCI: C
- Session-Id: U
- PPPoX Inner VLAN ID: Q

The permissible nas-port type values are:

Nas-port-types	Values	Whether value can be derived from associated interface
VIRTUAL_PPPOEOVLAN	36	Yes

Nas-port-types	Values	Whether value can be derived from associated interface
VIRTUAL_PPPOEOQINQ	37	Yes
VIRTUAL_IPOEOVLAN	43	Yes
VIRTUAL_IPOEOQINQ	44	Yes

Note If a NAS-Port format is not configured for a NAS-Port-Type, the system looks for a default CLI configuration for the NAS-Port format. In the absence of both these configurations, for sessions with that particular NAS-Port-Type, the NAS-Port attribute is not sent to the RADIUS server.

Making RADIUS Server Settings

In order to make BNG interact with the RADIUS server, certain server specific settings must be made on the BNG router.

For more making RADIUS server settings, see Configuring RADIUS Server, on page 33.

Restriction

The service profile push or asynchronously pushing a profile to the system is not supported. To download a profile from Radius, the profile must be requested initially as part of the subscriber request. Only service-update is supported and can be used to change a service that was previously downloaded.

Balancing Transaction Load on the RADIUS Server

Table 3: Feature History

Feature Name	Release Information	Description
Balancing Transaction Load on the RADIUS Server	2024.03.0	This feature enhances performance by distributing AAA messages across servers, ensuring faster response times. It selects the RADIUS server with the fewest outstanding transactions, rather than using the previous First server or Round Robin methods, which did not account for server load. This results in a more efficient handling of authentication, authorization, and accounting tasks.

The Balancing Transaction Load on the RADIUS Server feature provides a mechanism to share the load of RADIUS access and accounting transactions, across a set of RADIUS servers. Each AAA request processing is considered to be a transaction. cnBNG distributes batches of transactions to servers within a server group.

When the first transaction for a new batch is received, cnBNG determines the server with the lowest number of outstanding transactions in its queue. This server is assigned that batch of transactions. cnBNG keeps repeating this determination process to ensure that the server with the least-outstanding transactions always gets a new batch. This method is known as the least-outstanding method of load balancing.

The size of each batch is configurable. Changes in the batch size may impact the CPU load and network throughput. There is no standard size for batches, but generally, more than 50 transactions is considered large, and fewer than 25 is considered small.

Retransmitted messages go to the same server. The number of outstanding RADIUS messages is tracked per server at the pod level. Load balancing applies to the named RADIUS server groups or a global server group specified in the subscriber profile. In AAA method lists, this group must be referred to as "radius." All servers in the RADIUS group are subject to load balancing.

RADIUS Server Status and Automated Testing

The Balancing Transaction Load on the RADIUS Server feature checks the status of servers before sending transaction batches. Only live servers receive transaction batches.

Transactions are not sent to servers marked as dead. A server remains marked as dead until its timer expires. The server is included again only when the RADIUS automated tester verifies it as alive. Otherwise, the server is excluded from the selection algorithm.

The RADIUS automated tester periodically sends a request to the server. If the server returns an **Access-Reject** message, it is alive. If not, it remains marked as dead until detected as alive.

If a server is unresponsive, it is marked as dead, and transactions fail over to the next available server.

When using the RADIUS automated tester, verify that the authentication, authorization, and accounting (AAA) servers respond to test packets sent by the network access server (NAS). Incorrect configurations may cause packet drops and servers to be erroneously marked as dead.

For configuring the load balancing on the RADIUS server, see Configuring RADIUS Server Selection Logic, on page 34.

RADIUS Change of Authorization Overview

The RADIUS Change of Authorization (CoA) function allows the RADIUS server to change the authorization settings for a subscriber who is already authorized. CoA is an extension to the RADIUS standard that allows sending asynchronous messages from RADIUS servers to a RADIUS client, like BNG.

Note A CoA server can be a different from the RADIUS server.

To identify the subscriber whose configuration needs to be changed, a RADIUS CoA server supports and uses a variety of keys (RADIUS attributes) such as Accounting-Session-ID, Username, IP-Address, and ipv4:vrf-id.

The RADIUS CoA supports:

- account-update — BNG parses and applies the attributes received as part of the CoA profile. Only subscriber-specific attributes are supported and applied on the user profile.
- activate-service — BNG starts a predefined service on a subscriber. The service settings can either be defined locally by a dynamic template, or downloaded from the RADIUS server.
- deactivate-service — BNG stops a previously started service on the subscriber, which is equivalent to deactivating a dynamic-template.

For a list of supported Vendor-Specific Attributes for account operations, see Vendor-Specific Attributes for Account Operations.

Note In order for BNG to enable interim accounting, it is mandatory for the CoA request to have both accounting method list from the dynamic-template and Acct-Interim-Interval attribute from the user profile. This behavior is applicable for accounting enabled through dynamic-template. Whereas, from Cisco IOS XR Software Release 5.3.0 and later, the CoA request needs to have only the Acct-Interim-Interval attribute in the user profile.

Service Activate from CoA

BNG supports activating services through CoA requests. The CoA **service-activate** command is used for activating services. The CoA request for the service activate should contain these attributes:

- "subscriber:command=activate-service" Cisco VSA
- "subscriber:service-name=<service name>" Cisco VSA
- Other attributes that are part of the service profile

The "<subscriber:sa=<service-name>" can also be used to activate services from CoA and through RADIUS.

Duplicate service activate requests can be sent to BNG from the CoA server. BNG does not take any action on services that are already activated. BNG sends a CoA ACK message to the CoA server under these scenarios:

- When a duplicate request with identical parameters comes from the CoA for a service that is already active.
- When a duplicate request with identical parameters comes from the CoA to apply a parameterized service.

BNG sends a CoA NACK message to the CoA server with an error code as an invalid attribute under these scenarios:

- When a request comes from the CoA to deactivate a non-parameterized service that is not applied to the session.
- When a request comes from the CoA to deactivate a parameterized service that is not applied to the session.
- When a duplicate request to apply a parameterized service is made with non-identical parameters from the CoA.
- When a request with non-identical parameters comes from CoA to deactivate a parameterized service.

Service Update from CoA

The service update feature allows an existing service-profile to be updated with a new RADIUS attribute list representing the updated service. This impacts any subscriber who is already activated with the service and new subscriber who activate the service in the future. The new CoA **service-update** command is used for activating this feature. The CoA request for the service update should have these attributes:

- "subscriber:command=service-update" Cisco VSA
- "subscriber:service-name=<service name>" Cisco VSA
- Other attributes that are part of the service profile

A service update CoA should have a minimum of these attributes:

- vsa cisco generic 1 string "subscriber:command=service-update"
- vsa cisco generic 1 string "subscriber:service-name=<service name>"

Web Logon with RADIUS Based CoA

To support Web Logon, a set of Policy Rule Events need to be configured in an ordered manner. These events are as follows:

- session-start:
  - On the start of a session, a subscriber is setup to get internet connectivity. The service is activated to redirect HTTP traffic to a Web portal for web-based logon.
  - Start the timer with duration for the maximum waiting period for authentication.
- account-logon—The Web portal collects the user credentials such as username and password and triggers a CoA account-logon command. When this event is triggered, subscriber username and password are authenticated by the RADIUS server. Once the authentication is successful, the HTTP redirect service is deactivated, granting user access to already connected internet setup. Also, the timer established in session-start must be stopped. However, if the authentication fails during account-logon, BNG sends a NAK CoA request, allowing for further authentication attempts to take place.
- timer expiry—When the timer expires, the subscriber session is disconnected based on the configuration.

Multi-Action Change of Authorization

BNG supports multi-action Change of Authorization (CoA) wherein service providers can activate and deactivate multiple services using a single CoA request. Multi-action CoA is supported for **Service-Activate** and **Service-Deactivate** commands.

During the multi-action CoA request, if any of the COA requests fail to activate or deactivate, then any of the services which have been activated or deactivated as part of that CoA request is rolled back to its previous state. The session restores back to the its pre-MA-CoA state upon failure to activation or deactivation.

An Example of a Multi-Action Change of Authorization Use Case

The following example lists the sequence of events that occur in the case of a PTA session initiation.

1. PTA session's web traffic redirected to a service portal (HTTP Redirect)
2. The user activates the first level of service through the service portal. A multi-action COA request is initiated in the following sequence.
   a. Deactivate redirection
   b. Activate Turbo Button 1
   c. Activate VoIP with two channels
3. The user activates the second level of service through the service portal. A multi-action COA request is initiated in the following sequence.
   a. Deactivate Turbo Button 1

b. Activate Turbo Button 2

c. Deactivate VoIP with two channels

d. Activate VoIP with 4 channels

Interworking with Service-Level Accounting

BNG supports Service-Level Accounting, where a service is a collection of features that are activated and deactivated as a group. Service-Level Accounting and MA-CoA features are independent, that is, they can be applied separately. However, MA-CoA accounts for services that are activated or deactivated that have Service-Level Accounting enabled through the dynamic template configuration.

Generating Accounting Records

The following cases describes how the multi-action CoA records are generated for accounting purposes.

MA-CoA ACK Case

- If MA-CoA request contains only service activate commands, then START accounting record for those services are generated after the CoA Ack is sent out.
- If MA-CoA request contains only deactivate services or combination of activate and deactivate services, then for those services START or STOP accounting records are generated after the CoA Ack is sent out.

MA-CoA NAK Case (Rollback scenario)

- If MA-CoA request fails due to presence of invalid command formats or due to internal software failure or due to presence of invalid service names, that are not defined in the box, in such cases the accounting START or STOP messages are not generated upon rollback.
- If MA-CoA request fails due to internal feature programming failure, then the Service-START or Service-STOP accounting records may be generated for the services that were activated or deactivated before the failure. After the failure, the rollback is initiated and appropriate Service-START or Service-STOP records are generated for these services.

Sample MA-COA Request

```
exec /bin/echo
"Cisco-AVPair='subscriber:sd=svcQoSAcct1',Cisco-AVPair='subscriber:sd=svcQoSAcct2',
Cisco-AVPair='subscriber:sd=svcQoSAcct3',Cisco-AVPair='subscriber:sa=qosin_coa',
Cisco-AVPair='subscriber:sa=qosout_coa',
Acct-Session-Id=00000001" | /usr/local/bin/radclient  -r 1 -x 5.11.17.31:1700 coa coa
```

Enhanced CoA with Conditional Retry Logic

Table 4: Feature History

Feature Name	Release Information	Description
Enhanced CoA with Conditional Retry Logic	2024.02.0	We have introduced a conditional approach to Change of Authorization (CoA) retries based on the Error-Cause AVPs carried in CoA response messages. The CoA client uses the error cause reason and determines whether to initiate a CoA retry. This enhancement can reduce unnecessary traffic and processing overhead, resulting in more efficient network operations and better allocation of resources.

If a CoA request is erroneous, the cnBNG rejects it with a CoA Negative Acknowledgment (NAK) response. The CoA client treats all NAK responses equally, and attempts to send CoA requests automatically. This logic of sending unconditional retries for all error types strains network resources, leading to potential performance degradation for both the policy plane and the cn-BNG.

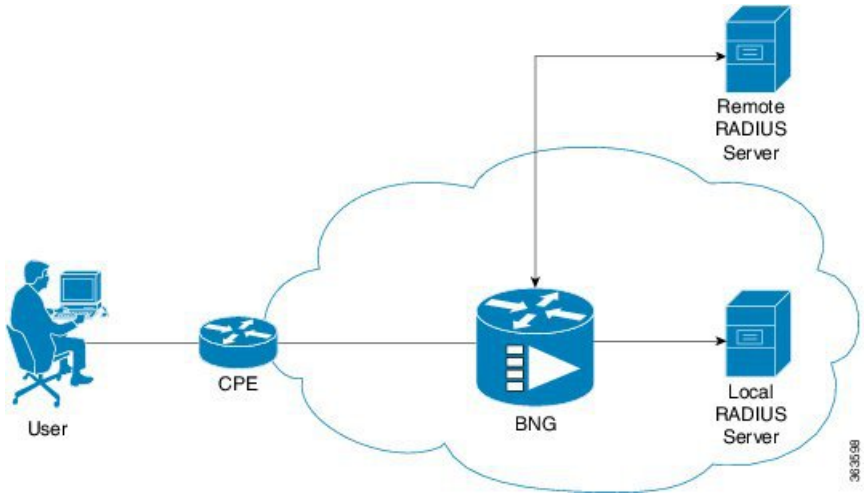
The Enhanced CoA with Conditional Retry Logic feature enables the CoA client to use the error cause carried in a CoA NAK response message and determine whether to send the CoA request again. For example, if the error cause is “503 Session Context Not found,” indicating that the subscriber session is nonexistent or already terminated, the CoA client would recognize that retrying the request would be futile as the cnBNG does not recognize the session context.

By adopting a conditional approach to CoA retries based on the Error-Cause AVPs, the system can avoid unnecessary traffic and processing overhead, resulting in a more efficient network performance.

User Authentication and Authorization in the Local Network

The user authentication and authorization in the local network feature in BNG provides the option to perform subscriber authorization locally (in a subscriber's network), instead of both remote authentication and authorization that occurs in RADIUS servers. With the User Authentication and Authorization in the Local Network feature, you can run the RADIUS server locally in your network, manage, and configure the RADIUS server locally in your network to the profile that is required for the environment. In the case of a remote RADIUS server, the RADIUS server is maintained by an external regulatory body (not within the subscriber's network) and subscriber will not be able to manage or configure the server.

Figure 1: User Authentication and Authorization in the Local Network



User Authentication and Authorization in the Local Network feature is used in a case when a user wants to perform a two-level authentication or authorization, first, a remote authentication (or authorization) followed by a local authorization (or authentication).

Note All the debug commands applicable to AAA server are applicable on User Authentication and Authorization in the Local Network feature.

Service Accounting

Accounting records for each service enabled on a subscriber can be sent to the configured RADIUS server. These records can include service-start, service-stop, and service-interim records containing the current state of the service and any associated counters. This feature is the Service Accounting feature. Service accounting records are consolidated accounting records that represent the collection of features that make up a service as part of a subscriber session.

Service accounting starts when a subscriber session comes up with a service enabled on it. This can happen through a dynamic template applied through a control policy, through access-accept (AA) messages when the session is authorized, or through a change of authorization (CoA), when a new service is applied on a subscriber session. Service accounting stops either when the session is terminated, or a service is removed from the session through CoA, or some other event that deactivates the service. Start records have no counters; interim and stop records with QoS counters are generated when service accounting is enabled for QoS. Interim accounting records can be generated, in between start and stop accounting, as an option with a pre-defined periodic interval. When the interim period is zero, interim accounting records are not created. Different interim intervals are based on every service for each session. Service accounting is enabled on each template, based on the configuration.

From Release 2024.02.0 onwards, different interim intervals for service and session accounting is supported. We recommend that you set the session interim accounting interval as a multiple of the service interim interval. For example, if the service accounting interim is set to 5 minutes, the session interim accounting interval must be set to a multiple of 5 minutes (such as 5, 10, 15 minutes, and so on).

For more information on service accounting for QoS, refer to Authentication, Authorization, and Accounting Functions, on page 1. For more information on commands to configure service accounting, refer to the Configuring Service Accounting.

Note The policy-map associated to a dynamic template can be edited to change the service parameters. However, this does not update the accounting records. Therefore, to generate all the accounting records accurately, it is recommended that a new service with all the required service parameters be created and associated to the new service, through a CoA.

For service accounting, statistics for ingress and egress QoS policies, which are applied under each service for a given subscriber, may need to be reported as part of the accounting interim and stop records. For each service, these QoS counters can be reported as part of the accounting records:

- BytesIn — Aggregate of bytes matching all classes of the ingress QoS policy for the service minus the policer drops.
- PacketsIn — Aggregate of packets matching all classes of the ingress QoS policy for the service minus the policer drops.
- BytesOut — Aggregate of bytes matching all classes of the egress QoS policy for the service minus the queuing drops.
- PacketsOut — Aggregate of packets matching all classes of the egress QoS policy for the service minus the queuing drops

Dynamic template features that support accounting statistic collection and require that their statistics be reported in the AAA service accounting records can enable accounting statistics on their features using the newly-introduced optional **acct-stats** configuration option. This option is not available for the features that do not support statistic collection. By default, QoS accounting statistics are disabled to optimize performance.

Note The QoS counters for each direction is reported only if a QoS policy is applied for that service in the given direction. For example, if a service does not have an ingress policy applied, BytesIn and PacketsIn counters are reported as being 0.

Pre-requisites

- Subscriber accounting, the parent accounting record for service accounting, must be configured to enable the service accounting feature to work.
- The keyword **acct-stats** must be configured in service-policy configuration to enable the service accounting feature to report feature counter information as part of the records.

Restriction

- Service accounting is supported on bundle subscriber interfaces but not on line card subscriber interfaces.
- IPv4 and IPv6 subscriber sessions has a single set of service accounting records. They are merged into one set of bytes_in, bytes_out, packets_in, packets_out counters.
- Service accounting is not supported for static sessions.

RADIUS Accounting Message Handling

Table 5: Feature History

Feature Name	Release Information	Description
RADIUS Accounting Message Handling	2024.03.0	We have enhanced the system performance by preventing packet identifier (PID) exhaustion with the new Asynchronous mode for sending RADIUS Accounting messages. The cnBNG now releases PIDs after dispatching messages, rather than waiting for a long time to receive server responses. This change allows for PID reuse, mitigating scale issues and improving KPIs.

The cnBNG currently sends RADIUS Accounting messages to servers in synchronous mode. It reserves a packet identifier (PID) for each message. The PID is only released when a response arrives from the server. If the server fails to respond promptly, the cnBNG attempts to resend the message. The PID remains reserved until all timeouts and retransmissions are complete. During periods of high traffic, this can lead to PID exhaustion. As a result, cnBNG may drop new incoming RADIUS messages due to the lack of available PIDs, leading to a decrease in Key Performance Indicators (KPIs).

Asynchronous RADIUS Accounting Messages

To improve performance, cnBNG can switch to asynchronous mode for sending RADIUS Accounting messages. In this mode, cnBNG does not wait for a server response. After sending a message, the cnBNG waits for a timeout of one second, and then releases the PID. This allows the PID to be reused for subsequent messages. However, a brief delay is implemented before reusing a PID. This change can prevent PID exhaustion during high traffic and enhance KPIs. Note that the asynchronous mode applies only to RADIUS Accounting messages.

Configure Asynchronous RADIUS Accounting

Configure Asynchronous mode for Session Accounting

Use this configuration to enable asynchronous mode for session accounting.

```
config
  profile aaa aaa_name
    accounting
      session { acct-interim-async true/false | acct-start-async true/false |
acct-stop-async true/false }
       commit
```

Configure Asynchronous mode for Service Accounting

Use this configuration to enable asynchronous mode for service accounting.

```
config
  profile aaa aaa_name
    accounting
      service { acct-interim-async true/false | acct-start-async true/false |
acct-stop-async true/false }
       commit
```

Note If asynchronous RADIUS accounting is configured, message retransmission functionality will be unavailable. Lost messages due to network issues or server errors will not be retried by the Control Plane.

NOTES:

- **profile aaa** *aaa_name*: Specifies the AAA profile name and enters the AAA configuration mode.
- **accounting**: Enters the accounting sub-mode.
- **session [service] acct-interim-async** *true/false* : Specifies the option to enable or disable the asynchronous mode when interim updates are sent.
- **session [service] acct-start-async** *true/false*: Specifies the option to enable or disable the asynchronous mode when an Accounting-Start request is sent to AAA.
- **session [service] acct-stop-async** *true/false*: Specifies the option to enable or disable the asynchronous mode when an Accounting-Stop request is sent.

Standard Compliance

The AAA features are aligned with the following standards:

- RFC 2865 - Remote Authentication Dial In User Service (RADIUS)
- RFC 2866 - RADIUS Accounting
- RFC 5176 - Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS)

RADIUS Automated Testing

For subscriber authentication and accounting, the cnBNG-CP chooses a RADIUS server from the list of configured servers, and sends RADIUS messages. If the RADIUS server is unreachable, it is considered dead, and is excluded from the selection algorithm until a 'dead-timer' expires. After this timer expires, the dead server is re-included in the selection list without checking if it is now reachable. This causes the cnBNB-CP to retransmit messages to the still-unreachable server, causing retransmissions and delays, which negatively impact Key Performance Indicators (KPIs).

The RADIUS Automated Testing feature allows the cnBNG-CP to periodically check the status of the RADIUS server until the server is considered dead or the dead-timer expires. With this feature, if the dead-timer expires, the cnBNG-CP attempts to send authentication and accounting TEST messages to the RADIUS server that is currently unreachable. If the server does not respond to these messages, it is marked as dead, and this process continues until the server is reachable. If the RADIUS server responds within the set number of retransmissions and timeouts, it is marked as available, and is then included in the selection algorithm list that is used to choose RADIUS servers.

There are two configuration scenarios:

- The following happens when RADIUS Automated Testing is enabled without the Idle-timer functionality:

  When the dead-detection criteria is met, the dead-timer starts with the value configured for **deadtime** CLI. Whenever the dead-timer expires, TEST messages are sent to the RADIUS server to check if the

server is reachable. If the RADIUS server responds, the dead-timer stops, and the RADIUS server is marked UP. If the RADIUS server doesn't respond, the dead-timer is restarted.

- The following happens when the **idle-timer** CLI along with **auto-test enable** is configured:

  When the RADIUS server is UP, the status of the server is checked periodically by sending TEST RADIUS messages to the selected RADIUS server as per the configured idle-timer value. If the RADIUS server doesn't respond to the TEST messages, then the RADIUS server is marked as dead. The dead-timer starts as per the value configured in the **deadtime** CLI, and the periodic TEST message is stopped. Once the dead-timer expires, TEST RADIUS messages are sent to the RADIUS server, and the dead-timer is restarted. If the server responds to the TEST messages, the server is marked as UP, the dead-timer is stopped, and periodic TEST messages are restarted. If the server does not respond to TEST messages, on every dead-timer expiry, TEST RADIUS messages are sent to check reachability.

Restrictions

- RADIUS server availability based on VRF is not supported.
- **Show peers** command is not instance-aware.
- Round-trip time (RTT) is based on the server level, not the instance level.
- Server status is maintained per instance. All other statuses are maintained at a global level.
- Presently, only two VIPs are supported.

Software Upgrades

When you upgrade your current software (release version prior to cnBNG 2024.01) to cnBNG 2024.01 and later releases, perform the following steps:

- Fetch the keys of the RADIUS server from ETCD using the following command:

  ```
  kubectl exec -it -n bng <etcd pod name> -- etcdctl get --prefix "" --keys-only | grep serv
  ```

- Delete the key from ETCD using the following command:

  ```
  kubectl exec -it -n bng <etcd pod name> -- etcdctl del key “serverkey”
  ```

Note Rolling upgrade is not supported for this software upgrade scenario.

Configure RADIUS Automated Testing

This section describes how to enable RADIUS Automated Testing feature for accounting and authorization.

```
config
  profile radius
    server ipv4_address port_number
      type { acct | auth }
      auto-test enable
      auto-test enable idle-timer number
      commit
```

To disable the RADIUS Automated Testing feature, use the **no auto-test enable** command, and to disable the idle-timer functionality, use the **no auto-test idle-timer** command.

Note When the RADIUS Automated Testing feature is disabled, the status of the RADIUS server is reset to UP (reachable).

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **server** *ipv4_address port_number*: Specifies the IPv4 address and port of the RADIUS server.
- **type { acct | auth }**: Specifies the type of the RADIUS server. It can be one of the following:
  - **acct**: RADIUS server used for the accounting requests
  - **auth**: RADIUS server used for the authentication requests
- **auto-test enable**: Enables RADIUS automated testing.
- **auto-test enable idle-timer** *minutes*: Enables the idle-timer functionality. *minutes* value ranges from 1 through 30.
- **commit**: Commits the configuration.

Verifying RADIUS Automated Testing

Use the following **show** commands to verify the RADIUS Automated Testing feature.

```
bng# show radius auth-server 10.1.45.112:1812
Mon Nov  6  11:01:25.394 UTC+00:00
-----------------------------------------------------------
 Server: 10.1.45.112, port: 1812, status: instance 1: up , port-type: Auth
 75 requests, 0 pending, 0 retransmits
 0 accepts, 75 rejects, 0 timeouts
 0 bad responses, 0 bad authenticators
 0 unknown types, 0 dropped, 1004 ms latest rtt
 Auto Test Stats:
 instance 1:
  Requests:74 Pending:0 Retransmits:0
  Rejects:74 Timeouts:0 Responses:0
  BadAuth:0 Dropped:0 BadResp:0
-----------------------------------------------------------

bng# show radius acct-server 10.1.45.112:1813
Mon Nov  6  11:22:58.330 UTC+00:00
-----------------------------------------------------------
 Server: 10.1.45.112, port: 1813, status: instance 1: up , port-type: Acct
 338 requests, 0 pending, 0 retransmits
 338 responses, 0 timeouts
 0 bad responses, 0 bad authenticators
 0 unknown types, 0 dropped, 3 ms latest rtt
 Auto Test Stats:
 instance 1:
  Requests:337 Pending:0 Retransmits:0
  Rejects:0 Timeouts:0 Responses:337
  BadAuth:0 Dropped:0 BadResp:0
```

Configuring AAA Functions

This section describes how to configure the following Authentication, Authorization, and Accounting (AAA) functions on the Control Plane (CP).

The configuration of the AAA functions involves the following procedures:

- Configuring AAA Attributes
- Configuring the CoA-NAS Interface
- Configuring Method Order for AAA
- Configuring RADIUS Accounting Options
- Configuring RADIUS Accounting Server Group
- Configuring RADIUS Attributes
- Configuring RADIUS-Dead Time
- Configuring RADIUS-Detect Dead Server
- Configuring RADIUS Pod
- Configuring RADIUS Maximum Retry
- Configuring RADIUS NAS-IP
- Configuring RADIUS Server
- Configuring RADIUS Server Group
- Configuring RADIUS Server Selection Logic
- Configuring RADIUS Timeout

Configuring AAA Attributes

Use the following commands to configure a function for the AAA attribute format.

```
config
  profile attribute-format attribute_format_name
    format-order { addr | circuit-id-tag | client-mac-address |
      addr | circuit-id-tag | client-mac-address |
      client-mac-address-custom1 | client-mac-address-custom2 |
      client-mac-address-ietf | client-mac-address-raw |
      dhcp-client-id | dhcp-client-id-spl | dhcp-user-class |
      dhcp-vendor-class | dhcpv4-client-id-spl |
      dhcpv4-vendor-class | dhcpv6-client-id-ent-ident |
      dhcpv6-interface-id | dhcpv6-vendor-class-string |
      inner-vlan-id | outer-vlan-id | physical-adapter |
      physical-chassis | physical-port | physical-slot |
      physical-subslot | port-type | pppoe-session-id |
      remote-id-tag | service-name | user-plane | username }
```

```
format-string format_string
commit
```

NOTES:

- **profile attribute-format** *attribute_format_name*: Specifies the AAA attributes and enters the Attribute Format Configuration mode.
- **authorization**: Enters the Authorization sub-mode.
- **format-order** *attribute_format* **| identifier { addr | circuit-id-tag | client-mac-address | client-mac-address-custom1 | client-mac-address-custom2 | client-mac-address-ietf | client-mac-address-raw | dhcp-client-id | dhcp-client-id-spl | dhcp-user-class | dhcp-vendor-class | dhcpv4-client-id-spl | dhcpv4-vendor-class | dhcpv6-client-id-ent-ident | dhcpv6-interface-id | dhcpv6-vendor-class-string | inner-vlan-id | outer-vlan-id | physical-adapter | physical-chassis | physical-port | physical-slot | physical-subslot | port-type | pppoe-session-id | remote-id-tag | service-name | username } | value** *value* **}**: Specifies the AAA attribute format order as follows:
  - **addr**: Specifies the IPv4 address of the subscriber.
  - **circuit-id-tag**: Specifies the circuit identifier tag.
  - **client-mac-address**: Specifies the client MAC address in AABB.CCDD.EEFF format.
  - **client-mac-address-custom1**: Specifies the first custom client MAC address in AABB.CCDD.EEFF format.
  - **client-mac-address-custom2**: Specifies the second custom client MAC address in AABB.CCDD.EEFF format.
  - **client-mac-address-ietf**: Specifies the client MAC address in Internet Engineering Task Force (IETF) format. That is, AA-BB-CC-DD-EE-FF format.
  - **client-mac-address-raw**: Specifies the client MAC address in raw (AABBCCDDEEFF) format.
  - **dhcp-client-id**: Specifies the DHCP client identifier.
  - **dhcp-client-id-spl**: Specifies the DHCP client identifier special string.
  - **dhcp-user-class**: Specifies the DHCP user class.
  - **dhcp-vendor-class**: Specifies the DHCP vendor class.
  - **dhcpv4-client-id-spl**: Specifies the DHCPv4 client identifier special string.
  - **dhcpv4-vendor-class**: Specifies the DHCPv4 vendor class.
  - **dhcpv6-client-id-ent-ident**: Specifies the DHCPv6 client and enterprise identifiers.
  - **dhcpv6-interface-id**: Specifies the DHCPv6 interface identifier.
  - **dhcpv6-vendor-class-string**: Specifies the DHCPv6 vendor class string.
  - **inner-vlan-id**: Specifies the inner VLAN identifier.
  - **outer-vlan-id**: Specifies the outer VLAN identifier.
  - **physical-adapter**: Specifies the physical adapter.
  - **physical-chassis**: Specifies the physical chassis.

- **physical-port**: Specifies the physical port.
- **physical-slot**: Specifies the physical slot.
- **physical-subslot**: Specifies the physical subslot.
- **port-type**: Specifies the interface or port type.
- **pppoe-session-id**: Specifies the PPPoE physical identifier.
- **remote-id-tag**: Specifies the remote identifier tag.
- **service-name**: Specifies the service name.
- **user-plane**: Specifies the User Plane (UP).
- **username**: Specifies the username.

- **format-string** *format_string*: Specifies the AAA format pattern. The *format_string* specifies the format string. Each identifier is represented by '%s' tuple. Any other character set is treated as a delimiter. For each '%s' in the format-string, the format-order identifier is used.

Note

- Validation on the number of '%s' in format-string and number of entries in format-order are not performed.
- For backward compatibility, the format-order still takes the delimiter configuration. In this scenario, the format-order takes precedence and the format-string is silently ignored.
- Use the delimiters either in the format-order (as in Release 2021.01) or in format-string (as in Release 2021.03).

Configuring the CoA-NAS Interface

Use the following configuration to define Change of Authorization (CoA) NAS interface in the RADIUS endpoint.

```
config
   endpoint radius
      interface coa-nas
         vip-ip ipv4_address vip-port port_number
         vip-ipv6 ipv6_address vip-ipv6-port port_number
         end
```

NOTES:

- **endpoint radius**: Enters the RADIUS endpoint configuration mode.
- **interface coa-nas**: This keyword defines a new interface "coa-nas", and allows to enter the CoA NAS interface configuration mode.
- **vip-ip** *ipv4_address* **vip-port** *port_number*: Configures the IPv4 address of the host. *ipv4_address* must be in standard IPv4 dotted decimal notation.

You can configure a list of VIP-IPs to listen to the inbound CoA or DM requests.

vip-port *port_number*: Specify the port number of the UDP proxy. By default, the port number is 3799. This default value is used only when the VIP-IP is specified.

Important This configuration allows only port to be specified per IP.

The BNG (udp-pxy) listens to the inbound CoA or DM request messages on these ports and ACK or NAK messages sent with the respective source IP and port.

- **vip-ipv6** *ipv6_address* **vip-ipv6-port** *port_number*: Configures the IPv6 address of the host.

  vip-ipv6-port*port_number*: Specify the port number of the UDP proxy.

Configuring Method Order for AAA

Use the following commands to assign the method order for the server group to use for subscriber authentication, authorization, and accounting.

Authentication

```
config
  profile aaa aaa_name
    authentication
      method-order custom_server_group
      commit
```

NOTES:

- **profile aaa** *aaa_name*: Specifies the AAA profile name and enters the AAA Configuration mode.
- **authentication**: Enters the Authentication sub-mode.
- **method-order** *custom_server_group*: Specifies the method-order to be applied by default for subscriber authentication.

  custom_server_group specifies the name of the server group where the method-order is applied.

Authorization

```
config
  profile aaa aaa_name
    authorization
      password password
      type subscriber method-order custom_server_group
     username { format attribute_format | identifier { addr | circuit-id-tag
 | client-mac-address | client-mac-address-custom1 |
client-mac-address-custom2 | client-mac-address-ietf |
client-mac-address-raw | dhcp-client-id | dhcp-client-id-spl |
dhcp-user-class | dhcp-vendor-class | dhcpv4-client-id-spl |
dhcpv4-vendor-class | dhcpv6-client-id-ent-ident | dhcpv6-interface-id |
 dhcpv6-vendor-class-string | inner-vlan-id | outer-vlan-id |
```

```
physical-adapter | physical-chassis | physical-port | physical-slot |
physical-subslot | port-type | pppoe-session-id | remote-id-tag |
service-name | username } | value value }
       commit
```

NOTES:

- **profile aaa** *aaa_name*: Specifies the AAA profile name and enters the AAA Configuration mode.
- **authorization**: Enters the Authorization sub-mode.
- **password** *password* : Specifies the password for subscriber authentication.
- **type subscriber method-order** *custom_server_group*: Specifies the method-order to be applied by default for subscriber authorization.

  custom_server_group specifies the name of the server group where the method-order is applied.
- **username { format** *attribute_format* **| identifier { addr | circuit-id-tag | client-mac-address | client-mac-address-custom1 | client-mac-address-custom2 | client-mac-address-ietf | client-mac-address-raw | dhcp-client-id | dhcp-client-id-spl | dhcp-user-class | dhcp-vendor-class | dhcpv4-client-id-spl | dhcpv4-vendor-class | dhcpv6-client-id-ent-ident | dhcpv6-interface-id | dhcpv6-vendor-class-string | inner-vlan-id | outer-vlan-id | physical-adapter | physical-chassis | physical-port | physical-slot | physical-subslot | port-type | pppoe-session-id | remote-id-tag | service-name | username } | value** *value* **}**: Specifies the username format, identifier, or value.
  - **format** *attribute_format*: Specifies the username attribute format.
  - **identifier { addr | circuit-id-tag | client-mac-address | client-mac-address-custom1 | client-mac-address-custom2 | client-mac-address-ietf | client-mac-address-raw | dhcp-client-id | dhcp-client-id-spl | dhcp-user-class | dhcp-vendor-class | dhcpv4-client-id-spl | dhcpv4-vendor-class | dhcpv6-client-id-ent-ident | dhcpv6-interface-id | dhcpv6-vendor-class-string | inner-vlan-id | outer-vlan-id | physical-adapter | physical-chassis | physical-port | physical-slot | physical-subslot | port-type | pppoe-session-id | remote-id-tag | service-name | username }**: Specifies the username identifiers as follows:
    - **addr**: Specifies the IPv4 address of the subscriber.
    - **circuit-id-tag**: Specifies the circuit identifier tag.
    - **client-mac-address**: Specifies the client MAC address in AABB.CCDD.EEFF format.
    - **client-mac-address-custom1**: Specifies the first custom client MAC address in AABB.CCDD.EEFF format.
    - **client-mac-address-custom2**: Specifies the second custom client MAC address in AABB.CCDD.EEFF format.
    - **client-mac-address-ietf**: Specifies the client MAC address in Internet Engineering Task Force (IETF) format. That is, AA-BB-CC-DD-EE-FF format.
    - **client-mac-address-raw**: Specifies the client MAC address in raw (AABBCCDDEEFF) format.
    - **dhcp-client-id**: Specifies the DHCP client identifier.
    - **dhcp-client-id-spl**: Specifies the DHCP client identifier special string.
    - **dhcp-user-class**: Specifies the DHCP user class.

- **dhcp-vendor-class**: Specifies the DHCP vendor class.
- **dhcpv4-client-id-spl**: Specifies the DHCPv4 client identifier special string.
- **dhcpv4-vendor-class**: Specifies the DHCPv4 vendor class.
- **dhcpv6-client-id-ent-ident**: Specifies the DHCPv6 client and enterprise identifiers.
- **dhcpv6-interface-id**: Specifies the DHCPv6 interface identifier.
- **dhcpv6-vendor-class-string**: Specifies the DHCPv6 vendor class string.
- **inner-vlan-id**: Specifies the inner VLAN identifier.
- **outer-vlan-id**: Specifies the outer VLAN identifier.
- **physical-adapter**: Specifies the physical adapter.
- **physical-chassis**: Specifies the physical chassis.
- **physical-port**: Specifies the physical port.
- **physical-slot**: Specifies the physical slot.
- **physical-subslot**: Specifies the physical subslot.
- **port-type**: Specifies the interface or port type.
- **pppoe-session-id**: Specifies the PPPoE physical identifier.
- **remote-id-tag**: Specifies the remote identifier tag.
- **service-name**: Specifies the service name.
- **username**: Specifies the username.

Accounting

```
config
  profile aaa aaa_name
     accounting
        method-order custom_server_group
        commit
```

NOTES:

- **profile aaa** *aaa_name*: Specifies the AAA profile name and enters the AAA Configuration mode.
- **accounting**: Enters the Accounting sub-mode.
- **method-order** *custom_server_group*: Specifies the method-order to be applied by default for subscriber accounting.

  custom_server_group specifies the name of the server group where the method-order is applied.

Configuring RADIUS Accounting Options

This section describes how to configure the RADIUS accounting options.

```
config
  profile radius accounting
     algorithm { first-server | round-robin }
     attribute { nas-identifier value | nas-ip { ipv4_address | user-plane-ip
 } |
     nas-ipv6 ipv6_address |
     nas-port { nas_port } | { format-e format_e
               { nas-port-type nas_port_type } }
     deadtime value
     detect-dead-server response-timeout value
     max-retry value
     timeout value
     commit
```

NOTES:

- **profile radius accounting**: Enters the RADIUS accounting configuration mode.
- **algorithm { first-server | round-robin }**: Defines the algorithm for selecting the RADIUS server.
  - **first-server**: Sets the selection logic as highest priority first. This is the default behavior.
  - **round-robin**: Sets the selection logic as round-robin order of servers.
- **attribute { nas-identifier** *value* **| nas-ip {** *ipv4_address* **| user-plane-ip } | nas-port { format-e** *format_e_value* **| nas-port-type** *nas_port_type* **} }**: Configures the RADIUS identification parameters.
  - **nas-identifier** *value*: Specifies the attribute name by which the system will be identified in Accounting-Request messages. *value* must be an alphanumeric string.
  - **nas-ip** *ipv4_address*: Specifies the NAS IPv4 address. *ipv4_address* must be an IPv4 address in dotted decimal notation.
  - **nas-ip user-plane-ip**: Enables the `user-plane-ip-address` AVPair to use the configured User-Plane IP address in Accounting-Request messages.
  - **nas-ipv6** *ipv6_address*: Specifies the NAS IPv6 address.
  - **nas-port {** *nas_port* **} | { format-e** *format_e* **{ nas-port-type** *nas_port_type* **} }**: Specifies the nas-port attributes.
    - *nas_port* configures the NAS port value. The NAS port value ranges from 1 to 4294967295.

Note If none of the NAS port configurations are present, the existing default nas-port logic is applied. That is, setting a fixed-number per radius-pod.

    - **format-e** *format_e_value* : Specifies the custom attribute formation support for nas-port. The nas-port is a 32 bit integer format. The configuration takes a 32 length of characters, each presenting a particular attribute mapping. The *format_e_value* pattern is: 01FSAPRiLUVQ]*):
    - 0 – Set bit to 0

1 – Set bit to 1

F - PHY_SHELF

S – PHY_SLOT

A – PHY_ADAPTER

P - PHY_PORT

R - PHY_CHASSIS

i - PHY_SUBSLOT

L - PHY_CHANNEL

V - OUTER_VLAN_ID

Q - INNER_VLAN_ID

U - PPPOE_SESSION_ID

nas-port-type *nas_port_type*: Specifies the NAS port type. The supported values range from 0 to 44.

Note

- The nas-port-type configuration is not in scope of the Control Plane. It is derived from the interface-type.
- The supported NAS port types are 36 , 37, 43, and 44.
- The NAS port type value takes precedence over the common NAS port format-e.

- **deadtime** *value*: Sets the time to elapse between RADIUS server marked unreachable and when we can re-attempt to connect.

  value must be an integer from 0 through 65535. Default: 10 minutes.

- **detect-dead-server response-timeout** *value*: Sets the timeout value that marks a server as "dead" when a packet is not received for the specified number of seconds.

  value must be an integer from 1 through 65535. Default: 10 seconds.

- **max-retry** *value*: Sets the maximum number of times that the system will attempt retry with the RADIUS server.

  value must be an integer from 0 through 65535. Default: 2

- **timeout** *value*: Sets the time to wait for response from the RADIUS server before retransmitting.

  value must be an integer from 1 through 65535. Default: 2 seconds.

- **commit**: Commits the configuration.
- All the keyword options under the RADIUS accounting configuration mode are also available within the RADIUS configuration mode.

Configuring RADIUS Accounting Server Group

This section describes how to configure the RADIUS server group.

```
configure
   profile radius
      server-group group_name
      commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **server group** *group_name*: Specifies the name of server group for use in RADIUS accounting. *group_name* must be an alphanumeric string.
- **commit**: Commits the configuration.

Configuring RADIUS Attributes

This section describes how to configure the RADIUS attributes for authentication and accounting.

```
config
   profile radius
     attribute { nas-identifier value | nas-ip { ipv4_address | user-plane-ip
 } | nas-ipv6 ipv6_address }
      commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **attribute { nas-identifier** *value* **| nas-ip** *ipv4_address* **}** : Configures the RADIUS identification parameters.
  - **nas-identifier** *value*: Specifies the attribute name by which the system will be identified in Accounting-Request messages. *value* must be an alphanumeric string.
  - **nas-ip** *ipv4_address*: Specifies the NAS IPv4 address. *ipv4_address* must be an IPv4 address in dotted decimal notation.
  - **nas-ip user-plane-ip**: Enables the `user-plane-ip-address` AVPair to use the configured User-Plane IP address in Access-Request or Accounting-Request messages.
  - **nas-ipv6** *ipv6_address*: Specifies the NAS IPv6 address.
- **commit**: Commits the configuration.

Sample Configuration

The following is a sample configuration.

```
config
   profile radius
      attribute
      nas-identifier Ciscobng
      nas-ip        10.1.32.83
```

```
    nas-ipv6 2001::250:56ff:fe95:658
    nas-ip user-plane-ip
    exit
exit
```

Configuring RADIUS Attribute Format

Use the following commands to configure the RADIUS identification parameters.

```
configure
profile radius attribute
     called-station-id { format-name format_name
               | nas-port-type nas_port_type }
     calling-station-id { format-name format_name
               | nas-port-type nas_port_type }
     nas-identifier nas_identifier
     nas-identifier-format nas_identifier_format
     nas-ip { ipv4_address| user-plane-ip }
     nas-ipv6 ipv6_address |
     nas-port { nas_port } | { format-e format_e
               { nas-port-type nas_port_type } }
     nas-port-id nas_port_id { format-name format_name|
               | nas-port-type nas_port_type }
     commit
```

NOTES:

- **profile radius attribute**: Enters the Profile RADIUS Attribute Configuration mode.
- **called-station-id { format-name** *format_name* **| nas-port-type** *nas_port_type* **}** : Specifies the AAA called-station-id attribute.

  format-name *format_name*: Specifies the called-station-id format name.

  nas-port-type *nas_port_type*: Specifies the NAS port type. The supported values range from 0 to 44.

  - nas-port-type configuration is not in scope of the Control Plane. It is derived dependng on the interface-type.
  - The supported NAS port types are 36 , 37, 43, and 44.

- **calling-station-id { format-name** *format_name* **| nas-port-type** *nas_port_type* **}** : Specifies the AAA calling-station-id attribute.
- **nas-identifier { format-name** *format_name* **| nas-port-type** *nas_port_type* **}**: Specifies the attribute name with which the system is identified in the Access-Request messages. The identifier string ranges from 1 to 32 characters.
- **nas-identifier-format { format-name** *format_name* **| nas-port-type** *nas_port_type* **}**: Specifies the AAA nas-identifier-format attribute.
- **nas-ip** *ipv4_address*: Specifies the AAA NAS IPv4 address.
- **nas-ip user-plane-ip**: Enables the `user-plane-ip-address` AVPair to use the configured User-Plane IP address in Access-Request or Accounting-Request messages.
- **nas-ipv6** *ipv6_address*: Specifies the NAS IPv6 address.

- **nas-port {** *nas_port* **} | { format-e** *format_e* **{ nas-port-type** *nas_port_type* **} }**: Specifies the nas-port attributes.
  - *nas_port* configures the NAS port value. The NAS port value ranges from 1 to 4294967295.

Note If none of the NAS port configurations are present, the existing default nas-port logic is applied. That is, setting a fixed-number per radius-pod.

- **format-e** *format_e_value* : Specifies the custom attribute formation support for nas-port. The nas-port is a 32 bit integer format. The configuration takes a 32 length of characters, each presenting a particular attribute mapping. The *format_e_value* pattern is: 01FSAPRiLUVQ]*):
- 0 – Set bit to 0

  1 – Set bit to 1

  F - PHY_SHELF

  S – PHY_SLOT

  A – PHY_ADAPTER

  P - PHY_PORT

  R - PHY_CHASSIS

  i - PHY_SUBSLOT

  L - PHY_CHANNEL

  V - OUTER_VLAN_ID

  Q - INNER_VLAN_ID

  U - PPPOE_SESSION_ID

  nas-port-type *nas_port_type*: Specifies the NAS port type. The supported values range from 0 to 44.

Note
- The nas-port-type configuration is not in scope of the Control Plane. It is derived from the interface-type.
- The supported NAS port types are 36 , 37, 43, and 44.
- The NAS port type value takes precedence over the common NAS port format-e.

- **nas-port-id** *nas_port_id*: Specifies the AAA NAS port-id attribute.

Configuring RADIUS Dead Time

This section describes how to configure the RADIUS dead time.

```
config
  profile radius
     deadtime value
     commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **deadtime** *value*: Sets the time to elapse between RADIUS server marked unreachable and when an reattempt to connect can be made.

  value must be an integer from 0 through 65535. Default: 10 minutes.
- **commit**: Commits the configuration.

Sample Configuration

The following is a sample configuration.

```
config
   profile radius
     deadtime 15
     exit
```

Configuring RADIUS Detect Dead Server

This section describes how to configure the RADIUS detect dead server.

```
config
  profile radius
     detect-dead-server response-timeout value
     commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **detect-dead-server response-timeout** *value*: Sets the timeout value that marks a server as "dead" when a packet is not received for the specified number of seconds.

  value must be an integer from 1 through 65535. Default: 10 seconds.
- **commit**: Commits the configuration.

Sample Configuration

The following is a sample configuration.

```
config
   profile radius
      detect-dead-server response-timeout 100
      exit
```

Configuring RADIUS NAS-IP

This section describes how to configure the RADIUS NAS-IP.

Global RADIUS NAS-IP Configuration

Use the following configuration to configure the NAS-IP address.

```
config
   endpoint radius-dns
      interface radius-client
         vip-ip ipv4_address
         commit
```

NOTES:

- **endpoint radius-dns**: Enters the endpoint radius-ep configuration mode.
- **interface radius-client**: Enters the radius-client interface-type configuration mode.
- **vip-ip** *ipv4_address*: Sets the NAS-IP value, which is also used as the source-IP in UDP requests towards the RADIUS server.
- **commit**: Commits the configuration.

Configuration Example:

```
config
   endpoint radius-dns
      interface radius-client
      vip-ip 209.165.200.228
      exit
   exit
exit
```

Configuring RADIUS Pod

This section describes how to configure the RADIUS pod.

```
config
   endpoint radius
      replicas number_of_replicas
      commit
```

NOTES:

- **endpoint radius**: Enters the RADIUS endpoint configuration mode.
- **replicas** *number_of_replicas*: Sets the number of replicas required.
- **commit**: Commits the configuration.

Sample Configuration

The following is a sample configuration.

```
config
   endpoint radius
      replicas 3
      exit
```

Configuring RADIUS Retries

This section describes how to configure the maximum RADIUS retries.

```
config
  profile radius
    max-retry value
    commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **max-retry** *value*: Sets the maximum number of times that the system will attempt retry with the RADIUS server.

  value must be an integer from 0 through 65535. Default: 2
- **commit**: Commits the configuration.

Sample Configuration

The following is a sample configuration.

```
config
   profile radius
      max-retry 2
      exit
```

Configuring RADIUS Server

This section describes how to configure the RADIUS server settings.

```
config
  profile radius
    server ipv4/ipv6_address port_number
      secret secret_key
      priority priority_value
      type { acct | auth }
      commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **server** *ipv4/ipv6_address port_number*: Specifies the IPv4/IPv6 address and port of the RADIUS server.
- **secret** *secret_key*: Specifies the secret key.
- **priority** *priority_value*: Specifies the server priority.
- **type { acct | auth }**: Specifies the type of the RADIUS server. It can be one of the following:
  - acct: RADIUS server used for the accounting requests
  - auth: RADIUS server used for the authentication requests
- **commit**: Commits the configuration.

Configuring RADIUS Server Group

Use the following commands to configure the RADIUS server group.

```
config
  profile server-group server_group_name
    radius-group radius_server_group_name
    commit
```

NOTES:

- **profile server-group** *server_group_name*: Specifies the profile server group name to enter the Profile Server Group Configuration mode.
- **radius-group** *radius_server_group_name*: Specifies the RADIUS group server name.

Sample Configuration

The following is a sample configuration:

```
server-group automation-server-group
server auth 10.1.36.121 1812
exit
server acct 10.1.36.121 1813
exit
server auth 2001::10:1:36:121 1812
exit
server acct 2001::10:1:36:121 1813
exit
```

Configuring RADIUS Server Selection Logic

This section describes how to configure the RADIUS server selection logic.

```
config
  profile radius
    algorithm { first-server | round-robin | least-outstanding [
batch-size number ] }
    commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **algorithm { first-server | round-robin | least-outstanding [batch-size] }**: Defines the algorithm for selecting the RADIUS server.
  - **first-server**: Sets the selection logic as highest priority first. This is the default behavior.
  - **round-robin**: Sets the selection logic as round-robin order of servers.
  - **least-outstanding** : Sets the selection logic based on the server with the lowest number of outstanding transactions in its queue.
    - **batch-size** *number*: (Optional) Specifies the size of the batch.

      If you do not configure the **batch-size** value in the CLI command, the system takes the default batch size.

- **commit**: Commits the configuration.

Sample Configuration

The following is a sample configuration.

```
config
   profile radius
      algorithm least-outstanding batch-size 30
      exit
```

Configuring RADIUS Timeout

This section describes how to configure the RADIUS timeout.

```
config
   profile radius
      timeout value
      commit
```

NOTES:

- **profile radius**: Enters the RADIUS configuration mode.
- **timeout** *value_in_seconds*: Sets the time to wait for response from the RADIUS server before retransmitting.

  value must be an integer from 1 through 65535. Default: 2 seconds.
- **commit**: Commits the configuration.

Sample Configuration

The following is a sample configuration.

```
config
   profile radius
      timeout 4
      exit
```