RBFS RADIUS Services Guide

Version 21.3.1, 14 April 2021

Registered Address	Support	Sales
26, Kingston Terrace, Princeton, New Jersey 08540, United States		
		+91 80 4850 5445
http://www.rtbrick.com	support@rtbrick.com	sales@rtbrick.com

Table of Contents

1. Overview

The modular, scalable subscriber management that RtBrick calls the next generation access infrastructure (ng-access) provides support for protocols such as PPPoE, IPoE, L2TP and RADIUS.

The subscriber management infrastructure provides the next generation of internet access protocols designed for carrier grade services in regards to scalability and robustness. One of the challenges for carrier networks is interwork with numerous client devices which requires a well implemented, industry proven access protocol stack, including support for all relevant RFCs.

This implementation is designed to be a set of distributed services for increased scaling and stability. The subscriber management implementation has three components:

- subscriberd: subscriber management and AAA (local, RADIUS, and other support)
- pppoed: PPPoE/PPP session handling
- l2tpd: L2TP tunnel and session handling

The subscriber daemon (subscriberd) is the central application, keeping the current subscriber state as well as being responsible for Authentication, Authorization and Accounting (AAA).

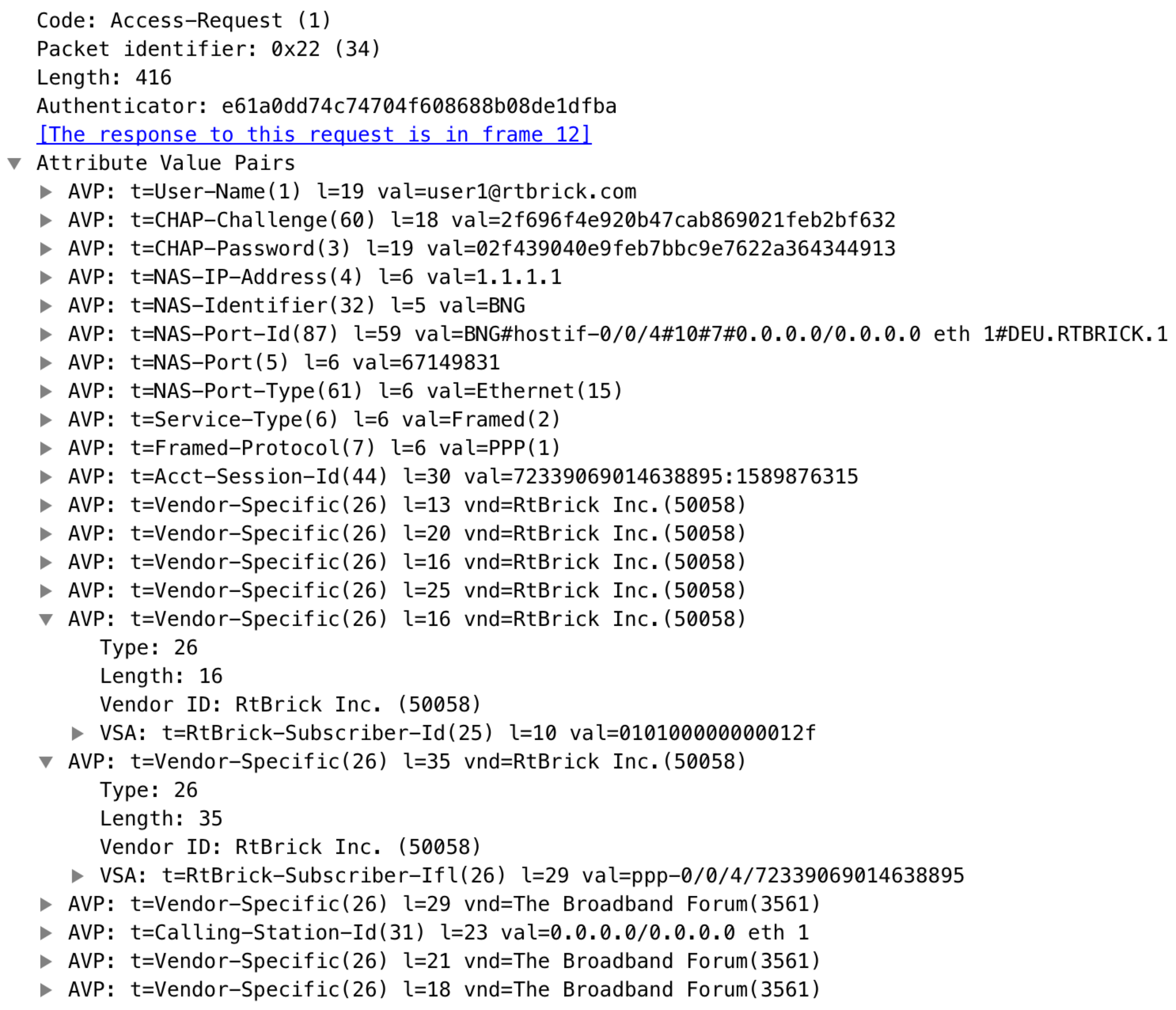
This document describes the subscriber management RADIUS service implementation on RBFS. The term subscriber describes an access user or session from a higher level decoupled from underlying protocols like PPPoE or IPoE. Subscribers in RBFS can be managed locally or via RADIUS where this document considers RADIUS only. Each subscriber is uniquely identified by a 64bit number called subscriber-id. Remote Authentication Dial-In User Service (RADIUS) is a networking protocol that provides centralized Authentication, Authorization and Accounting (AAA) management for all types of subscribers (PPPoE, or IPoE). RADIUS servers can perform as authentication and accounting servers or change of authorization (CoA) clients. Authentication servers maintain authentication records for subscribers. The subscriber daemon requests authentication in RADIUS access-request messages before permitting subscribers access. Accounting servers handle accounting records for subscribers. The subscriber daemon transmits RADIUS accounting-start, interim and stop messages to the servers. Accounting is the process of tracking subscriber activity and network resource usage in a subscriber session. This includes the session time called time accounting and the number of packets and bytes transmitted during the session called volume accounting.

A RADIUS server can behave as a change of authorization (CoA) client allowing dynamic changes for subscriber sessions. The subscriber daemon supports both RADIUS CoA messages and disconnect messages. CoA messages can modify the characteristics of existing subscriber sessions without loss of service, disconnect messages can terminate subscriber sessions.

Each RADIUS request from subscriber daemon includes the RADIUS accounting-session-id attribute (type 44) with a format which is configurable in the AAA profile and includes at least the subscriber-id to identify the corresponding subscriber. The default format (<subscriber-id>.<timestamp>) includes also an unix timestamp to ensure that the tuple of NAS-Identifier (e.g. hostname) and Accounting-Session-Id is global unique to be usable as key in RADIUS databases.

Additionally to subscriber-id and accounting-session-id each subscriber consists also of a subscriber-ifl build based on physical port information and subscriber-id (ifp: ifp-0/0/1 and subscriber-id: 72339069014638610 subscriber-ifl: ppp-0/0/1/72339069014638610) which is required as handle in the RBFS forwarding infrastructure. All those three informations are part of the RADIUS access-request message:

- accounting-session-id: standard attribute 44
- subscriber-id: RtBrick VSA (26-50058-25 RtBrick-Subscriber-Id)
- subscriber-ifl: RtBrick VSA (26-50058-26 RtBrick-Subscriber-Ifl)



```
    Code: Access-Request (1)
    Packet identifier: 0x22 (34)
    Length: 416
    Authenticator: e61a0dd74c74704f608688b08de1dfba
    [The response to this request is in frame 12]
  ▼ Attribute Value Pairs
    ▶ AVP: t=User-Name(1) l=19 val=user1@rtbrick.com
    ▶ AVP: t=CHAP-Challenge(60) l=18 val=2f696f4e920b47cab869021feb2bf632
    ▶ AVP: t=CHAP-Password(3) l=19 val=02f439040e9feb7bbc9e7622a364344913
    ▶ AVP: t=NAS-IP-Address(4) l=6 val=1.1.1.1
    ▶ AVP: t=NAS-Identifier(32) l=5 val=BNG
    ▶ AVP: t=NAS-Port-Id(87) l=59 val=BNG#hostif-0/0/4#10#7#0.0.0.0/0.0.0.0 eth 1#DEU.RTBRICK.1
    ▶ AVP: t=NAS-Port(5) l=6 val=67149831
    ▶ AVP: t=NAS-Port-Type(61) l=6 val=Ethernet(15)
    ▶ AVP: t=Service-Type(6) l=6 val=Framed(2)
    ▶ AVP: t=Framed-Protocol(7) l=6 val=PPP(1)
    ▶ AVP: t=Acct-Session-Id(44) l=30 val=72339069014638895:1589876315
    ▶ AVP: t=Vendor-Specific(26) l=13 vnd=RtBrick Inc.(50058)
    ▶ AVP: t=Vendor-Specific(26) l=20 vnd=RtBrick Inc.(50058)
    ▶ AVP: t=Vendor-Specific(26) l=16 vnd=RtBrick Inc.(50058)
    ▶ AVP: t=Vendor-Specific(26) l=25 vnd=RtBrick Inc.(50058)
    ▼ AVP: t=Vendor-Specific(26) l=16 vnd=RtBrick Inc.(50058)
        Type: 26
        Length: 16
        Vendor ID: RtBrick Inc. (50058)
      ▶ VSA: t=RtBrick-Subscriber-Id(25) l=10 val=010100000000012f
    ▼ AVP: t=Vendor-Specific(26) l=35 vnd=RtBrick Inc.(50058)
        Type: 26
        Length: 35
        Vendor ID: RtBrick Inc. (50058)
      ▶ VSA: t=RtBrick-Subscriber-Ifl(26) l=29 val=ppp-0/0/4/72339069014638895
    ▶ AVP: t=Vendor-Specific(26) l=29 vnd=The Broadband Forum(3561)
    ▶ AVP: t=Calling-Station-Id(31) l=23 val=0.0.0.0/0.0.0.0 eth 1
    ▶ AVP: t=Vendor-Specific(26) l=21 vnd=The Broadband Forum(3561)
    ▶ AVP: t=Vendor-Specific(26) l=18 vnd=The Broadband Forum(3561)
```

The subscriber-id is an unsigned 64bit integer which is shown as a hex number in wireshark.

Each subscriber is formed based on configuration profiles and individual settings retrieved via RADIUS. Conflicts between RADIUS defined attributes and profile attributes are solved by prioritizing those received from RADIUS which is common best practices for broadband access concentrators. New subscribers are signalled via RADIUS access-request and either accepted by RADIUS access-accept or rejected by RADIUS access-reject message from RADIUS server. The RADIUS access-accept includes all attributes required to form the subscriber like IP addresses, DNS servers and referenced configuration profiles. Some of those attributes can be changed by RADIUS dynamically using CoA requests without disconnecting the subscriber.

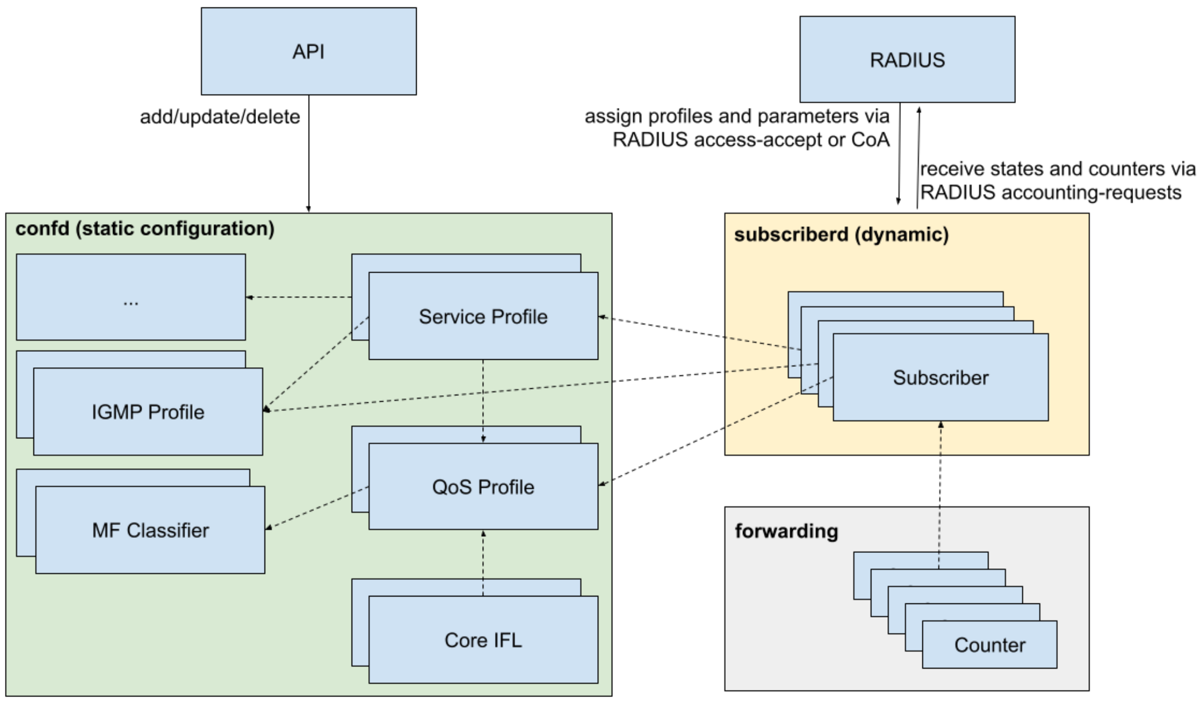
Some of those attributes refer to RADIUS services which are described in detail in this document. The term RADIUS services described the ability to dynamically control the properties of a session via RADIUS used by access providers to build their different products and services based on their network infrastructure. This is used to control QoS settings like shaper rates dynamically based on changed line quality or possible volume quotas. It is also used to dynamically enable or disable services like Voice or IPTV on a per subscriber basis.

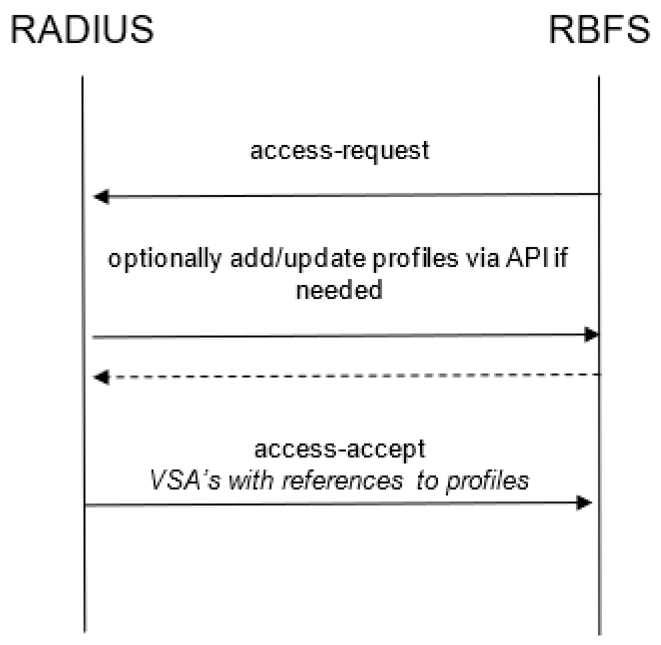
The RADIUS accounting accuracy should comply with §45g of the German Telecommunications Act (TKG) further named by TKG§45. This applies for time and volume based products and services. Products or services with unlimited volumes but with changed properties after a certain amount of traffic are considered as volume based products as well. One example here is a fair use policy which enforces a rate limit after a certain volume. The requirements for time based accounting apply only to products or services which are charged by time or changed properties after a certain time which is not very common in the market today.

2. RADIUS Service Solution Overview

The general concept of RADIUS services in RtBrick FullStack (RBFS) consists of pre-configured profiles in the global static configuration assigned to dynamic subscribers via RADIUS VSA in access-accept and CoA request messages. The profiles are managed as any other configuration in RBFS via corresponding API which is not explained here in detail.

The following pictures show the concept of global static configurations referenced by RADIUS controlled dynamic subscribers.


API
add/update/delete
RADIUS
assign profiles and parameters via RADIUS access-accept or CoA
receive states and counters via RADIUS accounting-requests
confd (static configuration)
...
Service Profile
IGMP Profile
QoS Profile
MF Classifier
Core IFL
subscriberd (dynamic)
Subscriber
forwarding
Counter


RADIUS
RBFS
access-request
optionally add/update profiles via API if needed
access-accept
VSA's with references to profiles

The required profiles and filters should be present before service becomes active which can be done immediately after start-up or between access-request and access-accept as shown in the flow diagram on the left.

3. RADIUS Control

This chapter explains the subscriber and service related RBFS extensions and RtBrick VSA related to RADIUS services.

3.1. RADIUS Session and Idle Timeout

The session and idle timeout values are initialized with zero which means infinity or disabled. Those values will be optionally overwritten with the values in AAA configuration profile and RADIUS access-accept if present. The priority is RADIUS attributes over AAA configuration profile.

The session (attribute 27) and idle (attribute 28) timeout RADIUS attributes are defined in RFC 2865.

The idle timeout is based on outgoing logical interface statistics (OutLIF) for the subscriber-ifl to determine subscriber inactivity.

3.2. RADIUS Service Profile

VSA 26-50058-70 - RtBrick-Service-Profile

Subscribers can be associated with a service-profile which defines the actual service properties like QoS or IGMP profiles and configuration settings. The service profile can be assigned or changed using the RtBrick-Service-Profile VSA.

```
RtBrick-Service-Profile = <service-profile>
```

This attribute is supported via RADIUS access-accept and CoA requests.

All dynamic QoS settings like shaper and policer rates will be reset if the new service-profile includes a qos-profile attribute also if active qos-profile and old qos-profile is equal. All QoS settings remain unchanged if the referenced service profile does not include the qos-profile attribute. If the referenced service profile updates the qos-profile attribute and additional shaper or policer rates are included in the same CoA request which updates the service-profile, those shaper and policer settings will be applied to the new QoS configuration profile after reset. This means that we reset all incremental changes done before. In example if Voice shaper rate has changed to another value, after profile change the default value from profile is used.

All dynamic multicast settings like IGMP status, IGMP profile, IGMP version and IGMP max- members will be reset if this attribute is received via CoA. Therefore assigning a new service profile via CoA without IGMP enabled in the service profile will disable IGMP also without sending RtBrick-IGMP-Status.

RADIUS CoA requests referencing a service profile which is not found on device will be rejected with CoA-NAK (invalid-request) but without changing any subscriber QoS settings. This behavior is different for RADIUS access-accept where service profiles are just ignored if not found. In both cases a warning is generated in subscriber daemon log if a referenced service profile is not found.

3.3. RADIUS AAA Profile

VSA 26-50058-69 - RtBrick-AAA-Profile

This attribute allows to change the associated AAA profile from RADIUS access-accept. This is primarily used to change the accounting adjustment values which are defined in this profile. Simple example here is to use different adjustment values for L2TP and local terminated PPPoE sessions. Another valid use case is to assign different RADIUS accounting servers for in example L2TP sessions or wholesale customers.

```
RtBrick-AAA-Profile = <aaa-profile>
```

This attribute is supported via RADIUS access-accept only.

3.4. RADIUS QoS Profile

VSA 26-50058-62 - RtBrick-QoS-Profile

```
RtBrick-QoS-Profile = <qos-profile>
```

This attribute is supported by RADIUS access-accept and CoA request. The QoS configuration profile can be either selected via service-profile or directly using this attribute which has priority of the service-profile.

All dynamic QoS settings like MFC, queue sizes, shaper and policer rates will be reset if this attribute is present in a CoA request also if new qos-profile and old qos-profile is equal. If additional shaper or policer rates are included in the same CoA request which updates the service-profile, those shaper and policer settings will be applied to the new QoS configuration after reset.

The subscriber management infrastructure does not check if a referenced QoS profile exists or not. This is handled by forwarding infrastructure which continues processing the subscriber QoS settings as soon as the profile was added. In the meantime there is no QoS configuration applied.

3.4.1. RADIUS QoS Parent Scheduler

VSA 26-50058-64 - RtBrick-QoS-Parent-Scheduler

The parent scheduler element of the scheduler-map assigned to the subscriber can be selected with this attribute. If not present, the scheduler-map will be directly bound to the local IFP where the session is established.

This attribute is supported in RADIUS access-accept only (no CoA support) and will set the parent scheduler element of the subscriber.

Providing a QoS parent scheduler which is not present on the corresponding IFP will lead to black howling of all egress data traffic. Control traffic is not impacted and therefore the session will remain.

3.4.2. RADIUS QoS Shaper

VSA 26-50058-63 - RtBrick-QoS-Shaper

Subscribers can be associated with a QoS profile which is assigned via service-profile or directly via corresponding VSA (RtBrick-QoS-Profile). This profile is used to instantiate the QoS resources for the subscriber including schedulers, queues and shapers. The assigned shaper instances can be updated using the RtBrick-QoS-Shaper VSA (attribute 26-50058-63) which will apply to the QoS instance of the corresponding subscriber only, but not to the other subscribers using the same QoS profile. It is only possible to update existing shapers dynamically but it is not possible to create a new shaper via RADIUS.

The RtBrick-QoS-Shaper value is a string which contains a list of multiple shaper settings separated by semicolon. Each shaper setting contains a shaper name, high flow rate and low flow rate separated by comma. The actual shaper rate is the sum of high and low flow rate.

```
RtBrick-QoS-Shaper = <shaper-name>,<high-kbps>,<low-kbps>;<shaper-name>,...
```

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
RtBrick-QoS-Shaper = name=<shaper-name>,high=<high-kbps>,low=<low-
kbps>;...
```

This attribute is supported by RADIUS access-accept and CoA request.

Updating a single shaper (e.g. voice_shaper) via CoA does not require to include other shapers meaning that only the shapers included in the attribute will be updated and all other shapers remain unchanged.

Assume the following example which adapts the session and voice shaper instance of a subscriber.

```
supervisor@rtbrick: op> show config forwarding-options class-of-service
shaper shaper_session
{
  "rtbrick-config:shaper": {
    "shaper-name": "shaper_session",
    "shaping-rate-high": 48000,
    "shaping-rate-low": 2000
  }
}
supervisor@rtbrick: op> show config forwarding-options class-of-service
shaper shaper_voice
{
  "rtbrick-config:shaper": {
    "shaper-name": "shaper_voice",
    "shaping-rate-high": 1000,
    "shaping-rate-low": 0
  }
}
```

RADIUS VSA Change Session Shaper Only

```
RtBrick-QoS-Shaper: shaper_session,14000,2000
```

RADIUS VSA Change Session and Voice Shaper

```
RtBrick-QoS-Shaper: shaper_session,14000,2000;shaper_voice,2000
```

All active dynamic shapers are stored in the table **global.access.1.subscriber.qos.shaper** to handover those information to forwarding infrastructure. This table can be used to verify the dynamic shapers which are active for a given subscriber.

The CLI command show subscriber <id> qos displays those information in a nice human readable format.

```
supervisor@rtbrick: op> show subscriber 72339069014638610 qos
Subscriber-Id: 72339069014638610
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    IFL: ppp-0/0/1/72339069014638610
    Profile: qos_profile
    Parent: pon1
    Dynamic Shaper: shaper_voice
        Rate Low: 0 kbps
        Rate High: 2000 kbps
    Dynamic Shaper: shaper_session
        Rate Low: 2000 kbps
        Rate High: 14000 kbps
```

A shaper rate of 0 means infinity!

3.4.3. RADIUS QoS Policer

VSA 26-50058-65 - RtBrick-QoS-Policer

The RtBrick-QoS-Policer value is a string which contains a list of multiple policer level settings separated by semicolon. Each setting contains a level, cir, cbs, pir, pbs, max-cir and max-pir separated by comma.

```
RtBrick-QoS-Policer = <level>,<cir>,<cbs>,<pir>,<pbs>,<max-cir>,<max-pir>;<level>...

Example:
RtBrick-QoS-Policer = 1,2000,200;2,8000,800;3,0,800;4,0,800
```

- **level**: 1 (highest priority) to 4 (lowest priority)
- **cir**: Ingress policer committed information rate (kbps)
- **cbs**: Ingress policer committed burst size (kbps)
- **pir**: Ingress policer peak information rate (kbps)
- **pbs**: Ingress policer peak burst size (kbps)
- **max-cir**: max ingress policer committed information rate (kbps)
- **max-pir**: max ingress policer peak information rate (kbps)

If PIR and PBS are not defined, the values from CIR and CBS are used as PIR and PBS as well. The max CIR and max PIR attributes are optional default set to unlimited.

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
RtBrick-QoS-Policer = level=<level>,cir=<cir>,cbs=<cbs>,pir=<pir>,pbs=<pbs>,max-cir=<max-cir>,max-pir=<max-pir>;...

Example:
RtBrick-QoS-Policer = level=4,cir=1m,cbs=256;cir=2m,cbs=512,level=3
```

This attribute is supported by RADIUS access-accept and CoA request.

Updating a single policer level via CoA does not require to include other levels meaning that only the levels included in the attribute will be updated and all other levels remain unchanged. It is only possible to update existing policer levels dynamically but it is not possible to create a new level via RADIUS.

Assume the following example which adapts the just one level as well as all levels of a subscriber.

```
supervisor@rtbrick: op> show config forwarding-options class-of-service
policer policer-residential
{
  "rtbrick-config:policer": {
    "policer-name": "policer-residential",
    "level1-rates": {
      "cir": 1000,
      "cbs": 100,
      "pir": 1000,
      "pbs": 100
    },
    "level2-rates": {
      "cir": 4000,
      "cbs": 400,
      "pir": 4000,
      "pbs": 400
    },
    "level3-rates": {
      "cir": 0,
      "cbs": 800,
      "pir": 0,
      "pbs": 800
    },
    "level4-rates": {
      "cir": 0,
      "cbs": 800,
      "pir": 0,
      "pbs": 800
    },
    "levels": 4,
    "type": "two-rate-three-color"
  }
}
```

RADIUS VSA Change Level 2 Only

```
RtBrick-QoS-Policer: 2,12000,1200
```

RADIUS VSA Change all Levels

```
RtBrick-QoS-Policer: 1,2000,200;2,8000,800;3,0,800;4,0,800
```

All active dynamic policer level settings are stored in the table **global.access.1.subscriber.qos** to handover those information to forwarding infrastructure. This table can be also used to verify the dynamic policer settings for a given subscriber.

The CLI command show subscriber <id> qos displays those information in a nice human readable format.

```
supervisor@rtbrick: op> show subscriber 72339069014638610 qos
Subscriber-Id: 72339069014638610
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    IFL: ppp-0/0/1/72339069014638610
    Profile: qos_profile
    Parent: pon1
    Dynamic Ingress Policer Level 1:
        CIR: 2000 kbps CBS: 200 kbps
        PIR: 2000 kbps PBS: 200 kbps
    Dynamic Ingress Policer Level 2:
        CIR: 8000 kbps CBS: 800 kbps
        PIR: 8000 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 3:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 4:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Shaper: shaper_voice
        Rate Low: 0 kbps
        Rate High: 2000 kbps
    Dynamic Shaper: shaper_session
        Rate Low: 2000 kbps
        Rate High: 14000 kbps
```

3.4.4. RADIUS QoS MFC

VSA 26-50058-66 - RtBrick-QoS-MFC

The multifield classifier can be either derived from qos-profile or directly using this attribute which has priority of the qos-profile.

RtBrick-QoS-MFC = <mfc-name>

This attribute is supported by RADIUS access-accept and CoA request.

The subscriber management infrastructure does not check if a referenced multifield classifier exists or not. This is handled by forwarding infrastructure which continues processing the subscriber QoS settings as soon as the classifier was added.

The string received in these attributes should be stored as mf_classifier_name in the RADIUS attribute object and as well as in the corresponding subscriber-qos object.

3.4.5. RADIUS QoS Queues

VSA 26-50058-67 - RtBrick-QoS-Queues

Subscribers can be associated with a QoS profile which is assigned via service-profile or directly via corresponding VSA (RtBrick-QoS-Profile). This profile is used to instantiate the QoS resources for the subscriber including schedulers, queues and shapers. The assigned queue instances can be updated using the RtBrick-QoS-Queues VSA (attribute 26-50058-67) which will apply to the QoS instance of the corresponding subscriber only, but not to the other subscribers using the same QoS profile. It is only possible to update existing queues dynamically but it is not possible to add queues via RADIUS.

The RtBrick-QoS-Queues value is a string which contains a list of multiple queue settings separated by semicolon. Each queue setting contains a queue name and queue size in bytes separated by comma.

```
RtBrick-QoS-Queues = <queue-name>,<size-bytes>;<queue-name>,<size-bytes>;...
```

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
RtBrick-QoS-Queues = name=<queue-name>,size=<size-bytes>;name=...
```

This attribute is supported by RADIUS access-accept and CoA request.

Updating a single queue (e.g. best-effort) via CoA does not require to include other queues meaning that only the queues included in the attribute will be updated and all other queues remain unchanged.

The subscriber management infrastructure does not check if a referenced queue exists or not. This is handled by forwarding infrastructure which continues processing the subscriber QoS settings as soon as the queue was added.

All active dynamic queues are stored in the table **global.access.1.subscriber.qos.queue** to handover those information to forwarding infrastructure. This table can be also used to verify the dynamic queues which are active for a given subscriber.

The CLI command `show subscriber <id> qos` displays those information in a nice human readable format.

```
supervisor@rtbrick: op> show subscriber 72339069014638610 qos
Subscriber-Id: 72339069014638610
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    IFL: ppp-0/0/1/72339069014638610
    Profile: qos_profile
    Parent: pon1
    Dynamic Ingress Policer Level 1:
        CIR: 2000 kbps CBS: 200 kbps
        PIR: 2000 kbps PBS: 200 kbps
    Dynamic Ingress Policer Level 2:
        CIR: 8000 kbps CBS: 800 kbps
        PIR: 8000 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 3:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 4:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Shaper: shaper_voice
        Rate Low: 0 kbps
        Rate High: 2000 kbps
    Dynamic Shaper: shaper_session
        Rate Low: 2000 kbps
        Rate High: 14000 kbps
    Dynamic Queue: voice
        Size: 200000 byte
```

3.4.6. RADIUS IGMP Attributes

IGMP service can be dynamically enabled on a subscriber using Radius IGMP Attributes. These attributes are supported by RADIUS access-accept and CoA requests. Changes via CoA will reset the existing IGMP configuration therefore changing one attribute like max members requires to send all other attributes as well.

Following IGMP related attributes are supported:

VSA 26-50058-71 - RtBrick-IGMP-Status

This attribute can dynamically enable/disable IGMP for a subscriber

Value	Code	Description
DISABLED	0	Disable IGMP
ENABLED	1	Enable IGMP

VSA 26-50058-72 - RtBrick-IGMP-Profile

RtBrick-IGMP-Profile = <igmp-profile>

This attribute specifies the IGMP-profile to be associated with the subscriber. IGMP profile is configured locally in IGMP with all the IGMP related attributes. This attribute can dynamically associate such a locally configured profile to a subscriber

The subscriber management infrastructure does not check if a referenced IGMP profile exists or not. This is handled by IGMP infrastructure which continues processing the subscriber IGMP settings as soon as the profile was added.

VSA 26-50058-73 - RtBrick-IGMP-Version

This attribute can specify the version of IGMP for a subscriber.

Value	Code	Description
V1	1	IGMP Version 1 (not supported)
V2	2	IGMP Version 2
V3	3	IGMP Version 3 (Default version if this attribute is not set)

VSA 26-50058-74 - RtBrick-IGMP-Max-Members

This integer attribute can specify the number of parallel streaming (maximum IGMP membership) for a subscriber. By default it’s taken as 1 if this attribute is not set.

All active dynamic IGMP attributes are stored in the table **global.access.1.subscriber.igmp.service** to handover those information to IGMP. This table can be also used to verify the dynamic IGMP attributes which are active for a given subscriber.

3.5. RADIUS Ascend-Data-Filter Attribute

Attribute 26-529-242 - Ascend-Data-Filter

The Ascend-Data-Filter attribute describes per subscriber filter terms in a simple binary format as described in the following table. Multiple of those attributes in a single RADIUS access-accept or CoA request message are combined to a dynamic filter where each attribute itself is one filter term. The order of those attributes in the RADIUS message is used to order the corresponding terms in the filter. This means that the first filter in the RADIUS packet has priority over the next and or last filter in the RADIUS packet.

Changes in such a filter via CoA requires that all attributes of the new filter must be sent. Therefore adding a single filter term requires sending the existing filter terms plus the new one or general speaking sending the intended target filter.

This filter is installed as a global packet filter placed before policer in ingress and before scheduler in egress to prevent that traffic dropped here is counted in accounting or consuming rate credits.

Field	**Bytes**	**Values**	**Comments**
Type	1	0 = ignore 1 = IPv4 3 = IPv6	
Action	1	0 = discard 1 = accept	
Direction	1	0 = egress 1 = ingress	
Reserved	1	0	

Source
 	
Address
	IPv4 = 4
	
	IPv6 = 16
		source address	Match on source address is not supported for filters in ingress direction and automatically replaced with the subscriber addresses.

Destination
 	
Address
	IPv4 = 4
	
	IPv6 = 16
		destination address	Match on destination address is not supported for filters in egress direction and automatically replaced with the subscriber addresses.

Source Prefix Length	1	0 = ignore prefix length	The number of leading one bits in the source address mask. Specifies the bits of interest.

Field	Bytes	Values	Comments
Destination Prefix Length	1	0 = ignore prefix length	The number of leading one bits in the destination address mask. Specifies the bits of interest.
Protocol	1	0 = ignore IPv4 protocol number IPv6 next header	
Reserved	1	0	
Source Port	2	port number	UDP/TCP source port in network byte order (big endian)
Destination Port	2	port number	UDP/TCP destination port in network byte order (big endian)
Source Port Qualifier	1	0 = no compare 1 = less than 2 = equal to 3 = greater than 4 = not equal to	The options 1, 3 and 4 are not implemented and mapped to 0 if received.
Destination Port Qualifier	1	0 = no compare 1 = less than 2 = equal to 3 = greater than 4 = not equal to	The options 1, 3 and 4 are not implemented and mapped to 0 if received.

Field	Bytes	Values	Comments
Not Used	0 - N	0	Trailing bytes after destination port qualifier ignored.

The filter can be deleted dynamically by sending a single Ascend-Data-Filter attribute with a zero value.

Example:

```
Ascend-Data-Filter =
0x010001000000000000a0afffe0020000000000000000000000000000000000000

    01          IPv4            // type
    00          discard         // action
    01          ingress         // direction
    00          -               // reserved
    00000000    -               // source address
    0a0afffe    10.10.255.254   // destination address
    00          -               // source prefix length
    20          /32             // destination prefix length
    00          -               // protocol
    00          -               // reserved
    0000        -               // source port
    0000        -               // destination port
    00          -               // source port qualifier
    00          -               // destination port qualifier
    00000...    -               // ignored
```

For ingress filters it is not permitted to match based on source address which is automatically replaced with the subscriber addresses. A similar limitation is valid for egress filters matching on destination address which is also replaced with subscriber addresses.

All active subscriber filters are stored in the table **global.access.1.subscriber.filter** to handover those information to forwarding infrastructure. This table can be also used to verify the filters which are active for a given subscriber.

The CLI command show subscriber <id> acl displays those filters in a nice human readable format.

3.6. RADIUS Access Line Attributes

The access line identification and characterization information are defined in the Broadband Forum (BBF) formerly known DSL Forum attributes including Agent-Remote-Id and Agent-Circuit-Id.

See the following references for more information about access line attributes.

- RFC 4679 DSL Forum Vendor-Specific RADIUS Attributes
- RFC 6320 ANCP
- Broadband Forum TR-101
- draft-lihawi-ancp-protocol-access-extension-04

Those attributes will be sent in RADIUS access and accounting requests to the RADIUS server if learned from the underlying access protocol like PPPoE (BBF TR-101).

RBFS also provides the possibility to set those values via RADIUS by using the same attributes in the RADIUS access-accept or CoA request message.

Access Line Attributes Supported (RADIUS, PPPoE and L2TP):

Attribute	Name
26-3561-1	Agent-Circuit-Id
26-3561-2	Agent-Remote-Id
26-3561-3	Access-Aggregation-Circuit-ID-ASCII
26-3561-6	Access-Aggregation-Circuit-ID-ASCII
26-3561-18	PON-Access-Line-Attributes
26-3561-129	Actual-Data-Rate-Upstream
26-3561-130	Actual-Data-Rate-Downstream
26-3561-131	Minimum-Data-Rate-Upstream
26-3561-132	Minimum-Data-Rate-Downstream
26-3561-133	Attainable-Data-Rate-Upstream
26-3561-134	Attainable-Data-Rate-Downstream
26-3561-135	Maximum-Data-Rate-Upstream
26-3561-136	Maximum-Data-Rate-Downstream
26-3561-137	Minimum-Data-Rate-Upstream-Low-Power
26-3561-138	Minimum-Data-Rate-Downstream-Low-Power
26-3561-139	Maximum-Interleaving-Delay-Upstream
26-3561-140	Actual-Interleaving-Delay-Upstream
26-3561-141	Maximum-Interleaving-Delay-Downstream

Attribute	Name
26-3561-142	Actual-Interleaving-Delay-Downstream
26-3561-144	Access-Loop-Encapsulation
26-3561-145	DSL-Type
26-3561-151	PON-Access-Type
26-3561-155	Expected-Throughput-Upstream
26-3561-156	Expected-Throughput-Downstream
26-3561-157	Attainable-Expected-Throughput-Upstream
26-3561-158	Attainable-Expected-Throughput-Downstream
26-3561-159	Gamma-Data-Rate-Upstream
26-3561-160	Gamma-Data-Rate-Downstream
26-3561-161	Attainable-Gamma-Data-Rate-Upstream
26-3561-161	Attainable-Gamma-Data-Rate-Downstream
26-3561-176	ONT-ONU-Average-Data-Rate-Downstream
26-3561-177	ONT-ONU-Peak-Data-Rate-Downstream
26-3561-178	ONT-ONU-Maximum-Data-Rate-Upstream
26-3561-179	ONT-ONU-Assured-Data-Rate-Upstream
26-3561-180	PON-Tree-Maximum-Data-Rate-Upstream
26-3561-181	PON-Tree-Maximum-Data-Rate-Downstream

This table includes already the new attributes defined in draft-lihawi-ancp-protocol-access-extension-04 which results in possible changes in case draft is updated.

Changes in at least one of those line attributes via CoA triggers a RADIUS interim accounting request with the new values.

Changes of actual data rate upstream/downstream via CoA request may trigger the L2TP LAC to send a CSUN request with updated connection speed to the LNS depending on actual L2TP configuration.

3.7. RADIUS L2TP

Tunnelling of PPPoE sessions via L2TPv2 (L2TP LAC) is supported on RBFS and can be controlled via RADIUS as described in RFC 2868 RADIUS Attributes for Tunnel Protocol Support with some limitations:

- No support of FQDN format for IP addresses
- No support Tunnel-Medium-Type other than IPv4

In addition to the standard attributes, the following vendor specific attributes are supported for L2TP.

VSA 26-50058-40 - RtBrick-L2TP-Pool

Instead of RADIUS tunnel attributes it is also possible to configure local L2TP tunnel pools and assign them with this attribute.

VSA 26-50058-41 - RtBrick-L2TP-Profile

The default L2TP configuration profile assigned in the access configuration profile can be changed in RADIUS access-accept to allow different L2TP configurations based on tunnel endpoints. This is typically used to enable or disable L2TP CSUN updates (RFC5515).

VSA 26-50058-42 - RtBrick-L2TP-Tx-Connect-Speed

VSA 26-50058-43 - RtBrick-L2TP-Rx-Connect-Speed

Those attributes are supported in RADIUS access-accept and CoA requests to set the corresponding L2TP connect speed values directly. Per default the L2TP connect speed is derived from actual data rate upstream/downstream of the configured source (PPPoE-IA or RADIUS) if present or alternatively set to port speed which can be overridden with the connect speed attributes. Changes of connect speed via CoA request will trigger the L2TP LAC to send a CSUN request to the LNS if enabled.

Those two attributes are defined as 4 byte integers with speed defined in kbits.

3.8. RADIUS Lawful Interception

Lawful interception (LI) refers to the facilities in telecommunications and telephone networks that allow law enforcement agencies (LEA) with court orders or other legal authorization to selectively intercept individual subscribers. The term mediation device (MED) used in this document describes the element which receives and optionally converts the intercepted traffic into the format expected by the law enforcement agencies of the corresponding countries.

All of the following attributes must be present in RADIUS access-accept or CoA request to control lawful interception (LI) via RADIUS. Those attributes are salt encrypted using the algorithm described in RFC 2868 for the Tunnel-Password. This encryption algorithm is defined for RADIUS access-accept messages only. To support CoA requests the request authenticator should be replaced with 16 zero bytes which is common industry standard.

The LI action NOOP can be used to obfuscate lawful interception requests (fake requests) to prevent that just the presence of those attributes indicates that a subscriber is intercepted. LI requests via RADIUS will show up in the same table as requests via REST or HTTP RPC API (global.access.1.li_request).

Failed LI activations are not signalled via RADIUS to prevent that just the presence of CoA response NAK shows that LI request is not fake (action NOOP).

VSA 26-50058-140 - RtBrick-LI-Action (salt encrypted integer)

Value	Code	Description
NOOP	0	No action / Ignore LI request
ON	1	Start LI / Add LI request
OFF	2	Stop LI / Delete LI request

VSA 26-50058-141 - RtBrick-LI-Identifier (salt encrypted integer)

Device unique lawful interception identifier (LIID) within the range from 1 to 4194303.

VSA 26-50058-142 - RtBrick-LI-Direction (salt encrypted integer)

Value	Code	Description
INGRESS	1	Ingress mirroring only (from subscriber)
EGRESS	2	Egress mirroring only (to subscriber)
BOTH	3	Bidirectional mirroring (from and to subscriber)

VSA 26-50058-143 - RtBrick-LI-MED-Instance (salt encrypted string)

Routing instance through which the mediation device is reachable.

VSA 26-50058-144 - RtBrick-LI-MED-IP (salt encrypted IPv4 address)

IPv4 address of the mediation device.

VSA 26-50058-145 - RtBrick-LI-MED-Port (salt encrypted integer)

UDP port between 49152 and 65535 set in the mirrored traffic.

3.8.1. RADIUS Terminate Codes

VSA 26-50058-27 - RtBrick-Terminate-Code

The RtBrick-Terminate-Code is sent along with the standard Acct-Terminate-Cause (attribute 49). The list below shows the RtBrick termination codes with the mapping to the standard cause.

RtBrick- Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
0	10 (NAS Request)	NA
1	10 (NAS Request)	Subscriber Management Unknown Error
2	9 (NAS Error)	SubscriberD Internal Error
3	9 (NAS Error)	PPPoED Internal Error
4	9 (NAS Error)	PPPoED Object Deleted *This code is used if an PPPoE session object is deleted which is an indication for a crash or some other issues in PPPoE daemon.*
5	9 (NAS Error)	L2TPD Internal Error
6	9 (NAS Error)	L2TPD Object Deleted *This code is used if an L2TP session object is deleted which is an indication for a crash or some other issues in L2TP daemon.*
7	10 (NAS Request)	AAA Profile Not Found
8	10 (NAS Request)	RADIUS Profile Not Found
9	10 (NAS Request)	Authentication Type Missing
10	10 (NAS Request)	Authentication Order Missing or Invalid
11	10 (NAS Request)	Authentication Failure
12	10 (NAS Request)	Local Authentication Failed

RtBrick- Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
13	10 (NAS Request)	Accounting-Request-On Wait
14	10 (NAS Request)	No RADIUS Authentication Server Configured
15	10 (NAS Request)	RADIUS Authentication Failed
16	17 (User Error)	Authentication Rejected
17	17 (User Error)	Local Authentication Rejected
18	17 (User Error)	RADIUS Authentication Rejected
19	5 (Session Timeout)	Session Timeout
20	4 (Idle Timeout)	Idle Timeout
21	6 (Admin Reset)	Clear Session
22	10 (NAS Request)	RADIUS CoA Disconnect
23	9 (NAS Error)	PPPoE Unknown Error
24	10 (NAS Request)	PPPoE PADT Received
25	9 (NAS Error)	PPPoE LCP Error
26	10 (NAS Request)	PPPoE LCP Generic Error
27	10 (NAS Request)	PPPoE LCP Terminate Request Received
28	10 (NAS Request)	PPPoE LCP Maximum Reject / NAK Exceeded
29	10 (NAS Request)	PPPoE LCP Negotiation Failed
30	10 (NAS Request)	PPPoE LCP Configuration-Request Exceeded
31	10 (NAS Request)	PPPoE LCP Echo-Request Timeout Exceeded
32	9 (NAS Error)	PPPoE PAP Error
33	9 (NAS Error)	PPPoE CHAP Error
34	9 (NAS Error)	PPPoE IPCP Error
35	9 (NAS Error)	PPPoE IP6CP Error
36	9 (NAS Error)	PPPoE NCP Initialization Failed

RtBrick- Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
37	10 (NAS Request)	PPPoE NCP Down
38	6 (Admin Reset)	PPPoE Clear Session
39	10 (NAS Request)	PPPoE Upper Layer Down
40	8 (Port Error)	PPPoE Interface Down
41	10 (NAS Request)	PPPoE Configuration Deleted
42	9 (NAS Error)	PPPoE Access Configuration Profile Not Found
43	9 (NAS Error)	L2TP Unknown Error
44	10 (NAS Request)	L2TP Tunnel Down
45	10 (NAS Request)	L2TP Tunnel Dead
46	10 (NAS Request)	L2TP Tunnel Deleted
47	10 (NAS Request)	L2TP No Tunnel Available
48	10 (NAS Request)	L2TP CDN Request
49	9 (NAS Error)	L2TP Profile Not Found
50	9 (NAS Error)	L2TP Access Configuration Profile Not Found
51	9 (NAS Error)	L2TP No Tunnel Pool Error
52	9 (NAS Error)	L2TP Local Tunnel Pool Error
53	9 (NAS Error)	L2TP RADIUS Tunnel Pool Error
54	6 (Admin Reset)	L2TP Clear Session
55	9 (NAS Error)	Access Configuration Profile Not Found
56	9 (NAS Error)	Local IPv4 Address Pool Not Found
57	10 (NAS Request)	Local IPv4 Address Pool Exhausted
58	9 (NAS Error)	Local IPv6 Prefix Pool Not Found
59	10 (NAS Request)	Local IPv6 Prefix Pool Exhausted

RtBrick- Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
60	9 (NAS Error)	Local IPv6 Delegated Prefix Pool Not Found
61	10 (NAS Request)	Local IPv6 Delegated Prefix Pool Exhausted
62	10 (NAS Request)	PPPoE CHAP Response Timeout
63	10 (NAS Request)	L2TP Session Deleted
64	10 (NAS Request)	Duplicate IPv4 address detected
65	10 (NAS Request)	Duplicate IPv6 prefix detected
66	10 (NAS Request)	Duplicate delegated IPv6 prefix detected
67	9 (NAS Error)	Routing instance not found
68	6 (Admin Reset)	Clear Session Force
69	6 (Admin Reset)	Test AAA Request Object Deleted

It is recommended to raise operational alarms for every termination cause received with a value of 9 (NAS Error) for further investigations.

3.8.2. RADIUS CoA Error Handling

CoA requests should be retried a few times in the unlikely event of CoA NAK with error-cause (RFC 5176 attribute 101) other than session-context-not-found (503) or missing-attribute (402). There is no rollback of failed CoA requests to the former state but retrying the same request is supported because changes are implemented in a more declarative way.

4. RADIUS Accounting

RADIUS accounting servers handle accounting records for subscribers. The subscriber daemon transmits RADIUS Accounting-Start, Interim and Stop messages to these servers. Accounting is the process of tracking subscriber activity and network resource usage in a subscriber session. This includes the session time called time accounting and the number of packets and bytes transmitted during the session called volume accounting.

A RADIUS Acct-Status-Type attribute is used by the RADIUS client (subscriber daemon) to mark the start of accounting (for example, upon booting) by specifying Accounting-On and to mark the end of accounting (for example, just before a scheduled reboot) by specifying Accounting-Off. This message is often used by RADIUS servers to automatically close/terminate all open accounting records/sessions for the corresponding client and therefore must not be sent to servers belonging to a profile which was already used/started for accounting.

Per default, the assumption is that all servers referenced by a RADIUS profile share the same states and therefore accounting-on must be only sent to one of those before the first accounting-start is sent.

RADIUS Accounting-On/Off messages are disabled by default and can be optionally enabled in the RADIUS profile configuration. There is also an additional configuration option to optionally wait for Accounting-On response which prevents any new session until accounting has started meaning that Accounting-On response is received.

Accounting-Off is currently not implemented!

RADIUS accounting requests are often used for billing and therefore should be able to store and retry over a longer period (common up to 24 hours or more) which can be optionally enabled in the RADIUS profile configuration using the accounting backup configuration option.

4.1. Time Accounting

All accounting relevant timestamps are retrieved using the UNIX system call clock_gettime internally stored in the datastore as 64 bit microseconds timestamp with the lower order 32 bit representing the seconds since January 1, 1970 00:00 UTC which is also known as epoch or unix timestamp and the higher order 32 bit representing the microseconds which are per definition less than 1.000.000 (one second). The seconds part is counted in full seconds which is similar to always rounded down.

The RADIUS attribute event-timestamp (type 55) defined in RFC2869 is included in each RADIUS Accounting-Request packet to record the time that this event occurred in seconds since January 1, 1970 00:00 UTC. A max allowed deviation of

500 milliseconds of a timestamp in seconds requires mathematical rounding of the internal 64 bit microseconds timestamps to the 32 bit RADIUS event-timestamp in seconds. This means to round down if the microseconds part of the timestamp is less than 0.5 seconds and rounded up if equal or greater than 0.5 seconds.

Start Timestamps

In RBFS the timestamp indicating the PPPoE session start will be generated after successful negotiation of at least one of the PPP network control protocols (IPCP for IPv4 or IP6CP for IPv6). In case of L2TP tunneled sessions (LAC), the successful L2TP session setup after sending ICCN is used as a start timestamp. In both cases this is also the trigger for the RADIUS accounting-request start where this timestamp is used as RADIUS event-timestamp.

Stop Timestamps

The timestamp indicating the session stop will be generated more or less immediately after termination request (e.g. timeout, user request, nas request, …).

The RADIUS attribute Acct-Session-Time (type 46) defined in RFC2866 indicates how many seconds the user has received service and can only be present in Accounting-Request records where the Acct-Status-Type is set to Interim or Stop. This time will be calculated based on the internal session start and stop time in microseconds and mathematical rounded to seconds. For RADIUS accounting interim requests the current event time is used instead of stop time.

4.2. Volume Accounting

Based on §45g of the German Telecommunications Act (TKG), the maximum deviation for accounting relevant counters must be less than 1% per billing period which is typically one month.

The whole RBFS architecture is based on an event-based publish and subscribe model using the BDS infrastructure. Each process publishes all information and states to BDS and interested processes subscribe to that information. The whole inter process communications (IPC) is replaced by BDS table operations which allows to build a hyper scaled distributed software system. This means related to accounting that each counter is requested from data plane and updated in BDS unsolicited based on configurable intervals typically set between 5 to 30 seconds. This applies to any type of counters like interface counters or QoS statistics. This means that CLI commands or API requests will return the last updated counter and not the counter of the time of request. Therefore each BDS object contains a timestamp indicating the time of the last counter update which allows to use these counters with the required accuracy.

Subscriber volume accounting is based on multiple sources like logical interface (LIF), class/queue-, policer- and control-traffic statistics.

All those counters are layer 2 (L2) counters and some of them may reset through configuration changes like QoS counters after applying a new QoS profile. Therefore the subscriber management application daemon is doing some post processing of all counters to ensure that no accounting information is lost and to calculate the final accounting values. This includes also optional configurable counter adjustments.

There is a BDS database object in the table **local.access.subscriber.accounting** created for each subscriber which stores all counters and attributes required to calculate the actual volume counters. This object is created together with the subscriber and automatically deleted if the actual subscriber object is removed.

The volume accounting counters must not reset if the actual hardware counter is deleted or has changed/reset to zero. Therefore instead of using the source counter directly, the delta since the last interval is calculated and added to the final counter.

The usage of callback functions to counter delete and change events ensures that no accounting relevant information will be lost through reconfiguration of subscriber sessions.

The function which removes the subscriber counters from the data plane will receive the final counters before removal and update them into BDS counter objects before this is deleted.

Interims Volume Counters

RADIUS accounting interim requests will also use the last updated counters with current time as event-timestamp.

Stop Volume Counters

The session termination might be triggered by REST API, CLI (clear subscriber …) RADIUS CoA disconnect request, session timeout, idle timeout or client request (PADT). The corresponding RADIUS accounting stop requests will be delayed to wait for the final counter update but uses the timestamp of the terminate request.

4.3. Interims Accounting

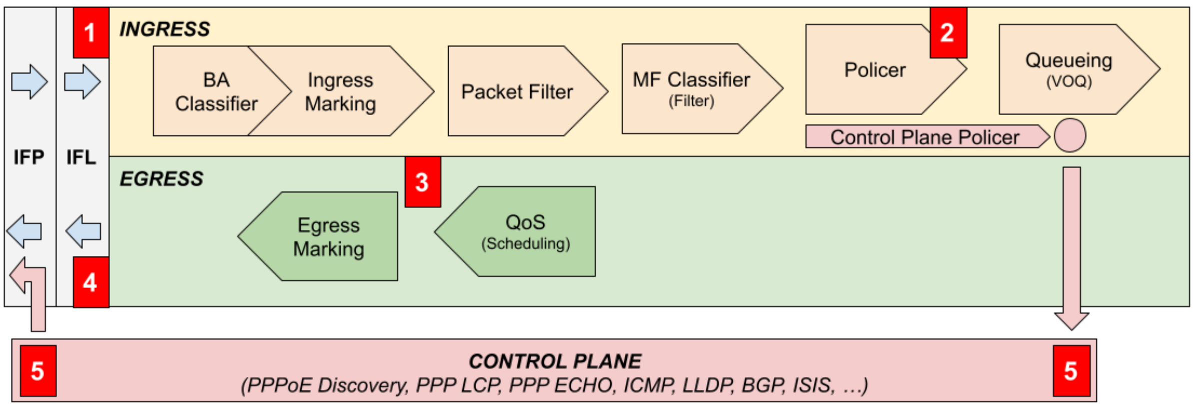
To receive RADIUS counters in fast intervals the corresponding session interim accounting interval can be set depending on the actual needs to any value between one second and multiple hours or days (recommended is at least 30 seconds). This interval can be also updated via CoA request using the Acct-Interim-Interval attribute which triggers an interim accounting immediately and uses the new interval from now onwards. This can be also used to request interim accounting requests on demand. Sending a CoA request with the applied interval triggers an accounting request but the original interval is not changed.

5. RADIUS Counters

The packets and byte counters of each session, traffic class (class 0 - 7) as well as policer counters (level 1 to 4) are supported via RADIUS accounting interims and stop messages. Those counters are layer two counters per default which can be adjusted using correction values and factors from the AAA configuration profile.

Subscriber accounting is based on multiple sources like LIF-, class/queue-, policer- and control-traffic statistics which are described below.

- The InLIF (Incoming Logical Interface) stats count all traffic received on the logical interface including control traffic are traffic dropped by ingress policer.
- The policer stats count all traffic accepted by policer (color green) per level (1-4). Ingress control traffic will be hit by a separate control plane policer and therefore not counted in the session policer stats.
- The class/queue stats count all egress traffic except control traffic which is directly sent to the IFP.
- The OutLIF (Outgoing Logical Interface) stats count all traffic sent on the logical interface except control traffic which is directly sent to the IFP.
- The control stats count all traffic received and sent from or to the control plane for a given subscriber (counted in PPPoE daemon for PPPoE sessions).



Because counter updates are not atomic operations, the sum of all class counters might be different from the session counters.

The counted bytes per packet can be adjusted as described in chapter Configuring Accounting Adjustments of the Subscriber Management Configuration Guide.

All subscriber accounting attributes and counters are stored without adjustment (L2) in the table **local.access.subscriber.accounting** and remain until the subscriber session is finally removed after response to RADIUS accounting stop.

The command show subscriber <id> accounting displays the adjusted subscriber accounting information. It is also possible to display the values before adjustment using show subscriber <id> accounting origin.

```
supervisor@rtbrick: op> show subscriber 72339069014638637 accounting
Subscriber-Id: 72339069014638637
    IFL: ppp-0/0/1/72339069014638637
    Start Timestamp: Wed Feb 24 09:06:47 GMT +0000 2021
    Idle Timestamp: Wed Feb 24 09:06:47 GMT +0000 2021
    Session-Timeout: 86400 seconds
    Idle-Timeout: 7200 seconds
    Session Statistics:
        Ingress: 0 packets 0 bytes
        Egress: 0 packets 0 bytes
    LIF Statistics:
        Ingress: 14 packets 896 bytes
        Egress: 0 packets 0 bytes
    Egress Class (Queue) Statistics:
        class-0: 0 packets 0 bytes
        class-1: 0 packets 0 bytes
        class-2: 0 packets 0 bytes
        class-3: 0 packets 0 bytes
        class-4: 0 packets 0 bytes
        class-5: 0 packets 0 bytes
        class-6: 0 packets 0 bytes
        class-7: 0 packets 0 bytes
    Ingress Policer Statistics:
        Level 1: 0 packets 0 bytes
        Level 2: 0 packets 0 bytes
        Level 3: 0 packets 0 bytes
        Level 4: 0 packets 0 bytes
```

5.1. Session Counters

Per default the session counters are calculated using the LIF statistics which include all traffic received on the logical interface (InLIF) and all traffic sent on the logical interface (OutLIF) except control traffic which is directly sent to the IFP. Ingress traffic sent to CPU or transit traffic dropped by policer is still counted where egress traffic dropped by QoS is not counted.

Alternatively it is possible to use the sum of all policer counters for ingress session counters which can be changed in the AAA configuration profile by setting the accounting ingress source to POLICER (default is LIF) to include only accepted transit traffic.

Using LIF in egress and POLICER in ingress results into a symmetric behavior where only accepted transit traffic is counted.

Following the list of the RtBrick class counter attributes.

Attribute	Name	Description
42	Acct-Input-Octets	Incoming session bytes (uint32)
43	Acct-Output-Octets	Outgoing session bytes (uint32)
47	Acct-Input-Packets	Incoming session packets (uint32)
48	Acct-Output-Packets	Outgoing session packets (uint32)
52	Acct-Input-Gigawords	Acct-Input-Octets overflow counter (uint32)
53	Acct-Output-Gigawords	Acct-Output-Octets overflow counter (uint32)

The internal 64 bit counters are split over the standard 32 bit octet and gigaword counters by using the most significant 32 bit as gigawords and the least significant 32 bits as octets.

32 Bit Acct-Input/Output-Gigawords	32 Bit Acct-Input/Output-Octets
	Internal 64 Bit Counter

5.2. Class Counters

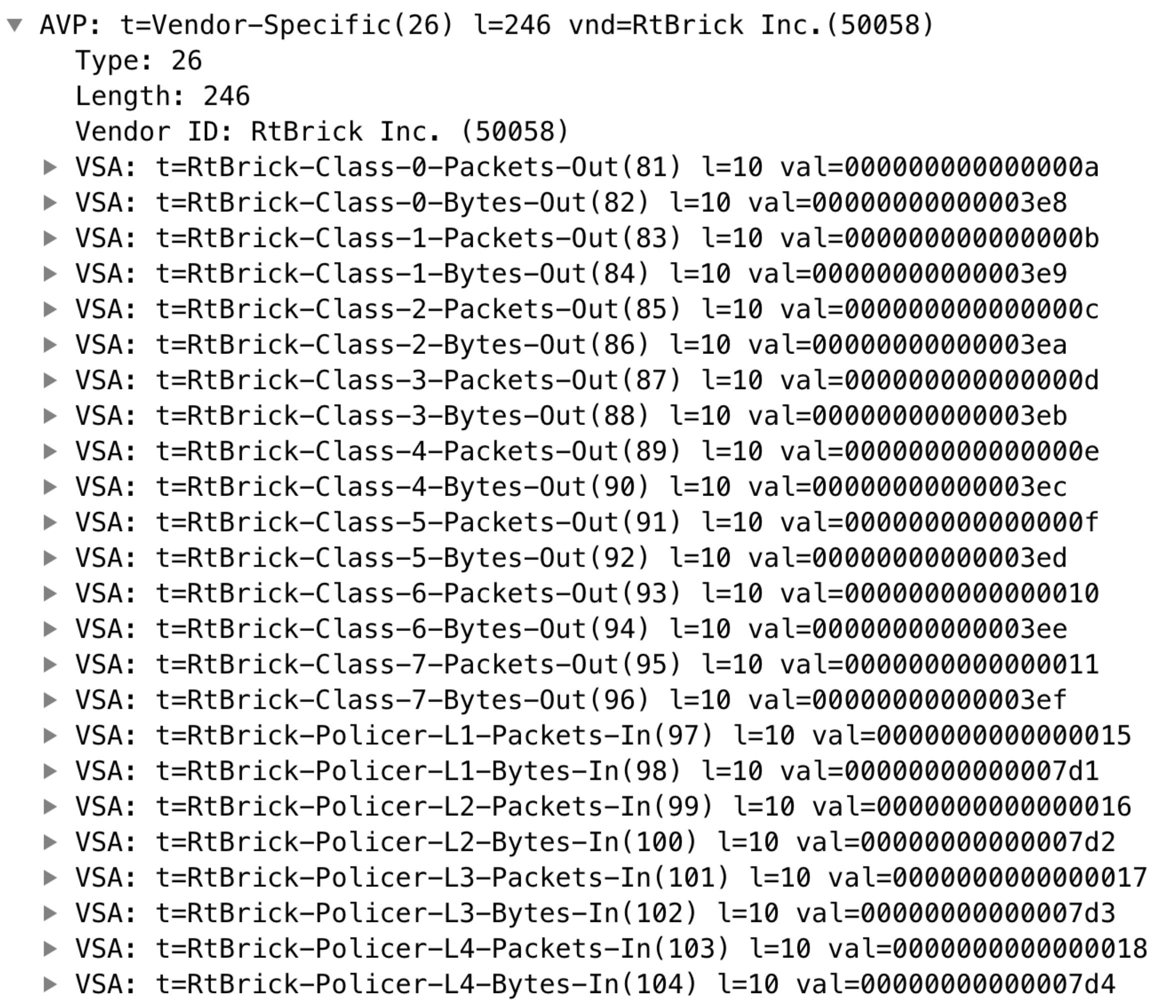
The outgoing class counters are filled by queue counters which ensures that only traffic leaving the device is counted here.

Following the list of the RtBrick class counter attributes.

Attribute	Name	Description
26-50058-81	RtBrick-Class-0-Packets-Out	Outgoing Class-0 packets (uint64)
26-50058-82	RtBrick-Class-0-Bytes-Out	Outgoing Class-0 bytes (uint64)
26-50058-83	RtBrick-Class-1-Packets-Out	Outgoing Class-1 packets (uint64)
26-50058-84	RtBrick-Class-1-Bytes-Out	Outgoing Class-1 bytes (uint64)
26-50058-85	RtBrick-Class-2-Packets-Out	Outgoing Class-2 packets (uint64)
26-50058-86	RtBrick-Class-2-Bytes-Out	Outgoing Class-2 bytes (uint64)
26-50058-87	RtBrick-Class-3-Packets-Out	Outgoing Class-3 packets (uint64)

Attribute	Name	Description
26-50058-88	RtBrick-Class-3-Bytes-Out	Outgoing Class-3 bytes (uint64)
26-50058-98	RtBrick-Class-4-Packets-Out	Outgoing Class-4 packets (uint64)
26-50058-90	RtBrick-Class-4-Bytes-Out	Outgoing Class-4 bytes (uint64)
26-50058-91	RtBrick-Class-5-Packets-Out	Outgoing Class-5 packets (uint64)
26-50058-92	RtBrick-Class-5-Bytes-Out	Outgoing Class-5 bytes (uint64)
26-50058-93	RtBrick-Class-6-Packets-Out	Outgoing Class-6 packets (uint64)
26-50058-94	RtBrick-Class-6-Bytes-Out	Outgoing Class-6 bytes (uint64)
26-50058-95	RtBrick-Class-7-Packets-Out	Outgoing Class-7 packets (uint64)
26-50058-96	RtBrick-Class-7-Bytes-Out	Outgoing Class-7 bytes (uint64)

Those attributes will be encoded as subattributes (RFC2865) similar to broadband forum (BBF) line attributes to reduce the size of the RADIUS message. Counters with a zero value will be also excluded from the packet.



```
▼ AVP: t=Vendor-Specific(26) l=246 vnd=RtBrick Inc.(50058)
    Type: 26
    Length: 246
    Vendor ID: RtBrick Inc. (50058)
  ▶ VSA: t=RtBrick-Class-0-Packets-Out(81) l=10 val=000000000000000a
  ▶ VSA: t=RtBrick-Class-0-Bytes-Out(82) l=10 val=00000000000003e8
  ▶ VSA: t=RtBrick-Class-1-Packets-Out(83) l=10 val=000000000000000b
  ▶ VSA: t=RtBrick-Class-1-Bytes-Out(84) l=10 val=00000000000003e9
  ▶ VSA: t=RtBrick-Class-2-Packets-Out(85) l=10 val=000000000000000c
  ▶ VSA: t=RtBrick-Class-2-Bytes-Out(86) l=10 val=00000000000003ea
  ▶ VSA: t=RtBrick-Class-3-Packets-Out(87) l=10 val=000000000000000d
  ▶ VSA: t=RtBrick-Class-3-Bytes-Out(88) l=10 val=00000000000003eb
  ▶ VSA: t=RtBrick-Class-4-Packets-Out(89) l=10 val=000000000000000e
  ▶ VSA: t=RtBrick-Class-4-Bytes-Out(90) l=10 val=00000000000003ec
  ▶ VSA: t=RtBrick-Class-5-Packets-Out(91) l=10 val=000000000000000f
  ▶ VSA: t=RtBrick-Class-5-Bytes-Out(92) l=10 val=00000000000003ed
  ▶ VSA: t=RtBrick-Class-6-Packets-Out(93) l=10 val=0000000000000010
  ▶ VSA: t=RtBrick-Class-6-Bytes-Out(94) l=10 val=00000000000003ee
  ▶ VSA: t=RtBrick-Class-7-Packets-Out(95) l=10 val=0000000000000011
  ▶ VSA: t=RtBrick-Class-7-Bytes-Out(96) l=10 val=00000000000003ef
  ▶ VSA: t=RtBrick-Policer-L1-Packets-In(97) l=10 val=0000000000000015
  ▶ VSA: t=RtBrick-Policer-L1-Bytes-In(98) l=10 val=00000000000007d1
  ▶ VSA: t=RtBrick-Policer-L2-Packets-In(99) l=10 val=0000000000000016
  ▶ VSA: t=RtBrick-Policer-L2-Bytes-In(100) l=10 val=00000000000007d2
  ▶ VSA: t=RtBrick-Policer-L3-Packets-In(101) l=10 val=0000000000000017
  ▶ VSA: t=RtBrick-Policer-L3-Bytes-In(102) l=10 val=00000000000007d3
  ▶ VSA: t=RtBrick-Policer-L4-Packets-In(103) l=10 val=0000000000000018
  ▶ VSA: t=RtBrick-Policer-L4-Bytes-In(104) l=10 val=00000000000007d4
```

5.3. Policer Counters

The incoming policer attributes count all traffic accepted (colored green) by ingress policers with dedicated packet and byte values per level.

Following the list of the RtBrick policer counter attributes.

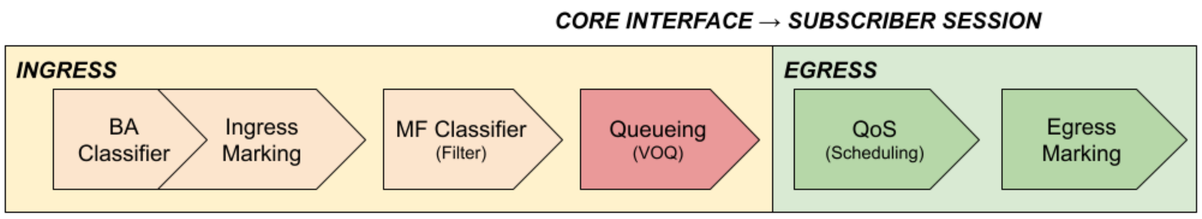
Attribute	Name	Description
26-50058-97	RtBrick-Policer-L1-Packets-In	Incoming Policer L1 accepted packets (uint64)
26-50058-98	RtBrick-Policer-L1-Bytes-In	Incoming Policer L1 accepted bytes (uint64)
26-50058-99	RtBrick-Policer-L2-Packets-In	Incoming Policer L2 accepted packets (uint64)
26-50058-100	RtBrick-Policer-L2-Bytes-In	Incoming Policer L2 accepted bytes (uint64)
26-50058-101	RtBrick-Policer-L3-Packets-In	Incoming Policer L3 accepted packets (uint64)

Attribute	Name	Description
26-50058-102	RtBrick-Policer-L3-Bytes-In	Incoming Policer L3 accepted bytes (uint64)
26-50058-103	RtBrick-Policer-L4-Packets-In	Incoming Policer L4 accepted packets (uint64)
26-50058-104	RtBrick-Policer-L4-Bytes-In	Incoming Policer L4 accepted bytes (uint64)

Those attributes will be encoded as subattributes (RFC2865) similar to broadband forum (BBF) line attributes to reduce the size of the RADIUS message. Counters with a zero value will be also excluded from the packet.

5.4. Service Classification and Accounting

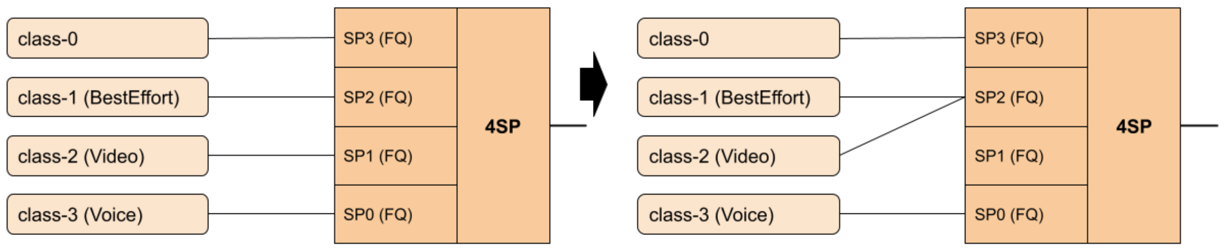
The underlying virtual output queueing (VOQ) is a common and efficient architecture for high performance switches but brings some major limitations. One obvious limitation is that queues must be allocated in ingress and canΓÇÖt be changed in egress as shown in the picture below.



Service classification and accounting is based on standard behavior aggregate (BA) and multifield classifier (MF) bound global or to ingress interfaces matching on some protocol fields assigning a traffic class. Downstream traffic (to subscriber) is counted per subscriber and class using queue counters to count only traffic not dropped by QoS. Upstream traffic (from subscribers) is countered per subscriber and policer level.

Some service providers would like to control if premium traffic is handled differently on a per subscriber basis which can be also achieved with VOQ as explained below.

Let us assume a product which handles video streaming services with higher priority or excludes those traffic from actual data volume. With VOQ the classification as Video is done on the ingress core interface equally for all subscribers regardless of the booked product. Now instead of changing the queue per subscriber in egress, the behavior of the queue can be changed to behave equally to BestEffort which is supported in egress per subscriber.



The corresponding Video traffic is now threaded equally to BestEffort but still counted separately. For accounting purposes the traffic of BestEffort and Video can be simply added in the billing infrastructure to get the actual amount of BestEffort traffic.

In upstream, the expected class and policer level can be assigned per subscriber because BA and MF classifiers are bound to the PPPoE session in ingress.

6. RADIUS Accounting Adjustment Configuration

The accounting adjustment allows to do some basic counter adjustment for RADIUS interims and stop accounting request messages using the following parameters from the AAA configuration profile (**global.access.aaa.profile.config**).

This counter adjustment allows normalizing counters with different encapsulations (double tagged, untagged, …) to L3 counters for example.

Parameter	Type	Description
accounting_ingress_source	uint8	Source of session ingress counter (1: LIF or 3: POLICER) *Default: LIF*
accounting_egress_source	uint8	Source of session egress counter (1: LIF or 2: CLASS) *Default: LIF*
ingress_byte_adjustment_value	float	Adjust ingress LIF counters by +/-N bytes per packet *Default: 0.00*
ingress_byte_adjustment_factor	float	Adjust ingress LIF counters by factor (executed after adjustment value) *Default: 1.00*
egress_byte_adjustment_value	float	Adjust egress LIF counters by +/-N bytes per packet *Default: 0.00*
egress_byte_adjustment_factor	float	Adjust egress LIF counters by factor (executed after adjustment value) *Default: 1.00*

Parameter	Type	Description
accounting_egress_class_byte_adjustment_value	float	Adjust egress CLASS counters by +/-N bytes per packet *Default: 0.00*
accounting_egress_class_byte_adjustment_factor	float	Adjust egress POLICER counters by factor (executed after adjustment value) *Default: 1.00*
accounting_ingress_policer_byte_adjustment_value	float	Adjust ingress POLICER counters by +/-N bytes per packet *Default: 0.00*
accounting_ingress_policer_byte_adjustment_factor	float	Adjust ingress LIF counters by factor (executed after adjustment value) *Default: 1.00*

The byte adjustment value supports positive and negative values like -20.0 or 20.0. Provided decimal digits in the adjustment values are ignored.

The byte adjustment factors support positive values and only the first two decimal digits are used like 0.98 (-2%) or 1.02 (+2%).

7. RADIUS Redundancy

It is possible to configure multiple RADIUS authentication and accounting servers for redundancy and or load-balancing.

The following two algorithms are supported:

- **DIRECT (default):** Requests are sent to the server following the one where the last request was sent. If the subscriber daemon receives no response from the server, requests are sent to the next server and so on.
- **ROUND-ROBIN:** Requests are sent to the server following the one where the last request was sent. If the subscriber daemon router receives no response from the server, requests are sent to the next server and so on.

include::src/radius_service_examples.adoc]

8. Test AAA

The test AAA subscriber feature allows operators to verify and test authentication and accounting (e.g. local or RADIUS) by emulating a subscriber without the need for external clients to be connected. This is a commonly used feature used during troubleshooting or to validate the RADIUS configuration.

Test subscribers will be created by adding a request object into the test aaa request table (**global.subscriber.1.test.aaa.request**) via API or CLI. This request object includes beside username and password also a spoofed interface, outer-vlan, inner-vlan and MAC address required to build corresponding attributes like NAS port identifiers or the vendor specific RtBrick-Access-Stack. It is also possible to add an Agent-Circuit-Id or Agent-Remote-Id to test line based authentication.

This new object will first trigger a validation plugin to reject invalid requests (e.g. subscriber-id out of range, missing mandatory attributes, …). The add callback for this object will trigger the creation of the actual subscriber. Deleting the request object will trigger the termination of the subscriber. If this subscriber terminates for other reasons like CoA or CLI, the subscriber remains in a terminated state until the request object is deleted.

Compared to real subscribers, test subscribers will not trigger any forwarding state. Therefore no object will be created in the following tables for those subscribers:

- global.access.1.subscriber.ifl

But the following tables will be populated for operators to check dynamic QoS, filters or IGMP:

- global.access.1.subscriber.qos
- global.access.1.subscriber.qos.shaper
- global.access.1.subscriber.filter
- global.access.1.subscriber.igmp.service

Those entries are ignored internally as there is no matching subscriber interface (IFL) (global.access.1.subscriber.ifl). The only exception here is the filter table meaning that received filters (ADF) learned from RADIUS are installed in the ACL table and applied with the IP addresses of the test subscribers.

Test AAA Subscriber-Id:

In RBFS each subscriber is uniquely identified by subscriber-id which is a 64 bit number allocated in a distributed fashion by the application creating the subscriber (for example, pppoed for pppoe sessions). This uniqueness is achieved with the following format.

The first 8 bits are used to identify the application:

- 0x00 subscriberd
- 0x01 pppoed
- 0x02 l2tpd

The next 8 bits are used to identify the application sharding instance allocated (for example, pppoed.1, pppoed.2, and so on).

The remaining 48 bits are used to uniquely identify the subscriber, which theoretically allows up to 281474976710656 subscribers per application instance.

An emulated test subscriber needs also a subscriber-id.

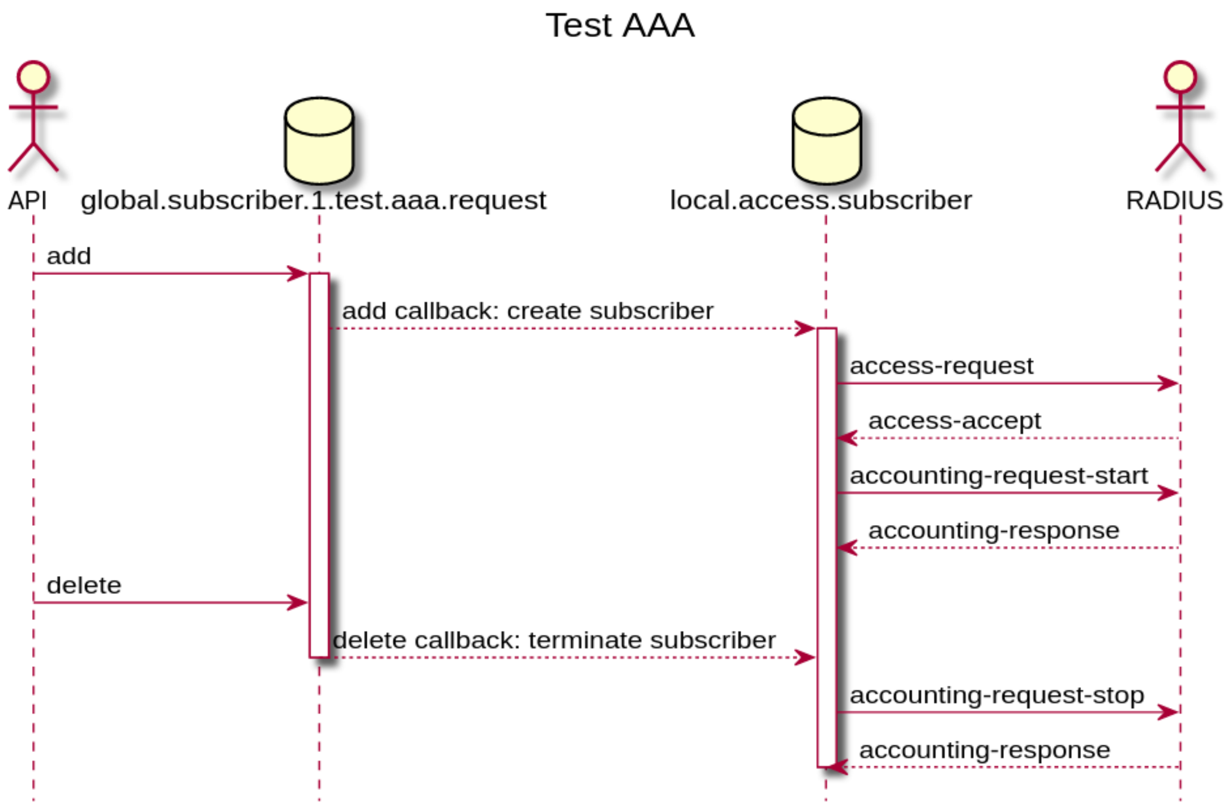
```
-----
* 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

* +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

* | app-id (0) | app-instance | reserved |

* +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

* | subscriber

* +-+-+-+-+

-----
```

Assuming subscriber daemon instance 1 (subscriberd.1), the valid range for test subscriber identifiers is between 281474976710656 and 281479271677951.

```
Min Subscriber-Id: 281474976710656
0000 0000 0000 0001 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000

Max Subscriber-Id: 281479271677951
0000 0000 0000 0001 0000 0000 0000 0000 1111 1111 1111 1111 1111 1111 1111
1111
```

The subscriber identifier will be allocated manually by operators from this range which has the advantage to easier identify the subscriber created by request.

Test AAA RADIUS Flow:



8.1. Test AAA Attributes

Attribute	Mandatory	API	CLI	Default
Subscriber-Id	Yes	subscriber_id	subscriber-id	
Access Profile	Yes	access_profile_name	access-profile	
AAA Profile	Yes	aaa_profile_name	aaa-profile	
Username	No	user_name	username	test
Password	No	password	password	test
Agent-Circuit-id	No	aci	aci	
Agent-Remote-id	No	ari	ari	
Interface (IFP)	*	interface_name	interface	ifp-0/0/0
Outer VLAN	No	outer_vlan	outer-vlan	0
Inner VLAN	No	inner_vlan	inner-vlan	0
MAC Address	No	client_mac	**	00:00:00:00:00:00

** The interface is mandatory via CLI but optional via API.*

*** The client MAC address is currently supported via API only.*

8.2. Test AAA via CLI

Note: In CLI strings must not contain the ΓÇ£#ΓÇ¥ and therefore this is not permitted for username, aci or ari set via CLI. This limitation is valid for CLI only as subscribers created via API are allowed to use this the ΓÇ£#ΓÇ¥ in the mentioned strings.

Add Test Subscriber:

```
supervisor@leaf1: op> test subscriber aaa request 281474976710656 pppoe-dual
aaa-default hostif-0/0/1 username user1@rtbrick.com aci "0.0.0.0/0.0.0.0 eth
0/0" ari DEU.RTBRICK.01
```

List All Test Subscribers:

```
supervisor@leaf1: op> show subscriber test
Subscriber-Id          Interface        VLAN      State             Username
281474976710656        hostif-0/0/1     0:0       ESTABLISHED
user1@rtbrick.com
```

Delete Test Subscriber:

```
supervisor@leaf1: op> test subscriber aaa delete 281474976710656
```

8.3. Test AAA via API

Add Test Subscriber:

```
{{rbfs_url}}/api/v1/rbfs/elements/{{element}}/services/subscriberd.1/proxy/bd
s/object/add
```

```
{
   "table": {
       "table_name": "global.access.1.test.aaa.request"
   },
   "objects": [
       {
           "attribute": {
               "subscriber_id": 281474976710658,
               "aaa_profile_name": "aaa-default",
               "access_profile_name": "pppoe-dual",
               "user_name": "user@rtbrick.com",
               "ari": "DEU.RTBRICK.01",
               "aci": "0.0.0.0/0.0.0.0 eth 0/0"
           }
       }
   ]
}
```

Delete Test Subscriber:

```
{{rbfs_url}}/api/v1/rbfs/elements/{{element}}/services/subscriberd.1/proxy/bd
s/object/delete
```

```
{
   "table": {
       "table_name": "global.access.1.test.aaa.request"
   },
   "objects": [
       {
           "attribute": {
               "subscriber_id": 281474976710658
           }
       }
   ]
}
```

List All Test Subscribers:

```
{{rbfs_url}}/api/v1/rbfs/elements/{{element}}/services/subscriberd.1/proxy/bd
s/table/walk
```

```
{
   "table": {
       "table_name": "global.access.1.test.aaa.request"
   }
}
```

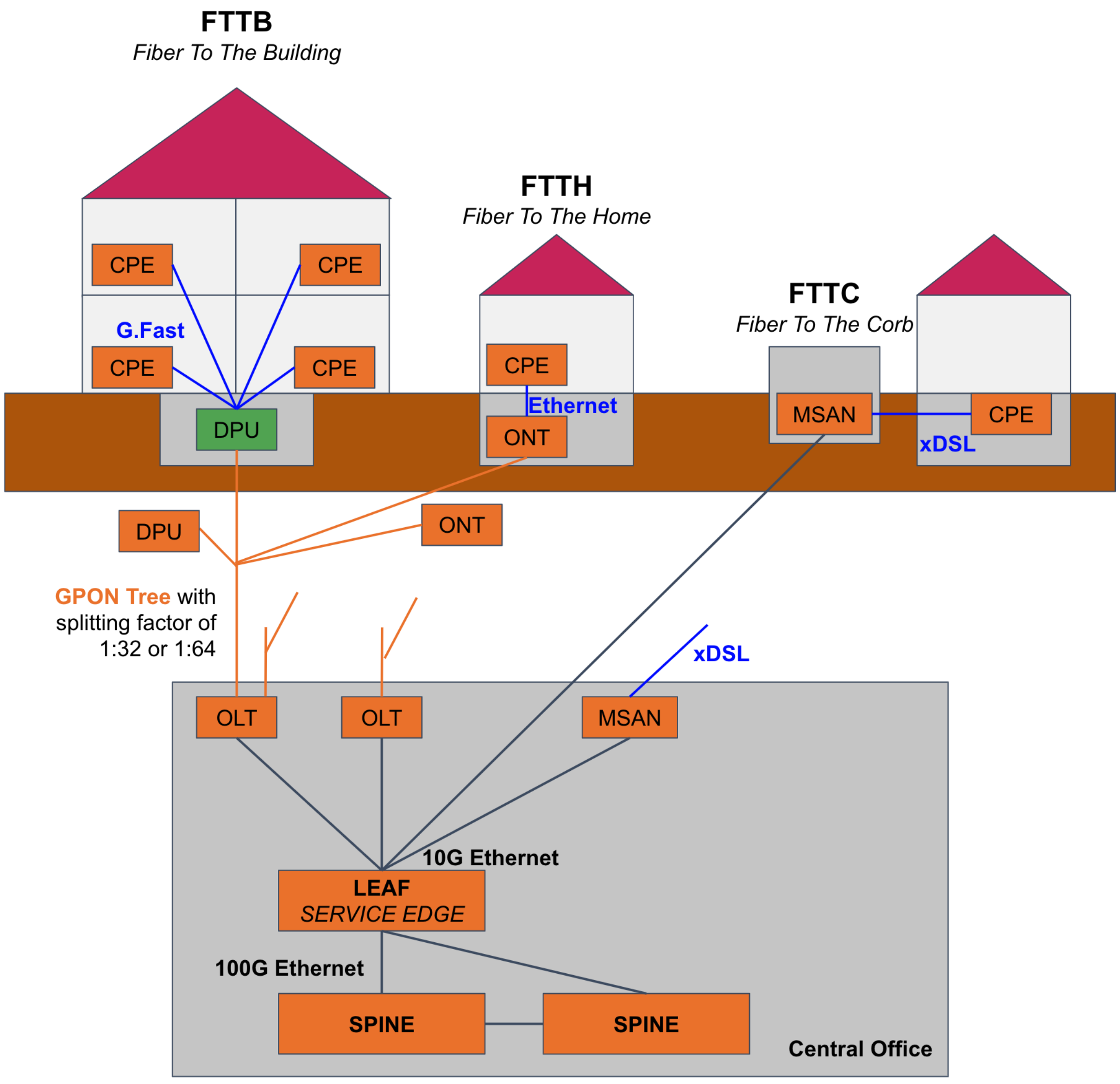
9. Appendix A - RADIUS Dictionary

Following the RtBrick RADIUS dictionary in the well known FreeRADIUS format.

```
# This dictionary applies to RtBrick Full Stack (RBFS)
# https://www.rtbrick.com/
#
VENDOR          RtBrick 50058

BEGIN-VENDOR    RtBrick

ATTRIBUTE       RtBrick-Access-Hostname              21  string
ATTRIBUTE       RtBrick-Access-Port                  22  string
ATTRIBUTE       RtBrick-Access-Stack                 23  string
ATTRIBUTE       RtBrick-Access-MAC-Address           24  string

ATTRIBUTE       RtBrick-Subscriber-Id                25  integer64
ATTRIBUTE       RtBrick-Subscriber-Ifl               26  string

ATTRIBUTE       RtBrick-Terminate-Code               27  integer

ATTRIBUTE       RtBrick-L2TP-Pool                    40  string
ATTRIBUTE       RtBrick-L2TP-Profile                 41  string
ATTRIBUTE       RtBrick-L2TP-Tx-Connect-Speed        42  integer
ATTRIBUTE       RtBrick-L2TP-Rx-Connect-Speed        43  integer

ATTRIBUTE       RtBrick-QoS-Profile                  62  string
ATTRIBUTE       RtBrick-QoS-Shaper                   63  string
ATTRIBUTE       RtBrick-QoS-Parent-Scheduler         64  string
ATTRIBUTE       RtBrick-QoS-Policer                  65  string
ATTRIBUTE       RtBrick-QoS-MFC                      66  string
ATTRIBUTE       RtBrick-QoS-Queues                   67  string

ATTRIBUTE       RtBrick-AAA-Profile                  69  string
ATTRIBUTE       RtBrick-Service-Profile              70  string

ATTRIBUTE       RtBrick-IGMP-Status                  71  integer
ATTRIBUTE       RtBrick-IGMP-Profile                 72  string
ATTRIBUTE       RtBrick-IGMP-Version                 73  integer
ATTRIBUTE       RtBrick-IGMP-Max-Members             74  integer

ATTRIBUTE       RtBrick-Class-0-Packets-Out          81  integer64
ATTRIBUTE       RtBrick-Class-0-Bytes-Out            82  integer64
ATTRIBUTE       RtBrick-Class-1-Packets-Out          83  integer64
ATTRIBUTE       RtBrick-Class-1-Bytes-Out            84  integer64
ATTRIBUTE       RtBrick-Class-2-Packets-Out          85  integer64
ATTRIBUTE       RtBrick-Class-2-Bytes-Out            86  integer64
ATTRIBUTE       RtBrick-Class-3-Packets-Out          87  integer64
ATTRIBUTE       RtBrick-Class-3-Bytes-Out            88  integer64
ATTRIBUTE       RtBrick-Class-4-Packets-Out          89  integer64
ATTRIBUTE       RtBrick-Class-4-Bytes-Out            90  integer64
ATTRIBUTE       RtBrick-Class-5-Packets-Out          91  integer64
ATTRIBUTE       RtBrick-Class-5-Bytes-Out            92  integer64
ATTRIBUTE       RtBrick-Class-6-Packets-Out          93  integer64
ATTRIBUTE       RtBrick-Class-6-Bytes-Out            94  integer64
ATTRIBUTE       RtBrick-Class-7-Packets-Out          95  integer64
ATTRIBUTE       RtBrick-Class-7-Bytes-Out            96  integer64

ATTRIBUTE       RtBrick-Policer-L1-Packets-In        97  integer64
ATTRIBUTE       RtBrick-Policer-L1-Bytes-In          98  integer64
ATTRIBUTE       RtBrick-Policer-L2-Packets-In        99  integer64
ATTRIBUTE       RtBrick-Policer-L2-Bytes-In         100  integer64
ATTRIBUTE       RtBrick-Policer-L3-Packets-In       101  integer64
ATTRIBUTE       RtBrick-Policer-L3-Bytes-In         102  integer64
ATTRIBUTE       RtBrick-Policer-L4-Packets-In       103  integer64
ATTRIBUTE       RtBrick-Policer-L4-Bytes-In         104  integer64

ATTRIBUTE       RtBrick-DNS-Primary                 131  ipaddr
ATTRIBUTE       RtBrick-DNS-Secondary               132  ipaddr
ATTRIBUTE       RtBrick-DNS-Ternary                 133  ipaddr
ATTRIBUTE       RtBrick-DNS-Primary-IPv6            134  ipv6addr
ATTRIBUTE       RtBrick-DNS-Secondary-IPv6          135  ipv6addr
ATTRIBUTE       RtBrick-DNS-Ternary-IPv6            136  ipv6addr

ATTRIBUTE       RtBrick-Instance-IPv4               137  string
ATTRIBUTE       RtBrick-Instance-IPv6               138  string

ATTRIBUTE       RtBrick-LI-Action                   140  integer encrypt=2
ATTRIBUTE       RtBrick-LI-Identifier               141  integer encrypt=2
ATTRIBUTE       RtBrick-LI-Direction                142  integer encrypt=2
ATTRIBUTE       RtBrick-LI-MED-Instance             143  string  encrypt=2
ATTRIBUTE       RtBrick-LI-MED-IP                   144  ipaddr  encrypt=2
ATTRIBUTE       RtBrick-LI-MED-Port                 145  integer encrypt=2

#         VALUE MAPS
#
#       Attribute                Name         Number
VALUE   RtBrick-LI-Action        NOOP         0
VALUE   RtBrick-LI-Action        ON           1
VALUE   RtBrick-LI-Action        OFF          2

VALUE   RtBrick-LI-Direction     INGRESS      1
VALUE   RtBrick-LI-Direction     EGRESS       2
VALUE   RtBrick-LI-Direction     BOTH         3

VALUE   RtBrick-IGMP-Status      DISABLED     0
VALUE   RtBrick-IGMP-Status      ENABLED      1

VALUE   RtBrick-IGMP-Version     V1           1
VALUE   RtBrick-IGMP-Version     V2           2
VALUE   RtBrick-IGMP-Version     V3           3


END-VENDOR RtBrick
```

10. Appendix B - Access Deployments

The following picture shows different deployments typically used in access networks.



The LEAF and SPINE are bare metal switches running RBFS where the LEAF is responsible for service edge functionalities like PPPoE subscriber termination and the SPINE builds the actual fabric core. Multiple LEAF switches will be connected to a pair of spine switches.