Network Working Group	A. Lior
Internet-Draft	Bridgewater Systems
Intended status: Informational	P. Yegani
Expires: May 03, 2012	Juniper
	K. Chowdhury
	Starent Networks
	H. Tschofenig
	Nokia Siemens Networks
	A. Pashalidis
	KUL
	October 31, 2011

Prepaid Extensions to Remote Authentication Dial-In User Service (RADIUS)

draft-lior-radius-prepaid-extensions-20.txt

Abstract

This document specifies an extension to the Remote Authentication Dial-In User Service (RADIUS) protocol that enables service providers to charge for prepaid services. The supported charging models supported are volume-based, duration-based, and based on one-time events.

Table of Contents

*1. Introduction

*1.1. Terminology

*1.2. Overview

*1.2.1. Architectural Model

*1.2.2. Motivation

*1.3. Assumptions

*1.4. Example Use Case

*2. Supported Features

*2.1. Services and Quotas

*2.2. Resource Pools

*2.3. Rating Groups

*2.4. Tariff Switching

*2.5. Support for Roaming

*2.6. Dynamic Termination

*2.7. One Time Event

*2.7.1. One-Time Charging

*2.7.2. Resource Consumption Query

*2.7.3. Service Price Enquiry

*2.7.4. Balance Check

*2.7.5. Refund

*3. Operations

*3.1. Capability Discovery

*3.2. Authentication and Authorization Operation

*3.3. Session Start Operation

*3.4. Mid-Session Operation

*3.5. Dynamic Operations

*3.5.1. Unsolicited Session Termination Operation

*3.5.2. Unsolicited Change of Authorization Operation

*3.6. Termination Operation

*3.7. Mobile IP Operations

*3.8. Operation Considerations for Multiple Services

*3.8.1. Initial Quota Request

*3.8.2. Quota Update

*3.8.3. Termination

*3.8.4. Dynamic Operations

*3.8.5. Support for Resource Pools

*3.8.6. One-time Charging

*3.8.7. Error Handling

*3.8.8. Accounting Considerations

*4. Attributes

*4.1. PrePaid Accounting Capability (PPAC) Attribute

*4.2. Session Termination Capability Attribute

*4.3. Prepaid Accounting Operation (PPAQ) Attribute

*4.4. Fields

*4.5. Prepaid Tariff Switching (PTS) Attribute

*5. Diameter RADIUS Interoperability

*6. Security Considerations

*7. Table of Attributes

*8. IANA Considerations

*9. Acknowledgements

*10. References

*10.1. Normative References

*10.2. Informative References

*Appendix A. Example flows

*Appendix A.1. A simple flow

*Appendix A.2. A flow with prepaid tariff switching

*Appendix A.3. Resource pools and Rating Groups

*Appendix A.4. One-time charging

*Appendix A.5. Price enquiry

*Appendix A.6. Balance check

*Appendix B. Translation between RADIUS Prepaid and Diameter Credit Control

*Authors' Addresses

1. Introduction

This document specifies an extension to the RADIUS protocol that enables service providers to perform accounting and charging in an "online" fashion. In particular, they enable the service provider to

*(a) ensure that subscriber's remaining funds suffice before the service is delivered, and

*(b) interrupt service provision when the funds are exhausted.

These capabilities are typically used in scenarios where the subscriber maintains a prepaid account with the service provider; hence, this extension is called the "prepaid" extension for RADIUS. The functionality described in this document is often referred as "online charging" in comparison to "offline charging" support provided by RFC 2866 [RFC2866].
Note that this document does not follow the RADIUS design guidelines outlined in RFC 6158 [RFC6158] since this document predates RFC 6158 and documents existing implementations.
The extensions were designed with the following goals in mind:

*Make use of existing infrastructure as much as possible (including enabling the interworking of RADIUS-based and Diameter-based infrastructures), and thereby limit the amount of necessary capital expenditures,

*provide the ability to rate service requests in an "online" fashion,

*provide the ability to charge the user's account prior to service provision,

*protect against revenue loss, i.e., to prevent an end user from obtaining service when the available funds do not suffice,

*protect against fraud, and

*be deployable for a number of services independent of the access network technology.

The architecture between the entities that execute the RADIUS protocols, with the extensions defined in this document, assumes that the rating of chargeable events does not occur in the element that provides the service. Instead, the rating may be performed at a dedicated server, termed the "prepaid-enabled AAA server" or simply "prepaid server" (PPS). Alternatively, the actual rating may occur in an entity related to this prepaid server.
Furthermore, this document assumes that a "quota server" is available which, through co-ordination with the rating entity and an account balance manager, is able to provide a quota indication for a particular user when requested. This quota server may or may not coexist in the prepaid server.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Prepaid Client (PPC):

The entity which triggers the RADIUS message exchange, including the prepaid extensions defined in this document. The PPC provides the service to the users, and executes the RADIUS client which, for the purposes of this document, is termed the "PrePaid Client" (PPC). When the prepaid service is used the PPC collects service event information and reports it while the services is provided to the user. This event information is sent to the PPS using the extensions defined in this document.

Prepaid Server (PPS):

The entity that interacts with the PPC using the RADIUS prepaid extensions defined in this document.

Rating Entity:

This entity converts the credit that is allocated by the PPS into a "quota". This quota is then returned to the requesting PPC via the PPS. The rating entity may also determine that during service provision a tariff switch will occur. In this case the rating entity will include details of when exactly tariff switch will occur.

Quota:

A quota denotes the amount of granted units to be consumed without performing another credit control interaction.

Home Network:

The network which contains the user profile and the user's prepaid account.

Authorize-Only Access Request:

A RADIUS message of type "Access Request" (code field = 1) that contains a "Service-Type" AVP (type = 6) with value "Authorize-Only".

Offline Charging:

Offline charging is a process where charging information for resource usage is collected concurrently with that resource usage. The charging information is then passed through a chain of logical charging functions. At the end of this process, Charging Data Record (CDR) files are generated, which are then transferred to the operator's billing domain for the purpose of subscriber billing and/or inter-operator accounting (or additional functions, e.g., statistics, at the operator's discretion). The billing domain typically comprises post-processing systems, such as the operator's billing system or billing mediation device. In conclusion, offline charging is a mechanism where charging information does not affect, in real-time, the service rendered. [TS32240]

Online Charging:

Online charging is a process where charging information for resource usage is collected concurrently with that resource usage in the same fashion as in offline charging. However, authorization for the network resource usage must be obtained prior to the actual resource

usage to occur. This authorization is granted by the PPS upon request from the PPC. When receiving a resource usage request, the PPS assembles the relevant charging information and generates a charging event in real-time. The PPS then returns an appropriate resource usage authorization. The resource usage authorization may be limited in its scope (e.g., volume of data or duration), therefore the authorization may have to be renewed from time to time as long as the resource usage persists. Note that the charging information utilized in online charging is not necessarily identical to the charging information employed in offline charging. In conclusion, online charging is a mechanism where charging information can affect, in real-time, the service rendered and therefore a direct interaction of the charging mechanism with the control of resource usage is required. [TS32240]

1.2. Overview

This section provides an overview of the prepaid charging models and architectures, which are supported by the extensions described in this document.
A number of models of how to charge customers for services in a prepaid manner are supported:

*Volume-based charging (e.g., 2 Cents/KiloByte).

*Duration-based charging (e.g., 3 Cents/minute).

*Resource-based charging (e.g., 3 videos for 10 Euros)

*Event-based charging (e.g., 7 Cents/ring tone) .

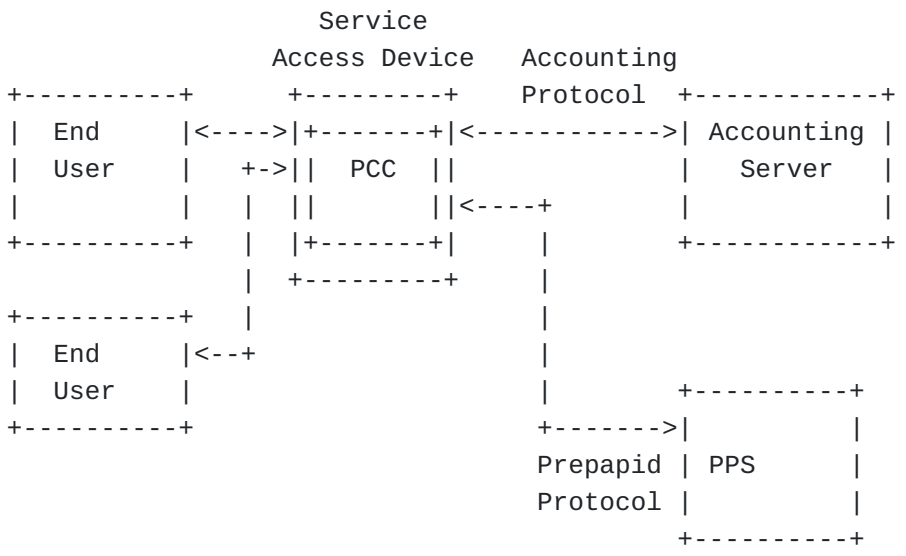
This draft assumes that the user maintains a prepaid account with his home network. This account may be used to fund multiple services, some of which may use the extensions defined in this document, and some may use other mechanisms. The interworking of these mechanisms is outside the scope of this document. Similarly, the means by which the subscriber obtains funds is also outside the scope of this document.

1.2.1. Architectural Model

This section describes the architectural model of the protocol extensions described in this document. Figure 1 describes the involved entities.
The end user establishes a connection with one of possibly multiple PPCs during service access. The selected PPC communicates with a HAAA server (directly or indirectly via a broker network).
The interface between the HAAA and the PPS is implemented using the RADIUS protocol together with the extensions described in this document. However, in cases where the PPS does not implement the RADIUS

protocol, the implementation would have to map the requirements defined in this document to a functionally equivalent protocol.

The requesting PPC meters the consumption of the service according to the instructions provided by the PPS. After service completion, or on reception of a subsequent request for service, the PPS deducts the corresponding amount of credit from the user account. When a user terminates an on-going service, the PPC informs the PPS with a suitable indication about the unused portion of the allocated quota. The PPS then refunds the user account accordingly. Note that multiple PPSs may be deployed for reasons of redundancy and load sharing. The system may also employ multiple rating servers.



The PPS and the accounting server in this architecture MAY be combined.

The PPC must have the ability to meter the consumption of a prepaid data session. This metering is typically based on time (i.e., seconds) or volume (i.e., octets).

The device running the PPC may also have "Dynamic Session Capabilities", such as the ability to terminate a data session or to change the filters associated with a specific data session by processing "Disconnect" messages and "Change of Authorization" messages as per RFC 3576 [RFC3576].

This document assumes that the PPS is used as the AAA server. There are three types of AAA server, as follows.

The AAA server in the home network (HAAA) is responsible for authentication of the subscriber. In addition, the HAAA communicates with the PPS using the RADIUS protocol in order to authorize subscribers.

This document assumes that the PPS communicates with the HAAA for the purposes of authentication and authorisation. The PPS, in turn, interfaces to entities which

*keep the subscriber's account balance (balance manager),

*rate access service requests in real-time (rating engine), and

*manage quota for a particular prepaid service (quota server).

The balance manager, the rating engine and the quota server belong to the service provider's backend infrastructure and are outside the scope of this specification. In particular, as far as this specification is concerned, they are assumed to exist in the PPS.
Accounting messages are not needed to deliver a prepaid service. However, accounting messages can be used to keep the PPS up-to-date as to what is happening with the prepaid data session.

1.2.2. Motivation

Why not use existing RADIUS attributes to construct a protocol for prepaid scenarios? This could lead to a solution where no code has to be modified at existing devices.
It is indeed possible to construct a solution for prepaid scenarios using existing RADIUS attributes. The RADIUS server would send an Access-Accept message containing a Session-Timeout(27) and include a Termination-Action(29) in the RADIUS-request. Upon receiving the Access-Accept message, the NAS would meter the duration of the session and upon termination of the session the NAS would generate an Access-Request message again. The RADIUS server would then re-authenticate the session and reply with an Access-Accept message indicating the amount of additional time in a Session-Timeout(27). Alternatively, it could respond with an Access-Reject message if there were no more resources in the user account.
Moreover, if the user terminates the session prematurely, the NAS could indicate this in the accounting stream so that unused funds can be returned into the prepaid user account.
Unfortunately, the above "solution" has a number of drawbacks, including the following.

*It only supports time-based charging. The solution presented in this document supports multiple charging metrics.

*Using accounting messages to recoup unused time may be problematic because RADIUS accounting messages are not delivered in real-time. A RADIUS server may store-and-forward accounting messages in batches. Thus, relying on accounting messages for the purposes of prepaid may cause revenue leakage. The solution presented in this document does not rely on Accounting packets at all. It uses Access-Request messages, which are required to flow through any network in real-time.

*Session-Timeout(27) is not a mandatory attribute. If a prepaid subscriber is served by a NAS that does not adhere to Session-

Timeout then that subscriber may use the service for an undetermined period of time.

* Termination-Action(29) presents its own issues. Firstly, the behaviour of Termination-Action(29) is not mandatory. Secondly, according to RFC 2865 [RFC2865], Termination-Action fires when the provision of the service has completed. However, service should not be terminated when negotiating additional quota, because this should happen in a manner transparent to the subscriber. Due to the fact that Termination-Action occurs when the service is completed, it is unclear whether or not user experience would be affected if this attribute would be used in a prepaid scenario. The RADIUS server might even allocate a new IP address to the subscriber device after a Termination-Action. Also, the RADIUS server has no way of telling why a given Access-Request message was generated. The RADIUS server might have to wait for the corresponding accounting packet to determine the reason. Finally, re-authenticating the subscriber may take too long. The solution presented in this document allows quota replenishing to occur without affecting user experience. No re-authentication is required and quotas can be negotiated before the available credit actually runs out.

* Due to the fact that the standard RADIUS attributes are not mandatory, the correct prepaid operation is really an act of faith on the part of the RADIUS server. If Session-Timeout(27) and/or Termination-Action(29) are not supported, the prepaid subscriber might be able to obtain the service for free. The solution described in this document requires that a PPC informs the RADIUS server, regardless of whether or not the latter supports the prepaid extensions. The RADIUS server can then determine whether or not service should be granted. For example, if a prepaid subscriber is connected to a NAS that does not support prepaid, the RADIUS server can either instruct the NAS to tunnel the traffic to another entity in the home network (e.g., an Home Agent) that supports prepaid, or cause it to provide only a restricted service.

The solution presented in this document requires the support of two mandatory and one optional attribute. Furthermore, it does not require a great amount of additional code at a NAS (or similar device) that already supports time or volume-based metering. The solution requires that RADIUS entities advertise their prepaid capabilities in an Access-Request and that they generate an Access-Request packet with Service-Type="Authorize-Only" in order to obtain more quota when or before the current quota is used up. It also requires the NAS to send an Access-Request with Service-Type="Authorize-Only" when the session terminates in order to refund the subscriber account.

1.3. Assumptions

This document makes the following assumptions.

*The values carried in the Service Identifiers are pre-configured between the PPC and the PPS.

*The decision about the service rating happens at the PPS.

*The decision whether credit control requests for two services are placed in a resource pool are made by the PPS.

*The decision which services belong to the same rating group are pre-configured at the PPC. Once a rating group is authorized it is not necessary to re-authorize an additional service that belongs to the same rating group at the PPS again.

*A price enquiry is done purely for the purpose of providing AoC for the end user, not for processing at the PPC nor to trigger any specific actions.

1.4. Example Use Case

This section describes the sequence of events in an example RADIUS prepaid transaction.

1. When an end host attaches to a network (for example, using IEEE 802.1X), as usual, the PPC that is servicing the subscriber uses the AAA infrastructure in order to authenticate and authorize the subscriber with respect to the requested service. In order to do this, it sends a RADIUS Access-Request to the AAA server. This Access-Request contains the subscriber's credentials and may contain the prepaid capabilities of the PPC.

2. The authentication procedure proceeds. This may involve several message exchanges, as it is the case with the Extensible Authentication Protocol (EAP) [RFC2284]. Once the subscriber has been successfully authenticated, the home AAA server determines that the subscriber is a prepaid subscriber and requests authorisation from the PPS. This request MUST include the prepaid capabilities of the serving PPC.

3. The PPS, possibly with the help of the backend infrastructure, validates that the subscriber has a prepaid account and that the account is active. It further validates that the PPC has the appropriate prepaid capabilities. If all is in order, the PPS authorises the subscriber to use the network. Otherwise it rejects the request. The decision is sent to the AAA system in the form of a response message. In the case of success, this

message contains attributes that indicate the allocation of a portion of the subscriber credit. This portion is called the "initial quota" and is expressed in units of time or volume. The response may also include a threshold value. Note that only a portion of the user's funds is allocated because the user may be engaged in other services that may draw on the same account. For example, the user may be engaged in a data session and a voice session. Although these two services would draw from the same account, they form separate parts of the overall system. If the entire quota was allocated to the data session then the user would have no more funds for a voice session.

4. The AAA system incorporates the attributes received from the PPS into an Access-Accept message that it sends to the PPC. Note that the AAA system is responsible for authorizing the service whereas the prepaid system is responsible for prepaid authorization.

5. Upon receiving the Access-Response, the PPC starts the prepaid data session and meters the session based on time or volume, as indicated in the message.

6. Once the consumption approaches the allocated limit (as expressed by the threshold), the PPC will request additional quota. Re-authorization for additional quota flows through the AAA system to the PPS. The PPS revalidates the subscriber account and subtracts the previously allocated quota from the current balance. If there is remaining balance, it reauthorizes the request with an additional quota allotment. Otherwise, the PPS rejects the request. Note that the replenishment of the quota is a re-authorization procedure and does not require the subscriber to authenticate himself again.

7. Upon receiving a re-allotment of the quota, the PPC continues to provide the requested service until the new threshold is reached. If the request for additional quota cannot be fulfilled then the PPC lets the subscriber use the remaining quota and terminates the session. Alternatively, instead of terminating the session, the PPC may restrict service access in such a way that the subscriber can only reach a particular web server. This web server maybe used to allow the subscriber to replenish his account. This restriction can also be used to allow new subscribers to set up prepaid accounts in the first place.

8. Should the subscriber terminate the session before the quota is exhausted, the remaining balance allotted to the session is refunded into his account.

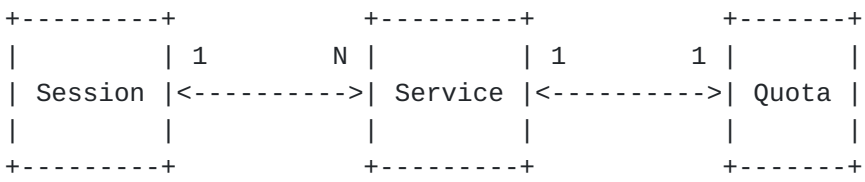
Note that the subscriber may have disconnected while the PPC is waiting for the initial quota. The entire allocated quota will have to be credited back to the subscribers account in this case. Also note that the PPS maintains session state for the subscriber. This state includes how much account balance was allocated during the last quota enquiry and how much is left in the account. Therefore, it is required that all messages about the session reach the same (and correct) PPS.
For a simple message flow, along the lines of this use case, please see Appendix Appendix A.

2. Supported Features

This section describes the features that are supported by the extensions specified in this document.

2.1. Services and Quotas

Examples of services that the user may be using are browsing the web, participating in a VoIP conversation, watching streaming video and downloading a ring tone. Some operators may want to distinguish between these services and to charge them at different rates and meters them differently. Therefore, the prepaid solution needs to be able to distinguish services, and allocate quota to the services using different unit types (time, volume) and allow for those quotas to be consumed at different rates.

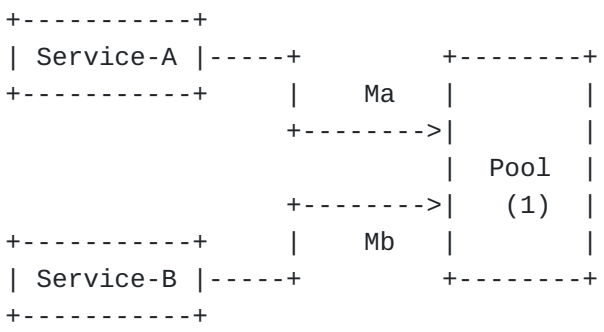


As shown in Figure 2, a session may be associated with multiple services. Each service is identified by a service identifier (Service-ID). The format of the Service-ID is not in the scope of this document. It may, for example, be expressed as a 5-tuple {i.e., source IP address, destination IP address, source port, destination port, and protocol type}. Each service is associated with a quota whereby a quota might be applicable to multiple services. An example message flow that involves multiple services within a single session is given in the Appendix Appendix A.

2.2. Resource Pools

When working with multiple services a new problem arises because one service may consume its quota faster than another service. When the user balance is close to exhaustion, a situation could arise where one service is unable to obtain quota while another service has plenty of quota remaining. Unless the quotas can be rebalanced, the SAD would

then have to terminate the former service. Moreover, it is likely that each service generates a certain amount of RADIUS prepaid traffic. In an environment with many users and charged services, this amount of traffic may become a considerable overhead that could lead to inefficiency.
One method to circumvent the above situation is to use a so-called "resource pool". Resource pools enable the allocation of resources to multiple services of a session by allocating resources to a pool and have services draw their quota from the pool at a rate appropriate to that service. When the quota that has been allocated to the pool is close to exhaustion, the entire pool (rather than individual services) is replenished.



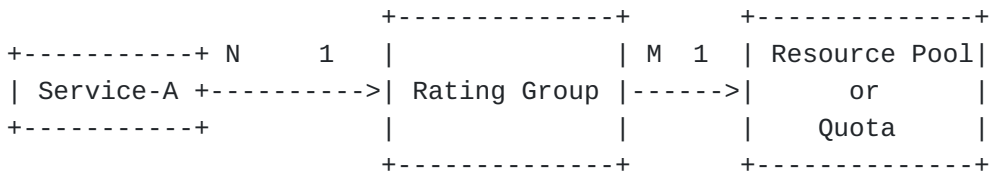
As shown in Figure 3, Service-A and Service-B are bound to Pool(1). Ma and Mb are the pool multipliers (that are associated with Service-A and Service-B respectively) that determine the rate at which Service-A and Service-B draw from the pool.
The pool is initialized by taking the quota allocated to service n and multiplying it by Mn. Therefore, the amount of resources allocated to a pool is given by Poolr = Ma*Qa + Mb*Qb + . . ., where Qn denotes the amount of quota that is allocated to service n. Further, the pool is considered to be empty if

Poolr <= Ca*Ma + Cb*Mb + . . .,

where Ca and Cb are resources consumed by Service-A and Service-B respectively.
Note that the resources assigned to the pool are not associated with a metric. That is, Service-A can be rated at $1 per MB and Service-B can rated at $0.10 per minute. In this case, if $5 worth of resources are allocated for service-A to the pool and if Ma = 10, then 50 units would be placed into the pool. If a further $5 are allocated for service-B to the pool, then M=1 and 50 units are deposited into the pool. The pool would then have a total sum of 100 units to be shared between the two services. The PPC would then mater the services such that each Mbyte used by Service-A will draw 10 units from the pool and each minute used by Service-B will draw 1 unit from the pool.

2.3. Rating Groups

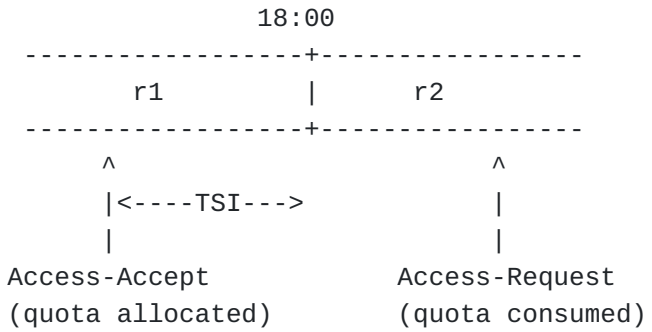
A Rating Group gathers a set of services, identified by a service identifier, and subject to the same cost and rating type (e.g., $0.1/ minute).
The rating of a service can be quite complex. While some operators follow linear pricing models, others may wish to apply more complex functions. For example, a service provider may wish to rate a service such that the first N MBytes are free, then the next M Mbytes are rated at $1 per MB and volume above (N+M) MB be rated at $0.50 per MB. Such a function could be implemented by repeated message exchanges in the prepaid system.
To avert the need to exchange many messages while still supporting such complex rating functions, the concept of the Rating Group was introduced.
As shown in Figure 5, a Rating Group is associated with one or more services and defines the rate that the services associated with the Rating Group consume an allocated amount of quota.



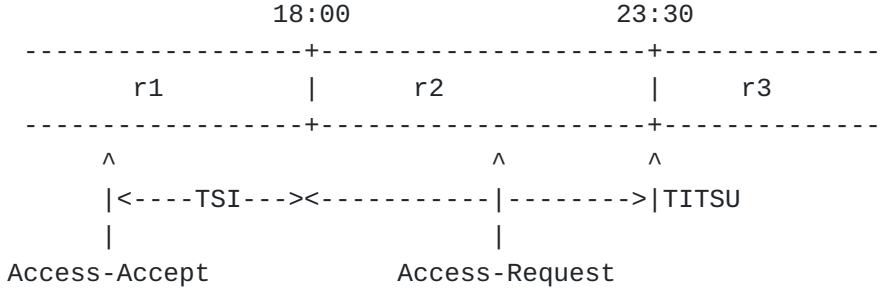
During the usage of a service that is associated with a Rating Group, the PPC sends the ID of the Rating Group to the PPS. The PPS authorises the Rating Group by allocating a quota to it and assign it to a Resource Pool. When an additional service that belongs to an already authorised Rating Group is instantiated, the PPC does not need to re-authorize this service. This effectively means that the PPC meters the service such that it draws from the already allocated quota. Therefore, no RADIUS messages need to be exchanged in this case. This limits the amount of traffic between the PPC and the PPS. An example of a flow that uses Rating Groups is given in Appendix Appendix A.3

2.4. Tariff Switching

Tariff is the set of parameters defining the utilization charges for the use of a particular service.
This mechanism is useful if, for example, as shown in Figure 6, traffic before 18:00 is rated at rate r1 and traffic after 18:00 is rated at rate r2. The mechanism requires the PPC to report usage before and after the switch occured.



The PPC indicates support for tariff switching by setting the appropriate bit in the PPAC. If the PPS needs to signal a tariff switch time it will send a PTS attribute that indicates the point in time when the switch will occur. This indication represents the number of seconds from current time (TariffSwitchInterval TSI).
At some point after the tariff switch the PPC sends another Access-Request, as a result of either the user having logged off or the volume threshold being reached. The PPC reports how much volume was used in total (in a PPAQ attribute) and how much volume was used after the tariff switch (in a PTS VUATS subtype attribute).
In situations with multiple tariff switches, the PPS has to specify the length of the tariff switch period using the TimeIntervalAfterTariffSwitchUpdate (TITSU) field in the PTS attribute, as shown in Figure 7.



When a TITSU is specified in the PTS, the PPC MUST generate an Access-Request within the time after TSI and before TITSU expires. Note that, typically, the PPC will be triggered by the Volume Threshold. However, it is possible that, during period r2, resources are not entirely consumed and, thus, the threshold is not reached. The TITSU attribute ensures that, even in this case, the PPC will generate the new Access-Request in good time.
For time based services, the quota is continuously consumed at the regular rate of 60 seconds per minute. At the time when credit resources are allocated, the server already knows how many units will be consumed before the tariff time change and how many units will be consumed afterward. Similarly, the server can determine the units consumed at the before rate and the units consumed at the rate

afterward in the event that the end user closes the session before the consumption of the allotted quota. There is no need for additional traffic between the PPC and the PPS in the case of tariff time changes for continuous time based service. Therefore, the tariff change mechanism is not used for such services. For time-based services in which the quota is not continuously consumed at a regular rate, the tariff change mechanism described for volume and event units may be used.

2.5. Support for Roaming

In certain networks it is essential for prepaid data services to be available to roaming subscribers. Support for both static and dynamic roaming models is needed. In a static roaming scenario the subscriber connects to a foreign network which has a roaming agreement either directly with the home network, or through a broker network. When the subscriber logs into another foreign network, a new login procedure has to be executed.
In a dynamic roaming scenario the subscriber may move between networks while maintaining his connection. In such a scenario the data session is seamlessly handed off between the networks.
In both roaming scenarios, the subscriber always authenticates himself to the home network. Authorization for the prepaid session and quota replenishing occurs at the home network and more specifically at the PPS where state is being maintained.
Roaming is challenging because a subscriber who established a prepaid data session may move to another PPC that does not support the prepaid extensions.

2.6. Dynamic Termination

When fraud or an error is detected, either only the affected session, or all sessions of the affected subscriber should be immediately terminated. Under certain conditions, the system may wish to terminate the session in order to make sure that the user is not charged for services it does not use.
Certain handoff procedures used in dynamic roaming scenarios require that the system terminates the subscribers prepaid data session at a PPC. This is the case, for example, when time-based prepaid is used and the mobile subscriber performs a dormant handoff.

2.7. One Time Event

2.7.1. One-Time Charging

One-time charging is a mode of operation where the RADIUS prepaid extensions are used for charging of a service that is provided instansteneously. An example of such an event is the purchase of a ring tone. Subscription based services can also be modeled as a one-time

event. In this case the one-time service event is the purchase of a subscription.
For a given user, one-time charging may occur in parallel with other charging models. For example, the subscriber may be connected to the Internet, which is metered (based on time or volume), while he also purchases a ring tone (a one-time-based event).
Note that it is up to the service providers to decide whether or not the user will be charged for the download of, for example, the video and also be charged for the data volume required to download the video. The facilities provided by this document gives the service provider the capability to achieve their service charging business goals.
The PPC signals one-time charging to the PPS with an indication that identifies the service and the units that should be debited from the user account.
A PPC may decide to perform one-time charging and the PPC may need to authenticate the user before sending the relevant message to the user's home AAA server (and to the PPS).
Note that one-time charging can also be used to credit the prepaid account. For example, the PPC can return resources to the subscriber by issuing a one-time charging request that includes the amount of resources to be credited into the account.

2.7.2. Resource Consumption Query

It should be possible for the PPS to query the PPC for the current resource consumption and to adjust the users account balance. For example, a request to the PPS is made (e.g., a one-time charging event), the account is depleted and resources have been allocated to the PPC. The PPS should have the ability to query the PPC and, if it has the spare resources, to reassign the quotas to the PPC and to the pending request. Note that the PPS does not know resource usage until the PPC request for more resources. This can be a long time.
In the absence of this capability the PPS can minimize the effect of this phenomenon by allocating small quotas, a practice that results in more message exchanges.

2.7.3. Service Price Enquiry

The PPC may need to know the price of the service event. Services offered by application service providers whose prices are not known in the PPC might exist. The end user might also want to get an estimation of the price of a service event before requesting it.
A PPC issues a PPAQ to the PPS including the Requested-Action SubType with the value set to "Price Enquiry" (2). The request includes enough information to identify the service, namely a Service-Identifier or a Rating-Group-Identifer.
The PPS calculates the cost of the requested service event, but it does not perform any account balance check or credit reservation from the account.

The estimated cost of the requested service event is returned to the PPS with a PPAQ in the Cost-Information SubType. The PPC may transfer the information to the end user as an advice of charge.
More information regarding the price enquiry functionality is provided in Section 4.3.15 and in Section 4.3.17.

2.7.4. Balance Check

The PPC may only have to verify that the end user's account balance covers the cost of a certain service without reserving any units from the account at the time of the inquiry. This method does not guarantee that credit would be left when the PPC requests the debiting of the account with a separate request.
A PPC issues a PPAQ to the PPS including the Requested-Action SubType with the value set to "Balance Check" (1). The request includes enough information to identify the service, namely a Service-Identifier or a Rating-Group-Identifer.
The PPS makes the balance check, but it does not make any credit-reservation from the account.
The result of balance check, namely "Success" (1) or "Failure" (2), is returned to the PPC in the Check-Balance-Result SubType conveyed in the PPAQ attribute from the PPS to the PPC.
More information regarding the balance check functionality is provided in Section 4.3.15 and in Section 4.3.16.

2.7.5. Refund

Some services may refund service units to the end user's account; for example, gaming services.
To initiate refunding the PPC includes the PPAQ attribute in an Access-Request packet and the amount (as a negative value) to be refunded is specified using the Resource Quota and Resource Quota overflow subtypes. This functionality is similar to one-time charging with the difference that refunding uses negative values
Information about the service need to be provided by the PPC to allow service identification, namely the Service-ID field of the PPAQ identifies the prepaid service.
Note that a monetary amount itself to be refunded is not provided but rather abstract units. Based on prior out-of-band agreements between the PPC and the PPS these abstract units are translated into a monetary amount.
More information regarding the refund functionality is provided in Section 3.8.6.

3. Operations

This section contains the normative text for the prepaid extension.

3.1. Capability Discovery

The PPC initiates the authentication and authorization procedure by sending a RADIUS Access-Request to the HAAA. Since the PPC MUST include a PPAC attribute in the RADIUS Access-Request. The PPAC attribute indicates to the PPS which prepaid capabilities are possessed by the PPC. These are required in order to complete the prepaid authorization procedure.Moreover, if the PPC supports the Disconnect-Message or the Change-of-Authorization capabilities, then it SHOULD include the Session Termination attribute.
In certain deployments, there may be other ways to terminate a data session, or change authorization of an active session. For example, some PPCs provide a session termination service via Telnet or SNMP. In these cases, the AAA server MAY add the Dynamic-Capabilities message to the Access-Request. Upon receiving the Change-of-Authorization message, the AAA server would then be responsible for terminating the session using the means that are supported by the device.
If the authentication procedure involves multiple message exchanges (as it is the case with EAP), the PPC MUST include the PPAC attribute in at least the last Access-Request of the authentication procedure.

3.2. Authentication and Authorization Operation

Once the Access-Request arrives at the HAAA, the HAAA authenticates the subscriber. If this fails, the HAAA sends an Access-Reject message to the client. If authentication succeeds, the HAAA determines whether or not the subscriber is a prepaid subscriber. If the subscriber is not a prepaid subscriber, then the HAAA responds as usual with an Access-Accept or an Access-Reject message. If the subscriber is a prepaid subscriber then the HAAA MAY forward the Access-Request to the PPS for further authorization.
The Access-Request contains the PPAC attribute and the Dynamic-Capabilities attribute if one was included. The User-Name(1) attribute MAY be set to a value that identifies the subscriber. This attribute is used by the PPS to locate his account. For added security, the HAAA MAY also set the User-Password(2) attribute to the password used between the HAAA and the PPS.
The PPS locates the subscriber account and authorizes him. During this procedure, the PPS takes into consideration the PPCs capabilities. Upon successful authorization, the PPS generates an Access-Accept containing an PPAC attribute and an PPAQ attribute. The PPAC attribute returned to the client indicates the type of prepaid service to be provided for the session. The PPAQ attribute includes the following information.

*The QID, which is set by the PPS to a unique value, is used to correlate quota requests.

*Volume and/or Time quota, which is set to a value representing a portion of the subscriber's credit.

*Time or Volume Threshold that indicates when the PPC should request additional quota. This information is optional.

*The IP address of the serving PPS and one or more alternative PPSs. This is used by the HAAA to route subsequent quota replenishing messages to the appropriate PPS(s).

*A State attribute, as defined in RFC 2865 [RFC2865]. This is necessary in order to satisfy the requirements of Section 5.44 of RFC 2865 [RFC2865], which mandates that an Access-Request with Service-Type="Authorize-Only" must contain a State attribute. Since the PPC sends subsequent quota replenishment requests in the form of such "Authorize-Only" requests, a State attribute MUST be present in all Access-Accept messages that also carry a PPAQ attribute.

Note: The Idle-Timeout(28) attribute can be used to trigger the premature termination of a prepaid service, for example as a result of inactivity.
Depending on site policies, after failed authorization, the PPS may generate an Access-Reject in order to terminate the session immediately. Alternatively, the PPS may generate an Access-Accept blocking some or all of the traffic and/or redirect some or all of the traffic to a location to a fixed server. (This feature could be used, for example, to prompt the user to replenish their account.) Blocking of traffic is achieved by either Filter-ID(11) or NAS-Filter-Rule (see [RFC4849]). A description of the redirect functionality is outside the scope of this document. The time period before the session is blocked/ redirected is specified by the Session-Timeout(27) attribute.
Upon receiving an Access-Accept from the PPS, the HAAA appends the usual service attributes and forward the packet to the PPC. The HAAA SHOULD NOT overwrite any attributes already set by the PPS. If the HAAA receives an Access-Reject message, it will simply forward the packet to its client. Depending on site policies, if the HAAA does not receive an Access-Accept or an Access-Reject message from the PPS it MAY do nothing or send an Access-Reject or an Access- Accept message back to the PPC.

3.3. Session Start Operation

The start of the session is indicated by the arrival of an Accounting-Request(Start) packet. The Accounting-Request (Start) MAY be routed to the PPS such that it can confirm the initial quota allocation.
Note that the role of the PPS is not to record accounting messages and therefore it SHOULD NOT respond with an Accounting Response packet. If the PPS does not receive the Accounting-Request(start) message it will only know that the session has started upon the first reception of a quota replenishment operation.

If the PPS does not receive indication directly (via Accounting-Request(start)) or indirectly, it SHOULD, after some configurable time, deduce that the session has not started. If the PPC supports termination capabilities, the PPS SHOULD send a Disconnect Message to the PPC as a measure to ensure that the session is indeed dead.

3.4. Mid-Session Operation

During the lifetime of a prepaid data session the PPC may request the replenishment of the quotas using an Authorize-Only Access-Request message. Once either the allocated quota has been exhausted or the threshold has been reached, the PPC MUST send an Access-Request with Service-Type(6) set to a value of "Authorize-Only" and the PPAQ attribute.

The PPC MUST also include NAS identifiers, and Session Identifier attributes in the Authorize-Only Access-Request. The Session Identifier should be the same as the one used during the initial Access-Request. For example, if the User-Name(1) attribute was used in the Access-Request it has to be included in the Authorize-Only Access-Request as well, especially if the User-Name(1) attribute is used to route the Access-Request to the Home AAA server.

The Authorize-Only Access-Request MUST NOT include a User Password and MUST NOT include a CHAP Password. In order to enable the receiver to authenticate the message, the PPC MUST include a Message-Authenticator(80). In order to satisfy the requirements of Section 5.44 of RFC 2865 [RFC2865], the PPC MUST also include the State attribute. It is anticipated that the inclusion of the State attribute will enable the PPS to map the Authorize-Only Access Request to the authentication context that was established when the PPC authenticated itself at the beginning of the session. The PPC computes the value for the Message-Authenticator and the State attributes according to RFC 2869 [RFC2869] and RFC 2865 [RFC2865] respectively.

When the HAAA receives an Authorize-Only Access-Request that contains a PPAQ, it validates the message using the Message-Authenticator(80), according to RFC 2869. If the HAAA receives an Authorize-Only Access-Request that contains a PPAQ and either no or an invalid Message-Authenticator(80) it SHALL silently discard the message. An Authorize Only Access-Request message that does not contain a PPAQ is either erroneous or belongs to another application (for example, a Change of Authorization message [RFC3576]). In this case the Authorize-Only Access-Request is either silently discarded or handled by another application.

Once the Authorize-Only Access-Request message is validated, the HAAA SHALL forward the Authorize-Only Access-Request to the appropriate PPS. The HAAA MUST forward the Authorize-Only Access-Request to the PPS specified in the PPAQ. The HAAA MUST add a Message-Authenticator(80) to the message, according to RFC 2869. As with the Access-Request message, the HAAA MAY modify the User-Name(1) attribute such that it identifies the user to the PPS.

When the PPS receives the Authorize-Only Access-Request containing a PPAQ attribute, it MUST validate the Message-Authenticator(80) as described in RFC 2869. If validation fails, the PPS MUST silently discard the message. If it receives an Authorize-Only Access-Request message that does not contain a PPAQ, it MUST silently discard the message.

The PPS locates the prepaid session state and uses the QID contained within the PPAQ to detect replays. The PPS takes the most recently allocated quota and subtracts it from the user balance. If sufficient balance remains, the PPS authorizes the PPS and allocates additional quota. The PPS may also calculate a new threshold value. Upon successful re-authorization, the PPS generates an Access-Accept containing the PPAQ attribute.

Depending on site policies, upon unsuccessful authorization, the PPS generates an Access-Reject or an Access-Accept with Filter-Id(11) or Ascend-Data-Filter attribute (if supported) and the Session-Timeout(27) attribute such that the subscriber can get access to a restricted set of locations for a short period of time. This feature could be used to enable users to replenish their accounts, create new accounts, or to access free content.

Upon receiving an Access-Accept from the PPS, the HAAA forwards the message to its client. If the HAAA receives an Access-Reject message, it forwards the message. Depending on site policies, if the HAAA does not receive an Access-Accept or an Access-Reject message from the PPS it MAY do nothing or it MAY send an Access-Reject message back to its client.

Upon receiving an Access-Accept, the PPC updates its quotas and threshold parameters with the values contained in the PPAQ attribute. Note that the PPS MAY update the PrePaidServer attribute(s) and these may have to be saved as well. If the Access-Accept message contains a Filter-Id(11), an Ascend-Data-Filter attribute, or Session Timeout(27), the PPC SHALL restrict the subscriber session accordingly.

3.5. Dynamic Operations

The PPS may take advantage of the dynamic capabilities that are supported by the PPC as advertised in the Session Termination and the PPAC attribute during the initial Access-Request. There are two types of actions that the PPS may perform. Firstly, it may request the session to be terminated. Secondly, it may request the attributes associated with the session to be modified. More specifically, it may modify a previously sent PPAQ.

Both of these actions require that the session be uniquely identified at the PPC as described in [RFC3576].

3.5.1. Unsolicited Session Termination Operation

At anytime during a session the PPS may send a Disconnect Message in order to terminate a session, see in [RFC3576]. Upon successful

termination of a session the PPC MUST return any unused quota to the PPS by issuing an Authorize-Only Access-Request containing the PPAQ which contains any unused quota and the Update-Reason set to "Remote Forced Disconnection".

3.5.2. Unsolicited Change of Authorization Operation

At any time during the session the PPC may receive a Change of Authorization (CoA) message. A PPS may send a new quota to either add or to remove quota that is allocated to the service. If the Change of Authorization contains a PPAQ then that PPAQ overrides a previously received PPAQ. The PPS MUST NOT change the units used in the PPAQ.
If the newly received PPAQ reduces the amount of allocated quota beyond what is already used then the PPC accepts the new PPAQ and act as it normally would when the quota is used up. For example, if the threshold is reached then is request a quota update.

3.6. Termination Operation

The termination phase is initiated when (i) the subscriber logs off, (ii) the subscriber balance is exhausted, or (iii) when the PPC receives a Disconnect Message.
In case the user logged off, or the PPC receives a Disconnect Message, the PPC sends an Authorize-Only Access-Request message with a PPAQ and Update-Reason attribute set to either "Client Service Termination" or "Remote Forced Disconnect". This message indicates the amount of consumed quota.
In case the currently allocated quota is exhausted, if the PPAQ contained the Termination-Action subytype, the PPC follows the specified action.

3.7. Mobile IP Operations

In roaming scenarios with Mobile-IP, the prepaid data session should be maintained transparently if the HA is acting as the access device hosting the PPC. As the subscriber device associates with a new access service device (AP or PDSN that supports PPC capability), this service access device sends a RADIUS Access-Request and the subscriber is re-authenticated and reauthorized. The service access device SHALL include the PPAC attribute in the RADIUS Access-Request. In this manner, the procedure follows the Authentication and Authorization procedure described earlier.
If the HA was acting as the service access device before handoff, then the prepaid session does not undergo any change after the handoff because the Mobile IP session is anchored at the HA and the user's Home IP address does not change.
In the case of a wireless access point or PDSN acting as the service access device, it is likely that the user's (care-of) IP address will change. The prepaid session will be affected by this. In this scenario

the service access device shall send an Access-Request message which is routed to the home network and SHALL reach the PPS that is serving this session. The PPS correlates the new authorization request with the existing active session and assigns a quota to the new request. Any outstanding quota at the old service access device SHALL be returned to the PPS if the Mobile-IP nodes (HA and FA) support registration revocation (Mobile IPv4 only). Specifically, the quota SHOULD be returned when the service access device sends the Authorize-Only Access-Request with PPAQ Update-Reason set to either "Remote Forced Disconnect" or "Client Service Termination". In order to trigger the sending of this last Authorize-Only Access- Request, the PPS may issue a Disconnect Message [3576] to the service access device.
Even if the subscriber moves to a service access device that does not have prepaid capabilities can the prepaid data service continue. This can be done by requesting the Home Agent (assuming it has such capabilities) to take over the responsibilities of the service access device (i.e. metering). This scenario will be discussed in detail in a later version of this document.

3.8. Operation Considerations for Multiple Services

This section describes the support for multiple prepaid services on a single PPC. Message flows illustrating the various interactions are presented in Appendix Appendix A.
A PPC that supports prepaid operations for multi-services SHOULD set the "Multi-Services Supported" bit in the PPAC. When working with multi-services, we need to differentiate between the services. A Service-Id attribute is used in the PPAQ in order to uniquely differentiate between the services. The exact definition of the Service-Id attribute is outside the scope of this document.
A PPAQ that contains a Service-Id is associated with that service. A PPAQ that contains a Rating-Group-Id is associated with that Rating-Group. A PPAQ MUST NOT contain both a Rating-Group-Id and a Service-Id. A PPAQ that contains neither a Rating-Group-Id nor a Service-Id then the default service is used, i.e., the "Access Service".

3.8.1. Initial Quota Request

When operations with multiple services is desired then the PPC requests the initial quota by sending a PPAQ containing the Service-Id in an Authorize-Only Access-Request packet for that service. Similarly, if the PPC supports rating groups then it may request a quota for the rating group by sending a PPAQ containing the Rating-Group-Id. In both cases the Update-Reason is set to "Initial-Request". The Authorize-Only Access-Request message MAY contain more than one PPAQ.
Upon receiving an Authorize-Only Access-Accept message containing one or more PPAQs, the PPS allocates resources to each PPAQ. Each PPAQ is assigned a unique QID that MUST appear in subsequent PPAQ updates for that service or rating group. Additionally, the PPAQ MUST contain the

Service-ID or Rating-Group-Id, unless the PPAQ is the generic "Access Service".

3.8.2. Quota Update

Once the services start to utilize their allotted quota they will eventually need to replenish their quotas (either the threshold is reached or no more quota remains). In order to replenish the quota, the PPC sends an Authorize-Only Access-Request message containing one or more PPAQs. Each PPAQ MUST contain the appropriate QID, Service-ID or Rating-Group-Id (or neither the Service-ID or Rating-Group-Id if the quota replenishment is for the "Access Service"). The Update-Reason filed indicates either "Threshold reached"(3), or "Quota reached"(4). Upon receiving an Authorize-Only Access-Request packet with one or more PPAQs the PPS responds with a new PPAQ for that service. The PPAQ contains a new QID, the Service-Id or the Rating-Group-Id, and a new QID. If the PPS does not grant additional quota for the service it MUST include the Termination-Action subfield in the PPAQ that will instruct the PPC to take appropriate actions.

3.8.3. Termination

When the allotted quota for a service is exhausted, the PPC shall act in accordance with the flags set in the Termination-Action subtype. If the Termination-Action subtype is absent then the service MUST be terminated. If the service is to be terminated, then the PPC shall send a PPAQ with the appropriate QID, the Service-Id, the used quota, and the Update-Reason set to "Client Service Termination".
If the "Access Service" has terminated, then all other services must be terminated as well. In this case the PPC MUST report on all issued quotas for the various services. The Update-Reason field should be set to "Access Service Terminated".

3.8.4. Dynamic Operations

Dynamic operations for multi-services are similar to dynamic operations described for single service operations. The PPS MAY send a COA message containing a PPAQ for an existing service instance. The PPC matches the PPAQ with the service using the Service-ID or the Rating-Group-Id attribute. The new quota could differ from the previously allocated value.
A disconnect message terminates the "Access Service". As such the PPC MUST report all unused quotas by sending an Authorize-Only Access Request message containing a PPAQ for each active service. The Update-Reason MAY indicate that the reason for the update.

3.8.5. Support for Resource Pools

If the PPC supports pools as indicated by setting the "Pools supported" bit in the PPAC then the PPS may associate a quota with a Pool by

including the Pool-Id and the Pool-Multiplier in the PPAQ. When Resource Pools are used, the PPAQ MUST NOT use the threshold field.

3.8.6. One-time Charging

To initiate one-time charging the PPC includes the PPAQ attribute in an Access-Request packet. The Service-ID field of the PPAQ identifies the prepaid service. The amount to be charged is specified using the Resource Quota and Resource Quota overflow subtypes. If the value specified is negative then the resources are credited to the user account. This functionality corresponds to refunding.
The QID subtype MUST be set to a unique value and is used by the PPS to detect duplicates. The Update Reason field MUST be set to One-Time Charging. Upon receiving a One-Time charge PPAQ, the RADIUS server authenticates the user and, if successful, passes the PPAQ to the PPS. The PPS locates the account and debits or credits it accordingly. The PPS MUST respond to the PPS with an Access-Accept message if successful, or an Access-Reject message otherwise.
In case of a successful operation the HAAA forwards the message to the PPC with an Access Accept message. Since this is a one-time charge the PPC MUST NOT allow the session to continue. Therefore, the RADIUS server SHOULD include in the Access-Accept a Session-Timeout set to 0. Upon receiving an Access-Accept response the PPC SHOULD generate an Accounting Stop message.
A PPAQ used for One-Time charging MAY appear in an Authorize-Only Access Request. This is the case when the session already exists. The PPS responds with an Access-Accept to indicate that the user account has been debited or an Access-Reject otherwise.

3.8.7. Error Handling

If the PPS receives a PPAQ with an invalid QID it MUST ignore that PPAQ.
If the PPS receives a PPAQ containing a Service-Id, or a Rating-Group-Id that it does not recognize, then it MUST ignore that PPAQ.
If the PPC receives a PPAQ containing a Service-Id, or a Rating-Group-Id that it does not recognize, then it MUST ignore that PPAQ.
If the PPC receives a PPAQ that contains a Pool-Id without a Pool-Multiplier or a Pool-Multiplier without a Pool-Id it MUST ignore that PPAQ.

3.8.8. Accounting Considerations

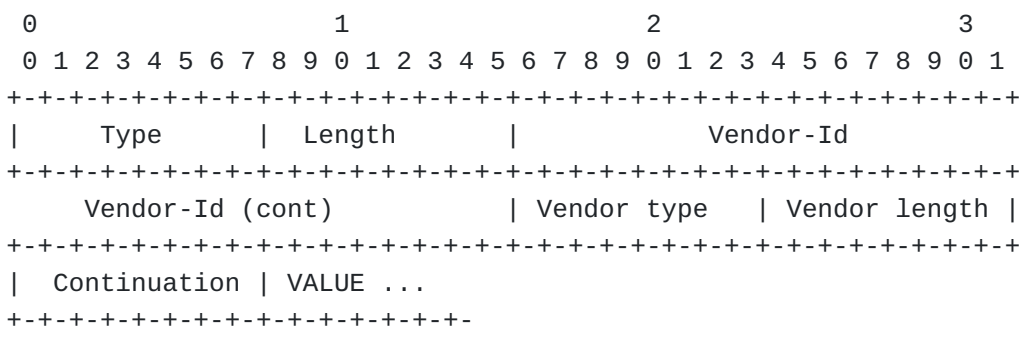
Although typically generated, accounting messages are not required to deliver a prepaid data service. When generated, accounting messages are used for auditing purposes and for billing. Accounting messages associated with prepaid data sessions should include the PPAQ attribute.

4. Attributes

This section specifies the attributes that implement the RADIUS extensions for prepaid.

4.1. PrePaid Accounting Capability (PPAC) Attribute

The PrepaidAccountingCapability (PPAC) attribute is sent in the Access-Request message by a PPC to describe its prepaid capabilities.



The fields have the following meaning and encoding:

Type:

26 for Vendor-Specific

Length:

6 + 3 + length of SubTypes

Vendor-Id:

The Vendor-Id value for WiMAX is 24757.

Vendor type:

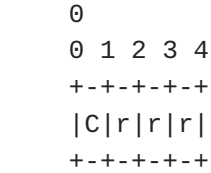
35 for PPAC

Vendor length:

3 + length of VALUE

Continuation:

The Continuation Field is defined as follows:



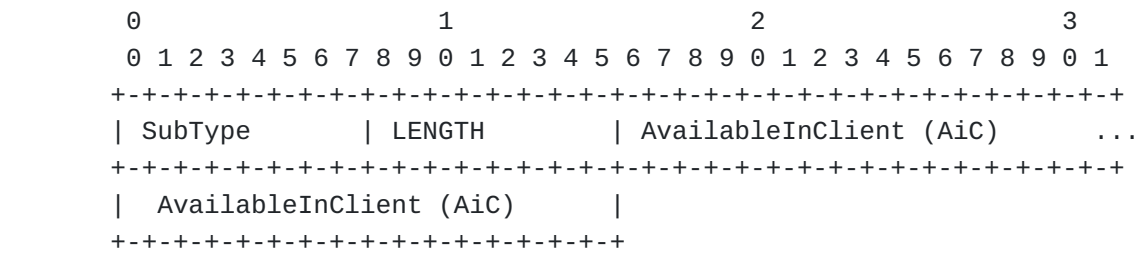
The C-bit of the continuation field indicates if a attribute is being fragmented. When the C-bit is set to one ‘1’ this indicates that the attribute is being fragmented that is the next VSA of the same type is to be appended to this attribute. When the C-bit is set to zero '0' this indicates that the next attribute is not a fragment of this attribute.

An attribute that is not being fragmented will have the C-bit set to '0'. An attribute that is being fragmented will have its C-bit set to '1' for all fragments until the last fragment, which will have its C-bit set to '0' indicating it's the last fragment of the attribute. The r-bits are reserved for future use. They SHALL be set to zero by the sender and SHALL be ignored by the receiver.

The value of the C-bit MUST be 0.

VALUE :

The content of the AvailableInClient (AiC) SubType fields are encoded using the data type String.



The fields have the following meaning and encoding:

SubType : Value (1)

LENGTH : 2 + 4

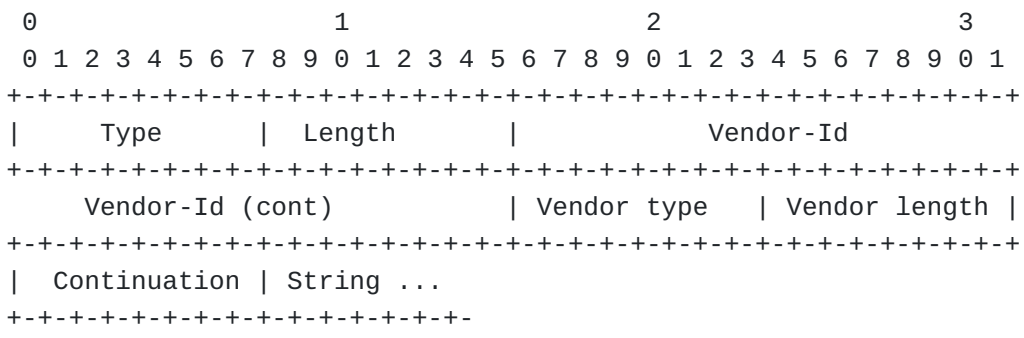
AvailableInClient (AiC): The bitmap is encoded as:

Value	Description
0x00000001	Volume metering supported
0x00000010	Duration metering supported
0x00000100	Resource metering supported
0x00001000	Pools supported
0x00010000	Rating groups supported
0x00100000	Multi-Services supported
0x01000000	Tariff Switch supported
0x10000000	Reserved

The AvailableInClient (AiC) SubType is encoding as follows:

4.2. Session Termination Capability Attribute

The Session Termination Capability attribute is included in the RADIUS Access-Request message towards the RADIUS server to indicate whether or not the NAS supports Dynamic Authorization.



The fields have the following meaning and encoding:

Type:

26 for Vendor-Specific

Length:

6 + 3 + 4

Vendor-Id:

The Vendor-Id value for WiMAX is 24757.

Vendor type:

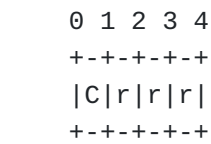
36 for Session Termination Capability

Vendor length:

3 + 4

Continuation:

The Continuation Field is defined as follows:



The C-bit of the continuation field indicates if a attribute is being fragmented. When the C-bit is set to one ‘1’ this indicates that the attribute is being fragmented that is the next VSA of the same type is to be appended to this attribute. When the C-bit is set to zero '0' this indicates that the next attribute is

not a fragment of this attribute.
An attribute that is not being fragmented will have the C-bit set to '0'. An attribute that is being fragmented will have its C-bit set to '1' for all fragments until the last fragment, which will have its C-bit set to '0' indicating it's the last fragment of the attribute. The r-bits are reserved for future use. They SHALL be set to zero by the sender and SHALL be ignored by the receiver.

The value of the C-bit MUST be 0.

String:

This field is encoded as a bitmap.

Value	Description
0x00000000	Reserved
0x00000001	Dynamic Authorization Extensions (RFC 3576) is supported.
...	All further values are reserved.

4.3. Prepaid Accounting Operation (PPAQ) Attribute

One or more PPAQ attributes are sent in an Access Request, Authorize-Only Access-Request and Access-Accept message. In an Access Request message, the PPAQ attribute is used to facilitate one-time charging transactions. In Authorize-Only Access-Request messages it is used for one-time charging, report usage and to request further quota. It is also used in order to request prepaid quota for a new service instance. In an Access-Accept message it is used in order to allocate the (initial and subsequent) quotas.
When multiple services are supported, a PPAQ is associated with a specific service as indicated by the presence of a Service-Id, a Rating-Group-Id, or the "Access Service" (as indicated by the absence of both, the Service-Id and the Rating-Group-Id).
Note: Either Volume-Quota, Time-Quota, or Resource-Quota SubTypes MUST appear in the PPAQ attribute, except for the price enquiry message exchange where these subtypes MUST be absent. A single PPAQ attribute MUST NOT contain more than one Service-Id, MUST NOT contain more than one Rating-Group-Id, and MUST NOT contain both a Service-Id and a Rating-Group-Id. A PPAQ that does not contain a Service-ID or a Rating-Group-Id refers to the "Access Service". A PPAQ MUST NOT contain more than one Pool-Id. A PPAQ that contains a Pool-Id MUST also contain a Pool-Multiplier SubType.

The PPAQ attribute, as shown in Figure 12, has a variable length (greater than 8, encoded into one octet), and consists of a variable number of subtypes. Unused subtypes are omitted from the message. The following table summarizes the presence of various SubTypes in the PPAQ attribute carried in the Access-Request and Access-Accept messages.

Request	Accept	#	SubType Name
0-1(g)	0-1(m,n)	1	Quota Identifier
0-1(a,g)	0-1(a,k,n)	2	VolumeQuota
0	0-1(a,m,n)	3	VolumeThreshold
0-1(b,g)	0-1(b,k,n)	4	DurationQuota
0	0-1(b,m,n)	5	DurationThreshold
0-1(c,g)	0-1(c,k,n)	6	ResourceQuota
0	0-1(c,m,n)	7	ResourceThreshold
0-1(d,g)	0	8	Update-Reason
0-n(e,g)	0-n(e,m,n)	9	PrepaidServer
0-1(g,h,j)	0-1(m,n)	10	Service-ID
0-1(g,h,j)	0-1(m,n)	11	Rating-Group-ID
0	0-1(m,n)	12	Termination-Action
0	0-1(m,n)	13	Pool-ID
0	0-1(f,m,n)	14	Pool-Multiplier
0-1(g)	0	15	Requested-Action
0	0-1(k,m,n)	16	Check-Balance-Result
0	0-1(n)	17	Cost-Information

None of the above-listed SubTypes appears in the Access-Reject nor in the Access-Challenge.
Notes:

*(a) SHALL be present if volume based charging is used. SHALL NOT be present otherwise. Volume- Threshold is optional.

*(b) SHALL be present if duration-based charging is used. SHALL NOT be present otherwise. Duration- Threshold is optional.

*(c) SHALL be present if resource-based charging is used. SHALL NOT be present otherwise. Resource- Threshold is optional.

*(d) SHALL be present in an Authorize-Only Access-Request.

*(e) MAY be present in an Access-Accept. If present in Access Accept it SHALL be present in Access- Request (except for the first Access-Request).

*(f) Pool-Multiplier SHALL be present when Pool-ID is present otherwise Pool-Multiplier SHALL NOT be present in the PPAQ.

*(g) If Requested-Action is present then Service-ID SHALL also be present and all other attributes SHALL NOT be present.

*(h) PPAQ SHALL NOT contain both a Service-ID and a Rating-Group-ID.

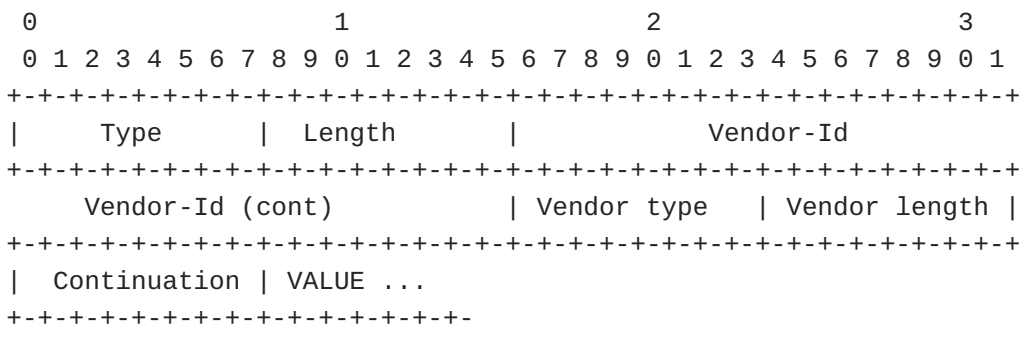
*(j) A PPAQ that does not contain a Service-ID or a Rating-Group-Id refers to the "Access Service"(ISF).

*(k) If Balance-Check-Result is present and set to 0 then either Volume-Quota, Duration-Quota or Resource- Quota SHALL be present.

*(m) If Balance-Check-Result is present then Service-ID SHALL also be present and other attributes (tagged with m) SHALL NOT be present.

*(n) The PPAQ in which a Cost-Information occurs SHALL NOT include a Quota-Identifier, because no quota is actually reserved by the PPS. The Service-ID SHALL be present with the Cost-Information for that Service-ID may not be present if the Cost-Information cannot be provided. All other attribute SHALL not appear.

In the following subsections the various subtypes of the PPAQ attribute are specified.



The fields have the following meaning and encoding:

Type:

26 for Vendor-Specific

Length:

6 + 3 + length of SubTypes

Vendor-Id:

The Vendor-Id value for WiMAX is 24757.

Vendor type:

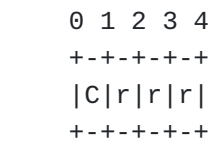
37 for PPAQ

Vendor length:

3 + length of SubTypes

Continuation:

The Continuation Field is defined as follows:



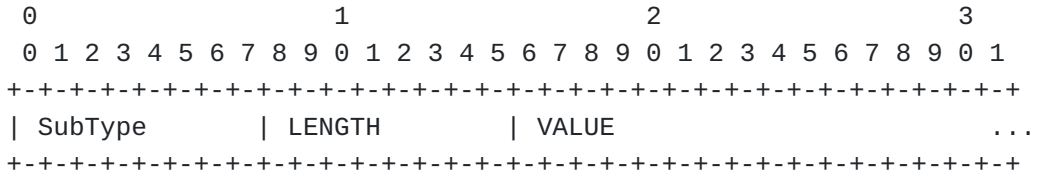
The C-bit of the continuation field indicates if a attribute is being fragmented. When the C-bit is set to one ‘1’ this indicates that the attribute is being fragmented that is the next VSA of the same type is to be appended to this attribute. When the C-bit is set to zero '0' this indicates that the next attribute is

not a fragment of this attribute.
An attribute that is not being fragmented will have the C-bit set to '0'. An attribute that is being fragmented will have its C-bit set to '1' for all fragments until the last fragment, which will have its C-bit set to '0' indicating it's the last fragment of the attribute. The r-bits are reserved for future use. They SHALL be set to zero by the sender and SHALL be ignored by the receiver.

The value of the C-bit MAY be 0 or 1.

VALUE:

Data type String
Each SubType is then encoded in the following style:



The fields have the following meaning and encoding:

SubType :

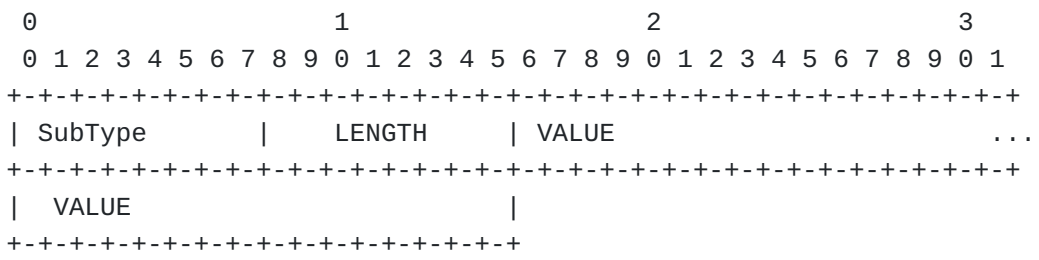
Contains an 8 bit unsigned integer.

LENGTH :

Contains an 8 bit unsigned integer.
The value of the LENGTH field is calculated as the length of the VALUE field plus two octets (one octet for the length of the SubType field and another octet for the length of the LENGTH field).

4.3.1. Quota Identifier (QID) SubType

The Quota Identifier (QID) is generated by the PPS and subsequently returned in a PPAQ->QID subtype from the PPC to the PPS. This field has the semantic of a transaction identifier and therefore changes with every transaction initiated by the PPS to the PPC.



The fields have the following meaning and encoding:

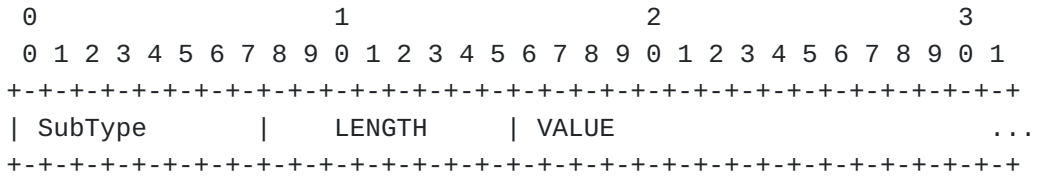
SubType : Value(1)

LENGTH : 2 + length of VALUE field.

VALUE : Data type String

4.3.2. VolumeQuota SubType

TVolumeQuota SubType is only present when volume-based charging is used. In a RADIUS Access-Accept message (PPS to PPC direction), it indicates the volume (in octets) allocated for the session by the PPS. In a RADIUS Authorize-Only Access-Request message (PPC to PPS direction), it indicates the totally used volume (in octets) for both inbound and outbound traffic. The Exponent Field, if present, MUST NOT encode a negative number or zero.



The fields have the following meaning and encoding:

SubType : Value(2)

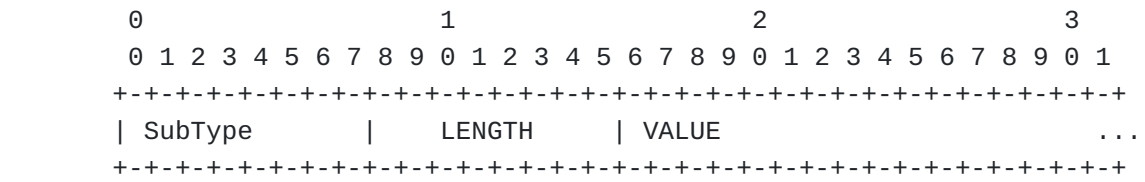
LENGTH : 2 + (8 or 12)

VALUE : Data type String

The content of the VALUE field either contains the Value-Digits Field (8 octets long) or the Value-Digits Field plus the Exponent Field (12 octets long).

4.3.3. VolumeThreshold SubType

The value of the type field of the VolumeThreshold SubType is TBD and its length is 10 or 14 octets. This SubType is optionally present if the VolumeQuota is present in a RADIUS Access-Accept message (PPS to PPC direction). It is generated by the PPS and indicates the volume (in octets) that has to be consumed before a new quota is requested. This threshold MUST NOT be larger than the VolumeQuota. The Exponent Field, if present, MUST NOT encode a negative number or zero.



The fields have the following meaning and encoding:

SubType : Value(3)

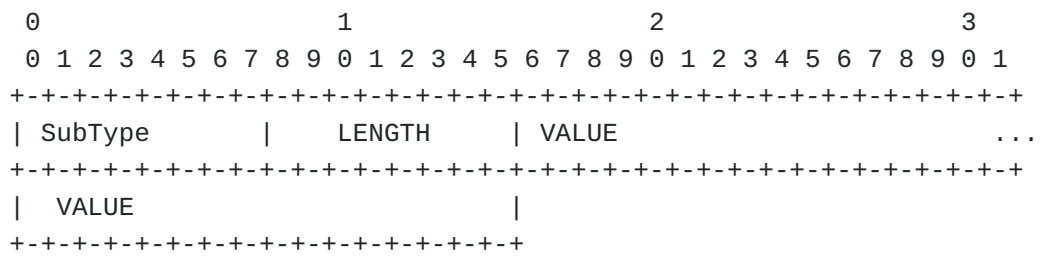
LENGTH : 2 + (8 or 12)

VALUE : Data type String

The content of the VALUE field either contains the Value-Digits Field (8 octets long) or the Value-Digits SubType plus the Exponent Field 12 octets long).

4.3.4. DurationQuota SubType

The optional DurationQuota SubType is only present if duration-based charging is used. In a RADIUS Access-Accept message (PPS to PPC direction), it indicates the duration (in seconds) allocated for the session by the PPS. In a RADIUS Access-Request message (PPC to PPS direction), it indicates the total duration (in seconds) since the start of the accounting session related to the QID subtype of the PPAQ attribute in which it occurs.



The fields have the following meaning and encoding:

SubType : Value(4)

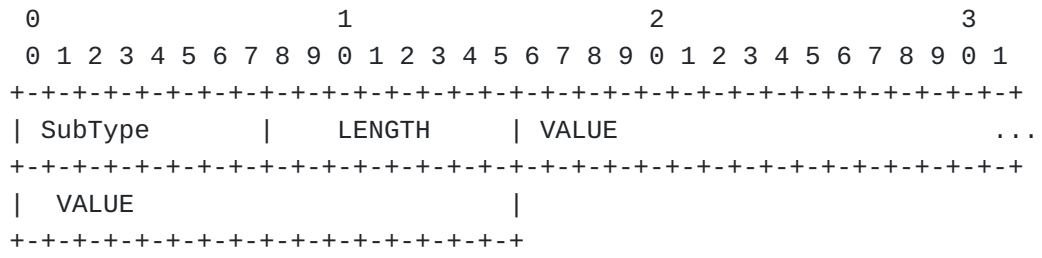
LENGTH : 2 + 4

VALUE : Data type string.

The content of this field contains a 32-bit unsigned integer (with the values 0 to +4,294,967,295).

4.3.5. DurationThreshold SubType

The DurationThreshold SubType is optionally present if the DurationQuota is present in a RADIUS Access-Accept message (PPS to PPC direction). It represents the duration (in seconds) after which new quota should be requested. This threshold MUST NOT be larger than the DurationQuota SubType.



The fields have the following meaning and encoding:

SubType : Value(5)

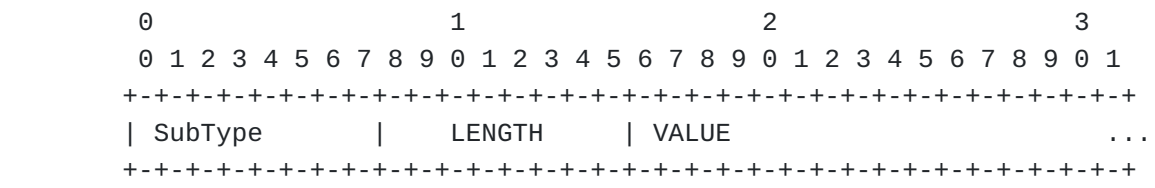
LENGTH : 2 + 4

VALUE : Data type string.

The content of this field contains a 32-bit unsigned integer (with the values 0 to +4,294,967,295).

4.3.6. ResourceQuota SubType

The optional ResourceQuota SubType is only present if resource-based or one-time charging is used. In the RADIUS Access-Accept message (PPS to PPC direction) it indicates the resources allocated for the session by the PPS. In RADIUS Authorize-Only Access-Request message (PPC to PPS direction), it indicates the resources used in total, including both incoming and outgoing chargeable traffic. In one-time charging scenarios, the subtype represents the number of units to charge the user. The attribute consists of a Value-Digits Field and optionally an Exponent Field (as indicated by the length field).



The fields have the following meaning and encoding:

SubType : Value(6)

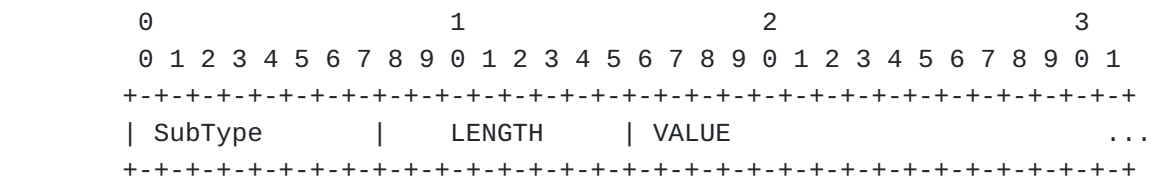
LENGTH : 2 + (8 or 12)

VALUE : Data type String

The content of the VALUE field either contains the Value-Digits Field (8 octets long) or the Value-Digits Field plus the Exponent Field (12 octets long).

4.3.7. ResourceThreshold SubType

The semantic of the ResourceThreshold SubType follow those of the VolumeThreshold and DurationThreshold SubType.



The fields have the following meaning and encoding:

SubType : Value(7)

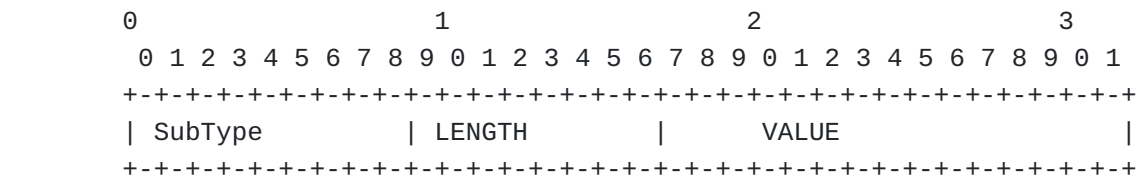
LENGTH : 2 + (8 or 12)

VALUE : Data type String

The content of the VALUE field either contains the Value-Digits Field (8 octets long) or the Value-Digits Field plus the Exponent Field (12 octets long).

4.3.8. Update-Reason SubType

The Update-Reason SubType is present in a RADIUS Access-Request message (PPC to PPS direction) and indicates the reason for initiating the quota update operation. Update reasons (6), (7), (8) and (9) indicate that the associated resources are released at the PPC side, and that therefore the PPS MUST NOT allocate a new quota in the RADIUS Access-Accept message.



The fields have the following meaning and encoding:

SubType : Value(8)

LENGTH : 2 + 1

VALUE : Data type string

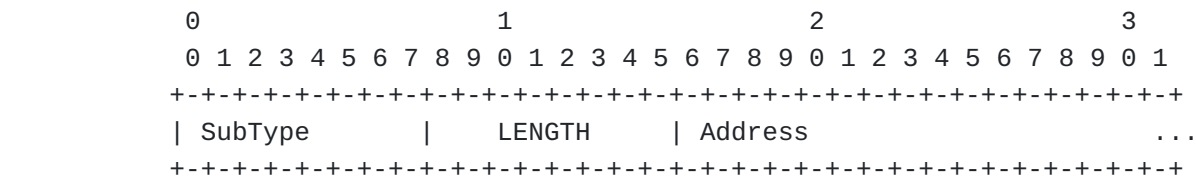
This field contains a byte (with the values 0 to 255).

The following values for the Update-Reason SubType are defined:

Value	Description
0	Reserved
1	Pre-initialization
2	Initial Request
3	Threshold Reached
4	Quota Reached
5	TITSU Approaching
6	Remote Forced Disconnect
7	Client Service Termination
8	"Access Service" Terminated
9	Service not established
10	One-Time Charging
11...255	**Available for IANA registration**

4.3.9. PrepaidServer SubType

The optional PrepaidServer SubType indicates the address of the serving PPS. If present, the Home RADIUS server uses this address to route the message to the serving PPS. Multiple instances of this subtype MAY be present in a single PPAQ attribute.
If present in the PrepaidServer SubType of an incoming RADIUS Access-Accept message, the PPC returns this SubType back without modifying it in the subsequent RADIUS Access-Request message. If multiple values are present, the PPC MUST NOT change their order.



The fields have the following meaning and encoding:

SubType : Value(9)

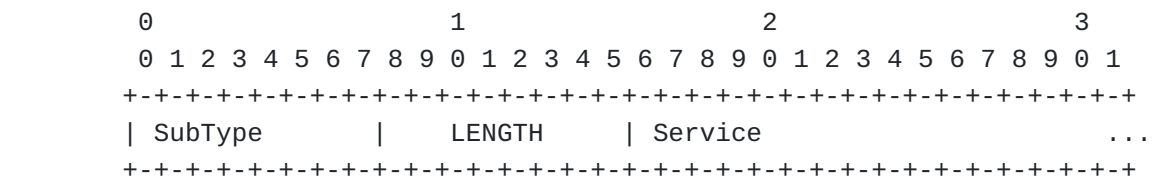
LENGTH : 2 + (4 or 16)

Address : Data type String

For IPv4, the Address field is 4 octets based on the encoding of the NAS-IP-Address defined in RFC 2138. For IPv6, the Address field is 16 octets long.

4.3.10. Service-ID SubType

The Service-ID SubType is handled as an opaque string that uniquely describes the service instance for which prepaid metering should be applied.
A Service-Id could be an IP 5-tuple (source address, source port, destination address, destination port, protocol). If a Service-ID SubType is present in the PPAQ, the entire PPAQ attribute refers to that service. If a PPAQ attribute does not contain a Service-Id or Rating-Group-ID, then the PPAQ attribute refers to the "Access Service".



The fields have the following meaning and encoding:

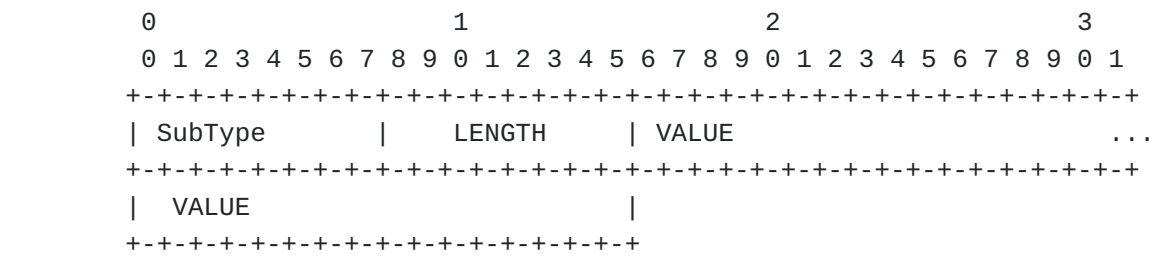
SubType : Value(10)

LENGTH : 2 + length (Service)

Service : Data type String

4.3.11. Rating-Group-ID SubType

The Rating-Group-ID SubType indicates that this PPAQ attribute is associated with resources allocated to a Rating Group with the corresponding ID. This SubType is encoded as a string. A single PPAQ MUST NOT contain more than one Rating-Group-ID.



The fields have the following meaning and encoding:

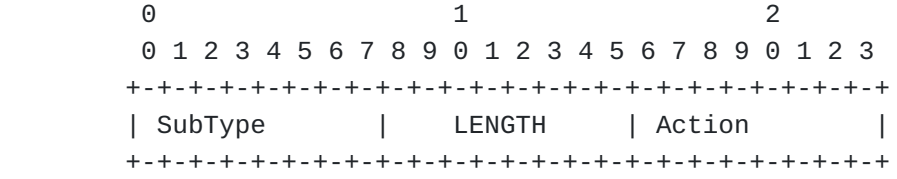
SubType : Value(11)

LENGTH : 2 + 4

VALUE : Data type string with a length of 4 octets

4.3.12. Termination-Action SubType

The value of the type field of the Termination-Action SubType is TBD. The length of this SubType is 3 octets. This SubType contains an enumeration of the action to take when the PPS does not grant additional quota. Valid actions are as follows.



The fields have the following meaning and encoding:

SubType : Value(12)

LENGTH : 2 + 1

Action : Data type string.

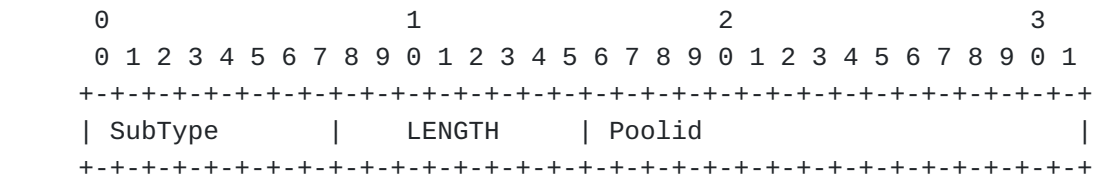
The content is a byte (with the values 0 to +255).

The following values for the Termination-Action SubType are defined:

Value	Description
0	Reserved
1	Terminate
2	Request More Quota
3	Redirect/Filter
4..255	**Available for IANA registration**

4.3.13. Pool-ID SubType

The Pool-ID SubType identifies the resource pool. It is encoded as a string.



The fields have the following meaning and encoding:

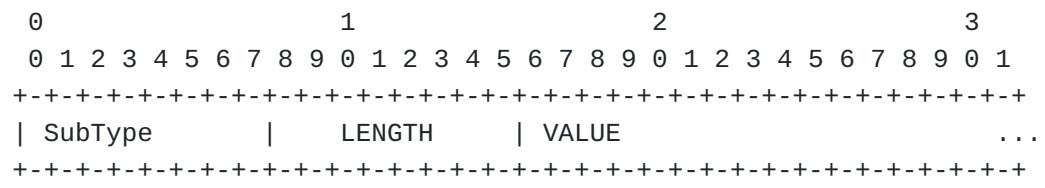
SubType : Value(13)

LENGTH : 2 + 4

Poolid : Data type string with length of 4 octets.

4.3.14. Pool-Multiplier SubType

The Pool-Multiplier SubType determines the weight of resources when they are inserted into the pool that is identified by the accompanying Pool-ID SubType, and the rate at which resources are taken out of the pool by the relevant Service or Rating-Group.



The fields have the following meaning and encoding:

SubType : Value(14)

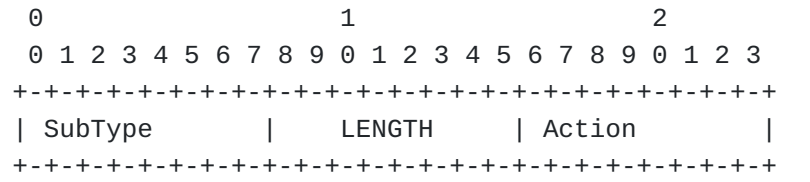
LENGTH : 2 + (8 or 12)

VALUE : Data type String

The content of the VALUE field either contains the Value-Digits Field (8 octets long) or the Value-Digits Field plus the Exponent Field (12 octets long).

4.3.15. Requested-Action SubType

The Requested-Action SubType is only be present in messages sent from the PPC to the PPS. It indicates that the PPC desires the PPS to perform the indicated action and to return the result. The PPAQ in which a Requested-Action SubType occurs MUST NOT contain a QID, and MUST contain a Service-Identifier or a Rating-Group-Identifer that allows the PPS to uniquely identify the service for which the indicated action is requested.



The fields have the following meaning and encoding:

SubType : Value(15)

LENGTH : 2 + 1

Action : Data type string.

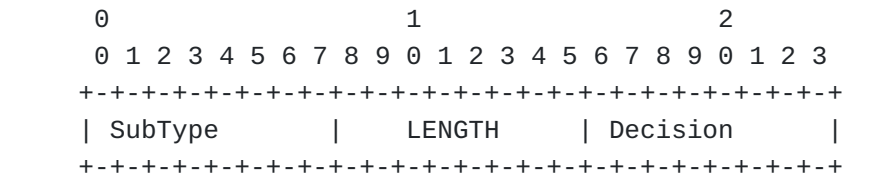
The content is a byte (with the values 0 to +255).
The values are listed below.

The following values for the Action field of the Requested-Action SubType are defined:

Value	Description
0	Reserved
1	Balance Check
2	Price Enquiry
3..255	**Available for IANA registration**

4.3.16. Check-Balance-Result SubType

The Check-Balance-Result SubType can only be present in messages sent from the PPS to the PPC. It indicates the balance check decision of the PPS about a previously received Balance Check Request (as indicated in a Requested-Action SubType). The PPAQ attribute in which a Check-Balance-Result occurs MUST NOT include a QID beause no quota is reserved by the PPS.



The fields have the following meaning and encoding:

SubType : Value(16)

LENGTH : 2 + 1

Decision : Data type string.

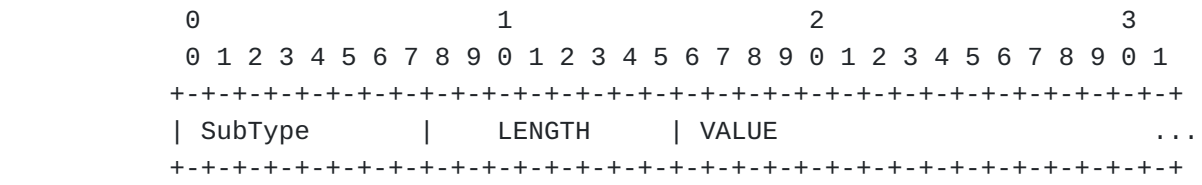
The content is a byte (with the values 0 to +255).
The values are listed below.

The following values for the Decision field of the Check-Balance-Result SubType are defined:

Value	Description
0	Success; Sufficient funds available in the user's prepaid account
1	Failure; Insufficient funds available
2..255	**Available for IANA registration**

4.3.17. Cost-Information SubType

The Cost-Information SubType is used in order to return the cost information of a service, which the PPC can transfer transparently to the end user. This SubType is sent from the PPS to the PPC as a response to a "Price Enquiry", as indicated by the Requested-Action SubType.



The fields have the following meaning and encoding:

SubType : Value(17)

LENGTH : 2 + 12 + 4 + length of the Cost-Unit Field

VALUE : Data type String

The content of the VALUE field contains (in this order) the Value-Digits Field, Exponent Field, Currency-Code Field and the Cost-Unit Field.

For example, the cost of 7.75 Malawi kwacha per hour would be encoded as follows. Value-Digits = 775, Exponent = -2, Currency Code = 454, and Cost-Unit = "hour".
The PPAQ that carries a Cost-Information MUST NOT include a QID.
The Currency-Code Field is of type Unsigned32 and used inside the Check-Balance-Result SubType and contains a currency code that specifies in which currency the values of AVPs containing monetary units were given. It is specified by using the numeric values defined in the ISO 3166-1 standard.
The Cost-Unit Field is used inside the Check-Balance-Result SubType and contains a human readable UTF8 encoded string that can be displayed to the end user. It specifies the applicable unit to the Cost-Information when the service cost is a cost per unit (e.g., cost of the service is

$1 per minute). The Cost-Unit can, for example, be minutes, hours, days, kilobytes, megabytes, etc.

4.4. Fields

4.4.1. Value-Digits Field

The Value-Digits Field is an Unsigned64 value (with a length of 8 octets) that contains the significant digits of the number. If decimal values are needed to present the number, the scaling MUST be indicated with a related Exponent Field, see Section 4.4.2.
For example, the decimal number 0.05 is encoded by a Value-Digits Field set to 5, and a scaling that is indicated with the Exponent Field set to -2.
The encoding of this SubType is not done in an TLV format but rather the encoded value is added to existing subtypes.

4.4.2. Exponent Field

The Exponent field is a Integer32 value that contains the exponent value that is to be applied to the accompanying Value-Digit Field.

4.5. Prepaid Tariff Switching (PTS) Attribute

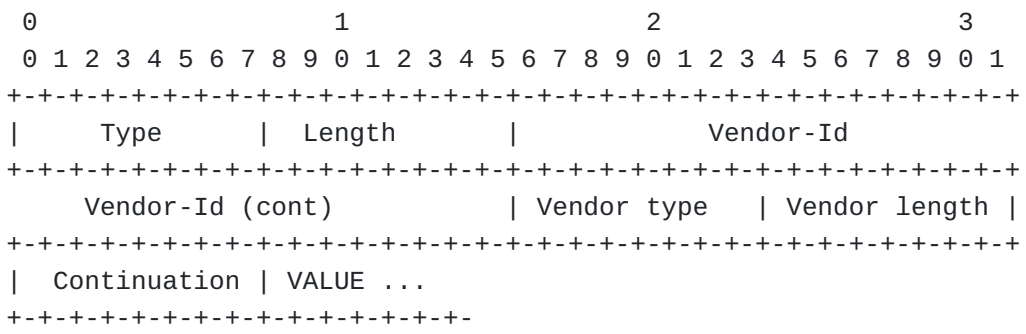
This specification defines the PTS attribute, which allows to switch from one rate to another during service provision. Support for tariff switching is optional to implement and to use for the PPC and the PPS. PPCs use the flag "Tariff Switching supported" in the AvailableInClient field of the PPAC attribute in order to indicate support for tariff switching. PPSs employ the PTS attribute in order to announce their support for tariff switching.
If a RADIUS message contains a PTS attribute, it MUST also contain at least one PPAQ attribute. If a RADIUS Access-Request message contains a PTS attribute or the "Tariff Switching supported" flag in the AvailableInClient field of the PPAC attribute, it MUST also contain an Event-Timestamp RADIUS attribute (see [RFC2869]).
Every PTS attribute MUST include a QID SubType, as specified in Section 4.3.1. In a RADIUS Access-Request message sent from the PPC to the PPS, the QID SubType MUST contain the value of the Quota Identifier SubType that was previously received from the PPS and MUST be the same as the value carried in the QID SubType of one of the PPAQ attributes included the same RADIUS message.
If multiple services are supported and if the PPAQ is associated with a service as indicated by the Service-ID SubType, then the PTS refers to the tariff switch for that service. If the PPAQ does not have a Service-ID, then the PTS refers to tariff switch for the "Access Service".
A PPAQ attribute that is transported along with a PTS attribute and has the same value as the QID SubType contained in the PTS attribute in its

own QID SubType is referred to as the "accompanying PPAQ attribute". If a PPS receives an Access-Request message from a PPC, it associates a unique value for the QID SubType to this request.
The following table summarizes the presence of various SubTypes in the PTS attribute carried in the Access-Request and Access-Accept messages.

Request	Accept	#	SubType Name
1	1	1	Quota Identifier
1	0	2	VolumeUsedAfterTariffSwitch
0	0-1	3	TarrifSwitchInterval
0	0-1(a)	4	TimeIntervalAfterTarriffSwitchUpdate

None of the above-listed SubTypes appears in the Access-Reject nor in the Access-Challenge.
Notes:

*(a) The PPS SHALL include this AVP if there is another tariff switch period after the period that ends as indicated by the TSI attribute.



The fields have the following meaning and encoding:

Type:

26 for Vendor-Specific

Length:

6 + 3 + length of SubTypes

Vendor-Id:

The Vendor-Id value for WiMAX is 24757.

Vendor type:

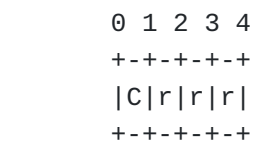
38 for PTS

Vendor length:

3 + length of SubTypes

Continuation:

The Continuation Field is defined as follows:



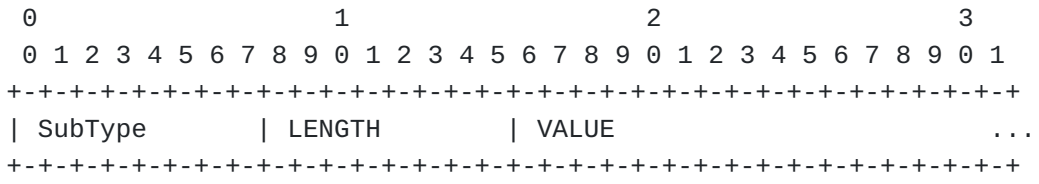
The C-bit of the continuation field indicates if a attribute is being fragmented. When the C-bit is set to one ‘1’ this indicates that the attribute is being fragmented that is the next VSA of the same type is to be appended to this attribute. When the C-bit is set to zero '0' this indicates that the next attribute is

not a fragment of this attribute.
An attribute that is not being fragmented will
have the C-bit set to '0'. An attribute that is
being fragmented will have its C-bit set to '1'
for all fragments until the last fragment, which
will have its C-bit set to '0' indicating it's
the last fragment of the attribute. The r-bits
are reserved for future use. They SHALL be set
to zero by the sender and SHALL be ignored by
the receiver.

The value of the C-bit MAY be 0 or 1.

VALUE : Variable length content of data type String containing
one or multiple SubTypes.

Each SubType is then encoding in the following style:



The fields have the following meaning and encoding:

SubType:

Contains an 8 bit unsigned integer.

LENGTH:

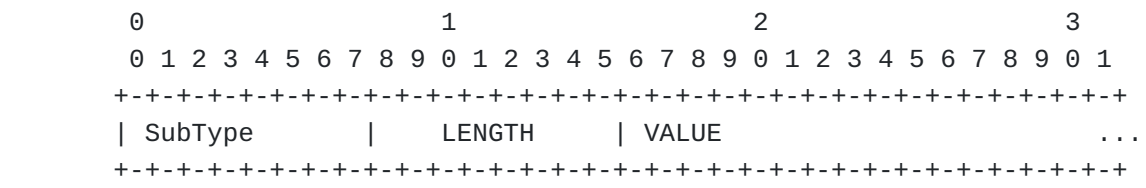
Contains an 8 bit unsigned integer.

VALUE:

Contains the content of the SubType.

4.5.1. VolumeUsedAfterTariffSwitch SubType

The optional VolumeUsedAfterTariffSwitch (VUATS) SubType is used in the RADIUS Access-Request messages (PPC to PPS direction). It indicates the volume (in octets) used during a session after the last tariff switch for the service specified via the QID SubType and the accompanying PPAQ attribute.



The fields have the following meaning and encoding:

SubType : Value(23)

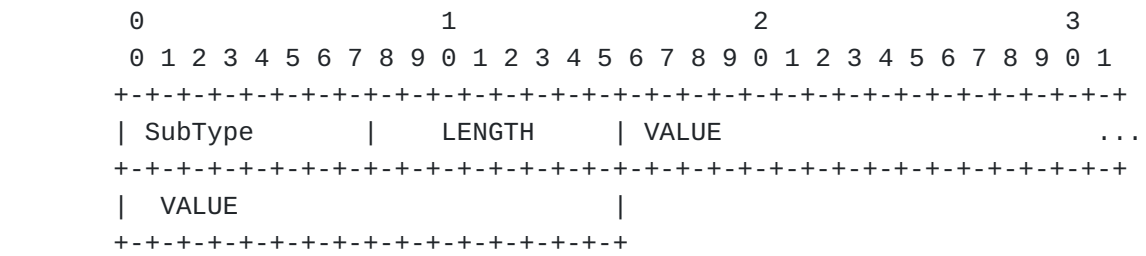
LENGTH : variable (8 or 14)

VALUE : Data type String

The content of the VALUE field either contains the Value-Digits Field or the Value-Digits Field plus the Exponent Field. The length field indicates whether one or both subtypes are included.

4.5.2. TariffSwitchInterval SubType

The TariffSwitchInterval (TSI) SubType MUST be present in each PTS attribute that is part of a RADIUS Access-Accept message (PPS to PPC direction). It indicates the interval (in seconds) between the value of Event-Timestamp RADIUS attribute (see [RFC2869]) of the corresponding RADIUS Access-Request message and the next tariff switch condition.



The fields have the following meaning and encoding:

SubType : Value(24)

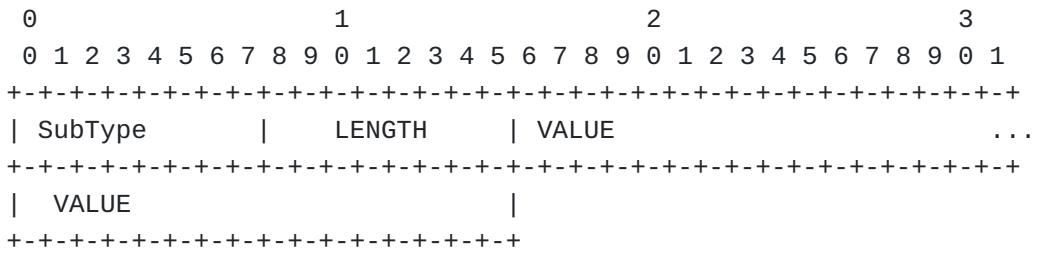
LENGTH : 6

VALUE : Data type string

This field contains a 16-bit unsigned integer (with the values 0 to +65,535).

4.5.3. TimeIntervalafterTariffSwitchUpdate SubType

The PPS MUST include TimeIntervalafterTariffSwitchUpdate (TITSU) SubType if there is another tariff switch period after the period that ends as indicated by the TSI SubType. The value of the TITSU SubType contains the number of seconds of the tariff period that begins immediately after the period that ends as indicated by the TSI attribute. If the TITSU SubType is not present, the PPC assumes that the tariff period, which ends as indicated by the TSI SubType, lasts until further notice. If TITSU is specified, the PPC MUST send a quota update before the point in time specified by the TITSU SubType (see Figure 7).



The fields have the following meaning and encoding:

SubType : Value(25)

LENGTH : 6

VALUE : Data type string

This field contains a 16-bit unsigned integer
(with the values 0 to +65,535).

5. Diameter RADIUS Interoperability

The RADIUS Prepaid extension described in this document may need to interoperate with the Diameter Credit Control Application. Two interoperability scenarios exist, as follows. Either the AAA server is Diameter based and the AAA client is RADIUS based, or the AAA client is Diameter based and the AAA server is RADIUS based.
The Diameter Credit Control Application [RFC4006] describes how to implement a prepaid accounting system using a Diameter based infrastructure.
A possible procedure for accomplishing the translation of the functionality defined in Diameter Credit Control and in the RADIUS Prepaid specification is described in Appendix Appendix B.

6. Security Considerations

This specification describes the use of RADIUS for purposes of real-time accounting. Threats and security issues for this application are described in [RFC3579] and [RFC3580]; security issues encountered in roaming are described in [RFC2607].
This document specifies new attributes that can be included in existing RADIUS packets, which are protected as described in [RFC3579] and [RFC3576]. See those documents for a more detailed description.
The security mechanisms supported in RADIUS are focused on preventing an attacker from spoofing packets or modifying packets in transit. They do not prevent an authorized RADIUS/Diameter server or proxy from modifying, inserting, or removing attributes with malicious intent. When the attributes defined in this document are modified or removed by a RADIUS proxy they may lead to incorrect accounting records being delivered to users or wrong resource consumption being collected.
The described mechanism add the mechanism of capability negotiation, so that a RADIUS server can automatically discover whether a NAS supports the real-time accounting features described in this document (and even more detailed capabilities can be learned by the RADIUS server). These mechanisms are being added to ensure that neither the NAS nor the RADIUS server make incorrect assumptions about the capabilities of the other party; potentially leading to incorrect accounting and improper access to the network or other services.

7. Table of Attributes

The following table provides a guide which attributes may be found in which RADIUS messages, and in what quantity.

Request	Accept	Reject	Challenge	Accounting Request	#	Attribute
0-1	0	0	0	0	35	PPAC
0-1	0	0	0	0	36	Session Termination Capability
0+	0+	0	0	0+	37	PPAQ
0+	0+	0	0	0+	38	PTS

8. IANA Considerations

This document contains a number of instructions to IANA.

8.1. RADIUS Attributes

This document does not require IANA to register the following four RADIUS attributes as the code registered by the Wimax Forum is re-used. The Wimax Forum SMI Network Management Private Enterprise Code is 24757.

Attribute Name	Attribute Type Value
Prepaid-Accounting-Capability (PPAC)	35
Session Termination Capability	36
Prepaid-Accounting-Operation (PPAQ)	37
Prepaid Tariff Switching (PTS)	38

8.2. New Registry: PPAC SubTypes

Section 4.1 defines the SubTypes used within the PPAC attribute. IANA is asked to create a registry for these SubTypes. Each registry entry consists of a 8 bit number together with a description of the PPAC SubType. This document creates the following PPAC SubTypes for this registry:

Value	SubType Name
0	Reserved
1	AvailableInClient
2..255	**Available for IANA registration**

Section 4.1.
Following the policies outline in [RFC3575] the available SubTypes (i.e., value 0 and values 2-255) with a description of their semantic will be assigned after the expert review process. Updates can be provided based on expert approval only. Based on expert approval it is possible to mark entries as "deprecated". A designated expert will be appointed by the IESG.
Each registration must include a number for the SubType and the semantic of the SubType.

8.3. New Registry: PPAQ SubTypes

Section 4.3 defines the SubTypes used within the PPAQ attribute. IANA is asked to create a registry for these SubTypes. Each registry entry consists of a 8 bit number together with a description of the PPAQ SubType. This document creates the following PPAQ SubTypes for this registry:

Value	SubType Name
0	Reserved
1	Quota Identifier
2	VolumeQuota
3	VolumeThreshold
4	DurationQuota
5	DurationThreshold
6	ResourceQuota
7	ResourceThreshold
8	Update-Reason
9	PrepaidServer
10	Service-ID
11	Rating-Group-ID
12	Termination-Action
13	Pool-ID
14	Pool-Multiplier
15	Requested-Action
16	Check-Balance-Result
17..255	**Available for IANA registration**

Section 4.3.
Following the policies outline in [RFC3575] the available SubTypes (i.e., value 0 and values 22-255) with a description of their semantic will be assigned after the expert review process. Updates can be provided based on expert approval only. Based on expert approval it is possible to mark entries as "deprecated". A designated expert will be appointed by the IESG.
Each registration must include a number for the SubType and the semantic of the SubType.

8.4. New Registry: PTS SubTypes

Section 4.5 defines the SubTypes used within the PTS attribute. IANA is asked to create a registry for these SubTypes. Each registry entry consists of a 8 bit number together with a description of the PTS SubType. This document creates the following PTS SubTypes for this registry:

Value	SubType Name
0	Reserved
1	Quota Identifier
2	VolumeUsedAfterTariffSwitch
3	TariffSwitchInterval
4	TimeIntervalafterTariffSwitchUpdate
5..255	**Available for IANA registration**

Section 4.5.
Following the policies outline in [RFC3575] the available SubTypes (i.e., value 0 and values 5-255) with a description of their semantic will be assigned after the expert review process. Updates can be provided based on expert approval only. Based on expert approval it is possible to mark entries as "deprecated". A designated expert will be appointed by the IESG.
Each registration must include a number for the SubType and the semantic of the SubType.

8.5. New Registry: Update-Reason

Section 4.3.8 defines the Update-Reason SubType. IANA is asked to create a registry for the values contained in the Update-Reason SubType, as shown in Figure 21. Each registry entry consists of a 16 bit number together with a description of the update reason.
Following the policies outline in [RFC3575] the available values together with a description of their semantic will be assigned after the expert review process. Updates can be provided based on expert approval only. Based on expert approval it is possible to mark entries as "deprecated". A designated expert will be appointed by the IESG.

8.6. New Registry: Termination-Action

Section 4.3.12 defines the Termination-Action SubType. IANA is asked to create a registry for the values contained in the Termination-Action SubType, as shown in Figure 26. Each registry entry consists of a 8 bit number together with a description of the termination action.
Following the policies outline in [RFC3575] the available values together with a description of their semantic will be assigned after the expert review process. Updates can be provided based on expert approval only. Based on expert approval it is possible to mark entries as "deprecated". A designated expert will be appointed by the IESG.

8.7. New Registry: Requested-Action

Section 4.3.15 defines the Requested-Action SubType. IANA is asked to create a registry for the values contained in the Requested-Action

SubType, as shown in Figure 30. Each registry entry consists of a 8 bit number together with a description of the requested reason. Following the policies outline in [RFC3575] the available values together with a description of their semantic will be assigned after the expert review process. Updates can be provided based on expert approval only. Based on expert approval it is possible to mark entries as "deprecated". A designated expert will be appointed by the IESG.

8.8. New Registry: Check-Balance-Result

Section 4.3.16 defines the Check-Balance-Result SubType. IANA is asked to create a registry for the values contained in the Check-Balance-Result SubType, as shown in Figure 32. Each registry entry consists of a 8 bit number together with a description of the requested reason. Following the policies outline in [RFC3575] the available values together with a description of their semantic will be assigned after the expert review process. Updates can be provided based on expert approval only. Based on expert approval it is possible to mark entries as "deprecated". A designated expert will be appointed by the IESG.

9. Acknowledgements

The authors would like to thank Bernard Aboba, Christian Guenther, Dirk Kroeselberg and John Loughney for their feedback throughout the development of this document. Additionally, the authors would like to thank the members of the Wimax Forum and the members of 3GPP2 for their help with this specification.

10. References

10.1. Normative References

[RFC2119]	Bradner, S, "RFC 2119: Key words for use in RFCs to Indicate Requirement Levels", March 1997.
[RFC2865]	Rigney, C, Rubens, A, Simpson, W and S Willens, "RFC 2865: Remote Authentication Dial In User Server (RADIUS)", June 2000.
[RFC3576]	Chiba, M, Dommety, G, Eklund, M, Mitton, D and B Adoba, "RFC 3576: Dynamic Authorization Extensions to Remote Authentication Dial-In User Service (RADIUS)", February 2003.

10.2. Informative References

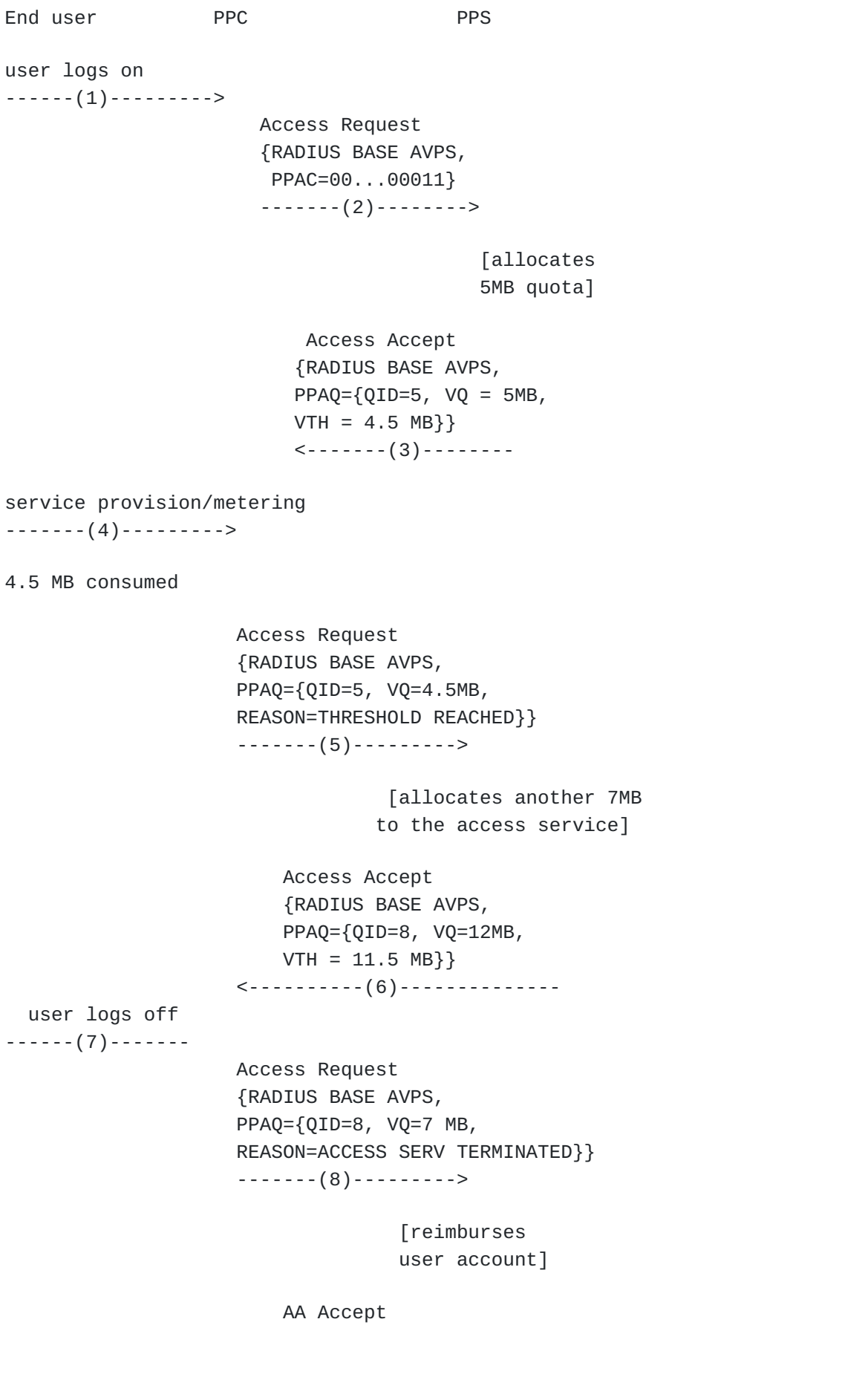
[RFC2866]	Rigney, C, "RFC 2866: RADIUS Accounting", June 2000.
[RFC2869]	Rigney, C, Willats, W and P Calhoun, "RFC 2869: RADIUS Extensions", June 2000.
[RFC3748]	

	Adoba, B, Blunk, L, Vollbrecht, J, Carlson, J and H Levkowetz, "RFC 3748: Extensible Authentication Protocol", June 2004.
[RFC4006]	Hakala, H, Mattila, L, Koskinen, J-P, Stura, M and J Loughney, "RFC 4006: Diameter Credit Control Application", August 2005.
[RFC2284]	Blunk, L.J. and J.R. Vollbrecht, "PPP Extensible Authentication Protocol (EAP)", RFC 2284, March 1998.
[RFC4849]	Congdon, P., Sanchez, M. and B. Aboba, "RADIUS Filter Rule Attribute", RFC 4849, April 2007.
[RFC3575]	Aboba, B., "IANA Considerations for RADIUS (Remote Authentication Dial In User Service)", RFC 3575, July 2003.
[RFC2607]	Aboba, B. and J. Vollbrecht, "Proxy Chaining and Policy Implementation in Roaming", RFC 2607, June 1999.
[RFC3579]	Aboba, B. and P. Calhoun, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)", RFC 3579, September 2003.
[RFC6158]	DeKok, A. and G. Weber, "RADIUS Design Guidelines", BCP 158, RFC 6158, March 2011.
[RFC3580]	Congdon, P., Aboba, B., Smith, A., Zorn, G. and J. Roese, "IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines", RFC 3580, September 2003.

Appendix A. Example flows

Note: This section is informative.
This section presents certain example flows that involve the RADIUS prepaid extensions. By no means is the intent of this section to specify or recommend business logic, rating strategies, and application-level behaviour. The intent of this section is purely to illustrate some fictive scenarios and the RADIUS prepaid message flows that could be associated with these scenarios. The contents of this section should be regarded as a collection of informative examples that aim to provide guidance to implementors.

Appendix A.1. A simple flow


End user
PPC
PPS
user logs on
------(1)--------->
Access Request
{RADIUS BASE AVPS,
PPAC=00...00011}
-------(2)-------->
[allocates
5MB quota]
Access Accept
{RADIUS BASE AVPS,
PPAQ={QID=5, VQ = 5MB,
VTH = 4.5 MB}}
<-------(3)--------
service provision/metering
-------(4)--------->
4.5 MB consumed
Access Request
{RADIUS BASE AVPS,
PPAQ={QID=5, VQ=4.5MB,
REASON=THRESHOLD REACHED}}
-------(5)--------->
[allocates another 7MB
to the access service]
Access Accept
{RADIUS BASE AVPS,
PPAQ={QID=8, VQ=12MB,
VTH = 11.5 MB}}
<----------(6)--------------
user logs off
------(7)-------
Access Request
{RADIUS BASE AVPS,
PPAQ={QID=8, VQ=7 MB,
REASON=ACCESS SERV TERMINATED}}
-------(8)--------->
[reimburses
user account]
AA Accept

{RADIUS BASE AVPS}
<-------(9)--------

The user logs on (1). The PPC sends a RADIUS Access Request message to the PPS (2), and includes the prepaid-specific PPAC AVP. This AVP indicates that both duration-based and volume-based metering is supported. However, it also indicated that multiple services, rating groups and resource pools are not supported. Note that, since this is not an "Authorize-Only" message, no PPAQ attribute with Update Reason="initial request" is included (see Section 3.7.1). The PPS then authenticates the user and authorizes the access service, as is usual in RADIUS. Note that the PPAC AVP is appended by the PPC in at least the last message that is sent to the home AAA server during this possibly multiple-round exchange.

If authentication and authorization is successful (in this example this is assumed), then the PPS identifies the user's prepaid account from the included base RADIUS AVPs, and determines the capabilities of the PPC from the PPAC attribute. Assuming that sufficient funds are available in the user's prepaid account, the PPS reserves some of these and rates the service. In this example, the PPS reserves, say, 2 Euros and determines that the access service is rated at 0.4 Euro per MB. This results in 5 MB of quota being granted. The PPS also determines that the PPC should ask for this quota to be replenished once 4.5 MB have been consumed. Thus, it creates an Access Accept message with a Volume-Threshold indication of 4.5MB. It further associates the QID=5 to this reservation. This identifier can be used to later uniquely identify the prepaid session, user, account, etc. The resulting Access Accept message is sent to the PPC (3).

Upon reception of message (4), the PPC provides the access service to the user and meters it accordingly. At some point in time, the threshold is reached, i.e., 4.5MB of "access service" have been consumed by the user. At that point, the PPC generates an Authorize-Only Access Request that contains the usual RADIUS attributes and a PPAQ attributes that reports the amount of consumed quota, and the request for replenishment, i.e., the Update-Reason= THRESHOLD REACHED (5). Note that the QID in this message is the same as the one previously received from the PPS.

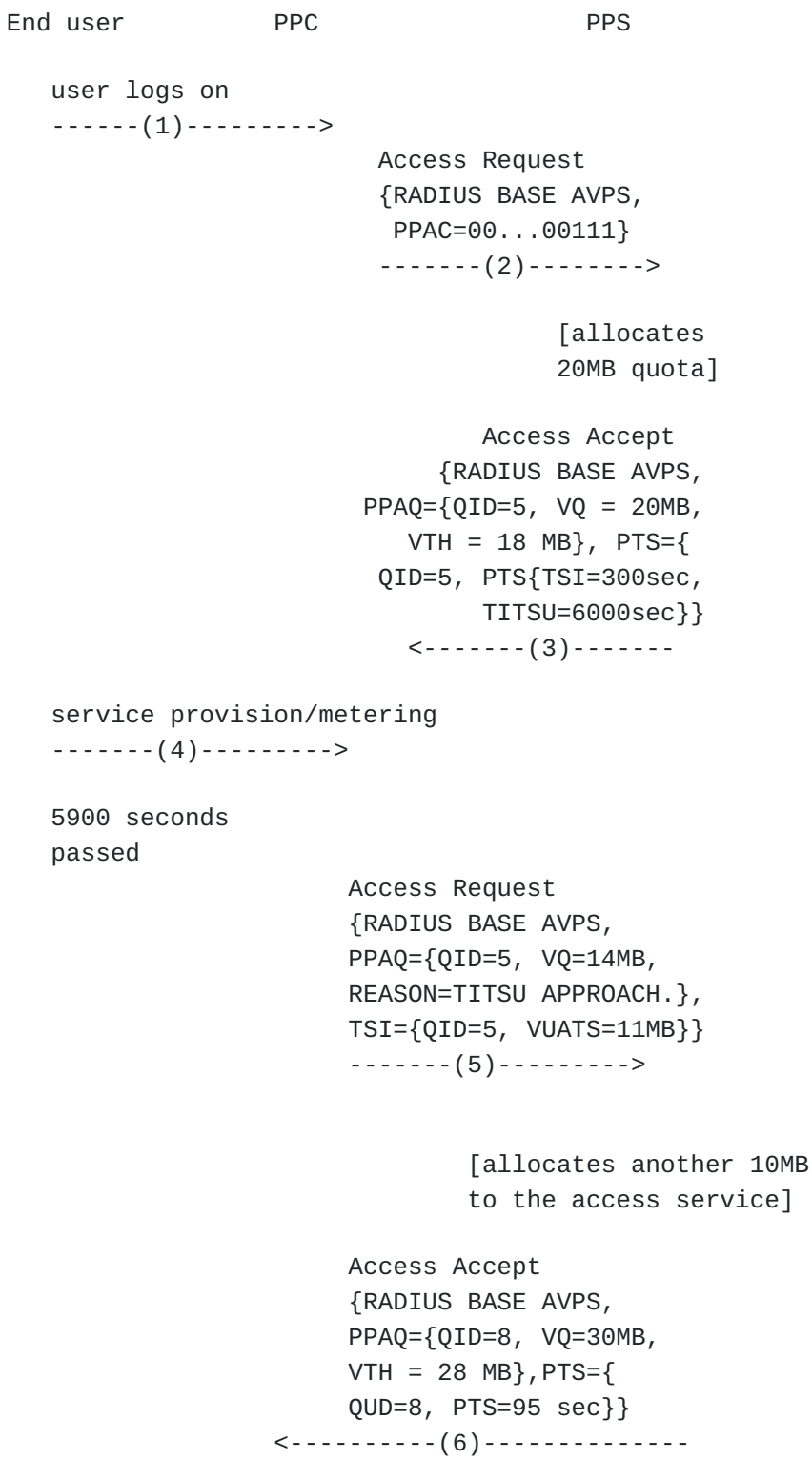
Upon reception of message (5), the PPS identifies the user and his account from the QID. In also determines that a prepaid session is ongoing, and that enough credit remains in the prepaid account in order for the access service to continue being provided. Since 4.5 MB have been consumed, the PPS subtracts 1.8 Euros from the user's prepaid account. The PPS decides to reserve another 2.8 euros from the user's account. (This results in 3 euros being reserved in total at this point in time.) As the access service is rated at 0.4 euros per MB, the PPS determines that another 7 MB of quota should be granted. This results in a total cumulative quota allocation of 12 MB for the access service. The PPS further calculates the new threshold value of 11.5 MB. Since

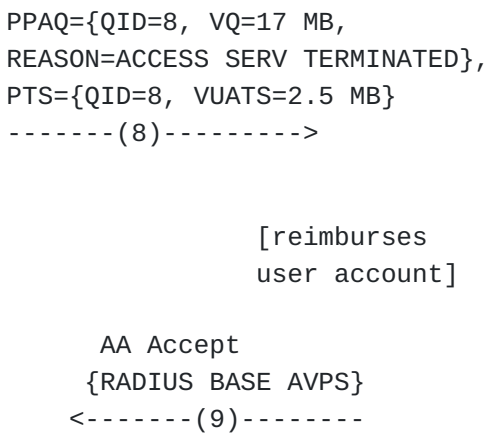
this is a new quota reservation, the PPS also allocates a new QID to it, in this example QID=8. The resulting RADIUS message is sent to the PPC (6).

Upon reception of message (6), the PPC updates its records and continues provisioning access to the user. At some point the user logs off (7). The PPC must then report how many resources were consumed, so that the PPC can subtract the appropriate monetary amount from the user's prepaid account. To this end the PPC constructs an Authorize-Only Access Request message with a PPAQ attributes for the access service. In this example, 7 MB were consumed by the access service in total. The PPC reports 7 MB its final message (8). The PPS correlates the report, using the QID, to the previous session state. It determines, from the previous records, that the access service had consumed another 4.5 MB before (as indicated in message (6)). This means that, of the 7 MB, only 3.5 MB have not yet been subtracted from the user's account. Thus, the PPS subtracts another 1.4 euros from the user's account and, since the session is to be terminated (REASON=ACCESS SERVICE TERMINATED), releases any reserved monetary amount.

The PPS responds with an Access Response as required by the RADIUS base specification (9).

Appendix A.2. A flow with prepaid tariff switching


End user
PPC
PPS
user logs on
------(1)--------->
Access Request
{RADIUS BASE AVPS,
PPAC=00...00111}
-------(2)-------->
[allocates
20MB quota]
Access Accept
{RADIUS BASE AVPS,
PPAQ={QID=5, VQ = 20MB,
VTH = 18 MB}, PTS={
QID=5, PTS{TSI=300sec,
TITSU=6000sec}}
<-------(3)-------
service provision/metering
-------(4)--------->
5900 seconds
passed
Access Request
{RADIUS BASE AVPS,
PPAQ={QID=5, VQ=14MB,
REASON=TITSU APPROACH.},
TSI={QID=5, VUATS=11MB}}
-------(5)--------->
[allocates another 10MB
to the access service]
Access Accept
{RADIUS BASE AVPS,
PPAQ={QID=8, VQ=30MB,
VTH = 28 MB},PTS={
QUD=8, PTS=95 sec}}
<----------(6)--------------
user logs off
------(7)-------
Access Request
{RADIUS BASE AVPS,



The user logs on (1). The PPC sends a RADIUS Access Request message to the home AAA server (2), and includes the prepaid-specific PPAC AVP. This AVP indicates that both duration-based and volume-based metering is supported, as well as tariff switching. The home AAA server then may authenticate and user and authorize the access service, as is usual in RADIUS. Note that the PPAC AVP is appended by the PPC in at least the last message that is sent to the PPS during this possibly multiple-round exchange.

If authentication and authorization is successful (in this example this is assumed), the PPS identifies the user's prepaid account from the included base RADIUS AVPs, and determines the capabilities of the PPC from the PPAC attribute. In this example, it is assumed that a tariff switch is about to occur in 300 seconds from the current time. Suppose that the access service is currently rated at 0.5 euros per MB and in the next tariff period it is rated at 0.6 euros per MB. Suppose further that a third tariff period is about to start in 6000 seconds from current time and that that access service is rated at 0.8 euros per MB in that period. The PPS then decides to reserve 12 euros from the user's account. Since it is conceivable that the user may consume all allocated quota in the (more expensive) "0.6-euro" period, the PPS reserves 20 MB of quota, and determines a threshold value of 18 MB. It constructs a Radius Access Accept message with a PPAQ attribute that reflects these choices, and carries a Quota Identifier QID=5. It further adds a PTS AVP in the message which is linked to the PPAQ via the common QID value. The PTS AVP contains a TSI attribute indicating that a tariff switch will occur in 300 seconds. It also includes a TITSU attribute with the value of 6000 seconds. This is included in order to make sure that the PPC will report the consumed quota before the "2-euro" tariff period will start. The message is sent to the PPC (3).

Upon reception of message (3), the PPC provides the access service to the user and meters it accordingly (4). It also keeps track of time. That is, it remembers how many octets are consumed before and how many after the tariff switch that will take place in 300 seconds.

In this example it is assumed that the user consumes the allocated quota rather slowly. In particular, nearly 6000 seconds (the value indicated by TITSU SubType) pass without the threshold of 18 MB being reached. The PPC notices this and must therefore report usage and request the quota to be replenished despite the fact that the threshold has not been reached. In this example, it decides to do so 100 seconds before the 6000 seconds are reached. To this end, it constructs an Authorization Access Request message including a PPAQ that indicates that 14 MB have been consumed up to now. It also includes a PTS attribute in order to indicate, using the VUATS SubType, that 11 MB of these were consumed after the tariff switch. The message is sent to the the PPS (5).

The PPS can link the message to previous session state via the QID. It now rates the consumed volume as follows. The 11 MB that were consumed after the tariff switch correspond to 11 * 0.6 = 6.6 euros and the remaining 14-11=3 MB to 3 * 0.5 = 1.5 euros. Thus, the PPS subtracts the amount of 6.6+1.5=8.1 euros from the user's account, which leads to a remainder of 12 - 8.1 = 3.9 euros being reserved.

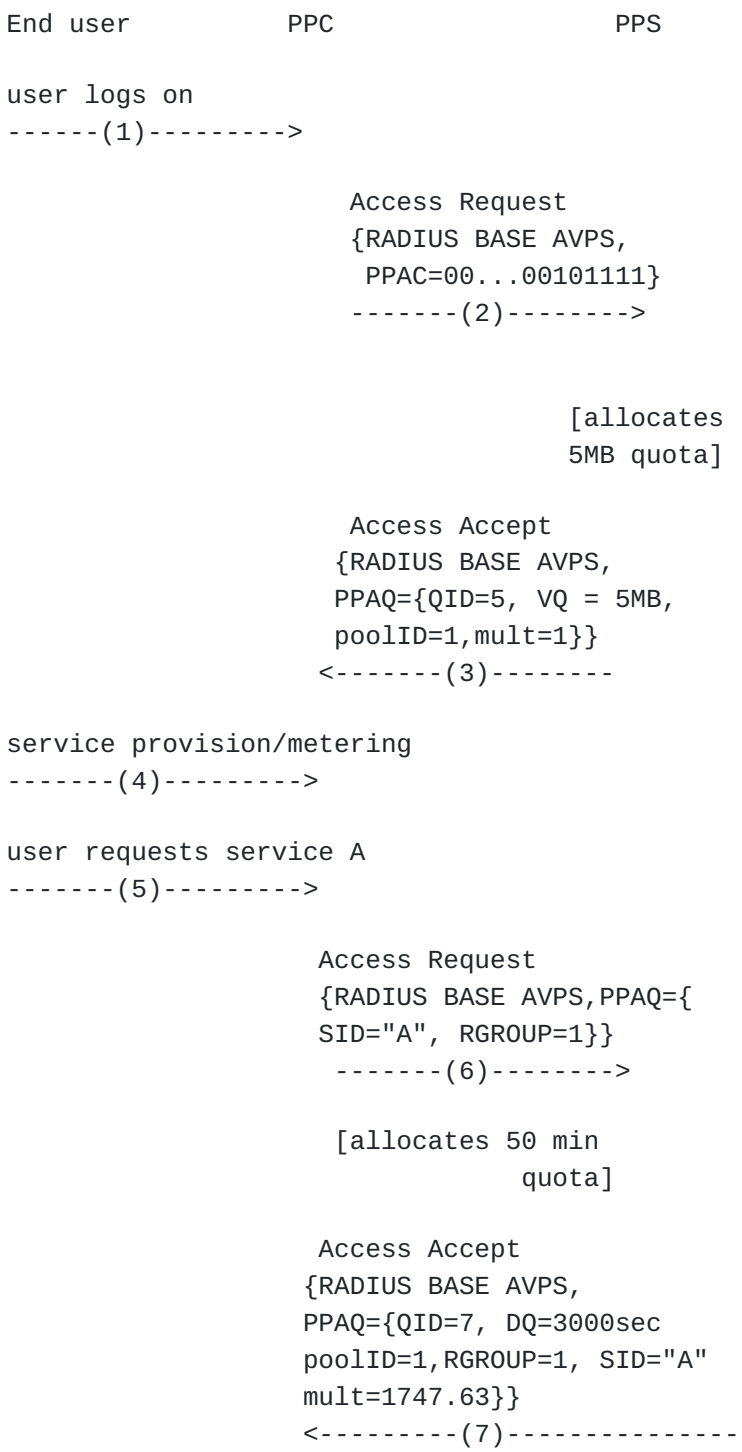
The PPS now determines that message (5) was sent in order to replenish the quota for this prepaid session. This can be deduced from the UPDATE REASON field, which indicates that the PPC sent this message because the time indicated by the TITSU SubType is approacing. The PPS now determines that enough credit remains in the user's prepaid account in order for the access service to continue being provided and decides to reserve another 8.9 euros from the user's account. Since it is conceivable that the user will consume the 6 unused MB of quota from the previous allocation, as well as the entire quota that is to be allocated now, entirely in the "0.8-euro" period, the quota that should now be granted in addition to the previous 20 MB should be 10 MB. This is because 0.9 of the 8.9 euros are being reserved in order to "cover the worst case scenario". The fact that 0.9 euros are reserved for this purpose is due to the fact that the unused 6 MB from the previous allocation correspond to 4.8 euros (with 0.8 euros per MB). This is 4.8 - 3.9 = 0.9 euros more than the amount of funds that are still "reserved" from the previous allocation. (After this reservation, the total amount of reserved money is 8.9 + 3.9 = 12.8 euros, which corresponds to 16 (10+6) MB being consumed in the "0.8-euro" period.)

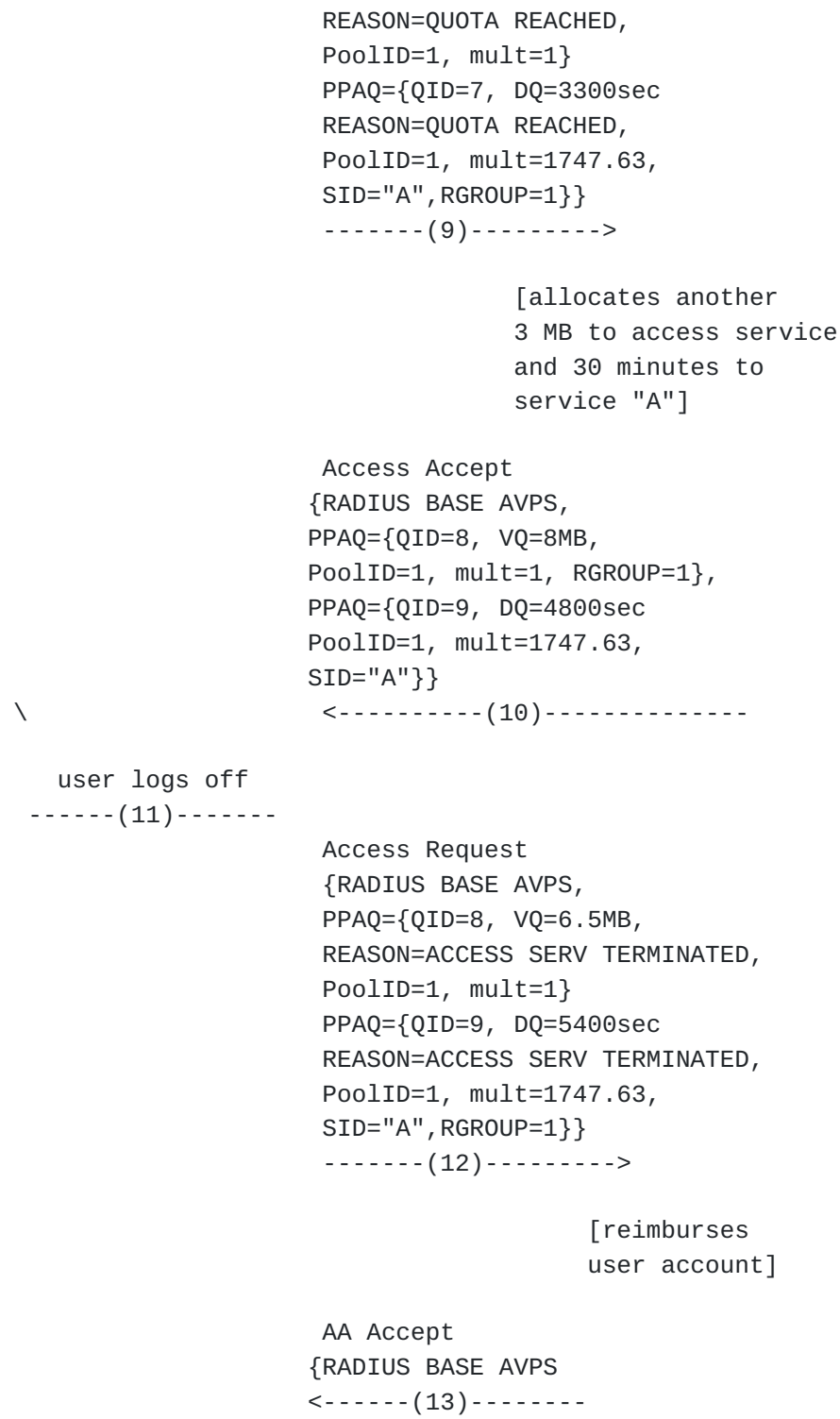
Since quotas are encoded in a cumulative way in RADIUS, the PPS includes a VolumeQuota of 30 MB into the Access Accept message (6). The PPS further calculates the new threshold value of 28 MB. Since this is a new quota reservation, the PPS also allocates a new QID to it, in this example QID=8. The resulting RADIUS message is sent to the PPC (6).

Upon reception of message (6), the PPC updates its records and continues providing access to the user. At some point the user logs off (7). The PPC must then report how many resources were consumed, so that the PPC can subtract the appropriate monetary amount from the user's prepaid account. To this end the PPC constructs an Authorize- Only Access Request message with a PPAQ attributes for the access service.

In this example, 17 MB were consumed by the access service in total. The PPC reports 17 MB its final message (8). The PPS correlates the report, using the QID, to the previous session state. It determines, from the previous records, that the access service had consumed 14 MB before (as indicated in message (5)). This means that, of the 17 MB, only the monetary equivalent for 3 MB have not yet been subtracted from the user's account. The PPS calculates how much should be deducted from the user's account as follows. Since the VUATS SubType indicates that 2.5MB were consumed after the tariff switch, only 0.5 MB were consumed before that. Thus, the monetary equivalent is 0.5 * 0.6 + 2.5 * 0.8 = 3.6 euros. That is, the PPS subtracts 3.6 euros from the user's prepaid account. Since the session has by now be terminated by the PPC (REASON=ACCESS SERVICE TERMINATED), the PPS now releases any reserved monetary amount, in this example 12.8 - 3.6 = 9.2 euros.
The PPS responds with an Access Response as required by the RADIUS base specification (9).
Remark: In this example, two tariff switches take place. In other scenarios, of course, only one tariff switch may occur. In such scenarios the TITSU SubType is not used.

Appendix A.3. Resource pools and Rating Groups


End user
PPC
PPS
user logs on
------(1)--------->
Access Request
{RADIUS BASE AVPS,
PPAC=00...00101111}
-------(2)-------->
[allocates
5MB quota]
Access Accept
{RADIUS BASE AVPS,
PPAQ={QID=5, VQ = 5MB,
poolID=1,mult=1}}
<-------(3)--------
service provision/metering
-------(4)--------->
user requests service A
-------(5)--------->
Access Request
{RADIUS BASE AVPS,PPAQ={
SID="A", RGROUP=1}}
-------(6)-------->
[allocates 50 min
quota]
Access Accept
{RADIUS BASE AVPS,
PPAQ={QID=7, DQ=3000sec
poolID=1,RGROUP=1, SID="A"
mult=1747.63}}
<---------(7)---------------
user requests service B
-------(8)-------->
Pool 1 close to exhaustion
Access Request
{RADIUS BASE AVPS,
PPAQ={QID=5, VQ=4MB,



The user logs on (1). The PPC sends a RADIUS Access Request message to the PPS (2), and includes the prepaid-specific PPAC AVP, indicating that multiple services, rating groups and resource pools are supported. Note that, since this is not an "Authorize- Only" message, no PPAQ attribute with Update Reason="initial request" is included (see Section 3.7.1). The PPS then may authenticate the user and authorize the access

service, as is usual in RADIUS. Note that the PPAC AVP is appended by the PPC in at least the last message that is sent to the PPS during this possibly multiple-round exchange.

If authentication and authorization is successful (in this example this is assumed), the PPS identifies the user's prepaid account from the included base RADIUS AVPs, and determines the capabilities of the PPC from the PPAC attribute. Assuming that sufficient funds are available in the user's prepaid account, the PPS reserves some of these and rates the service. In this example, the PPS reserves 5 Euros and determines that the access service is rated at 1 Euro per MB. In anticipation that the user requests more chargeable services throughout this prepaid session, and since this is supported by the PPC, the PPS further associates a resource pool with this reservation, in this example PoolID=1. The PPC also specifies the multiplier = 1 for the access service. Note that, since 5MB = 5242880 octets, 1 unit in the resource pool corresponds to 5 / 5242880 euros, which is about 0.000095367431640625 Eurocents. (However, the PPC does not need to know that.) Moreover, the PPS associates the QID=5 to this reservation. This identifier can be used to later uniquely identify the prepaid session, user, account, etc. The resulting Access Accept message is sent to PPC (3).

Upon reception of message (3), the PPC provides the access service to the user and meters it accordingly (4). That is, for every octet consumed, the PPC subtracts 1 unit (since the multiplier is 1) from the resouce pool with PoolID=1.

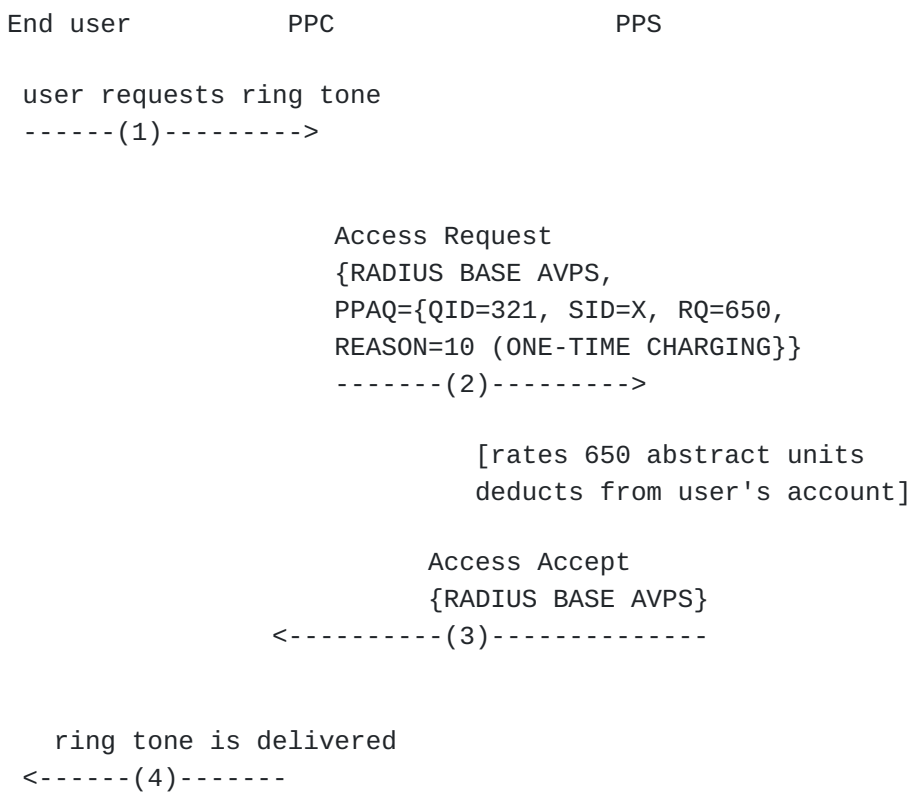
At some point in time, the user requests another chargeable service, namely service A (5). The PPC generates an Authorize-Only Access Request that contains the usual RADIUS attributes and the Service-ID identifying service A (6). The PPC has determined that service A is rated in an identical way as at least one more service. Thus, service A has been configured to belong to a rating group, in this example the group with Rating-Group-ID=1. This identifier is included is message (6).

Upon reception of message (6), the PPS identifies the user and his account from the base RADIUS attributes, the fact that a prepaid session is ongoing, and determines that enough credit remains in the prepaid account in order for service A to be provided. The PPS also determines that service A is rated at 0.10 euros per minute. The PPS decides to reserve another 5 euros from the users account; this corresponds to 50 minutes or, as encoded in the DurationQuota SubType, 3000 seconds. As service A draws from the same prepaid account as the access service, the PPS associates this reservation with the same resource pool as the previous reservation (QID=5), namely the pool with PoolID=1. Note that, in order for the abstract units in the pool to be consistent, the multiplier has to be 1747.63. This is because each second corresponds to about 0.10 / 60 = 0.00167 euros, which is about 1747.63 times the value of an abstract resource pool unit, as this was determined by the first allocation of quota to the pool (i.e., 0.000095367431640625 Eurocents). Since this is a new quota reservation,

the PPS also allocates a new QID to it, in this example QID=7. The resulting RADIUS message is sent to the PPC (7).
Upon reception of message (7), the PPC adjusts the units in resource pool 1. That is, it first determines how much quota had been allocated to service A in the past, and subtracts this from the quota reservation found in the message. Since this is the first quota reservation for service A, there is nothing to subtract. Thus, it adds 3000 * 1747.63 = 5242890 units to the pool and remembers that 3000 seconds have been allocated to service A during this prepaid session. The PPC then provides service A to the user, and meters it against resource pool 1. That is, for every second it subtracts 1747.63 units from the pool.
At some point in time, the user requests service B (8). The PPC determines that service B is rated exactly in the same way as service A, i.e., that they belong to the same rating group, namely the one with Rating-Group-ID=1. Since this rating group has been effectively authorised by the allocation of quota with QID=7, the PPC provides service B to the user immediately. It is rated in the same way as service A, i.e., for every second provided, 1747.63 units are subtracted from credit pool 1.
At some point in time, resource pool 1 is close to exhaustion. (For example, the PPC may determine that the pool is "close to exhaustion" when has less than 10% its initial amount of units.) At that point, the PPC needs to ask for replenishment for the pool. Suppose that, at that point in time, 4MB of "access service", 45 minutes of "service A", and 10 minutes of "service B" were provided to the user. Note that this corresponds to (4*1048576) + (55*60*1747.63) = 4194304 + 5767179 = 9961483 abstract service units from the pool. The PPC constructs an Authorize-Only Access Request message that reports the usage for the "access service" and "service A". This message contains two PPAQ attributeS, is sent to the PPS (9). Note that is the message it appears that "service A" has consumed more than it was allocated (i.e., 55 minutes although only 50 minutes were initially allocated to it). This is not a a problem since the PPS knows that "service A" was drawing from the same pool as the "access service" and that the "access service" did only consume 4 out of the 5 MB it was allocated.
Upon reception of message (9), the PPS subtracts 4 euros from the user's account for the "access service" and another 5.5 euros for "service A". (This includes the charge incurred by "service B" up to that point in time, although the PPS is not aware of "service B" being provisioned to the user.) The PPS then determines that sufficient funds remain in the prepaid account in order for both services to be continued. The PPS decides to reserve another 3MB for the access service and 30 minutes for "service A". This corresponds to 3+3=6 euros. Since in RADIUS prepaid the quotas are encoded in a cumulative manner, the PPAQ attribute that grants the quota for the "access service" contains a Volume-Quota SubType of 8MB (8388608 octets), which is the 5MB that were initially allocated, plus the 3MB allocated now. The resource pool identifier is, as previously, PoolID=1 and the multiplier is 1. Similarly, the PPAQ that grants quota for "service A"

contains 4800 seconds (the initial 3000 plus 1800 that correspond to the 30 additional minutes). Again, the PoolID=1 and multiplier=1747.63. The resulting Access Response message is sent to the PPC (10).
When the PPC received message (10) it checks how much quota has been allocated previously to the "access service". It finds that the answer is 5MB (5242880 octets); thus, out of the 8MB (8388608 octets) that are indicated by the PPAQ with QID=8, only 3MB (3145728 octets) have not yet been added to resource pool 1. The PPC thus adds 3145728 abstract units to resource pool 1 (since the multiplier is 1). The PPC then acts similarly on the other PPAQ attribute that exists in message (11). That is, the PPC determines that 3000 seconds of quota for "service A" had already been added to the pool. Thus only 1800 out of the 4800 should be additionally added to the pool. Since the applicable multiplier here is 1747.63, the PPC adds further 3145734 abstract units to the pool 1. The PPC then continues to provide the access service, "service A" and "service B" to the user, and meters them against the pool, as previously.
At some point the user logs off (11). The PPC must then report how many resources were consumed, so that the PPC can subtract the appropriate monetary amount from the user's prepaid account. To this end the PPC constructs an Authorize-Only Access Request message with two PPAQ attributes; one for the access service and one for "service A". Suppose that, in total, 6.5MB were consumed by the access service, 70 minutes were consumed by "service A" and 20 minutes by "service B". The PPC reports 6.5MB (6815744 octets) and 90 minutes (5400 seconds) in its final message (12). The PPS determines, from the previous records, that the access service consumed another 2.5MB (since 4MB out of the 6.5MB were already reported in message (9), and that "service A" consumed further 600 seconds. This corresponds to 2.5 + (600/60)*0.1 = 2.5+ 1=3.5 euros. Thus, the PPS only subtracts 2.5 out of the 6 previously reserved euros from the user's prepaid account and responds with an Access Response as required by the RADIUS base specification (13).

Appendix A.4. One-time charging

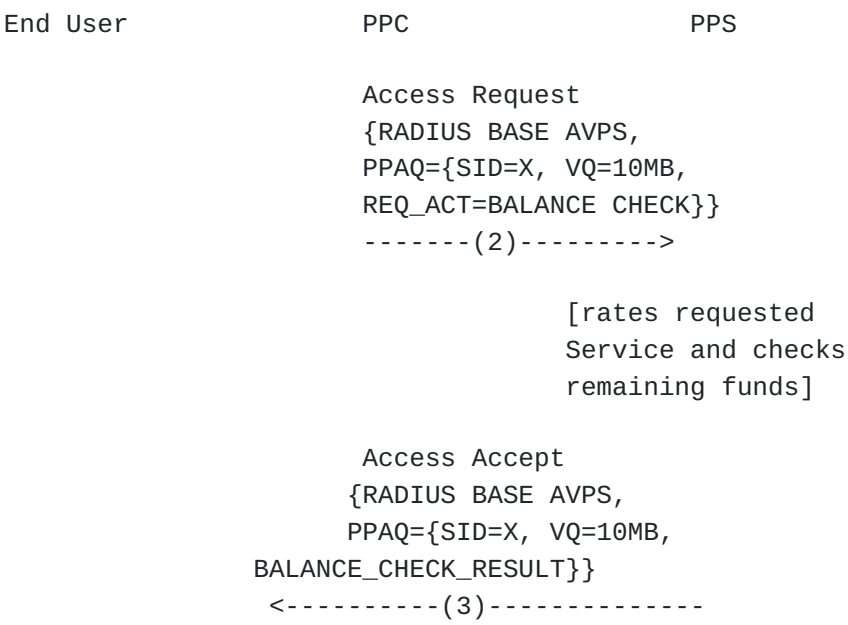


The user requests a chargeable ring tone (1). The PPC sends a RADIUS Access Request message to the PPS (2), and includes a PPAQ attribute with Update Reason="one-time charging" is included (see Section 3.8.6). The Service ID indicates to the PPS that the charging event is connected to a ring tone, so that the PPS can rate the event accordingly. The PPAQ also contains a unique Quota Identifier. The PPS then may authenticate the user as is usual in RADIUS. If authentication is successful (in this example this is assumed), the home AAA server forwards the PPC converts the 650 reported abstract units into monetary value, according to the service type, and debit the user's account accordingly. This happens, of course, only if sufficient funds are available in the user's prepaid account. The PPC then responds with an an Access Accept message (3). The PPS adds a "session timeout = 0 AVP" (see Section 3.8.6).

Appendix A.5. Price enquiry

Please refer to Section 2.7.3 for an explanation of this message flow.

Appendix A.6. Balance check



Please refer to Section 2.7.4 for an explanation of this message flow.

Appendix B. Translation between RADIUS Prepaid and Diameter Credit Control

Note: This section is informative.
In scenarios where the service metering device uses the "RADIUS prepaid" (RPP) protocol for accounting and prepaid charging while the AAA infrastructure uses the "Diameter Credit Control" (DCC) protocol, a translation agent that enables the interoperation of both systems, is desirable. This also applies vice versa, i.e., in scenarios where the AAA infrastructure uses RADIUS and the service metering device uses Diameter.
The idea of such a translation agent would be to convert incoming RPP (resp. DCC) messages into outgoing DCC (resp. RPP) messages. It would be, in principle, desirable for the translation agent to be stateless. That is, the agent should not be required to internally maintain information about each ongoing RADIUS or Diameter session. However, under the current specification of RPP and DCC, this appears to be impossible due to a number of reasons. These include the following.

1. The transport mechanism for DCC is TCP, which requires per-session state to be maintained at both endpoints of the communication. Note, however, that, in principle, each DCC message could be sent over a dedicated TCP connection which is torn down as soon as the message is sent. This, however, is likely to be unacceptable in terms of efficiency.

2. While RPP messages encode the cumulative amount of consumed/requested resources, DCC messages carry the difference from the

previous message. This means that the translation agent has to maintain the current amount of consumed/requested resources in order to be able to calculate the correct amount to be put into an outgoing message.

The translator maps each incoming RPP (resp. DCC) message into an outgoing DCC (resp. RPP) message, and possibly establishes or updates local state that is associated with the session. The translated (i.e., outgoing) message is a function of the incoming message as well as existing state that is associated with the current session. Translation occurs on an attribute-by-attribute basis. Certain attributes are translated without consideration of local per-session state. Other attributes, namely those that are bound to a particular session, require such consideration. The translation agent has to identify the session (and possibly subsession) an incoming message belongs to in order to consult the appropriate local per-session state. Note that certain DCC attributes cannot be translated due to their semantics not being present in RPP, and vice versa. This results in the messages, in which these attributes occur, not being delivered to their intended destination. In such cases it is desirable to inform the originator about the failure and terminate the session.
In each scenario (i.e., RPP client / DCC AAA infrastructure and DCC client / RPP AAA infrastructure), the translator operates in two directions, namely RPP to DCC and vice versa. In the following sections, the notation c->s means that the attribute in question may occur only in the direction from the client to the server. The notation s->c denotes the converse and the notation c<->s denotes that the attribute may occur in messages that are directed in either direction.

Appendix B.1. Session Identification

The translation agent has to keep per-session state in order to perform its task. A session may be identified based on the RPP identifier or the DCC session identifier. That is, the translation agent should always maintain a pair of (RPP, DCC) session identifiers and maintain the per-session state in association with that pair. This per-session state must be addressable by either of these two identifiers. Moreover, an RPP session identifier must uniquely correspond to a DCC identifier. (If this holds, the converse also holds.) Each subsession identifier within an RPP session must also uniquely correspond to a subsession identifier within its corresponding DCC session. (If this holds the converse also holds.)

Appendix B.2. Translation between RADIUS Prepaid and Diameter Credit Control

This section describes the translator in the "RPP client / DCC AAA infrastructure" case. In other words, in this section it is assumed that the client "talks" RPP and the AAA inftrastructure "talks" DCC.

The translator is assumed to sit somewhere in the middle and to mediate between client and server.
For each RPP AVP (i.e., AVPs that are specified in the present document), the transformation into a semantically equivalent DCC AVP (if such an AVP exists), along with what per-session state the translator has to create or consult, is described. For clarity of exposition, each RPP AVP is addressed in a separate subsection. Since in this scenario, the PPC is typically the initiator a session, the focus is on the RPP AVPs.

Appendix B.2.1. PPAC (c<->s)

A DCC client is assumed to always support Volume metering, Duration metering, Resource metering, Pools, Rating groups, and Tariff Switching. Thus, if a PPAQ that indicates any of the above is sent client->server, the translator does the following: It lets message go through but remembers what exactly the client supports. If the server later requests (servier -> client direction) an unsupported metering to be performed, send failure to server and cause the session to be terminated at the client.
If a PPAC indicates support for multiple services (0x00000020), the translator maps this onto a DCC Multiple-Services- Indicator AVP.

Appendix B.2.2. Service Termination Attribute (c->s)

The Diameter base protocol assumes that the client always supports dynamic session termination. If this AVP is present, the translator does not need to do anything, i.e., there exists no DCC AVP that this AVP can be mapped to. If this AVP is absent, the message in which it appears should either be discarded and originator should be informed of a failure, or the message can be passed on (without this AVP being mapped onto a DCC AVP). However, in the latter case, the translator has to remember that the client does not support dynamic termination. Thus, the translatior has to initiate the normal session termination procedure with the client when (if) dynamic termination is later initiated by the server.

Appendix B.2.3. Quota Identifier Attribute (c<->s)

When quota is allocated for the first time by the DCC server, the translator has to create a QID AVP, as required by this specification. The translator later uses a QID AVP that is sent in the client-to-server direction in order to identify the corresponding DCC session. The QID has to be saved in the translator's per session state.

Appendix B.2.4. Volume Quota Attribute (c<->s)

If this AVP occurs in a message that is sent in the server-to-client direction, it is translated into a Granted-Service-Unit AVP with an embedded CC-Total-Octets AVP.

If this AVP occurs in a message that is sent in the client-to-server direction, then it is translated into a Used-Service-Unit AVP with an embedded CC-Total-Octets AVP. Note that only the difference between current cumulative quota for the (sub)session and the quota in incoming messages is indicated in the translated DCC message. Local state is updated with cumulative consumed resources.
Conversely, if the server grants quota using the DCC Granted-Service-Unit AVP with an embedded CC-Total-Octets AVP, then the translation agent must translate this into a Volume Quota Attribute. Again, local state must be consulted so that the cumulative amount of octets is indicated in the Volume Quota attribute.

Appendix B.2.5. Duration Quota Attribute (c<->s)

If this AVP occurs in a message that is sent in the server-to-client direction, it is translated into a Granted-Service-Unit AVP with an embedded CC-Time AVP.
If this AVP occurs in a message that is sent in the client-to-server direction, then it is translated into a Used-Service-Unit AVP with an embedded CC-Time AVP. Note that only the difference between current cumulative quota for the (sub)session and the quota in incoming messages is indicated in the translated DCC message. Local state is updated with cumulative consumed resources (i.e., time).
Conversely, if the server grants quota using the DCC Granted-Service-Unit AVP with an embedded CC-Time AVP, then the translation agent must translate this into a Duration Quota attribute. Again, local state must be consulted so that the cumulative amount of seconds is indicated in the Duaration Quota attribute.

Appendix B.2.6. Resource Quota Attribute (c<->s)

If this AVP occurs in a message that is sent in the server-to-client direction, it is translated into a Granted-Service-Unit AVP with an embedded CC-Service-Specific-Units AVP.
If this AVP occurs in a message that is sent in the client-to-server direction, then it is translated into a Used-Service-Unit AVP with an embedded CC-Service-Specific-Units AVP. Note that only the difference between current cumulative quota for the (sub)session and the quota in incoming messages is indicated in the translated DCC message. Local state is updated with cumulative consumed resources (i.e., resources).
Conversely, if the server grants quota using the DCC Granted-Service-Unit AVP with an embedded CC-Service-Specific-Units AVP, then the translation agent must translate this into a Resource Quota attribute. Again, local state must be consulted so that the cumulative amount of resource units is indicated in the Resource Quota attribute.
Note that the "resource" type is application dependent. This means that a DCC application unit corresponds to n RPP application units, where n may be any real number. If n is not 1, then the RPP/DCC translator must be aware of that and translate resource units accordingly.

Appendix B.2.7. Value Digits Attribute (c<->s)

The encoding of this AVP is similar in RPP and DCC, and the value it holds may have to be evaluated in conjunction with an acommpanying "Exponent" AVP. It should be kept in mind that, in RPP the cumulative amount of granted/consumed quota is typically encoded into an AVP of this type, while in DCC only the difference from a previous message.

Appendix B.2.8. Exponent Attribute (c<->s)

The encoding of this AVP is similar in RPP and DCC, and the value it holds may have to be evaluated in conjunction with an acommpanying "Value Digits" AVP. It should be kept in mind that, in RPP the cumulative amount of granted/consumed quota is typically encoded into a related "Value Digits" and "Exponent" AVP pair, while in DCC only the difference from a previous message is encoded into such a pair.

Appendix B.2.9. Volume/Duration/Resource Threshold Attributes (s->c)

In DCC the concept of "threshold" does not exist. Instead, the DCC client is assumed to ask for the replenishment of quota in good time. In RPP, on the other hand, the server may optionally include a threshold AVP, as an indication to the PPC about when to ask for quota replenishment.
Thus, in this scenario, there is no need for the translator to ever include a threshold attribute into the messages that it sends to the PPC. If, however, there is a need for a threshold attribute to be present in order to avoid a possible service provision

Appendix B.2.10. Update Reason Attribute (c->s)

The DCC AVP that is semantically closer to the Update Reason AVP than any other AVP is the CC-Request-Type AVP. This AVP indicates whether the message is which it appears is intended to indicate an "initial", an "intermediate" or a "final interrogation". Morever, in case of the session being terminated at the client, it indicates the reason for this termination.
The following list lists the possible values of an "Update Reason" attribute, along with corresponding values for the CC-Request-Type AVP.

*Pre-initialization: No action/value defined.

*Initial Request: Typically an "intial interrogation" is triggered as a result of the reception of the message that contains this Update Reason AVP. Hence, CC-Request-Type AVP indicates "INITIAL_REQUEST".

*Threshold Reached: The reception of the message containing this Update Reason AVP typically triggers an "intermediate

interrogation". Hence, CC-Request-Type AVP indicates "UPDATE_REQUEST".

*Quota Reached: The reception of the message containing this Update Reason AVP typically triggers an "intermediate interrogation". Hence, CC-Request-Type AVP indicates "UPDATE_REQUEST".

*TITSU Approaching: The reception of the message containing this Update Reason AVP typically triggers an "intermediate interrogation". Hence, CC-Request-Type AVP indicates "UPDATE_REQUEST".

*Remote Forced Disconnect: Reception of such an Update Reason indicates that the client has terminated the session. The corresponding value for the CC-Request-Type AVP is "TERMINATION_REQUEST".

*Client Service Termination: Reception of such an Update Reason indicates that the client has terminated the session. The corresponding value for the CC-Request-Type AVP is "TERMINATION_REQUEST".

*"Access Service" Terminated: Reception of such an Update Reason indicates that the client has terminated the session. The corresponding value for the CC-Request-Type AVP is "TERMINATION_REQUEST".

*Service not established: Reception of such an Update Reason indicates that the client has terminated the session. The corresponding value for the CC-Request-Type AVP is "TERMINATION_REQUEST".

*One-Time Charging: Such an Update Reason indicates that a one-time charging event is initiated by the client. The corresponding value for the CC-Request-Type AVP is "EVENT_REQUEST". Note that a "Requested-Action: AVP MUST also be included in the outgoing DCC message. Typically, this would be of the type "DIRECT_DEBITING", or "REFUND_ACCOUNT", depending on other AVPs present in the message.

Appendix B.2.11. PrepaidServer Attribute (s<->c)

The PPC typically never sets the value of a PrepaidServer attribute. Instead, it repeats those values that it receives from the AAA infrastructure, in this scenario from the translator. This attribute is therefore not used in a translation scenario. Nevertheless, the translator must make sure that messages about the same RPP session are forwarded to the same DCC server, throughout the whole session. This may be easy to guarantee since the transport of Diameter is TCP.

Appendix B.2.12. Service-ID Attribute (s<->c)

The DCC equivalent of a RPP "Service-ID" AVP is the combination of Service-Context-Id and Service-Identifier AVPs. The translator must keep a static equivalence table of the RPP Service-ID and the corresponding DCC combination in order to correctly translate an RPP service identifier into DCC and back.

Appendix B.2.13. Rating-Group-ID Attribute (s<->c)

The DCC equivalent of a RPP "Rating-Group-ID" AVP is also called a "Rating-Group-ID". Depending on the configuration, this AVP may contain the same value on both the RPP and the DCC side of the communication. If, however, static rating groups are configured between the RCC client and the translator, and different rating groups between the DCC server and the translator, then the translator has to maintain a static translation table for the rating group identifier. In any case, the translation of a rating group AVP, is not a function of the translator's local per-session state.

Appendix B.2.14. Termination-Action Attribute (s->c)

The DCC equivalent of the "Termination-Action" AVP is called the "Final-Unit-Action" AVP. In this scenario (RPP client and DCC AAA infrastructure), a DCC "Final-Unit-Action" AVP is translated into a "Termination-Action" AVP. The following list contains the possible "Final-Unit-Action" values along with their "Termination-Action" equivalent.

*TERMINATE (DCC): This value has a direct equivalent in RPP, also called "Terminate".

*REDIRECT (DCC): If this value appears in a "Final-Unit-Action" AVP, then a "Redirect-Server-Address" AVP must also appear in the same DCC message. The translator translates these two AVPs into a "Termination-Action" with value "Redirect/Filter" and an eqiovalent NAS-Filter-Rule attribute (specified in http://www.ietf.org/internet-drafts/draft-ietf-radext-ieee802-00.txt).

*RESTRICT_ACCESS (DCC): If this value appears in a "Final-Unit-Action" AVP, then a "Restriction-Filter-Rule" AVP must also appear in the same DCC message. The translator translates these two AVPs into a "Termination-Action" with value "Redirect/Filter" and an eqiovalent Filter-ID attribute (specified in http://www.ietf.org/internet-drafts/draft-ietf-radext-ieee802-00.txt).

*In the absence of a "Final-Unit-Action" AVP, the DCC server assumes that the DCC client will ask for replenishment of quota at some suitable time. In RPP, this is explicitly conveyed via a "Termination-Action" AVP with the value "Request More Quota".

Thus, in the absence of a "Final-Unit-Action" AVP, the translator in this scenario appends such an AVP into the outgoing RPP message.

Appendix B.2.15. Pool-ID Attribute (s<->c)

The DCC equivalent of a RPP "Pool-ID" AVP is also called a "Pool-ID". Typically, no translation needs to be done to the "Pool-ID" attribute.

Appendix B.2.16. Multiplier Attribute (s<->c)

The multiplier attribute, which is a pair of "Value-Digits" and "Exponent" AVPs, typically needs no translation, since the value it carries (inside a "Value-Digits" and an "Exponent" AVP) represents the rating of the service or rating group to which it refers, with respect to abstract units. As such, the same multiplier value would typically applyt be conveyed from a DCC server to an PPC, and vice versa.

Appendix B.2.17. Requested-Action Attribute (c->s)

The "Requested Action" AVP can be directly translated into its DCC equivalent, which carries the same name.

1. Balance Check (PCC): CHECK_BALANCE (DCC)

2. Price Enquiry (PCC): PRICE_ENQUIRY (DCC)

Appendix B.2.18. Check-Balance-Result Attribute (s->c)

This attribute carries only a binary value. Hence, its translation is straightforward.

Appendix B.2.19. Cost-Information Attribute (s->c)

This attribute consists of a Value-Digits AVP, an Exponent AVP, a Currency Code AVP, and a Cost-Unit AVP. All these AVPs do likewise exist in DCC, and carry identical semantics in the context of the "Cost-Information" AVP. Thus, the translation of this attribute is straightforward.

Appendix B.2.20. VolumeUsedAfterTariffSwitch attribute (c->s)

This attribute carries the amount of octets that were consumed after a tariff change. It always appears in a message with an accompanying PPAQ attribute in which the total amount of octets (i.e., those that were consumed both before and after the tariff switch) is reported. Thus, the translation agent can compute the amount of octets that were consumed before the tariff change.
In DCC, the two amounts, i.e., the octets that were consumed before a tariff change and those that were consumed afterwards, are reported in

separate Used-Service-Unit AVPs. The two Used-Service-Unit AVPs have an embedded CC-Total-Octets AVP that indicates the appropriate amount of octets. Furthermore, the Used-Service-Unit AVP that carries the amount that was consumed before the tariff switch also carries an embedded Tariff-Change-Usage AVP with the value UNIT_BEFORE_TARIFF_CHANGE (0). Similarly, the Used-Service-Unit AVP that carries the amount that was consumed after the tariff switch also carries an embedded Tariff-Change-Usage AVP with the value UNIT_AFTER_TARIFF_CHANGE (1).

Authors' Addresses

Avi Lior Lior Bridgewater Systems 303 Terry Fox Drive Ottawa, Ontario Suite 100 Canada EMail: avi@bridgewatersystems.com

Parviz Yegani Yegani Juniper EMail: pyegani@juniper.net

Kuntal Chowdhury Chowdhury Starent Networks 30 International Place, 3rd Floor Tewksbury, MA 01876 USA EMail: kchowdhury@starentnetworks.com

Hannes Tschofenig Tschofenig Nokia Siemens Networks Linnoitustie 6 Espoo, 02600 Finland Phone: +358 (50) 4871445 EMail: Hannes.Tschofenig@gmx.net URI: http://www.tschofenig.priv.at

Andreas Pashalidis Pashalidis K.U.Leuven, ESAT/SCD/COSIC Kasteelpark Arenberg 10, bus 2446 Leuven-Heverlee, B-3001 Belgium EMail: andreas.pashalidis@esat.kuleuven.be