Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended status: Best Current Practice
Expires: March 27, 2015

L. Morand, Ed.
Orange Labs
V. Fajardo
Fluke Networks
H. Tschofenig

September 23, 2014

Diameter Applications Design Guidelines
draft-ietf-dime-app-design-guide-28

Abstract

The Diameter base protocol provides facilities for protocol extensibility enabling to define new Diameter applications or modify existing applications. This document is a companion document to the Diameter Base protocol that further explains and clarifies the rules to extend Diameter. Furthermore, this document provides guidelines to Diameter application designers reusing/defining Diameter applications or creating generic Diameter extensions.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 27, 2015.

Copyright Notice

Table of Contents

1. Introduction

The Diameter base protocol [RFC6733] is intended to provide an Authentication, Authorization, and Accounting (AAA) framework for applications such as network access or IP mobility in both local and roaming situations. This protocol provides the ability for Diameter peers to exchange messages carrying data in the form of Attribute-Value Pairs (AVPs).

The Diameter base protocol provides facilities to extend Diameter (see Section 1.3 of [RFC6733]) to support new functionality. In the context of this document, extending Diameter means one of the following:

1. Addition of new functionality to an existing Diameter application without defining a new application.

2. Addition of new functionality to an existing Diameter application that requires the definition of a new application.

3. The definition of an entirely new Diameter application to offer functionality not supported by existing applications.

4. The definition of a new generic functionality that can be reused across different applications.

All of these choices are design decisions that can be done by any combination of reusing existing or defining new commands, AVPs or AVP values. However, application designers do not have complete freedom when making their design. A number of rules have been defined in [RFC6733] that place constraints on when an extension requires the allocation of a new Diameter application identifier or a new command code value. The objective of this document is the following:

- Clarify the Diameter extensibility rules as defined in the Diameter base protocol.

- Discuss design choices and provide guidelines when defining new applications.

- Present trade-off choices.

2. Terminology

This document reuses the terminology defined in [RFC6733]. Additionally, the following terms and acronyms are used in this application:

Application Extension of the Diameter base protocol [RFC6733] via the addition of new commands or AVPs. Each application is uniquely identified by an IANA-allocated application identifier value.

Command Diameter request or answer carrying AVPs between Diameter endpoints. Each command is uniquely identified by a IANA-allocated command code value and is described by a Command Code Format (CCF) for an application.

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 [RFC2119].

3. Overview

As designed, the Diameter base protocol [RFC6733] can be seen as a two-layer protocol. The lower layer is mainly responsible for managing connections between neighboring peers and for message routing. The upper layer is where the Diameter applications reside. This model is in line with a Diameter node having an application layer and a peer-to-peer delivery layer. The Diameter base protocol document defines the architecture and behavior of the message delivery layer and then provides the framework for designing Diameter applications on the application layer. This framework includes definitions of application sessions and accounting support (see Section 8 and Section 9 of [RFC6733]). Accordingly, a Diameter node is seen in this document as a single instance of a Diameter message delivery layer and one or more Diameter applications using it.

The Diameter base protocol is designed to be extensible and the principles are described in the Section 1.3 of [RFC6733]. As a summary, Diameter can be extended by:

1. Defining new AVP values

2. Creating new AVPs

3. Creating new commands

4. Creating new applications

As a main guiding principle, application designers SHOULD follow the following recommendation: "try to re-use as much as possible!". It will reduce the time to finalize specification writing, and it will lead to a smaller implementation effort as well as reduce the need for testing. In general, it is clever to avoid duplicate effort when possible.

However, re-use is not appropriate when the existing functionality does not fit the new requirement and/or the re-use leads to ambiguity.

The impact on extending existing applications can be categorized into two groups:

Minor Extension: Enhancing the functional scope of an existing application by the addition of optional features to support. Such enhancement has no backward compatibility issue with the existing application.

A typical example would be the definition of a new optional AVP for use in an existing command. Diameter implementations supporting the existing application but not the new AVP will simply ignore it, without consequences for the Diameter message handling, as described in [RFC6733]. The standardization effort will be fairly small.

Major Extension: Enhancing an application that requires the definition of a new Diameter application. Such enhancement causes backward compatibility issue with existing implementations supporting the application.

Typical examples would be the creation of a new command for providing functionality not supported by existing applications or the definition of a new AVP to be carried in an existing command with the M-bit set in the AVP flags (see Section 4.1 of [RFC6733] for definition of the "M-bit"). For such extension, a significant specification effort is required and a careful approach is recommended.

4. Reusing Existing Diameter Applications

An existing application may need to be enhanced to fulfill new requirements and these modifications can be at the command level and/or at the AVP level. The following sections describe the possible modifications that can be performed on existing applications and their related impact.

4.1. Adding a New Command

Adding a new command to an existing application is considered as a major extension and requires a new Diameter application to be defined, as stated in the Section 1.3.4 of [RFC6733]. The need for a new application is because a Diameter node that is not upgraded to support the new command(s) within the (existing) application would reject any unknown command with the protocol error DIAMETER_COMMAND_UNSUPPORTED and cause the failure of the transaction. The new application ensures that Diameter nodes only receive commands within the context of applications they support.

Adding a new command means either defining a completely new command or importing the command's Command Code Format (CCF) syntax from another application whereby the new application inherits some or all of the functionality of the application where the command came from. In the former case, the decision to create a new application is straightforward since this is typically a result of adding a new functionality that does not exist yet. For the latter, the decision to create a new application will depend on whether importing the command in a new application is more suitable than simply using the existing application as it is in conjunction with any other application.

An example considers the Diameter EAP application [RFC4072] and the Diameter Network Access Server application [RFC7155]. When network access authentication using EAP is required, the Diameter EAP commands (Diameter-EAP-Request/Diameter-EAP-Answer) are used; otherwise the Diameter Network Access Server application will be used. When the Diameter EAP application is used, the accounting exchanges defined in the Diameter Network Access Server may be used.

However, in general, it is difficult to come to a hard guideline, and so a case-by-case study of each application requirement should be applied. Before adding or importing a command, application designers should consider the following:

- Can the new functionality be fulfilled by creating a new command independent from any existing command? In this case, the

resulting new application and the existing application can work independent of, but cooperating with each other.

- Can the existing command be reused without major extensions and therefore without the need for the definition of a new application, e.g. new functionality introduced by the creation of new optional AVPs.

It is important to note that importing commands too liberally could result in a monolithic and hard to manage application supporting too many different features.

4.2. Deleting an Existing Command

Although this process is not typical, removing a command from an application requires a new Diameter application to be defined and then it is considered as a major extension. This is due to the fact that the reception of the deleted command would systematically result in a protocol error (i.e., DIAMETER_COMMAND_UNSUPPORTED).

It is unusual to delete an existing command from an application for the sake of deleting it or the functionality it represents. An exception might be if the intent of the deletion is to create a newer variance of the same application that is somehow simpler than the application initially specified.

4.3. Reusing Existing Commands

This section discusses rules in adding and/or deleting AVPs from an existing command of an existing application. The cases described in this section may not necessarily result in the creation of new applications.

From a historical point of view, it is worth to note that there was a strong recommendation to re-use existing commands in the [RFC3588] to prevent rapid depletion of code values available for vendor-specific commands. However, [RFC6733] has relaxed the allocation policy and enlarged the range of available code values for vendor-specific applications. Although reuse of existing commands is still RECOMMENDED, protocol designers can consider defining a new command when it provides a solution more suitable than the twisting of an existing command’s use and applications.

4.3.1. Adding AVPs to a Command

Based on the rules in [RFC6733], AVPs that are added to an existing command can be categorized into:

- Mandatory (to understand) AVPs. As defined in [RFC6733], these are AVPs with the M-bit flag set in this command, which means that a Diameter node receiving them is required to understand not only their values but also their semantics. Failure to do so will cause an message handling error: either a error message with the result-code set to DIAMETER_AVP_UNSUPPORTED if the AVP not understood in a request or a application specific error handling if the given AVP is in an answer.

- Optional (to understand) AVPs. As defined in [RFC6733], these are AVPs with the M-bit flag cleared in this command. A Diameter node receiving these AVPs can simply ignore them if it does not support them.

It is important to note that the definition given above are independent of whether these AVPs are required or optional in the command as specified by the command's Command Code Format (CCF) syntax [RFC6733].

NOTE: As stated in [RFC6733], the M-bit setting for a given AVP is relevant to an application and each command within that application that includes the AVP.

The rules are strict in the case where the AVPs to be added in an exiting command are mandatory to understand, i.e., they have the M-bit set. A mandatory AVP MUST NOT be added to an existing command without defining a new Diameter application, as stated in [RFC6733]. This falls into the "Major Extensions" category. Despite the clarity of the rule, ambiguity still arises when evaluating whether a new AVP being added should be mandatory to begin with. Application designers should consider the following questions when deciding about the M-bit for a new AVP:

- Would it be required for the receiving side to be able to process and understand the AVP and its content?

- Would the new AVPs change the state machine of the application?

- Would the presence of the new AVP lead to a different number of round-trips, effectively changing the state machine of the application?

- Would the new AVP be used to differentiate between old and new variances of the same application whereby the two variances are not backward compatible?

- Would the new AVP have duality in meaning, i.e., be used to carry application-related information as well as to indicate that the message is for a new application?

If the answer to at least one of the questions is "yes" then the M-bit MUST be set for the new AVP and a new Diameter application MUST be defined. This list of questions is non-exhaustive and other criteria MAY be taken into account in the decision process.

If application designers are instead contemplating the use of optional AVPs, i.e., with the M-bit cleared, there are still pitfalls that will cause interoperability problems and therefore must be avoided. Some examples of these pitfalls are :

- Use of optional AVPs with intersecting meaning. One AVP has partially the same usage and meaning as another AVP. The presence of both can lead to confusion.

- An optional AVPs with dual purpose, i.e., to carry application data as well as to indicate support for one or more features. This has a tendency to introduce interpretation issues.

- Adding one or more optional AVPs and indicating (usually within descriptive text for the command) that at least one of them has to be understood by the receiver of the command. This would be equivalent to adding a mandatory AVP, i.e., an AVP with the M-bit set, to the command.

4.3.2. Deleting AVPs from a Command

Application designers may want to reuse an existing command but some of the AVP present in the command’s CCF syntax specification may be irrelevant for the functionality foreseen to be supported by this command. It may be then tempting to delete those AVPs from the command.

The impacts of deleting an AVP from a command depends on its command code format specification and M-bit setting:

- Case 1: Deleting an AVP that is indicated as a required AVP (noted as {AVP}) in the command’s CCF syntax specification (regardless of the M-bit setting).

  In this case, a new command code and subsequently a new Diameter application MUST be specified.

- Case 2: Deleting an AVP, which has the M-bit set, and is indicated as optional AVP (noted as [AVP]) in the command CCF) in the command’s CCF syntax specification.

  In this case, no new command code has to be specified but the definition of a new Diameter application is REQUIRED.

- Case 3: Deleting an AVP, which has the M-bit cleared, and is indicated as [AVP] in the command's CCF syntax specification.

  In this case, the AVP can be deleted without consequences.

Application designers SHOULD attempt the reuse the command's CCF syntax specification without modification and simply ignore (but not delete) any optional AVP that will not be used. This is to maintain compatibility with existing applications that will not know about the new functionality as well as maintain the integrity of existing dictionaries.

4.3.3. Changing the Flags Setting of AVP in existing Commands

Although unusual, implementors may want to change the setting of the AVP flags a given AVP used in a command.

Into an existing command, a AVP that was initially defined as mandatory AVP to understand, i.e., an AVP with the M-bit flag set in the command, MAY be safely turned to an optional AVP, i.e., with the M-bit cleared. Any node supporting the existing application will still understand the AVP, whatever the setting of the M-bit. On the contrary, an AVP initially defined as an optional AVP to understand, i.e., an AVP with the M-bit flag cleared in the command, MUST NOT be changed into a mandatory AVP with the M-bit flag set without defining a new Diameter application. Setting the M-bit for an AVP that was defined as an optional AVP is equivalent to adding a new mandatory AVP to an existing command and the rules given in the section 4.3.1 apply.

All other AVP flags (V-bit, P-bit, reserved bits) MUST remain unchanged.

4.4. Reusing Existing AVPs

This section discusses rules in reusing existing AVP when reusing an existing command or defining a new command in a new application.

4.4.1. Setting of the AVP Flags

When reusing existing AVPs in a new application, application designers MUST specify the setting of the M-bit flag for a new Diameter application and, if necessary, for every command of the application that can carry these AVPs. In general, for AVPs defined outside of the Diameter base protocol, the characteristics of an AVP are tied to its role within a given application and the commands used in this application.

All other AVP flags (V-bit, P-bit, reserved bits) MUST remain unchanged.

4.4.2. Reuse of AVP of Type Enumerated

When reusing an AVP of type Enumerated in a command for a new application, it is RECOMMENDED to avoid modifying the set of valid values defined for this AVP. Modifying the set of Enumerated values includes adding a value or deprecating the use of a value defined initially for the AVP. Modifying the set of values will impact the application defining this AVP and all the applications using this AVP, causing potential interoperability issues: a value used by a peer that will not be recognized by all the nodes between the client and the server will cause an error response with the Result-Code AVP set to DIAMETER_INVALID_AVP_VALUE. When the full range of values defined for this Enumerated AVP is not suitable for the new application, it is RECOMMENDED to define a new AVP to avoid backwards compatibility issues with existing implementations.

5. Defining New Diameter Applications

5.1. Introduction

This section discusses the case where new applications have requirements that cannot be fulfilled by existing applications and would require definition of completely new commands, AVPs and/or AVP values. Typically, there is little ambiguity about the decision to create these types of applications. Some examples are the interfaces defined for the IP Multimedia Subsystem of 3GPP, e.g., Cx/Dx ([TS29.228] and [TS29.229]), Sh ([TS29.328] and [TS29.329]) etc.

Application designers SHOULD try to import existing AVPs and AVP values for any newly defined commands. In certain cases where accounting will be used, the models described in Section 5.10 SHOULD also be considered.

Additional considerations are described in the following sections.

5.2. Defining New Commands

As a general recommendation, commands SHOULD NOT be defined from scratch. It is instead RECOMMENDED to re-use an existing command offering similar functionality and use it as a starting point. Code re-use lead to a smaller implementation effort as well as reduce the need for testing.

Moreover, the new command’s CCF syntax specification SHOULD be carefully defined when considering applicability and extensibility of

the application. If most of the AVPs contained in the command are indicated as fixed or required, it might be difficult to reuse the same command and therefore the same application in a slightly changed environment. Defining a command with most of the AVPs indicated as optional is considered as a good design choice in many cases, despite the flexibility it introduces in the protocol. Protocol designers MUST clearly state the reasons why these optional AVPs might or might not be present and properly define the corresponding behavior of the Diameter nodes when these AVPs are absent from the command.

NOTE: As a hint for protocol designers, it is not sufficient to just look at the command's CCF syntax specification. It is also necessary to carefully read through the accompanying text in the specification.

In the same way, the CCF syntax specification SHOULD be defined such that it will be possible to add any arbitrary optional AVPs with the M-bit cleared (including vendor-specific AVPs) without modifying the application. For this purpose, "* [AVP]" SHOULD be added in the command's CCF, which allows the addition of any arbitrary number of optional AVPs as described in [RFC6733].

5.3. Use of Application-Id in a Message

When designing new applications, application designers SHOULD specify that the Application Id carried in all session-level messages is the Application Id of the application using those messages. This includes the session-level messages defined in Diameter base protocol, i.e., RAR/RAA, STR/STA, ASR/ASA and possibly ACR/ACA in the coupled accounting model, see Section 5.10. Some existing specifications do not adhere to this rule for historical reasons. However, this guidance SHOULD be followed by new applications to avoid routing problems.

When a new application has been allocated with a new Application Id and it also reuses existing commands with or without modifications, the commands SHOULD use the newly allocated Application Id in the header and in all relevant Application Id AVPs (Auth-Application-Id or Acct-Application-Id) present in the commands message body.

Additionally, application designers using Vendor-Specific-Application-Id AVP SHOULD NOT use the Vendor-Id AVP to further dissect or differentiate the vendor-specification Application Id. Diameter routing is not based on the Vendor-Id. As such, the Vendor-Id SHOULD NOT be used as an additional input for routing or delivery of messages. The Vendor-Id AVP is an informational AVP only and kept for backward compatibility reasons.

5.4. Application-Specific Session State Machines

Section 8 of [RFC6733] provides session state machines for authentication, authorization and accounting (AAA) services and these session state machines are not intended to cover behavior outside of AAA. If a new application cannot clearly be categorized into any of these AAA services, it is RECOMMENDED that the application defines its own session state machine. Support for server-initiated request is a clear example where an application-specific session state machine would be needed, for example, the Rw interface for ITU-T push model (cf.[Q.3303.3]).

5.5. Session-Id AVP and Session Management

Diameter applications are usually designed with the aim of managing user sessions (e.g., Diameter network access session (NASREQ) application [RFC4005]) or specific service access session (e.g., Diameter SIP application [RFC4740]). In the Diameter base protocol, session state is referenced using the Session-Id AVP. All Diameter messages that use the same Session-Id will be bound to the same session. Diameter-based session management also implies that both Diameter client and server (and potentially proxy agents along the path) maintain session state information.

However, some applications may not need to rely on the Session-Id to identify and manage sessions because other information can be used instead to correlate Diameter messages. Indeed, the User-Name AVP or any other specific AVP can be present in every Diameter message and used therefore for message correlation. Some applications might not require the notion of Diameter session concept at all. For such applications, the Auth-Session-State AVP is usually set to NO_STATE_MAINTAINED in all Diameter messages and these applications are therefore designed as a set of stand-alone transactions. Even if an explicit access session termination is required, application-specific commands are defined and used instead of the Session-Termination-Request/Answer (STR/STA) or Abort-Session-Request/Answer (ASR/ASA) defined in the Diameter base protocol [RFC6733]. In such a case, the Session-Id is not significant.

Based on these considerations, protocol designers should carefully appraise whether the Diameter application being defined relies on the session management specified in the Diameter base protocol:

- If it is, the Diameter command defined for the new application MUST include the Session-Id AVP defined in the Diameter base protocol [RFC6733] and the Session-Id AVP MUST be used for correlation of messages related to the same session. Guidance on

the use of the Auth-Session-State AVP is given in the Diameter base protocol [RFC6733].

- o Otherwise, because session management is not required or the application relies on its own session management mechanism, Diameter commands for the application need not include the Session-Id AVP. If any specific session management concept is supported by the application, the application documentation MUST clearly specify how the session is handled between client and server (and possibly Diameter agents in the path). Moreover, because the application is not maintaining session state at the Diameter base protocol level, the Auth-Session-State AVP MUST be included in all Diameter commands for the application and MUST be set to NO_STATE_MAINTAINED.

5.6. Use of Enumerated Type AVPs

The type Enumerated was initially defined to provide a list of valid values for an AVP with their respective interpretation described in the specification. For instance, AVPs of type Enumerated can be used to provide further information on the reason for the termination of a session or a specific action to perform upon the reception of the request.

As described in the section 4.4.2 above, defining an AVP of type Enumerated presents some limitations in term of extensibility and reusability. Indeed, the finite set of valid values defined at the definition of the AVP of type Enumerated cannot be modified in practice without causing backward compatibility issues with existing implementations. As a consequence, AVPs of Type Enumerated MUST NOT be extended by adding new values to support new capabilities. Diameter protocol designers SHOULD carefully consider before defining an Enumerated AVP whether the set of values will remain unchanged or new values may be required in a near future. If such extension is foreseen or cannot be avoided, it is RECOMMENED to rather define AVPs of type Unsigned32 or Unsigned64 in which the data field would contain an address space representing "values" that would have the same use of Enumerated values. Whereas only the initial values defined at the definition of the AVP of type Enumerated are valid as described in section 4.4.2, any value from the address space from 0 to 2^32 - 1 for AVPs of type Unsigned32 or from 0 to 2^64 - 1 for AVPs of type Unsigned64 is valid at the Diameter base protocol level and will not interoperability issues for intermediary nodes between clients and servers. Only clients and servers will be able to process the values at the application layer.

For illustration, an AVP describing possible access networks would be defined as follow:

Access-Network-Type AVP (XXX) is of type Unsigned32 and contains a 32-bit address space representing types of access networks. This application defines the following classes of access networks, all identified by the thousands digit in the decimal notation:

- 1xxx (Mobile Access Networks)
- 2xxx (Fixed Access Network)
- 3xxx (Wireless Access Networks)

Values that fall within the Mobile Access Networks category are used to inform a peer that a request has been sent for a user attached to a mobile access network. The following values are defined in this application:

1001: 3GPP-GERAN

The user is attached to a GSM EDGE Radio Access Network.

1002: 3GPP-UTRAN-FDD

The user is attached to a UMTS access network that uses frequency-division duplexing for duplexing.

Unlike Enumerated AVP, any new value can be added in the address space defined by this Unsigned32 AVP without modifying the definition of the AVP. There is therefore no risk of backward compatibility issue, especially when intermediate nodes may be present between Diameter endpoints.

In the same line, AVPs of type Enumerated are too often used as a simple Boolean flag, indicating for instance a specific permission or capability, and therefore only two values are defined, e.g., TRUE/FALSE, AUTORIZED/UNAUTHORIZED or SUPPORTED/UNSUPPORTED. This is a sub-optimal design since it limits the extensibility of the application: any new capability/permission would have to be supported by a new AVP or new Enumerated value of the already defined AVP, with the backward compatibility issues described above. Instead of using an Enumerated AVP for a Boolean flag, protocol designers SHOULD use AVPs of type Unsigned32 or Unsigned64 AVP in which the data field would be defined as bit mask whose bit settings are described in the relevant Diameter application specification. Such AVPs can be reused and extended without major impact on the Diameter application. The bit mask SHOULD leave room for future additions. Examples of AVPs that use bit masks are the Session-Binding AVP defined in [RFC6733] and the MIP6-Feature-Vector AVP defined in [RFC5447].

5.7. Application-Specific Message Routing

As described in [RFC6733], a Diameter request that needs to be sent to a home server serving a specific realm, but not to a specific server (such as the first request of a series of round trips), will contain a Destination-Realm AVP and no Destination-Host AVP.

For such a request, the message routing usually relies only on the Destination-Realm AVP and the Application Id present in the request message header. However, some applications may need to rely on the User-Name AVP or any other application-specific AVP present in the request to determine the final destination of a request, e.g., to find the target AAA server hosting the authorization information for a given user when multiple AAA servers are addressable in the realm.

In such a context, basic routing mechanisms described in [RFC6733] are not fully suitable, and additional application-level routing mechanisms MUST be described in the application documentation to provide such specific AVP-based routing. Such functionality will be basically hosted by an application-specific proxy agent that will be responsible for routing decisions based on the received specific AVPs.

Examples of such application-specific routing functions can be found in the Cx/Dx applications ([TS29.228] and [TS29.229]) of the 3GPP IP Multimedia Subsystem, in which the proxy agent (Subscriber Location Function aka SLF) uses specific application-level identities found in the request to determine the final destination of the message.

Whatever the criteria used to establish the routing path of the request, the routing of the answer MUST follow the reverse path of the request, as described in [RFC6733], with the answer being sent to the source of the received request, using transaction states and hop-by-hop identifier matching. This ensures that the Diameter Relay or Proxy agents in the request routing path will be able to release the transaction state upon receipt of the corresponding answer, avoiding unnecessary failover. Moreover, especially in roaming cases, proxy agents in the path must be able to apply local policies when receiving the answer from the server during authentication/ authorization and/or accounting procedures, and maintain up-to-date session state information by keeping track of all authorized active sessions. Therefore, application designers MUST NOT modify the answer-routing principles described in [RFC6733] when defining a new application.

5.8. Translation Agents

As defined in [RFC6733], a translation agent is a device that provides interworking between Diameter and another AAA protocol, such as RADIUS .

In the case of RADIUS, it was initially thought that defining the translation function would be straightforward by adopting few basic principles, e.g., by the use of a shared range of code values for RADIUS attributes and Diameter AVPs. Guidelines for implementing a RADIUS-Diameter translation agent were put into the Diameter NASREQ Application ([RFC4005]).

However, it was acknowledged that such translation mechanism was not so obvious and deeper protocol analysis was required to ensure efficient interworking between RADIUS and Diameter. Moreover, the interworking requirements depend on the functionalities provided by the Diameter application under specification, and a case-by-case analysis is required. As a consequence, all the material related to RADIUS-to-Diameter translation is removed from the new version of the Diameter NASREQ application specification [RFC7155], which deprecates the RFC4005 ([RFC4005]).

Therefore, protocol designers SHOULD NOT assume the availability of a "standard" Diameter-to-RADIUS gateways agent when planning to interoperate with the RADIUS infrastructure. They SHOULD specify the required translation mechanism along with the Diameter application, if needed. This recommendation applies for any kind of translation.

5.9. End-to-End Application Capabilities Exchange

Diameter applications can rely on optional AVPs to exchange application-specific capabilities and features. These AVPs can be exchanged on an end-to-end basis at the application layer. Examples of this can be found with the MIP6-Feature-Vector AVP in [RFC5447] and the QoS-Capability AVP in [RFC5777].

End-to-end capabilities AVPs can be added as optional AVPs with the M-bit cleared to existing applications to announce support of new functionality. Receivers that do not understand these AVPs or the AVP values can simply ignore them, as stated in [RFC6733]. When supported, receivers of these AVPs can discover the additional functionality supported by the Diameter end-point originating the request and behave accordingly when processing the request. Senders of these AVPs can safely assume the receiving end-point does not support any functionality carried by the AVP if it is not present in corresponding response. This is useful in cases where deployment

choices are offered, and the generic design can be made available for a number of applications.

When used in a new application, these end-to-end capabilities AVPs SHOULD be added as optional AVP into the CCF of the commands used by the new application. Protocol designers SHOULD clearly specify this end-to-end capabilities exchange and the corresponding behaviour of the Diameter nodes supporting the application.

It is also important to note that this end-to-end capabilities exchange relying on the use of optional AVPs is not meant as a generic mechanism to support extensibility of Diameter applications with arbitrary functionality. When the added features drastically change the Diameter application or when Diameter agents must be upgraded to support the new features, a new application SHOULD be defined, as recommended in [RFC6733].

5.10. Diameter Accounting Support

Accounting can be treated as an auxiliary application that is used in support of other applications. In most cases, accounting support is required when defining new applications. This document provides two possible models for using accounting:

Split Accounting Model:

In this model, the accounting messages will use the Diameter base accounting Application Id (value of 3). The design implication for this is that the accounting is treated as an independent application, especially for Diameter routing. This means that accounting commands emanating from an application may be routed separately from the rest of the other application messages. This may also imply that the messages end up in a central accounting server. A split accounting model is a good design choice when:

* The application itself does not define its own accounting commands.

* The overall system architecture permits the use of centralized accounting for one or more Diameter applications.

Centralizing accounting may have advantages but there are also drawbacks. The model assumes that the accounting server can differentiate received accounting messages. Since the received accounting messages can be for any application and/or service, the accounting server MUST have a method to match accounting messages with applications and/or services being accounted for. This may

mean defining new AVPs, checking the presence, absence or contents of existing AVPs, or checking the contents of the accounting record itself. One of these means could be to insert into the request sent to the accounting server an Auth-Application-Id AVP containing the identifier of the application for which the accounting request is sent. But in general, there is no clean and generic scheme for sorting these messages. Therefore, this model SHOULD NOT be used when all received accounting messages cannot be clearly identified and sorted. For most cases, the use of Coupled Accounting Model is RECOMMENDED.

Coupled Accounting Model:

In this model, the accounting messages will use the Application Id of the application using the accounting service. The design implication for this is that the accounting messages are tightly coupled with the application itself; meaning that accounting messages will be routed like the other application messages. It would then be the responsibility of the application server (application entity receiving the ACR message) to send the accounting records carried by the accounting messages to the proper accounting server. The application server is also responsible for formulating a proper response (ACA). A coupled accounting model is a good design choice when:

* The system architecture or deployment does not provide an accounting server that supports Diameter. Consequently, the application server MUST be provisioned to use a different protocol to access the accounting server, e.g., via LDAP, SOAP etc. This case includes the support of older accounting systems that are not Diameter aware.

* The system architecture or deployment requires that the accounting service for the specific application should be handled by the application itself.

In all cases above, there will generally be no direct Diameter access to the accounting server.

These models provide a basis for using accounting messages. Application designers may obviously deviate from these models provided that the factors being addressed here have also been taken into account. As a general recommendation, application designers SHOULD NOT define a new set of commands to carry application-specific accounting records.

5.11. Diameter Security Mechanisms

As specified in [RFC6733], the Diameter message exchange SHOULD be secured between neighboring Diameter peers using TLS/TCP or DTLS/SCTP. However, IPsec MAY also be deployed to secure communication between Diameter peers. When IPsec is used instead of TLS or DTLS, the following recommendations apply.

IPsec ESP [RFC4301] in transport mode with non-null encryption and authentication algorithms MUST be used to provide per-packet authentication, integrity protection and confidentiality, and support the replay protection mechanisms of IPsec. IKEv2 [RFC5996] SHOULD be used for performing mutual authentication and for establishing and maintaining security associations (SAs).

IKEv1 [RFC2409] was used with RFC 3588 [RFC3588] and for easier migration from IKEv1 based implementations both RSA digital signatures and pre-shared keys SHOULD be supported in IKEv2. However, if IKEv1 is used, implementers SHOULD follow the guidelines given in Section 13.1 of RFC 3588 [RFC3588].

6. Defining Generic Diameter Extensions

Generic Diameter extensions are AVPs, commands or applications that are designed to support other Diameter applications. They are auxiliary applications meant to improve or enhance the Diameter protocol itself or Diameter applications/functionality. Some examples include the extensions to support realm-based redirection of Diameter requests (see [RFC7075]), convey a specific set of priority parameters influencing the distribution of resources (see [RFC6735]), and the support for QoS AVPs (see [RFC5777]).

Since generic extensions may cover many aspects of Diameter and Diameter applications, it is not possible to enumerate all scenarios. However, some of the most common considerations are as follows:

Backward Compatibility:

> When defining generic extensions designed to be supported by existing Diameter applications, protocol designers MUST consider the potential impacts of the introduction of the new extension on the behavior of node that would not be yet upgraded to support/understand this new extension. Designers MUST also ensure that new extensions do not break expected message delivery layer behavior.

Forward Compatibility:

Protocol designers MUST ensure that their design will not introduce undue restrictions for future applications.

Trade-off in Signaling:

Designers may have to choose between the use of optional AVPs piggybacked onto existing commands versus defining new commands and applications. Optional AVPs are simpler to implement and may not need changes to existing applications. However, this ties the sending of extension data to the application's transmission of a message. This has consequences if the application and the extensions have different timing requirements. The use of commands and applications solves this issue, but the trade-off is the additional complexity of defining and deploying a new application. It is left up to the designer to find a good balance among these trade-offs based on the requirements of the extension.

In practice, generic extensions often use optional AVPs because they are simple and non-intrusive to the application that would carry them. Peers that do not support the generic extensions need not understand nor recognize these optional AVPs. However, it is RECOMMENDED that the authors of the extension specify the context or usage of the optional AVPs. As an example, in the case that the AVP can be used only by a specific set of applications then the specification MUST enumerate these applications and the scenarios when the optional AVPs will be used. In the case where the optional AVPs can be carried by any application, it should be sufficient to specify such a use case and perhaps provide specific examples of applications using them.

In most cases, these optional AVPs piggybacked by applications would be defined as a Grouped AVP and it would encapsulate all the functionality of the generic extension. In practice, it is not uncommon that the Grouped AVP will encapsulate an existing AVP that has previously been defined as mandatory ('M'-bit set) e.g., 3GPP IMS Cx/Dx interfaces ([TS29.228] and [TS29.229]).

7. Guidelines for Registrations of Diameter Values

As summarized in the Section 3 of this document and further described in the Section 1.3 of [RFC6733], there are four main ways to extend Diameter. The process for defining new functionality slightly varies based on the different extensions. This section provides protocol designers with some guidance regarding the definition of values for possible Diameter extensions and the necessary interaction with IANA to register the new functionality.

a. Defining new AVP values

The specifications defining AVPs and AVP values MUST provide guidance for defining new values and the corresponding policy for adding these values. For example, the RFC 5777 [RFC5777] defines the Treatment-Action AVP which contains a list of valid values corresponding to pre-defined actions (drop, shape, mark, permit). This set of values can be extended following the Specification Required policy defined in [RFC5226]. As a second example, the Diameter base specification [RFC6733] defines the Result-Code AVP that contains a 32-bit address space used to identity possible errors. According to the Section 11.3.2 of [RFC6733], new values can be assigned by IANA via an IETF Review process [RFC5226].

b. Creating new AVPs

Two different types of AVP Codes namespaces can be used to create a new AVPs:

* IETF AVP Codes namespace;

* Vendor-specific AVP Codes namespace.

In the latter case, a vendor needs to be first assigned by IANA with a private enterprise number, which can be used within the Vendor-Id field of the vendor-specific AVP. This enterprise number delimits a private namespace in which the vendor is responsible for vendor-specific AVP code value assignment. The absence of a Vendor-Id or a Vendor-Id value of zero (0) in the AVP header identifies standard AVPs from the IETF AVP Codes namespace managed by IANA. The allocation of code values from the IANA-managed namespace is conditioned by an Expert Review of the specification defining the AVPs or an IETF review if a block of AVPs needs to be assigned. Moreover, the remaining bits of the AVP Flags field of the AVP header are also assigned via Standard Action if the creation of new AVP Flags is desired.

c. Creating new commands

Unlike the AVP Code namespace, the Command Code namespace is flat but the range of values is subdivided into three chunks with distinct IANA registration policies:

* A range of standard Command Code values that are allocated via IETF review;

* A range of vendor-specific Command Code values that are allocated on a First-Come/First-Served basis;

* A range of values reserved only for experimental and testing purposes.

As for AVP Flags, the remaining bits of the Command Flags field of the Diameter header are also assigned via a Standards Action to create new Command Flags if required.

d. Creating new applications

Similarly to the Command Code namespace, the Application-Id namespace is flat but divided into two distinct ranges:

* A range of values reserved for standard Application-Ids allocated after Expert Review of the specification defining the standard application;

* A range for values for vendor specific applications, allocated by IANA on a First-Come/First-Serve basis.

The IANA AAA parameters page can be found at http://www.iana.org/assignments/aaa-parameters and the enterprise number IANA page is available at http://www.iana.org/assignments/enterprise-numbers. More details on the policies followed by IANA for namespace management (e.g. First-Come/First-Served, Expert Review, IETF Review, etc.) can be found in [RFC5226].

NOTE:
When the same functionality/extension is used by more than one vendor, it is RECOMMENDED to define a standard extension. Moreover, a vendor-specific extension SHOULD be registered to avoid interoperability issues in the same network. With this aim, the registration policy of vendor-specific extension has been simplified with the publication of [RFC6733] and the namespace reserved for vendor-specific extensions is large enough to avoid exhaustion.

8. IANA Considerations

This document does not require actions by IANA.

9. Security Considerations

This document provides guidelines and considerations for extending Diameter and Diameter applications. Although such an extension may be related to a security functionality, the document does not explicitly give additional guidance on enhancing Diameter with respect to security. However, as a general guideline, it is recommended that any Diameter extension SHOULD NOT break the security

concept given in the [RFC6733]. In particular, it is reminded here that any command defined or reused in a new Diameter application SHOULD be secured by using TLS [RFC5246] or DTLS/SCTP [RFC6083] and MUST NOT be used without one of TLS, DTLS, or IPsec [RFC4301]. When defining a new Diameter extension, any possible impact of the existing security principles described in the [RFC6733] MUST be carefully appraised and documented in the Diameter application specification.

10. Contributors

The content of this document was influenced by a design team created to revisit the Diameter extensibility rules. The team was formed in February 2008 and finished its work in June 2008. Except the authors, the design team members were:

- Avi Lior
- Glen Zorn
- Jari Arkko
- Jouni Korhonen
- Mark Jones
- Tolga Asveren
- Glenn McGregor
- Dave Frascone

We would like to thank Tolga Asveren, Glenn McGregor, and John Loughney for their contributions as co-authors to earlier versions of this document.

11. Acknowledgments

We greatly appreciate the insight provided by Diameter implementers who have highlighted the issues and concerns being addressed by this document. The authors would also like to thank Jean Mahoney, Ben Campbell, Sebastien Decugis and Benoit Claise for their invaluable detailed reviews and comments on this document.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

12.2. Informative References

[Q.3303.3] 3rd Generation Partnership Project, "ITU-T Recommendation Q.3303.3, "Resource control protocol no. 3 (rcp3): Protocol at the Rw interface between the Policy Decision Physical Entity (PD-PE) and the Policy Enforcement Physical Entity (PE-PE): Diameter"", 2008.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC4005] Calhoun, P., Zorn, G., Spence, D., and D. Mitton, "Diameter Network Access Server Application", RFC 4005, August 2005.

[RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4740] Garcia-Martin, M., Belinchon, M., Pallares-Lopez, M., Canales-Valenzuela, C., and K. Tammi, "Diameter Session Initiation Protocol (SIP) Application", RFC 4740, November 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5447] Korhonen, J., Bournelle, J., Tschofenig, H., Perkins, C., and K. Chowdhury, "Diameter Mobile IPv6: Support for Network Access Server to Diameter Server Interaction", RFC 5447, February 2009.

[RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Jones, M., and A. Lior, "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, February 2010.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

[RFC6083] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)", RFC 6083, January 2011.

[RFC6735] Carlberg, K. and T. Taylor, "Diameter Priority Attribute-Value Pairs", RFC 6735, October 2012.

[RFC7075] Tsou, T., Hao, R., and T. Taylor, "Realm-Based Redirection In Diameter", RFC 7075, November 2013.

[RFC7155] Zorn, G., "Diameter Network Access Server Application", RFC 7155, April 2014.

[TS29.228]
3rd Generation Partnership Project, "3GPP TS 29.228; Technical Specification Group Core Network and Terminals; IP Multimedia (IM) Subsystem Cx and Dx Interfaces; Signalling flows and message contents", <http://www.3gpp.org/ftp/Specs/html-info/29272.htm>.

[TS29.229]
3rd Generation Partnership Project, "3GPP TS 29.229; Technical Specification Group Core Network and Terminals; Cx and Dx interfaces based on the Diameter protocol; Protocol details", <http://www.3gpp.org/ftp/Specs/html-info/29229.htm>.

[TS29.328]
3rd Generation Partnership Project, "3GPP TS 29.328; Technical Specification Group Core Network and Terminals; IP Multimedia (IM) Subsystem Sh interface; signalling flows and message content", <http://www.3gpp.org/ftp/Specs/html-info/29328.htm>.

[TS29.329]
            3rd Generation Partnership Project, "3GPP TS 29.329;
            Technical Specification Group Core Network and Terminals;
            Sh Interface based on the Diameter protocol; Protocol
            details",
            <http://www.3gpp.org/ftp/Specs/html-info/29329.htm>.

Authors' Addresses

Lionel Morand (editor)
Orange Labs
38/40 rue du General Leclerc
Issy-Les-Moulineaux Cedex 9 92794
France

Phone: +33145296257
Email: lionel.morand@orange.com

Victor Fajardo
Fluke Networks

Email: vf0213@gmail.com

Hannes Tschofenig
Hall in Tirol 6060
Austria

Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 27, 2011

K. Jiao
Huawei
G. Zorn
Network Zen
October 24, 2010

The Diameter Capabilities Update Application
draft-ietf-dime-capablities-update-07

Abstract

This document defines a new Diameter application and associated command codes. The Capabilities Update application is intended to allow the dynamic update of certain Diameter peer capabilities while the peer-to-peer connection is in the open state.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 27, 2011.

Table of Contents

1. Introduction

Capabilities exchange is an important component of the Diameter Base Protocol [I-D.ietf-dime-rfc3588bis], allowing peers to exchange identities and Diameter capabilities (protocol version number, supported Diameter applications, security mechanisms, etc.). As defined in RFC 3588, however, the capabilities exchange process takes place only once, at the inception of a transport connection between a given pair of peers. Therefore, if a peer's capabilities change (due to software update, for example), the existing connection(s) must be torn down (along with all of the associated user sessions) and restarted before the modified capabilities can be advertised.

This document defines a new Diameter application intended to allow the dynamic update of a subset of Diameter peer capabilities over an existing connection. Because the Capabilities Update application specified herein operates over an existing transport connection, modification of certain capabilities is prohibited. Specifically, modifying the security mechanism in use is not allowed; if the security method used between a pair of peers is changed the affected connection MUST be restarted.

2. Specification of Requirements

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].

3. Diameter Protocol Considerations

This section details the relationship of the Diameter Capabilities Update application to the Diameter Base Protocol.

This document specifies Diameter Application-ID <TBD1>. Diameter nodes conforming to this specification MUST advertise support by including the value <TBD1> in the Auth-Application-Id of the Capabilities-Exchange-Request (CER) and Capabilities-Exchange-Answer (CEA) commands [I-D.ietf-dime-rfc3588bis].

4. Capabilities Update

When the capabilities of a Diameter node conforming to this specification change, it MUST notify all of the nodes with which it has an open transport connection and which have also advertised support for the Capabilities Update application using the

Capabilities-Update-Request (CUR) message (Section 4.1.1). This message allows the update of a peer's capabilities (supported Diameter applications, etc.).

A Diameter node only issues a given command to those peers that have advertised support for the Diameter application that defines the command; a Diameter node must cache the supported applications in order to ensure that unrecognized commands and/or Attribute-Value Pairs (AVPs) are not unnecessarily sent to a peer.

The receiver of the CUR MUST determine common applications by computing the intersection of its own set of supported Application Id against all of the application identifier AVPs (Auth-Application-Id, Acct-Application-Id and Vendor-Specific- Application-Id) present in the CUR. The value of the Vendor-Id AVP in the Vendor-Specific-Application-Id MUST NOT be used during computation.

If the receiver of a CUR does not have any applications in common with the sender then it MUST return a Capabilities-Update-Answer (CUA) (Section 4.1.2) with the Result-Code AVP set to DIAMETER_NO_COMMON_APPLICATION [I-D.ietf-dime-rfc3588bis], and SHOULD disconnect the transport layer connection; however, if active sessions are using the connection, peers MAY delay disconnection until the sessions can be redirected or gracefully terminated. Note that receiving a CUA from a peer advertising itself as a Relay (see [I-D.ietf-dime-rfc3588bis], Section 2.4) MUST be interpreted as having common applications with the peer.

As for CER/CEA messages, the CUR and CUA messages MUST NOT be proxied, redirected or relayed.

Even though the CUR/CUA messages cannot be proxied, it is still possible for an upstream agent to receive a message for which there are no peers available to handle the application that corresponds to the Command-Code. This could happen if, for example, the peers are too busy or down. In such instances, the 'E' bit MUST be set in the answer message with the Result-Code AVP set to DIAMETER_UNABLE_TO_DELIVER to inform the downstream peer to take action (e.g., re-routing requests to an alternate peer).

4.1. Command-Code Values

This section defines Command-Code [I-D.ietf-dime-rfc3588bis] values that MUST be supported by all Diameter implementations conforming to this specification. The following Command Codes are defined (using modified ABNF [I-D.ietf-dime-rfc3588bis]) in this document: Capabilities-Update-Request (CUR, Section 4.1.1) and Capabilities-Update-Answer (CUA, Section 4.1.2).

4.1.1. Capabilities-Update-Request

The Capabilities-Update-Request (CUR), indicated by the Command-Code set to <CUCC> and the Command Flags' 'R' bit set, is sent to update local capabilities. Upon detection of a transport failure, this message MUST NOT be sent to an alternate peer.

When Diameter is run over the Stream Control Transmission Protocol [RFC4960], which allows connections to span multiple interfaces and multiple IP addresses, the Capabilities-Update-Request message MUST contain one Host-IP-Address AVP for each potential IP address that may be locally used when transmitting Diameter messages.

Message Format

```
<CUR> ::= < Diameter Header: <CUCC>, REQ >
          { Origin-Host }
          { Origin-Realm }
        1* { Host-IP-Address }
          { Vendor-Id }
          { Product-Name }
          [ Origin-State-Id ]
         * [ Supported-Vendor-Id ]
         * [ Auth-Application-Id ]
         * [ Acct-Application-Id ]
         * [ Vendor-Specific-Application-Id ]
          [ Firmware-Revision ]
         * [ AVP ]
```

4.1.2. Capabilities-Update-Answer

The Capabilities-Update-Answer, indicated by the Command-Code set to <CUCC> and the Command Flags' 'R' bit cleared, is sent in response to a CUR message.

Message Format

```
<CUA> ::= < Diameter Header: <CUCC> >
            { Origin-Host }
            { Origin-Realm }
            { Result-Code }
            [ Error-Message ]
           * [ AVP ]
```

5. Security Considerations

The security considerations applicable to the Diameter Base Protocol [I-D.ietf-dime-rfc3588bis] are also applicable to this document.

6. IANA Considerations

This section explains the criteria to be used by the IANA for assignment of numbers within namespaces used within this document.

6.1. Application Identifier

This specification assigns the value <TBD1> from the Application Identifiers namespace [I-D.ietf-dime-rfc3588bis]. See Section 3 for the assignment of the namespace in this specification.

6.2. Command Codes

This specification assigns the value <CUCC> from the Command Codes namespace [I-D.ietf-dime-rfc3588bis]. See Section 4.1 for the assignment of the namespace in this specification.

7. Contributors

This document is based upon work done by Tina Tsou.

8. Acknowledgements

Thanks to Sebastien Decugis, Niklas Neumann, Subash Comerica, Lionel Morand, Dan Romascanu, Dan Harkins and Ravi for helpful review and discussion.

9. References

9.1. Normative References

[I-D.ietf-dime-rfc3588bis]
Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", draft-ietf-dime-rfc3588bis-25 (work in progress), September 2010.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

Authors' Addresses

Jiao Kang
Huawei Technologies
Section B1, Huawei Industrial Base
Bantian, Longgang District
Shenzhen 518129
P.R. China

Phone: +86 755 2878-6690
Email: kangjiao@huawei.com

Glen Zorn
Network Zen
227/358 Thanon Sanphawut
Bang Na, Bangkok 10260
Thailand

Phone: +66 (0) 87-040-4617
Email: gwz@net-zen.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 16, 2011

G. Zorn
Network Zen
S. Comerica
Juniper Networks
June 14, 2011

Diameter Base Protocol MIB
draft-ietf-dime-diameter-base-protocol-mib-06.txt

Abstract

Along with providing support for certain basic authentication, authorization and accounting functions, the Diameter protocol is designed to provide a framework for AAA applications.

This document defines the Management Information Base (MIB) module which describes the minimum set of objects needed to manage an implementation of the Diameter protocol.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 16, 2011.

Copyright Notice

Table of Contents

1. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58 ([RFC2578], [RFC2579], [RFC2580]). In particular, it describes managed objects used for managing the base Diameter protocol [I-D.ietf-dime-rfc3588bis].

2. Requirements Language

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].

3. Overview

This MIB defines objects supporting the management of the Diameter base protocol as defined in RFC 3588 [I-D.ietf-dime-rfc3588bis]. Objects related to Diameter applications are defined in separate documents.

4. Diameter Base Protocol MIB Definitions

```
DIAMETER-BASE-PROTOCOL-MIB DEFINITIONS ::= BEGIN

IMPORTS
      InetAddressType,
      InetAddress
            FROM INET-ADDRESS-MIB -- [RFC4001]
      MODULE-IDENTITY,
      OBJECT-TYPE,
      NOTIFICATION-TYPE,
      Integer32,
      Counter32,
      Unsigned32,
      Gauge32,
```

```
      TimeTicks
            FROM SNMPv2-SMI -- [RFC2578]
      SnmpAdminString
            FROM SNMP-FRAMEWORK-MIB -- [RFC3411]
      NOTIFICATION-GROUP,
      MODULE-COMPLIANCE,
      OBJECT-GROUP
            FROM SNMPv2-CONF -- [RFC2580]
      RowStatus,
      TruthValue,
      StorageType
            FROM SNMPv2-TC; -- [RFC2579]


diameterBaseProtocolMIB MODULE-IDENTITY
      LAST-UPDATED "201105040000Z" -- 4 May 2011
      ORGANIZATION "IETF dime Working Group."
      CONTACT-INFO
          "Glen Zorn
           Network Zen
           227/358 Thanon Sanphawut
           Bang Na, Bangkok 10260
           Thailand
           Email: gwz@net-zen.net"
       DESCRIPTION
           "The MIB module for entities implementing the
           Diameter Base Protocol.

           Copyright (c) 2011 IETF Trust and the persons identified
           as the document authors.  All rights reserved.

           The initial version of this MIB module was published
           in RFC yyyy; for full legal notices see the RFC itself.
           Supplementary information may be available on
           http://www.ietf.org/copyrights/ianamib.html."

-- RFC Ed.: replace yyyy with actual RFC number and remove this note

      REVISION "201105040000Z" -- 4 May 2011
      DESCRIPTION "Initial version as published in RFC yyyy"
-- RFC Ed.: replace yyyy with actual RFC number and remove this note
      ::= { mib-2 119 }  -- Experimental value assigned by IANA.

-- Top-Level Components of this MIB.

diameterBaseNotifications  OBJECT IDENTIFIER ::=
                                 { diameterBaseProtocolMIB 0 }
diameterBaseObjects        OBJECT IDENTIFIER ::=
```

```
                                    { diameterBaseProtocolMIB 1 }
diameterBaseConform        OBJECT IDENTIFIER ::=
                                    { diameterBaseProtocolMIB 2 }

dbpLocalCfgs    OBJECT IDENTIFIER ::= { diameterBaseObjects 1 }
dbpLocalStats   OBJECT IDENTIFIER ::= { diameterBaseObjects 2 }
dbpPeerCfgs     OBJECT IDENTIFIER ::= { diameterBaseObjects 3 }
dbpPeerStats    OBJECT IDENTIFIER ::= { diameterBaseObjects 4 }
dbpRealmCfgs    OBJECT IDENTIFIER ::= { diameterBaseObjects 5 }
dbpRealmStats   OBJECT IDENTIFIER ::= { diameterBaseObjects 6 }
dbpNotifCfgs    OBJECT IDENTIFIER ::= { diameterBaseObjects 7 }


-- Textual Conventions

ServiceType ::= TEXTUAL-CONVENTION
    DISPLAY-HINT  "d"
    STATUS        current
    DESCRIPTION
          "An enumerated value which provides an indication of the
          type of services supported for each Diameter application:
          accounting, authentication or both.

     SYNTAX        INTEGER { acct(1),
                             auth(2),
                             both(3) }
    }



-- Protocol Error Notifications

dbpProtocolErrorNotifEnabled OBJECT-TYPE
       SYNTAX      TruthValue
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
             "Setting the value of this object to True(1)
              enables the dbpProtocolErrorNotif notification.
              The value persists across resets."
       DEFVAL      {false}
       ::= { dbpNotifCfgs 1 }

dbpProtocolErrorNotif NOTIFICATION-TYPE
       OBJECTS {
                  dbpPeerId,
                  dbpPerPeerStatsProtocolErrors
                }
```

```
       STATUS     current
       DESCRIPTION
             "An dbpProtocolError Notification is sent when both the
              following conditions are true:
              1) the value of dbpProtocolErrorNotifEnabled is True(1)
              2) the value of dbpPerPeerStatsProtocolErrors changes
              An agent must not generate more than one
              dbpProtocolError 'notification event' in a five second
              period, where a 'notification event' is the
              transmission of a single Notification PDU to a list of
              Notification destinations.
              If additional protocol errors occur within the
              five second 'throttling' period, then these
              notification events should be suppressed by the agent.
              An NMS should periodically check the value of
              dbpPerPeerStatsProtocolErrors to detect any missed
              dbpProtocolError notification events, e.g. due to
              throttling or transmission loss."
      ::= { diameterBaseNotifications 1 }


-- Transient Error Notifications

dbpTransientFailureNotifEnabled OBJECT-TYPE
       SYNTAX      TruthValue
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
             "Setting the value of this object to True(1)
              enables the dbpTransientFailure Notification.
              The value persists across resets."
       ::= { dbpNotifCfgs 2 }

dbpTransientFailureNotif NOTIFICATION-TYPE
       OBJECTS  {
                   dbpPeerId,
                   dbpPerPeerStatsTransientFailures
                }
       STATUS     current
       DESCRIPTION
             "An dbpTransientFailure Notification is sent when both
              the following conditions are true:
              1) the value of dbpTransientFailureNotifEnabled
                 is True(1)
              2) the value of dbpPerPeerStatsTransientFailures
                 changes
              An agent must not generate more than one
              dbpTransientFailure 'notification event' in a five
```

```
              second period, where a 'notification event' is the
              transmission of a single notification PDU to a list
              of notification destinations.
              If additional transient failures occur
              within the five second 'throttling' period, then
              these notification events should be suppressed
              by the agent.
              An NMS should periodically check the value of
              dbpPerPeerStatsTransientFailures to detect any
              missed dbpTransientFailure notification events,
              e.g. due to throttling or transmission loss."
      ::= { diameterBaseNotifications 2 }


-- Permanent Failure Notifications

dbpPermanentFailureNotifEnabled OBJECT-TYPE
       SYNTAX      TruthValue
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
             "Setting the value of this object to True(1)
              enables the dbpPermanentFailure notification.
              The value persists across resets."
       DEFVAL    { false }
       ::= { dbpNotifCfgs 3 }

dbpPermanentFailureNotif NOTIFICATION-TYPE
       OBJECTS  {
                   dbpPeerId,
                   dbpPerPeerStatsPermanentFailures
                 }
       STATUS     current
       DESCRIPTION
             "An dbpPermanentFailure notification is sent when
              both the following conditions are true:
              1) the value of dbpPermanentFailureNotifEnabled
                 is True(1)
              2) the value of dbpPerPeerStatsPermanentFailures
                 changes
              An agent must not generate more than one
              dbpPermanentFailure 'notification event' in a five
              second period, where a 'notification event' is the
              transmission of a single notification PDU to a list
              of notification destinations.
              If additional permanent failures occur within the
              five second 'throttling' period, then these
              trap-events should be suppressed by the agent.
```

```
                An NMS should periodically check the value of
                dbpPerPeerStatsPermanentFailures to detect
                any missed dbpPermanentFailure trap-events,
                e.g. due to throttling or transmission loss."
        ::= { diameterBaseNotifications 3 }


-- Connection Down Notifs

dbpPeerConnectionDownNotifEnabled OBJECT-TYPE
        SYNTAX      TruthValue
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
              "Setting the value of this object to True(1)
               enables the dbpPeerConnectionDown notification.
               The value persists across resets."
        DEFVAL      { false }
        ::= { dbpNotifCfgs 4 }

dbpPeerConnectionDownNotif NOTIFICATION-TYPE
        OBJECTS  {
                    dbpLocalId,
                    dbpPeerId
                 }
        STATUS     current
        DESCRIPTION
              "An dbpPeerConnectionDown notification is sent when
               both the following conditions are true:
               1) the value of dbpPeerConnectionDownNotifEnabled is
                  True(1)
               2) dbpPerPeerStatsState changes to closed(1)
               An agent must not generate more than one
               dbpPeerConnectionDown
               'notification event' in a five second period, where a
               'notification event' is the transmission of a single
               notification PDU to a list of notification
               destinations.
               If additional 'transport down' events occur within the
               five second 'throttling' period, then these trap-events
               should be suppressed by the agent."
        ::= { diameterBaseNotifications 4 }


-- Connection Up Notifications

dbpPeerConnectionUpNotifEnabled OBJECT-TYPE
        SYNTAX      TruthValue
```

```
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
             "Setting the value of this object to True(1)
              enables the dbpPeerConnectionUp notification.
              The value persists across resets."
       DEFVAL      { false }
       ::= { dbpNotifCfgs 5 }

dbpPeerConnectionUpNotif NOTIFICATION-TYPE
       OBJECTS  {
                   dbpLocalId,
                   dbpPeerId
                 }
       STATUS     current
       DESCRIPTION
             "An dbpPeerConnectionUp notification is sent
              when both thefollowing conditions are true:
              1) the value of dbpPeerConnectionUpNotifEnabled is
                 True(1)
              2) the value of dbpPerPeerStatsState changes to
                 either rOpen(6)or iOpen(7)
              An agent must not generate
              more than one dbpPeerConnectionUp
              'notification event' in a
              five second period, where a 'notification event' is the
              transmission of a single notification PDU to a
              list of notification destinations.
              If additional 'connection up' events
              occur within the five second 'throttling' period,
              then these trap-events should be suppressed by the
              agent."
      ::= { diameterBaseNotifications 5 }


-- Local Configs

dbpLocalId OBJECT-TYPE
       SYNTAX     SnmpAdminString
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The implementation-specific identification
              string for the Diameter software in use on
              the system; for example: 'diameterd'"
       ::= { dbpLocalCfgs 1 }

dbpLocalTcpListenPort OBJECT-TYPE
```

```
       SYNTAX      Unsigned32 (1..65535)
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
             "Diameter TCP 'listen' port."
       ::= { dbpLocalCfgs 3 }

dbpLocalSctpListenPort OBJECT-TYPE
       SYNTAX      Unsigned32 (1..65535)
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
             "Diameter SCTP 'listen' port."
       ::= { dbpLocalCfgs 4 }

dbpLocalOriginHost OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION     "This object represents the host name of
                    the local peer.
                    The value persists across resets."
    DEFVAL          { "" }
    ::= { dbpLocalCfgs 5 }

dbpLocalRealm OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION     "This object represents the Local Realm Name."
    DEFVAL          { "" }
    ::= { dbpLocalCfgs 6 }


dbpLocalStatsTotalMessagesIn OBJECT-TYPE
       SYNTAX      Counter32 UNITS "messages"
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
             "The total number of Diameter Base Protocol
              messages received since the last reset."
       ::= { dbpLocalStats 1 }

dbpLocalStatsTotalMessagesOut OBJECT-TYPE
       SYNTAX      Counter32 UNITS "messages"
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
```

```
              "The total number of Diameter Base Protocol
               messages transmitted since the last reset."
       ::= { dbpLocalStats 2 }

dbpLocalStatsTotalUpTime OBJECT-TYPE
       SYNTAX     TimeTicks
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "This object represents the total amount of
              time this Diameter peer has been up from the
              beginning of time until now.  The value is cumulative
              and persists over resets."
       ::= { dbpLocalStats 3 }

dbpLocalResetTime OBJECT-TYPE
       SYNTAX     TimeTicks
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "If the peer keeps persistent state (e.g., a process)
              and supports a 'reset' operation (e.g., can be told to
              re-read configuration files), this value will be the
              time elapsed (in hundredths of a second) since the
              peer was 'reset'.  For software that does not
              have persistence or does not support a 'reset'
              operation, this value is undefined."
       ::= { dbpLocalStats 4 }

dbpLocalConfigReset OBJECT-TYPE
       SYNTAX     INTEGER { other(1),
                            initializing(2),
                            running(3),
                            reset(4) }
       MAX-ACCESS read-write
       STATUS     current
       DESCRIPTION
             "Status/action object to reinitialize any persistent
              local state.  When set to reset(4), any persistent
              local state (such as a process) is reinitialized as
              if the software had just been started.  This value will
              never be returned by a read operation.  When read,
              one of the following values will be returned:
                 other(1) - peer in some unknown state;
                 initializing(2) - peer (re)initializing;
                 running(3) - peer currently running."
       DEFVAL      { other }
       ::= { dbpLocalStats 5 }
```

```
dbpLocalApplTable OBJECT-TYPE
       SYNTAX     SEQUENCE OF DbpLocalApplEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "The table listing the Diameter applications
              supported by this peer."
       ::= { dbpLocalCfgs 7 }

dbpLocalApplEntry OBJECT-TYPE
       SYNTAX     DbpLocalApplEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "A row entry representing a Diameter
              application on this peer."
       INDEX      { dbpLocalApplIndex }
       ::= { dbpLocalApplTable 1 }

DbpLocalApplEntry ::= SEQUENCE {
       dbpLocalApplIndex  Unsigned32,
       dbpLocalApplStorageType    StorageType,
       dbpLocalApplRowStatus      RowStatus
}

dbpLocalApplIndex OBJECT-TYPE
       SYNTAX      Unsigned32 ( 1..4294967295 )
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
               "A number uniquely identifying a
               supported Diameter application. Upon reload,
               dbpLocalApplIndex values may be changed."
       ::= { dbpLocalApplEntry 1 }

dbpLocalApplStorageType OBJECT-TYPE
    SYNTAX          StorageType
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
            "The storage type for this conceptual row. None of
            the columnar objects is writable when the conceptual
            row is permanent."
    REFERENCE       "Textual Conventions for SMIv2, Section 2."
    DEFVAL          { nonVolatile }
    ::= { dbpLocalApplEntry 2 }

dbpLocalApplRowStatus OBJECT-TYPE
```

```
        SYNTAX     RowStatus
        MAX-ACCESS read-create
        STATUS     current
        DESCRIPTION
             "The status of this conceptual row.

             To create a row in this table, a manager must
             set this object to either createAndGo(4) or
             createAndWait(5).

             Until instances of all corresponding columns are
             appropriately configured, the value of the
             corresponding instance of the dbpLocalApplRowStatus
             column is 'notReady'.

             In particular, a newly created row cannot be made
             active until the corresponding dbpLocalApplIndex
             has been set.

             dbpLocalApplIndex may not be modified while
             the value of this object is active(1): An attempt
             to set these objects while the value of
             dbpLocalApplRowStatus is active(1) will result in
             an inconsistentValue error.

             Entries in this table with dbpLocalApplRowStatus equal
             to active(1) remain in the table until destroyed.

             Entries in this table with dbpLocalApplRowStatus equal
             to values other than active(1) will be destroyed after
             timeout (5 minutes)."
        ::= { dbpLocalApplEntry 3 }

dbpPeerTable OBJECT-TYPE
        SYNTAX     SEQUENCE OF DbpPeerEntry
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
              "The table listing information regarding
               the discovered or configured Diameter
               peers."
        ::= { dbpPeerCfgs 1 }

dbpPeerEntry OBJECT-TYPE
        SYNTAX     DbpPeerEntry
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
```

```
              "A row entry representing a discovered
               or configured Diameter peer."
       INDEX      { dbpPeerIndex }
       ::= { dbpPeerTable 1 }

DbpPeerEntry ::= SEQUENCE {
       dbpPeerIndex                       Unsigned32,
       dbpPeerId                       SnmpAdminString,
       dbpPeerPortConnect                 Unsigned32,
       dbpPeerPortListen                  Unsigned32,
       dbpPeerTransportProtocol            Integer32,
       dbpPeerSecurity                     Integer32,
       dbpPeerFirmwareRevision         SnmpAdminString,
       dbpPeerStorageType                 StorageType,
       dbpPeerRowStatus                     RowStatus }

dbpPeerIndex OBJECT-TYPE
       SYNTAX     Unsigned32 (1..4294967295)
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
              "A number uniquely identifying each Diameter peer
               with which the local peer communicates.
               Upon reload, dbpPeerIndex values may be changed."
       ::= { dbpPeerEntry  1 }

dbpPeerId OBJECT-TYPE
       SYNTAX     SnmpAdminString
       MAX-ACCESS read-create
       STATUS     current
       DESCRIPTION
              "The local identifier for the Diameter peer.
               It must be unique and non-empty."
       ::= { dbpPeerEntry  2 }

dbpPeerPortConnect OBJECT-TYPE
       SYNTAX     Unsigned32 (0|1..65535)
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
              "The connection port used
               to connect to the Diameter peer.
               If there is no active connection, this
               value will be zero(0)."
       ::= { dbpPeerEntry  3 }

dbpPeerPortListen OBJECT-TYPE
       SYNTAX     Unsigned32 (1..65535)
```

```
       MAX-ACCESS read-write
       STATUS     current
       DESCRIPTION
             "The port the peer is listening on."
       ::= { dbpPeerEntry  4 }

dbpPeerTransportProtocol OBJECT-TYPE
       SYNTAX     INTEGER { tcp(1),
                            sctp(2) }
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The transport protocol (tcp/sctp) the
              Diameter peer is using."
       ::= { dbpPeerEntry  5 }

dbpPeerSecurity OBJECT-TYPE
       SYNTAX     INTEGER { other(1),
                            tls(2),
                            ipsec(3) }
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The security the Diameter peer is using.

              other(1) - Unknown Security Protocol
              tls(2)   - Transport Layer Security Protocol
              ipsec(3) - Internet Protocol Security"
       DEFVAL          { other }
       ::= { dbpPeerEntry  6 }

dbpPeerFirmwareRevision OBJECT-TYPE
       SYNTAX     SnmpAdminString
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Firmware revision of peer.
              If the Entity MIB is supported by
              the node, then the contents of this object MUST be
              identical to those of the entPhysicalFirmwareRev
              object [RFC4133]. If no firmware
              revision, the revision of the Diameter software
              module may be reported instead."
       ::= { dbpPeerEntry 7 }

dbpPeerStorageType OBJECT-TYPE
    SYNTAX          StorageType
    MAX-ACCESS      read-create
```

```
    STATUS          current
    DESCRIPTION
            "The storage type for this conceptual row.
            Only the dbpPeerPortListen object is writable when
            the conceptual row is permanent."
    REFERENCE       "Textual Conventions for SMIv2, Section 2."
    DEFVAL          { nonVolatile }
    ::= { dbpPeerEntry 8 }

dbpPeerRowStatus OBJECT-TYPE
       SYNTAX     RowStatus
       MAX-ACCESS read-create
       STATUS     current
       DESCRIPTION
             "Status of the peer entry: creating the entry
              enables the peer, destroying the entry disables
              the peer."
       ::= {dbpPeerEntry 9 }

dbpPeerIpAddrTable OBJECT-TYPE
       SYNTAX     SEQUENCE OF DbpPeerIpAddrEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "The table listing the Diameter
              peer IP addresses."
       ::= { dbpPeerCfgs 2 }

dbpPeerIpAddrEntry OBJECT-TYPE
       SYNTAX     DbpPeerIpAddrEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "A row entry representing a
              the IP address of a Diameter peer."
       INDEX      {
                    dbpPeerIndex,
                    dbpPeerIpAddressIndex }
       ::= { dbpPeerIpAddrTable 1 }

DbpPeerIpAddrEntry ::= SEQUENCE {
       dbpPeerIpAddressIndex Unsigned32,
       dbpPeerIpAddressType  InetAddressType,
       dbpPeerIpAddress      InetAddress }

dbpPeerIpAddressIndex OBJECT-TYPE
       SYNTAX     Unsigned32 (1..4294967295)
       MAX-ACCESS not-accessible
```

```
        STATUS     current
        DESCRIPTION
              "A number uniquely identifying an IP Address
               supported by this Diameter peer."
        ::= { dbpPeerIpAddrEntry  1 }

dbpPeerIpAddressType OBJECT-TYPE
        SYNTAX      InetAddressType
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
              "The type of address stored in dbpPeerIpAddress."
        ::= {dbpPeerIpAddrEntry 2}

dbpPeerIpAddress OBJECT-TYPE
        SYNTAX      InetAddress
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
              "The active IP Address(es) used for connections."
        ::= {dbpPeerIpAddrEntry 3}

dbpAppAdvToPeerTable OBJECT-TYPE
        SYNTAX     SEQUENCE OF DbpAppAdvToPeerEntry
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
              "The table listing the applications advertised by
               this host to each peer and the types of service
               supported: accounting, authentication or both."
        ::= { dbpLocalCfgs 8 }

dbpAppAdvToPeerEntry OBJECT-TYPE
        SYNTAX     DbpAppAdvToPeerEntry
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
              "A row entry representing a discovered or
               configured Diameter peer."
        INDEX      { dbpPeerIndex,
                     dbpAppAdvToPeerVendorId,
                     dbpAppAdvToPeerIndex }
        ::= { dbpAppAdvToPeerTable 1 }

DbpAppAdvToPeerEntry ::= SEQUENCE {
        dbpAppAdvToPeerVendorId                Unsigned32,
        dbpAppAdvToPeerIndex                   Unsigned32,
        dbpAppAdvToPeerServices               ServiceType,
```

```
       dbpAppAdvToPeerStorageType             StorageType,
       dbpAppAdvToPeerRowStatus                  RowStatus }

dbpAppAdvToPeerVendorId OBJECT-TYPE
       SYNTAX     Unsigned32 ( 1..4294967295 )
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "The IANA Enterprise Code value assigned to
              the vendor of the Diameter device."
       ::= { dbpAppAdvToPeerEntry  1 }

dbpAppAdvToPeerIndex OBJECT-TYPE
       SYNTAX     Unsigned32 ( 1..4294967295 )
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "A number uniquely identifying a Diameter
              application advertised as supported by
              this host to each peer.  Upon reload,
              dbpAppAdvToPeerIndex values may be
              changed"
       ::= { dbpAppAdvToPeerEntry  2 }

dbpAppAdvToPeerServices OBJECT-TYPE
       SYNTAX     ServiceType
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The type of services supported for each application,
              accounting, authentication or both."
       ::= { dbpAppAdvToPeerEntry  3 }

dbpAppAdvToPeerStorageType OBJECT-TYPE
    SYNTAX          StorageType
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
            "The storage type for this conceptual row.
            None of the objects are writable when the
            conceptual row is permanent."
    REFERENCE       "Textual Conventions for SMIv2, Section 2."
    DEFVAL          { nonVolatile }
    ::= { dbpAppAdvToPeerEntry 4 }

dbpAppAdvToPeerRowStatus OBJECT-TYPE
       SYNTAX     RowStatus
       MAX-ACCESS read-create
```

```
       STATUS     current
       DESCRIPTION
             "Status of the entry: creating the entry causes the
              application to be advertised, destroying the entry
              ceases advertisement."
       ::= { dbpAppAdvToPeerEntry  5 }

-- Applications advertised BY peers

dbpAppAdvFromPeerTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF DbpAppAdvFromPeerEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
            "The table listing the applications advertised by
            each peer to this host and the types of service
            supported: accounting, authentication or both."
    ::= { dbpPeerCfgs 3 }

dbpAppAdvFromPeerEntry OBJECT-TYPE
    SYNTAX          DbpAppAdvFromPeerEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
            "A row entry representing a discovered or
            configured Diameter peer."
    INDEX           {
                        dbpPeerIndex,
                        dbpAppAdvFromPeerVendorId,
                        dbpAppAdvFromPeerIndex
                    }
    ::= { dbpAppAdvFromPeerTable 1 }

DbpAppAdvFromPeerEntry ::= SEQUENCE {
        dbpAppAdvFromPeerVendorId Unsigned32,
        dbpAppAdvFromPeerIndex    Unsigned32,
        dbpAppAdvFromPeerType     ServiceType
}

dbpAppAdvFromPeerVendorId OBJECT-TYPE
    SYNTAX          Unsigned32 (1..4294967295 )
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
            "The IANA Enterprise Code value assigned to
            the vendor of the Diameter application."
    ::= { dbpAppAdvFromPeerEntry 1 }
```

```
dbpAppAdvFromPeerIndex OBJECT-TYPE
    SYNTAX          Unsigned32 (1..4294967295 )
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
            "A number uniquely identifying the applications
            advertised as supported from each Diameter peer."
    ::= { dbpAppAdvFromPeerEntry 2 }

dbpAppAdvFromPeerType OBJECT-TYPE
    SYNTAX          ServiceType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
            "The type of services supported for each application,
            accounting, authentication or both."
    ::= { dbpAppAdvFromPeerEntry 3 }

-- table of vendor-IDs supported by each peer

dbpPeerVendorTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF DbpPeerVendorEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
            "The table listing the Vendor IDs
            supported by the peer."
    ::= { dbpPeerCfgs 4 }

dbpPeerVendorEntry OBJECT-TYPE
    SYNTAX          DbpPeerVendorEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
            "A row entry representing a
            Vendor ID supported by the peer."
    INDEX           {
                        dbpPeerIndex,
                        dbpPeerVendorIndex
                    }
    ::= { dbpPeerVendorTable 1 }

DbpPeerVendorEntry ::= SEQUENCE {
        dbpPeerVendorIndex       Unsigned32,
        dbpPeerVendorId          INTEGER,
        dbpPeerVendorStorageType StorageType,
        dbpPeerVendorRowStatus   RowStatus
}
```

```
dbpPeerVendorIndex OBJECT-TYPE
    SYNTAX          Unsigned32 (1..4294967295)
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
            "A number uniquely identifying the Vendor
            ID supported by the peer. Upon reload,
            dbpPeerVendorIndex values may be changed."
    ::= { dbpPeerVendorEntry 1 }

dbpPeerVendorId OBJECT-TYPE
    SYNTAX          INTEGER  {
                        diameterVendorIetf (0),
                        diameterVendorCisco (9),
                        diameterVendor3gpp (10415),
                        diameterVendorVodafone (12645)
                    }
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
            "The active vendor ID used for peer connections.
            diameterVendorIetf (0)           -- IETF
            diameterVendorCisco (9)         -- Cisco Systems
            diameterVendor3gpp (10415)       --  3GPP
            diameterVendorVodafone (12645)     --- Vodafone"
    DEFVAL          { diameterVendorIetf }
    ::= { dbpPeerVendorEntry 2 }

dbpPeerVendorStorageType OBJECT-TYPE
    SYNTAX          StorageType
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
            "The storage type for this conceptual row.
            None of the objects are writable when the
            conceptual row is permanent."
    REFERENCE       "Textual Conventions for SMIv2, Section 2."
    DEFVAL          { nonVolatile }
    ::= { dbpPeerVendorEntry 3 }

dbpPeerVendorRowStatus OBJECT-TYPE
    SYNTAX          RowStatus
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
            "The status of this conceptual row.

            To create a row in this table, a manager must
```

```
            set this object to either createAndGo(4) or
            createAndWait(5).

            Until instances of all corresponding columns are
            appropriately configured, the value of the
            corresponding instance of the dbpPeerVendorRowStatus
            column is 'notReady'.

            In particular, a newly created row cannot be made
            active until the corresponding dbpPeerVendorId has been
            set. Also, a newly created row cannot be made active
            until the corresponding 'dbpPeerIndex' has been set.

            dbpPeerVendorId may not be modified while the
            value of this object is active(1):
            An attempt to set these objects while the value of
            dbpPeerVendorRowStatus is active(1) will result in
            an inconsistentValue error.

            Entries in this table with dbpPeerVendorRowStatus equal
            to active(1) remain in the table until destroyed.

            Entries in this table with dbpPeerVendorRowStatus equal
            to values other than active(1) will be destroyed after
            timeout (5 minutes)."
    ::= { dbpPeerVendorEntry 4 }


dbpPerPeerInfoTable OBJECT-TYPE
       SYNTAX     SEQUENCE OF DbpPerPeerInfoEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "The table listing Diameter per-peer information."
       ::= { dbpPeerInfo 1 }

dbpPerPeerInfoEntry OBJECT-TYPE
       SYNTAX     DbpPerPeerInfoEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "A row entry representing a Diameter peer."
       INDEX     { dbpPeerIndex }
       ::= { dbpPerPeerInfoTable 1 }

DbpPerPeeInfoEntry ::= SEQUENCE {
       dbpPerPeerInfoState               Integer32,
       dbpPerPeeInfoStateDuration       TimeTicks,
```

```
      dbpPerPeerInfoLastDiscCause       Integer32,
      dbpPerPeerInfoWhoInitDisconnect   Integer32,
      dbpPerPeerStatsDWCurrentStatus     Integer32,
      dbpPerPeerStatsTimeoutConnAtmpts   Counter32,
      dbpPerPeerStatsASRsIn              Counter32,
      dbpPerPeerStatsASRsOut             Counter32,
      dbpPerPeerStatsASAsIn              Counter32,
      dbpPerPeerStatsASAsOut             Counter32,
      dbpPerPeerStatsACRsIn              Counter32,
      dbpPerPeerStatsACRsOut             Counter32,
      dbpPerPeerStatsACAsIn              Counter32,
      dbpPerPeerStatsACAsOut             Counter32,
      dbpPerPeerStatsCERsIn              Counter32,
      dbpPerPeerStatsCERsOut             Counter32,
      dbpPerPeerStatsCEAsIn              Counter32,
      dbpPerPeerStatsCEAsOut             Counter32,
      dbpPerPeerStatsDWRsIn              Counter32,
      dbpPerPeerStatsDWRsOut             Counter32,
      dbpPerPeerStatsDWAsIn              Counter32,
      dbpPerPeerStatsDWAsOut             Counter32,
      dbpPerPeerStatsDPRsIn              Counter32,
      dbpPerPeerStatsDPRsOut             Counter32,
      dbpPerPeerStatsDPAsIn              Counter32,
      dbpPerPeerStatsDPAsOut             Counter32,
      dbpPerPeerStatsRARsIn              Counter32,
      dbpPerPeerStatsRARsOut             Counter32,
      dbpPerPeerStatsRAAsIn              Counter32,
      dbpPerPeerStatsRAAsOut             Counter32,
      dbpPerPeerStatsSTRsIn              Counter32,
      dbpPerPeerStatsSTRsOut             Counter32,
      dbpPerPeerStatsSTAsIn              Counter32,
      dbpPerPeerStatsSTAsOut             Counter32,
      dbpPerPeerInfoDWReqTimer           TimeTicks,
      dbpPerPeerStatsRedirectEvents      Counter32,
      dbpPerPeerStatsAccDupRequests      Counter32,
      dbpPerPeerStatsMalformedReqsts     Counter32,
      dbpPerPeerStatsAccsNotRecorded     Counter32,
      dbpPerPeerStatsAccRetrans          Counter32,
      dbpPerPeerStatsTotalRetrans        Counter32,
      dbpPerPeerStatsAccPendReqstsOut    Gauge32,
      dbpPerPeerStatsAccReqstsDropped    Counter32,
      dbpPerPeerStatsHByHDropMessages    Counter32,
      dbpPerPeerStatsEToEDupMessages     Counter32,
      dbpPerPeerStatsUnknownTypes        Counter32,
      dbpPerPeerStatsProtocolErrors      Counter32,
      dbpPerPeerStatsTransientFailures   Counter32,
      dbpPerPeerStatsPermanentFailures   Counter32,
      dbpPerPeerStatsTransportDown       Counter32 }
```

```
dbpPerPeerInfoState OBJECT-TYPE
       SYNTAX     INTEGER { closed(1),
                            waitConnAck(2),
                            waitICea(3),
                            elect(4),
                            waitReturns(5),
                            rOpen(6),
                            iOpen(7),
                            closing(8) }
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Connection state in the Peer State Machine of
              the peer with which this Diameter peer is
              communicating.

              closed      - Connection closed with this peer.
              waitConnAck - Waiting for an acknowledgment
                            from this peer.
              waitICea    - Waiting for a Capabilities-Exchange-Answer
                            from this peer.
              elect       - When the remote and local peers are both
                            trying to bring up a connection with
                            each other at the same time.  An
                            election process begins which
                            determines which socket remains open.
              waitReturns - Waiting for election returns.
              r-open      - Responder transport connection is
                            used for communication.
              i-open      - Initiator transport connection is
                            used for communication.
              closing     - Actively closing and doing cleanup."
       ::= { dbpPerPeerInfoEntry 1 }

dbpPerPeerInfoStateDuration OBJECT-TYPE
       SYNTAX     TimeTicks
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The elapsed time (in hundredths of a second)
              since the last state change."
       ::= { dbpPerPeerInfoEntry 2 }

dbpPerPeerInfoLastDiscCause OBJECT-TYPE
       SYNTAX     INTEGER { rebooting(1),
                            busy(2),
                            doNotWantToTalk(3),
                            election(4) }
```

```
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "The last cause for a peer's disconnection.

               rebooting       - A scheduled reboot is imminent.
               busy            - The peer's internal resources are
                                 constrained, and it has determined
                                 that the transport connection needs
                                 to be shutdown.
               doNotWantToTalk - The peer has determined that
                                 it does not see a need for the
                                 transport connection to exist,
                                 since it does not expect any
                                 messages to be exchanged in
                                 the foreseeable future.
               electionLost    - The peer has determined that it
                                 has lost the election process
                                 and has therefore disconnected
                                 the transport connection."
        ::= { dbpPerPeerInfoEntry 3 }

dbpPerPeerInfoWhoInitDisconnect OBJECT-TYPE
        SYNTAX     INTEGER { host(1),
                             peer(2) }
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Did the host or peer initiate the disconnect?

               host - If this peer initiated the disconnect.
               peer - If the peer with which this peer was
                      connected initiated the disconnect."
        ::= { dbpPerPeerInfoEntry 4 }

dbpPerPeerStatsDWCurrentStatus OBJECT-TYPE
        SYNTAX     INTEGER { okay(1),
                             suspect(2),
                             down(3),
                             reopen(4) }
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "okay    - Indicates the connection is presumed working.
               suspect - Indicates the connection is possibly
                         congested or down.
               down    - The peer is no longer reachable, causing
                         the transport connection to be shutdown.
```

```
                reopen  - Three watchdog messages are exchanged with
                          accepted round trip times, and the connection
                          to the peer is considered stabilized."
        ::= { dbpPerPeerInfoEntry 5 }

dbpPerPeerStatsTimeoutConnAtmpts OBJECT-TYPE
        SYNTAX     Counter32 UNITS "connection attempts"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "If there is no transport connection with a peer,
               this is the number of times the local peer has attempted
               to connect to that peer.  This is reset on
               connection."
        ::= { dbpPerPeerInfoEntry 6 }

dbpPerPeerStatsASRsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Abort-Session-Request messages
               received from the peer."
        ::= { dbpPerPeerInfoEntry 7 }

dbpPerPeerStatsASRsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Abort-Session-Request
               messages sent to the peer."
        ::= { dbpPerPeerInfoEntry 8 }

dbpPerPeerStatsASAsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Abort-Session-Answer
               messages received from the peer."
        ::= { dbpPerPeerInfoEntry 9 }

dbpPerPeerStatsASAsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
```

```
              "Number of Abort-Session-Answer
               messages sent to the peer."
       ::= { dbpPerPeerInfoEntry 10 }

dbpPerPeerStatsACRsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
              "Number of Accounting-Request messages
               received from the peer."
       ::= { dbpPerPeerInfoEntry 11 }

dbpPerPeerStatsACRsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
              "Number of Accounting-Request messages
               sent to the peer."
       ::= { dbpPerPeerInfoEntry 12 }

dbpPerPeerStatsACAsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
              "Number of Accounting-Answer messages
               received from the peer."
       ::= { dbpPerPeerInfoEntry 13 }

dbpPerPeerStatsACAsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
              "Number of Accounting-Answer messages
               sent to the peer."
       ::= { dbpPerPeerInfoEntry 14 }

dbpPerPeerStatsCERsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
              "Number of Capabilities-Exchange-Request
               messages received from the peer."
       ::= { dbpPerPeerInfoEntry 15 }
```

```
dbpPerPeerStatsCERsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Capabilities-Exchange-Request
              messages sent to the peer."
       ::= { dbpPerPeerInfoEntry 16 }

dbpPerPeerStatsCEAsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Capabilities-Exchange-Answer
              messages received from the peer."
       ::= { dbpPerPeerInfoEntry 17 }

dbpPerPeerStatsCEAsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Capabilities-Exchange-Answer
              messages sent to the peer."
       ::= { dbpPerPeerInfoEntry 18 }

dbpPerPeerStatsDWRsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Device-Watchdog-Request
              messages received from the peer."
       ::= { dbpPerPeerInfoEntry 19 }

dbpPerPeerStatsDWRsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Device-Watchdog-Request
              messages sent to the peer."
       ::= { dbpPerPeerInfoEntry 20 }

dbpPerPeerStatsDWAsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
```

```
        STATUS     current
        DESCRIPTION
              "Number of Device-Watchdog-Answer
               messages received from the peer."
        ::= { dbpPerPeerInfoEntry 21 }

dbpPerPeerStatsDWAsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Device-Watchdog-Answer
               messages sent to the peer."
        ::= { dbpPerPeerInfoEntry 22 }

dbpPerPeerStatsDPRsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Disconnect-Peer-Request messages
               received."
        ::= { dbpPerPeerInfoEntry 23 }

dbpPerPeerStatsDPRsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Disconnect-Peer-Request messages
               sent."
        ::= { dbpPerPeerInfoEntry 24 }

dbpPerPeerStatsDPAsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Disconnect-Peer-Answer messages
               received."
        ::= { dbpPerPeerInfoEntry 25 }

dbpPerPeerStatsDPAsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Disconnect-Peer-Answer messages
```

```
                sent."
        ::= { dbpPerPeerInfoEntry 26 }

dbpPerPeerStatsRARsIn  OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Re-Auth-Request messages
               received."
        ::= { dbpPerPeerInfoEntry 27 }

dbpPerPeerStatsRARsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Re-Auth-Request messages
               sent."
        ::= { dbpPerPeerInfoEntry 28 }

dbpPerPeerStatsRAAsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Re-Auth-Answer messages
               received."
        ::= { dbpPerPeerInfoEntry 29 }

dbpPerPeerStatsRAAsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Re-Auth-Answer messages
               sent."
        ::= { dbpPerPeerInfoEntry 30 }

dbpPerPeerStatsSTRsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Session-Termination-Request
               messages received from the peer."
        ::= { dbpPerPeerInfoEntry 31 }
```

```
dbpPerPeerStatsSTRsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Session-Termination-Request
              messages sent to the peer."
       ::= { dbpPerPeerInfoEntry 32 }

dbpPerPeerStatsSTAsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Session-Termination-Answer
              messages received from the peer."
       ::= { dbpPerPeerInfoEntry 33 }

dbpPerPeerStatsSTAsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Session-Termination-Answer
              messages sent to the peer."
       ::= { dbpPerPeerInfoEntry 34 }

dbpPerPeerInfoDWReqTimer OBJECT-TYPE
       SYNTAX     TimeTicks
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Device-Watchdog Request Timer, which
              is the interval between messages sent to
              peers."
       ::= { dbpPerPeerInfoEntry 35 }

dbpPerPeerStatsRedirectEvents OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Redirect Event count, which is the number
              of redirects sent from a peer."
       ::= { dbpPerPeerInfoEntry 36 }

dbpPerPeerStatsAccDupRequests OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
```

```
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "The number of duplicate Diameter Accounting-Request
               messages received."
        ::= { dbpPerPeerInfoEntry 37 }

dbpPerPeerStatsMalformedReqsts OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "The number of malformed Diameter
               messages received."
        ::= { dbpPerPeerInfoEntry 38 }

dbpPerPeerStatsAccsNotRecorded OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "The number of Diameter Accounting-Request messages
               which were received and responded to but not
               recorded."
        ::= { dbpPerPeerInfoEntry 39 }

dbpPerPeerStatsAccRetrans OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "The number of Diameter Accounting-Request messages
               retransmitted to this Diameter peer."
        ::= { dbpPerPeerInfoEntry 40 }

dbpPerPeerStatsTotalRetrans OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "The number of Diameter messages retransmitted
               to this Diameter peer, not to include Diameter
               Accounting-Request messages retransmitted."
        ::= { dbpPerPeerInfoEntry 41 }

dbpPerPeerStatsAccPendReqstsOut OBJECT-TYPE
        SYNTAX     Gauge32
        MAX-ACCESS read-only
```

```
       STATUS     current
       DESCRIPTION
          "The number of Diameter Accounting-Request messages
           sent to this peer that have not yet timed out or
           received a response. This variable is incremented when an
           Accounting-Request is received by this server and decremented
           due to the transmission of an Accounting-Response, a timeout
           or a retransmission."
       ::= { dbpPerPeerInfoEntry 42 }

dbpPerPeerStatsAccReqstsDropped OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
          "The number of Accounting-Requests to this server
           that have been dropped."
       ::= { dbpPerPeerInfoEntry 43 }

dbpPerPeerStatsHByHDropMessages OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "An answer message that is received with an unknown
              Hop-by-Hop Identifier.  Does not include Accounting
              Requests dropped."
       ::= { dbpPerPeerInfoEntry 44 }

dbpPerPeerStatsEToEDupMessages OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Duplicate answer messages that are to be locally
              consumed.  Does not include duplicate Accounting
              Requests received."
       ::= { dbpPerPeerInfoEntry 45 }

dbpPerPeerStatsUnknownTypes OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The number of Diameter messages of unknown type
              which were received."
       ::= { dbpPerPeerInfoEntry 46 }
```

```
dbpPerPeerStatsProtocolErrors OBJECT-TYPE
       SYNTAX     Counter32 UNITS "errors"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of protocol errors returned to peer,
              but not including redirects."
       ::= { dbpPerPeerInfoEntry 47 }

dbpPerPeerStatsTransientFailures OBJECT-TYPE
       SYNTAX     Counter32 UNITS "errors"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Transient Failure count."
       ::= { dbpPerPeerInfoEntry 48 }

dbpPerPeerStatsPermanentFailures OBJECT-TYPE
       SYNTAX     Counter32 UNITS "errors"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of permanent failures returned to peer."
       ::= { dbpPerPeerInfoEntry 49 }

dbpPerPeerStatsTransportDown OBJECT-TYPE
       SYNTAX     Counter32 UNITS "errors"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of unexpected transport failures."
       ::= { dbpPerPeerInfoEntry 50 }


-- Realm-based Routing Table

dbpRealmMessageRouteTable OBJECT-TYPE
       SYNTAX     SEQUENCE OF DbpRealmMessageRouteEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "The table listing the Diameter
              Realm-based Message Route information."
       ::= { dbpRealmStats 1 }

dbpRealmMessageRouteEntry OBJECT-TYPE
       SYNTAX     DbpRealmMessageRouteEntry
       MAX-ACCESS not-accessible
```

```
        STATUS     current
        DESCRIPTION
              "A row entry representing a Diameter
               Realm Based Message Route."
        INDEX      { dbpRealmMessageRouteIndex }
        ::= { dbpRealmMessageRouteTable 1 }

DbpRealmMessageRouteEntry ::= SEQUENCE {
        dbpRealmMessageRouteIndex             Unsigned32,
        dbpRealmMessageRouteRealm         SnmpAdminString,
        dbpRealmMessageRouteApp               Unsigned32,
        dbpRealmMessageRouteType             ServiceType,
        dbpRealmMessageRouteAction             Integer32,
        dbpRealmMessageRouteACRsIn             Counter32,
        dbpRealmMessageRouteACRsOut            Counter32,
        dbpRealmMessageRouteACAsIn             Counter32,
        dbpRealmMessageRouteACAsOut            Counter32,
        dbpRealmMessageRouteRARsIn             Counter32,
        dbpRealmMessageRouteRARsOut            Counter32,
        dbpRealmMessageRouteRAAsIn             Counter32,
        dbpRealmMessageRouteRAAsOut            Counter32,
        dbpRealmMessageRouteSTRsIn             Counter32,
        dbpRealmMessageRouteSTRsOut            Counter32,
        dbpRealmMessageRouteSTAsIn             Counter32,
        dbpRealmMessageRouteSTAsOut            Counter32,
        dbpRealmMessageRouteASRsIn             Counter32,
        dbpRealmMessageRouteASRsOut            Counter32,
        dbpRealmMessageRouteASAsIn             Counter32,
        dbpRealmMessageRouteASAsOut            Counter32,
        dbpRealmMessageRouteAccRetrans         Counter32,
        dbpRealmMessageRouteAccDupReqsts       Counter32,
        dbpRealmMessageRoutePendReqstsOut        Gauge32,
        dbpRealmMessageRouteReqstsDrop         Counter32 }

dbpRealmMessageRouteIndex OBJECT-TYPE
        SYNTAX     Unsigned32 (1..4294967295)
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
              "A number uniquely identifying each Realm."
        ::= { dbpRealmMessageRouteEntry 1 }

dbpRealmMessageRouteRealm OBJECT-TYPE
        SYNTAX     SnmpAdminString
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Realm name"
```

```
       ::= { dbpRealmMessageRouteEntry 2 }

dbpRealmMessageRouteApp OBJECT-TYPE
       SYNTAX     Unsigned32 (1..4294967295)
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Application id used to route messages
              to this realm."
       ::= { dbpRealmMessageRouteEntry 3 }

dbpRealmMessageRouteType OBJECT-TYPE
       SYNTAX     ServiceType
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The types of service supported for each
              realm application: accounting,
              authentication or both."
       ::= { dbpRealmMessageRouteEntry 4 }

dbpRealmMessageRouteAction OBJECT-TYPE
       SYNTAX     INTEGER { local(1),
                            relay(2),
                            proxy(3),
                            redirect(4) }
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The action is used to identify how a
              message should be treated based on the realm,
              application and type.
              local    - Diameter messages that resolve to a
                         route entry with the Local Action set to
                         Local can be satisfied locally, and do
                         not need to be routed to another peer.
              relay    - All Diameter messages that fall within
                         this category MUST be routed to a
                         next-hop peer, without modifying any
                         non-routing AVPs.
              proxy    - All Diameter messages that fall within this
                         category MUST be routed to a next-hop
                         peer.
              redirect - Diameter messages that fall within this
                         category MUST have the identity of the home
                         Diameter peer(s) appended, and returned
                         to the sender of the message."
       ::= { dbpRealmMessageRouteEntry 5 }
```

```
-- Per-realm message statistics

dbpRealmMessageRouteACRsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Accounting-Request messages
              received from the realm."
       ::= { dbpRealmMessageRouteEntry 6 }

dbpRealmMessageRouteACRsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Accounting-Request messages
              sent to the realm."
       ::= { dbpRealmMessageRouteEntry 7 }

dbpRealmMessageRouteACAsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Accounting-Answer messages
              received from the realm."
       ::= { dbpRealmMessageRouteEntry 8 }

dbpRealmMessageRouteACAsOut OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Accounting-Answer messages
              sent to the realm."
       ::= { dbpRealmMessageRouteEntry 9 }

dbpRealmMessageRouteRARsIn OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "Number of Re-Auth-Request messages
              received from the realm."
       ::= { dbpRealmMessageRouteEntry 10 }

dbpRealmMessageRouteRARsOut OBJECT-TYPE
```

```
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Re-Auth-Request messages
               sent to the realm."
        ::= { dbpRealmMessageRouteEntry 11 }

dbpRealmMessageRouteRAAsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Re-Auth-Answer messages
               received from the realm."
        ::= { dbpRealmMessageRouteEntry 12 }

dbpRealmMessageRouteRAAsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Re-Auth-Answer messages
               sent to the realm."
        ::= { dbpRealmMessageRouteEntry 13 }

dbpRealmMessageRouteSTRsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Session-Termination-Request messages
               received from the realm."
        ::= { dbpRealmMessageRouteEntry 14 }

dbpRealmMessageRouteSTRsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Session-Termination-Request messages
               sent to the realm."
        ::= { dbpRealmMessageRouteEntry 15 }

dbpRealmMessageRouteSTAsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
```

```
        DESCRIPTION
              "Number of Session-Termination-Answer messages
               received from the realm."
        ::= { dbpRealmMessageRouteEntry 16 }

dbpRealmMessageRouteSTAsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Session-Termination-Answer messages
               sent to the realm."
        ::= { dbpRealmMessageRouteEntry 17 }

dbpRealmMessageRouteASRsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Abort-Session-Request messages
               received from the realm."
        ::= { dbpRealmMessageRouteEntry 18 }

dbpRealmMessageRouteASRsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Abort-Session-Request messages
               sent to the realm."
        ::= { dbpRealmMessageRouteEntry 19 }

dbpRealmMessageRouteASAsIn OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Abort-Session-Answer messages
               received from the realm."
        ::= { dbpRealmMessageRouteEntry 20 }

dbpRealmMessageRouteASAsOut OBJECT-TYPE
        SYNTAX     Counter32 UNITS "messages"
        MAX-ACCESS read-only
        STATUS     current
        DESCRIPTION
              "Number of Abort-Session-Answer messages
               sent to the realm."
```

```
       ::= { dbpRealmMessageRouteEntry 21 }

dbpRealmMessageRouteAccRetrans OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The number of Diameter accounting messages
              retransmitted to this realm."
       ::= { dbpRealmMessageRouteEntry 22 }

dbpRealmMessageRouteAccDupReqsts OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The number of duplicate Diameter accounting
              messages sent to this realm."
       ::= { dbpRealmMessageRouteEntry 23 }

dbpRealmMessageRoutePendReqstsOut OBJECT-TYPE
       SYNTAX     Gauge32
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
          "The number of Diameter Accounting-Request messages
           sent to this peer that have not yet timed out or
           received a response. This variable is incremented when an
           Accounting-Request is sent to this peer and decremented
           due to receipt of an Accounting-Response, a timeout or
           a retransmission."
       ::= { dbpRealmMessageRouteEntry 24 }

dbpRealmMessageRouteReqstsDrop OBJECT-TYPE
       SYNTAX     Counter32 UNITS "messages"
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The number of requsts dropped by this realm."
       ::= { dbpRealmMessageRouteEntry 25 }

dbpRealmKnownPeersTable OBJECT-TYPE
       SYNTAX     SEQUENCE OF DbpRealmKnownPeersEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "The table listing the Diameter
              Realms and known peers.
```

```
              This is an ordered list, where the ordering
              signifies the order in which the peers are
              tried."
       ::= { dbpRealmCfgs 1 }

dbpRealmKnownPeersEntry OBJECT-TYPE
       SYNTAX     DbpRealmKnownPeersEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "A row entry representing a Diameter
              Realm and known peers."
       INDEX      { dbpRealmMessageRouteIndex,
                    dbpRealmKnownPeersIndex }
       ::= { dbpRealmKnownPeersTable 1 }

DbpRealmKnownPeersEntry ::= SEQUENCE {
       dbpRealmKnownPeersIndex                Unsigned32,
       dbpRealmKnownPeers                     Unsigned32,
       dbpRealmKnownPeersChosen                  Integer32 }

dbpRealmKnownPeersIndex OBJECT-TYPE
       SYNTAX     Unsigned32 (1..4294967295)
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
             "A sequential identifier number."
       ::= { dbpRealmKnownPeersEntry 1 }

dbpRealmKnownPeers OBJECT-TYPE
       SYNTAX     Unsigned32 (1..4294967295)
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "The index of the peer this realm knows about.
              Same as the dbpPeerIndex"
       ::= { dbpRealmKnownPeersEntry 2 }

dbpRealmKnownPeersChosen OBJECT-TYPE
       SYNTAX     INTEGER { other(1),
                            roundRobin(2),
                            loadBalance(3),
                            firstPreferred(4),
                            mostRecentFirst(5) }
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
             "How the realm chooses which peer to send
```

```
              messages to.
              roundRobin      - The peer used for each transaction
                                is selected based on the order in
                                which peers are configured.
              loadBalance     - The peer used for each transaction
                                is based on the load metric (
                                implementation dependent) of all
                                peers defined for the realm, with
                                the least loaded peer selected
                                first.
              firstPreferred  - The first defined peer is always
                                used for transactions unless
                                failover occurs.
              mostRecentFirst - The most recently used peer is
                                used first for each transaction."
       ::= { dbpRealmKnownPeersEntry 3 }

-- Conformance

diameterBaseProtocolMIBCompliances
                    OBJECT IDENTIFIER ::= { diameterBaseConform 1 }

diameterBaseProtocolMIBGroups
                    OBJECT IDENTIFIER ::= { diameterBaseConform 2 }

-- Compliance Statements

diameterBPMIBCompliances MODULE-COMPLIANCE
       STATUS     current
       DESCRIPTION
            "The compliance statement for Diameter Base
             Protocol entities."
       MODULE -- this module
            MANDATORY-GROUPS {
                              dbpLocalCfgGroup,
                              dbpPeerCfgGroup,
                              dbpPeerStatsGroup,
                              dbpNotificationsGroup,
                              dbpNotifCfgGroup }

            GROUP             dbpLocalCfgSkippedGroup
            DESCRIPTION
               "This group is only mandatory for a system that
                implements all the local config objects."

            GROUP             dbpLocalStatsSkippedGroup
            DESCRIPTION
               "This group is only mandatory for a system that
```

```
                implements all the local statistics objects."

            GROUP              dbpPeerCfgSkippedGroup
            DESCRIPTION
               "This group is only mandatory for a system that
                implements all the peer config objects."

            GROUP              dbpPeerStatsSkippedGroup
            DESCRIPTION
               "This group is only mandatory for a system that
                implements all the peer statistic objects."

            GROUP              dbpRealmCfgSkippedGroup
            DESCRIPTION
               "This group is only mandatory for a system that
                implementsall the realm config objects."

            GROUP              dbpRealmStatsSkippedGroup
            DESCRIPTION
               "This group is only mandatory for a system that
                implements all the realm statistic objects."

       ::= { diameterBaseProtocolMIBCompliances 1 }

-- Units of Conformance

dbpLocalCfgGroup OBJECT-GROUP
      OBJECTS {
              dbpLocalRealm,
              dbpLocalOriginHost,
              dbpLocalVendorId,
              dbpLocalVendorStorageType,
              dbpLocalVendorRowStatus
      }
      STATUS  current
      DESCRIPTION
            "A collection of objects providing configuration common
             to the peer."
      ::= { diameterBaseProtocolMIBGroups 1 }

dbpPeerCfgGroup OBJECT-GROUP
    OBJECTS   {
              dbpPeerId,
              dbpPeerPortConnect,
              dbpPeerPortListen,
              dbpPeerProtocol,
              dbpPeerSecurity,
              dbpPeerFirmwareRevision,
```

```
                dbpPeerStorageType,
                dbpPeerRowStatus,
                dbpPeerIpAddressType,
                dbpPeerIpAddress,
                dbpPeerVendorId,
                dbpPeerVendorStorageType,
                dbpPeerVendorRowStatus
    }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing configuration
            of the Diameter peers."
    ::= { diameterBaseProtocolMIBGroups 2 }

dbpPeerStatsGroup OBJECT-GROUP
    OBJECTS   {
                dbpPeerStatsState,
                dbpPeerStatsStateDuration,
                dbpPeerStatsLastDiscCause,
                dbpPeerStatsWhoInitDisconnect,
                dbpPeerStatsDWCurrentStatus,
                dbpPeerStatsTimeoutConnAtmpts,
                dbpPeerStatsASRsIn,
                dbpPeerStatsASRsOut,
                dbpPeerStatsASAsIn,
                dbpPeerStatsASAsOut,
                dbpPeerStatsACRsIn,
                dbpPeerStatsACRsOut,
                dbpPeerStatsACAsIn,
                dbpPeerStatsACAsOut,
                dbpPeerStatsCERsIn,
                dbpPeerStatsCERsOut,
                dbpPeerStatsCEAsIn,
                dbpPeerStatsCEAsOut,
                dbpPeerStatsDWRsIn,
                dbpPeerStatsDWRsOut,
                dbpPeerStatsDWAsIn,
                dbpPeerStatsDWAsOut,
                dbpPeerStatsDPRsIn,
                dbpPeerStatsDPRsOut,
                dbpPeerStatsDPAsIn,
                dbpPeerStatsDPAsOut,
                dbpPeerStatsRARsIn,
                dbpPeerStatsRARsOut,
                dbpPeerStatsRAAsIn,
                dbpPeerStatsRAAsOut,
                dbpPeerStatsSTRsIn,
                dbpPeerStatsSTRsOut,
```

```
                dbpPeerStatsSTAsIn,
                dbpPeerStatsSTAsOut,
                dbpPeerStatsDWReqTimer,
                dbpPeerStatsRedirectEvents,
                dbpPeerStatsAccDupRequests,
                dbpPeerStatsMalformedReqsts,
                dbpPeerStatsAccsNotRecorded,
                dbpPeerStatsAccRetrans,
                dbpPeerStatsTotalRetrans,
                dbpPeerStatsAccPendReqstsOut,
                dbpPeerStatsAccReqstsDropped,
                dbpPeerStatsHByHDropMessages,
                dbpPeerStatsEToEDupMessages,
                dbpPeerStatsUnknownTypes,
                dbpPeerStatsProtocolErrors,
                dbpPeerStatsTransientFailures,
                dbpPeerStatsPermanentFailures,
                dbpPeerStatsTransportDown
    }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing statistics
             of the Diameter peers."
    ::= { diameterBaseProtocolMIBGroups 3 }

dbpNotificationsGroup NOTIFICATION-GROUP
   NOTIFICATIONS    {
                    dbpProtocolErrorNotif,
                    dbpTransientFailureNotif,
                    dbpPermanentFailureNotif,
                    dbpPeerConnectionDownNotif,
                    dbpPeerConnectionUpNotif
                    }
    STATUS          current
    DESCRIPTION
            "The set of notifications which an agent is required
             to implement."
    ::= { diameterBaseProtocolMIBGroups 4 }

dbpNotifCfgGroup OBJECT-GROUP
    OBJECTS         {
                    dbpProtocolErrorNotifEnabled,
                    dbpTransientFailureNotifEnabled,
                    dbpPermanentFailureNotifEnabled,
                    dbpPeerConnectionDownNotifEnabled,
                    dbpPeerConnectionUpNotifEnabled
                    }
    STATUS          current
```

```
    DESCRIPTION
            "A collection of objects providing configuration for
            base protocol notifications."
    ::= { diameterBaseProtocolMIBGroups 5 }

dbpLocalCfgSkippedGroup OBJECT-GROUP
    OBJECTS         {
                    dbpLocalId,
                    dbpLocalTcpListenPort,
                    dbpLocalSctpListenPort,
                    dbpLocalStatsTotalPacketsIn,
                    dbpLocalStatsTotalPacketsOut,
                    dbpLocalStatsTotalUpTime,
                    dbpLocalResetTime,
                    dbpLocalConfigReset,
                    dbpLocalApplStorageType,
                    dbpLocalApplRowStatus,
                    dbpAppAdvToPeerServices,
                    dbpAppAdvToPeerStorageType,
                    dbpAppAdvToPeerRowStatus
                    }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing configuration common
            to the peer."
    ::= { diameterBaseProtocolMIBGroups 6 }

dbpLocalStatsSkippedGroup OBJECT-GROUP
    OBJECTS         {
                    dbpLocalStatsTotalPacketsIn,
                    dbpLocalStatsTotalPacketsOut,
                    dbpLocalStatsTotalUpTime,
                    dbpLocalResetTime,
                    dbpLocalConfigReset
                    }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing statistics common
            to the peer."
    ::= { diameterBaseProtocolMIBGroups 7 }

dbpPeerCfgSkippedGroup OBJECT-GROUP
    OBJECTS         { cdbpAppAdvFromPeerType }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing configuration for
            Diameter peers."
    ::= { diameterBaseProtocolMIBGroups 8 }
```

```
dbpPeerStatsSkippedGroup OBJECT-GROUP
    OBJECTS         {
                    dbpPeerStatsDWCurrentStatus,
                    dbpPeerStatsDWReqTimer,
                    dbpPeerStatsRedirectEvents,
                    dbpPeerStatsAccDupRequests,
                    dbpPeerStatsETоEDupMessages
                    }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing statistics
            of Diameter peers."
    ::= { diameterBaseProtocolMIBGroups 9 }

dbpRealmCfgSkippedGroup OBJECT-GROUP
    OBJECTS         {
                    dbpRealmKnownPeers,
                    dbpRealmKnownPeersChosen
                    }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing configuration for
            realm message routing."
    ::= { diameterBaseProtocolMIBGroups 10 }

dbpRealmStatsSkippedGroup OBJECT-GROUP
    OBJECTS         {
                    dbpRealmMessageRouteRealm,
                    dbpRealmMessageRouteApp,
                    dbpRealmMessageRouteType,
                    dbpRealmMessageRouteAction,
                    dbpRealmMessageRouteACRsIn,
                    dbpRealmMessageRouteACRsOut,
                    dbpRealmMessageRouteACAsIn,
                    dbpRealmMessageRouteACAsOut,
                    dbpRealmMessageRouteRARsIn,
                    dbpRealmMessageRouteRARsOut,
                    dbpRealmMessageRouteRAAsIn,
                    dbpRealmMessageRouteRAAsOut,
                    dbpRealmMessageRouteSTRsIn,
                    dbpRealmMessageRouteSTRsOut,
                    dbpRealmMessageRouteSTAsIn,
                    dbpRealmMessageRouteSTAsOut,
                    dbpRealmMessageRouteASRsIn,
                    dbpRealmMessageRouteASRsOut,
                    dbpRealmMessageRouteASAsIn,
                    dbpRealmMessageRouteASAsOut,
                    dbpRealmMessageRouteAccRetrans,
```

```
                    dbpRealmMessageRouteAccDupReqsts,
                    dbpRealmMessageRoutePendReqstsOut,
                    dbpRealmMessageRouteReqstsDrop
                    }
    STATUS          current
    DESCRIPTION
            "A collection of objects providing statistics
            of realm message routing."
    ::= { diameterBaseProtocolMIBGroups 11 }

END
```

5. IANA Considerations

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
diameterBaseProtocolMIB	{ mib-2 XXX }

Editor's Note (to be removed prior to publication) The IANA is requested to assign a value for "XXX" under the 'mib-2' subtree and to record the assignment in the SMI Numbers registry. When the assignment has been made, the RFC Editor is asked to replace "XXX" (here and in the MIB module) with the assigned value and to remove this note.

6. Security Considerations

There are managed objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations.

There are several of managed objects in this MIB that may contain sensitive information. These are:

- diameterHostAddress
- diameterPeerServerAddress
- diameterPeerIpAddress

These can be used to determine the address of the Diameter host, and/or peers with which the host is communicating. This information could be useful in impersonating the host or peer.

It is important to control GET access to these objects and possibly to even encrypt the values of these object when sending them over the network via SNMP. Not all versions of SNMP provide features for such a secure environment.

SNMPv1 by itself is not a secure environment. Even if the network itself is secure (for example by using IPSec), there is no control as to who on the secure network is allowed to access and GET (read) the objects in this MIB.

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [RFC3414] and the View-based Access Control Model [RFC3415] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/ create/delete) them.

7. Contributors

This document is based upon and derived from work done by Jay Koehler, Mark Eklund, Hai Li and Subash Comerica.

8. Acknowledgements

Thanks to David Battle for his participation and suggestions in designing the table structures; Dan Romascanu, Kevin Lingle, Sumanth Mithra, Tolga Asveren, Tina Tsou, Mark Jones, John Loughney and Biswaranjan Panda for reviewing the MIB and making invaluable suggestions; and Greg Weber for his help in representing the MIB at IETF meetings.

9. References

9.1. Normative References

[I-D.ietf-dime-rfc3588bis]
Fajardo, V., Arkko, J., Loughney, J., and G. Zorn,

"Diameter Base Protocol", draft-ietf-dime-rfc3588bis-26 (work in progress), January 2011.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

[RFC4133] Bierman, A. and K. McCloghrie, "Entity MIB (Version 3)", RFC 4133, August 2005.

9.2. Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, December 2002.

[RFC3415] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, December 2002.

Authors' Addresses

Glen Zorn
Network Zen
227/358 Thanon Sanphawut
Bang Na, Bangkok 10260
Thailand

Phone: +66 (0) 87 040-4617
Email: gwz@net-zen.net

Subash T. Comerica
Juniper Networks
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA

Phone: +1 (408) 936-0830
Email: scomerica@juniper.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 19, 2010

G. Zorn, Ed.
Network Zen
S. Comerica
Cisco Systems

January 15, 2010

Diameter Credit Control Application MIB
draft-ietf-dime-diameter-cc-appl-mib-03.txt

Abstract

Along with providing support for certain basic authentication, authorization and accounting functions, the Diameter base protocol is intended to provide a framework for AAA applications.

This document defines the Management Information Base (MIB) module which describes the minimum set of objects needed to manage an implementation of the Diameter Credit Control application.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on July 19, 2010.

Table of Contents

1. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58 ([RFC2578], [RFC2579], [RFC2580]). In particular, it describes managed objects used for managing the Diameter Credit Control Application [RFC4006].

Discussion of this draft may be directed to the dime Working Group of the IETF (dime@ietf.org)..

2. Requirements Language

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].

3. Overview

The base Diameter protocol [RFC3588] is never used alone; it is always extended for a particular application.

This MIB defines objects supporting the management of the Diameter Credit Control Application protocol as described in [RFC4006]. The MIB specification for the Diameter base protocol [I-D.ietf-dime-diameter-base-protocol-mib] SHOULD be implemented prior to the implementation of this MIB.

4. Diameter Credit Control Application MIB Definitions

```
DIAMETER-CC-APPLICATION-MIB DEFINITIONS ::= BEGIN

   IMPORTS
       MODULE-IDENTITY,
       OBJECT-TYPE,
       Unsigned32,
```

```
       Counter32,
       mib-2
           FROM SNMPv2-SMI -- [RFC2578]

       MODULE-COMPLIANCE,
       OBJECT-GROUP
           FROM SNMPv2-CONF -- [RFC2580]
       StorageType,
       RowStatus
           FROM SNMPv2-TC -- [RFC2579]
       InetAddressType,
       InetAddress
           FROM INET-ADDRESS-MIB -- [RFC4001]
       SnmpAdminString
           FROM SNMP-FRAMEWORK-MIB; -- [RFC3411]

   diameterCCAMIB MODULE-IDENTITY
         LAST-UPDATED "201001150000Z" -- 15 January 2010
         ORGANIZATION "IETF dime Working Group."
         CONTACT-INFO
             "Subash Comerica
              Cisco Systems
              Global Development Centre, Prestige Waterford
              No. 9 Brunton Road
              BGL3/MZ/
              Bangalore, Karnataka  560025
              India
              Phone: +91 80 4103 6427
              Email: subashtc@cisco.com"
          DESCRIPTION
              "The MIB module for entities implementing the
              Diameter Credit Control Application, RFC 4006.

              Copyright (C) The Internet Society (2010).  This initial
              version of this MIB module was published in RFC yyyy;
              for full legal notices see the RFC itself.  Supplementary
              information may be available on
              http://www.ietf.org/copyrights/ianamib.html."

   -- RFC Ed.: replace yyyy with actual RFC number and remove this note

         REVISION "201001150000Z" -- 15 January 2010
         DESCRIPTION "Initial version as published in RFC yyyy"

   -- RFC Ed.: replace yyyy with actual RFC number and remove this note

         ::= { mib-2 XXX }
```

```
   -- RFC Ed.: replace XXX with value assigned by IANA
   --          and remove this note

   -- Top-Level Components of this MIB.
   diameterCcAppMIB             OBJECT IDENTIFIER ::=
                                        { diameterCCAMIB 2 }
   diameterCcAppTraps           OBJECT IDENTIFIER ::=
                                        { diameterCcAppMIB 0 }
   diameterCcAppObjects         OBJECT IDENTIFIER ::=
                                        { diameterCcAppMIB 1 }
   diameterCcAppConform         OBJECT IDENTIFIER ::=
                                        { diameterCcAppMIB 2 }

   dccaHostCfgs         OBJECT IDENTIFIER ::= { diameterCcAppObjects 1 }
   dccaPeerCfgs        OBJECT IDENTIFIER ::= { diameterCcAppObjects 2 }
   dccaPeerStats        OBJECT IDENTIFIER ::= { diameterCcAppObjects 3 }

   dccaHostID OBJECT-TYPE
          SYNTAX     SnmpAdminString
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "The implementation identification string for
                 the Diameter software in use on the system,
                 for example; 'diameterd'"
          ::= { dccaHostCfgs 1 }

   dccaHostIpAddrTable OBJECT-TYPE
          SYNTAX          SEQUENCE OF DccaHostIpAddrEntry
          MAX-ACCESS      not-accessible
          STATUS          current
          DESCRIPTION
               "The table listing the Diameter
               Credit Control host's IP Addresses."
          ::= { dccaHostCfgs 2 }

   dccaHostIpAddrEntry OBJECT-TYPE
          SYNTAX          DccaHostIpAddrEntry
          MAX-ACCESS      not-accessible
          STATUS          current
          DESCRIPTION
               "A row entry representing a Diameter
               Credit Control host IP Address."
          INDEX           { dccaHostIpAddrIndex }
          ::= { dccaHostIpAddrTable 1 }

   DccaHostIpAddrEntry ::= SEQUENCE {
          dccaHostIpAddrIndex Unsigned32,
```

```
        dccaHostIpAddrType  InetAddressType,
        dccaHostIpAddress   InetAddress
   }

   dccaHostIpAddrIndex OBJECT-TYPE
        SYNTAX       Unsigned32 (1..4294967295 )
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
             "A number uniquely identifying the number
             of IP Addresses supported by this Diameter
             Credit Control host."
        ::= { dccaHostIpAddrEntry 1 }

   dccaHostIpAddrType OBJECT-TYPE
        SYNTAX       InetAddressType
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
             "The type of internet address stored
             in dccaHostIpAddress."
       ::= { dccaHostIpAddrEntry 2 }

   dccaHostIpAddress OBJECT-TYPE
        SYNTAX       InetAddress
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
             "The IP-Address of the host, which is of the
             type specified in dccaHostIpAddrType."
       ::= { dccaHostIpAddrEntry 3 }

   dccaPeerTable OBJECT-TYPE
        SYNTAX     SEQUENCE OF DccaPeerEntry
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
              "The table listing information regarding
               the discovered or configured Diameter
               Credit Control peers."
        ::= { dccaPeerCfgs 1 }

   dccaPeerEntry OBJECT-TYPE
        SYNTAX     DccaPeerEntry
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
              "A row entry representing a discovered
```

```
                or configured Diameter Credit Control
                peer."
        INDEX       { dccaPeerIndex }
        ::= { dccaPeerTable 1 }

   DccaPeerEntry ::= SEQUENCE {
        dccaPeerIndex              Unsigned32,
        dccaPeerId                 SnmpAdminString,
        dccaPeerFirmwareRevision   Unsigned32,
        dccaPeerStorageType        StorageType,
        dccaPeerRowStatus             RowStatus }

   dccaPeerIndex OBJECT-TYPE
        SYNTAX     Unsigned32 (1..4294967295)
        MAX-ACCESS not-accessible
        STATUS     current
        DESCRIPTION
              "A number uniquely identifying each Diameter
               Credit Control peer with which this host
               communicates."
        ::= { dccaPeerEntry  1 }

   dccaPeerId OBJECT-TYPE
        SYNTAX     SnmpAdminString
        MAX-ACCESS read-create
        STATUS     current
        DESCRIPTION
              "The server identifier for the Diameter
               Credit Control peer."
        ::= { dccaPeerEntry  2 }

   dccaPeerFirmwareRevision OBJECT-TYPE
        SYNTAX     Unsigned32 (1..4294967295)
        MAX-ACCESS read-create
        STATUS     current
        DESCRIPTION
              "Firmware revision of peer.  If no firmware
               revision, the revision of the Diameter
               Credit Control software
               module may be reported instead."
        ::= { dccaPeerEntry 3 }

   dccaPeerStorageType OBJECT-TYPE
       SYNTAX        StorageType
       MAX-ACCESS    read-create
       STATUS        current
       DESCRIPTION
               "The storage type for this conceptual row. None
```

```
                of the columnar objects is writable when the
                conceptual row is permanent."
        REFERENCE
                "Textual Conventions for SMIv2, Section 2."
        DEFVAL        { nonVolatile }
        ::= { dccaPeerEntry 4 }

    dccaPeerRowStatus OBJECT-TYPE
           SYNTAX     RowStatus
           MAX-ACCESS read-create
           STATUS     current
           DESCRIPTION
                "The status of this conceptual row.
                To create a row in this table, a manager must
                set this object to either createAndGo(4) or
                createAndWait(5).

                Until instances of all corresponding columns
                are appropriately configured, the value of the
                corresponding instance of the
                dccaPeerRowStatus column is 'notReady'.

                In particular, a newly created row cannot be
                made active until the corresponding dccaPeerId
                has been set.

                dccaPeerId may not be modified while the
                value of this object is active(1):
                An attempt to set these objects while the value
                of dccaPeerRowStatus is active(1) will result
                in an inconsistentValue error.

                Entries in this table with dccaPeerRowStatus
                equal to active(1) remain in the table until
                destroyed.

                Entries in this table with dccaPeerRowStatus
                equal to values other than active(1) will be
                destroyed after timeout (5 minutes).

                If a dccaPeerId being created via SNMP already
                exists in another active dccaPeerEntry, then a
                newly created row
                cannot be made active until the original row
                with the dccaPeerId value is destroyed.

                Upon reload, dccaPeerIndex values may be
                changed."
```

```
          ::= { dccaPeerEntry 5 }

   dccaPeerVendorTable OBJECT-TYPE
       SYNTAX          SEQUENCE OF DccaPeerVendorEntry
       MAX-ACCESS      not-accessible
       STATUS          current
       DESCRIPTION
               "The table listing the Vendor IDs
               supported by the peer."
       ::= { dccaPeerCfgs 2 }

   dccaPeerVendorEntry OBJECT-TYPE
       SYNTAX          DccaPeerVendorEntry
       MAX-ACCESS      not-accessible
       STATUS          current
       DESCRIPTION
               "A row entry representing a
               Vendor ID supported by the peer."
       INDEX           {
                           dccaPeerIndex,
                           dccaPeerVendorIndex
                       }
       ::= { dccaPeerVendorTable 1 }

   DccaPeerVendorEntry ::= SEQUENCE {
           dccaPeerVendorIndex       Unsigned32,
           dccaPeerVendorId          Unsigned32,
           dccaPeerVendorStorageType StorageType,
           dccaPeerVendorRowStatus   RowStatus
   }

   dccaPeerVendorIndex OBJECT-TYPE
       SYNTAX          Unsigned32 (1..4294967295 )
       MAX-ACCESS      not-accessible
       STATUS          current
       DESCRIPTION
               "A number uniquely identifying the Vendor
               ID supported by the peer."
       ::= { dccaPeerVendorEntry 1 }

   dccaPeerVendorId OBJECT-TYPE
       SYNTAX          Unsigned32
       MAX-ACCESS      read-create
       STATUS          current
       DESCRIPTION
               "The active Vendor IDs used for peer
                connections."
       ::= { dccaPeerVendorEntry 2 }
```

```
   dccaPeerVendorStorageType OBJECT-TYPE
       SYNTAX          StorageType
       MAX-ACCESS      read-create
       STATUS          current
       DESCRIPTION
               "The storage type for this conceptual row.  An
                agent implementing the table must allow adding
                dccaPeerVendorId into the table. None of the
                columnar objects is writable
                when the conceptual row is permanent."
       REFERENCE
                "Textual Conventions for SMIv2, Section 2."
       DEFVAL          { nonVolatile }
       ::= { dccaPeerVendorEntry 3 }

   dccaPeerVendorRowStatus OBJECT-TYPE
       SYNTAX          RowStatus
       MAX-ACCESS      read-create
       STATUS          current
       DESCRIPTION
               "The status of this conceptual row.

                To create a row in this table, a manager must
                set this object to either createAndGo(4) or
                createAndWait(5).

                Until instances of all corresponding columns
                are appropriately configured, the value of the
                corresponding instance of the
                dccaPeerVendorRowStatus column is 'notReady'.

                In particular, a newly created row cannot be
                made active until the corresponding
                dccaPeerVendorId has been set.

                dccaPeerVendorId may not be modified while the
                value of this object is active(1):
                An attempt to set these objects while the
                value of dccaPeerVendorRowStatus is active(1)
                will result in an inconsistentValue error.

                Entries in this table with
                dccaPeerVendorRowStatus equal to
                active(1) remain in the table until destroyed.

                Entries in this table with
                dccaPeerVendorRowStatus equal to
                values other than active(1) will be destroyed
```

```
                after timeout (5 minutes).

                If the peer vendor id being created via SNMP
                already exists
                in another active dccaPeerVendorEntry,
                then a newly
                created row cannot be made active until the
                original row with the peer vendor id value is
                destroyed.

                Upon reload, dccaPeerVendorIndex values may be
                changed."
        ::= { dccaPeerVendorEntry 4 }


    -- per-peer statistics

    dccaPerPeerStatsTable OBJECT-TYPE
           SYNTAX     SEQUENCE OF DccaPerPeerStatsEntry
           MAX-ACCESS not-accessible
           STATUS     current
           DESCRIPTION
                 "The table listing the Diameter
                  Credit Control per-peer Statistics."
           ::= { dccaPeerStats 1 }

    dccaPerPeerStatsEntry OBJECT-TYPE
           SYNTAX     DccaPerPeerStatsEntry
           MAX-ACCESS not-accessible
           STATUS     current
           DESCRIPTION
                 "A row entry representing a Diameter
                  Credit Control Peer."
           INDEX      { dccaPeerIndex }
           ::= { dccaPerPeerStatsTable 1 }

    DccaPerPeerStatsEntry ::= SEQUENCE {
           dccaPerPeerStatsCCRIn                      Counter32,
           dccaPerPeerStatsCCROut                     Counter32,
           dccaPerPeerStatsCCRDropped                 Counter32,
           dccaPerPeerStatsCCAIn                      Counter32,
           dccaPerPeerStatsCCAOut                     Counter32,
           dccaPerPeerStatsCCADropped                 Counter32,
           dccaPerPeerStatsRARIn                      Counter32,
           dccaPerPeerStatsRARDropped                 Counter32,
           dccaPerPeerStatsRAAOut                     Counter32,
           dccaPerPeerStatsRAADropped                 Counter32,
           dccaPerPeerStatsSTROut                     Counter32,
```

```
        dccaPerPeerStatsSTRDropped                  Counter32,
        dccaPerPeerStatsSTAIn                       Counter32,
        dccaPerPeerStatsSTADropped                  Counter32,
        dccaPerPeerStatsAAROut                      Counter32,
        dccaPerPeerStatsAARDropped                  Counter32,
        dccaPerPeerStatsAAAIn                       Counter32,
        dccaPerPeerStatsAAADropped                  Counter32,
        dccaPerPeerStatsASRIn                       Counter32,
        dccaPerPeerStatsASRDropped                  Counter32,
        dccaPerPeerStatsASAOut                      Counter32,
        dccaPerPeerStatsASADropped                  Counter32 }

   dccaPerPeerStatsCCRIn OBJECT-TYPE
          SYNTAX      Counter32
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                "Number of Diameter Credit-Control-Request
                (CCR) messages received, per peer."
          ::= { dccaPerPeerStatsEntry 2 }

   dccaPerPeerStatsCCROut OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter Credit-Control-Request (CCR)
                 messages sent, per peer."
          ::= { dccaPerPeerStatsEntry 3 }

   dccaPerPeerStatsCCRDropped OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter Credit-Control-Request (CCR)
                 messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 4 }

   dccaPerPeerStatsCCAIn OBJECT-TYPE
          SYNTAX      Counter32
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                "Number of Diameter Credit-Control-Answer (CCA)
                 messages received, per peer."
          ::= { dccaPerPeerStatsEntry 5 }
```

```
   dccaPerPeerStatsCCAOut OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter Credit-Control-Answer (CCA)
                 messages sent, per peer."
          ::= { dccaPerPeerStatsEntry 6 }

   dccaPerPeerStatsCCADropped OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter Credit-Control-Answer (CCA)
                 messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 7 }

   dccaPerPeerStatsRARIn OBJECT-TYPE
          SYNTAX      Counter32
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                "Number of Diameter Re-Auth-Request (RAR)
                 messages received, per peer."
          ::= { dccaPerPeerStatsEntry 8 }

   dccaPerPeerStatsRARDropped OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter Re-Auth-Request (RAR)
                 messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 9 }

   dccaPerPeerStatsRAAOut OBJECT-TYPE
          SYNTAX      Counter32
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                "Number of Diameter Re-Auth-Answer (RAA)
                 messages transmitted, per peer."
          ::= { dccaPerPeerStatsEntry 10 }

   dccaPerPeerStatsRAADropped OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
```

```
          STATUS     current
          DESCRIPTION
                "Number of Diameter Re-Auth-Answer (RAA)
                 messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 11 }

   dccaPerPeerStatsSTROut OBJECT-TYPE
          SYNTAX      Counter32
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                "Number of Diameter
                 Session-Termination-Request (STR)
                 messages transmitted, per peer."
          ::= { dccaPerPeerStatsEntry 12 }

   dccaPerPeerStatsSTRDropped OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter
                 Session-Termination-Request (STR)
                 messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 13 }

   dccaPerPeerStatsSTAIn OBJECT-TYPE
          SYNTAX      Counter32
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                "Number of Diameter
                 Session-Termination-Answer (STA)
                 messages received, per peer."
          ::= { dccaPerPeerStatsEntry 14 }

   dccaPerPeerStatsSTADropped OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter
                 Session-Termination-Answer (STA)
                 messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 15 }

   dccaPerPeerStatsAAROut OBJECT-TYPE
          SYNTAX      Counter32
```

```
            MAX-ACCESS  read-only
            STATUS      current
            DESCRIPTION
                  "Number of Diameter AA-Request (AAR)
                   messages transmitted, per peer."
            ::= { dccaPerPeerStatsEntry 16 }

      dccaPerPeerStatsAARDropped OBJECT-TYPE
            SYNTAX     Counter32
            MAX-ACCESS read-only
            STATUS     current
            DESCRIPTION
                  "Number of Diameter AA-Request (AAR)
                   messages dropped, per peer."
            ::= { dccaPerPeerStatsEntry 17 }

      dccaPerPeerStatsAAAIn OBJECT-TYPE
            SYNTAX      Counter32
            MAX-ACCESS  read-only
            STATUS      current
            DESCRIPTION
                  "Number of Diameter AA-Answer (AAA)
                   messages received, per peer."
            ::= { dccaPerPeerStatsEntry 18 }

      dccaPerPeerStatsAAADropped OBJECT-TYPE
            SYNTAX     Counter32
            MAX-ACCESS read-only
            STATUS     current
            DESCRIPTION
                  "Number of Diameter AA-Answer (AAA)
                   messages dropped, per peer."
            ::= { dccaPerPeerStatsEntry 19 }

      dccaPerPeerStatsASRIn OBJECT-TYPE
            SYNTAX      Counter32
            MAX-ACCESS  read-only
            STATUS      current
            DESCRIPTION
                  "Number of Diameter Abort-Session-Request
                   (ASR) messages received, per peer."
            ::= { dccaPerPeerStatsEntry 20 }

      dccaPerPeerStatsASRDropped OBJECT-TYPE
            SYNTAX     Counter32
            MAX-ACCESS read-only
            STATUS     current
            DESCRIPTION
```

```
                "Number of Diameter Abort-Session-Request
                 (ASR) messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 21 }

   dccaPerPeerStatsASAOut OBJECT-TYPE
          SYNTAX      Counter32
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                "Number of Diameter Abort-Session-Answer
                 (ASA) messages transmitted, per peer."
          ::= { dccaPerPeerStatsEntry 22 }

   dccaPerPeerStatsASADropped OBJECT-TYPE
          SYNTAX     Counter32
          MAX-ACCESS read-only
          STATUS     current
          DESCRIPTION
                "Number of Diameter Abort-Session-Answer
                 (ASA) messages dropped, per peer."
          ::= { dccaPerPeerStatsEntry 23 }


   -- Conformance dccaMIBCompliances

   dccaMIBCompliances
     OBJECT IDENTIFIER ::= { diameterCcAppConform 1 } dccaMIBGroups
     OBJECT IDENTIFIER ::= { diameterCcAppConform 2 }

   -- Compliance Statements

   dccaMIBCompliance MODULE-COMPLIANCE
          STATUS     current
          DESCRIPTION
               "The compliance statement for Diameter Credit
                Control application entities."
          MODULE -- this module
          MANDATORY-GROUPS { dccaPeerStatsGroup }

          GROUP
               dccaHostCfgGroup
          DESCRIPTION
               "This group is only mandatory for a system that
               supports Local DCCA Host configuration."

          GROUP
               dccaPeerCfgGroup
          DESCRIPTION
```

```
              "This group is only mandatory for a system that
              supports DCCA Peer configuration."

          ::= { dccaMIBCompliances 1 }

   -- Units of Conformance

   dccaHostCfgGroup OBJECT-GROUP
         OBJECTS {
                 dccaHostIpAddrType,
                 dccaHostIpAddress,
                 dccaHostID
         }
         STATUS  current
         DESCRIPTION
               "A collection of objects providing
                configuration common to the server."
         ::= { dccaMIBGroups 1 }

   dccaPeerCfgGroup OBJECT-GROUP
         OBJECTS {
                 dccaPeerId,
                 dccaPeerVendorId,
                 dccaPeerStorageType,
                 dccaPeerVendorStorageType,
                 dccaPeerFirmwareRevision,
                 dccaPeerRowStatus,
                 dccaPeerVendorRowStatus
         }
         STATUS  current
         DESCRIPTION
               "A collection of objects providing peer
                configuration common to the server."
       ::= { dccaMIBGroups 2 }

   dccaPeerStatsGroup OBJECT-GROUP
         OBJECTS {
                 dccaPerPeerStatsCCRIn,
                 dccaPerPeerStatsCCROut,
                 dccaPerPeerStatsCCRDropped,
                 dccaPerPeerStatsCCAIn,
                 dccaPerPeerStatsCCAOut,
                 dccaPerPeerStatsCCADropped,
                 dccaPerPeerStatsRARIn,
                 dccaPerPeerStatsRARDropped,
                 dccaPerPeerStatsRAAOut,
                 dccaPerPeerStatsRAADropped,
                 dccaPerPeerStatsSTROut,
```

```
                dccaPerPeerStatsSTRDropped,
                dccaPerPeerStatsSTAIn,
                dccaPerPeerStatsSTADropped,
                dccaPerPeerStatsAAROut,
                dccaPerPeerStatsAARDropped,
                dccaPerPeerStatsAAAIn,
                dccaPerPeerStatsAAADropped,
                dccaPerPeerStatsASRIn,
                dccaPerPeerStatsASRDropped,
                dccaPerPeerStatsASAOut,
                dccaPerPeerStatsASADropped
        }
        STATUS    current
        DESCRIPTION
                "A collection of objects providing peer
                 statistics common to the server."
        ::= { dccaMIBGroups 3 }

    END
```

5. IANA Considerations

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
diameterCCAMIB	{ mib-2 XXX }

Editor's Note (to be removed prior to publication) The IANA is requested to assign a value for "XXX" under the 'mib-2' subtree and to record the assignment in the SMI Numbers registry. When the assignment has been made, the RFC Editor is asked to replace "XXX" (here and in the MIB module) with the assigned value and to remove this note.

6. Security Considerations

SNMPv1 by itself is not a secure environment. Even if the network itself is secure (for example by using IPSec), there is no control as to who on the secure network is allowed to access and GET (read) the objects in this MIB.

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [RFC3414] and the View-based Access

Control Model [RFC3415] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

7. Acknowledgements

Thanks to Sumanth Mithra and Biswaranjan Panda for helpful suggestions and feedback.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

[RFC4006] Hakala, H., Mattila, L., Koskinen, J-P., Stura, M., and J. Loughney, "Diameter Credit-Control Application", RFC 4006,

August 2005.

8.2. Informative References

[I-D.ietf-dime-diameter-base-protocol-mib]
Zorn, G. and S. Comerica, "Diameter Base Protocol MIB", draft-ietf-dime-diameter-base-protocol-mib-04 (work in progress), November 2009.

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, December 2002.

[RFC3415] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, December 2002.

Authors' Addresses

Glen Zorn (editor)
Network Zen
1463 East Republican Street, #358
Seattle, Washington 98112
USA

Email: gwz@net-zen.net

Subash Comerica
Cisco Systems
Global Development Centre, Prestige Waterford
No. 9 Brunton Road
BGL3/MZ/
Bangalore, Karnataka 560025
India

Phone: +91 80 4103 6427
Email: subashtc@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 12, 2013

J. Bournelle
L. Morand
Orange Labs
S. Decugis
INSIDE Secure
Q. Wu
Huawei
G. Zorn
Network Zen
March 11, 2013

Diameter Support for the EAP Re-authentication Protocol (ERP)
draft-ietf-dime-erp-17.txt

Abstract

The EAP Re-authentication Protocol (ERP) defines extensions to the Extensible Authentication Protocol (EAP) to support efficient re-authentication between the peer and an EAP Re-authentication (ER) server through a compatible authenticator. This document specifies Diameter support for ERP. It defines a new Diameter ERP application to transport ERP messages between an ER authenticator and the ER server, and a set of new AVPs that can be used to transport the cryptographic material needed by the re-authentication server.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 12, 2013.

Copyright Notice

Table of Contents

1. Introduction

Cao, et al. [RFC6696] defines the EAP Re-authentication Protocol (ERP). It consists of the following steps:

Bootstrapping

A root key for re-authentication is derived from the Extended Master Session Key (EMSK) created during EAP authentication [RFC5295]. This root key is transported from the EAP server to the ER server.

Re-authentication

A one-round-trip exchange between the peer and the ER server, resulting in mutual authentication. To support the EAP reauthentication functionality, ERP defines two new EAP codes - EAP-Initiate and EAP-Finish.

This document defines how Diameter transports the ERP messages during the re-authentication process. For this purpose, we define a new Application Identifier for ERP, and re-use the Diameter EAP commands (DER/DEA).

This document also discusses the distribution of the root key during bootstrapping, in conjunction with either the initial EAP authentication (implicit bootstrapping) or the first ERP exchange (explicit bootstrapping). Security considerations for this key distribution are detailed in Section 7.4 of Salowey, et al. [RFC5295].

2. Terminology

This document uses terminology defined in Aboba, et al. [RFC3748], Salowey, et al. [RFC5295], Cao, et al. [RFC6696], and Eronen, et al. [RFC4072].

Following RFC 5295, the term "domain" herein refers to a key management domain unless otherwise qualified. Similarly, the terms "home domain", and "local domain" have the same meaning here as in RFC 6696.

The re-authentication Domain-Specific Root Key (rDSRK) is a re-authentication Root Key (rRK, [RFC6696]) derived from the DSRK instead of the EMSK.

"Root key" (RK) or "bootstrapping material" refers to the rRK or rDSRK derived from an EMSK, depending on whether the ER server is

located in the home or a foreign domain.

We use the notation "ERP/DER" and "ERP/DEA" in this document to refer to Diameter-EAP-Request and Diameter-EAP-Answer commands with the Application Id set to <Diameter ERP Application> (Section 10.1); the same commands are denoted "EAP/DER" and "EAP/DEA" when the Application Id in the message is set to <Diameter EAP Application> [RFC4072].

2.1. Requirements Language

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 [RFC2119].

3. Assumptions

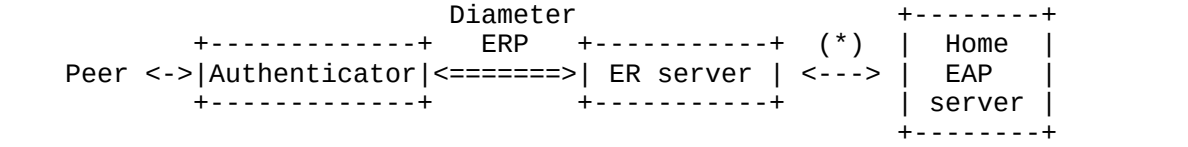
This document assumes the existence of at most one logical ER server entity in a given domain. If several physical servers are deployed for robustness, a replication mechanism must be deployed to synchronize the ERP state (e.g., root keys) between these servers. Any such replication mechanism is outside the scope of this document. If multiple ER servers are deployed in the domain, we assume that they can be used interchangeably. If multiple ER servers are deployed across multiple domains, we assume that only one ER server, topologically close to the peer, is involved in ERP, distance being measured in terms of Diameter hops.

This document also assumes the existence of at most one EAP server entity in the home domain. In case of multiple physical home EAP servers, if the ER server wants to reach the same home EAP server, the ER server SHOULD cache the Destination-Host AVP corresponding to the home EAP server it requests.

In general, it is assumed that key management domain names and Diameter realm names are identical for any given domain/realm.

4. Protocol Overview

The following figure illustrates the components involved in ERP and their interactions.



(*) Diameter EAP application, explicit bootstrapping scenario only.

Figure 1: Diameter ERP Overview.

The ER server is located either in the home domain (same as EAP server) or in the local domain (same as authenticator, when it differs from the home domain).

When the peer initiates an ERP exchange, the authenticator creates a Diameter-EAP-Request (DER) message [RFC4072]. The Application Id of the message is set to that of the Diameter ERP application Section 10.1 in the message. The generation of the ERP/DER message is detailed in Section 6.

If there is an ER server in the same domain as the authenticator (i.e., the local domain), Diameter routing MUST be configured so that this ERP/DER message reaches that server, even if the Destination-Realm is not the same as local domain.

If there is no local ER server, the message is routed according to its Destination-Realm AVP content, extracted from the realm component of the keyName-NAI attribute. As specified in RFC 6696, this realm is the home domain of the peer in the case of bootstrapping exchange ('B' flag is set in ERP message) or the domain of the bootstrapped ER server otherwise. .

If no ER server is available in the home domain either, the ERP/DER message cannot be delivered, and an error DIAMETER_UNABLE_TO_DELIVER MUST be generated as specified in RFC 6733 and returned to the authenticator. The authenticator MAY cache this information (with limited duration) to avoid further attempts to execute ERP with this realm. It MAY also fallback to full EAP authentication to authenticate the peer.

When an ER server receives the ERP/DER message, it searches its local database for a valid, unexpired root key matching the keyName part of the User-Name AVP. If such key is found, the ER server processes the ERP message as described in RFC 6696, then creates the ERP/DEA answer as described in Section 6. The rMSK is included in this answer.

Finally, the authenticator extracts the rMSK from the ERP/DEA as described in RFC 6696, and forwards the content of the EAP-Payload AVP, the EAP-Finish/Re-Auth message, to the peer.

The ER server may or may not possess the root key in its local database. If the EAP-Initiate/Re-Auth message has its 'B' flag set (Bootstrapping exchange) and the ER server possesses the root key, the ER server SHOULD respond directly to the peer that initiated the ERP exchange. Otherwise, the ER server SHOULD act as a proxy and forward the message to the home EAP server after changing its

Application Id to Diameter EAP and adding the ERP-RK-Request AVP to request the root key. See Section 5 for more detail on this process.

5. Bootstrapping the ER Server

The bootstrapping process involves the home EAP server and the ER server, but also impacts the peer and the authenticator. In ERP, the peer must derive the same keying material as the ER server. To achieve this, it must learn the domain name of the ER server. How this information is acquired is outside the scope of this specification, but the authenticator might be configured to advertize this domain name, especially in the case of re-authentication after a handover.

The bootstrapping of an ER server with a given root key happens either during the initial EAP authentication of the peer when the EMSK -- from which the root key is derived -- is created, during the first re-authentication, or sometime between those events. We only consider the first two possibilities in this specification, in the following sub-sections.

5.1. Bootstrapping During the Initial EAP authentication

Bootstrapping the ER server during the initial EAP authentication (also known as implicit bootstrapping) offers the advantage that the server is immediately available for re-authentication of the peer, thus minimizing the re-authentication delay. On the other hand, it is possible that only a small number of peers will use re-authentication in the local domain. Deriving and caching key material for all the peers (for example, for the peers that do not support ERP) is a waste of resources and should be avoided.

To achieve implicit bootstrapping, the ER server acts as a Diameter EAP Proxy , and Diameter routing MUST be configured so that Diameter EAP application messages are routed through this proxy. The figure bellow illustrates this mechanism.

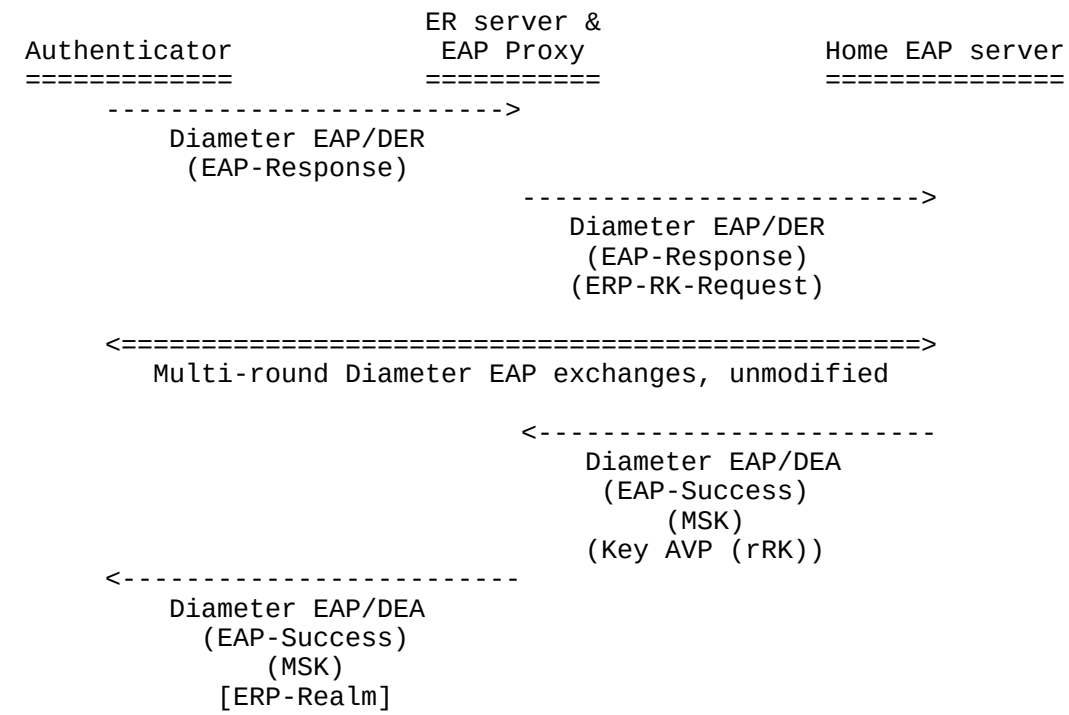


Figure 2: ERP Bootstrapping During Full EAP Authentication

The authenticator creates the first DER of the full EAP authentication and sends it to the ER server. The ER server proxies the first DER of the full EAP authentication and adds the ERP-RK-Request AVP inside, then forwards the request to the home EAP server.

If the home Diameter server does not support the Diameter ERP extensions, it simply ignores the ERP-RK-Request AVP and continues as specified in RFC 4072 [RFC4072]. If the server supports the ERP extensions, it saves the value of the ERP-Realm AVP found inside the ERP-RK-Request AVP, and continues with the EAP authentication. When the authentication completes, if it is successful and the EAP method has generated an EMSK, the server MUST derive the rRK as specified in RFC 6696, using the saved ERP realm name. It then includes the rRK inside a Key AVP (Section 8.3) with the Key-Type AVP set to rRK, before sending the DEA as usual.

When the ER server proxies a Diameter-EAP-Answer message with a Session-Id corresponding to a message to which it added an ERP-RK-Request AVP, and the Result-Code is DIAMETER_SUCCESS, it MUST examine the message and save and remove any Key AVP (Section 8.3) with Key-Type AVP set to rRK. If the message does not contain such Key AVP,

the ER server may cache the information that ERP is not possible for this session to avoid possible subsequent attempts. In any case, the information stored in ER server concerning a session should not have a lifetime greater than the EMSK for this session.

If the ER server is successfully bootstrapped, it should also add the ERP-Realm AVP after removing the Key AVP with Key-Type of rRK in the EAP/DEA message. This ERP-Realm information can be used by the authenticator to notify the peer that ER server is bootstrapped, and for which domain. How this information can be transmitted to the peer is outside the scope of this document. This information needs to be sent to the peer if both implicit and explicit bootstrapping mechanisms are possible, because the ERP message and the root key used for protecting this message are different in bootstrapping exchanges and non-bootstrapping exchanges.

5.2. Bootstrapping During the First Re-authentication

Bootstrapping the ER server during the first re-authentication (also known as explicit bootstrapping) is only needed when there is no ER server in the local domain and there is an ER server in the home domain. It is less resource-intensive, since the EMSK generated during initial EAP authentication is reused to derive root keys. On the other hand, the first re-authentication requires a one-round-trip exchange with the home EAP server, since the EMSK is generated during the initial EAP authentication and never leaves the home EAP server, which is less efficient than implicit bootstrapping.

The EAP-Initiate/Re-auth message is sent to the home ER server. The home ER server receives the ERP/DER message containing the EAP-Initiate/Re-Auth message with the 'B' flag set. It creates the new EAP/DER message using the received DRP/DER message and performs the following processing:

Set the Application Id in the header of the message to <Diameter EAP Application> [RFC4072]

Extract the ERP-RK-Request AVP from the ERP/DER message, which contains the name of the domain where the ER server is located and add it to the newly created ERP/DER message.

Then the newly created EAP/DER is sent and routed to the home Diameter EAP application server.

If the home Diameter EAP server does not support ERP extensions, EAP packets with an unknown ERP-specific code (EAP-Initiate) will not be understood. In such a case, the home Diameter EAP server MUST send an EAP/DEA with a Result-Code indicating a Permanent Failure (for

example, DIAMETER_ERROR_EAP_CODE_UNKNOWN or DIAMETER_UNABLE_TO_COMPLY). The Failed-AVP AVP MUST be included and contain a copy of the EAP-Payload AVP. Otherwise, it processes the DSRK request as described in RFC 6696. In particular, it includes the Domain- Name TLV attribute with the content from the ERP-Realm AVP. The server creates the EAP/DEA reply message [RFC4072] including an instance of the Key AVP (Section 8.3) with Key-Type AVP set to rRK and an instance of the Domain-Name TLV attribute with the content from the ERP-Realm AVP.

The ER server receives this EAP/DEA and proxies it as follows, in addition to standard proxy operations:

Set the Application Id back to Diameter ERP Application Id (Section 10.1)

Extract and cache the content of the Key AVP with Key-Type set to rRK, as described in Section 5.1).

The ERP/DEA message is then forwarded to the authenticator, that can use the rMSK as described in RFC 6696.

The figure below captures this proxy behavior:

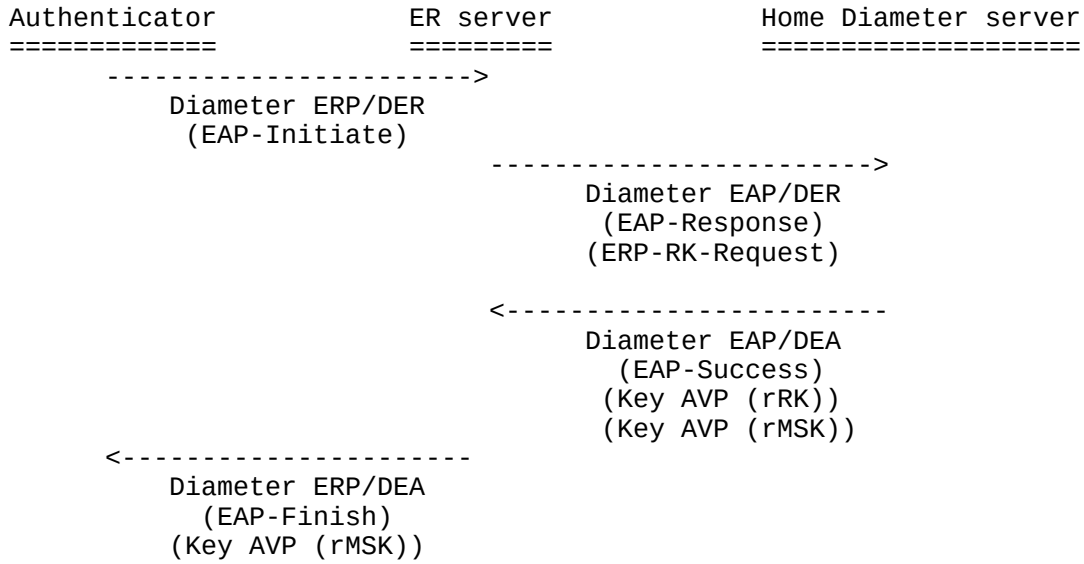


Figure 3: ERP Explicit Bootstrapping Message Flow

6. Re-Authentication

This section describes in detail a re-authentication exchange with an ER server that was previously bootstrapped. The following figure summarizes the re-authentication exchange.

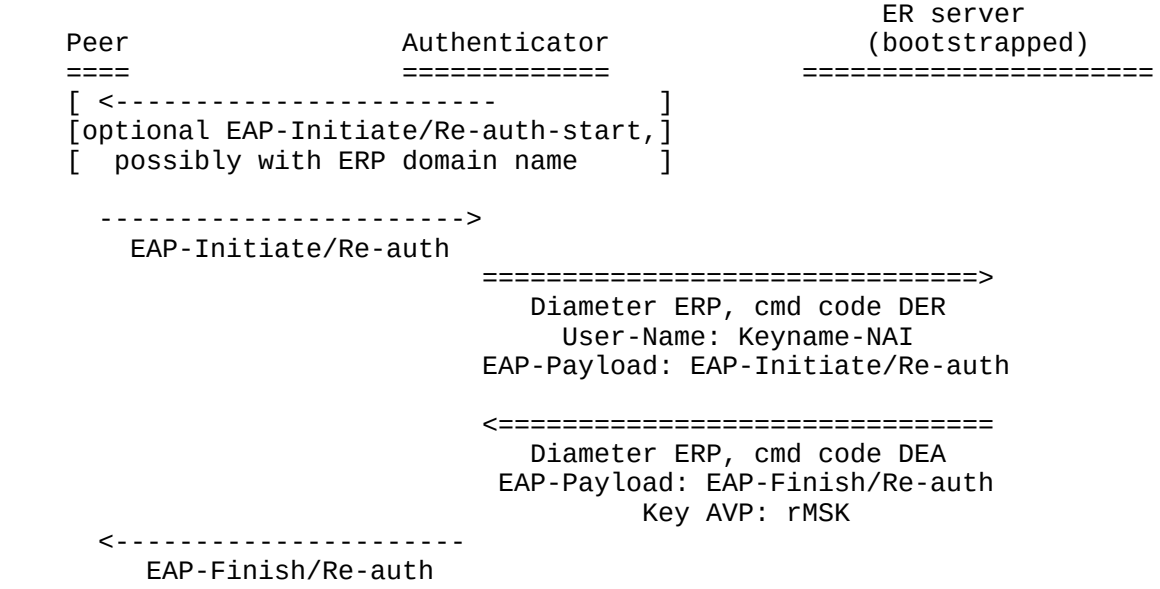


Figure 4: Diameter ERP Re-authentication Exchange

The peer sends an EAP-Initiate/Re-auth message to the ER server via the authenticator. Alternatively, the authenticator may send an EAP-Initiate/Re-auth-Start message to the peer to trigger the mechanism. In this case, the peer responds with an EAP-Initiate/Re-auth message.

If the authenticator does not support ERP (pure Diameter EAP [RFC4072] support), it discards the EAP packets with an unknown ERP-specific code (EAP-Initiate). The peer should fallback to full EAP authentication in this case.

When the authenticator receives an EAP-Initiate/Re-auth message from the peer, the message is processed as described in RFC 6696 with regard to the EAP state machine. It creates a Diameter ERP/DER message following the general process of Diameter EAP [RFC4072], with the following differences:

The Application Id in the header is set to <Diameter ERP> (code TBD1).

The value in Auth-Application-Id AVP is also set to <Diameter ERP>.

The keyName-NAI attribute from the ERP message is used to create the content of the User-Name and Destination-Realm AVPs.

The Auth-Request-Type AVP content is set to the appropriate value.

The EAP-Payload AVP contains the EAP-Initiate/Re-Auth message.

Then this ERP/DER message is sent as described in Section 4.

The ER server receives and processes this request as described in Section 4. It then creates an ERP/DEA message following the general process described in Eronen, et al. [RFC4072], with the following differences:

The Application Id in the header is set to <Diameter ERP> (code TBD1).

The value of the Auth-Application-Id AVP is also set to <Diameter ERP>.

The EAP-Payload AVP contains the EAP-Finish/Re-auth message.

If authentication is successful, an instance of the Key AVP containing the Re-authentication Master Session Key (rMSK) derived by ERP is included.

When the authenticator receives this ERP/DEA answer, it processes it as described in the Diameter EAP Application specification [RFC4072] and RFC 6696: the content of the EAP-Payload AVP is forwarded to the peer, and the contents of the Keying-Material AVP [RFC6734] is used as a shared secret for a secure association protocol specific to the lower-layer in use.

7. Application Id

We define a new Diameter application in this document, Diameter ERP Application, with an Application Id value of TBD1. Diameter nodes conforming to this specification in the role of ER server MUST advertise support by including an Auth-Application-Id AVP with a value of Diameter ERP in the Capabilities-Exchange-Request and Capabilities-Exchange-Answer commands [RFC6733].

The primary use of the Diameter ERP Application Id is to ensure proper routing of the messages, and that the nodes that advertise the support for this application do understand the new AVPs defined in

Section 8, although these AVP have the 'M' flag cleared.

8. AVPs

The following sub-sections discuss the AVPs used by the Diameter ERP application.

8.1. ERP-RK-Request AVP

The ERP-RK-Request AVP (AVP Code TBD2) is of type grouped AVP. This AVP is used by the ER server to indicate its willingness to act as ER server for a particular session.

This AVP has the M and V bits cleared.

```
ERP-RK-Request ::= < AVP Header: TBD2 >
                   { ERP-Realm }
                 * [ AVP ]
```

Figure 5: ERP-RK-Request ABNF

8.2. ERP-Realm AVP

The ERP-Realm AVP (AVP Code TBD3) is of type DiameterIdentity. It contains the name of the realm in which the ER server is located.

This AVP has the M and V bits cleared.

8.3. Key AVP

The Key AVP [RFC6734] is of type "Grouped" and is used to carry the rRK or rMSK and associated attributes. The usage of the Key AVP and its constituent AVPs in this application is specified in the following sub-sections.

8.3.1. Key-Type AVP

The value of the Key-Type AVP MUST be set to 1 for rRK or 2 for rMSK.

8.3.2. Keying-Material AVP

The Keying-Material AVP contains the rRK sent by the home EAP server to the ER server, in answer to a request containing an ERP-RK-Request AVP, or the rMSK sent by the ER server to the authenticator. How this material is derived and used is specified in RFC 6696.

8.3.3. Key-Name AVP

This AVP contains the EMSKname which identifies the keying material. The derivation of this name is specified in RFC 6696.

8.3.4. Key-Lifetime AVP

The Key-Lifetime AVP contains the lifetime of the keying material in seconds. It MUST NOT be greater than the remaining lifetime of the EMSK from which the material was derived.

9. Result-Code AVP Values

This section defines new Result-Code [RFC6733] values that MUST be supported by all Diameter implementations that conform to this specification.

9.1. Permanent Failures

Errors that fall within the Permanent Failures category are used to inform the peer that the request failed and SHOULD NOT be attempted again.

DIAMETER_ERROR_EAP_CODE_UNKNOWN (TBD4)

This error code is used by the Diameter server to inform the peer that the received EAP-PAYLOAD AVP contains an EAP packet with an unknown EAP code.

10. IANA Considerations

This document requires IANA registration of the following new elements in the Authentication, Authorization, and Accounting (AAA) Parameters registries [AAAPARAMS].

10.1. Diameter Application Identifier

This specification requires IANA to allocate a new value "Diameter ERP" (code: TBD1) in the "Application IDs" registry using the "Specification Required" policy [RFC5226]; see Section 11.3 of RFC 3588 [RFC3588] for further details.

10.2. New AVPs

This specification requires IANA to allocate new values from the "AVP Codes" registry according to the policy specified in Section 11.1 of Fajardo, et al. [RFC6733] for the following AVPs:

ERP-RK-Request (code: TBD2)

ERP-Realm (code: TBD3)

These AVPs are defined in Section 8.

10.3. New Permanent Failures Result-Code AVP Values

This specification requires IANA to allocate a new value from the "Result-Code AVP Values (code 268) - Permanent Failure" registry according to the policy specified in Section 11.3.2 of Fajardo, et al. [RFC6733] for the following Result-Code:

DIAMETER_ERROR_EAP_CODE_UNKNOWN (code: TBD4)

This result-code value is defined in Section 9.

11. Security Considerations

The security considerations from the following documents apply here:

- Eronen, et al. [RFC4072]
- Salowey, et al. [RFC5295]
- Cao, et al. [RFC6696]
- Fajardo, et al. [RFC6733]
- Zorn, et al. [RFC6734]

Because this application involves the transmission of sensitive data, including cryptographic keys, it MUST be protected using Transport Layer Security (TLS) [RFC5246], Datagram Transport Layer Security (DTLS) [RFC6347] or IP Encapsulating Security Payload (ESP) [RFC4303]. If TLS or DTLS is used, the bulk encryption algorithm negotiated MUST be non-null. If ESP is used, the encryption algorithm MUST be non-null.

12. Contributors

Hannes Tschofenig wrote the initial draft of this document.

Lakshminath Dondeti contributed to the early versions of the document.

13. Acknowledgements

Hannes Tschofenig, Zhen Cao, Benoit Claise, Elwyn Davies, Menachem Dodge, Vincent Roca, Stephen Farrell, Sean Turner, Pete Resnick, Russ Housley, Martin Stiemerling and Jouni Korhonen provided useful reviews.

Vidya Narayanan reviewed a rough draft version of the document and found some errors.

Many thanks to these people!

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5295] Salowey, J., Dondeti, L., Narayanan, V., and M. Nakhjiri, "Specification for the Derivation of Root Keys from an Extended Master Session Key (EMSK)", RFC 5295, August 2008.

[RFC6696] Cao, Z., He, B., Shi, Y., Wu, Q., and G. Zorn, "EAP Extensions for the EAP Re-authentication Protocol (ERP)", RFC 6696, July 2012.

[RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

[RFC6734] Zorn, G., Wu, Q., and V. Cakulev, "Diameter Attribute-Value Pairs for Cryptographic Key Transport", RFC 6734, October 2012.

14.2. Informative References

[AAAPARAMS] Internet Assigned Numbers Authority, "Authentication, Authorization, and Accounting (AAA) Parameters", http://www.iana.org/assignments/aaa-parameters/.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

Authors' Addresses

Julien Bournelle
Orange Labs
38-40 rue du general Leclerc
Issy-Les-Moulineaux 92794
France

EMail: julien.bournelle@orange-ftgroup.com

Lionel Morand
Orange Labs
38-40 rue du general Leclerc
Issy-Les-Moulineaux 92794
France

EMail: lionel.morand@orange.com

Sebastien Decugis
INSIDE Secure
41 Parc Club du Golf
Aix-en-Provence 13856
France

Phone: +33 (0)4 42 39 63 00
EMail: sdecugis@freediameter.net

Qin Wu
Huawei Technologies Co., Ltd
Site B, Floor 12F, Huihong Mansion, No.91 Baixia Rd.
Nanjing 210001
China

EMail: sunseawq@huawei.com

Glen Zorn
Network Zen
227/358 Thanon Sanphawut
Bang Na, Bangkok 10260
Thailand

EMail: glenzorn@gmail.com

Diameter Maintenance and Extensions (DIME)
Internet-Draft
Updates: 3588 (if approved)
Intended status: Standards Track
Expires: February 4, 2012

M. Jones
Bridgewater Systems
J. Korhonen
Nokia Siemens Networks
L. Morand
Orange Labs

August 3, 2011

Diameter S-NAPTR Usage
draft-ietf-dime-extended-naptr-09

Abstract

The Diameter base protocol specifies mechanisms whereby a given realm may advertise Diameter nodes and the supported transport protocol. However, these mechanisms do not reveal the Diameter applications that each node supports. A peer outside the realm would have to perform a Diameter capability exchange with every node until it discovers one that supports the required application. This document updates RFC3588 "Diameter Base Protocol" and describes an improvement using an extended format for the Straightforward-Naming Authority Pointer (S-NAPTR) Application Service Tag that allows for discovery of the supported applications without doing Diameter capability exchange beforehand.

Requirements Language

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 [RFC2119].

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 4, 2012.

Table of Contents

1. Introduction

The Diameter base protocol [RFC3588] specifies three mechanisms for the Diameter peer discovery. One of these involves the Diameter implementation performing a Naming Authority Pointer (NAPTR) query [RFC3403] for a server in a particular realm. These NAPTR records provide a mapping from a domain, to the DNS Service Locator (SRV) record [RFC2782] or A/AAAA record [RFC1035][RFC3596] for contacting a server with the specific transport protocol in the NAPTR services field.

The extended NAPTR usage for Diameter peer discovery defined by this document is based on the Straightforward-NAPTR (S-NAPTR) Dynamic Delegation Discovery System (DDDS) Application defined in [RFC3958]. This document updates the Diameter peer discovery procedure described in Section 11.6 of [RFC3588] and defines S-NAPTR Application Service and Application Protocol Tag values that permit the discovery of Diameter peers that support a specific Diameter application and transport protocol.

2. Terminology

The Diameter base protocol specification (Section 1.4 of [RFC3588]) and the Straightforward-NAPTR (S-NAPTR) DDDS application (section 2.1 in [RFC3958]) define the terminology used in this document.

3. Extended NAPTR Service Field Format

The NAPTR Service Field format defined by the S-NAPTR DDDS application in [RFC3958] follows this Augmented Backus-Naur Form (ABNF, [RFC5234]):

```
service-parms = [ [app-service] *(":" app-protocol)]
app-service   = experimental-service  / iana-registered-service
app-protocol  = experimental-protocol / iana-registered-protocol
experimental-service      = "x-" 1*30ALPHANUMSYM
experimental-protocol     = "x-" 1*30ALPHANUMSYM
iana-registered-service   = ALPHA *31ALPHANUMSYM
iana-registered-protocol  = ALPHA *31ALPHANUMSYM
ALPHA         =  %x41-5A / %x61-7A   ; A-Z / a-z
DIGIT         =  %x30-39 ; 0-9
SYM           =  %x2B / %x2D / %x2E  ; "+" / "-" / "."
ALPHANUMSYM   =  ALPHA / DIGIT / SYM
; The app-service and app-protocol tags are limited to 32
; characters and must start with an alphabetic character.
; The service-parms are considered case-insensitive.
```

This specification refines the "iana-registered-service" tag definition for the discovery of Diameter agents supporting a specific Diameter application as defined below.

```
iana-registered-service =/ aaa-service
aaa-service             = "aaa+ap" appln-id
appln-id                = 1*10DIGIT
                          ; Application identifier expressed as
                          ; a decimal integer without leading
                          ; zeros.
```

The appln-id element is the Application Identifier used to identify a specific Diameter Application. The Diameter Application Identifier is a 32-bit unsigned integer and values are allocated by IANA as defined in [RFC3588].

This specification also refines the "iana-registered-protocol" tag definition for the discovery of Diameter agents supporting a specific Diameter transport protocol as defined below.

```
iana-registered-protocol =/ aaa-protocol /
aaa-protocol             = "diameter." aaa-transport
aaa-transport            = "tcp" / "sctp" / "tls.tcp"
```

The S-NAPTR Application Protocol tags defined by this specification MUST NOT be parsed in any way by the querying application or resolver. The delimiter (".") is present in the tag to improve readability and does not imply a structure or namespace of any kind. The choice of delimiter (".") for the Application Protocol tag follows the format of existing S-NAPTR Application Protocol tag registry entries but this does not imply that that it shares semantics with any other specifications that create registry entries with the same format.

The S-NAPTR Application Service and Protocol tags defined by this specification are unrelated to the IANA Service Name and Transport Protocol Port Number Registry (see [I-D.ietf-tsvwg-iana-ports]).

The maximum length of the NAPTR service field is 256 octets including one octet length field (see Section 4.1 of RFC 3403 and Section 3.3 of [RFC1035]).

3.1. IETF Standard Track Diameter Applications

A Diameter agent MUST be capable of using the extended S-NAPTR Application Service Tag for dynamic discovery of a Diameter agent supporting Standard Track applications. Therefore, every IETF Standard Track Diameter application MUST be associated with a "aaa-

service" tag formatted as defined in this specification and allocated in accordance with the IANA policy (see Section 7).

For example, a NAPTR service field value of:

'aaa+ap6:diameter.sctp'

Means that the Diameter node in the SRV or A/AAAA record supports the Diameter Session Initiation Protocol (SIP) Application ('6') and SCTP as the transport protocol.

3.2. Vendor-specific Diameter Applications

S-NAPTR Application Service and Application Protocol Tag values can also be used to discover Diameter peers that support a vendor-specific Diameter application. In this case, the vendor-specific Diameter application MUST be associated with a "aaa-service" tag formatted as defined in this specification and allocated in accordance with the IANA policy (see Section 7).

For example, a NAPTR service field value of:

'aaa+ap16777251:diameter.sctp'

Means that the Diameter node in the SRV or A/AAAA record supports the Diameter 3GPP S6a Application ('16777251') and SCTP as the transport protocol.

4. Backwards Compatibility

Domain Name System (DNS) administrators SHOULD also provision legacy RFC 3588 style NAPTR records [RFC3403] in order to guarantee backwards compatibility with legacy RFC 3588 compliant Diameter peers. If the DNS administrator provisions both extended S-NAPTR records as defined in this specification and legacy RFC 3588 NAPTR records, then the extended S-NAPTR records MUST have higher priority (e.g. lower order and/or preference values) than legacy NAPTR records.

5. Extended NAPTR-based Diameter Peer Discovery

The Diameter Peer Discovery principles are described in Section 5.2 of [RFC3588]. This specification updates the NAPTR query procedure in the Diameter peer discovery mechanism by allowing the querying node to determine which applications are supported by resolved Diameter peers.

The extended format NAPTR records provide a mapping from a domain to the SRV record or A/AAAA record for contacting a server supporting a specific transport protocol and Diameter application. The resource record will contain an empty regular expression and a replacement value, which is the SRV record or the A/AAAA record for that particular transport protocol.

The assumption for this mechanism to work is that the DNS administrator of the queried domain has first provisioned the DNS with extended format NAPTR entries. The steps below replace the NAPTR query procedure steps in Section 5.2 of [RFC3588].

a. The Diameter implementation performs a NAPTR query for a server in a particular realm. The Diameter implementation has to know in advance which realm to look for a Diameter agent in and which Application Identifier it is interested in. For example, the realm could be deduced from the Network Access Identifier (NAI) in the User-Name AVP or extracted from the Destination-Realm AVP.

b. If the returned NAPTR service fields contain entries formatted as "aaa+apX:Y" where "X" indicates the Application Identifier and "Y" indicates the supported transport protocol(s), the target realm supports the extended format for NAPTR-based Diameter peer discovery defined in this document.

   If "X" contains the required Application Identifier and "Y" matches a supported transport protocol, the Diameter implementation resolves the "replacement" field entry to a target host using the lookup method appropriate for the "flags" field.

   If "X" does not contain the required Application Identifier or "Y" does not match a supported transport protocol, the Diameter implementation abandons the peer discovery.

c. If the returned NAPTR service fields contain entries formatted as "aaa+apX" where "X" indicates the Application Identifier, the target realm supports the extended format for NAPTR-based Diameter peer discovery defined in this document.

   If "X" contains the required Application Identifier, the Diameter implementation resolves the "replacement" field entry to a target host using the lookup method appropriate for the "flags" field and attempts to connect using all supported transport protocols following the order specified in section 2.1 of [RFC3588].

If "X" does not contain the required Application Identifier, the Diameter implementation abandons the peer discovery.

d. If the returned NAPTR service fields contain entries formatted as "aaa:X" where "X" indicates the supported transport protocol(s), the target realm supports Diameter but does not support the extended format for NAPTR-based Diameter peer discovery defined in this document.

   If "X" matches a supported transport protocol, the Diameter implementation resolves the "replacement" field entry to a target host using the lookup method appropriate for the "flags" field.

e. If the returned NAPTR service fields contain entries formatted as "aaa", the target realm supports Diameter but does not support the extended format for NAPTR-based Diameter peer discovery defined in this document. The Diameter implementation resolves the "replacement" field entry to a target host using the lookup method appropriate for the "flags" field and attempts to connect using all supported transport protocols following the order specified in section 2.1 of [RFC3588].

f. If the target realm does not support NAPTR-based Diameter peer discovery, the client proceeds with the next peer discovery mechanism described in Section 5.2 of [RFC3588].

5.1. Examples

As an example, consider a client that wishes to discover a Diameter server in the ex1.example.com realm that supports the Credit Control Application. The client performs a NAPTR query for that domain, and the following NAPTR records are returned:

```
;;        order pref flags service   regexp replacement
IN NAPTR  50    50   "s"   "aaa:diameter.sctp" ""
             _diameter._sctp.ex1.example.com
IN NAPTR  50    50   "s"   "aaa+ap1:diameter.sctp" ""
             _diameter._sctp.ex1.example.com
IN NAPTR  50    50   "s"   "aaa+ap4:diameter.sctp" ""
             _diameter._sctp.ex1.example.com
```

This indicates that the server supports NASREQ (ID=1) and Credit Control (ID=4) Applications over SCTP. If the client supports SCTP, it will be used, targeted to a host determined by an SRV lookup of _diameter._sctp.ex1.example.com.

That SRV lookup would return:

```
;;       Priority  Weight  Port    Target
IN SRV   0         1       3868    server1.ex1.example.com
IN SRV   0         2       3868    server2.ex1.example.com
```

As an alternative example, a client that wishes to discover a Diameter server in the ex2.example.com realm that supports the NASREQ application over SCTP. The client performs a NAPTR query for that domain, and the following NAPTR records are returned:

```
;;        order pref flags service   regexp replacement
IN NAPTR  150   50   "a"   "aaa:diameter.stcp"  ""
             server1.ex2.example.com
IN NAPTR  150   50   "a"   "aaa:diameter.tls.tcp"  ""
             server2.ex2.example.com
IN NAPTR  150   50   "a"   "aaa+ap1:diameter.stcp"  ""
             server1.ex2.example.com
IN NAPTR  150   50   "a"   "aaa+ap1:diameter.tls.tcp"  ""
             server2.ex2.example.com
```

This indicates that the server supports NASREQ (ID=1) over SCTP and TLS/TCP via hosts server1.ex2.example.com and server2.ex2.example.com respectively.

6. Usage Guidelines

Diameter is a peer to peer protocol whereas most of the applications that extend the base protocol behave like client/server applications. The role of the peer is not advertised in the NAPTR tags and not even communicated during Diameter capability negotiation (Capabilities-Exchange-Request and Capabilities-Exchange-Answer message exchange). For this reason, NAPTR-based Diameter peer discovery for an application defining client/server roles should only be used by a client to discover servers.

7. IANA Considerations

7.1. IETF Diameter Application Service Tags

IANA is requested to reserve a value of "aaa" for Diameter in the S-NAPTR Application Service Tag registry created by [RFC3958]. IANA is also requested to reserve the following S-NAPTR Application Service Tags for existing IETF Diameter applications in the same registry.

Tag	Diameter Application
aaa+ap1	NASREQ [RFC3588]
aaa+ap2	Mobile IPv4 [RFC4004]
aaa+ap3	Base Accounting [RFC3588]
aaa+ap4	Credit Control [RFC4006]
aaa+ap5	EAP [RFC4072]
aaa+ap6	SIP [RFC4740]
aaa+ap7	Mobile IPv6 IKE [RFC5778]
aaa+ap8	Mobile IPv6 Auth [RFC5778]
aaa+ap9	QoS [RFC5866]
aaa+ap4294967295	Relay [RFC3588]

Future IETF Diameter applications MUST reserve the S-NAPTR Application Service Tag corresponding to the allocated Diameter Application ID as defined in Section 3.

7.2. 3GPP Diameter Application Service Tags

IANA is requested to reserve the following S-NAPTR Application Service Tags for existing 3GPP Diameter applications in the S-NAPTR Application Service Tag registry created by [RFC3958].

Tag	Diameter Application
aaa+ap16777250	3GPP STa [TS29.273]
aaa+ap16777251	3GPP S6a [TS29.272]
aaa+ap16777264	3GPP SWm [TS29.273]
aaa+ap16777267	3GPP S9 [TS29.215]

Future 3GPP Diameter applications can reserve entries in the S-NAPTR Application Service Tag registry created by [RFC3958] which correspond to the allocated Diameter Application IDs as defined in Section 3.

7.3. WiMAX Forum Diameter Application Service Tags

IANA is requested to reserve the following S-NAPTR Application Service Tags for existing WiMAX Forum Diameter applications in the S-NAPTR Application Service Tag registry created by [RFC3958].

Tag	Diameter Application
aaa+ap16777281	WiMAX Network Access Authentication and Authorization Diameter Application (WNAAADA) [WiMAX]
aaa+ap16777282	WiMAX Network Accounting Diameter Application (WNADA) [WiMAX]
aaa+ap16777283	WiMAX MIP4 Diameter Application (WM4DA) [WiMAX]
aaa+ap16777284	WiMAX MIP6 Diameter Application (WM6DA) [WiMAX]
aaa+ap16777285	WiMAX DHCP Diameter Application (WDDA) [WiMAX]
aaa+ap16777286	WiMAX Location Authentication Authorization Diameter Application (WLAADA) [WiMAX]
aaa+ap16777287	WiMAX Policy and Charging Control R3 Policies Diameter Application (WiMAX PCC-R3-P) [WiMAX]
aaa+ap16777288	WiMAX Policy and Charging Control R3 Offline Charging Diameter Application (WiMAX PCC-R3-OFC) [WiMAX]
aaa+ap16777289	WiMAX Policy and Charging Control R3 Offline Charging Prime Diameter Application (WiMAX PCC-R3-OFC-PRIME) [WiMAX]
aaa+ap16777290	WiMAX Policy and Charging Control R3 Online Charging Diameter Application (WiMAX PCC-R3-OC) [WiMAX]

Future WiMAX Forum Diameter applications can reserve entries in the S-NAPTR Application Service Tag registry created by [RFC3958] which correspond to the allocated Diameter Application IDs as defined in Section 3.

7.4. Vendor-Specific Diameter Application Service Tags

Vendor-Specific Diameter Application IDs are allocated by IANA according to the "First Come First Served" policy and do not require an IETF specification. However, the S-NAPTR Application Service Tag registry created by [RFC3958] defines a registration policy of "Specification Required" with a further stipulation that the "specification" is an RFC (of any category). If a Vendor-Specific Diameter Application requires the functionality defined in this document, an RFC of any category MUST be published which reserves the S-NAPTR Application Service Tag corresponding to the Vendor-Specific Diameter Application ID as defined in Section 3.

7.5. Diameter Application Protocol Tags

IANA is requested to reserve the following S-NAPTR Application Protocol Tags for the Diameter transport protocols in the S-NAPTR

Application Protocol Tag registry created by [RFC3958].

Tag	Protocol
diameter.tcp	TCP
diameter.sctp	SCTP
diameter.tls.tcp	TLS/TCP

Future Diameter versions which introduce new transport protocols MUST reserve an appropriate S-NAPTR Application Protocol Tag in the S-NAPTR Application Protocol Tag registry created by [RFC3958].

8. Security Considerations

This document specifies an enhancement to RFC 3588 Diameter base protocol defined NAPTR service field format and also modifications to the NAPTR processing logic defined. The enhancements and modifications are based on the S-NAPTR, which is actually a simplification of the NAPTR, and therefore the same security considerations described in RFC 3588 are applicable to this document. No further extensions are required beyond the security mechanisms offered by RFC 3588. However, a malicious host doing S-NAPTR queries learns applications supported by Diameter agents in a certain realm faster, which might help the malicious host to scan potential targets for an attack more efficiently when some applications have known vulnerabilities.

9. Acknowledgments

We would like to thank Glen Zorn, Avi Lior, Itsuma Tanaka, Sebastien Decugis, Dan Romascanu, Adrian Farrel, David Harrington, Pete Resnick, Robert Sparks, Stephen Farrell, Wesley Eddy, Ralph Droms and Joe Touch and for their comprehensive review comments.

10. Editor’s Notes

This section to be removed prior to publication.

This draft updates sections of RFC3588 that are also being updated by RFC3588bis. At the time this draft was started, it was uncertain whether RFC3588bis would be published first. The authors of this draft decided to proceed optimistically assuming this draft would be published first with the understanding that minor updates are

required if this is not the case.

The application-neutral aspects of Diameter S-NAPTR usage (e.g "aaa: diameter.sctp") were also contributed to RFC3588bis to ensure that it would be functionally complete if it got published first and this draft would come along later to add the application-specific S-NAPTR entries (e.g."aaa+ap5:diameter.sctp").

Depending on the publication order, the S-NAPTR Application Service Tag registry value of "aaa" and the S-NAPTR Application Protocol Tags values ("diameter.tcp"/"diameter.sctp"/"diameter.tls.tcp") will need to be removed either from this draft or RFC3588bis.

11. Normative References

[I-D.ietf-tsvwg-iana-ports]
Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", draft-ietf-tsvwg-iana-ports-10 (work in progress), February 2011.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC3403] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, October 2003.

[RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, January 2005.

[RFC4004] Calhoun, P., Johansson, T., Perkins, C., Hiller, T., and P. McCann, "Diameter Mobile IPv4 Application", RFC 4004, August 2005.

[RFC4006] Hakala, H., Mattila, L., Koskinen, J-P., Stura, M., and J. Loughney, "Diameter Credit-Control Application", RFC 4006, August 2005.

[RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.

[RFC4740] Garcia-Martin, M., Belinchon, M., Pallares-Lopez, M., Canales-Valenzuela, C., and K. Tammi, "Diameter Session Initiation Protocol (SIP) Application", RFC 4740, November 2006.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5778] Korhonen, J., Tschofenig, H., Bournelle, J., Giaretta, G., and M. Nakhjiri, "Diameter Mobile IPv6: Support for Home Agent to Diameter Server Interaction", RFC 5778, February 2010.

[RFC5866] Sun, D., McCann, P., Tschofenig, H., Tsou, T., Doria, A., and G. Zorn, "Diameter Quality-of-Service Application", RFC 5866, May 2010.

[TS29.215]
3rd Generation Partnership Project, "3GPP TS 29.215; Technical Specification Group Core Network and Terminals; Policy and Charging Control (PCC) over S9 reference point; Stage 3 (Release 8)", <http://www.3gpp.org/ftp/Specs/html-info/29215.htm>.

[TS29.272]
3rd Generation Partnership Project, "3GPP TS 29.272; Technical Specification Group Core Network and Terminals; Evolved Packet System; Mobility Management Entity (MME) and Serving GPRS Support Node (SGSN) Related Interfaces Based on Diameter Protocol (Release 8)", <http://www.3gpp.org/ftp/Specs/html-info/29272.htm>.

[TS29.273]
3rd Generation Partnership Project, "3GPP TS 29.273; Technical Specification Group Core Network and Terminals; Evolved Packet System; 3GPP EPS AAA interfaces (Release

8)", <http://www.3gpp.org/ftp/Specs/html-info/29273.htm>.

[WiMAX] WiMAX Forum, "WiMAX Release 1.5", <http://www.wimaxforum.org/resources/documents/technical/T33>.

Authors' Addresses

Mark Jones
Bridgewater Systems

Email: mark@azu.ca

Jouni Korhonen
Nokia Siemens Networks

Email: jouni.nospam@gmail.com

Lionel Morand
Orange Labs

Email: lionel.morand@orange-ftgroup.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 17, 2012

V. Cakulev
Alcatel Lucent
A. Lior
Bridgewater Systems
S. Mizikovsky
Alcatel Lucent

November 14, 2011

Diameter IKEv2 SK: Shared Key-based Support for IKEv2 Server to Diameter Server Interaction

draft-ietf-dime-ikev2-psk-diameter-11.txt

Abstract

The Internet Key Exchange protocol version 2 (IKEv2) is a component of the IPsec architecture and is used to perform mutual authentication as well as to establish and to maintain IPsec security associations (SAs) between the respective parties. IKEv2 supports several different authentication mechanisms, such as the Extensible Authentication Protocol (EAP), certificates, and shared key.

Diameter interworking for Mobile IPv6 between the Home Agent, as a Diameter client, and the Diameter server has been specified. However, that specification focused on the usage of EAP and did not include support for shared key based authentication available with IKEv2. This document specifies the IKEv2 Server to the Diameter server communication when the IKEv2 Peer authenticates using the Internet Key Exchange v2 with Shared Key.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 17, 2012.

Copyright Notice

Table of Contents

1. Introduction

The Internet Key Exchange version 2 (IKEv2) protocol [RFC5996] is used to mutually authenticate two parities and to establish a security association (SA) that can be used to efficiently secure the communication between the IKEv2 Peer and Server, for example, using Encapsulating Security Payload (ESP) [RFC4303] and/or Authentication Header (AH) [RFC4302]. The IKEv2 protocol allows several different mechanisms for authenticating a IKEv2 Peer to be used, such as the Extensible Authentication Protocol, certificates, and shared key.

From a service provider perspective, it is important to ensure that a user is authorized to use the services. Therefore, the IKEv2 Server must verify that the IKEv2 Peer is authorized for the requested services possibly with the assistance of the operator's Diameter servers. [RFC5778] defines the home agent as a Diameter client to the Diameter server communication when the mobile node authenticates using the IKEv2 protocol with the Extensible Authentication Protocol (EAP) [RFC3748] or using the Mobile IPv6 Authentication Protocol [RFC4285]. This document specifies the IKEv2 Server to the Diameter server communication when the IKEv2 Peer authenticates using the Internet Key Exchange v2 with Shared Key.

Figure 1 depicts the reference architecture for this document.

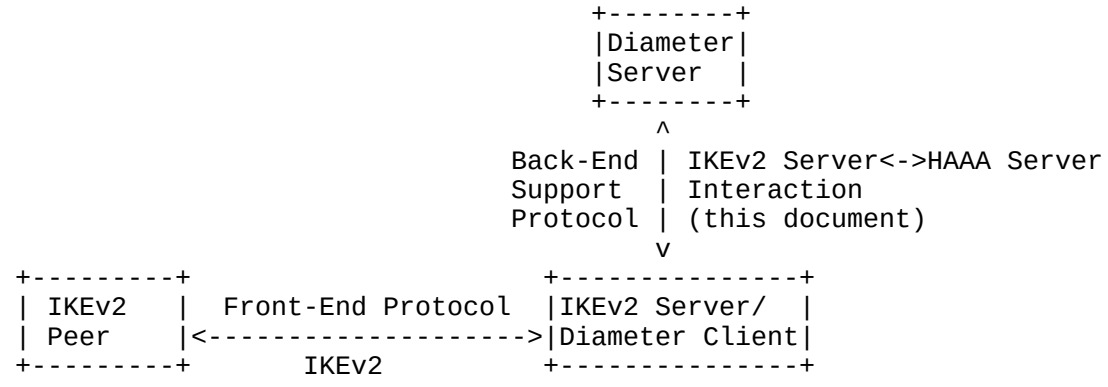


Figure 1: Architecture Overview

An example use case for this architecture is Mobile IPv6 deployment in which the Mobile IPv6 signaling between the Mobile Node and the Home Agent is protected using IPsec. The Mobile node acts as the

IKEv2 Peer and the Home Agent acts as an IKEv2 server. In this use case Internet Key Exchange v2 (IKEv2) with shared key based initiator authentication is used for the setup of the IPsec SAs. The HA obtains the shared key using the Diameter application specified in this document.

This document does not assume that the IKEv2 Server has the shared key (SK) with the IKEv2 Peer. Instead it assumes that the SK provided to the IKEv2 Peer, as well as the SK delivered to the IKEv2 Server by the Diameter Server, are established or derived using the same rules. Furthermore, it assumes that these rules are agreed to by the external protocol on a Peer side providing the key to the IKEv2 Peer, and on the Diameter Server side providing the key to the IKEv2 Server. This document allows for the SK to be obtained for a specific IKEv2 session and exchanged between IKEv2 Server and the Home Authentication, Authorization and Accounting (HAAA) server. The protocol provides IKEv2 attributes to allow the HAAA to compute the SK specific for the session if desired (see Section 10). This is accomplished through the use of a new Diameter application specifically designed for performing IKEv2 authorization decisions. This document focuses on the IKEv2 server, as a Diameter client, communicating to the Diameter server, and specifies the Diameter application needed for this communication. Other protocols leveraging this Diameter application MAY specify their own SK derivation scheme For example see [X.S0047] and [X.S0058]. This document specifies the default procedure for derivation of the SK used in IKEv2 authentication when protocols leveraging this Diameter application do not specify their own derivation procedure. Selection of either default or other SK derivation procedure is done by the external protocol between the Peer side providing the key to the IKEv2 Peer, and the Diameter Server, and is outside the scope of this document.

2. Requirements notation

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 [RFC2119].

2.1. Abbreviations

AH	Authentication Header
AVP	Attribute Value Pair
EAP	Extensible Authentication Protocol
ESP	Encapsulating Security Payload
ESP	Home Authentication, Authorization and Accounting
IKEv2	Internet Key Exchange version 2
NAI	Network Access Identifier
PSK	Pre-Shared Key
SA	Security Association
SK	Shared Key
SPI	Security Parameter Index

3. Application Identifier

This specification defines a new Diameter application and its respective Application Identifier:

Diameter IKE SK (IKESK) TBD1 by IANA

The IKESK Application Identifier is used when the IKEv2 Peer is to be authenticated and authorized using IKEv2 with SK-based authentication.

4. Protocol Description

4.1. Support for IKEv2 and Shared Keys

When IKEv2 is used with SK-based initiator authentication, the Diameter commands IKEv2-SK-Request/Answer defined in this document are used between IKEv2 server and a Home AAA server (HAAA) to authorize the IKEv2 Peer for the services. Upon receiving the IKE_AUTH message from the IKEv2 Peer, the IKEv2 Server uses the information received in IDi [RFC5996] to identify the IKEv2 Peer and the SPI if available to determine the correct SK for this IKEv2 Peer. If there is no SK found associated with this IKEv2 Peer, the IKEv2 Server MUST send an Authorize-Only (Auth-Request-Type set to "Authorize-Only") Diameter IKEv2-SK-Request message to the HAAA to obtain the SK. If the IDi payload extracted from the IKE_AUTH message contains an identity that is meaningful for the Diameter infrastructure, such as a Network Access Identifier (NAI), it SHALL be used by the IKEv2 Server to populate the User-Name AVP in the Diameter message. Otherwise it is out of scope of this document how the IKEv2 server maps the value received in IDi payload to the User-Name AVP and whether the User-Name AVP is included or not in the IKEv2-SK-Request message. The IKEv2 Server SHALL also include in the same Diameter message the IKEv2-Nonces AVP with the initiator and responder nonces (Ni and Nr) exchanged during initial IKEv2 exchange. Finally, the IKEv2 Server SHALL include in the IKEv2-SK-Request message the IKEv2-Identity AVP. Initiator-Identity AVP SHALL be populated with the IDi field extracted from the IKE_AUTH message. If IDr payload was included in the IKE_AUTH message received from the IKEv2 Peer, the IKEv2 Server SHALL also include Responder-Identity AVP populated with the received IDr.

The IKEv2 Server sends the IKEv2-SK-Request message to the IKEv2 Peer’s HAAA. The Diameter message is routed to the correct HAAA as per [RFC3588].

Upon receiving Diameter IKEv2-SK-Request message from the IKEv2 Server, the HAAA SHALL use the User-Name AVP (if present) and/or Initiator-Identity AVP to retrieve the associated keying material. When the default SK generation procedure specified in this document is used, the Peer side that provides the SK to the IKEv2 Peer, as well as the Diameter Server, SHALL use the same SK derivation which follows the methodology similar to that specified in Section 3.1 of [RFC5295], specifically:

SK = KDF(PSK, key label | "\0" | Ni | Nr | IDi | length)

Where:

- KDF is the default key derivation function based on HMAC-SHA-256 as specified in Section 3.1.2 of [RFC5295].

- Pre-Share Key (PSK) is the key available to the protocol leveraging this Diameter application, e.g., the long term shared secret, or the Extended Master Session Key (EMSK) as the result of prior EAP authentication etc. Selection of this value is left up to the protocol leveraging this Diameter application.

- Key label is set to 'sk4ikev2@ietf.org'.

- | denotes concatenation

- "\0" is a NULL octet (0x00 in hex)

- Length is a 2-octet unsigned integer in network byte order of the output key length in octets.

When applications using this protocol define their own SK generation algorithm it is strongly RECOMMENDED that the nonces Ni and Nr are used in the computation. It is also RECOMMENDED that IDi be used. IDr SHOULD NOT be used in the SK generation algorithm. Applications that want to use IDr in the computation should take into consideration that the IDr asserted by the IKEv2 peer may not be the same as the IDr returned by the IKEv2 responder. This mismatch will result in different SKs being generated. The HAAA returns the SK to the IKEv2 Server using the Key AVP as specified in [I-D.ietf-dime-local-keytran].

Once the IKEv2 Server receives the SK from the HAAA, the IKEv2 Server verifies the IKE_AUTH message received from the IKEv2 Peer. If the verification of AUTH is successful, the IKEv2 Server sends the IKE message back to the IKEv2 Peer.

4.2. Session Management

The HAAA may maintain Diameter session state or may be stateless. This is indicated by the presence or absence of the Auth-Session-State AVP included in the Answer message. The IKEv2 Server MUST support the Authorization Session State Machine defined in [RFC3588].

4.2.1. Session-Termination-Request/Answer

In the case where HAAA is maintaining session state, when the IKEv2 Server terminates the SA it SHALL send a Session-Termination-Request (STR) message [RFC3588] to inform the HAAA that the authorized session has been terminated.

The Session-Termination-Answer (STA) message [RFC3588] is sent by the HAAA to acknowledge the notification that the session has been terminated.

4.2.2. Abort-Session-Request/Answer

The Abort-Session-Request (ASR) message [RFC3588] is sent by the HAAA to the IKEv2 Server to terminate the authorized session. When the IKEv2 Server receives the ASR message, it MUST delete the corresponding IKE_SA and all CHILD_SAs set up through it.

The Abort-Session-Answer (ASA) message [RFC3588] is sent by the IKEv2 Server in response to an ASR message.

5. Command Codes for Diameter IKEv2 with SK

This section defines new Command-Code values that MUST be supported by all Diameter implementations conforming to this specification.

Command-Name	Abbrev.	Code	Reference	Application
IKEv2-SK-Request	IKESKR	TBD2	Section 5.1	IKESK
IKEv2-SK-Answer	IKESKA	TBD2	Section 5.2	IKESK

Table 1: Command Codes

5.1. IKEv2-SK-Request (IKESKR) Command

The IKEv2-SK-Request message, indicated with the Command-Code set to TBD2 and the 'R' bit set in the Command Flags field, is sent from the IKEv2 Server to the HAAA to initiate IKEv2 with SK authorization. In this case, the Application-ID field of the Diameter Header MUST be set to the Diameter IKE SK Application ID (value of TDB1).

Message format

```
<IKEv2-SK-Request> ::= < Diameter Header: TBD2, REQ, PXY >
                       < Session-Id >
                       { Auth-Application-Id }
                       { Origin-Host }
                       { Origin-Realm }
                       { Destination-Realm }
                       { Auth-Request-Type }
                       [ Destination-Host ]
                       [ NAS-Identifier ]
                       [ NAS-IP-Address ]
                       [ NAS-IPv6-Address ]
                       [ NAS-Port ]
                       [ Origin-State-Id ]
                       [ User-Name ]
                       [ Key-SPI ]
                       { IKEv2-Identity }
                       [ Auth-Session-State ]
                       { IKEv2-Nonces }
                     * [ Proxy-Info ]
                     * [ Route-Record ]
                       ...
                     * [ AVP ]
```

The IKEv2-SK-Request message MUST include a IKEv2-Nonces AVP containing the Ni and Nr nonces exchanged during initial IKEv2 exchange. The IKEv2-SK-Request message MAY contain a Key-SPI AVP (Key-SPI AVP is specified in [I-D.ietf-dime-local-keytran]). If included, it contains the Security Parameter Index (SPI) that HAAA SHALL use, in addition to the other parameters (e.g., Initiator-Identity), to identify the appropriate SK. The IKEv2-SK-Request message MUST include IKEv2-Identity AVP. The Initiator-Identity AVP SHALL contain IDi as received in IKE_AUTH message. Responder-Identity AVP SHALL be included in the IKEv2-SK-Request message, if IDr payload was included in the IKE_AUTH message received from the IKEv2 Peer. If included, Responder-Identity AVP contains the received IDr.

5.2. IKEv2-SK-Answer (IKESKA) Command

The IKEv2-SK-Answer (IKESKA) message, indicated by the Command-Code field set to TBD2 and the 'R' bit cleared in the Command Flags field, is sent by the HAAA to the IKEv2 Server in response to the IKESKR command. In this case, the Application-ID field of the Diameter Header MUST be set to the Diameter IKE SK Application ID (value of TDB1).

Message format

```
<IKEv2-SK-Answer> ::= < Diameter Header: TBD2, PXY >
                      < Session-Id >
                      { Auth-Application-Id }
                      { Auth-Request-Type }
                      { Result-Code }
                      { Origin-Host }
                      { Origin-Realm }
                      [ User-Name ]
                      [ Key ]
                      [ Responder-Identity ]
                      [ Auth-Session-State ]
                      [ Error-Message ]
                      [ Error-Reporting-Host ]
                    * [ Failed-AVP ]
                      [ Origin-State-Id ]
                    * [ Redirect-Host ]
                      [ Redirect-Host-Usage ]
                      [ Redirect-Max-Cache-Time ]
                    * [ Proxy-Info ]
                    * [ Route-Record ]
                      ...
                    * [ AVP ]
```

If the authorization procedure is successful then the IKEv2-SK-Answer message SHALL include the Key AVP as specified in [I-D.ietf-dime-local-keytran]. The value of the Key-Type AVP SHALL be set to IKEv2-SK (TBD3). The Keying-Material AVP SHALL contain the SK. If Key-SPI AVP is received in IKEv2-SK-Request, Key-SPI AVP SHALL be included in Key AVP. The Key-Lifetime AVP may be included and if it is included then the associated key SHALL NOT be used by the receiver of the answer if the lifetime has expired. Finally, Responder-Identity AVP may be included.

6. Attribute Value Pair Definitions

This section defines new AVPs for the IKEv2 with SK.

6.1. IKEv2-Nonces

The IKEv2-Nonces AVP (Code TBD4) is of type Grouped and contains the nonces exchanged between the IKEv2 Peer and the IKEv2 Server during IKEv2 initial exchange. The nonces are used for SK generation.

```
IKEv2-Nonces ::= < AVP Header: TBD4>
                 {Ni}
                 {Nr}
                *[AVP]
```

6.1.1. Ni

The Ni AVP (AVP Code TBD5) is of type OctetString and contains the IKEv2 initiator nonce as contained in Nonce Data field.

6.1.2. Nr

The Nr AVP (AVP Code TBD6) is of type OctetString and contains the IKEv2 responder nonce as contained in Nonce Data field.

6.2. IKEv2-Identity

The IKEv2-Identity AVP (Code TBD7) is of type Grouped and contains the Initiator and possibly Responder identities as included in IKE_AUTH message sent from the IKEv2 Peer to the IKEv2 Server.

```
IKEv2-Identity ::= < AVP Header: TBD7>
                   {Initiator-Identity}
                   [Responder-Identity]
                  *[AVP]
```

6.2.1. Initiator-Identity

The Initiator-Identity AVP (AVP Code TBD8) is of type Grouped and contains the identity type and identification data of the IDi payload of the IKE_AUTH message.

```
Initiator-Identity ::= < AVP Header: TBD8>
                       {ID-Type}
                       {Identification-Data}
```

```
                              *[AVP]
```

6.2.1.1. ID-Type

The ID-Type AVP (AVP Code TBD9) is of type Enumerated and contains the ID type value of IDi payload of the IKE_AUTH message.

6.2.1.2. Identification-Data

The Identification-Data AVP (AVP Code TBD10) is of type OctetString and contains the Identification Data field of IDi payload of the IKE_AUTH message.

6.2.2. Responder-Identity

The Responder-Identity AVP (AVP Code TBD11) is of type Grouped and contains the identity type and identification data of the IDr payload of the IKE_AUTH message.

```
            Responder-Identity ::= < AVP Header: TBD8>
                                    {ID-Type}
                                    {Identification-Data}
                                   *[AVP]
```

6.2.2.1. ID-Type

The ID-Type AVP (AVP Code TBD9) is of type Enumerated and contains the ID type value of IDr payload of the IKE_AUTH message.

6.2.2.2. Identification-Data

The Identification-Data AVP (AVP Code TBD10) is of type OctetString and contains the Identification Data field of IDr payload of the IKE_AUTH message.

7. AVP Occurrence Tables

The following tables present the AVPs defined or used in this document and their occurrences in Diameter messages. Note that AVPs that can only be present within a Grouped AVP are not represented in this table.

The table uses the following symbols:

0:

The AVP MUST NOT be present in the message.

0+:

Zero or more instances of the AVP MAY be present in the message.

0-1:

Zero or one instance of the AVP MAY be present in the message.

1:

One instance of the AVP MUST be present in the message.

	Command-Code	
AVP Name	IKESKR	IKESKA
Key	0	0-1
Key-SPI	0-1	0
IKEv2-Nonces	1	0
IKEv2-Identity	1	0
Responder-Identity	0	0-1

IKESKR and IKESKA Commands AVP Table

8. AVP Flag Rules

The following table describes the Diameter AVPs, their AVP Code values, types, and possible flag values. The Diameter base [RFC3588] specifies the AVP Flag rules for AVPs in Section 4.5.

				AVP Flag rules			
Attribute Name	AVP Code	Defined in	Value Type	MUST	MAY	SHOULD NOT	MUST NOT
Key	TBD	Note 1	Grouped	M	P		V
Keying-Material	TBD	Note 1	OctetString	M	P		V
Key-Lifetime	TBD	Note 1	Integer64	M	P		V
Key-SPI	TBD	Note 1	Unsigned32	M	P		V
Key-Type	TBD	Note 1	Enumerated	M	P		V
IKEv2-Nonces	TBD4	6.1	Grouped	M	P		V
Ni	TBD5	6.1.1	OctetString	M	P		V
Nr	TBD6	6.1.2	OctetString	M	P		V
IKEv2-Identity	TBD7	6.2	Grouped	M	P		V
Initiator-Identity	TBD8	6.2.1	Grouped	M	P		V
ID-Type	TBD9	6.2.1.1	Enumerated	M	P		V
Identification-Data	TBD10	6.2.1.2	OctetString	M	P		V
Responder-Identity	TBD11	6.2.2	Grouped	M	P		V

AVP Flag Rules Table

Note 1: Key, Keying-Material, Key-Type, Key-SPI and Key-Lifetime AVPs are defined in [I-D.ietf-dime-local-keytran].

9. IANA Considerations

Upon publication of this memo as an RFC, IANA is requested to assign values as described in the following sections.

9.1. Command Codes

IANA is requested to allocate a command code value for the following new command from the Command Code namespace defined in [RFC3588].

Command Code	Value
IKEv2-SK-Request/Answer	TBD2

9.2. AVP Codes

This specification requires IANA to register the following new AVPs from the AVP Code namespace defined in [RFC3588].

- IKEv2-Nonces - TBD4
- Ni - TBD5
- Nr - TBD6
- IKEv2-Identity - TBD7
- Initiator-Identity - TBD8
- ID-Type - TBD9
- Identification-Data - TBD10
- Responder-Identity - TBD11

The AVPs are defined in Section 6.

9.3. AVP Values

IANA is requested to create a new value for the Key-Type AVP. The new value TBD3 signifies that IKEv2 SK is being sent.

9.4. Application Identifier

This specification requires IANA to allocate one new value "Diameter IKE SK" from the Application Identifier namespace defined in [RFC3588].

Application Identifier	Value
Diameter IKE SK (IKESK)	TBD1

10. Security Considerations

The security considerations of the Diameter Base protocol [RFC3588] are applicable to this document (e.g., it is expected that Diameter protocol is used with security mechanism and that Diameter messages are secured).

In addition, the assumption is that the IKEv2 Server and the Diameter Server where the SK is generated are in a trusted relationship. Hence, the assumption is that there is an appropriate security mechanism to protect the communication between these servers. For example the IKEv2 Server and the Diameter server would be deployed in the same secure network or would utilize transport layer security as specified in [RFC3588].

The Diameter messages between the IKEv2 Server and the HAAA may be transported via one or more AAA brokers or Diameter agents. In this case, the IKEv2 Server to the Diameter server AAA communication is hop-by-hop protected, hence relies on the security properties of the intermediating AAA inter- connection networks, AAA brokers, and Diameter agents. Furthermore, any agents that process IKEv2-SK-Answer messages can see the contents of the Key AVP.

To mitigate the threat of exposing long lived PSK, this specification expects that the HAAA derives and returns the associated SK to the IKEv2 Server. Given that SK derivation is security-critical, for the SK derivation this specification recommends the use of short lived secrets, possibly based on a previous network access authentication, if such secrets are available. To ensure key freshness and to limit the key scope, this specification strongly recommends the use of nonces included in IKEv2-SK-Request. The specifics of key derivation depend on the security characteristics of the system that is leveraging this specification (for example see [X.S0047] and [X.S0058]), therefore this specification does not define how the Diameter server derives required keys for these systems. For systems and protocols that leverage this Diameter application but do not specify the key derivation procedure, this document specifies the default key derivation procedure that preserves expected security characteristics.

11. References

11.1. Normative References

[I-D.ietf-dime-local-keytran]
Zorn, G., Wu, W., and V. Cakulev, "Diameter Attribute-Value Pairs for Cryptographic Key Transport", draft-ietf-dime-local-keytran-10 (work in progress), May 2011.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC5295] Salowey, J., Dondeti, L., Narayanan, V., and M. Nakhjiri, "Specification for the Derivation of Root Keys from an Extended Master Session Key (EMSK)", RFC 5295, August 2008.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

11.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4285] Patel, A., Leung, K., Khalil, M., Akhtar, H., and K. Chowdhury, "Authentication Protocol for Mobile IPv6", RFC 4285, January 2006.

[RFC5778] Korhonen, J., Tschofenig, H., Bournelle, J., Giaretta, G., and M. Nakhjiri, "Diameter Mobile IPv6: Support for Home Agent to Diameter Server Interaction", RFC 5778, February 2010.

[X.S0047] 3GPP2: X.S0047, "Mobile IPv6 Enhancements", February 2009.

[X.S0058] 3GPP2: X.S0058, "WiMAX-HRPD Interworking: Core Network Aspects", June 2010.

Authors' Addresses

Violeta Cakulev
Alcatel Lucent
600 Mountain Ave.
3D-517
Murray Hill, NJ 07974
US

Phone: +1 908 582 3207
Email: violeta.cakulev@alcatel-lucent.com

Avi Lior
Bridgewater Systems
303 Terry Fox Drive
Ottawa, Ontario K2K 3J1
Canada

Phone: +1 613-591-6655
Email: avi@bridgewatersystems.com

Semyon Mizikovsky
Alcatel Lucent
600 Mountain Ave.
3C-506
Murray Hill, NJ 07974
US

Phone: +1 908 582 0729
Email: Simon.Mizikovsky@alcatel-lucent.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 20, 2012

G. Zorn
Network Zen
Q. Wu
Huawei
V. Cakulev
Alcatel Lucent

August 19, 2011

Diameter Attribute-Value Pairs for Cryptographic Key Transport
draft-ietf-dime-local-keytran-14

Abstract

Some Authentication, Authorization, and Accounting (AAA) applications require the transport of cryptographic keying material. This document specifies a set of Attribute-Value Pairs (AVPs) providing native Diameter support of cryptographic key delivery.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 20, 2012.

Copyright Notice

Table of Contents

1. Introduction

The Diameter EAP application [RFC4072] defines the EAP-Master-Session-Key and EAP-Key-Name AVPs for the purpose of transporting cryptographic keying material derived during the execution of certain Extensible Authentication Protocol (EAP) [RFC3748] methods (for example, EAP-TLS [RFC5216]). At most one instance of either of these AVPs is allowed in any Diameter message.

However, recent work (see, for example, [RFC5295]) has specified methods to derive other keys from the keying material created during EAP method execution that may require transport in addition to the MSK. In addition, the EAP Re-authentication Protocol (ERP) [RFC5296] specifies new keys that may need to be transported between Diameter nodes.

This note specifies a set of AVPs allowing the transport of multiple cryptographic keys in a single Diameter message.

2. Terminology

2.1. Standards Language

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].

2.2. Technical Terms and Acronyms

DSRK
: Domain-Specific Root Key [RFC5295].

MSK
: Master Session Key [RFC3748].

rMSK
: reauthentication MSK [RFC5296]. This is a per-authenticator key, derived from the rRK (below).

rRK
: reauthentication Root Key, derived from the EMSK Extended Master Session Key [RFC3748] or DSRK [RFC5296].

3. Attribute-Value Pair Definitions

This section defines new AVPs for the transport of cryptographic keys in the Diameter EAP application [RFC4072], as well as other Diameter applications.

3.1. Key AVP

The Key AVP (AVP Code <AC1>) is of type Grouped. It contains the type and keying material and, optionally, an indication of the usable lifetime of the key, the name of the key and a Security Parameter Index (SPI) with which the key is associated.

```
Key ::= < AVP Header: AC1 >
          < Key-Type >
          { Keying-Material }
          [ Key-Lifetime ]
          [ Key-Name ]
          [ Key-SPI ]
        * [ AVP ]
```

3.1.1. Key-Type AVP

The Key-Type AVP (AVP Code <AC2>) is of type Enumerated. This AVP identifies the type of the key being sent. The following decimal values are defined in this document:

DSRK (0)
A Domain-Specific Root Key [RFC5295].

rRK (1)
A reauthentication Root Key [RFC5296].

rMSK (2)
A reauthentication Master Session Key [RFC5296].

If additional values are needed, they are to be assigned by IANA according to the policy stated in Section 5.2,

3.1.2. Key-Name AVP

The Key-Name AVP (AVP Code <AC6>) is of type OctetString. It contains an opaque key identifier. Exactly how this name is generated and used depends on the key type and usage in question, and is beyond the scope of this document (see [RFC5247] and [RFC5295] for discussions of key name generation in the context of EAP).

3.1.3. Keying-Material AVP

The Keying-Material AVP (AVP Code <AC3>) is of type OctetString. The exact usage of this keying material depends upon several factors, including the type of the key and the link layer in use and is beyond the scope of this document.

3.1.4. Key-Lifetime AVP

The Key-Lifetime AVP (AVP Code <AC4>) is of type Unsigned32 and represents the period of time (in seconds) for which the contents of the Keying-Material AVP (Section 3.1.3) is valid.

NOTE:
Applications using this value SHOULD consider the beginning of the lifetime to be the point in time when the message containing the keying material is received. In addition, client implementations SHOULD check to ensure that the value is reasonable; for example, the lifetime of a key should not generally be longer than the session lifetime (see section 8.13 of [I-D.ietf-dime-rfc3588bis].

3.1.5. Key-SPI

The Key-SPI AVP (AVP Code <AC5>) is of type Unsigned32 and contains a SPI value that can be used with other parameters for identifying associated keying material.

4. Security Considerations

Transporting keys is a security-sensitive action. Some forms of keying material are already protected and can be sent safely over the open Internet. However, if a Key AVP contains Keying-Material that is not already protected, then the Diameter messages containing that Key AVP MUST only be sent protected via mutually authenticated TLS or IPsec.

The security considerations applicable to the Diameter Base Protocol [I-D.ietf-dime-rfc3588bis] are also applicable to this document, as are those in Section 8.4 of RFC 4072 [RFC4072].

5. IANA Considerations

Upon publication of this memo as an RFC, IANA is requested to assign values as described in the following sections.

5.1. AVP Codes

Codes must be assigned for the following AVPs using the policy specified in [I-D.ietf-dime-rfc3588bis], Section 11.1.1:

- Key (<AC1>, Section 3.1)
- Key-Type (<AC2>, Section 3.1.1)
- Keying-Material (<AC3>, Section 3.1.3)
- Key-Lifetime (<AC4>, Section 3.1.4)
- Key-SPI (<AC5>, Section 3.1.5)
- Key-Name (<AC6>, Section 3.1.2)

5.2. AVP Values

IANA is requested to create a new registry for values assigned to the Key-Type AVP and populated with the decimal values defined in this document (Section 3.1.1). New values may be assigned for the Key-Type AVP using the "Specification Required" policy [RFC5226]; once values have been assigned, they MUST NOT be deleted, replaced or modified.

6. Acknowledgements

Thanks (in no particular order) to Niclas Comstedt, Semyon Mizikovsky, Hannes Tschofenig, Joe Salowey, Tom Taylor, Frank Xia, Lionel Morand, Dan Romascanu, Bernard Aboba, Jouni Korhonen, Stephen Farrel, Joel Halpern, Phillip Hallam-Baker, Sean Turner and Sebastien Decugis for useful comments, suggestions and review.

7. References

7.1. Normative References

[I-D.ietf-dime-rfc3588bis] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", draft-ietf-dime-rfc3588bis-26 (work in progress), January 2011.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

7.2. Informative References

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, March 2008.

[RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, August 2008.

[RFC5295] Salowey, J., Dondeti, L., Narayanan, V., and M. Nakhjiri, "Specification for the Derivation of Root Keys from an Extended Master Session Key (EMSK)", RFC 5295, August 2008.

[RFC5296] Narayanan, V. and L. Dondeti, "EAP Extensions for EAP Re-authentication Protocol (ERP)", RFC 5296, August 2008.

Authors' Addresses

Glen Zorn
Network Zen
227/358 Thanon Sanphawut
Bang Na, Bangkok 10260
Thailand

Phone: +66 (0) 87-040-4617
Email: glenzorn@gmail.com

Qin Wu
Huawei Technologies Co., Ltd.
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 21001
China

Phone: +86-25-56623633
Email: sunseawq@huawei.com

Violeta Cakulev
Alcatel Lucent
600 Mountain Ave.
3D-517
Murray Hill, NJ 07974
US

Phone: +1 908 582 3207
Email: violeta.cakulev@alcatel-lucent.com

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: October 22, 2012

F. Brockners
S. Bhandari
Cisco
V. Singh

V. Fajardo
Telcordia Technologies

April 20, 2012

Diameter Network Address and Port Translation Control Application
draft-ietf-dime-nat-control-17

Abstract

This document describes the framework, messages, and procedures for the Diameter Network address and port translation Control Application. This Diameter application allows per endpoint control of Network Address Translators and Network Address and Port Translators, which are added to networks to cope with IPv4-address space depletion. This Diameter application allows external devices to configure and manage a Network Address Translator device - expanding the existing Diameter-based AAA and policy control capabilities with a Network Address Translators and Network Address and Port Translators control component. These external devices can be network elements in the data plane such as a Network Access Server, or can be more centralized control plane devices such as AAA-servers. This Diameter application establishes a context to commonly identify and manage endpoints on a gateway or server, and a Network Address Translator and Network Address and Port Translator device. This includes, for example, the control of the total number of Network Address Translator bindings allowed or the allocation of a specific Network Address Translator binding for a particular endpoint. In addition, it allows Network Address Translator devices to provide information relevant to accounting purposes.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 22, 2012.

Copyright Notice

Table of Contents

1. Introduction

Internet service providers deploy Network Address Translators (NATs) and Network Address and Port Translators (NAPTs) [RFC3022] in their networks. A key motivation for doing so is the depletion of available public IPv4 addresses. This document defines a Diameter application allowing providers to control the behavior of NAT and NAPT devices that implement IPv4-to-IPv4 network address and port translation [RFC2663] as well as stateful IPv6-to-IPv4 address family translation as defined in [RFC2663], [RFC6145], and [RFC6146]. The use of a Diameter application allows for simple integration into the existing Authentication, Authorization and Accounting (AAA) environment of a provider.

The Diameter Network address and port translation Control Application (DNCA) offers the following capabilities:

1. Limits or defines the number of NAPT/NAT bindings made available to an individual endpoint. The main motivation for restricting the number of bindings on a per endpoint basis is to protect the service of the service provider against denial of service attacks. If multiple endpoints share a single public IP address, these endpoints can share fate. If one endpoint would (either intentionally, or due to mis-behavior, mis-configuration, mal-ware, etc.) be able to consume all available bindings for a given single public IP address, service would be hampered (or might even become unavailable) for those other endpoints sharing the same public IP address. The efficiency of a NAPT deployment depends on the maximum number of bindings an endpoint could use. Given that the typical number of bindings an endpoint uses depends on the type of endpoint (e.g. a personal computer of a broadband user is expected to use a higher number of bindings than a simple mobile phone) and a NAPT device is often shared by different types of endpoints, it is desirable to actively manage the maximum number of bindings. This requirement is specified in REQ-3 of [I-D.ietf-behave-lsn-requirements]

2. Supports the allocation of specific NAPT/NAT bindings. Two types of specific bindings can be distinguished:

   * Allocation of a pre-defined NAT binding: Both the internal and external IP address and port pair are specified within the request. Some deployment cases, such as access to a web-server within a user’s home network with IP address and port, benefit from statically configured bindings.

   * Allocation of an external IP address for a given internal IP address: The allocated external IP address is reported back to

the requestor. In some deployment scenarios, the application requires immediate knowledge of the allocated binding for a given internal IP address but does not control the allocation of the external IP address; for example, SIP-proxy server deployments.

3. Defines the external address pool(s) to be used for allocating an external IP address: External address pools can either be pre-assigned at the NAPT/NAT device, or specified within a request. If pre-assigned address pools are used, a request needs to include a reference to identify the pool. Otherwise, the request contains a description of the IP address pool(s) to be used; for example, a list of IP-subnets. Such external address pools can be used to select the external IP address in NAPT/NAT bindings for multiple subscribers.

4. Generates reports and accounting records: Reports established bindings for a particular endpoint. The collected information is used by accounting systems for statistical purposes.

5. Queries and retrieves details about bindings on demand: This feature complements the previously mentioned accounting functionality (see item 4). This feature can be used by an entity to find NAT-bindings belonging to one or multiple endpoints on the NAT-device. The entity is not required to create a DNCA control session to perform the query, but would obviously still need to create a Diameter session complying to the security requirements.

6. Identifies a subscriber or endpoint on multiple network devices (NAT/NAPT device, the AAA-server, or the Network Access Server (NAS)): Endpoint identification is facilitated through a Global Endpoint ID. Endpoints are identified through a single or a set of classifiers, such as IP address, Virtual Local Area Network (VLAN) identifier, or interface identifier which uniquely identify the traffic associated with a particular global endpoint.

With the above capabilities, DNCA qualifies as a MIDCOM protocol [RFC3303], [RFC3304], [RFC5189] for middle boxes which perform NAT. The MIDCOM protocol evaluation [RFC4097] evaluated Diameter as a candidate protocol for MIDCOM. DNCA provides the extensions to the Diameter base protocol [RFC3588] following the MIDCOM protocol requirements, such as the support of NAT-specific rule transport, support for oddity of mapped ports, as well as support for consecutive range port numbers. DNCA adds to the MIDCOM protocol capabilities in that it allows to maintain the reference to an endpoint representing a user or subscriber in the control operation,

enabling the control of the behavior of a NAT-device on a per endpoint basis. Following the requirements of different operators and deployments, different management protocols are employed. Examples include e.g. SNMP [RFC3411] and NETCONF [RFC6241] which can both be used for device configuration. Similarly, DNCA is complementing existing MIDCOM implementations, offering a MIDCOM protocol option for operators with an operational environment that is Diameter-focused which desire to use Diameter to perform per endpoint NAT control. Note that in case an operator uses multiple methods and protocols to configure a NAT-device, such as for example command line interface, SNMP, NETCONF, or PCP, along with DNCA specified in this document, the operator MUST ensure that the configurations performed using the different methods and protocols do not conflict in order to ensure a proper operation of the NAT service.

This document is structured as follows: Section 2 lists terminology, while Section 3 provides an introduction to DNCA and its overall deployment framework. Sections 4 to 8 cover DNCA specifics, with Section 4 describing session management, Section 5 the use of the Diameter base protocol, Section 6 new commands, Section 7 Attribute Value Pairs(AVPs) used, and Section 8 accounting aspects. Section 9 presents AVP occurrence tables. IANA and security considerations are addressed in Sections 10 and 11.

2. Conventions

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 [RFC2119].

Abbreviations used in this document:

AAA: Authentication, Authorization, Accounting

DNCA: Diameter Network address and port translation Control Application

Endpoint: Managed entity of the DNCA. An endpoint represents a network element or device, associated with a subscriber, a user or a group of users. An endpoint is represented by a single access-session on a NAS. DNCA assumes a 1:1 relationship between an endpoint, the access-session it represents, and the associated DNCA session.

NAPT: Network Address and Port Translation, see also [RFC3022]

NAT: Network Address Translation (NAT and NAPT are used in this document interchangeably)

NAT-binding or binding: Association of two IP address/port pairs (with one IP address typically being private and the other one public) to facilitate NAT

NAT binding predefined template: Is a policy template or configuration that is predefined at the NAT-device. It may contain NAT-bindings, IP-address pools for allocating the external IP-address of a NAT-binding, the maximum number of allowed NAT-bindings for end-points, etc.

NAT-device: Network Address Translator or Network Address and Port Translator: An entity performing NAT or NAPT.

NAT-controller: Entity controlling the behavior of a NAT-device.

NAS: Network Access Server

NCR: NAT Control Request

NCA: NAT Control Answer

NAT44: IPv4 to IPv4 network address and port translation, see [RFC2663]

NAT64: IPv6 to IPv4 address family translation, see [RFC6145] and [RFC6146]

PPP: Point-to-Point Protocol [RFC1661]

3. Deployment Framework

3.1. Deployment Scenario

Figure 1 shows a typical network deployment for IPv4-Internet access. A user's IPv4 host (i.e. endpoint) gains access to the Internet though a NAS, which facilitates the authentication of the endpoint and configures the endpoints's connection according to the authorization and configuration data received from the AAA-server upon successful authentication. Public IPv4 addresses are used throughout the network. DNCA manages an endpoint that represents a network element or device or IPv4 host, associated with a subscriber, a user or a group of users. An endpoint is represented by a single access-session on a NAS. DNCA assumes a 1:1 relationship between an endpoint, the access-session it represents, and the associated DNCA

session.

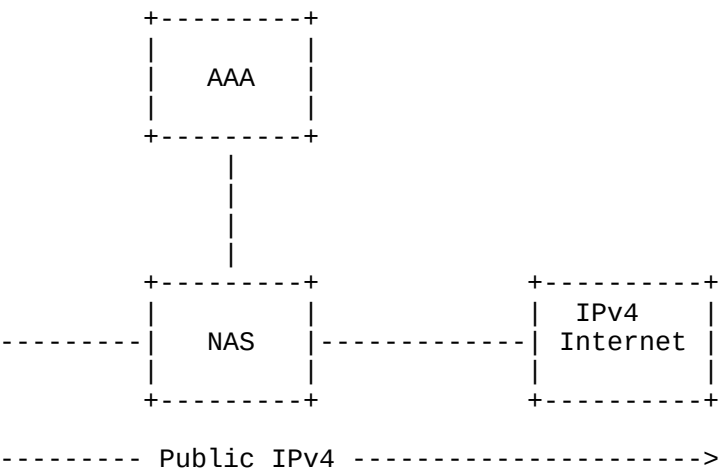


Figure 1: Typical network deployment for internet access

Figure 2 depicts the deployment scenario where a service provider places a NAT between the host and the public Internet. The objective is to provide the customer with connectivity to the public IPv4 Internet. The NAT-device performs network address and port (and optionally address family) translation, depending on whether the access network uses private IPv4 addresses or public IPv6 addresses, to public IPv4 addresses. Note that there may be more than one NAS, NAT-device, or AAA-entity in a deployment, although the figures only depict one entity each for clarity.

If the NAT-device would be put in place without any endpoint awareness, the service offerings of the service provider could be impacted as detailed in [I-D.ietf-behave-lsn-requirements]. This includes cases like:

- Provisioning static NAT bindings for particular endpoints
- Using different public IP address pools for different set of endpoints (for example, residential or business customers)
- Reporting allocated bindings on a per endpoint basis
- Integrate control of the NAT-device into the already existing per endpoint management infrastructure of the service provider

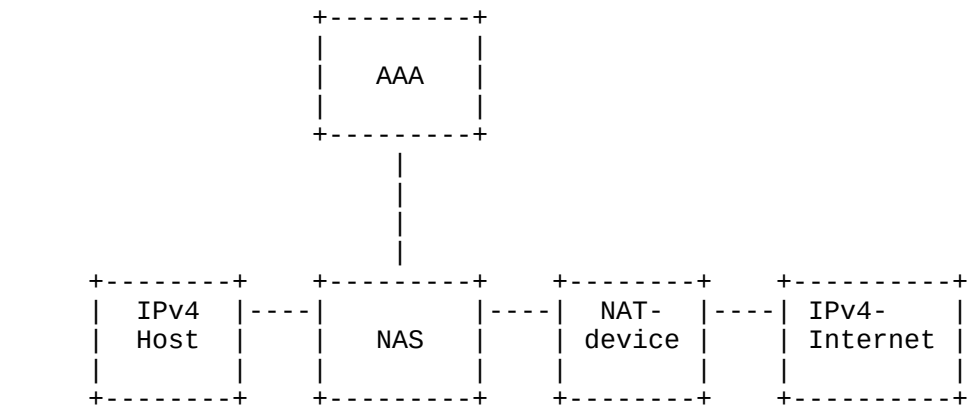


Figure 2: Access network deployment with NAT

Figure 2 shows a typical deployment for IPv4-Internet access involving a NAT-device within the service provider network. The figure describes two scenarios: One where an IPv4-host (with a private IPv4 address) accesses the IPv4-Internet, as well as one where an IPv6-host accesses the IPv4-Internet.

3.2. Diameter NAPT Control Application Overview

DNCA runs between two DNCA Diameter peers. One DNCA Diameter peer resides within the NAT-device, the other DNCA Diameter peer resides within a NAT-controller (discussed in Section 3.3). DNCA allows per endpoint control and management of NAT within the NAT-device. Based on Diameter, DNCA integrates well with the suite of Diameter applications deployed for per endpoint authentication, authorization, accounting, and policy control in service provider networks.

DNCA offers:

- Request and answer commands to control the allowed number of NAT bindings per endpoint to request the allocation of specific bindings for an endpoint, to define the address pool to be used for an endpoint.

- Provides per endpoint reporting of the allocated NAT bindings.

- o Provides unique identification of an endpoint on NAT-device, AAA-server and NAS, to simplify correlation of accounting data streams.

DNCA allows controlling the behavior of a NAT-device on a per endpoint basis during initial session establishment and at later stages by providing an update procedure for already established sessions. Using DNCA, per endpoint NAT binding information can be retrieved either using accounting mechanisms or through an explicit session query to the NAT.

3.3. Deployment Scenarios For DNCA

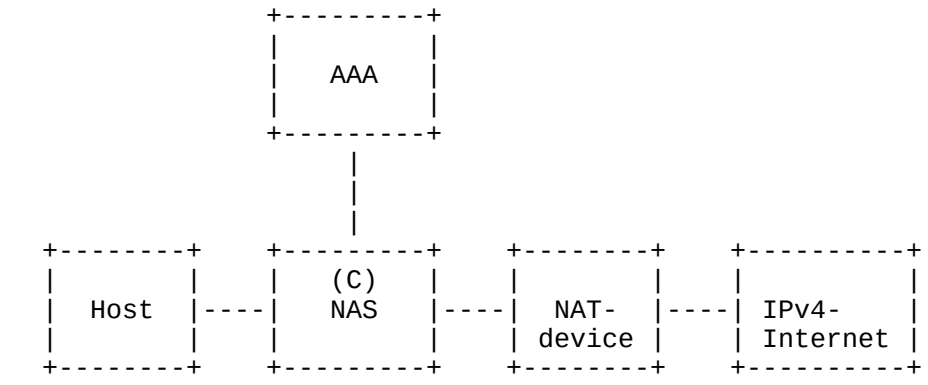
DNCA can be deployed in different ways. DNCA supports deployments with "n" NAT-controllers and "m" NAT-devices, with n and m equal to or greater than 1. From a DNCA perspective an operator should ensure that the session representing a particular endpoint is atomic. Any deployment MUST ensure that for any given endpoint only a single DNCA NAT-controller and is active at any point in time. This is to ensure that NAT-devices controlled by multiple NAT-controllers do not receive conflicting control requests for a particular endpoint, or would be unclear which NAT-controller to send accounting information to. Operational considerations MAY require an operator to use alternate control mechanisms or protocols such as SNMP or manual configuration via a Command-Line-Interface to apply per-endpoint NAT-specific configuration, like for example static NAT-bindings. For these cases, the NAT-device MUST allow the operator to configure a policy how configuration conflicts are resolved. Such a policy could for example specify that manually configured NAT-bindings using the Command-Line-Interface always take precedence over those configured using DNCA.

Two common deployment scenarios are outlined in Figure 3 ("integrated deployment") and Figure 4 ("autonomous deployment"). Per the note above, multiple instances of NAT-controllers and NAT-devices could be deployed. The figures only show single instances for reasons of clarity. The two shown scenarios differ in which entity fulfills the role of the NAT-controller. Within the figures (C) denotes the network element performing the role of the NAT-controller.

The integrated deployment approach hides the existence of the NAT-device from external servers, such as the AAA-server. It is suited for environments where minimal changes to the existing AAA deployment are desired. The NAS and the NAT-device are Diameter peers supporting the DNCA. The Diameter peer within the NAS, performing the role of the NAT-controller, initiates and manages sessions with the NAT-device, exchanges NAT specific configuration information and handles reporting and accounting information. The NAS receives

reporting and accounting information from the NAT-device. With this information, the NAS can provide a single accounting record for the endpoint. A system correlating the accounting information received from the NAS and NAT-device would not be needed.

An example network attachment for an integrated NAT deployment can be described as follows: An endpoint connects to the network, with the NAS being the point of attachment. After successful authentication, the NAS receives endpoint related authorization data from the AAA-server. A portion of the authorization data applies to per endpoint configuration on the NAS itself, another portion describes authorization and configuration information for NAT control aimed at the NAT-device. The NAS initiates a DNCA session to the NAT-device and sends relevant authorization and configuration information for the particular endpoint to the NAT-device. This can comprise NAT-bindings, which have to be pre-established for the endpoint, or management related configuration, such as the maximum number of NAT-bindings allowed for the endpoint. The NAT-device sends its per endpoint accounting information to the NAS, which aggregates the accounting information received from the NAT-device with its local accounting information for the endpoint into a single accounting stream towards the AAA-server.



For NAT44 deployments (IPv4 host):
<----- Private IPv4 ----------><--- Public IPv4 --->

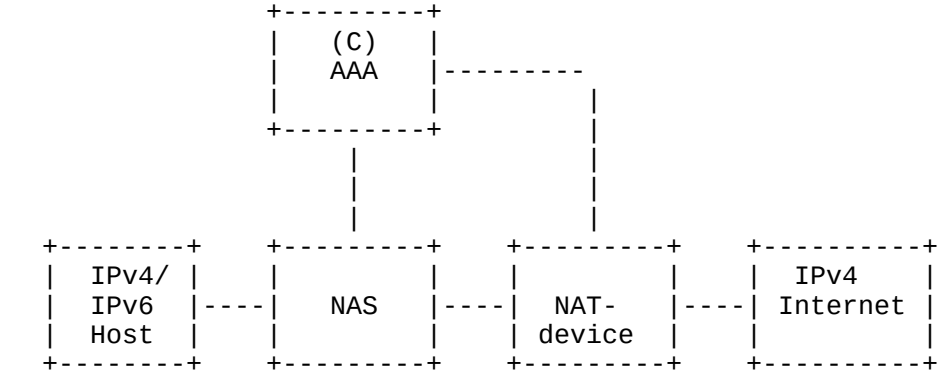
For NAT64 deployments (IPv6 host):
<----- Public IPv6 ----------><--- Public IPv4 --->

Figure 3: NAT control deployment: Integrated deployment

Figure 3 shows examples of integrated deployments. The figure describes two scenarios: One where an IPv4-host (with a private IPv4 address) accesses the IPv4-Internet, as well as one where an IPv6-

host accesses the IPv4-Internet.

The autonomous deployment approach decouples endpoint management on the NAS and NAT-device. In the autonomous deployment approach, the AAA-system and the NAT-device are the Diameter peers running the DNCA. The AAA-system also serves as NAT-controller. It manages the connection to the NAT-device, controls the per endpoint configuration, and also receives accounting and reporting information from the NAT-device. Different from the integrated deployment scenario, the autonomous deployment scenario does not "hide" the existence of the NAT-device from the AAA infrastructure. Here two accounting streams are received by the AAA-server for one particular endpoint, one from the NAS, and one from the NAT-device.



For NAT44 deployments (IPv4 host):
<----- Private IPv4 ----------><--- Public IPv4 --->

For NAT64 deployments (IPv6 host):
<----- Public IPv6 ----------><--- Public IPv4 --->

Figure 4: NAT control deployment: Autonomous deployment

Figure 4 shows examples of autonomous deployments. The figure describes two scenarios: One where an IPv4-host (with a private IPv4 address) accesses the IPv4-Internet, as well as one where an IPv6-host accesses the IPv4-Internet.

4. DNCA Session Establishment and Management

Note that this section forward references some of the commands and AVPs defined for DNCA. Please refer to Section 6 and Section 8 for details. DNCA runs between a Diameter peer residing in a NAT-controller and a Diameter peer residing in a NAT-device. Note that,

per what was already mentioned above, each DNCA session between Diameter peers in a NAT-controller and a NAT-device represents a single endpoint, with an endpoint being either a network element, a device or an IPv4 host associated with a subscriber, a user, or a group of users. The Diameter peer within the NAT-controller is always the control requesting entity: It initiates, updates, or terminates the sessions. Sessions are initiated when the NAT-controller learns about a new endpoint (i.e., host) that requires a NAT service. This could for example be due to the entity hosting the NAT-controller receiving authentication, authorization, or accounting requests for or from the endpoint. Alternate methods that could trigger session setup include local configuration, receipt of a packet from a formerly unknown IP-address, etc.

4.1. Session Establishment

The DNCA Diameter peer within the NAT-controller establishes a session with the DNCA Diameter peer within the NAT-device to control the behavior of the NAT function within the NAT-device. During session establishment, the DNCA Diameter peer within the NAT-controller passes along configuration information to DNCA Diameter peer within the NAT-device. The session configuration information comprises the maximum number of bindings allowed for the endpoint associated with this session, a set of pre-defined NAT bindings to be established for this endpoint, or a description of the address pool, that external addresses are to be allocated from.

The DNCA Diameter peer within the NAT-controller generates a NAT-Control Request (NCR) message to the DNCA Diameter peer within the NAT-device with NC-Request-Type AVP set to INITIAL_REQUEST to initiate a Diameter NAT control session. On receipt of a NCR the DNCA Diameter peer within the NAT-device sets up a new session for the endpoint associated with the endpoint classifier(s) contained in the NCR. The DNCA Diameter peer within the NAT-device notifies its DNCA Diameter peer within the NAT-controller about successful session setup using a NAT-Control Answer (NCA) message with Result-Code set to DIAMETER_SUCCESS. Figure 5 shows the initial protocol interaction between the two DNCA Diameter peers.

The initial NAT-Control-Request MAY contain configuration information for the session, which specifies the behavior of the NAT-device for the session. The configuration information that MAY be included, comprises:

- A list of NAT bindings, which should be pre-allocated for the session; for example, in case an endpoint requires a fixed external IP-address/port pair for an application.

- The maximum number of NAT-bindings allowed for an endpoint.

- A description of the external IP-address pool(s) to be used for the session.

- A reference to a NAT Binding Predefined template on the NAT-device, which is applied to the session. Such a NAT Binding Predefined template on the NAT-device may contain, for example, the name of the IP-address pool that external IP-addresses should be allocated from, the maximum number of bindings permitted for the endpoint, etc.

In certain cases, the NAT-device may not be able to perform the tasks requested within the NCR. These include the following:

- If a DNCA Diameter peer within the NAT-device receives a NCR from a DNCA Diameter peer within a NAT-controller with NC-Request-Type AVP set to INITIAL_REQUEST that identifies an already existing session; that is endpoint identifier match an already existing session, the DNCA Diameter peer within the NAT-device MUST return an NCA with Result-Code set to SESSION_EXISTS, and provide the Session-Id of the existing session in the Duplicate-Session-Id AVP.

- If a DNCA Diameter peer within the NAT-device receives a NCR from a DNCA Diameter peer within a NAT-controller with NC-Request-Type AVP set to INITIAL_REQUEST that matches more than one of the already existing sessions; that is, DNCA Diameter peer and endpoint identifier match already existing sessions, the DNCA Diameter peer within the NAT-device MUST return an NCA with Result-Code set to INSUFFICIENT-CLASSIFIERS. In case a DNCA Diameter peer receives a NCA that reports Insufficient-Classifiers, it MAY choose to retry establishing a new session using additional or more specific classifiers.

- If the NCR contains a NAT Binding predefined template not defined on the NAT-device, the DNCA Diameter peer within the NAT-device MUST return an NCA with Result-Code AVP set to UNKNOWN_BINDING_TEMPLATE_NAME.

- In case the NAT-device is unable to establish all of the bindings requested in the NCR, the DNCA Diameter peer MUST return an NCA with Result-Code set to BINDING_FAILURE. A DNCA Diameter peer within a NAT-device MUST treat a NCR as an atomic operation; hence none of the requested bindings will be established by the NAT-device. Either all requested actions within a NCR MUST be completed successfully, or the entire request fails.

- If a NAT-device cannot conform to a request to set the maximum number of NAT bindings allowed for a session, the DNCA Diameter peer in the NAT-device MUST return an NCA with Result-Code AVP set to MAX_BINDINGS_SET_FAILURE. Such a condition can for example occur if the operator specified the maximum number of NAT bindings through another mechanism, which per the operator's policy, takes precedence over DNCA.

- If a NAT-device does not have sufficient resources to process a request, the DNCA Diameter peer MUST return an NCA with Result-Code set to RESOURCE_FAILURE.

- In case Max-NAT-Bindings, NAT-Control-Definition as well as NAT-Control-Binding-Template are included in the NCR, and the values in Max-NAT-Bindings and NAT-Control-Definition contradict those specified in the pre-provisioned template on the NAT-device which NAT-Control-Binding-Template references, Max-NAT-Bindings and NAT-Control-Definition MUST override the values specified in the template that NAT-Control-Binding-Template refers to.

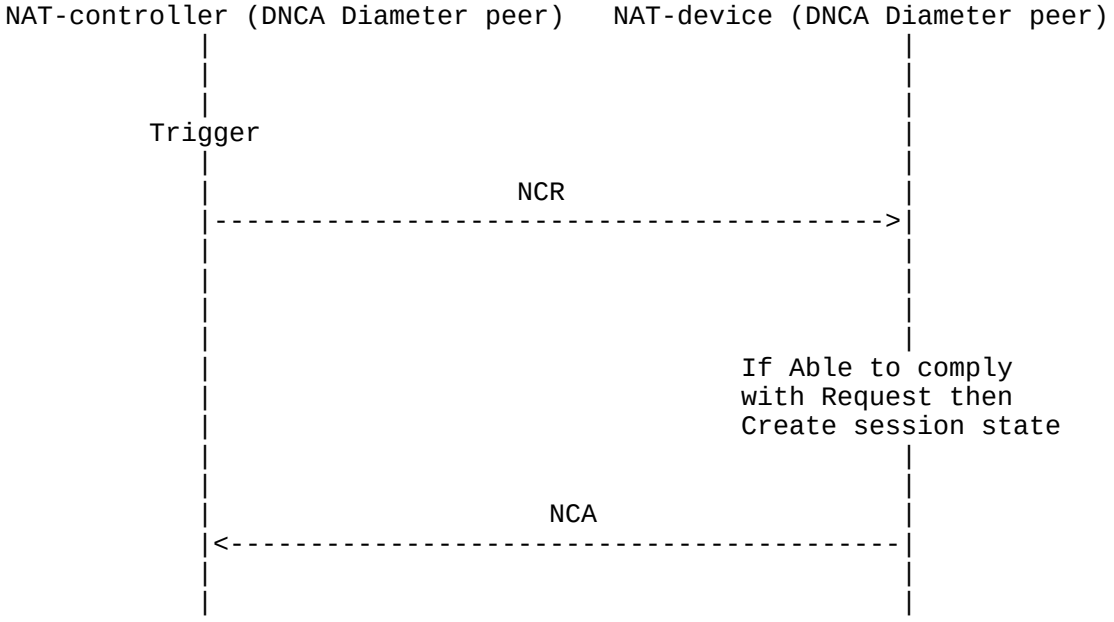


Figure 5: Initial NAT control request and session establishment

Note: The DNCA Diameter peer within the NAT-device creates session state only if it is able to comply with the NCR. On success it will reply with an NCA with Result-Code set to DIAMETER_SUCCESS.

4.2. Session Update

Session update is performed if the NAT-controller desires to change the behavior of the NAT-device for an existing session. Session update could be used, for example, to change the number of allowed bindings for a particular session, or establish or remove a pre-defined binding.

The DNCA Diameter peer within the NAT-controller generates a NCR message to the DNCA Diameter peer within the NAT-device with NC-Request-Type AVP set to UPDATE_REQUEST upon receiving a trigger signal. If the session is updated successfully, the DNCA Diameter peer within the NAT-device notifies the DNCA Diameter peer within the NAT-controller about the successful session update using a NAT-Control Answer (NCA) message with Result-Code set to DIAMETER_SUCCESS.Figure 6 shows the protocol interaction between the two DNCA Diameter peers.

In certain cases, the NAT-device may not be able to perform the tasks requested within the NCR. These include the following:

- If DNCA Diameter peer within a NAT-device receives an NCR update or query request for a non-existent session, it MUST set Result-Code in the answer to DIAMETER_UNKNOWN_SESSION_ID.

- If the NCR contains a NAT Binding Predefined template not defined on the NAT-device, an NCA with Result-Code AVP set to UNKNOWN_BINDING_TEMPLATE_NAME MUST be returned.

- If the NAT-device cannot establish the requested binding because the maximum number of allowed bindings has been reached for the endpoint classifier, an NCA with Result-Code AVP set to MAXIMUM_BINDINGS_REACHED_FOR_ENDPOINT MUST be returned to the DNCA Diameter peer.

- If the NAT-device cannot establish some or all of the bindings requested in an NCR, but has not yet reached the maximum number of allowed bindings for the endpoint, an NCA with Result-Code set to BINDING_FAILURE MUST be returned. As already noted, the DNCA Diameter peer in a NAT-device MUST treat an NCR as an atomic operation. Hence none of the requested bindings will be established by the NAT-device in case of failure. Actions requested within a NCR are either all successful or all fail.

- If the NAT-device cannot conform to a request to set the maximum number of bindings allowed for a session as specified by the Max-NAT-Bindings, the DNCA Diameter peer in the NAT-device MUST return an NCA with Result-Code AVP set to MAX_BINDINGS_SET_FAILURE.

- If the NAT-device does not have sufficient resources to process a request, an NCA with Result-Code set to RESOURCE_FAILURE MUST be returned.

- If an NCR changes the maximum number of NAT-bindings allowed for the endpoint defined through an earlier NCR, the new value MUST override any previously defined limit on the maximum number of NAT bindings set through DNCA. Note that prior to overwriting an existing value, the NAT-device MUST check whether the overwrite action conforms to the locally configured policy. Deployment dependent, an existing value could have been set by a protocol or mechanism different from DNCA and with higher priority. In which case, the NAT-device will refuse the change and the DNCA Diameter peer in the NAT-device MUST return an NCA with Result-Code AVP set to MAX_BINDINGS_SET_FAILURE. It depends on the implementation of the NAT-device on how the NAT-device copes with a case where the new value is lower than the actual number of allocated bindings. The NAT-device SHOULD refrain from enforcing the new limit immediately (that is, actively remove bindings), but rather disallows the establishment of new bindings until the current number of bindings is lower than the newly established maximum number of allowed bindings.

- If an NCR specifies a new NAT Binding Predefined template on the NAT-device, the NAT Binding Predefined template overrides any previously defined rule for the session. Existing NAT-bindings SHOULD NOT be impacted by the change of templates.

- In case Max-NAT-Binding, NAT-Control-Definition as well as NAT-Control-Binding-Template are included in the NCR, and the values in Max-NAT-Bindings and NAT-Control-Definition contradict those specified in the pre-provisioned template on the NAT-device which NAT-Control-Binding-Template references, Max-NAT-Bindings and NAT-Control-Definition MUST override the values specified in the template that the NAT-Control-Binding-Template refers to.

Note: Already established bindings for the session SHOULD NOT be affected in case the tasks requested within the NCR cannot be completed.

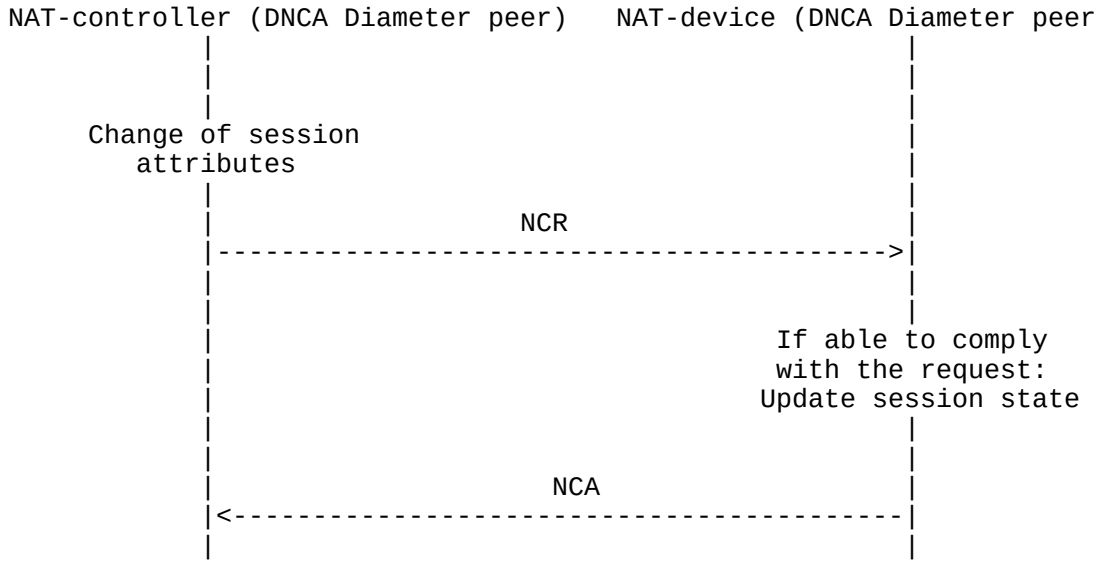


Figure 6: NAT control request for session update

4.3. Session and Binding Query

A Session and NAT-binding query MAY be used by the DNCA Diameter peer within the NAT-controller to either retrieve information on the current bindings for a particular session at the NAT-device or discover the session identifier for a particular external IP address/ port pair.

A DNCA Diameter peer within the NAT-controller starts a session query by sending an NCR message with NC-Request-Type AVP set to QUERY_REQUEST. Figure 7 shows the protocol interaction between the DNCA Diameter peers.

Two types of query requests exist. The first type of query request uses the session ID as input parameter to the query. It is to allow the DNCA Diameter peer within the NAT-controller to retrieve the current set of bindings for a specific session. The second type of query request is used to retrieve the session identifiers, along with the associated bindings, matching a criteria. This enables the DNCA Diameter peer within the NAT-controller to find those sessions, which utilize a specific external or internal IP-address.

1. Request a list of currently allocated NAT bindings for a particular session: On receiving a NCR, the NAT-device SHOULD look up the session information for the session ID contained in the NCR, and report all currently active NAT-bindings for the

session using an NCA message with Result-Code set to DIAMETER_SUCCESS. In this case the NCR MUST NOT contain a NAT-Control-Definition AVP. Each NAT-binding is reported in a NAT-Control-Definition AVP. In case the session ID is unknown, the DNCA Diameter peer within the NAT-device MUST return an NCA message with Result-Code set to DIAMETER_UNKNOWN_SESSION_ID.

2. Retrieve session IDs and bindings for internal IP-address or one or multiple external IP-address/port pairs: If the DNCA Diameter peer within the NAT-controller wishes to retrieve the session ID(s) for internal IP-address or one or multiple external IP-address/port pairs, it MUST include the internal IP-address as part of Framed-IP-Address or external IP-address/port pair(s) as part of the NAT-External-Address AVP of the NCR. The external IP-address/port pair(s) are pre-known to the controller via configuration, AAA interactions, or other means. The session ID is not included in the NCR or the NCA for this type of a query. The DNCA Diameter peer within the NAT-device SHOULD report the NAT-bindings and associated session IDs corresponding to the internal IP-address or external IP-address/port pairs in an NCA message using one or multiple instances of the NAT-Control-Definition AVP. The Result-Code is set to DIAMETER_SUCCESS. In case an external IP-address/port pair has no associated existing NAT-binding, the NAT-Control-Definition AVP contained in the reply just contains the NAT-External-Address AVP.

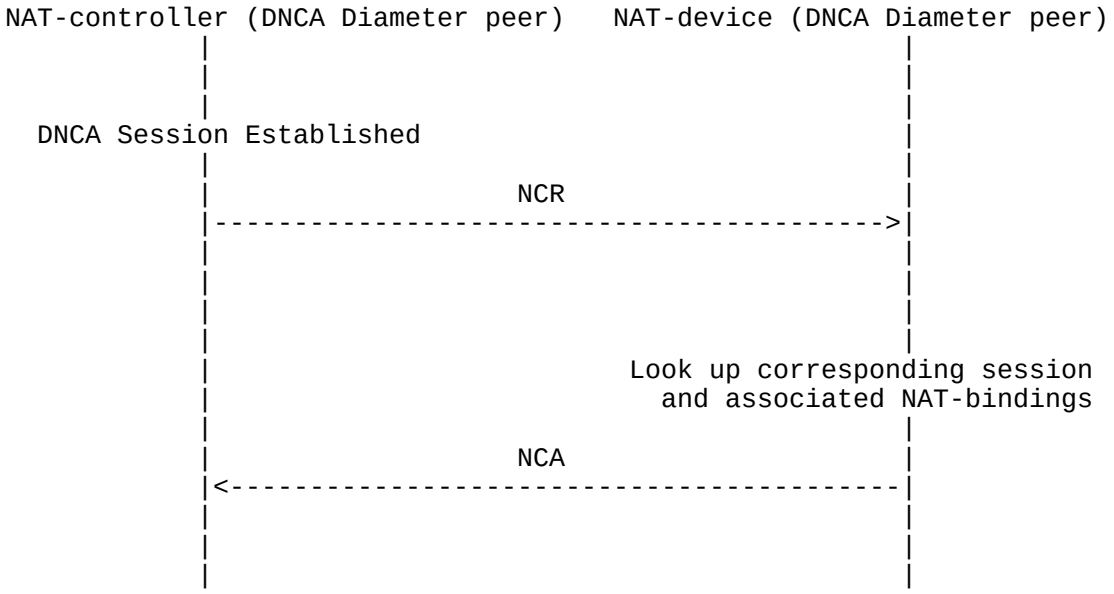


Figure 7: Session query

4.4. Session Termination

Similar to session initiation, session tear down MUST be initiated by the DNCA Diameter peer within the NAT-controller. The DNCA Diameter peer sends a Session Terminate Request (STR) message to its peer within the NAT-device upon receiving a trigger signal. The source of the trigger signal is outside the scope of this document. As part of STR message processing the DNCA Diameter peer within the NAT-device MAY send an accounting stop record reporting all bindings. All the NAT-bindings belonging to the session MUST be removed and the session state MUST be cleaned up. The DNCA Diameter peer within the NAT-device MUST notify its DNCA Diameter peer in the NAT-controller about successful session termination using a Session Terminate Answer (STA) message with Result-Code set to DIAMETER_SUCCESS. Figure 8 shows the protocol interaction between the two DNCA Diameter peers.

If a DNCA Diameter peer within a NAT-device receives a STR and fails to find a matching session, the DNCA Diameter peer MUST return a STA with Result-Code set to DIAMETER_UNKNOWN_SESSION_ID.

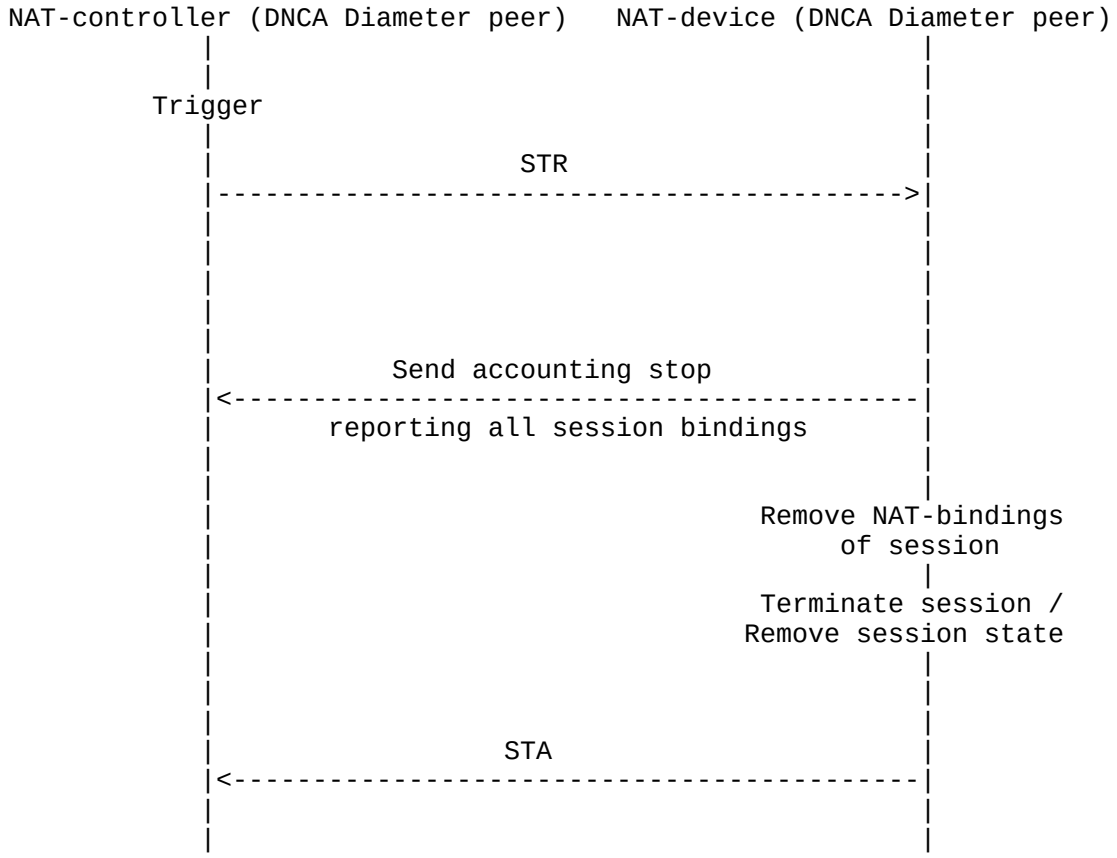


Figure 8: Terminate NAT control session

4.5. Session Abort

An Abort-Session-Request (ASR) message is sent from the DNCA Diameter peer within the NAT-device to the DNCA Diameter peer within the NAT-controller when it is unable to maintain a session due to resource limitations. The DNCA Diameter peer within the NAT-controller MUST acknowledge successful session abort using a Abort Session Answer (ASA) message with Result-Code set to DIAMETER_SUCCESS. Figure 9 shows the protocol interaction between the DNCA Diameter peers. The DNCA Diameter peers will start a session termination procedure as described in Section 4.4 following an ASA with Result-Code set to DIAMETER_SUCCESS.

If the DNCA Diameter peer within a NAT-controller receives an ASR but fails to find a matching session, it MUST return an ASA with Result-Code set to DIAMETER_UNKNOWN_SESSION_ID. If the DNCA Diameter peer

within the NAT-controller is unable to comply with the ASR for any other reason, an ASA with Result-Code set to DIAMETER_UNABLE_TO_COMPLY MUST be returned.

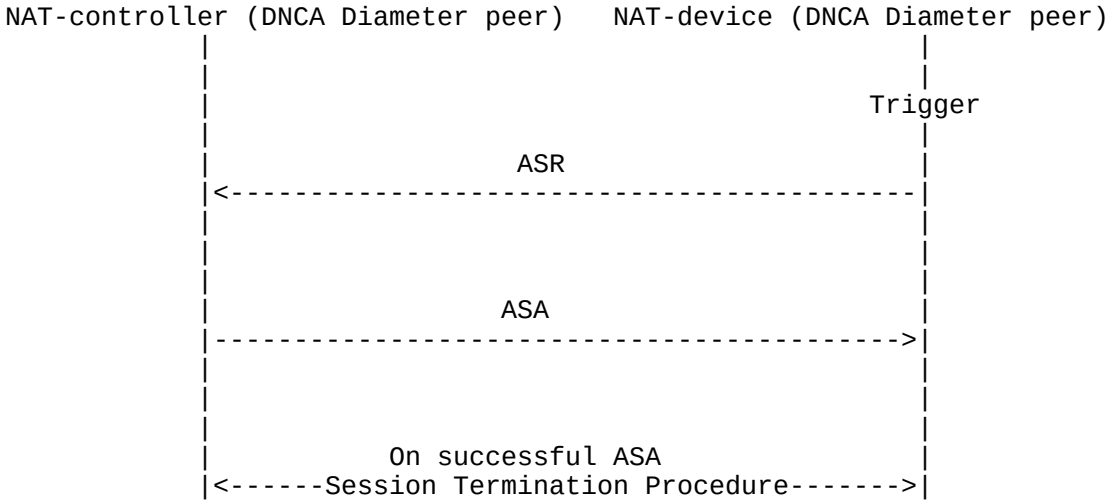


Figure 9: Abort NAT control session

4.6. Failure cases of the DNCA Diameter peers

This document does not specify the behavior in case the NAT-device and NAT-controller, or their respective DNCA Diameter peers are out of sync or lose state. This could happen for example if one of the entities restarts, in case of a (temporary) loss of network connectivity etc. Example failure cases include the following:

- o NAT-controller and the DNCA Diameter peer within the NAT-controller lose state (e.g., due to a restart). In this case,
  - * the DNCA Diameter peer within the NAT-device MAY receive an NCR with NC-Request-Type AVP set to INITIAL_REQUEST that matches an existing session of the DNCA Diameter peer within the NAT-device. The DNCA Diameter peer within the NAT-device MUST return Result-Code that contains Duplicate-Session-Id AVP to report the Session-ID of the existing session. The DNCA Diameter peer within the NAT-controller MAY send an explicit Session Terminate Request (STR) for the older session, which was lost.
  - * a DNCA Diameter peer MAY receive accounting records for a session that does not exist. The DNCA Diameter peer sends an accounting answer with Result-Code set to

DIAMETER_UNKNOWN_SESSION_ID in response. On receiving the response, the DNCA Diameter peer SHOULD clear the session and remove associated session state.

- o NAT-device and the DNCA Diameter peer within NAT-device lose state. In such a case, the DNCA Diameter peer MAY receive a NCR with NC-Request-Type AVP set to UPDATE_REQUEST for a non-existent session. The DNCA Diameter peer MUST return an NCA with Result-Code set to DIAMETER_UNKNOWN_SESSION_ID. When DNCA application within NAT-controller receives this NCA within Result-Code set to DIAMETER_UNKNOWN_SESSION_ID, it MAY try to reestablish DNCA session or disconnect corresponding access session.

- o The DNCA Diameter peer within the NAT-controller is unreachable, for example detected by Diameter device watchdog messages (as defined in Section 5.5 of [RFC3588]), or accounting requests from the DNCA Diameter peer fail to get a response, NAT-bindings and NAT-device state pertaining to that session MUST be cleaned up after a grace period that is configurable on the NAT-device. The grace period can be configured as zero or higher, depending on operator preference.

- o The DNCA Diameter peer within the NAT-device is unreachable or down and NCR fails to get a response. Handling of this case depends on the actual service offering of the service provider. The service provider could for example choose to stop offering connectivity service.

- o A discussion of the mechanisms how a NAT-device cleans up state in case the DNCA Diameter peer within the NAT-device crashes is outside the scope of this document. Implementers of NAT-devices could choose from a variety of options such as coupling the state (e.g. NAT bindings) to timers which require periodic refresh, or time out otherwise, operating system watchdogs for applications, etc.

5. Use of the Diameter Base Protocol

The Diameter Base Protocol defined by [RFC3588] applies with the clarifications listed in the present specification.

5.1. Securing Diameter Messages

For secure transport of Diameter messages, the recommendations in [RFC3588] apply.

DNCA Diameter peers SHOULD verify their identity during the

Capabilities Exchange Request procedure.

A DNCA Diameter peer within the NAT-device SHOULD verify that a DNCA Diameter peer that issues a NCR command is allowed to do so based on:

- The identity of the DNCA Diameter peer
- The type of NCR Command
- The content of the NCR Command
- Any combination of the above

5.2. Accounting Functionality

Accounting functionality (accounting session state machine, related command codes and AVPs) is defined in Section 9 below.

5.3. Use of Sessions

Each DNCA session MUST have a globally unique Session-ID as defined in [RFC3588], which MUST NOT be changed during the lifetime of a DNCA session. The Diameter Session-ID serves as the global endpoint identifier. The DNCA Diameter peers maintain state associated with the Session-ID. This globally unique Session-ID is used for updating, accounting, and terminating the session. A DNCA session MUST NOT have more than one outstanding request at any given instant. A DNCA Diameter peer sends an Abort-Session-Request as defined in [RFC3588] if it is unable to maintain sessions due to resource limitation.

5.4. Routing Considerations

It is assumed that the DNCA Diameter peer within a NAT-controller knows the DiameterIdentity of the Diameter peer within a NAT-device for a given endpoint. Both the Destination-Realm and Destination-Host AVPs are present in the request from a DNCA Diameter peer within a NAT-controller to a DNCA Diameter peer within a NAT-device.

5.5. Advertising Application Support

Diameter nodes conforming to this specification MUST advertise support for DNCA by including the value of TBD.APP-ID in the Auth-Application-Id of the Capabilities-Exchange-Request and Capabilities-Exchange-Answer command[RFC3588].

6. DNCA Commands

The following commands are used to establish, maintain and query NAT-bindings.

6.1. NAT-Control Request (NCR) Command

The NAT-Control Request (NCR) command, indicated by the command field set to TBD.COM-CODE and the "R" bit set in the Command Flags field, is sent from the DNCA Diameter peer within the NAT-controller to the DNCA Diameter peer within the NAT-device in order to install NAT-bindings.

User-Name, Logical-Access-Id, Physical-Access-ID, Framed-IP-Address, Framed-IPv6-Prefix, Framed-Interface-Id, EGRESS-VLANID, NAS-Port-ID, Address-Realm, Calling-Station-ID AVPs serve as identifiers for the endpoint.

Message format:

```
   < NC-Request > ::= < Diameter Header: TBD.COM-CODE, REQ, PXY>
                     { Auth-Application-Id }
                     { Origin-Host }
                     { Origin-Realm }
                     { Destination-Realm }
                     { Destination-Host }
                     { NC-Request-Type }
                     [ Session-Id ]
                     [ Origin-State-Id ]
                   *1 [ NAT-Control-Remove ]
                   *1 [ NAT-Control-Install ]
                     [ NAT-External-Address ]
                     [ User-Name ]
                     [ Logical-Access-Id ]
                     [ Physical-Access-ID ]
                     [ Framed-IP-Address ]
                     [ Framed-IPv6-Prefix ]
                     [ Framed-Interface-Id ]
                     [ EGRESS-VLANID]
                     [ NAS-Port-ID]
                     [ Address-Realm ]
                     [ Calling-Station-ID ]
                    * [ Proxy-Info ]
                    * [ Route-Record ]
                    * [ AVP ]
```

6.2. NAT-Control Answer (NCA) Command

The NAT-Control-Answer (NCA) command, indicated by the Command-Code field set to TBD.COM-CODE and the "R" bit cleared in the Command Flags field, is sent by the DNCA Diameter peer within the NAT-device in response to NAT-Control-Request command.

Message format:

```
<NC-Answer> ::= < Diameter Header: TBD.COM-CODE, PXY >
                { Origin-Host }
                { Origin-Realm }
                { Result-Code }
                [ Session-Id ]
                [ NC-Request-Type ]
              * [ NAT-Control-Definition ]
                [ Current-NAT-Bindings ]
                [ Origin-State-Id ]
                [ Error-Message ]
                [ Error-Reporting-Host ]
              * [ Failed-AVP ]
              * [ Proxy-Info ]
                [ Duplicate-Session-ID ]
              * [ Redirect-Host]
                [ Redirect-Host-Usage ]
                [ Redirect-Max-Cache-Time ]
              * [ Proxy-Info ]
              * [ Route-Record ]
              * [ Failed-AVP ]
              * [ AVP ]
```

7. NAT Control Application Session State Machine

This section contains a set of finite state machines, representing the life cycle of a DNCA session, which MUST be observed by all implementations of the DNCA Diameter application. The DNCA Diameter peers are stateful and the state machine maintained is similar to the stateful Client and Server authorization state machine described in [RFC3588]. When a session is moved to the Idle state, any resources that were allocated for the particular session must be released. Any event not listed in the state machines MUST be considered as an error condition, and an answer, if applicable, MUST be returned to the originator of the message.

In the state table, the event ’Failure to send NCR’ means that the DNCA Diameter peer within the NAT-controller is unable to send the NCR command to the desired destination. This could be due to the

peer being down, or due to the peer sending back the transient failure or temporary protocol error notification DIAMETER_TOO_BUSY or DIAMETER_LOOP_DETECTED in the Result-Code AVP of an NCA.

In the state table "FAILED NCA" means that the DNCA Diameter peer within the NAT-device was not able to honor the corresponding NCR. This can happen due to any transient and permanent error at the NAT-device or its associated DNCA Diameter peer within indicated by the following error Result-Code values: RESOURCE_FAILURE, UNKNOWN_BINDING_TEMPLATE_NAME, MAX_BINDINGS_SET_FAILURE, BINDING_FAILURE, MAXIMUM_BINDINGS_REACHED_FOR_ENDPOINT, SESSION_EXISTS, INSUFFICIENT_CLASSIFIERS.

The following state machine is observed by a DNCA Diameter peer within a NAT-controller. The state machine description uses the term "access session" to describe the connectivity service offered to the endpoint or host. "Access session" should not be confused with the Diameter session ID.

DNCA Diameter peer within a NAT-controller

State	Event	Action	New State
Idle	New endpoint detected that requires NAT Control	Send NCR Initial Request	Pending
Idle	ASR Received for unknown session	Send ASA with Result-Code = UNKNOWN_ SESSION_ID	Idle
Pending	Successful NCA received	Setup complete	Open
Pending	Successful NCA received but peer unable to provide service	Send STR	Discon
Pending	Error processing successful NCA	Send STR	Discon
Pending	Failed NCA received	Clean up	Idle
Open	NAT control	Send	Open

	update required	NCR Update Request	
Open	Successful NCA received		Open
Open	Failed NCA received	Clean up	Idle
Open	Access session end detected	Send STR	Discon
Open	ASR Received, access session will be terminated	Send ASA with Result-Code = SUCCESS, Send STR	Discon
Open	ASR Received, access session will not be terminated	Send ASA with Result-Code != SUCCESS	Open
Discon	ASR Received	Send ASA	Idle
Discon	STA Received	Discon. endpoint	Idle

The following state machine is observed by a DNCA Diameter peer within a NAT-device.

DNCA Diameter peer within a NAT-device

State	Event	Action	New State
Idle	NCR Query request received, and able to provide requested NAT Binding report	Send successful NCA	Idle
Idle	NCR received and able to provide requested NAT control service	Send successful NCA	Open
Idle	NCR request received, and unable to provide requested NAT control service	Send failed NCA	Idle

Open	NCR request received, and able to provide requested NAT control service	Send successful NCA	Open
Open	NCR request received, and unable to provide requested NAT control service	Send failed NCA, Clean up	Idle
Open	Unable to continue providing requested NAT control service	Send ASR	Discon
Open	Unplanned loss of session/ connection to DNCA Diameter peer in NAT controller detected (e.g. due to Diameter watchdog notification)	Clean up	Idle
Discon	Failure to send ASR	Wait, resend ASR	Discon
Discon	ASR successfully sent and ASA Received with Result-Code	Clean up	Idle
Not Discon	ASA Received	None	No change
Any	STR Received	Send STA, Clean up	Idle

8. DNCA AVPs

8.1. Reused Base Protocol AVPs

The following table describes the AVPs reused from Diameter Base Protocol [RFC3588]; their AVP Code values, types, and possible flag values; and whether the AVP MAY be encrypted. The [RFC3588] specifies the AVP Flag rules for AVPs in section 4.5. The Diameter AVP rules are defined in the [RFC3588], section 4.

Attribute Name	AVP Code	Data Type	AVP Flag rules: MUST	MAY	Encr
Acct-Interim-Interval	85	Unsigned32	M	P	Y
Auth-Application-Id	258	Unsigned32	M	P	N
Destination-Host	293	DiamIdent	M	P	N
Destination-Realm	283	DiamIdent	M	P	N
Error-Message	281	UTF8String	M	P	N
Error-Reporting-Host	294	DiamIdent	M	P	N
Failed-AVP	279	Grouped	M	P	N
Origin-Host	264	DiamIdent	M	P	N
Origin-Realm	296	DiamIdent	M	P	N
Origin-State-Id	278	Unsigned32	M	P	N
Proxy-Info	284	Grouped	M	P	N
Result-Code	268	Unsigned32	M	P	N
Route-Record	282	DiamIdent	M		N
Session-Id	263	UTF8String	M	P	Y
User-Name	1	UTF8String	M	P	Y

Table 1: DIAMETER AVPs used from Diameter base

The Auth-Application-Id AVP (AVP Code 258) is assigned by IANA to Diameter applications. The value of the Auth-Application-Id for the Diameter NAT Control Application is TBD.APP-ID. Please refer to [RFC3588] for the definition of the Diameter AVP flag rules and the associated abbreviations used in the table.

8.2. Additional Result-Code AVP Values

This section defines new values for the Result-Code AVP that SHALL be supported by all Diameter implementations that conform to the present document.

8.2.1. Success

No new Result-Code AVP value is defined within this category.

8.2.2. Transient Failures

Result-Code AVP values that fall within the transient failures category are those used to inform a peer that the request could not be satisfied at the time that it was received. The request may be able to be satisfied in the future.

The following new values of the Result-Code AVP are defined:

RESOURCE_FAILURE (TBD.RCX)

The DNCA Diameter peer within the NAT-device indicates that the binding could not be installed or a new session could not be created due to resource shortage.

8.2.3. Permanent Failures

The Result-Code AVP values, which fall within the permanent failures category are used to inform the peer that the request failed, and should not be attempted again. The request may be able to be satisfied in the future.

The following new values of the Result-Code AVP are defined:

UNKNOWN_BINDING_TEMPLATE_NAME (TBD.RCX)

The DNCA Diameter peer within the NAT-device indicates that the binding could not be installed or a new session could not be created because the specified NAT-Control-Binding-Template AVP, that refers to a predefined policy template in the NAT-device, is unknown.

BINDING_FAILURE (TBD.RCX)

The DNCA Diameter peer within the NAT-device indicates that the requested binding(s) could not be installed. For example: Requested ports are already in use.

MAX_BINDINGS_SET_FAILURE (TBD.RCX)

The DNCA Diameter peer within the NAT-device indicates that it failed to conform to a request to configure the maximum number of bindings for a session. For example: An operator defined the maximum number of bindings on the NAT-device using a method or protocol which takes precendence over DNCA.

MAXIMUM_BINDINGS_REACHED_FOR_ENDPOINT (TBD.RCX)

The DNCA Diameter peer within the NAT-device denies the request because the maximum number of allowed bindings has been reached for the specified endpoint classifier.

SESSION_EXISTS (TBD.RCX)

The DNCA Diameter peer within the NAT-device denies request to initialize a new session, if it already has a DNCA session that uses the same set of classifiers as indicated by the DNCA Diameter peer within the NAT-controller in the new session initialization request.

INSUFFICIENT_CLASSIFIERS (TBD.RCX)

The DNCA Diameter peer within the NAT-device requests to initialize a new session, if the classifiers in the request match more than one of the existing sessions on the DNCA Diameter peer within the NAT-device.

8.3. Reused NASREQ Diameter Application AVPs

The following table describes the AVPs reused from the Diameter Network Access Server Application [RFC4005]; their AVP Code values, types, and possible flag values; and whether the AVP MAY be encrypted. The [RFC3588] specifies the AVP Flag rules for AVPs in section 4.5. The Diameter AVP rules are defined in the [RFC3588], section 4.

			AVP Flag rules				
Attribute Name	AVP Code	Value Type	MUST	MAY	SHLD NOT	MUST NOT	Encr
NAS-Port	5	Unsigned32	M	P		V	Y
NAS-Port-Id	87	UTF8String	M	P		V	Y
Calling-Station-Id	31	UTF8String	M	P		V	Y
Framed-IP-Address	8	OctetString	M	P		V	Y
Framed-Interface-Id	96	Unsigned64	M	P		V	Y
Framed-IPv6-Prefix	97	OctetString	M	P		V	Y

Table 2: Reused NASREQ Diameter application AVPs. Please refer to [RFC3588] for the definition of the Diameter AVP flag rules and the associated abbreviations used in the table.

8.4. Reused AVPs from RFC 4675

The following table describes the AVPs reused from "RADIUS Attributes for Virtual LAN and Priority Support" specification [RFC4675]; their AVP Code values, types, and possible flag values; and whether the AVP MAY be encrypted. The [RFC3588] specifies the AVP Flag rules for AVPs in section 4.5. The Diameter AVP rules are defined in the

[RFC3588], section 4.

Attribute Name	AVP Code	Value Type	AVP Flag rules: MUST	MAY	SHLD NOT	MUST NOT	Encr
Egress-VLANID	56	OctetString	M	P		V	Y

Table 3: Reused attributes from RFC 4675. Please refer to [RFC3588] for the definition of the Diameter AVP flag rules and the associated abbreviations used in the table.

8.5. Reused AVPs from Diameter QoS Application

The following table describes the AVPs reused from the Traffic Classification and Quality of Service (QoS) Attributes for Diameter [RFC5777]; their AVP Code values, types, and possible flag values; and whether the AVP MAY be encrypted. The [RFC3588] specifies the AVP Flag rules for AVPs in section 4.5. The Diameter AVP rules are defined in the [RFC3588], section 4.

Attribute Name	AVP Code	Data Type	AVP Flag rules: MUST	MAY	Encr
Port	530	Integer32	M	P	Y
Protocol	513	Enumerated	M	P	Y
Direction	514	Enumerated	M	P	Y

Table 4: Reused QoS-attributes. Please refer to [RFC3588] for the definition of the Diameter AVP flag rules and the associated abbreviations used in the table.

8.6. Reused AVPs from ETSI ES 283 034, e4 Diameter Application

The following table describes the AVPs reused from the Diameter e4 Application [ETSIES283034]; their AVP Code values, types, and possible flag values; and whether the AVP MAY be encrypted. The [RFC3588] specifies the AVP Flag rules for AVPs in section 4.5. The Diameter AVP rules are defined in the [RFC3588], section 4. The Vendor-ID field in these AVP header will be set to ETSI (13019).

			AVP Flag rules		
Attribute Name	AVP Code	Data Type	MUST	MAY	Encr
Address-Realm	301	OctetString	M,V		Y
Logical-Access-Id	302	OctetString	V	M	Y
Physical-Access-ID	313	UTF8String	V	M	Y

Table 5: Reused AVPs from Diameter e4 application. Please refer to [RFC3588] for the definition of the Diameter AVP flag rules and the associated abbreviations used in the table.

8.7. DNCA Defined AVPs

The following table describes the new Diameter AVPs defined in this document; their AVP Code values, types, and possible flag values; and whether the AVP MAY be encrypted. The [RFC3588] specifies the AVP Flag rules for AVPs in section 4.5. The Diameter AVP rules are defined in the [RFC3588], section 4. The AVPs defined here MUST NOT have the V bit in the AVP Flag set.

Attribute Name	AVP Code		Data Type	AVP Flag rules: MUST	MAY	Encr
NC-Request-Type	TBD.AX	8.7.1	Enumerated	M	P	Y
NAT-Control-Install	TBD.AX	8.7.2	Grouped	M	P	Y
NAT-Control-Remove	TBD.AX	8.7.3	Grouped	M	P	Y
NAT-Control-Definition	TBD.AX	8.7.4	Grouped	M	P	Y
NAT-Internal-Address	TBD.AX	8.7.5	Grouped	M	P	Y
NAT-External-Address	TBD.AX	8.7.6	Grouped	M	P	Y
Max-NAT-Bindings	TBD.AX	8.7.7	Unsigned32	M	P	Y
NAT-Control-Binding-Template	TBD.AX	8.7.8	OctetString	M	P	Y
Duplicate-Session-ID	TBD.AX	8.7.9	UTF8String	M	P	Y
NAT-External-Port-Style	TBD.AX	8.7.10	Enumerated	M	P	Y
NAT-Control-Record	TBD.AX	9.2.1	Grouped	M	P	Y
NAT-Control-Binding-Status	TBD.AX	9.2.2	Enumerated	M	P	Y
Current-NAT-Bindings	TBD.AX	9.2.3	Unsigned32	M	P	Y

Table 6: New Diameter AVPs. Please refer to [RFC3588] for the definition of the Diameter AVP flag rules and the associated abbreviations used in the table.

8.7.1. NC-Request-Type AVP

The NC-Request-Type AVP (AVP Code TBD.AX) is of type Enumerated and contains the reason for sending the NAT-Control-Request command. It shall be present in all NAT-Control-Request messages.

The following values are defined:

INITIAL_REQUEST (1)

An Initial Request is to initiate a Diameter NAT control session between the DNCA Diameter peers.

UPDATE_REQUEST (2)

An Update Request is used to update bindings previously installed on a given access session, to add new binding on a

given access session, or to remove one or several binding(s) activated on a given access session.

QUERY_REQUEST (3)

Query Request is used to query a NAT-device about the currently installed bindings for an endpoint classifier.

8.7.2. NAT-Control-Install AVP

The NAT-Control AVP (AVP code TBD.AX) is of type Grouped, and it is used to activate or install NAT bindings. It also contains Max-NAT-Bindings that defines the maximum number of NAT bindings allowed for an endpoint and the NAT-Control-Binding-Template that references a predefined template on the NAT-device that may contain static binding, a maximum number of bindings allowed, an IP-address pool from which external binding addresses should be allocated, etc. If the NAT-External-Port-Style AVP is present, then the NAT-device MUST select the external ports for the NAT-Bindings as per the style specified. The NAT-External-Port-Style is applicable for NAT-Bindings defined by the NAT-Control-Definition AVPs whose NAT-External-Address or Port AVPs within the NAT-External-Address are unspecified.

AVP format:

```
NAT-Control-Install ::= < AVP Header: TBD.AX >
                         * [ NAT-Control-Definition ]
                           [ NAT-Control-Binding-Template ]
                           [ Max-NAT-Bindings ]
                           [ NAT-External-Port-Style ]
                         * [ AVP ]
```

8.7.3. NAT-Control-Remove AVP

The NAT-Control-Remove AVP (AVP code TBD.AX) is of type Grouped, and it is used to deactivate or remove NAT-bindings. At least one of the two AVPs (NAT-Control-Definition AVP, NAT-Control-Binding-Template AVP) SHOULD be present in the NAT-Control-Remove AVP.

AVP format:

```
NAT-Control-Remove ::= < AVP Header: TBD.AX >
                        * [ NAT-Control-Definition ]
                          [ NAT-Control-Binding-Template ]
                        * [ AVP ]
```

8.7.4. NAT-Control-Definition AVP

The NAT-Control-Definition AVP (AVP code TBD.AX) is of type Grouped, and it describes a binding.

The NAT-Control-Definition AVP uniquely identifies the binding between the DNCA Diameter peers.

If both the NAT-Internal-Address and NAT-External-Address AVP(s) are supplied, it is a pre-defined binding.

If the NAT-External-Address AVP is not specified then the NAT-device MUST select the external port as per the NAT-External-Port-Style AVP, if present in the NAT-Control-Definition AVP.

The Protocol AVP describes the transport protocol for the binding. The NAT-Control-Definition AVP can contain either zero or one Protocol AVP. If the Protocol AVP is omitted and if both internal and external IP-address are specified then the binding reserves the IP-addresses for all transport protocols.

The Direction AVP is of type Enumerated. It specifies the direction for the binding. The values of the enumeration applicable in this context are: "IN","OUT". If Direction AVP is OUT or absent, the NAT-Internal-Address refers to the IP-address of the endpoint that needs to be translated. If Direction AVP is "IN", NAT-Internal-Address is the destination IP-address that has to be translated.

AVP format:

```
  NAT-Control-Definition ::= < AVP Header: TBD.AX >
                              { NAT-Internal-Address }
                              [ Protocol ]
                              [ Direction ]
                              [ NAT-External-Address ]
                              [ Session-Id ]
                             * [ AVP ]
```

8.7.5. NAT-Internal-Address AVP

The NAT-Internal-Address AVP (AVP code TBD.AX) is of type Grouped. It describes the internal IP-address and port for a binding. Framed-IPV6-Prefix and Framed-IP-Address AVPs are mutually exclusive. The endpoint identifier Framed-IP-Address, Framed-IPv6-Prefix and the internal address in this NAT-Internal-Address AVP to install NAT-bindings for the session MUST match.

AVP format:

```
NAT-Internal-Address ::= < AVP Header: TBD.AX >
                           [ Framed-IP-Address ]
                           [ Framed-IPv6-Prefix ]
                           [ Port]
                         * [ AVP ]
```

8.7.6. NAT-External-Address AVP

The NAT-External-Address AVP (AVP code TBD.AX) is of type Grouped, and it describes the external IP-address and port for a binding. The external IP-address specified in this attribute can be reused for multiple endpoints by specifying the same address in the respective NAT-External-Address AVPs. If the external IP-address is not specified and the NAT-External-Port-Style AVP is specified in the NAT-Control-Definition AVP then the NAT-device MUST select external port as per the NAT-External-Port-Style AVP.

AVP format:

```
NAT-External-Address ::= < AVP Header: TBD.AX >
                           [ Framed-IP-Address ]
                           [ Port ]
                         * [ AVP ]
```

8.7.7. Max-NAT-Bindings

The Max-NAT-Bindings AVP (AVP code TBD.AX) is of type Unsigned32. It indicates the maximum number of NAT-bindings allowed for a particular endpoint.

8.7.8. NAT-Control-Binding-Template AVP

The NAT-Control-Binding-Template AVP (AVP code TBD.AX) is of type OctetString. It defines a name for a policy template that is predefined at the NAT-device. Details on the contents and structure of the template and configuration are outside the scope of this document. The policy to which this AVP refers to may contain NAT-bindings, IP-address pool for allocating the external IP-address of a NAT-binding, and maximum number of allowed NAT-bindings. Such policy template can be reused by specifying the same NAT-Control-Binding-Template AVP in the corresponding NAT-Control-Install AVPs of multiple endpoints.

8.7.9. Duplicate-Session-Id AVP

The Duplicate-Session-Id AVP (AVP Code TBD.AX) is of type UTF8String. It is used to report errors and contains the Session-Id of an existing session.

8.7.10. NAT-External-Port-Style AVP

The NAT-External-Port-Style AVP (AVP Code TBD.AX) is of type Enumerated and contains the style to be followed while selecting the external port for a NAT-Binding relative to the internal port.

The following values are defined:

FOLLOW_INTERNAL_PORT_STYLE (1)

External port numbers selected MUST follow the same sequence and oddity as the internal ports of the NAT-bindings. The port odditity is required to support protocols like RTP and RTCP as defined in [RFC3550]. If for example the internal port in a requested NAT-binding is odd numbered then the external port allocated MUST also be odd numbered, and vice versa for an even numbered port. In addition, the sequence of port numbering is maintained: If internal ports are consecutive, then the NAT-device MUST choose consecutive external ports for the NAT-bindings.

9. Accounting Commands

The DNCA reuses session based accounting as defined in the Diameter Base Protocol[RFC3588] to report the bindings per endpoint. This reporting is achieved by sending Diameter Accounting Requests (ACR) [Start, Interim and Stop] from the DNCA Diameter peer within the NAT-device to its associated DNCA Diameter peer within the NAT-controller.

The DNCA Diameter peer within the NAT-device sends an ACR Start on receiving a NCR with NC-Request-Type AVP set to INITIAL_REQUEST for a session or on creation of the first binding for a session requested in an earlier NCR. DNCA may send ACR Interim updates, if required, either due to a change in bindings resulting from a NCR with NC-Request-Type AVP set to UPDATE_REQUEST, or periodically as specified in Acct-Interim-Interval by the DNCA Diameter peer within the NAT-controller, or when it creates or tears down bindings. An ACR Stop is sent by the DNCA Diameter peer within the NAT-device on receiving STR.

The function of correlating the multiple bindings used by an endpoint at any given time is relegated to the post processor.

The DNCA Diameter peer within the NAT-device may trigger an interim accounting record when the maximum number of bindings, if received in an NCR, is reached.

9.1. NAT Control Accounting Messages

The ACR and ACA messages are reused as defined in the Diameter Base Protocol [RFC3588] for exchanging endpoint NAT binding details between the DNCA Diameter peers. The DNCA Application IDs is used in the accounting commands. ACR contains one or more optional NAT-Control-Record AVPs to report the bindings. The NAT-device indicates the number of allocated NAT bindings to the NAT-controller using the Current-NAT-Bindings AVP. This number needs to match the number of bindings identified as active within the NAT-Control-Record AVP.

9.2. NAT Control Accounting AVPs

In addition to AVPs for ACR specified in [RFC3588], the DNCA Diameter peer within the NAT-device must add the NAT-Control-Record AVP.

9.2.1. NAT-Control-Record

The NAT-Control-Record AVP (AVP code TBD.AX) is of type Grouped. It describes a binding and its status. If NAT-Control-Binding-Status is set to Created, Event-Timestamp indicates the binding creation time. If NAT-Control-Binding-Status is set to Removed, Event-Timestamp indicates the binding removal time. If NAT-Control-Binding-Status is active, Event-Timestamp need not be present; if a value is present, it indicates that binding is active at the given time.

```
  NAT-Control-Record ::= < AVP Header: TBD.AX >
                         { NAT-Control-Definition }
                         { NAT-Control-Binding-Status }
                         [ Event-Timestamp ]
```

9.2.2. NAT-Control-Binding-Status

The NAT-Control-Binding-Status AVP (AVP code TBD.AX) is of type enumerated. It indicates the status of the binding - created, removed, or active.

The following values are defined:

Created (1)

NAT binding is created.

Active (2)

NAT binding is active.

Removed (3)

NAT binding was removed.

9.2.3. Current-NAT-Bindings

The Current-NAT-Bindings AVP (AVP code TBD.AX) is of type Unsigned32. It indicates the number of NAT bindings active on the NAT-device.

10. AVP Occurrence Table

The following sections present the AVPs defined in this document and specify the Diameter messages in which they can be present. Note: AVPs that can only be present within a Grouped AVP are not represented in this table.

The table uses the following symbols:

0	The AVP MUST NOT be present in the message.
0+	Zero or more instances of the AVP can be present in the message.
0-1	Zero or one instance of the AVP can be present in the message. It is considered an error if there is more than one instance of the AVP.
1	One instance of the AVP MUST be present in the message.
1+	At least one instance of the AVP MUST be present in the message.

10.1. DNCA AVP Table for NAT Control Initial and Update Requests

The following table lists DNCA specific AVPs that have to be present in NCRs and NCAs with NC-Request-Type set to INITIAL_REQUEST or UPDATE_REQUEST.

	Command Code	
Attribute Name	NCR	NCA
NC-Request-Type	1	1
NAT-Control-Install	0-1	0
NAT-Control-Remove	0-1	0
NAT-Control-Definition	0	0
Current-NAT-Bindings	0	0
Duplicate-Session-Id	0	0-1

Note that any combination of "NAT-Control-Install" and "NAT-Control-Remove" AVPs could be present in an update or initial requests. Consider the following examples:

Neither "NAT-Control-Install AVP" nor "NAT-Control-Remove AVP" are present: This could for example be the case if the NAT-controller would only want to receive accounting information, but not control NAT-bindings.

Only "NAT-Control-Install AVP" is present: This could for example be the case if a new NAT-binding is installed for an existing session.

Only "NAT-Control-Remove AVP" is present: This could for example be the case if a new NAT-binding is removed from an existing session.

Both, "NAT-Control-Install AVP" and "NAT-Control-Remove AVP" are present: This could for example be the case if a formerly created NAT-binding is removed and a new NAT-binding is established within the same request.

10.2. DNCA AVP Table for Session Query request

The following table lists DNCA specific AVPs that have to be present in NCRs and NCAs with NC-Request-Type set to QUERY_REQUEST.

	Command Code	
Attribute Name	NCR	NCA
NC-Request-Type	1	1
NAT-Control-Install	0	0
NAT-Control-Remove	0	0
NAT-Control-Definition	0	0+
NAT-External-Address	0+	0
Current-NAT-Bindings	0	1
Duplicate-Session-Id	0	0

10.3. DNCA AVP Table for Accounting Message

The following table lists DNCA specific AVPs, which may or may not be present in ACR and ACA messages.

	Command Code	
Attribute Name	ACR	ACA
NAT-Control-Record	0+	0
Current-NAT-Bindings	1	0

11. IANA Considerations

This section contains the namespaces that have either been created in this specification, or the values assigned to existing namespaces managed by IANA.

In the subsections below, when we speak about review by a Designated Expert, please note that the designated expert will be assigned by the IESG. Initially, such Expert discussions take place on the AAA WG mailing list.

11.1. Application Identifier

This specification assigns the value <TBD.APP-ID>, 'Diameter NAT Control Application', to the Application Identifier namespace defined in [RFC3588]. See Section 4 for more information.

11.2. Command Codes

This specification uses the value <TBD.COM-CODE> from the Command code namespace defined in [RFC3588] for the NAT-Control-Request (NCR), NAT-Control-Answer (NCA) commands. See Section 6.1 and Section 6.2 for more information on these commands.

11.3. AVP Codes

This specification assigns the values <TBD.AX> from the AVP code namespace defined in [RFC3588]. See Section 8.7 for the assignment of the namespace in this specification.

11.4. Result-Code AVP Values

This specification assigns the values <TBD.RCX> (4xxx, 5xxx, 5xxx, 5xxx, 5xxx,5xxx) from the Result-Code AVP value namespace defined in [RFC3588]. See Section 8.2 for the assignment of the namespace in this specification.

11.5. NC-Request-Type AVP

As defined in Section 8.7.1, the NC-Request-Type AVP includes Enumerated type values 1 - 3. IANA has created and is maintaining a namespace for this AVP. All remaining values are available for assignment by a Designated Expert [RFC5226].

11.6. NAT-External-Port-Style AVP

As defined in Section 8.7.10, the NAT-External-Port-Style AVP includes Enumerated type value 1. IANA has created and is maintaining a namespace for this AVP. All remaining values are available for assignment by a Designated Expert [RFC5226].

11.7. NAT-Control-Binding-Status AVP

As defined in Section 8.7.1, the NAT-Control-Binding-Status AVP includes Enumerated type values 1 - 3. IANA has created and is maintaining a namespace for this AVP. All remaining values are available for assignment by a Designated Expert [RFC5226].

12. Security Considerations

This document describes procedures for controlling NAT related attributes and parameters by an entity, which is non-local to the device performing NAT. This section discusses security considerations for DNCA. This includes the interactions between the

Diameter peers within a NAT-controller and a NAT-device as well as general considerations for NAT-control in a service provider network.

Security between a NAT-controller and a NAT-device has a number of components: authentication, authorization, integrity, and confidentiality.

Authentication refers to confirming the identity of an originator for all datagrams received from the originator. Lack of authentication of Diameter messages between the Diameter peers can jeopardize the fundamental service of the peering network elements. A consequence of not authenticating the message sender by the recipient would be that an attacker could spoof the identity of a "legitimate" authorizing entity in order to change the behavior of the receiver. An attacker could for example launch a denial of service attack by setting the maximum number of bindings for a session on the NAT-device to zero; provision bindings on a NAT-device which include IP-addresses already in use in other parts of the network; or request session termination of the Diameter session and hamper an endpoint’s (i.e. a user’s) connectivity. Lack of authentication of a NAT-device to a NAT-controller could lead to situations where the NAT-device could provide a wrong view of the resources (i.e. NAT-bindings). In addition, NAT Binding Predefined template on the NAT-device could be configured differently than expected by the NAT-controller. Failing of any of the two DNCA Diameter peers to provide the required credentials should be subject to logging. The corresponding logging infrastructure of the operator SHOULD be built in a way that it can mitigate potential denial of service attacks resulting from large amounts of logging events. This could include proper dimensioning of the logging infrastructure combined with policing the maximum amount of logging events accepted by the logging system to a threshold which the system is known to be able to handle.

Authorization refers to whether a particular authorizing entity is authorized to signal a network element requests for one or more applications, adhering to a certain policy profile. Failing the authorization process might indicate a resource theft attempt or failure due to administrative and/or credential deficiencies. In either case, the network element should take the proper measures to log such attempts.

Integrity is required to ensure that a Diameter message exchanged between the Diameter peers has not been maliciously altered by intermediate devices. The result of a lack of data integrity enforcement in an untrusted environment could be that an impostor will alter the messages exchanged between the peers. This could cause a change of behavior of the peers, including the potential of a denial of service.

Confidentiality protection of Diameter messages ensures that the signaling data is accessible only to the authorized entities. When signaling messages between the DNCA Diameter peers traverse untrusted networks, lack of confidentiality will allow eavesdropping and traffic analysis.

Diameter offers security mechanisms to deal with the functionality demanded above. DNCA makes use of the capabilities offered by Diameter and the underlying transport protocols to deliver these requirements (see Section 5.1). If the DNCA communication traverses untrusted networks, messages between DNCA Diameter peers SHOULD be secured using either IPsec or TLS. Please refer to [RFC3588], section 13 for details. DNCA Diameter peers SHOULD perform bilateral authentication, authorization as well as procedures to ensure integrity and confidentiality of the information exchange. In addition the Session-Id chosen for a particular Diameter session SHOULD be chosen in a way that it is hard to guess in order to mitigate issues through potential message replay.

DNCA Diameter peers SHOULD have a mutual trust setup. This document does not specify a mechanisms for authorization between the DNCA Diameter peers. The DNCA Diameter peers SHOULD be provided with sufficient information to make an authorization decision. The information can come from various sources, for example the peering devices could store local authentication policy, listing the identities of authorized peers.

Any mechanism or protocol providing control of a NAT-device, and DNCA is an example of such a control mechanism, could allow for misuse of the NAT-device given that it enables the definition of per-destination or per-source rules. Misuse could include anti-competitive practices among providers, censorship, crime, etc. NAT-control could be used as a tool for preventing or redirecting access to particular sites. For instance, by controlling the NAT bindings, one could ensure that endpoints aren’t able to receive particular flows, or that those flows are redirected to a relay that snoops or tampers with traffic instead of directly forwarding the traffic to the intended endpoint. In addition one could set up a binding in a way that the source IP address used is one of a relay so that traffic coming back can be snooped on or interfered with. The operator also needs to consider security threats resulting from unplanned termination of the DNCA session. Unplanned session termination, which could e.g. happen due to an attacker taking down the NAT-controller, leads to the NAT-device cleaning up the state associated with this session after a grace period. If the grace period is set to zero, the endpoint will experience an immediate loss of connectivity to services reachable through the NAT-device following the termination of the DNCA session.The protections on DNCA and its

Diameter protocol exchanges don’t prevent such abuses of NAT-control. Prevention of mis-use or mis-configuration of a NAT-device by an authorized NAT-controller is beyond the scope of this protocol specification. A service provider deploying DNCA needs to make sure that higher layer processes and procedures are put in place which allow them to detect and mitigate misuses.

13. Examples

This section shows example DNCA message content and exchange.

13.1. DNCA Session Establishment Example

Figure 15 depicts a typical call flow for DNCA session establishment.

In this example, the NAT-controller:

a. requests a maximum of 100 NAT-bindings for the endpoint.

b. defines a static binding for a TCP connection which associates the internal IP-Address:Port 192.0.2.1:80 with the external IP-Address:Port 198.51.100.1:80 for the endpoint.

c. requests the use of a preconfigured template called "local-policy" while creating NAT-bindings for the endpoint.

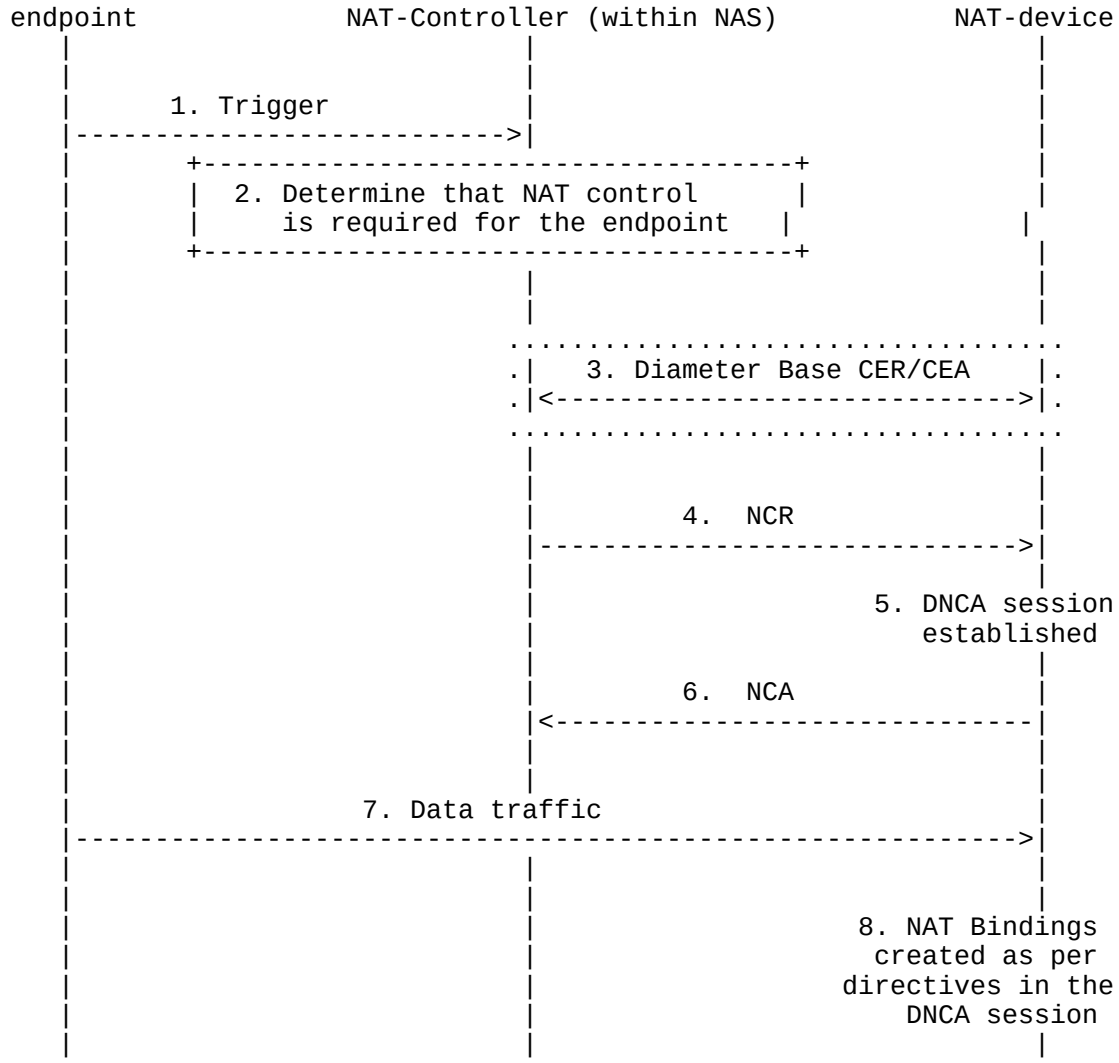


Figure 15: Initial NAT control request and session establishment example

Detailed description of the steps shown in Figure 15:

1. The NAT-controller (co-located with the NAS here) creates state for an endpoint based on a trigger. This could for example be the successful establishment of a Point-to-Point Protocol (PPP) [RFC1661] access session.

2. Based on the configuration of the DNCA Diameter peer within the NAT-controller, the NAT-controller determines that NAT-control is required and is to be enforced at a NAT-device.

3. If there is no Diameter session already established with the DNCA Diameter peer within NAT-device, a Diameter connection is established and Diameter Base CER/CEA are exchanged.

4. The NAT-Controller creates an NCR message (see below) and sends it to the NAT-device. This example shows IPv4 to IPv4 address and port translation. For IPv6 to IPv4 translation, the Framed-IP-Address AVP would be replaced by the Framed-IPv6-Address AVP with the value set to the IPv6 address of the endpoint.

```
< NC-Request > ::= < Diameter Header: TBD.COM-CODE, REQ, PXY>
                   Session-Id =  "natC.example.com:33041;23432;"
                   Auth-Application-Id = <DNCA Application ID>
                   Origin-Host = "natC.example.com"
                   Origin-Realm = "example.com"
                   Destination-Realm = "example.com"
                   Destination-Host = "nat-device.example.com"
                   NC-Request-Type = INITIAL_REQUEST
                   User-Name = "subscriber_example1"
                   Framed-IP-Address = "192.0.2.1"
                   NAT-Control-Install = {
                        NAT-Control-Definition = {
                           Protocol = TCP
                           Direction = OUT
                           NAT-Internal-Address = {
                                Framed-IP-Address = "192.0.2.1"
                                Port = 80
                           }
                           NAT-External-Address = {
                                Framed-IP-Address = "198.51.100.1"
                                Port = 80
                           }
                        }
                        Max-NAT-Bindings = 100
                        NAT-Control-Binding-Template = "local-policy"
                  }
```

5. The NAT-device establishes a DNCA session as it is able to comply with the request.

6. The NAT-device sends an NCA to indicate the successful completion of the request.

```
<NC-Answer> ::= < Diameter Header: TBD.COM-CODE, PXY >
                  Session-Id =  "natC.example.com:33041;23432;"
                  Origin-Host = "nat-device.example.com"
                  Origin-Realm = "example.com"
                  NC-Request-Type = INITIAL_REQUEST
                  Result-Code = DIAMETER_SUCCESS
```

7. The endpoint sends packets that reach the NAT-device.

8. The NAT-device performs NAT for traffic received from the endpoint with source address 192.0.2.1. Traffic with source IP-address 192.0.2.1 and port 80 are translated to the external IP-address 198.51.100.1 and port 80. Traffic with source IP-address 192.0.2.1 and a source port different from 80 will be translated to IP-address 198.51.100.1 and a port chosen by the NAT-device. Note that this example assumes that the NAT-device follows typical binding allocation rules for endpoints, in that only a single external IP-address is used for all traffic received from a single IP-address of an endpoint. The NAT-device will allow a maximum of 100 NAT-bindings be created for the endpoint.

13.2. DNCA Session Update with Port Style Example

This section gives an example for a DNCA session update: A new set of NAT-bindings is requested for an existing session. The request contains a directive (the "NAT-External-Port-Style" AVP set to FOLLOW_INTERNAL_PORT_STYLE) that directs the NAT-device to maintain port-sequence and port-oddity for the newly created NAT-bindings. In the example shown, the internal ports are UDP port 1036 and 1037. The NAT-device follows the directive selects the external ports accordingly. The NAT-device would for example create a mapping of 192.0.2.1:1036 to 198.51.100.1:5056 and 192.0.2.1:1037 to 198.51.100.1:5057, thereby maintaining port oddity (1036->5056, 1037->5057) and sequence (the consecutive internal ports 1036 and 1037 map to the consecutive external ports 5056 and 5057).

```
< NC-Request > ::= < Diameter Header: TBD.COM-CODE, REQ, PXY>
                   Session-Id =  "natC.example.com:33041;23432;"
                   Auth-Application-Id = <DNCA Application ID>
                   Origin-Host = "natC.example.com"
                   Origin-Realm = "example.com"
                   Destination-Realm = "example.com"
                   Destination-Host = "nat-device.example.com"
                   NC-Request-Type = UPDATE_REQUEST
                   NAT-Control-Install = {
                       NAT-Control-Definition = {
                           Protocol = UDP
                           Direction = OUT
                           NAT-Internal-Address = {
                                Framed-IP-Address = "192.0.2.1"
                                Port = 1035
                           }
                       }
                       NAT-Control-Definition = {
                           Protocol = UDP
                           Direction = OUT
                           NAT-Internal-Address = {
                                Framed-IP-Address = "192.0.2.1"
                                Port = 1036
                           }
                       }
                       NAT-External-Port-
                              Style = FOLLOW_INTERNAL_PORT_STYLE
                   }
```

13.3. DNCA Session Query Example

This section shows an example for DNCA session query for a subscriber whose internal IP-Address is 192.0.2.1.

```
< NC-Request > ::= < Diameter Header: TBD.COM-CODE, REQ, PXY>
                   Auth-Application-Id = <DNCA Application ID>
                   Origin-Host = "natC.example.com"
                   Origin-Realm = "example.com"
                   Destination-Realm = "example.com"
                   Destination-Host = "nat-device.example.com"
                   NC-Request-Type = QUERY_REQUEST
                   Framed-IP-Address = "192.0.2.1"
```

The NAT-device constructs an NCA to report all currently active NAT-bindings whose internal address is 192.0.2.1.

```
      <NC-Answer> ::= < Diameter Header: TBD.COM-CODE, PXY >
                     Origin-Host = "nat-device.example.com"
                     Origin-Realm = "example.com"
                     NC-Request-Type = QUERY_REQUEST
                     NAT-Control-Definition = {
                             Protocol = TCP
                             Direction = OUT
                             NAT-Internal-Address = {
                                 Framed-IP-Address = "192.0.2.1"
                                 Port = 80
                                }
                             NAT-External-Address = {
                                  Framed-IP-Address = "198.51.100.1"
                                  Port = 80
                                }
                             Session-Id = "natC.example.com:33041;23432;"
                     }
                     NAT-Control-Definition = {
                             Protocol = TCP
                             Direction = OUT
                             NAT-Internal-Address = {
                                 Framed-IP-Address = "192.0.2.1"
                                 Port = 1036
                                }
                             NAT-External-Address = {
                                  Framed-IP-Address = "198.51.100.1"
                                  Port = 5056
                                }
                             Session-Id = "natC.example.com:33041;23432;"
                     }
                     NAT-Control-Definition = {
                             Protocol = TCP
                             Direction = OUT
                             NAT-Internal-Address = {
                                 Framed-IP-Address = "192.0.2.1"
                                 Port = 1037
                                }
                             NAT-External-Address = {
                                  Framed-IP-Address = "198.51.100.1"
                                  Port = 5057
                                }
                             Session-Id = "natC.example.com:33041;23432;"
                     }
```

13.4. DNCA Session Termination Example

In this example the NAT-controller decides to terminate the previously established DNCA session. This could for example be the

case as a result of an access session (e.g. a PPP session) associated with an endpoint been torn down.

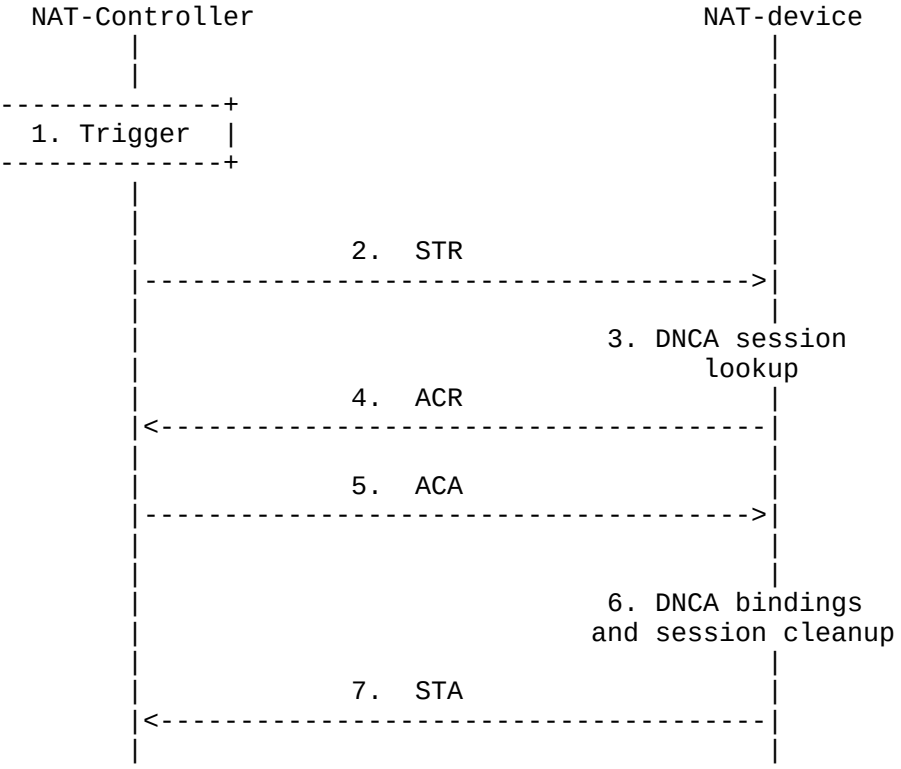


Figure 20: NAT control session termination example

The following steps describe the sequence of events for tearing down the DNCA session in the example above:

1. The NAT-controller receives a trigger that a DNCA session associated with a specific endpoint should be terminated. An example event could be the termination of the PPP [RFC1661] access session to an endpoint in a NAS. The NAS correspondingly triggers the NAT-controller request tear-down of the associated DNCA session.

2. The NAT-controller creates the required NCR message and sends it to the NAT-device:

```
< STR >     ::= < Diameter Header: 275, REQ, PXY>
                Session-Id =  "natC.example.com:33041;23432;"
                Auth-Application-Id = <DNCA Application ID>
                Origin-Host = "natC.example.com"
                Origin-Realm = "example.com"
                Destination-Realm = "example.com"
                Destination-Host = "nat-device.example.com"
                Termination-Cause = DIAMETER_LOGOUT
```

3. The NAT-device looks up the DNCA session based on the Session-Id AVP and finds a previously established active session.

4. The NAT-device reports all NAT-bindings established for that subscriber using an ACR:

```
< ACR >     ::= < Diameter Header: 271, REQ, PXY>
                Session-Id =  "natC.example.com:33041;23432;"
                Auth-Application-Id = <DNCA Application ID>
                Origin-Host = "nat-device.example.com"
                Origin-Realm = "example.com"
                Destination-Realm = "example.com"
                Destination-Host = "natC.example.com"
                Accounting-Record-Type = STOP_RECORD
                Accounting-Record-Number = 1
                NAT-Control-Record = {
                    NAT-Control-Definition = {
                        Protocol = TCP
                        Direction = OUT
                        NAT-Internal-Address = {
                            Framed-IP-Address = "192.0.2.1"
                            Port = 5001
                           }
                        NAT-External-Address = {
                             Framed-IP-Address = "198.51.100.1"
                             Port = 7777
                           }
                       }
                      NAT-Control-Binding-Status = Removed
                   }
```

5. The NAT-controller receives and processes the ACR as per its configuration. It responds with an ACA to the NAT-device.

```
<ACA>      ::= < Diameter Header: 271, PXY >
                 Session-Id =  "natC.example.com:33041;23432;"
                 Origin-Host = "natC.example.com"
                 Origin-Realm = "example.com"
                 Result-Code = DIAMETER_SUCCESS
                 Accounting-Record-Type = STOP_RECORD
                 Accounting-Record-Number = 1
```

6. On receipt of the ACA the NAT-device cleans up all NAT-bindings and associated session state for the endpoint.

7. NAT-device sends an STA. On receipt of the STA the NAT-controller will clean up the corresponding session state.

```
<STA>      ::= < Diameter Header: 275, PXY >
                 Session-Id =  "natC.example.com:33041;23432;"
                 Origin-Host = "nat-device.example.com"
                 Origin-Realm = "example.com"
                 Result-Code = DIAMETER_SUCCESS
```

14. Acknowledgements

The authors would like to thank Jari Arkko, Wesley Eddy, Stephen Farrell, Miguel A. Garcia, David Harrington, Jouni Korhonen, Matt Lepinski, Avi Lior, Chris Metz, Pallavi Mishra, Lionel Morand, Robert Sparks, Martin Stiemerling, Dave Thaler, Hannes Tschofenig, Sean Turner, Shashank Vikram, Greg Weber, and Glen Zorn for their input on this document.

15. Change History (to be removed prior to publication as an RFC)

Changes from -00 to -01

a. new values for Result-Code AVP used - instead of Experimental-Result AVP

b. added support for transport specific binding (UDP/TCP)

c. added support for twice-NAT

d. clarified the use of the two different types of query-requests

Changes from -01 to -02

a. Reference to pull mode removed, session initiation event clarified in section 4.1

b. added Redirect-* AVPs in NCA command

c. Removed reference to Called-Station-Id AVP in NCR command

d. Editorial changes

e. added support for bindings providing AFT (NAT64)

Changes from -02 to -03

a. Editorial changes

Changes from -03 to -04

a. Editorial changes suggested in WG last call review

b. Removed NCR Request type terminate and replaced with STR

c. All references to Auth-Session-State are removed and a new section to describe FSM for Manager and Agent has been added

d. Clarified reuse of External address and address pools among multiple subscribers

Changes from -04 to -05

a. Removed references to Large Scale NAT as per review comments

Changes from -05 to -06

a. Editorial changes

Changes from -06 to -07

a. Added a note in section 4.3 stating the state of pre-existing bindings on update failure

b. Security considerations are made consistent between sections 5.1 and 12

c. Editorial changes

Changes from -07 to -08

a. Added section 4.6 to describe session abort

b. Editorial changes

c. Nomenclature change: From DNCA Agent/Manager to DNCA Diameter peers identifying the location where they reside (NAT-controller or NAT-device)

d. IANA consideration Section format changes

e. Updated security section (included considerations directly, rather than referring to Diameter QoS similarities).

Changes from -08 to -09

a. expanded on the need for an SP controlling the maximum number of bindings of an endpoint (see introduction section)

b. added a paragraph in the security section outlining general mis-uses of NAT-control (non specific to DNCA), with DNCA being an example of such a NAT-control protocol

c. editorial changes

Changes from -09 to -10

a. Section 4 and security considerations updated with RFC 2119 language

b. NAT-External-Port-Style AVP added to aid external port oddity requirement as per MIDCOM framework

c. NAT related RFCs added in normative reference

d. Section 13 added to provide example DNCA message exchange flows

e. Added a description to provide DNCA comparison with MIDCOM

f. n:1 deployment model for NAT-controllers and NAT-devices explicitly specified

g. editorial changes as per IESG DISCUSS comments

Changes from -10 to -11

a. clarified DNCA session query to be done after Diameter session is established

b. Section 4.4 Session Termination updated to specify resource cleanup at NAT-Device upon session termination

c. Removed Framed-IP-Netmask AVP from NAT-External-Address as external address is fully defined by Framed-IP-Address AVP

d. Updated Section 12 to highlight Session-Id to be chosen such that it is hard to guess

e. editorial changes as per IESG DISCUSS

Changes from -11 to -12

a. endpoint replaces references to end point and user and defines what Endpoint means in this draft

b. editorial changes as per IESG DISCUSS

Changes from -12 to -13

a. Section 4.3 session query updated to use NAT-External-Address for external IP-address based query

Changes from -13 to -14

a. Added NAT-External-Address in NC-request for session query by external IP-address

b. Reordered all mandatory AVPs in NCR and NCA to appear before optional AVPs

Changes from -14 to -15

a. As part of IESG discuss - clarified that multiple methods if used along with DNCA for NAT control should be configured to prevent conflict.

b. Clarified misuse of NAT-device by a Diameter authorized NAT-controller using DNCA is beyond the scope of this protocol specification.

c. Editorial updates.

Changes from -15 to -16

a. Extended section covering case of a single NAT-device controlled by multiple NAT-ontrollers which use different protocols for configuring the NAT-device.

b. Added NAT-device state cleanup in case of unexpected/unplanned termination of Diameter session or application either on NAT-controller or NAT-device.

c. Added MAX_BINDINGS_SET_FAILURE failure case (for those scenarios where the maximum number of bindings cannot be set by the controller)

Change from -16 to -17

a. Clarified that the endpoint identifier Framed-IP-Address and the internal address in NAT-Internal-Address specified to install NAT-bindings for the session MUST match.

16. References

16.1. Normative References

[ETSIES283034] ETSI, "Telecommunications and Internet Converged Services and Protocols for Advanced Networks (TISPAN),Network Attachment Sub-System (NASS),e4 interface based on the Diameter protocol.", September 2008.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC4005] Calhoun, P., Zorn, G., Spence, D., and D. Mitton, "Diameter Network Access Server Application", RFC 4005, August 2005.

[RFC4675] Congdon, P., Sanchez, M., and B. Aboba, "RADIUS Attributes for Virtual LAN and Priority Support", RFC 4675, September 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Jones, M., and A. Lior, "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, February 2010.

16.2. Informative References

[I-D.ietf-behave-lsn-requirements]
Perreault, S., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common requirements for Carrier Grade NATs (CGNs)", draft-ietf-behave-lsn-requirements-05 (work in progress), November 2011.

[RFC1661] Simpson, W., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[RFC3303] Srisuresh, P., Kuthan, J., Rosenberg, J., Molitor, A., and A. Rayhan, "Middlebox communication architecture and framework", RFC 3303, August 2002.

[RFC3304] Swale, R., Mart, P., Sijben, P., Brim, S., and M. Shore, "Middlebox Communications (midcom) Protocol Requirements", RFC 3304, August 2002.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC4097] Barnes, M., "Middlebox Communications (MIDCOM) Protocol Evaluation", RFC 4097, June 2005.

[RFC5189] Stiemerling, M., Quittek, J., and T. Taylor, "Middlebox Communication (MIDCOM) Protocol Semantics", RFC 5189, March 2008.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6

Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

Authors' Addresses

Frank Brockners
Cisco
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Shwetha Bhandari
Cisco
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Email: shwethab@cisco.com

Vaneeta Singh
18, Cambridge Road
Bangalore 560008
India

Email: vaneeta.singh@gmail.com

Victor Fajardo
Telcordia Technologies
1 Telcordia Drive #1S-222
Piscataway, NJ 08854
USA

Email: vf0213@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 3, 2013

G. Zorn
Network Zen
Q. Wu
Huawei
J. Korhonen
NSN

August 2, 2012

Diameter Support for Proxy Mobile IPv6 Localized Routing
draft-ietf-dime-pmip6-lr-18

Abstract

In Proxy Mobile IPv6, packets received from a Mobile Node (MN) by the Mobile Access Gateway (MAG) to which it is attached are typically tunneled to a Local Mobility Anchor (LMA) for routing. The term "localized routing" refers to a method by which packets are routed directly between an MN's MAG and the MAG of its Correspondent Node (CN) without involving any LMA. In a Proxy Mobile IPv6 deployment, it may be desirable to control the establishment of localized routing sessions between two MAGs in a Proxy Mobile IPv6 domain by requiring that the session be authorized. This document specifies how to accomplish this using the Diameter protocol.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 3, 2013.

Copyright Notice

Table of Contents

1. Introduction

Proxy Mobile IPv6 (PMIPv6) [RFC5213] allows the Mobility Access Gateway (MAG) to optimize media delivery by locally routing packets from a Mobile Node to a Correspondent Node that is locally attached to an access link connected to the same Mobile Access Gateway, avoiding tunneling them to the Mobile Node's Local Mobility Anchor (LMA). This is referred to as "local routing" in RFC 5213. However, this mechanism is not applicable to the typical scenarios in which the MN and CN are connected to different MAGs and are registered to the same LMA or different LMAs. [RFC6279] takes those typical scenarios into account and defines the problem statement for PMIPv6 localized routing. [I-D.ietf-netext-pmip-lr] specifies the PMIPv6 localized routing protocol based on the scenarios A11, A12, and A21 [RFC6279], which is used to establish a localized routing path between two Mobile Access Gateways in a PMIPv6 domain.

However, there is no relevant work discussing how AAA-based mechanisms can be used to provide authorization to the Mobile Node's MAG or LMA for enabling localized routing between MAGs.

This document describes Diameter [I-D.ietf-dime-rfc3588bis] support for the authorization of PMIPv6 mobility entities in case of A11,A12,A21 during localized routing.

2. 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].

3. Solution Overview

This document addresses how to provide authorization to the Mobile Node's MAG or LMA for enabling localized routing and resolve the destination MN's MAG by means of interaction between the LMA and the AAA server. Figure 1 shows the reference architecture for Localized Routing Service Authorization. This reference architecture assumes that

- If MN and CN belong to different LMAs, MN and CN should share the same MAG (i.e.,A12 described in [RFC6279]), e.g., MN1 and CN2 in Figure 1 are attached to the same MAG1 and belong to LMA1 and LMA2 respectively. Note that LMA1 and LMA2 in Figure 1 are in the same provider domain (as described in [RFC6279]).

- o If MN and CN are attached to the different MAGs, MN and CN should belong to the same LMA (i.e.,A21 described in [RFC6279]), e.g., MN1 and CN3 in theFigure 1 are attached to the MAG1 and MAG3 respectively but belong to LMA1.

- o MN and CN may belong to the same LMA and are attached to the same MAG(i.e.,A11 described in [RFC6279]), e.g.,MN1 and CN1 in the Figure 1 are both attached to the MAG1 and belong to LMA1.

- o The MAG and LMA support Diameter client functionality.

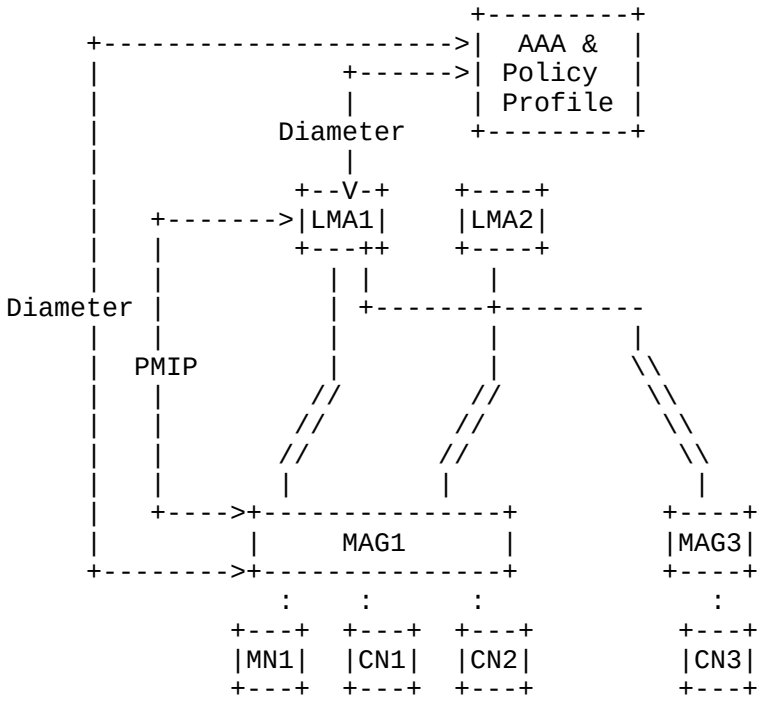


Figure 1: Localized Routing Service Authorization Reference Architecture

The interaction of the MAG and LMA with the AAA server according to the extension specified in this document is used to authorize the localized routing service.

4. Attribute Value Pair Used in this Document

This section describes Attribute Value Pairs (AVPs) defined by this

specification or re-used from existing specifications in a PMIPv6-specific way.

4.1. User-Name AVP

The User-Name AVP (AVP Code 1) is defined in [I-D.ietf-dime-rfc3588bis]. This AVP is used to carry the MN-Identifier (Mobile Node identifier) [RFC5213] in the AA-Request (AAR) message [I-D.ietf-dime-rfc4005bis].

4.2. PMIP6-IPv4-Home-Address AVP

The PMIP6-IPv4-Home-Address AVP (AVP Code 505) is defined in [RFC5779]. This AVP is used to carry the IPv4-MN-HoA (Mobile Node's IPv4 home address)[RFC5844] in the AA-Request (AAR) message [I-D.ietf-dime-rfc4005bis].

4.3. MIP6-Home-Link-Prefix AVP

The MIP6-Home-Link-Prefix AVP (AVP Code 125) is defined in [RFC5779]. This AVP is used to carry the MN-HNP (Mobile Node's home network prefix) in the AAR.

4.4. MIP6-Feature-Vector AVP

The MIP6-Feature-Vector AVP is defined in [RFC5447]. This document allocates a new capability flag bit according to the IANA rules in RFC 5447.

INTER_MAG_ROUTING_SUPPORTED (TBD)

Direct routing of IP packets between MNs anchored to different MAGs without involving any LMA is supported. This bit is used with MN-Identifier. When a MAG or LMA sets this bit in the MIP6-Feature-Vector and MN-Identifier corresponding to the Mobile Node is carried with this bit, it indicates to the home AAA server (HAAA) that the Mobile Node associated with this LMA is allowed to use localized routing.If this bit is cleared and MN-Identifier corresponding to the Mobile Node is carried with this bit, it indicates to the home AAA server (HAAA) that the Mobile Node associated with this LMA is not allowed to use localized routing. When a MAG or LMA sets this bit in the MIP6-Feature-Vector and MN-Identifiers corresponding to the Mobile Node and Correspondent Node are both carried with this bit, it indicates to the HAAA that localized routing of IP packets between Mobile Node and Correspondent Node anchored to different MAGs is supported. If this bit is cleared and MN- Identifiers corresponding to the Mobile Node and Correspondent Node are both carried with this bit

to HAAA, it indicates to the HAAA that localized routing of IP packets between Mobile Node and Correspondent Node anchored to different MAGs is not supported. If this bit is cleared in the returned MIP6-Feature-Vector AVP, the HAAA does not authorize direct routing of packets between MNs anchored to the different MAG. The MAG and LMA MUST support this policy feature on a per-MN and per-subscription basis.

5. Example Signaling Flows for Localized Routing Service Authorization

Localized Routing Service Authorization can happen during the network access authentication procedure [RFC5779] before localized routing is initialized. In this case, the preauthorized pairs of LMA/prefix sets can be downloaded to Proxy Mobile IPv6 entities during the RFC 5779 procedure. Localized routing can be initiated once the destination of a received packet matches one or more of the prefixes received during the RFC 5779 procedure.

Figure 2 shows an example scenario in which MAG1 acts as a Diameter client, processing the data packet from MN1 to MN2 and requesting authorization of localized routing (i.e.,MAG-Initiated LR authorization). In this example scenario, MN1 and MN2 are attached to the same MAG and anchored to the different LMAs (i.e.,A12 described in [RFC6279]). In this case, MAG1 knows that MN2 belongs to a different LMA (which can be determined by looking up the binding cache entries corresponding to MN1 and MN2 and comparing the addresses of LMA1 and LMA2). In order to setup a localized routing path with MAG2, MAG1 acts as Diameter client and sends an AAR message to the Diameter server. The message contains an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the LOCAL_MAG_ROUTING_SUPPORTED bit ([RFC5779],Section 5.5) set,two instances of the User-Name AVP ([I-D.ietf-dime-rfc3588bis], Section 8.14)containing MN1-Identifier and MN2-Identifier. In addition, the message may contain either an instance of the MIP6-Home-Link-Prefix AVP ([RFC5779], Section 5.3) or an instance of the PMIP6-IPv4- Home-Address AVP ([RFC5779], Section 5.2) containing the IP address/ HNP of MN1.

The Diameter server authorizes localized routing service by checking if MN1 and MN2 are allowed to use localized routing. If so, the Diameter server responds with an AAA message encapsulating an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the the LOCAL_MAG_ROUTING_SUPPORTED bit ([RFC5779],Section 5.5) set indicating direct routing of IP packets between MNs anchored to the same MAG is supported. MAG1 then knows the localized routing between MN1 and MN2 is allowed. Then MAG1 sends the Request messages respectively to LMA1 and LMA2. The

request message is the Localized Routing Initialization (LRI) message in Figure 2 and belongs to the Initial phase of the localized routing. LMA1 and LMA2 responds to MAG1 using the Localized Routing Acknowledge message (LRA inFigure 2) in accordance with [I-D.ietf-netext-pmip-lr].

In case of LRA_WAIT_TIME expiration [I-D.ietf-netext-pmip-lr],MAG1 should ask for authorization of localized routing again according to the procedure described above before LRI is retransmitted up to a maximum of LRI_RETRIES.

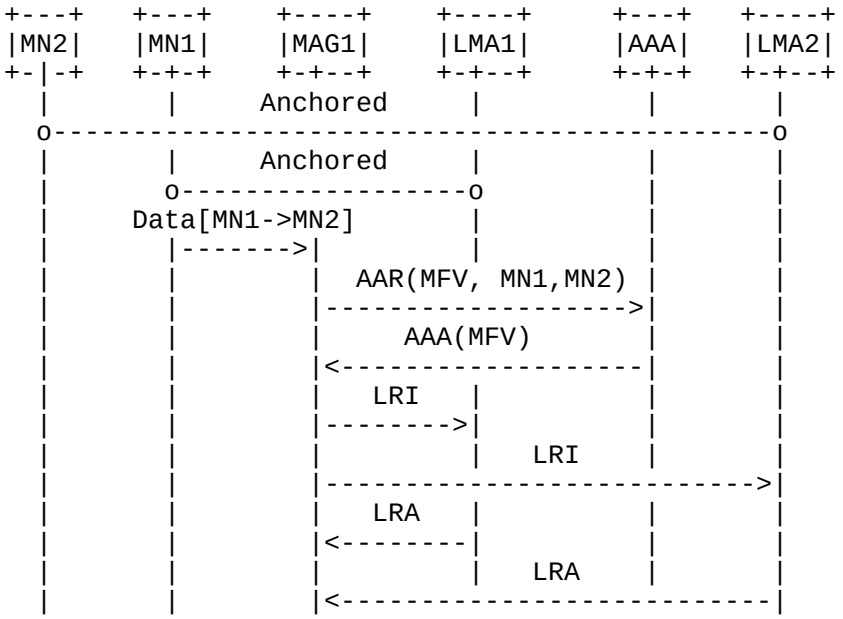


Figure 2: MAG-initiated Localized Routing Authorization in A12

Figure 3 shows the second example scenario, in which LMA1 acts as a Diameter client, processing the data packet from MN2 to MN1 and requesting the authorization of localized routing. In this scenario, MN1 and MN2 are attached to the different MAG and anchored to the same LMA (i.e., A21 described in [RFC6279]), LMA knows that MN1 and MN2 belong to the same LMA (which can be determined by looking up the binding cache entries corresponding to MN1 and MN2 and comparing the addresses of LMA corresponding to MN1 and LMA corresponding to MN2). In contrast with the signaling flow shown in Figure 2, it is LMA1 instead of MAG1 which initiates the setup of the localized routing path.

The Diameter client in LMA1 sends an AA-Request message to the Diameter server. The message contains an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the INTER_MAG_ROUTING_SUPPORTED bit (Section 4.5) set indicating direct routing of IP packets between MNs anchored to different MAGs is supported and two instances of the User-Name AVP ([I-D.ietf-dime-rfc3588bis], Section 8.14)containing MN1-Identifier and MN2-Identifier. The Diameter server authorizes the localized routing service by checking if MN1 and MN2 are allowed to use localized routing. If so, the Diameter server responds with an AA-Answer message encapsulating an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the INTER_MAG_ROUTING_SUPPORTED bit (Section 4.5) set indicating direct routing of IP packets between MNs anchored to different MAGs is supported. LMA1 then knows the localized routing is allowed. In success case, LMA1 responds to MAG1 in accordance with [I-D.ietf-netext-pmip-lr].

In case of LRA_WAIT_TIME expiration [I-D.ietf-netext-pmip-lr],LMA1 should ask for authorization of localized routing again according to the procedure described above before LRI is retransmitted up to a maximum of LRI_RETRIES.

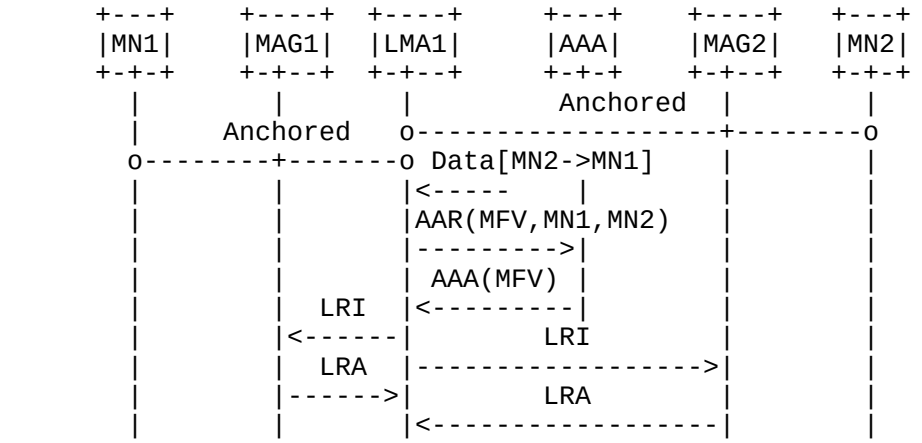


Figure 3: LMA-initiated Localized Routing Authorization in A21

Figure 4 shows another example scenario, in which LMA1 acts as a Diameter client, processing the data packet from MN2 to MN1 and requesting the authorization of localized routing. In this scenario, MN1 and MN2 are attached to the same MAG and anchored to the same LMA (i.e., A11 described in [RFC6279]), LMA knows that MN1 and MN2 belong to the same LMA (which can be determined by looking up the binding

cache entries corresponding to MN1 and MN2 and comparing the addresses of LMA corresponding to MN1 and LMA corresponding to MN2).

The Diameter client in LMA1 sends an AA-Request message to the Diameter server. The message contains an instance of the MIP6-Feature-Vector AVP ([RFC5447], Section 4.2.5) with the LOCAL_MAG_ROUTING_SUPPORTED bit set and two instances of the User-Name AVP ([I-D.ietf-dime-rfc3588bis], Section 8.14)containing MN1-Identifier and MN2-Identifier. The Diameter server authorizes the localized routing service by checking if MN1 and MN2 are allowed to use localized routing. If so, the Diameter server responds with an AA- Answer message encapsulating an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the LOCAL_MAG_ROUTING_SUPPORTED bit ([RFC5779],Section 5.5) set indicating direct routing of IP packets between MNs anchored to the same MAG is supported. LMA1 then knows the localized routing is allowed and responds to MAG1 for localized routing in accordance with [I-D.ietf-netext-pmip-lr].

In case of LRA_WAIT_TIME expiration [I-D.ietf-netext-pmip-lr], LMA1 should ask for authorization of localized routing again according to the procedure described above before LRI is retransmitted up to a maximum of LRI_RETRIES.

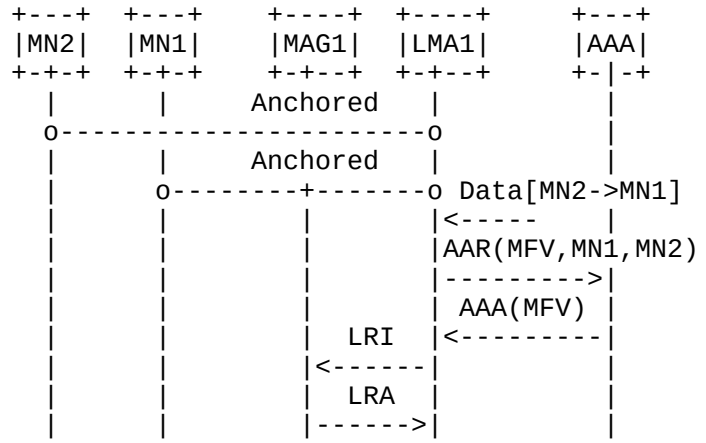


Figure 4: LMA-initiated Localized Routing Authorization in A11

6. Security Considerations

The security considerations for the Diameter NASREQ [I-D.ietf-dime-rfc4005bis] and Diameter Proxy Mobile IPv6 [RFC5779] applications are also applicable to this document.

The service authorization solicited by the MAG or the LMA relies upon the existing trust relationship between the MAG/LMA and the AAA server.

An authorised MAG could in principle track the movement of any participating CNs at the level of the MAG to which they are anchored. If such a MAG were compromised, or under the control of a bad-actor, then such tracking could represent a privacy breach for the set of tracked CNs. In such a case, the traffic pattern from the compromised MAG might be notable so monitoring for e.g. excessive queries from MAGs might be worthwhile.

7. IANA Considerations

This specification defines a new value in the Mobility Capability registry [RFC5447] for use with the MIP6-Feature-Vector AVP: INTER_MAG_ROUTING_SUPPORTED (see Section 4.4).

8. Contributors

Paulo Loureiro, Jinwei Xia and Yungui Wang all contributed to early versions of this document.

9. Acknowledgements

The authors would like to thank Marco Liebsch, Carlos Jesus Bernardos Cano, Dan Romascanu, Elwyn Davies, Basavaraj Patil, Ralph Droms, Stephen Farrel,Robert Sparks, Benoit Claise and Abhay Roy for their valuable comments and suggestions on this document.

10. References

10.1. Normative References

[I-D.ietf-dime-rfc3588bis]
Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", draft-ietf-dime-rfc3588bis-34 (work in progress), June 2012.

[I-D.ietf-dime-rfc4005bis]
Zorn, G., "Diameter Network Access Server Application", draft-ietf-dime-rfc4005bis-11 (work in progress), July 2012.

[I-D.ietf-netext-pmip-lr]
Krishnan, S., Koodli, R., Loureiro, P., Wu, Q., and A. Dutta, "Localized Routing for Proxy Mobile IPv6", draft-ietf-netext-pmip-lr-10 (work in progress), May 2012.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5213] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, August 2008.

[RFC5447] Korhonen, J., Bournelle, J., Tschofenig, H., Perkins, C., and K. Chowdhury, "Diameter Mobile IPv6: Support for Network Access Server to Diameter Server Interaction", RFC 5447, February 2009.

[RFC5779] Korhonen, J., Bournelle, J., Chowdhury, K., Muhanna, A., and U. Meyer, "Diameter Proxy Mobile IPv6: Mobile Access Gateway and Local Mobility Anchor Interaction with Diameter Server", RFC 5779, February 2010.

[RFC5844] Wakikawa, R. and S. Gundavelli, "IPv4 Support for Proxy Mobile IPv6", RFC 5844, May 2010.

10.2. Informative References

[RFC6279] Liebsch, M., Jeong, S., and Q. Wu, "Proxy Mobile IPv6 (PMIPv6) Localized Routing Problem Statement", RFC 6279, June 2011.

Authors' Addresses

Glen Zorn
Network Zen
227/358 Thanon Sanphawut
Bang Na, Bangkok 10260
Thailand

Phone: +66 (0) 87-040-4617
Email: glenzorn@gmail.com

Qin Wu
Huawei Technologies Co., Ltd.
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 21001
China

Phone: +86-25-84565892
Email: sunseawq@huawei.com

Jouni Korhonen
Nokia Siemens Networks
Linnoitustie 6
Espoo FI-02600,
Finland

Email: jouni.nospam@gmail.com

Diameter Maintenance and
Extensions (DIME)
Internet-Draft
Intended status: Standards Track

K. Carlberg, Ed.
G11
T. Taylor
PT Taylor Consulting
June 28, 2012

Diameter Priority Attribute Value Pairs
draft-ietf-dime-priority-avps-06.txt

Status of this Memo

Copyright Notice

Abstract

This document defines Attribute-Value Pair (AVP) containers for various priority parameters for use with Diameter and the AAA framework. The parameters themselves are defined in several different protocols that operate at either the network or application layer.

1. Introduction

This document defines a number of Attribute-Value Pairs (AVP) that can be used within the Diameter protocol [I-D.ietf-dime-rfc3588bis] to convey a specific set of priority parameters. These parameters are specified in other documents, but are briefly described below. The corresponding AVPs defined in Section 3 are an extension to those defined in [RFC5866]. We note that all the priority fields associated with the AVPs defined in this document are extensible and allow for additional values beyond what may have already been defined or registered with IANA.

Priority influences the distribution of resources, and in turn the QoS associated with that resource. This influence may be probabilistic, ranging between (but not including) 0% and 100%, or it may be in the form of a guarantee to either receive or not receive the resource.

Another example of how prioritization can be realized is articulated in Appendix A.3 (the priority by-pass model) of [RFC6401]. In this case, prioritized flows may gain access to resources that are never shared with non-prioritized flows.

1.1 Other Priority-Related AVPs

3rd Generation Partnership Project (3GPP) has defined several Diameter AVPs that support prioritization of sessions. The following AVPs are intended to be used for priority services (e.g., Multimedia Priority Service):

- Reservation-Priority AVP as defined in [ETSI]
- MPS-Identifier AVP as defined in [3GPPa]
- Priority-Level AVP (as part of the Allocation Retention Priority AVP) as defined in [3GPPb]
- Session-Priority AVP as defined in [3GPPc][3GPPd]

Both the Reservation-Priority AVP and the Priority-Level AVP can carry priority levels associated with a session initiated by a user. We note that these AVPs are defined from the allotment set aside for 3GPP for Diameter-based interfaces and are particularly aimed at IP Multimedia Subsystem (IMS) deployment environments. The above AVPs defined by 3GPP are to be viewed as private implementations operating within a walled

garden. In contrast, the priority related AVPs defined below in Section 3 are not constrained to IMS environments. The potential applicability or use case scenarios that involve coexistance between the above 3GPP defined priority related AVPs and those defined below in Section 3 is for further study.

2. Terminology and Abbreviations

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 RFC2119 [RFC2119].

3. Priority Parameter Encoding

This section defines a set of AVPs that correlate to priority fields defined in other protocols. This set of priority related AVPs is for use with the DIAMETER QoS application [RFC5866] and represents a continuation of the list of AVPs defined in [RFC5624]. The syntax notation used is that of [I-D.ietf-dime-rfc3588bis]. We note that the following subsections describe the prioritization field of a given protocol as well as the structure of the AVP corresponding to that field.

We stress that neither the priority related AVPs, nor the Diameter protocol, perform or realize QoS for a session or flow of packets. Rather, these AVPs are part of a mechanism to determine validation of the priority value.

3.1. Dual-Priority AVP

The Dual-Priority AVP is a grouped AVP consisting of two AVPs; the Preemption-Priority and the Defending-Priority AVP. These AVPs are derived from the corresponding priority fields specified in the Signaled Preemption Priority Policy Element [RFC3181] of RSVP [RFC2205].

In [RFC3181], the Defending-Priority value is set when the reservation has been admitted by the RSVP protocol. The Preemption-Priority field in [RFC3181] of a newly requested reservation is compared with the Defending-Priority value of a previously admitted flow. The actions taken based upon the result of this comparison are a function of local policy.

```
Dual-Priority  ::= < AVP Header: TBD >
            { Preemption-Priority }
            { Defending-Priority }
```

3.1.1. Preemption-Priority AVP

The Preemption-Priority AVP (AVP Code TBD) is of type Unsigned16. Higher values represent higher priority. The value encoded in this AVP is the same as the preemption priority value that would be encoded in the signaled preemption priority policy element.

3.1.2. Defending-Priority AVP

The Defending-Priority AVP (AVP Code TBD) is of type Unsigned16. Higher values represent higher priority. The value encoded in this AVP is the same as the defending priority value that would be encoded in the signaled preemption priority policy element.

3.2. Admission-Priority AVP

The Admission-Priority AVP (AVP Code TBD) is of type Unsigned8. The admission priority associated with an RSVP flow is used to increase the probability of session establishment for selected RSVP flows. Higher values represent higher priority. A given admission priority is encoded in this information element using the same value as when encoded in the admission priority parameter defined in Section 5.1 of [RFC6401].

3.3. SIP-Resource-Priority AVP

The SIP-Resource-Priority AVP is a grouped AVP consisting of two AVPs, the SIP-Resource-Priority-Namespace and the SIP-Resource- Priority-Value AVP, which are derived from the corresponding optional header fields in [rfc4412].

```
SIP-Resource-Priority ::= < AVP Header: TBD >
                { SIP-Resource-Priority-Namespace }
                { SIP-Resource-Priority-Value }
```

3.3.1. SIP-Resource-Priority-Namespace AVP

The SIP-Resource-Priority-Namespace AVP (AVP Code TBD) is of type UTF8String. This AVP contains a string that identifies a unique ordered set of priority values as described in [rfc4412].

3.3.2 SIP-Resource-Priority-Value AVP

The SIP-Resource-Priority-Value AVP (AVP Code TBD) is of type UTF8String. This AVP contains a string (i.e., a Namespace entry) that identifies a member of a set of priority values unique to the Namespace. Examples of Namespaces and corresponding sets of priority values are found in [rfc4412].

3.4. Application-Level-Resource-Priority AVP

The Application-Level-Resource-Priority (ALRP) AVP is a grouped AVP consisting of two AVPs, the ALRP-Namespace AVP and the ALRP-Value AVP.

```
Application-Level-Resource-Priority  ::= < AVP Header: TBD >
                                         { ALRP-Namespace }
                                         { ALRP-Value }
```

A description of the semantics of the parameter values can be found in [RFC4412] and in [RFC6401]. The registry set up by [RFC4412] provided string values for both the priority namespace and the priority values associated with that namespace. [RFC6401] modifies that registry to assign numerical values to both the namespace identifiers and the priority values within them. Consequently, SIP-Resource-Priority and Application-Level-Resource-Priority AVPs convey the same priority semantics, but with differing syntax. In the former case, an alpha-numeric encoding is used, while the latter case is constrained to a numeric-only value.

3.4.1. ALRP-Namespace AVP

The ALRP-Namespace AVP (AVP Code TBD) is of type Unsigned16. This AVP contains a numerical value identifying the namespace of the application-level resource priority as described in [RFC6401].

3.4.2. ALRP-Value AVP

The ALRP-Value AVP (AVP Code TBD) is of type Unsigned8. This AVP contains the priority value within the ALRP-Namespace, as described in [RFC6401].

4. Examples of Usage

Usage of the Dual-Priority, Admission-Priority, and Application-Level-Resource-Priority AVPs can all be illustrated by the same simple network scenario, although they would not all typically be used in the same network. The scenario is as follows:

An user with special authorization is authenticated by a Network Access Server (NAS), which acts as a client to a Diameter Server supporting the user’s desired application. Once the user has authenticated, the Diameter Server provides the NAS with information on the user’s authorized QoS, including instances of the Dual-Priority, Admission-Priority, and/or Application-Level-Resource-Priority AVPs.

Local policy governs the usage of the values conveyed by these AVPs at the NAS to decide whether the flow associated with the user’s

application can be admitted. If the decision is positive, the NAS forwards the authorized QoS values as objects in RSVP signalling. In particular, the values in the Dual-Priority AVP would be carried in the Signaled Preemption Priority Policy Element defined in [RFC3181], and so on. Each subsequent node would make its own decision taking account of the authorized QoS objects including the priority-related objects, again governed by local policy. The example assumes that the user session terminates on a host or server in the same administrative domain as the NAS, to avoid complications due to the restricted applicability of [RFC3181] and [RFC6401].

Local policy might for example indicate:

- which value to take if both Admission-Priority and Application-Level-Resource-Priority are present;

- what namespace or namespaces are recognized for use in Application-Level-Resource-Priority;

- which resources are subject to pre-emption if the values in Dual-Priority are high enough to allow it.

A scenario for the use of the SIP-Resource-Priority AVP will differ slightly from the previous one, in that the initial decision point would typically be a SIP proxy receiving a session initiation request containing a Resource-Priority header field and deciding whether to admit the session to the domain. Like the NAS, the SIP proxy would serve as client to a Diameter Server during the process of user authentication, and upon successful authentication would receive back from the Diameter Server AVPs indicating authorized QoS. Among these might be the SIP-Resource-Priority AVP, the contents of which would be compared with the contents of the Resource-Priority header field. Again, local policy would determine which namespaces would be accepted and what the effect of a given priority level would be on the admission decision.

For the sake of our example, suppose now that the SIP proxy signals using RSVP to the border router that will be admitting the media flows associated with the session. (This, of course, makes a few assumptions on routing and knowledge of that routing at the proxy.) The SIP proxy can indicate authorized QoS using various objects. In particular, it can map the values from the Resource-Priority header field to the corresponding numeric values as defined by [RFC6401], and send it using the Application-Level Resource Priority Policy Element.

5. IANA Considerations

5.1. AVP Codes

IANA is requested to allocate AVP codes for the following AVPs that are defined in this document.

AVP Name	AVP Code	Section Defined	Data Type
Dual-Priority	TBD	3.1	Grouped
Preemption-Priority	TBD	3.1.1	Unsigned16
Defending-Priority	TBD	3.1.2	Unsigned16
Admission-Priority	TBD	3.2	Unsigned8
SIP-Resource-Priority	TBD	3.3	Grouped
SIP-Resource-Priority-Namespace	TBD	3.3.1	UTF8String
SIP-Resource-Priority-Value	TBD	3.3.2	UTF8String
Application-Level-Resource-Priority	TBD	3.4	Grouped
ALRP-Namespace	TBD	3.4.1	Unsigned32
ALRP-Value	TBD	3.4.2	Unsigned32

5.2. QoS Profile

IANA is requested to allocate a new value from the Authentication, Authorization, and Accounting (AAA) Parameters/QoS Profile registry defined in [RFC5624] for the QoS profile defined in this document. The name of the profile is "Resource priority parameters".

6. Security Considerations

This document describes an extension for conveying Quality of Service information, and therefore follows the same security considerations of the Diameter QoS Application [RFC5866]. The values placed in the AVPs are not changed by this draft, nor are they changed in the Diameter QoS application. We recommend the use of mechanisms to ensure integrity when exchanging information from one protocol to an associated DIAMETER AVP. Examples of these integrity mechanisms would be use of S/MIME with SIP RPH, or an INTEGRITY object within a POLICY_DATA object within the context of RSVP. The consequences of changing values in the Priority AVPs may result in an allocation of additional or less resources.

Changes in integrity protected values SHOULD NOT be ignored, and appropriate protocol specific error messages SHOULD be sent back upstream. Note that we do not use the term "MUST NOT be ignored" because local policy of an administrative domain associated with other protocols acts as the final arbiter. In addition, some protocols associated with the AVPs defined in this document may be deployed within

a single administrative domain or "walled garden", and thus possible changes in values would reflect policies of that administrative domain.

The security considerations of the Diameter protocol itself are discussed in [I-D.ietf-dime-rfc3588bis]. Use of the AVPs defined in this document MUST take into consideration the security issues and requirements of the Diameter base protocol.

The authors also recommend that readers should familiarize themselves with the security considerations of the various protocols listed in the Normative References. This is because values placed in the AVPs defined in this draft are set/changed by other protocols.

7. Acknowledgements

We would like to thank Lionel Morand, Janet Gunn, Piers O'Hanlon, Lars Eggert, Jan Engelhardt, Francois LeFaucheur, John Loughney, An Nguyen, Dave Oran, James Polk, Martin Stiemerling, and Magnus Westerlund, David Harrington, Robert Sparks, and Dan Romascanu for their review and/or comments on previous versions of the draft.

8. References

8.1. Normative References

[I-D.ietf-dime-rfc3588bis]
Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", draft-ietf-dime-rfc3588bis-26 (work in progress), January 2011.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2205] Braden, B., et. al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997

[RFC3181] Herzog, S., "Signaled Preemption Priority Policy Element", RFC 3181, October 2001.

[RFC4412] Schulzrinne, H. and J. Polk, "Communications Resource Priority for the Session Initiation Protocol (SIP)", RFC 4412, February 2006.

[RFC5624] Korhonen, J., Tschofenig, H., and E. Davies, "Quality of Service Parameters for Usage with Diameter", RFC 5624,

Aug 2009.

[RFC5866] Sun, D., et. al., "Diameter Quality-of-Service Application", RFC 5866, May 2010.

[RFC6401] Faucheur, F., Polk, J., and K. Carlberg, "Resource ReSerVation Protocol (RSVP) Extensions for Emergency Services", RFC 6401, Oct 2011.

8.2. Informative References

[3GPPa] "TS 29.214: Policy and charging control over Rx reference point", 3GPP, March, 2011

[3GPPb] "TS 29.212: Policy and charging control over Gx reference point", 3GPP, October, 2010

[3GPPc] "TS 29.229: Cx and Dx interfaces based on the Diameter protocol; Protocol details", 3GPP, September, 2010

[3GPPd] "TS 29.329: Sh interface based on the Diameter protocol; Protocol details", 3GPP, September, 2010

[ETSI] "TS 183 017: Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); Resource and Admission Control", ETSI

Authors' Addresses

Ken Carlberg (editor)
G11
1601 Clarendon Dr
Arlington, VA 22209
United States

Email: carlberg@g11.org.uk

Tom Taylor
PT Taylor Consulting
1852 Lorraine Ave
Ottawa
Canada

Email: tom.taylor.stds@gmail.com

Internet Engineering Task Force
Internet-Draft
Updates: 6733 (if approved)
Intended status: Standards Track
Expires: April 04, 2014

T. Tsou
Huawei Technologies (USA)
R. Hao
Comcast Cable
T. Taylor, Ed.
Huawei Technologies

October 01, 2013

Realm-Based Redirection In Diameter
draft-ietf-dime-realm-based-redirect-13

Abstract

The Diameter protocol includes a capability for message redirection, controlled by an application-independent "redirect agent". In some circumstances, an operator may wish to redirect messages to an alternate domain without specifying individual hosts. This document specifies an application-specific mechanism by which a Diameter server or proxy (node) can perform such a redirection when S-NAPTR is not used for dynamic peer discovery. A node performing this new function is referred to as a "Realm-based Redirect Server".

This memo updates Sections 6.13 and 6.14 of RFC6733 with respect to the usage of the Redirect-Host-Usage and Redirect-Max-Cache-Time AVPs.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 04, 2014.

Table of Contents

1. Introduction

The Diameter base protocol [RFC6733] specifies a basic redirection service provided by redirect agent. The redirect indication returned by the redirect agent is described in Section 6.1.8 and Sections 6.12-6.14 of [RFC6733], and provides to the message sender one or more individual hosts as destination of the redirected message.

However, consider the case where an operator has offered a specific service but no longer wishes to do so. The operator has arranged for an alternative domain to provide the service. To aid in the transition to the new arrangement, the original operator maintains a redirect server to indicate to the message sender the alternative domain to which redirect the request. However, the original operator should be relieved from configuring in the redirect server a list of hosts to contact in the alternative operator's domain, and should

simply be able to provide redirect indications to the domain as a whole.

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 [RFC2119].

Within this specification, the term "realm-based redirection" is used to refer to a mode of operation where a realm rather than an individual host is returned as redirect indication.

The term "Realm-based Redirect Server" denotes the Diameter node (Diameter server or proxy) that returns the realm-based redirection. The behaviour of the Realm-based Redirect Server itself is a slight modification of the behaviour of a basic redirect agent as described in Section 6.1.8 of [RFC6733].

This document uses a number of terms consistently with their usage in [RFC6733]: "Diameter client", "Diameter node", Diameter peer", "Diameter server", "proxy", "realm" or "domain", "redirect agent", and "session" as defined in Section 1.2, and "application" as defined implicitly by Sections 1.3.4, 2.3, and 2.4.

2. Support of Realm-Based Redirection Within Applications

The DNS-based dynamic peer discovery mechanism defined in the Diameter base protocol [RFC6733] provides a simple mechanism for realm-based redirection, using the S-NAPTR DDDS application [RFC3958]. When S-NAPTR is used for peer discovery, redirection of Diameter requests from the original realm to a new realm may be performed by updating the existing NAPTR resource records for the original realm as follows: the NAPTR RR for the desired application(s) and supported application protocol(s) provided by the new realm will have an empty FLAG field and the REPLACEMENT field will contain the new realm to use for the next DNS lookup. The new realm can be arbitrary; the restriction in [RFC6733] that the NAPTR replacement field match the domain of the original query does not apply for realm-based redirect purposes.

However, the use of DNS-based dynamic peer discovery is optional for Diameter implementations. For deployments which do not make use of S-NAPTR peer discovery, support of realm-based redirection needs to be specified as part of functionality supported by a Diameter application. In this way, support of the considered Diameter application (discovered during capabilities exchange procedures as defined in Diameter base protocol [RFC6733]) indicates implicit

support of the realm-based redirection mechanism. A new application specification can incorporate the mechanism specified here by making it mandatory to implement for the application, and referencing this specification normatively.

The result of making realm-based redirection an application-specific behaviour is that it cannot be performed by a redirect agent as defined in [RFC6733], but MUST be performed instead by an application-aware Diameter node (Diameter server or proxy) (hereafter called a "Realm-based Redirect Server").

An application can specify that realm-based redirection operates only on complete sessions beginning with the initial message, or on every message within the application, even if earlier messages of the same session were not redirected. This distinction matters only when realm-based redirection is first initiated. In the former case, existing sessions will not be disrupted by the deployment of realm-based redirection. In the latter case, existing sessions will be disrupted if they are stateful.

3. Realm-Based Redirection

This section specifies an extension of the Diameter base protocol [RFC6733] to achieve realm-based redirection. The elements of this solution are:

- a new result code, DIAMETER_REALM_REDIRECT_INDICATION (3xxx TBD1);
- a new attribute-value pair (AVP), Redirect-Realm (code TBD2); and
- associated behaviour at Diameter nodes implementing this specification.

This behaviour includes the optional use of the Redirect-Host-Usage and Redirect-Max-Cache-Time AVPs. In this document, these AVPs apply to the peer discovered by a node acting on the redirect server’s response, an extension to their normal usage as described in Sections 6.13 and 6.14 of [RFC6733].

Section 3.2.2 and Section 3.2.3 describe how a proxy or client may update its routing table for the application and initial realm, as a result of selecting a peer in the new realm after realm-based redirection. Note that as a result, the proxy or client will automatically route subsequent requests for that application to the new realm (with the possible exception of requests within sessions already established with the initial realm) until the cached routing entry expires. This should be borne in mind if the rerouting is intended to be temporary.

3.1. Configuration of the Realm-based Redirect Server

A Diameter node (Diameter server or proxy) acting as Realm-based Redirect Server MUST be configured as follows to execute realm-based redirection:

- configured with an application that incorporates realm-based redirection;
- the Local Action field of the routing table described in Section 2.7 of [RFC6733] is set to LOCAL;
- an application-specific field is set to indicate that the required local action is to perform realm-based redirection;
- an associated application-specific field is configured with the identities of one or more realms to which the request should be redirected.

3.2. Behaviour of Diameter Nodes

3.2.1. Behaviour at the Realm-based Redirect Server

As mentioned in Section 2, an application can specify that realm-based redirection operates only on complete sessions beginning with the initial message (i.e., to prevent disruption of established sessions), or on every message within the application, even if earlier messages of the same session were not redirected.

If a Realm-based Redirect Server configured as described in Section 3.1 receives a request to which realm-based redirection applies, the Realm-based Redirect Server MUST reply with an answer message with the 'E' bit set, while maintaining the Hop-by-Hop Identifier in the header. The Realm-based Redirect Server MUST include the Result-Code AVP set to DIAMETER_REALM_REDIRECT_INDICATION. The Realm-based Redirect Server MUST also include the alternate realm identifier(s) with which it has been configured, each in a separate Redirect-Realm AVP instance.

The Realm-based Redirect Server MAY include a copy of the Redirect-Host-Usage AVP, which SHOULD be set to REALM_AND_APPLICATION. If this AVP is added, the Redirect-Max-Cache-Time AVP MUST also be included. Note that these AVPs apply to the peer discovered by a node acting on the Realm-based Redirect Server's response, as described in the next section. This is an extension of their normal usage as described by Sections 6.13 and 6.14 of [RFC6733].

Realm-based redirection MAY be applied even if a Destination-Host AVP is present in the request, depending on the operator-based policy.

3.2.2. Proxy Behaviour

A proxy conforming to this specification that receives an answer message with the Result-Code AVP set to DIAMETER_REALM_REDIRECT_INDICATION MUST attempt to reroute the original request to a server in a realm identified by a Redirect-Realm AVP instance in the answer message, and if it fails MUST forward the indication toward the client. To reroute the request, it MUST take the following actions:

1. Select a specific realm from amongst those identified in instances of the Redirect-Realm AVP in the answer message.

2. If successful, locate and establish a route to a peer in the realm given by the Redirect-Realm AVP, using normal discovery procedures as described in Section 5.2 of [RFC6733].

3. If again successful:

   a. update its cache of routing entries for the realm and application to which the original request was directed, taking into account the Redirect-Host-Usage and Redirect-Max-Cache-Time AVPs, if present in the answer.

   b. Remove the Destination-Host (if present) and Destination-Realm AVPs from the original request and add a new Destination-Realm AVP containing the realm selected in the initial step.

   c. Forward the modified request.

4. If either of the preceding steps 2-3 fail and additional realms have been identified in the original answer, select another instance of the Redirect-Realm AVP in that answer and repeat steps 2-3 for the realm that it identifies.

3.2.3. Client Behaviour

A client conforming to this specification MUST be prepared to receive either an answer message containing a Result-Code AVP set to DIAMETER_REALM_REDIRECT_INDICATION, or, as the result of proxy action, some other result from a realm differing from the one to which it sent the original request. In the case where it receives DIAMETER_REALM_REDIRECT_INDICATION, the client SHOULD follow the same

steps prescribed in the previous section for a proxy, in order both to update its routing table and to obtain service for the original request.

3.3. The Redirect-Realm AVP

The Redirect-Realm AVP (code TBD2) is of type DiameterIdentity. It specifies a realm to which a node receiving a redirect indication containing the result code value DIAMETER_REALM_REDIRECT_INDICATION and the Redirect-Realm AVP SHOULD route the original request.

3.4. DIAMETER_REALM_REDIRECT_INDICATION Protocol Error Code

The DIAMETER_REALM_REDIRECT_INDICATION (3xxx TBD1) Protocol error code indicates that a server has determined that the request within an application supporting realm-based redirection could not be satisfied locally and the initiator of the request SHOULD direct the request directly to a peer within a realm that has been identified in the response. When set, the Redirect-Realm AVP MUST be present.

4. Security Considerations

The general recommendations given in the section 13 of the Diameter base protocol [RFC6733] apply. Specific security recommendations related to the realm-based redirection defined in this document are described below.

Realm-based redirection implies a change of the business relationships between organizations. Before redirecting a request towards a realm different from the initial realm, the client or proxy MUST ensure that the authorization checks have been performed at each connection along the path toward the realm identified in the realm-based redirect indication. Details on Diameter authorization path set-up are given in section 2.9 of [RFC6733]. Section 13 of [RFC6733] provides recommendations on how to authenticate and secure each peer-to-peer connection (using on TLS, DTLS or IPsec) along the way, thus permitting the necessary hop-by-hop authorization checks.

Although it is assumed that the administrative domains are secure, a compromised Diameter node acting as Realm-Based Redirect Server would be able to redirect a large number of Diameter requests towards a victim domain which would then be flooded with undesired Diameter requests. Such an attack is nevertheless discouraged by the use of secure Diameter peer-to-peer connections and authorization checks, since these would enable a potential victim domain to discover from where an attack is coming. That in itself, however, does not prevent such a DoS attack.

Because realm-based redirection defined in this document implies that the Destination-Realm AVP in a client-initiated request can be changed by a Diameter proxy in the path between the client and the server, any cryptographic algorithm that would use the Destination-Realm AVP as input to the calculation performed by the client and the server would be broken by this form of redirection. Application specifications that would rely on such cryptographic algorithm SHOULD NOT incorporate this realm-based redirection.

5. IANA Considerations

This specification adds a new AVP code [TBD2] Redirect-Realm in the AVP Code registry under Authentication, Authorization, and Accounting (AAA) Parameters.

This specification allocates a new Result-Code value DIAMETER_REALM_REDIRECT_INDICATION (3xxx TBD1) in the Result-Code AVP Values (code 268) - Protocol Errors registry under Authentication, Authorization, and Accounting (AAA) Parameters.

6. Acknowledgements

Glen Zorn, Sebastien Decugis, Wolfgang Steigerwald, Mark Jones, Victor Fajardo, Jouni Korhonen, Avi Lior, and Lionel Morand contributed comments that helped to shape this document. As shepherd, Lionel contributed a second set of comments that added polish to the document before it was submitted to the IESG. Benoit Claise picked up additional points which were quickly resolved with Lionel’s help. During IETF Last Call Review, Enrico Marocco picked up some important editorial corrections. Stefan Winter contributed text on the use of S-NAPTR as an alternative method of realm-based redirection already specified in [RFC6733]. Derek Atkins performed a review on behalf of the Security Directorate. Lionel noted one more correction.

Finally, this document benefited from comments and discussion raised by IESG members Stewart Bryant, Stephen Farrell, Barry Leiba, Pete Resnick, Jaari Arkko, and Sean Turner during IESG review.

The authors are very grateful to Lionel Morand for his active role as document shepherd. At each stage, he worked to summarize and resolve comments, making the editor’s role easy. Thank you.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

7.2. Informative References

[RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, January 2005.

Authors' Addresses

Tina Tsou
Huawei Technologies (USA)
2330 Central Expressway
Santa Clara, CA 95050
USA

Phone: +1 408 330 4424
Email: Tina.Tsou.Zouting@huawei.com
URI: http://tinatsou.weebly.com/contact.html

Ruibing Hao
Comcast Cable
One Comcast Center
Philadelphia, PA 19103
USA

Phone: +1 215 286 3991(O)
Email: Ruibing_Hao@cable.comcast.com

Tom Taylor (editor)
Huawei Technologies
Ottawa
Canada

Email: tom.taylor.stds@gmail.com

Network Working Group
Internet-Draft
Obsoletes: 3588 5719 (if approved)
Intended status: Standards Track
Expires: December 25, 2012

V. Fajardo, Ed.
Telcordia Technologies
J. Arkko
Ericsson Research
J. Loughney
Nokia Research Center
G. Zorn, Ed.
Network Zen
June 23, 2012

Diameter Base Protocol
draft-ietf-dime-rfc3588bis-34.txt

Abstract

The Diameter base protocol is intended to provide an Authentication, Authorization and Accounting (AAA) framework for applications such as network access or IP mobility in both local and roaming situations. This document specifies the message format, transport, error reporting, accounting and security services used by all Diameter applications. The Diameter base protocol as defined in this document obsoletes RFC 3588 and RFC 5719 and must be supported by all new Diameter implementations.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 25, 2012.

Copyright Notice

Table of Contents

1. Introduction

Authentication, Authorization and Accounting (AAA) protocols such as TACACS [RFC1492] and RADIUS [RFC2865] were initially deployed to provide dial-up PPP [RFC1661] and terminal server access. Over time, AAA support was needed on many new access technologies, the scale and complexity of AAA networks grew, and AAA was also used on new applications (such as voice over IP). This lead to new demands on AAA protocols.

Network access requirements for AAA protocols are summarized in [RFC2989]. These include:

Failover

[RFC2865] does not define failover mechanisms, and as a result, failover behavior differs between implementations. In order to provide well-defined failover behavior, Diameter supports application-layer acknowledgements, and defines failover algorithms and the associated state machine. This is described in Section 5.5 [RFC3539].

Transmission-level security

[RFC2865] defines an application-layer authentication and integrity scheme that is required only for use with Response packets. While [RFC2869] defines an additional authentication and integrity mechanism, use is only required during Extensible Authentication Protocol (EAP) [RFC3748] sessions. While attribute-hiding is supported, [RFC2865] does not provide support for per-packet confidentiality. In accounting, [RFC2866] assumes that replay protection is provided by the backend billing server, rather than within the protocol itself.

While [RFC3162] defines the use of IPsec with RADIUS, support for IPsec is not required. In order to provide universal support for transmission-level security, and enable both intra- and inter-domain AAA deployments, Diameter provides support for TLS/TCP and DTLS/SCTP. Security is discussed in Section 13.

Reliable transport

RADIUS runs over UDP, and does not define retransmission behavior; as a result, reliability varies between implementations. As described in [RFC2975], this is a major issue in accounting, where

packet loss may translate directly into revenue loss. In order to provide well defined transport behavior, Diameter runs over reliable transport mechanisms (TCP, SCTP) as defined in [RFC3539].

Agent support

[RFC2865] does not provide for explicit support for agents, including Proxies, Redirects and Relays. Since the expected behavior is not defined, it varies between implementations. Diameter defines agent behavior explicitly; this is described in Section 2.8.

Server-initiated messages

While RADIUS server-initiated messages are defined [RFC5176], support is optional. This makes it difficult to implement features such as unsolicited disconnect or re-authentication/ re-authorization on demand across a heterogeneous deployment. To address this issue, support for server-initiated messages is mandatory in Diameter.

Transition support

While Diameter does not share a common protocol data unit (PDU) with RADIUS, considerable effort has been expended in enabling backward compatibility with RADIUS, so that the two protocols may be deployed in the same network. Initially, it is expected that Diameter will be deployed within new network devices, as well as within gateways enabling communication between legacy RADIUS devices and Diameter agents. This capability enables Diameter support to be added to legacy networks, by addition of a gateway or server speaking both RADIUS and Diameter.

In addition to addressing the above requirements, Diameter also provides support for the following:

Capability negotiation

RADIUS does not support error messages, capability negotiation, or a mandatory/non-mandatory flag for attributes. Since RADIUS clients and servers are not aware of each other’s capabilities, they may not be able to successfully negotiate a mutually acceptable service, or in some cases, even be aware of what service has been implemented. Diameter includes support for error

handling (Section 7), capability negotiation (Section 5.3), and mandatory/non-mandatory Attribute-Value Pairs (AVPs) (Section 4.1).

Peer discovery and configuration

RADIUS implementations typically require that the name or address of servers or clients be manually configured, along with the corresponding shared secrets. This results in a large administrative burden, and creates the temptation to reuse the RADIUS shared secret, which can result in major security vulnerabilities if the Request Authenticator is not globally and temporally unique as required in [RFC2865]. Through DNS, Diameter enables dynamic discovery of peers (see Section 5.2). Derivation of dynamic session keys is enabled via transmission-level security.

Over time, the capabilities of Network Access Server (NAS) devices have increased substantially. As a result, while Diameter is a considerably more sophisticated protocol than RADIUS, it remains feasible to implement it within embedded devices.

1.1. Diameter Protocol

The Diameter base protocol provides the following facilities:

- Ability to exchange messages and deliver AVPs
- Capabilities negotiation
- Error notification
- Extensibility, through addition of new applications, commands and AVPs (required in [RFC2989]).
- Basic services necessary for applications, such as handling of user sessions or accounting

All data delivered by the protocol is in the form of AVPs. Some of these AVP values are used by the Diameter protocol itself, while others deliver data associated with particular applications that employ Diameter. AVPs may be arbitrarily added to Diameter messages, the only restriction being that the Command Code Format specification Section 3.2 is satisfied. AVPs are used by the base Diameter protocol to support the following required features:

- Transporting of user authentication information, for the purposes of enabling the Diameter server to authenticate the user.

- Transporting of service-specific authorization information, between client and servers, allowing the peers to decide whether a user's access request should be granted.

- Exchanging resource usage information, which may be used for accounting purposes, capacity planning, etc.

- Routing, relaying, proxying and redirecting of Diameter messages through a server hierarchy.

The Diameter base protocol satisfies the minimum requirements for an AAA protocol, as specified by [RFC2989]. The base protocol may be used by itself for accounting purposes only, or it may be used with a Diameter application, such as Mobile IPv4 [RFC4004], or network access [RFC4005]. It is also possible for the base protocol to be extended for use in new applications, via the addition of new commands or AVPs. The initial focus of Diameter was network access and accounting applications. A truly generic AAA protocol used by many applications might provide functionality not provided by Diameter. Therefore, it is imperative that the designers of new applications understand their requirements before using Diameter. See Section 1.3.4 for more information on Diameter applications.

Any node can initiate a request. In that sense, Diameter is a peer-to-peer protocol. In this document, a Diameter Client is a device at the edge of the network that performs access control, such as a Network Access Server (NAS) or a Foreign Agent (FA). A Diameter client generates Diameter messages to request authentication, authorization, and accounting services for the user. A Diameter agent is a node that does not provide local user authentication or authorization services; agents include proxies, redirects and relay agents. A Diameter server performs authentication and/or authorization of the user. A Diameter node may act as an agent for certain requests while acting as a server for others.

The Diameter protocol also supports server-initiated messages, such as a request to abort service to a particular user.

1.1.1. Description of the Document Set

The Diameter specification consists of an updated version of the base protocol specification (this document) and the Transport Profile [RFC3539]. This document obsoletes both RFC 3588 and RFC 5719. A summary of the base protocol updates included in this document can be found in Section 1.1.3.

This document defines the base protocol specification for AAA, which includes support for accounting. There are also a myriad of applications documents describing applications that use this base specification for Authentication, Authorization and Accounting. These application documents specify how to use the Diameter protocol within the context of their application.

The Transport Profile document [RFC3539] discusses transport layer issues that arise with AAA protocols and recommendations on how to overcome these issues. This document also defines the Diameter failover algorithm and state machine.

Clarifications on the Routing of Diameter Request based on Username and the Realm [RFC5729] defines specific behavior on how to route requests based on the content of the User-Name AVP (Attribute Value Pair).

1.1.2. Conventions Used in This Document

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 [RFC2119].

1.1.3. Changes from RFC3588

This document obsoletes RFC 3588 but is fully backward compatible with that document. The changes introduced in this document focus on fixing issues that have surfaced during implementation of Diameter [RFC3588]. An overview of some the major changes are given below.

- Deprecated the use of Inband-Security AVP for negotiating transport layer security. It has been generally considered that bootstrapping of TLS via Inband-Security AVP creates certain security risk because it does not completely protect the information carried in the CER (Capabilities Exchange Request)/CEA (Capabilities Exchange Answer). This version of Diameter adopted a common approach of defining a well-known secured port that peers should use when communicating via TLS/TCP and DTLS/SCTP. This new approach augments the existing Inband-Security negotiation but does not completely replace it. The old method is kept for backwards compatibility reasons.

- Deprecated the exchange of CER/CEA messages in the open state. This feature was implied in the peer state machine table of [RFC3588] but it was not clearly defined anywhere else in that document. As work on this document progressed, it became clear that the multiplicity of meaning and use of Application Id AVPs in the CER/CEA messages (and the messages themselves) is seen as an

abuse of the Diameter extensibility rules and thus required simplification. It is assumed that the capabilities exchange in the open state will be re-introduced in a separate specification which clearly defines new commands for this feature.

- Simplified Security Requirements. The use of a secured transport for exchanging Diameter messages remains mandatory. However, TLS/TCP and DTLS/SCTP has become the primary method of securing Diameter and IPsec is a secondary alternative. See Section 13 for details. The support for the End-to-End security framework (E2E-Sequence AVP and 'P'-bit in the AVP header) has also been deprecated.

- Diameter Extensibility Changes. This includes fixes to the Diameter extensibility description (Section 1.3 and others) to better aid Diameter application designers; in addition, the new specification relaxes the policy with respect to the allocation of command codes for vendor-specific uses.

- Application Id Usage. Clarify the proper use of Application Id information which can be found in multiple places within a Diameter message. This includes correlating Application Ids found in the message headers and AVPs. These changes also clearly specify the proper Application Id value to use for specific base protocol messages (ASR/ASA, STR/STA) as well as clarifying the content and use of Vendor-Specific-Application-Id.

- Routing Fixes. This document more clearly specifies what information (AVPs and Application Id) can be used for making general routing decisions. A rule for the prioritization of redirect routing criteria when multiple route entries are found via redirects has also been added (see Section 6.13.

- Simplification of Diameter Peer Discovery. The Diameter discovery process now supports only widely used discovery schemes; the rest have been deprecated (see Section 5.2 for details).

There are many other many miscellaneous fixes that have been introduced in this document that may not be considered significant but they are important nonetheless. Examples are removal of obsolete types, fixes to the state machine, clarification of the election process, message validation, fixes to Failed-AVP and Result-Code AVP values, etc. All of the errata previously filed against RFC 3588 have been fixed. A comprehensive list of changes is not shown here for practical reasons.

1.2. Terminology

AAA

Authentication, Authorization and Accounting.

ABNF

Augmented Backus-Naur Form [RFC5234]. A metalanguage with its own formal syntax and rules. It is based on the Backus-Naur Form and is used to define message exchanges in a bi-directional communications protocol.

Accounting

The act of collecting information on resource usage for the purpose of capacity planning, auditing, billing or cost allocation.

Accounting Record

An accounting record represents a summary of the resource consumption of a user over the entire session. Accounting servers creating the accounting record may do so by processing interim accounting events or accounting events from several devices serving the same user.

Authentication

The act of verifying the identity of an entity (subject).

Authorization

The act of determining whether a requesting entity (subject) will be allowed access to a resource (object).

Attribute-Value Pair (AVP)

The Diameter protocol consists of a header followed by one or more Attribute-Value-Pairs (AVPs). An AVP includes a header and is used to encapsulate protocol-specific data (e.g., routing information) as well as authentication, authorization or

accounting information.

Command Code Format (CCF)

A modified form of ABNF used to define Diameter commands (see Section 3.2).

Diameter Agent

A Diameter Agent is a Diameter Node that provides either relay, proxy, redirect or translation services.

Diameter Client

A Diameter Client is a Diameter Node that supports Diameter client applications as well as the base protocol. Diameter Clients are often implemented in devices situated at the edge of a network and provide access control services for that network. Typical examples of Diameter Clients include the Network Access Server (NAS) and the Mobile IP Foreign Agent (FA).

Diameter Node

A Diameter Node is a host process that implements the Diameter protocol, and acts either as a Client, Agent or Server.

Diameter Peer

Two Diameter Nodes sharing a direct TCP or SCTP transport connection are called Diameter Peers.

Diameter Server

A Diameter Server is a Diameter Node that handles authentication, authorization and accounting requests for a particular realm. By its very nature, a Diameter Server must support Diameter server applications in addition to the base protocol.

Downstream

Downstream is used to identify the direction of a particular Diameter message from the Home Server towards the Diameter Client.

Home Realm

A Home Realm is the administrative domain with which the user maintains an account relationship.

Home Server

A Diameter Server which serves the Home Realm.

Interim accounting

An interim accounting message provides a snapshot of usage during a user's session. It is typically implemented in order to provide for partial accounting of a user's session in the case a device reboot or other network problem prevents the delivery of a session summary message or session record.

Local Realm

A local realm is the administrative domain providing services to a user. An administrative domain may act as a local realm for certain users, while being a home realm for others.

Multi-session

A multi-session represents a logical linking of several sessions. Multi-sessions are tracked by using the Acct-Multi-Session-Id. An example of a multi-session would be a Multi-link PPP bundle. Each leg of the bundle would be a session while the entire bundle would be a multi-session.

Network Access Identifier

The Network Access Identifier, or NAI [RFC4282], is used in the Diameter protocol to extract a user's identity and realm. The identity is used to identify the user during authentication and/or authorization, while the realm is used for message routing

purposes.

Proxy Agent or Proxy

In addition to forwarding requests and responses, proxies make policy decisions relating to resource usage and provisioning. This is typically accomplished by tracking the state of NAS devices. While proxies typically do not respond to client Requests prior to receiving a Response from the server, they may originate Reject messages in cases where policies are violated. As a result, proxies need to understand the semantics of the messages passing through them, and may not support all Diameter applications.

Realm

The string in the NAI that immediately follows the '@' character. NAI realm names are required to be unique, and are piggybacked on the administration of the DNS namespace. Diameter makes use of the realm, also loosely referred to as domain, to determine whether messages can be satisfied locally, or whether they must be routed or redirected. In RADIUS, realm names are not necessarily piggybacked on the DNS namespace but may be independent of it.

Real-time Accounting

Real-time accounting involves the processing of information on resource usage within a defined time window. Time constraints are typically imposed in order to limit financial risk. The Diameter Credit Control Application [RFC4006] is an example of an application that defines real-time accounting functionality.

Relay Agent or Relay

Relays forward requests and responses based on routing-related AVPs and routing table entries. Since relays do not make policy decisions, they do not examine or alter non-routing AVPs. As a result, relays never originate messages, do not need to understand the semantics of messages or non-routing AVPs, and are capable of handling any Diameter application or message type. Since relays make decisions based on information in routing AVPs and realm forwarding tables they do not keep state on NAS resource usage or sessions in progress.

Redirect Agent

Rather than forwarding requests and responses between clients and servers, redirect agents refer clients to servers and allow them to communicate directly. Since redirect agents do not sit in the forwarding path, they do not alter any AVPs transiting between client and server. Redirect agents do not originate messages and are capable of handling any message type, although they may be configured only to redirect messages of certain types, while acting as relay or proxy agents for other types. As with relay agents, redirect agents do not keep state with respect to sessions or NAS resources.

Session

A session is a related progression of events devoted to a particular activity. Diameter application documents provide guidelines as to when a session begins and ends. All Diameter packets with the same Session-Id are considered to be part of the same session.

Stateful Agent

A stateful agent is one that maintains session state information, by keeping track of all authorized active sessions. Each authorized session is bound to a particular service, and its state is considered active either until it is notified otherwise, or by expiration.

Sub-session

A sub-session represents a distinct service (e.g., QoS or data characteristics) provided to a given session. These services may happen concurrently (e.g., simultaneous voice and data transfer during the same session) or serially. These changes in sessions are tracked with the Accounting-Sub-Session-Id.

Transaction state

The Diameter protocol requires that agents maintain transaction state, which is used for failover purposes. Transaction state implies that upon forwarding a request, the Hop-by-Hop identifier is saved; the field is replaced with a locally unique identifier, which is restored to its original value when the corresponding

answer is received. The request's state is released upon receipt of the answer. A stateless agent is one that only maintains transaction state.

Translation Agent

A translation agent is a stateful Diameter node that performs protocol translation between Diameter and another AAA protocol, such as RADIUS.

Upstream

Upstream is used to identify the direction of a particular Diameter message from the Diameter Client towards the Home Server.

User

The entity or device requesting or using some resource, in support of which a Diameter client has generated a request.

1.3. Approach to Extensibility

The Diameter protocol is designed to be extensible, using several mechanisms, including:

- Defining new AVP values
- Creating new AVPs
- Creating new commands
- Creating new applications

From the point of view of extensibility Diameter authentication, authorization and accounting applications are treated in the same way.

Note: Protocol designers should try to re-use existing functionality, namely AVP values, AVPs, commands, and Diameter applications. Reuse simplifies standardization and implementation. To avoid potential interoperability issues it is important to ensure that the semantics of the re-used features are well understood. Given that Diameter can also carry RADIUS attributes as Diameter AVPs, such re-use considerations apply also to existing RADIUS attributes that may be

useful in a Diameter application.

1.3.1. Defining New AVP Values

In order to allocate a new AVP value for AVPs defined in the Diameter Base protocol, the IETF needs to approve a new RFC that describes the AVP value. IANA considerations for these AVP values are discussed in Section 11.3.

The allocation of AVP values for other AVPs is guided by the IANA considerations of the document that defines those AVPs. Typically, allocation of new values for an AVP defined in an IETF RFC would require IETF Review [RFC5226], whereas values for vendor-specific AVPs can be allocated by the vendor.

1.3.2. Creating New AVPs

A new AVP being defined MUST use one of the data types listed in Section 4.2 or Section 4.3. If an appropriate derived data type is already defined, it SHOULD be used instead of a base data type to encourage reusability and good design practice.

In the event that a logical grouping of AVPs is necessary, and multiple "groups" are possible in a given command, it is recommended that a Grouped AVP be used (see Section 4.4).

The creation of new AVPs can happen in various ways. The recommended approach is to define a new general-purpose AVP in a standards track RFC approved by the IETF. However, as described in Section 11.1.1 there are also other mechanisms.

1.3.3. Creating New Commands

A new Command Code MUST be allocated when required AVPs (those indicated as {AVP} in the CCF definition) are added to, deleted from or redefined in (for example, by changing a required AVP into an optional one) an existing command.

Furthermore, if the transport characteristics of a command are changed (for example, with respect to the number of round trips required) a new Command Code MUST be registered.

A change to the CCF of a command, such as described above, MUST result in the definition of a new Command Code. This subsequently leads to the need to define a new Diameter Application for any application that will use that new Command.

The IANA considerations for command codes are discussed in

Section 3.1.

1.3.4. Creating New Diameter Applications

Every Diameter application specification MUST have an IANA assigned Application Id (see Section 2.4). The managed Application Id space is flat and there is no relationship between different Diameter applications with respect to their Application Ids. As such, there is no versioning support provided by these application Ids itself; every Diameter application is a standalone application. If the application has a relationship with other Diameter applications, such a relationship is not known to Diameter.

Before describing the rules for creating new Diameter applications it is important to discuss the semantics of the AVP occurrences as stated in the CCF and the M-bit flag (Section 4.1) for an AVP. There is no relationship imposed between the two; they are set independently.

- The CCF indicates what AVPs are placed into a Diameter Command by the sender of that Command. Often, since there are multiple modes of protocol interactions many of the AVPs are indicated as optional.

- The M-bit allows the sender to indicate to the receiver whether or not understanding the semantics of an AVP and its content is mandatory. If the M-bit is set by the sender and the receiver does not understand the AVP or the values carried within that AVP then a failure is generated (see Section 7).

It is the decision of the protocol designer when to develop a new Diameter application rather than extending Diameter in other ways. However, a new Diameter application MUST be created when one or more of the following criteria are met:

M-bit Setting

An AVP with the M-bit in the MUST column of the AVP flag table is added to an existing Command/Application.

An AVP with the M-bit in the MAY column of the AVP flag table is added to an existing Command/Application.

Note: The M-bit setting for a given AVP is relevant to an Application and each command within that application which includes the AVP. That is, if an AVP appears in two commands for application Foo and the M-bit settings are different in each

command, then there should be two AVP flag tables describing when to set the M-bit.

Commands

A new command is used within the existing application either because an additional command is added, an existing command has been modified so that a new Command Code had to be registered, or a command has been deleted.

AVP Flag bits

An existing application changes the meaning/semantics of their AVP Flags or adds new flag bits then a new Diameter application MUST be created.

If the CCF definition of a command allows it, an implementation may add arbitrary optional AVPs with the M-bit cleared (including vendor-specific AVPs) to that command without needing to define a new application. Please refer to Section 11.1.1 for details.

2. Protocol Overview

The base Diameter protocol concerns itself with establishing connections to peers, capabilities negotiation, how messages are sent and routed through peers, and how the connections are eventually torn down. The base protocol also defines certain rules that apply to all message exchanges between Diameter nodes.

Communication between Diameter peers begins with one peer sending a message to another Diameter peer. The set of AVPs included in the message is determined by a particular Diameter application. One AVP that is included to reference a user's session is the Session-Id.

The initial request for authentication and/or authorization of a user would include the Session-Id AVP. The Session-Id is then used in all subsequent messages to identify the user's session (see Section 8 for more information). The communicating party may accept the request, or reject it by returning an answer message with the Result-Code AVP set to indicate an error occurred. The specific behavior of the Diameter server or client receiving a request depends on the Diameter application employed.

Session state (associated with a Session-Id) MUST be freed upon receipt of the Session-Termination-Request, Session-Termination-Answer, expiration of authorized service time in the Session-Timeout AVP, and according to rules established in a particular Diameter

application.

The base Diameter protocol may be used by itself for accounting applications. For authentication and authorization, it is always extended for a particular application.

Diameter Clients MUST support the base protocol, which includes accounting. In addition, they MUST fully support each Diameter application that is needed to implement the client's service, e.g., NASREQ and/or Mobile IPv4. A Diameter Client MUST be referred to as "Diameter X Client" where X is the application which it supports, and not a "Diameter Client".

Diameter Servers MUST support the base protocol, which includes accounting. In addition, they MUST fully support each Diameter application that is needed to implement the intended service, e.g., NASREQ and/or Mobile IPv4. A Diameter Server MUST be referred to as "Diameter X Server" where X is the application which it supports, and not a "Diameter Server".

Diameter Relays and redirect agents are transparent to the Diameter applications but they MUST support the Diameter base protocol, which includes accounting, and all Diameter applications.

Diameter proxies MUST support the base protocol, which includes accounting. In addition, they MUST fully support each Diameter application that is needed to implement proxied services, e.g., NASREQ and/or Mobile IPv4. A Diameter proxy MUST be referred to as "Diameter X Proxy" where X is the application which it supports, and not a "Diameter Proxy".

2.1. Transport

The Diameter Transport profile is defined in [RFC3539].

The base Diameter protocol is run on port 3868 for both TCP [RFC793] and SCTP [RFC4960]. For TLS [RFC5246] and DTLS [RFC6347], a Diameter node that initiate a connection prior to any message exchanges MUST run on port <TBD>. It is assumed that TLS is run on top of TCP when it is used and DTLS is run on top of SCTP when it is used.

If the Diameter peer does not support receiving TLS/TCP and DTLS/SCTP connections on port <TBD> (i.e., the peer complies only with [RFC3588]), then the initiator MAY revert to using TCP or SCTP on port 3868. Note that this scheme is kept only for the purpose of backward compatibility and that there are inherent security vulnerabilities when the initial CER/CEA messages are sent unprotected (see Section 5.6).

Diameter clients MUST support either TCP or SCTP; agents and servers SHOULD support both.

A Diameter node MAY initiate connections from a source port other than the one that it declares it accepts incoming connections on, and MUST always be prepared to receive connections on port 3868 for TCP or SCTP and port <TBD> for TLS/TCP and DTLS/SCTP connections. When DNS-based peer discovery (Section 5.2) is used, the port numbers received from SRV records take precedence over the default ports (3868 and <TBD>).

A given Diameter instance of the peer state machine MUST NOT use more than one transport connection to communicate with a given peer, unless multiple instances exist on the peer in which case a separate connection per process is allowed.

When no transport connection exists with a peer, an attempt to connect SHOULD be periodically made. This behavior is handled via the Tc timer (see Section 12 for details), whose recommended value is 30 seconds. There are certain exceptions to this rule, such as when a peer has terminated the transport connection stating that it does not wish to communicate.

When connecting to a peer and either zero or more transports are specified, TLS SHOULD be tried first, followed by DTLS, then by TCP and finally by SCTP. See Section 5.2 for more information on peer discovery.

Diameter implementations SHOULD be able to interpret ICMP protocol port unreachable messages as explicit indications that the server is not reachable, subject to security policy on trusting such messages. Further guidance regarding the treatment of ICMP errors can be found in [RFC5927] and [RFC5461]. Diameter implementations SHOULD also be able to interpret a reset from the transport and timed-out connection attempts. If Diameter receives data from the lower layer that cannot be parsed or identified as a Diameter error made by the peer, the stream is compromised and cannot be recovered. The transport connection MUST be closed using a RESET call (send a TCP RST bit) or an SCTP ABORT message (graceful closure is compromised).

2.1.1. SCTP Guidelines

Diameter messages SHOULD be mapped into SCTP streams in a way that avoids head-of-the-line (HOL) blocking. Among different ways of performing the mapping that fulfill this requirement it is RECOMMENDED that a Diameter node sends every Diameter message (request or response) over the stream zero with the unordered flag set. However, Diameter nodes MAY select and implement other design

alternatives for avoiding HOL blocking such as using multiple streams with the unordered flag cleared (as originally instructed in RFC3588). On the receiving side, a Diameter entity MUST be ready to receive Diameter messages over any stream and it is free to return responses over a different stream. This way, both sides manage the available streams in the sending direction, independently of the streams chosen by the other side to send a particular Diameter message. These messages can be out-of-order and belong to different Diameter sessions.

Out-of-order delivery has special concerns during a connection establishment and termination. When a connection is established, the responder side sends a CEA message and moves to R-Open state as specified in Section 5.6. If an application message is sent shortly after the CEA and delivered out-of-order, the initiator side, still in Wait-I-CEA state, will discard the application message and close the connection. In order to avoid this race condition, the receiver side SHOULD NOT use out-of-order delivery methods until the first message has been received from the initiator, proving that it has moved to I-Open state. To trigger such message, the receiver side could send a DWR immediatly after sending CEA. Upon reception of the corresponding DWA, the receiver side should start using out-of-order delivery methods to counter the HOL blocking.

Another race condition may occur when DPR and DPA messages are used. Both DPR and DPA are small in size, thus they may be delivered faster to the peer than application messages when out-of-order delivery mechanism is used. Therefore, it is possible that a DPR/DPA exchange completes while application messages are still in transit, resulting to a loss of these messages. An implementation could mitigate this race condition, for example, using timers and wait for a short period of time for pending application level messages to arrive before proceeding to disconnect the transport connection. Eventually, lost messages are handled by the retransmission mechanism described in Section 5.5.4.

A Diameter agent SHOULD use dedicated payload protocol identifiers (PPID) for clear text and encrypted SCTP DATA chunks instead of only using the unspecified payload protocol identifier (value 0). For this purpose two PPID values are allocated. The PPID value <TBD2> is for Diameter messages in clear text SCTP DATA chunks and the PPID value <TBD3> is for Diameter messages in protected DTLS/SCTP DATA chunks.

2.2. Securing Diameter Messages

Connections between Diameter peers SHOULD be protected by TLS/TCP and DTLS/SCTP. All Diameter base protocol implementations MUST support

the use of TLS/TCP and DTLS/SCTP. If desired, alternative security mechanisms that are independent of Diameter, such as IPsec [RFC4301], can be deployed to secure connections between peers. The Diameter protocol MUST NOT be used without one of TLS, DTLS or IPsec.

2.3. Diameter Application Compliance

Application Ids are advertised during the capabilities exchange phase (see Section 5.3). Advertising support of an application implies that the sender supports the functionality specified in the respective Diameter application specification.

Implementations MAY add arbitrary optional AVPs with the M-bit cleared (including vendor-specific AVPs) to a command defined in an application, but only if the command's CCF syntax specification allows for it. Please refer to Section 11.1.1 for details.

2.4. Application Identifiers

Each Diameter application MUST have an IANA assigned Application Id. The base protocol does not require an Application Id since its support is mandatory. During the capabilities exchange, Diameter nodes inform their peers of locally supported applications. Furthermore, all Diameter messages contain an Application Id, which is used in the message forwarding process.

The following Application Id values are defined:

Diameter Common Messages	0
Diameter Base Accounting	3
Relay	0xffffffff

Relay and redirect agents MUST advertise the Relay Application Identifier, while all other Diameter nodes MUST advertise locally supported applications. The receiver of a Capabilities Exchange message advertising Relay service MUST assume that the sender supports all current and future applications.

Diameter relay and proxy agents are responsible for finding an upstream server that supports the application of a particular message. If none can be found, an error message is returned with the Result-Code AVP set to DIAMETER_UNABLE_TO_DELIVER.

2.5. Connections vs. Sessions

This section attempts to provide the reader with an understanding of the difference between connection and session, which are terms used extensively throughout this document.

A connection refers to a transport level connection between two peers that is used to send and receive Diameter messages. A session is a logical concept at the application layer existing between the Diameter client and the Diameter server; it is identified via the Session-Id AVP.

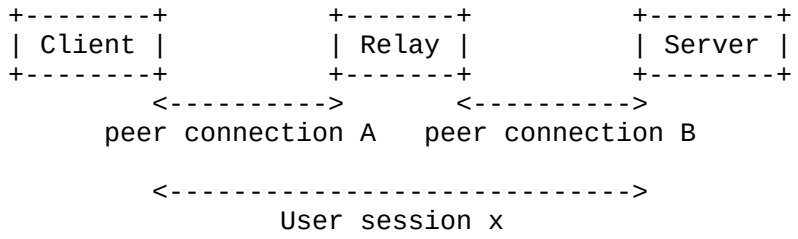


Figure 1: Diameter connections and sessions

In the example provided in Figure 1, peer connection A is established between the Client and the Relay. Peer connection B is established between the Relay and the Server. User session X spans from the Client via the Relay to the Server. Each "user" of a service causes an auth request to be sent, with a unique session identifier. Once accepted by the server, both the client and the server are aware of the session.

It is important to note that there is no relationship between a connection and a session, and that Diameter messages for multiple sessions are all multiplexed through a single connection. Also note that Diameter messages pertaining to the session, both application specific and those that are defined in this document such as ASR/ASA, RAR/RAA and STR/STA MUST carry the Application Id of the application. Diameter messages pertaining to peer connection establishment and maintenance such as CER/CEA, DWR/DWA and DPR/DPA MUST carry an Application Id of zero (0).

2.6. Peer Table

The Diameter Peer Table is used in message forwarding, and referenced by the Routing Table. A Peer Table entry contains the following fields:

Host identity

Following the conventions described for the DiameterIdentity derived AVP data format in Section 4.3.1, this field contains the contents of the Origin-Host (Section 6.3) AVP found in the CER or CEA message.

StatusT

This is the state of the peer entry, and MUST match one of the values listed in Section 5.6.

Static or Dynamic

Specifies whether a peer entry was statically configured or dynamically discovered.

Expiration time

Specifies the time at which dynamically discovered peer table entries are to be either refreshed, or expired. If public key certificates are used for Diameter security (e.g., with TLS), this value MUST NOT be greater than the expiry times in the relevant certificates.

TLS/TCP and DTLS/SCTP Enabled

Specifies whether TLS/TCP and DTLS/SCTP is to be used when communicating with the peer.

Additional security information, when needed (e.g., keys, certificates).

2.7. Routing Table

All Realm-Based routing lookups are performed against what is commonly known as the Routing Table (see Section 12). Each Routing Table entry contains the following fields:

Realm Name

This is the field that MUST be used as a primary key in the routing table lookups. Note that some implementations perform their lookups based on longest-match-from-the-right on the realm rather than requiring an exact match.

Application Identifier

An application is identified by an Application Id. A route entry can have a different destination based on the Application Id in

the message header. This field MUST be used as a secondary key field in routing table lookups.

Local Action

The Local Action field is used to identify how a message should be treated. The following actions are supported:

1. LOCAL - Diameter messages that can be satisfied locally, and do not need to be routed to another Diameter entity.

2. RELAY - All Diameter messages that fall within this category MUST be routed to a next hop Diameter entity that is indicated by the identifier described below. Routing is done without modifying any non-routing AVPs. See Section 6.1.9 for relaying guidelines.

3. PROXY - All Diameter messages that fall within this category MUST be routed to a next Diameter entity that is indicated by the identifier described below. The local server MAY apply its local policies to the message by including new AVPs to the message prior to routing. See Section 6.1.9 for proxying guidelines.

4. REDIRECT - Diameter messages that fall within this category MUST have the identity of the home Diameter server(s) appended, and returned to the sender of the message. See Section 6.1.8 for redirection guidelines.

Server Identifier

The identity of one or more servers to which the message is to be routed. This identity MUST also be present in the Host Identity field of the Peer Table (Section 2.6). When the Local Action is set to RELAY or PROXY, this field contains the identity of the server(s) to which the message MUST be routed. When the Local Action field is set to REDIRECT, this field contains the identity of one or more servers to which the message MUST be redirected.

Static or Dynamic

Specifies whether a route entry was statically configured or dynamically discovered.

Expiration time

Specifies the time at which a dynamically discovered route table entry expires. If public key certificates are used for Diameter security (e.g., with TLS), this value MUST NOT be greater than the expiry time in the relevant certificates.

It is important to note that Diameter agents MUST support at least one of the LOCAL, RELAY, PROXY or REDIRECT modes of operation. Agents do not need to support all modes of operation in order to conform with the protocol specification, but MUST follow the protocol compliance guidelines in Section 2. Relay agents and proxies MUST NOT reorder AVPs.

The routing table MAY include a default entry that MUST be used for any requests not matching any of the other entries. The routing table MAY consist of only such an entry.

When a request is routed, the target server MUST have advertised the Application Id (see Section 2.4) for the given message, or have advertised itself as a relay or proxy agent. Otherwise, an error is returned with the Result-Code AVP set to DIAMETER_UNABLE_TO_DELIVER.

2.8. Role of Diameter Agents

In addition to clients and servers, the Diameter protocol introduces relay, proxy, redirect, and translation agents, each of which is defined in Section 1.2. Diameter agents are useful for several reasons:

- They can distribute administration of systems to a configurable grouping, including the maintenance of security associations.
- They can be used for concentration of requests from an number of co-located or distributed NAS equipment sets to a set of like user groups.
- They can do value-added processing to the requests or responses.
- They can be used for load balancing.
- A complex network will have multiple authentication sources, they can sort requests and forward towards the correct target.

The Diameter protocol requires that agents maintain transaction state, which is used for failover purposes. Transaction state implies that upon forwarding a request, its Hop-by-Hop identifier is saved; the field is replaced with a locally unique identifier, which

is restored to its original value when the corresponding answer is received. The request's state is released upon receipt of the answer. A stateless agent is one that only maintains transaction state.

The Proxy-Info AVP allows stateless agents to add local state to a Diameter request, with the guarantee that the same state will be present in the answer. However, the protocol's failover procedures require that agents maintain a copy of pending requests.

A stateful agent is one that maintains session state information by keeping track of all authorized active sessions. Each authorized session is bound to a particular service, and its state is considered active either until the agent is notified otherwise, or the session expires. Each authorized session has an expiration, which is communicated by Diameter servers via the Session-Timeout AVP.

Maintaining session state may be useful in certain applications, such as:

- Protocol translation (e.g., RADIUS <-> Diameter)
- Limiting resources authorized to a particular user
- Per user or transaction auditing

A Diameter agent MAY act in a stateful manner for some requests and be stateless for others. A Diameter implementation MAY act as one type of agent for some requests, and as another type of agent for others.

2.8.1. Relay Agents

Relay Agents are Diameter agents that accept requests and route messages to other Diameter nodes based on information found in the messages (e.g., Destination-Realm). This routing decision is performed using a list of supported realms, and known peers. This is known as the Routing Table, as is defined further in Section 2.7.

Relays may, for example, be used to aggregate requests from multiple Network Access Servers (NASes) within a common geographical area (POP). The use of Relays is advantageous since it eliminates the need for NASes to be configured with the necessary security information they would otherwise require to communicate with Diameter servers in other realms. Likewise, this reduces the configuration load on Diameter servers that would otherwise be necessary when NASes are added, changed or deleted.

Relays modify Diameter messages by inserting and removing routing information, but do not modify any other portion of a message. Relays SHOULD NOT maintain session state but MUST maintain transaction state.

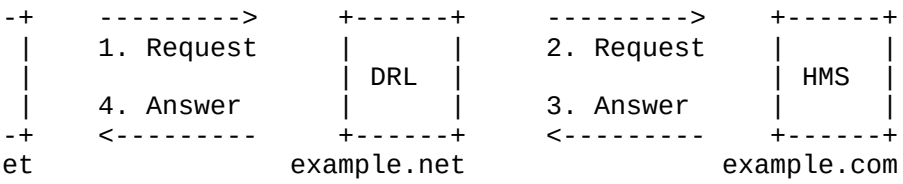


Figure 2: Relaying of Diameter messages

The example provided in Figure 2 depicts a request issued from NAS, which is an access device, for the user bob@example.com. Prior to issuing the request, NAS performs a Diameter route lookup, using "example.com" as the key, and determines that the message is to be relayed to DRL, which is a Diameter Relay. DRL performs the same route lookup as NAS, and relays the message to HMS, which is example.com's Home Diameter Server. HMS identifies that the request can be locally supported (via the realm), processes the authentication and/or authorization request, and replies with an answer, which is routed back to NAS using saved transaction state.

Since Relays do not perform any application level processing, they provide relaying services for all Diameter applications, and therefore MUST advertise the Relay Application Id.

2.8.2. Proxy Agents

Similarly to relays, proxy agents route Diameter messages using the Diameter Routing Table. However, they differ since they modify messages to implement policy enforcement. This requires that proxies maintain the state of their downstream peers (e.g., access devices) to enforce resource usage, provide admission control, and provisioning.

Proxies may, for example, be used in call control centers or access ISPs that provide outsourced connections, they can monitor the number and types of ports in use, and make allocation and admission decisions according to their configuration.

Since enforcing policies requires an understanding of the service being provided, Proxies MUST only advertise the Diameter applications they support.

2.8.3. Redirect Agents

Redirect agents are useful in scenarios where the Diameter routing configuration needs to be centralized. An example is a redirect agent that provides services to all members of a consortium, but does not wish to be burdened with relaying all messages between realms. This scenario is advantageous since it does not require that the consortium provide routing updates to its members when changes are made to a member's infrastructure.

Since redirect agents do not relay messages, and only return an answer with the information necessary for Diameter agents to communicate directly, they do not modify messages. Since redirect agents do not receive answer messages, they cannot maintain session state.

The example provided in Figure 3 depicts a request issued from the access device, NAS, for the user bob@example.com. The message is forwarded by the NAS to its relay, DRL, which does not have a routing entry in its Diameter Routing Table for example.com. DRL has a default route configured to DRD, which is a redirect agent that returns a redirect notification to DRL, as well as HMS' contact information. Upon receipt of the redirect notification, DRL establishes a transport connection with HMS, if one doesn't already exist, and forwards the request to it.

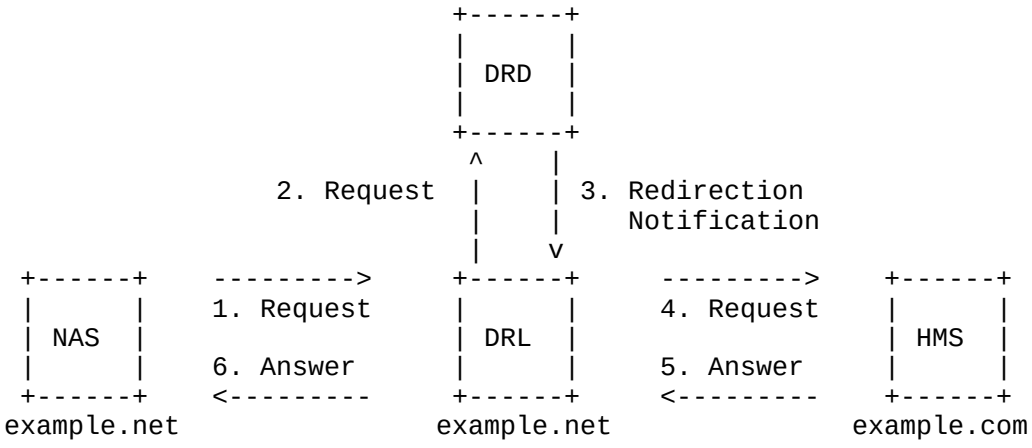


Figure 3: Redirecting a Diameter Message

Since redirect agents do not perform any application level processing, they provide relaying services for all Diameter applications, and therefore MUST advertise the Relay Application Identifier.

2.8.4. Translation Agents

A translation agent is a device that provides translation between two protocols (e.g., RADIUS<->Diameter, TACACS+<->Diameter). Translation agents are likely to be used as aggregation servers to communicate with a Diameter infrastructure, while allowing for the embedded systems to be migrated at a slower pace.

Given that the Diameter protocol introduces the concept of long-lived authorized sessions, translation agents MUST be session stateful and MUST maintain transaction state.

Translation of messages can only occur if the agent recognizes the application of a particular request, and therefore translation agents MUST only advertise their locally supported applications.

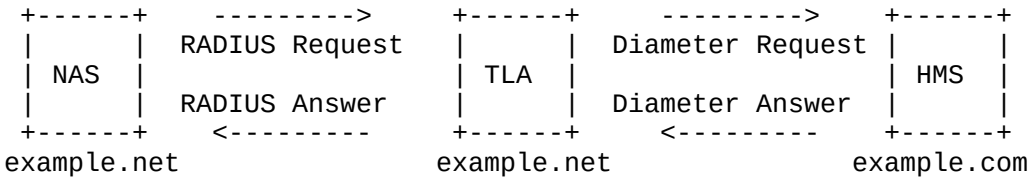


Figure 4: Translation of RADIUS to Diameter

2.9. Diameter Path Authorization

As noted in Section 2.2, Diameter provides transmission level security for each connection using TLS/TCP and DTLS/SCTP. Therefore, each connection can be authenticated, replay and integrity protected.

In addition to authenticating each connection, each connection as well as the entire session MUST also be authorized. Before initiating a connection, a Diameter Peer MUST check that its peers are authorized to act in their roles. For example, a Diameter peer may be authentic, but that does not mean that it is authorized to act as a Diameter Server advertising a set of Diameter applications.

Prior to bringing up a connection, authorization checks are performed at each connection along the path. Diameter capabilities negotiation (CER/CEA) also MUST be carried out, in order to determine what Diameter applications are supported by each peer. Diameter sessions MUST be routed only through authorized nodes that have advertised support for the Diameter application required by the session.

As noted in Section 6.1.9, a relay or proxy agent MUST append a Route-Record AVP to all requests forwarded. The AVP contains the identity of the peer the request was received from.

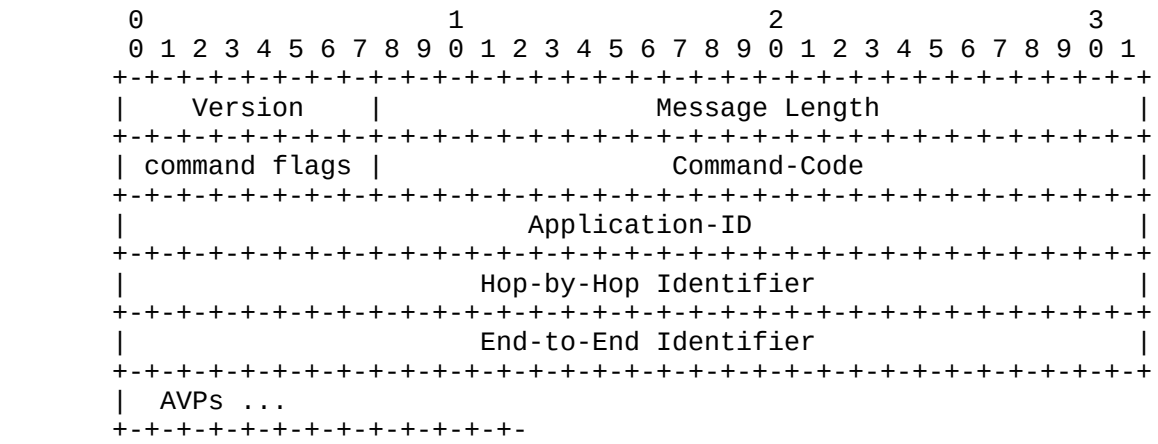
The home Diameter server, prior to authorizing a session, MUST check the Route-Record AVPs to make sure that the route traversed by the request is acceptable. For example, administrators within the home realm may not wish to honor requests that have been routed through an untrusted realm. By authorizing a request, the home Diameter server is implicitly indicating its willingness to engage in the business transaction as specified by the contractual relationship between the server and the previous hop. A DIAMETER_AUTHORIZATION_REJECTED error message (see Section 7.1.5) is sent if the route traversed by the request is unacceptable.

A home realm may also wish to check that each accounting request message corresponds to a Diameter response authorizing the session. Accounting requests without corresponding authorization responses SHOULD be subjected to further scrutiny, as should accounting requests indicating a difference between the requested and provided service.

Forwarding of an authorization response is considered evidence of a willingness to take on financial risk relative to the session. A local realm may wish to limit this exposure, for example, by establishing credit limits for intermediate realms and refusing to accept responses which would violate those limits. By issuing an accounting request corresponding to the authorization response, the local realm implicitly indicates its agreement to provide the service indicated in the authorization response. If the service cannot be provided by the local realm, then a DIAMETER_UNABLE_TO_COMPLY error message MUST be sent within the accounting request; a Diameter client receiving an authorization response for a service that it cannot perform MUST NOT substitute an alternate service, and then send accounting requests for the alternate service instead.

3. Diameter Header

A summary of the Diameter header format is shown below. The fields are transmitted in network byte order.



Version

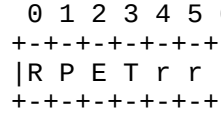
This Version field MUST be set to 1 to indicate Diameter Version 1.

Message Length

The Message Length field is three octets and indicates the length of the Diameter message including the header fields and the padded AVPs. Thus the message length field is always a multiple of 4.

Command Flags

The Command Flags field is eight bits. The following bits are assigned:



R(equest)

If set, the message is a request. If cleared, the message is an answer.

P(roxiable)

If set, the message MAY be proxied, relayed or redirected. If cleared, the message MUST be locally processed.

E(rror)

If set, the message contains a protocol error, and the message will not conform to the CCF described for this command. Messages with the 'E' bit set are commonly referred to as error messages. This bit MUST NOT be set in request messages (see Section 7.2).

T(Potentially re-transmitted message)

This flag is set after a link failover procedure, to aid the removal of duplicate requests. It is set when resending requests not yet acknowledged, as an indication of a possible duplicate due to a link failure. This bit MUST be cleared when sending a request for the first time, otherwise the sender MUST set this flag. Diameter agents only need to be concerned about the number of requests they send based on a single received request; retransmissions by other entities need not be tracked. Diameter agents that receive a request with the T flag set, MUST keep the T flag set in the forwarded request. This flag MUST NOT be set if an error answer message (e.g., a protocol error) has been received for the earlier message. It can be set only in cases where no answer has been received from the server for a request and the request is sent again. This flag MUST NOT be set in answer messages.

r(eserved)

These flag bits are reserved for future use, and MUST be set to zero, and ignored by the receiver.

Command-Code

The Command-Code field is three octets, and is used in order to communicate the command associated with the message. The 24-bit address space is managed by IANA (see Section 3.1).

Command-Code values 16,777,214 and 16,777,215 (hexadecimal values FFFFFE -FFFFFF) are reserved for experimental use (see Section 11.2).

Application-ID

Application-ID is four octets and is used to identify to which application the message is applicable for. The application can be an authentication application, an accounting application or a

vendor specific application.

The value of the application-id field in the header MUST be the same as any relevant application-id AVPs contained in the message.

Hop-by-Hop Identifier

The Hop-by-Hop Identifier is an unsigned 32-bit integer field (in network byte order) and aids in matching requests and replies. The sender MUST ensure that the Hop-by-Hop identifier in a request is unique on a given connection at any given time, and MAY attempt to ensure that the number is unique across reboots. The sender of an Answer message MUST ensure that the Hop-by-Hop Identifier field contains the same value that was found in the corresponding request. The Hop-by-Hop identifier is normally a monotonically increasing number, whose start value was randomly generated. An answer message that is received with an unknown Hop-by-Hop Identifier MUST be discarded.

End-to-End Identifier

The End-to-End Identifier is an unsigned 32-bit integer field (in network byte order) and is used to detect duplicate messages. Upon reboot implementations MAY set the high order 12 bits to contain the low order 12 bits of current time, and the low order 20 bits to a random value. Senders of request messages MUST insert a unique identifier on each message. The identifier MUST remain locally unique for a period of at least 4 minutes, even across reboots. The originator of an Answer message MUST ensure that the End-to-End Identifier field contains the same value that was found in the corresponding request. The End-to-End Identifier MUST NOT be modified by Diameter agents of any kind. The combination of the Origin-Host AVP (Section 6.3 and this field is used to detect duplicates. Duplicate requests SHOULD cause the same answer to be transmitted (modulo the hop-by-hop Identifier field and any routing AVPs that may be present), and MUST NOT affect any state that was set when the original request was processed. Duplicate answer messages that are to be locally consumed (see Section 6.2) SHOULD be silently discarded.

AVPs

AVPs are a method of encapsulating information relevant to the Diameter message. See Section 4 for more information on AVPs.

3.1. Command Codes

Each command Request/Answer pair is assigned a command code, and the sub-type (i.e., request or answer) is identified via the 'R' bit in the Command Flags field of the Diameter header.

Every Diameter message MUST contain a command code in its header's Command-Code field, which is used to determine the action that is to be taken for a particular message. The following Command Codes are defined in the Diameter base protocol:

Command-Name	Abbrev.	Code	Reference
Abort-Session-Request	ASR	274	8.5.1
Abort-Session-Answer	ASA	274	8.5.2
Accounting-Request	ACR	271	9.7.1
Accounting-Answer	ACA	271	9.7.2
Capabilities-Exchange-Request	CER	257	5.3.1
Capabilities-Exchange-Answer	CEA	257	5.3.2
Device-Watchdog-Request	DWR	280	5.5.1
Device-Watchdog-Answer	DWA	280	5.5.2
Disconnect-Peer-Request	DPR	282	5.4.1
Disconnect-Peer-Answer	DPA	282	5.4.2
Re-Auth-Request	RAR	258	8.3.1
Re-Auth-Answer	RAA	258	8.3.2
Session-Termination-Request	STR	275	8.4.1
Session-Termination-Answer	STA	275	8.4.2

3.2. Command Code Format Specification

Every Command Code defined MUST include a corresponding Command Code Format (CCF) specification, which is used to define the AVPs that MUST or MAY be present when sending the message. The following ABNF specifies the CCF used in the definition:

```
command-def      = "<" command-name ">" "::=" diameter-message

command-name     = diameter-name

diameter-name    = ALPHA *(ALPHA / DIGIT / "-")

diameter-message = header   *fixed  *required *optional
```

```
   header          = "<" "Diameter Header:" command-id
                     [r-bit] [p-bit] [e-bit] [application-id] ">"

   application-id  = 1*DIGIT

   command-id      = 1*DIGIT
                     ; The Command Code assigned to the command

   r-bit           = ", REQ"
                     ; If present, the 'R' bit in the Command
                     ; Flags is set, indicating that the message
                     ; is a request, as opposed to an answer.

   p-bit           = ", PXY"
                     ; If present, the 'P' bit in the Command
                     ; Flags is set, indicating that the message
                     ; is proxiable.

   e-bit           = ", ERR"
                     ; If present, the 'E' bit in the Command
                     ; Flags is set, indicating that the answer
                     ; message contains a Result-Code AVP in
                     ; the "protocol error" class.

   fixed           = [qual] "<" avp-spec ">"
                     ; Defines the fixed position of an AVP

   required        = [qual] "{" avp-spec "}"
                     ; The AVP MUST be present and can appear
                     ; anywhere in the message.


   optional        = [qual] "[" avp-name "]"
                     ; The avp-name in the 'optional' rule cannot
                     ; evaluate to any AVP Name which is included
                     ; in a fixed or required rule.  The AVP can
                     ; appear anywhere in the message.
                     ;
                     ; NOTE:  "[" and "]" have a slightly different
                     ; meaning than in ABNF. These braces
                     ; cannot be used to express optional fixed rules
                     ; (such as an optional ICV at the end). To do
                     ; this, the convention is '0*1fixed'.

   qual            = [min] "*" [max]
                     ; See ABNF conventions, RFC 5234, Section 4.
                     ; The absence of any qualifiers depends on
                     ; whether it precedes a fixed, required, or
```

```
                    ; optional rule. If a fixed or required rule has
                    ; no qualifier, then exactly one such AVP MUST
                    ; be present.  If an optional rule has no
                    ; qualifier, then 0 or 1 such AVP may be
                    ; present. If an optional rule has a qualifier,
                    ; then the value of min MUST be 0 if present.

   min              = 1*DIGIT
                    ; The minimum number of times the element may
                    ; be present. If absent, the default value is zero
                    ; for fixed and optional rules and one for
                    ; required  rules. The value MUST be at least one
                    ; for required rules.

   max              = 1*DIGIT
                    ; The maximum number of times the element may
                    ; be present. If absent, the default value is
                    ; infinity. A value of zero implies the AVP MUST
                    ; NOT be present.

   avp-spec         = diameter-name
                    ; The avp-spec has to be an AVP Name, defined
                    ; in the base or extended Diameter
                    ; specifications.

   avp-name         = avp-spec / "AVP"
                    ; The string "AVP" stands for *any* arbitrary AVP
                    ; Name, not otherwise listed in that command code
                    ; definition. The inclusion of this string
                    ; is recommended for all CCFs to allow for
                    ; extensibility.
```

The following is a definition of a fictitious command code:

```
   Example-Request ::= < Diameter Header: 9999999, REQ, PXY >
                       { User-Name }
                      1* { Origin-Host }
                       * [ AVP ]
```

3.3. Diameter Command Naming Conventions

Diameter command names typically includes one or more English words followed by the verb Request or Answer. Each English word is delimited by a hyphen. A three-letter acronym for both the request and answer is also normally provided.

An example is a message set used to terminate a session. The command name is Session-Terminate-Request and Session-Terminate-Answer, while the acronyms are STR and STA, respectively.

Both the request and the answer for a given command share the same command code. The request is identified by the R(equest) bit in the Diameter header set to one (1), to ask that a particular action be performed, such as authorizing a user or terminating a session. Once the receiver has completed the request it issues the corresponding answer, which includes a result code that communicates one of the following:

- The request was successful
- The request failed
- An additional request has to be sent to provide information the peer requires prior to returning a successful or failed answer.
- The receiver could not process the request, but provides information about a Diameter peer that is able to satisfy the request, known as redirect.

Additional information, encoded within AVPs, may also be included in answer messages.

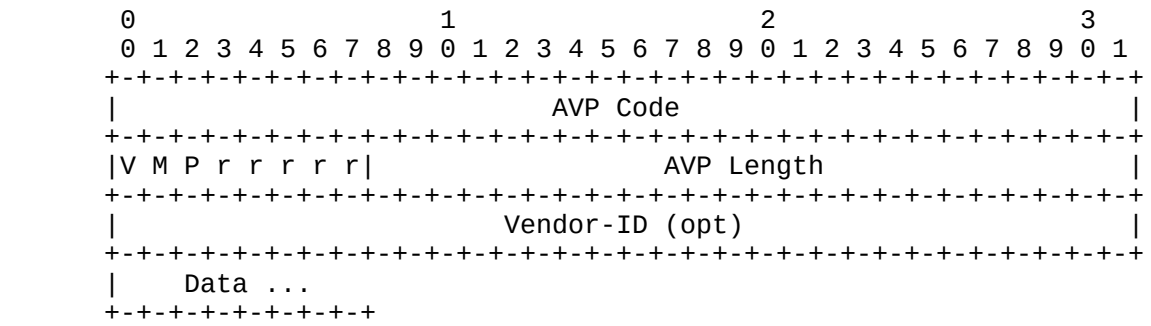
4. Diameter AVPs

Diameter AVPs carry specific authentication, accounting, authorization and routing information as well as configuration details for the request and reply.

Each AVP of type OctetString MUST be padded to align on a 32-bit boundary, while other AVP types align naturally. A number of zero-valued bytes are added to the end of the AVP Data field till a word boundary is reached. The length of the padding is not reflected in the AVP Length field.

4.1. AVP Header

The fields in the AVP header MUST be sent in network byte order. The format of the header is:



AVP Code

The AVP Code, combined with the Vendor-Id field, identifies the attribute uniquely. AVP numbers 1 through 255 are reserved for re-use of RADIUS attributes, without setting the Vendor-Id field. AVP numbers 256 and above are used for Diameter, which are allocated by IANA (see Section 11.1.1).

AVP Flags

The AVP Flags field informs the receiver how each attribute must be handled. New Diameter applications SHOULD NOT define additional AVP Flag bits. Note however, that new Diameter applications MAY define additional bits within the AVP Header, and an unrecognized bit SHOULD be considered an error. The sender of the AVP MUST set 'r' (reserved) bits to 0 and the receiver SHOULD ignore all 'r' (reserved) bits. The 'P' bit has been reserved for future usage of end-to-end security. At the time of writing there are no end-to-end security mechanisms specified therefore the 'P' bit SHOULD be set to 0.

The 'M' Bit, known as the Mandatory bit, indicates whether the receiver of the AVP MUST parse and understand the semantic of the AVP including its content. The receiving entity MUST return an appropriate error message if it receives an AVP that has the M-bit set but does not understand it. An exception applies when the AVP is embedded within a Grouped AVP. See Section 4.4 for details. Diameter Relay and redirect agents MUST NOT reject messages with unrecognized AVPs.

The 'M' bit MUST be set according to the rules defined in the application specification which introduces or re-uses this AVP. Within a given application, the M-bit setting for an AVP is either defined for all command types or for each command type.

AVPs with the 'M' bit cleared are informational only and a receiver that receives a message with such an AVP that is not supported, or whose value is not supported, MAY simply ignore the AVP.

The 'V' bit, known as the Vendor-Specific bit, indicates whether the optional Vendor-ID field is present in the AVP header. When set the AVP Code belongs to the specific vendor code address space.

AVP Length

The AVP Length field is three octets, and indicates the number of octets in this AVP including the AVP Code, AVP Length, AVP Flags, Vendor-ID field (if present) and the AVP data. If a message is received with an invalid attribute length, the message MUST be rejected.

4.1.1. Optional Header Elements

The AVP Header contains one optional field. This field is only present if the respective bit-flag is enabled.

Vendor-ID

The Vendor-ID field is present if the 'V' bit is set in the AVP Flags field. The optional four-octet Vendor-ID field contains the IANA assigned "SMI Network Management Private Enterprise Codes" [ENTERPRISE] value, encoded in network byte order. Any vendor or standardization organization that are also treated like vendors in the IANA managed "SMI Network Management Private Enterprise Codes" space wishing to implement a vendor-specific Diameter AVP MUST use their own Vendor-ID along with their privately managed AVP address space, guaranteeing that they will not collide with any other vendor's vendor-specific AVP(s), nor with future IETF AVPs.

A vendor ID value of zero (0) corresponds to the IETF adopted AVP values, as managed by the IANA. Since the absence of the vendor ID field implies that the AVP in question is not vendor specific, implementations MUST NOT use the zero (0) vendor ID.

4.2. Basic AVP Data Formats

The Data field is zero or more octets and contains information specific to the Attribute. The format and length of the Data field is determined by the AVP Code and AVP Length fields. The format of the Data field MUST be one of the following base data types or a data

type derived from the base data types. In the event that a new Basic AVP Data Format is needed, a new version of this RFC MUST be created.

OctetString

The data contains arbitrary data of variable length. Unless otherwise noted, the AVP Length field MUST be set to at least 8 (12 if the 'V' bit is enabled). AVP Values of this type that are not a multiple of four-octets in length is followed by the necessary padding so that the next AVP (if any) will start on a 32-bit boundary.

Integer32

32 bit signed value, in network byte order. The AVP Length field MUST be set to 12 (16 if the 'V' bit is enabled).

Integer64

64 bit signed value, in network byte order. The AVP Length field MUST be set to 16 (20 if the 'V' bit is enabled).

Unsigned32

32 bit unsigned value, in network byte order. The AVP Length field MUST be set to 12 (16 if the 'V' bit is enabled).

Unsigned64

64 bit unsigned value, in network byte order. The AVP Length field MUST be set to 16 (20 if the 'V' bit is enabled).

Float32

This represents floating point values of single precision as described by [FLOATPOINT]. The 32-bit value is transmitted in network byte order. The AVP Length field MUST be set to 12 (16 if the 'V' bit is enabled).

Float64

This represents floating point values of double precision as described by [FLOATPOINT]. The 64-bit value is transmitted in network byte order. The AVP Length field MUST be set to 16 (20 if the 'V' bit is enabled).

Grouped

The Data field is specified as a sequence of AVPs. Each of these AVPs follows - in the order in which they are specified - including their headers and padding. The AVP Length field is set to 8 (12 if the 'V' bit is enabled) plus the total length of all included AVPs, including their headers and padding. Thus the AVP length field of an AVP of type Grouped is always a multiple of 4.

4.3. Derived AVP Data Formats

In addition to using the Basic AVP Data Formats, applications may define data formats derived from the Basic AVP Data Formats. An application that defines new Derived AVP Data Formats MUST include them in a section entitled "Derived AVP Data Formats", using the same format as the definitions below. Each new definition MUST be either defined or listed with a reference to the RFC that defines the format.

4.3.1. Common Derived AVP Data Formats

The following are commonly used Derived AVP Data Formats.

Address

The Address format is derived from the OctetString AVP Base Format. It is a discriminated union, representing, for example a 32-bit (IPv4) [RFC791] or 128-bit (IPv6) [RFC4291] address, most significant octet first. The first two octets of the Address AVP represents the AddressType, which contains an Address Family defined in [IANAADFAM]. The AddressType is used to discriminate the content and format of the remaining octets.

Time

The Time format is derived from the OctetString AVP Base Format. The string MUST contain four octets, in the same format as the

first four bytes are in the NTP timestamp format. The NTP Timestamp format is defined in Chapter 3 of [RFC5905].

This represents the number of seconds since 0h on 1 January 1900 with respect to the Coordinated Universal Time (UTC).

On 6h 28m 16s UTC, 7 February 2036 the time value will overflow. SNTP [RFC5905] describes a procedure to extend the time to 2104. This procedure MUST be supported by all Diameter nodes.

UTF8String

The UTF8String format is derived from the OctetString AVP Base Format. This is a human readable string represented using the ISO/IEC IS 10646-1 character set, encoded as an OctetString using the UTF-8 transformation format [RFC3629].

Since additional code points are added by amendments to the 10646 standard from time to time, implementations MUST be prepared to encounter any code point from 0x00000001 to 0x7fffffff. Byte sequences that do not correspond to the valid encoding of a code point into UTF-8 charset or are outside this range are prohibited.

The use of control codes SHOULD be avoided. When it is necessary to represent a new line, the control code sequence CR LF SHOULD be used.

The use of leading or trailing white space SHOULD be avoided.

For code points not directly supported by user interface hardware or software, an alternative means of entry and display, such as hexadecimal, MAY be provided.

For information encoded in 7-bit US-ASCII, the UTF-8 charset is identical to the US-ASCII charset.

UTF-8 may require multiple bytes to represent a single character / code point; thus the length of an UTF8String in octets may be different from the number of characters encoded.

Note that the AVP Length field of an UTF8String is measured in octets, not characters.

DiameterIdentity

The DiameterIdentity format is derived from the OctetString AVP Base Format.

DiameterIdentity = FQDN/Realm

DiameterIdentity value is used to uniquely identify either:

* A Diameter node for purposes of duplicate connection and routing loop detection.

* A Realm to determine whether messages can be satisfied locally, or whether they must be routed or redirected.

When a DiameterIdentity is used to identify a Diameter node the contents of the string MUST be the FQDN of the Diameter node. If multiple Diameter nodes run on the same host, each Diameter node MUST be assigned a unique DiameterIdentity. If a Diameter node can be identified by several FQDNs, a single FQDN should be picked at startup, and used as the only DiameterIdentity for that node, whatever the connection it is sent on. Note that in this document, DiameterIdentity is in ASCII form in order to be compatible with existing DNS infrastructure. See Appendix D for interactions between the Diameter protocol and Internationalized Domain Name (IDNs).

DiameterURI

The DiameterURI MUST follow the Uniform Resource Identifiers (RFC3986) syntax [RFC3986] rules specified below:

```
   "aaa://" FQDN [ port ] [ transport ] [ protocol ]

                  ; No transport security

   "aaas://" FQDN [ port ] [ transport ] [ protocol ]

                  ; Transport security used

   FQDN              = < Fully Qualified Domain Name >

   port              = ":" 1*DIGIT

                  ; One of the ports used to listen for
                  ; incoming connections.
                  ; If absent, the default Diameter port
                  ; (3868) is assumed if no transport
                  ; security is used and port <TBD> when
                  ; transport security (TLS/TCP and DTLS/SCTP)
                  ; is used.

   transport         = ";transport=" transport-protocol

                  ; One of the transports used to listen
                  ; for incoming connections.  If absent,
                  ; the default protocol is assumed to be TCP.
                  ; UDP MUST NOT be used when the aaa-protocol
                  ; field is set to diameter.

   transport-protocol = ( "tcp" / "sctp" / "udp" )

   protocol          = ";protocol=" aaa-protocol

                  ; If absent, the default AAA protocol
                  ; is Diameter.

   aaa-protocol      = ( "diameter" / "radius" / "tacacs+" )
```

The following are examples of valid Diameter host identities:

```
aaa://host.example.com;transport=tcp
aaa://host.example.com:6666;transport=tcp
aaa://host.example.com;protocol=diameter
aaa://host.example.com:6666;protocol=diameter
aaa://host.example.com:6666;transport=tcp;protocol=diameter
aaa://host.example.com:1813;transport=udp;protocol=radius
```

Enumerated

Enumerated is derived from the Integer32 AVP Base Format. The definition contains a list of valid values and their interpretation and is described in the Diameter application introducing the AVP.

IPFilterRule

The IPFilterRule format is derived from the OctetString AVP Base Format and uses the ASCII charset. The rule syntax is a modified subset of ipfw(8) from FreeBSD. Packets may be filtered based on the following information that is associated with it:

Direction	(in or out)
Source and destination IP address	(possibly masked)
Protocol	
Source and destination port	(lists or ranges)
TCP flags	
IP fragment flag	
IP options	
ICMP types	

Rules for the appropriate direction are evaluated in order, with the first matched rule terminating the evaluation. Each packet is evaluated once. If no rule matches, the packet is dropped if the last rule evaluated was a permit, and passed if the last rule was a deny.

IPFilterRule filters MUST follow the format:

```
action dir proto from src to dst [options]

action       permit - Allow packets that match the rule.
             deny   - Drop packets that match the rule.

dir          "in" is from the terminal, "out" is to the
             terminal.

proto        An IP protocol specified by number.  The "ip"
             keyword means any protocol will match.

src and dst  <address/mask> [ports]

             The <address/mask> may be specified as:
             ipno       An IPv4 or IPv6 number in dotted-
                        quad or canonical IPv6 form.  Only
```

this exact IP number will match the rule.

ipno/bits An IP number as above with a mask width of the form 192.0.2.10/24. In this case, all IP numbers from 192.0.2.0 to 192.0.2.255 will match. The bit width MUST be valid for the IP version and the IP number MUST NOT have bits set beyond the mask. For a match to occur, the same IP version must be present in the packet that was used in describing the IP address. To test for a particular IP version, the bits part can be set to zero. The keyword "any" is 0.0.0.0/0 or the IPv6 equivalent. The keyword "assigned" is the address or set of addresses assigned to the terminal. For IPv4, a typical first rule is often "deny in ip! assigned"

The sense of the match can be inverted by preceding an address with the not modifier (!), causing all other addresses to be matched instead. This does not affect the selection of port numbers.

With the TCP, UDP and SCTP protocols, optional ports may be specified as:

```
{port/port-port}[,ports[,...]]
```

The '-' notation specifies a range of ports (including boundaries).

Fragmented packets that have a non-zero offset (i.e., not the first fragment) will never match a rule that has one or more port specifications. See the frag option for details on matching fragmented packets.

options:

frag Match if the packet is a fragment and this is not the first fragment of the datagram. frag may not be used in conjunction with either tcpflags or TCP/UDP port specifications.

ipoptions spec
Match if the IP header contains the comma separated list of options specified in spec. The supported IP options are:

ssrr (strict source route), lsrr (loose source route), rr (record packet route) and ts (timestamp). The absence of a particular option may be denoted with a '!'.

tcpoptions spec
Match if the TCP header contains the comma separated list of options specified in spec. The supported TCP options are:

mss (maximum segment size), window (tcp window advertisement), sack (selective ack), ts (rfc1323 timestamp) and cc (rfc1644 t/tcp connection count). The absence of a particular option may be denoted with a '!'.

established
TCP packets only. Match packets that have the RST or ACK bits set.

setup TCP packets only. Match packets that have the SYN bit set but no ACK bit.

tcpflags spec
TCP packets only. Match if the TCP header contains the comma separated list of flags specified in spec. The supported TCP flags are:

fin, syn, rst, psh, ack and urg. The absence of a particular flag may be denoted with a '!'. A rule that contains a tcpflags specification can never match a fragmented packet that has a non-zero offset. See the frag option for details on matching fragmented packets.

icmptypes types
ICMP packets only. Match if the ICMP type is in the list types. The list may be specified as any combination of ranges or individual types separated by commas. Both the numeric values and the symbolic values listed below can be used. The supported ICMP types are:

echo reply (0), destination unreachable (3), source quench (4), redirect (5), echo request (8), router advertisement (9), router solicitation (10), time-to-live exceeded (11), IP header bad (12), timestamp request (13), timestamp reply (14), information request (15), information reply (16), address mask request (17) and address mask reply (18).

There is one kind of packet that the access device MUST always discard, that is an IP fragment with a fragment offset of one. This is a valid packet, but it only has one use, to try to circumvent firewalls.

An access device that is unable to interpret or apply a deny rule MUST terminate the session. An access device that is unable to interpret or apply a permit rule MAY apply a more restrictive rule. An access device MAY apply deny rules of its own before the supplied rules, for example to protect the access device owner's infrastructure.

4.4. Grouped AVP Values

The Diameter protocol allows AVP values of type 'Grouped'. This implies that the Data field is actually a sequence of AVPs. It is possible to include an AVP with a Grouped type within a Grouped type, that is, to nest them. AVPs within an AVP of type Grouped have the same padding requirements as non-Grouped AVPs, as defined in Section 4.4.

The AVP Code numbering space of all AVPs included in a Grouped AVP is the same as for non-grouped AVPs. Receivers of a Grouped AVP that does not have the 'M' (mandatory) bit set and one or more of the encapsulated AVPs within the group has the 'M' (mandatory) bit set MAY simply be ignored if the Grouped AVP itself is unrecognized. The rule applies even if the encapsulated AVP with its 'M' (mandatory) bit set is further encapsulated within other sub-groups; i.e. other Grouped AVPs embedded within the Grouped AVP.

Every Grouped AVP defined MUST include a corresponding grammar, using CCF [RFC5234] (with modifications), as defined below.

```
grouped-avp-def  = "<" name ">" "::=" avp

name-fmt         = ALPHA *(ALPHA / DIGIT / "-")

name             = name-fmt
                   ; The name has to be the name of an AVP,
                   ; defined in the base or extended Diameter
                   ; specifications.

avp              = header *fixed *required *optional

header           = "<" "AVP-Header:" avpcode [vendor] ">"

avpcode          = 1*DIGIT
                   ; The AVP Code assigned to the Grouped AVP

vendor           = 1*DIGIT
                   ; The Vendor-ID assigned to the Grouped AVP.
                   ; If absent, the default value of zero is
                   ; used.
```

4.4.1. Example AVP with a Grouped Data type

The Example-AVP (AVP Code 999999) is of type Grouped and is used to clarify how Grouped AVP values work. The Grouped Data field has the following CCF grammar:

```
   Example-AVP  ::= < AVP Header: 999999 >
                     { Origin-Host }
                   1*{ Session-Id }
                    *[ AVP ]
```

An Example-AVP with Grouped Data follows.

The Origin-Host AVP (Section 6.3) is required. In this case:

```
   Origin-Host = "example.com".
```

One or more Session-Ids must follow. Here there are two:

```
   Session-Id =
     "grump.example.com:33041;23432;893;0AF3B81"

   Session-Id =
     "grump.example.com:33054;23561;2358;0AF3B82"
```

optional AVPs included are

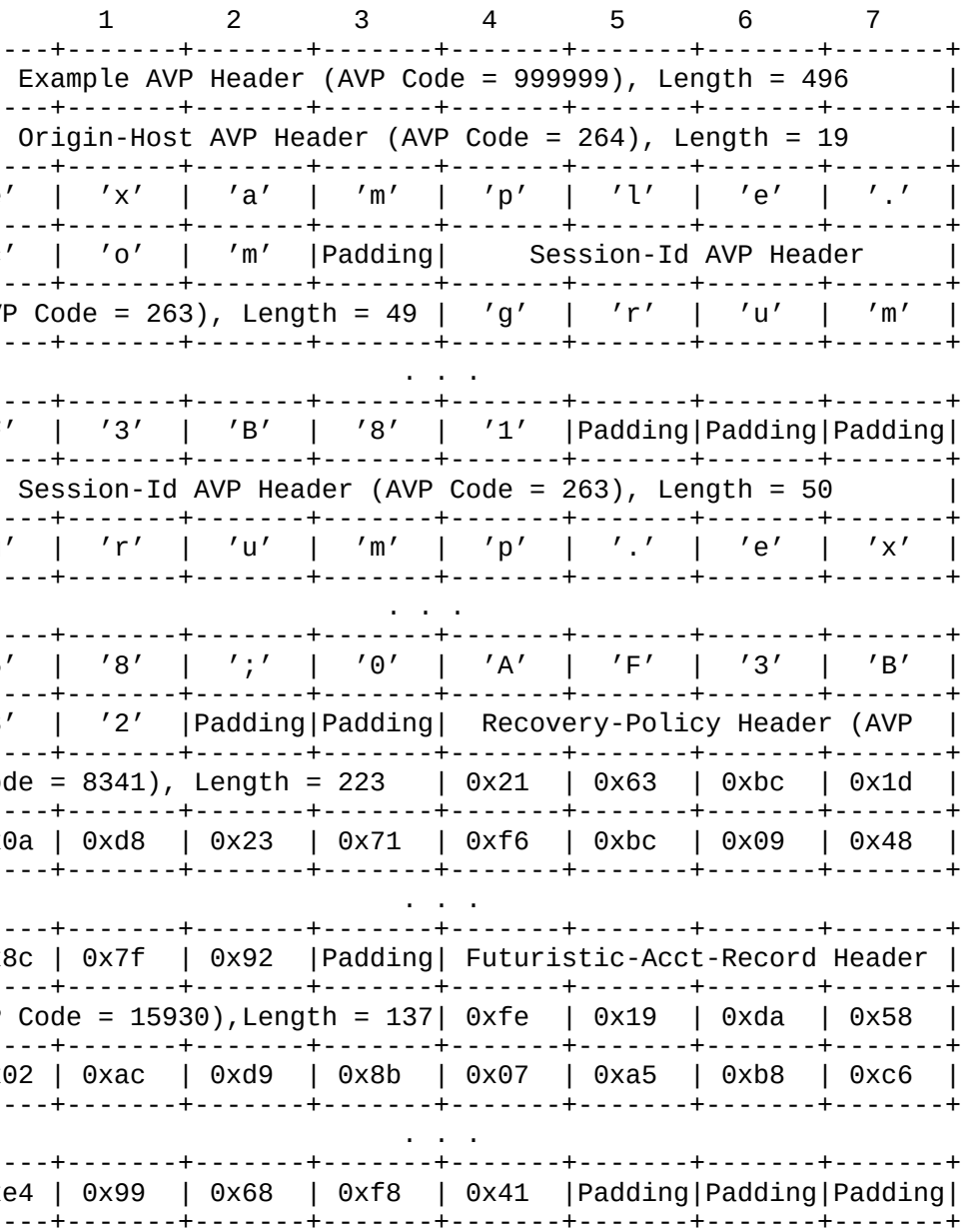
```
   Recovery-Policy = <binary>
      2163bc1d0ad82371f6bc09484133c3f09ad74a0dd5346d54195a7cf0b35
      2cabc881839a4fdcfbc1769e2677a4c1fb499284c5f70b48f58503a45c5
      c2d6943f82d5930f2b7c1da640f476f0e9c9572a50db8ea6e51e1c2c7bd
      f8bb43dc995144b8dbe297ac739493946803e1cee3e15d9b765008a1b2a
      cf4ac777c80041d72c01e691cf751dbf86e85f509f3988e5875dc905119
      26841f00f0e29a6d1ddc1a842289d440268681e052b30fb638045f7779c
      1d873c784f054f688f5001559ecff64865ef975f3e60d2fd7966b8c7f92

   Futuristic-Acct-Record = <binary>
      fe19da5802acd98b07a5b86cb4d5d03f0314ab9ef1ad0b67111ff3b90a0
      57fe29620bf3585fd2dd9fcc38ce62f6cc208c6163c008f4258d1bc88b8
      17694a74ccad3ec69269461b14b2e7a4c111fb239e33714da207983f58c
      41d018d56fe938f3cbf089aac12a912a2f0d1923a9390e5f789cb2e5067
      d3427475e49968f841
```

The data for the optional AVPs is represented in hex since the format of these AVPs is neither known at the time of definition of the Example-AVP group, nor (likely) at the time when the example instance of this AVP is interpreted - except by Diameter implementations which support the same set of AVPs. The encoding example illustrates how padding is used and how length fields are calculated. Also note that AVPs may be present in the Grouped AVP value which the receiver cannot interpret (here, the Recover-Policy and Futuristic-Acct-Record AVPs). The length of the Example-AVP is the sum of all the length of the member AVPs including their padding plus the Example-AVP header

size.

This AVP would be encoded as follows:



4.5. Diameter Base Protocol AVPs

The following table describes the Diameter AVPs defined in the base protocol, their AVP Code values, types, possible flag values.

Due to space constraints, the short form DiamIdent is used to represent DiameterIdentity.

Attribute Name	AVP Code	Section Defined	Data Type	AVP Flag rules: MUST	AVP Flag rules: MUST NOT
Acct-Interim-Interval	85	9.8.2	Unsigned32	M	V
Accounting-Realtime-Required	483	9.8.7	Enumerated	M	V
Acct-Multi-Session-Id	50	9.8.5	UTF8String	M	V
Accounting-Record-Number	485	9.8.3	Unsigned32	M	V
Accounting-Record-Type	480	9.8.1	Enumerated	M	V
Accounting-Session-Id	44	9.8.4	OctetString	M	V
Accounting-Sub-Session-Id	287	9.8.6	Unsigned64	M	V
Acct-Application-Id	259	6.9	Unsigned32	M	V
Auth-Application-Id	258	6.8	Unsigned32	M	V
Auth-Request-Type	274	8.7	Enumerated	M	V
Authorization-Lifetime	291	8.9	Unsigned32	M	V
Auth-Grace-Period	276	8.10	Unsigned32	M	V
Auth-Session-State	277	8.11	Enumerated	M	V
Re-Auth-Request-Type	285	8.12	Enumerated	M	V
Class	25	8.20	OctetString	M	V
Destination-Host	293	6.5	DiamIdent	M	V
Destination-Realm	283	6.6	DiamIdent	M	V
Disconnect-Cause	273	5.4.3	Enumerated	M	V
Error-Message	281	7.3	UTF8String		V,M
Error-Reporting-Host	294	7.4	DiamIdent		V,M
Event-Timestamp	55	8.21	Time	M	V
Experimental-Result	297	7.6	Grouped	M	V

Attribute Name	AVP Code	Section Defined	Data Type	AVP Flag rules	
				MUST	MUST NOT
Experimental-Result-Code	298	7.7	Unsigned32	M	V
Failed-AVP	279	7.5	Grouped	M	V
Firmware-Revision	267	5.3.4	Unsigned32		V,M
Host-IP-Address	257	5.3.5	Address	M	V
Inband-Security-Id	299	6.10	Unsigned32	M	V
Multi-Round-Time-Out	272	8.19	Unsigned32	M	V
Origin-Host	264	6.3	DiamIdent	M	V
Origin-Realm	296	6.4	DiamIdent	M	V
Origin-State-Id	278	8.16	Unsigned32	M	V
Product-Name	269	5.3.7	UTF8String		V,M
Proxy-Host	280	6.7.3	DiamIdent	M	V
Proxy-Info	284	6.7.2	Grouped	M	V
Proxy-State	33	6.7.4	OctetString	M	V
Redirect-Host	292	6.12	DiamURI	M	V
Redirect-Host-Usage	261	6.13	Enumerated	M	V
Redirect-Max-Cache-Time	262	6.14	Unsigned32	M	V
Result-Code	268	7.1	Unsigned32	M	V
Route-Record	282	6.7.1	DiamIdent	M	V
Session-Id	263	8.8	UTF8String	M	V
Session-Timeout	27	8.13	Unsigned32	M	V
Session-Binding	270	8.17	Unsigned32	M	V
Session-Server-Failover	271	8.18	Enumerated	M	V
Supported-Vendor-Id	265	5.3.6	Unsigned32	M	V
Termination-Cause	295	8.15	Enumerated	M	V
User-Name	1	8.14	UTF8String	M	V
Vendor-Id	266	5.3.3	Unsigned32	M	V
Vendor-Specific-Application-Id	260	6.11	Grouped	M	V

5. Diameter Peers

This section describes how Diameter nodes establish connections and communicate with peers.

5.1. Peer Connections

Connections between diameter peers are established using their valid DiameterIdentity. A Diameter node initiating a connection to a peer MUST know the peers DiameterIdentity. Methods for discovering a Diameter peer can be found in Section 5.2.

Although a Diameter node may have many possible peers that it is able to communicate with, it may not be economical to have an established connection to all of them. At a minimum, a Diameter node SHOULD have an established connection with two peers per realm, known as the primary and secondary peers. Of course, a node MAY have additional connections, if it is deemed necessary. Typically, all messages for a realm are sent to the primary peer, but in the event that failover procedures are invoked, any pending requests are sent to the secondary peer. However, implementations are free to load balance requests between a set of peers.

Note that a given peer MAY act as a primary for a given realm, while acting as a secondary for another realm.

When a peer is deemed suspect, which could occur for various reasons, including not receiving a DWA within an allotted timeframe, no new requests should be forwarded to the peer, but failover procedures are invoked. When an active peer is moved to this mode, additional connections SHOULD be established to ensure that the necessary number of active connections exists.

There are two ways that a peer is removed from the suspect peer list:

1. The peer is no longer reachable, causing the transport connection to be shutdown. The peer is moved to the closed state.

2. Three watchdog messages are exchanged with accepted round trip times, and the connection to the peer is considered stabilized.

In the event the peer being removed is either the primary or secondary, an alternate peer SHOULD replace the deleted peer, and assume the role of either primary or secondary.

5.2. Diameter Peer Discovery

Allowing for dynamic Diameter agent discovery makes possible simpler and more robust deployment of Diameter services. In order to promote interoperable implementations of Diameter peer discovery, the following mechanisms (manual configuration and DNS) are described. These are based on existing IETF standards. Both mechanisms MUST be supported by all Diameter implementations; either MAY be used.

There are two cases where Diameter peer discovery may be performed. The first is when a Diameter client needs to discover a first-hop Diameter agent. The second case is when a Diameter agent needs to discover another agent - for further handling of a Diameter operation. In both cases, the following 'search order' is recommended:

1. The Diameter implementation consults its list of static (manually) configured Diameter agent locations. These will be used if they exist and respond.

2. The Diameter implementation performs a NAPTR query for a server in a particular realm. The Diameter implementation has to know in advance which realm to look for a Diameter agent. This could be deduced, for example, from the 'realm' in a NAI that a Diameter implementation needed to perform a Diameter operation on.

   The NAPTR usage in Diameter follows the S-NAPTR DDDS application [RFC3958] in which the SERVICE field includes tags for the desired application and supported application protocol. The application service tag for a Diameter application is 'aaa' and the supported application protocol tags are 'diameter.tcp', 'diameter.sctp', 'diameter.dtls' or 'diameter.tls.tcp' [RFC6408].

   The client can follow the resolution process defined by the S-NAPTR DDDS [RFC3958] application to find a matching SRV, A or AAAA record of a suitable peer. The domain suffixes in the NAPTR replacement field SHOULD match the domain of the original query. An example can be found in Appendix B.

3. If no NAPTR records are found, the requester directly queries for one of the following SRV records: for Diameter over TCP, use "_diameter._tcp.realm"; for Diameter over TLS, use "_diameters._tcp.realm"; for Diameter over SCTP, use "_diameter._sctp.realm"; for Diameter over DTLS, use "_diameters._sctp.realm". If SRV records are found then the

requester can perform address record query (A RR's and/or AAAA RR's) for the target hostname specified in the SRV records following the rules given in Gulbrandsen, et al. [RFC2782]. If no SRV records are found, the requester gives up.

If the server is using a site certificate, the domain name in the NAPTR query and the domain name in the replacement field MUST both be valid based on the site certificate handed out by the server in the TLS/TCP and DTLS/SCTP or IKE exchange. Similarly, the domain name in the SRV query and the domain name in the target in the SRV record MUST both be valid based on the same site certificate. Otherwise, an attacker could modify the DNS records to contain replacement values in a different domain, and the client could not validate that this was the desired behavior, or the result of an attack.

Also, the Diameter Peer MUST check to make sure that the discovered peers are authorized to act in its role. Authentication via IKE or TLS/TCP and DTLS/SCTP, or validation of DNS RRs via DNSSEC is not sufficient to conclude this. For example, a web server may have obtained a valid TLS/TCP and DTLS/SCTP certificate, and secured RRs may be included in the DNS, but this does not imply that it is authorized to act as a Diameter Server.

Authorization can be achieved for example, by configuration of a Diameter Server Certification Authority (CA). The Server CA issues a certificate to the Diameter Server, which includes an Object Identifier (OID) to indicate the subject is a Diameter Server in the Extended Key Usage extension [RFC5280]. This certificate is then used during TLS/TCP, DTLS/SCTP, or IKE security negotiation. Note, however, that at the time of writing no Diameter Server Certification Authorities exist.

A dynamically discovered peer causes an entry in the Peer Table (see Section 2.6) to be created. Note that entries created via DNS MUST expire (or be refreshed) within the DNS TTL. If a peer is discovered outside of the local realm, a routing table entry (see Section 2.7) for the peer's realm is created. The routing table entry's expiration MUST match the peer's expiration value.

5.3. Capabilities Exchange

When two Diameter peers establish a transport connection, they MUST exchange the Capabilities Exchange messages, as specified in the peer state machine (see Section 5.6). This message allows the discovery of a peer's identity and its capabilities (protocol version number, the identifiers of supported Diameter applications, security mechanisms, etc.)

The receiver only issues commands to its peers that have advertised support for the Diameter application that defines the command. A Diameter node MUST cache the supported Application Ids in order to ensure that unrecognized commands and/or AVPs are not unnecessarily sent to a peer.

A receiver of a Capabilities-Exchange-Req (CER) message that does not have any applications in common with the sender MUST return a Capabilities-Exchange-Answer (CEA) with the Result-Code AVP set to DIAMETER_NO_COMMON_APPLICATION, and SHOULD disconnect the transport layer connection. Note that receiving a CER or CEA from a peer advertising itself as a Relay (see Section 2.4) MUST be interpreted as having common applications with the peer.

The receiver of the Capabilities-Exchange-Request (CER) MUST determine common applications by computing the intersection of its own set of supported Application Id against all of the application identifier AVPs (Auth-Application-Id, Acct-Application-Id and Vendor-Specific-Application-Id) present in the CER. The value of the Vendor-Id AVP in the Vendor-Specific-Application-Id MUST NOT be used during computation. The sender of the Capabilities-Exchange-Answer (CEA) SHOULD include all of its supported applications as a hint to the receiver regarding all of its application capabilities.

Diameter implementations SHOULD first attempt to establish a TLS/TCP and DTLS/SCTP connection prior to the CER/CEA exchange. This protects the capabilities information of both peers. To support older Diameter implementations that do not fully conform to this document, the transport security MAY still be negotiated via Inband-Security AVP. In this case, the receiver of a Capabilities-Exchange-Req (CER) message that does not have any security mechanisms in common with the sender MUST return a Capabilities-Exchange-Answer (CEA) with the Result-Code AVP set to DIAMETER_NO_COMMON_SECURITY, and SHOULD disconnect the transport layer connection.

CERs received from unknown peers MAY be silently discarded, or a CEA MAY be issued with the Result-Code AVP set to DIAMETER_UNKNOWN_PEER. In both cases, the transport connection is closed. If the local policy permits receiving CERs from unknown hosts, a successful CEA MAY be returned. If a CER from an unknown peer is answered with a successful CEA, the lifetime of the peer entry is equal to the lifetime of the transport connection. In case of a transport failure, all the pending transactions destined to the unknown peer can be discarded.

The CER and CEA messages MUST NOT be proxied, redirected or relayed.

Since the CER/CEA messages cannot be proxied, it is still possible

that an upstream agent receives a message for which it has no available peers to handle the application that corresponds to the Command-Code. In such instances, the 'E' bit is set in the answer message (Section 7) with the Result-Code AVP set to DIAMETER_UNABLE_TO_DELIVER to inform the downstream to take action (e.g., re-routing request to an alternate peer).

With the exception of the Capabilities-Exchange-Request message, a message of type Request that includes the Auth-Application-Id or Acct-Application-Id AVPs, or a message with an application-specific command code, MAY only be forwarded to a host that has explicitly advertised support for the application (or has advertised the Relay Application Id).

5.3.1. Capabilities-Exchange-Request

The Capabilities-Exchange-Request (CER), indicated by the Command-Code set to 257 and the Command Flags' 'R' bit set, is sent to exchange local capabilities. Upon detection of a transport failure, this message MUST NOT be sent to an alternate peer.

When Diameter is run over SCTP [RFC4960] or DTLS/SCTP [RFC6083], which allow for connections to span multiple interfaces and multiple IP addresses, the Capabilities-Exchange-Request message MUST contain one Host-IP-Address AVP for each potential IP address that MAY be locally used when transmitting Diameter messages.

Message Format

```
<CER> ::= < Diameter Header: 257, REQ >
          { Origin-Host }
          { Origin-Realm }
        1* { Host-IP-Address }
          { Vendor-Id }
          { Product-Name }
          [ Origin-State-Id ]
         * [ Supported-Vendor-Id ]
         * [ Auth-Application-Id ]
         * [ Inband-Security-Id ]
         * [ Acct-Application-Id ]
         * [ Vendor-Specific-Application-Id ]
          [ Firmware-Revision ]
         * [ AVP ]
```

5.3.2. Capabilities-Exchange-Answer

The Capabilities-Exchange-Answer (CEA), indicated by the Command-Code set to 257 and the Command Flags' 'R' bit cleared, is sent in response to a CER message.

When Diameter is run over SCTP [RFC4960] or DTLS/SCTP [RFC6083], which allow connections to span multiple interfaces, hence, multiple IP addresses, the Capabilities-Exchange-Answer message MUST contain one Host-IP-Address AVP for each potential IP address that MAY be locally used when transmitting Diameter messages.

Message Format

```
<CEA> ::= < Diameter Header: 257 >
          { Result-Code }
          { Origin-Host }
          { Origin-Realm }
        1* { Host-IP-Address }
          { Vendor-Id }
          { Product-Name }
          [ Origin-State-Id ]
          [ Error-Message ]
          [ Failed-AVP ]
         * [ Supported-Vendor-Id ]
         * [ Auth-Application-Id ]
         * [ Inband-Security-Id ]
         * [ Acct-Application-Id ]
         * [ Vendor-Specific-Application-Id ]
           [ Firmware-Revision ]
         * [ AVP ]
```

5.3.3. Vendor-Id AVP

The Vendor-Id AVP (AVP Code 266) is of type Unsigned32 and contains the IANA "SMI Network Management Private Enterprise Codes" [ENTERPRISE] value assigned to the Diameter Software vendor. It is envisioned that the combination of the Vendor-Id, Product-Name (Section 5.3.7) and the Firmware-Revision (Section 5.3.4) AVPs may provide useful debugging information.

A Vendor-Id value of zero in the CER or CEA messages is reserved and indicates that this field is ignored.

5.3.4. Firmware-Revision AVP

The Firmware-Revision AVP (AVP Code 267) is of type Unsigned32 and is used to inform a Diameter peer of the firmware revision of the

issuing device.

For devices that do not have a firmware revision (general purpose computers running Diameter software modules, for instance), the revision of the Diameter software module may be reported instead.

5.3.5. Host-IP-Address AVP

The Host-IP-Address AVP (AVP Code 257) is of type Address and is used to inform a Diameter peer of the sender's IP address. All source addresses that a Diameter node expects to use with SCTP [RFC4960] or DTLS/SCTP [RFC6083] MUST be advertised in the CER and CEA messages by including a Host-IP-Address AVP for each address.

5.3.6. Supported-Vendor-Id AVP

The Supported-Vendor-Id AVP (AVP Code 265) is of type Unsigned32 and contains the IANA "SMI Network Management Private Enterprise Codes" [ENTERPRISE] value assigned to a vendor other than the device vendor but including the application vendor. This is used in the CER and CEA messages in order to inform the peer that the sender supports (a subset of) the vendor-specific AVPs defined by the vendor identified in this AVP. The value of this AVP MUST NOT be set to zero. Multiple instances of this AVP containing the same value SHOULD NOT be sent.

5.3.7. Product-Name AVP

The Product-Name AVP (AVP Code 269) is of type UTF8String, and contains the vendor assigned name for the product. The Product-Name AVP SHOULD remain constant across firmware revisions for the same product.

5.4. Disconnecting Peer connections

When a Diameter node disconnects one of its transport connections, its peer cannot know the reason for the disconnect, and will most likely assume that a connectivity problem occurred, or that the peer has rebooted. In these cases, the peer may periodically attempt to reconnect, as stated in Section 2.1. In the event that the disconnect was a result of either a shortage of internal resources, or simply that the node in question has no intentions of forwarding any Diameter messages to the peer in the foreseeable future, a periodic connection request would not be welcomed. The Disconnection-Reason AVP contains the reason the Diameter node issued the Disconnect-Peer-Request message.

The Disconnect-Peer-Request message is used by a Diameter node to

inform its peer of its intent to disconnect the transport layer, and that the peer shouldn't reconnect unless it has a valid reason to do so (e.g., message to be forwarded). Upon receipt of the message, the Disconnect-Peer-Answer is returned, which SHOULD contain an error if messages have recently been forwarded, and are likely in flight, which would otherwise cause a race condition.

The receiver of the Disconnect-Peer-Answer initiates the transport disconnect. The sender of the Disconnect-Peer-Answer should be able to detect the transport closure and cleanup the connection.

5.4.1. Disconnect-Peer-Request

The Disconnect-Peer-Request (DPR), indicated by the Command-Code set to 282 and the Command Flags' 'R' bit set, is sent to a peer to inform its intentions to shutdown the transport connection. Upon detection of a transport failure, this message MUST NOT be sent to an alternate peer.

Message Format

```
<DPR>  ::= < Diameter Header: 282, REQ >
           { Origin-Host }
           { Origin-Realm }
           { Disconnect-Cause }
         * [ AVP ]
```

5.4.2. Disconnect-Peer-Answer

The Disconnect-Peer-Answer (DPA), indicated by the Command-Code set to 282 and the Command Flags' 'R' bit cleared, is sent as a response to the Disconnect-Peer-Request message. Upon receipt of this message, the transport connection is shutdown.

Message Format

```
<DPA>  ::= < Diameter Header: 282 >
           { Result-Code }
           { Origin-Host }
           { Origin-Realm }
           [ Error-Message ]
           [ Failed-AVP ]
         * [ AVP ]
```

5.4.3. Disconnect-Cause AVP

The Disconnect-Cause AVP (AVP Code 273) is of type Enumerated. A Diameter node MUST include this AVP in the Disconnect-Peer-Request message to inform the peer of the reason for its intention to shutdown the transport connection. The following values are supported:

REBOOTING 0
A scheduled reboot is imminent. Receiver of DPR with above result code MAY attempt reconnection.

BUSY 1
The peer's internal resources are constrained, and it has determined that the transport connection needs to be closed. Receiver of DPR with above result code SHOULD NOT attempt reconnection.

DO_NOT_WANT_TO_TALK_TO_YOU 2
The peer has determined that it does not see a need for the transport connection to exist, since it does not expect any messages to be exchanged in the near future. Receiver of DPR with above result code SHOULD NOT attempt reconnection.

5.5. Transport Failure Detection

Given the nature of the Diameter protocol, it is recommended that transport failures be detected as soon as possible. Detecting such failures will minimize the occurrence of messages sent to unavailable agents, resulting in unnecessary delays, and will provide better failover performance. The Device-Watchdog-Request and Device-Watchdog-Answer messages, defined in this section, are used to pro-actively detect transport failures.

5.5.1. Device-Watchdog-Request

The Device-Watchdog-Request (DWR), indicated by the Command-Code set to 280 and the Command Flags' 'R' bit set, is sent to a peer when no traffic has been exchanged between two peers (see Section 5.5.3). Upon detection of a transport failure, this message MUST NOT be sent to an alternate peer.

Message Format

```
<DWR>  ::= < Diameter Header: 280, REQ >
           { Origin-Host }
           { Origin-Realm }
           [ Origin-State-Id ]
```

```
                * [ AVP ]
```

5.5.2. Device-Watchdog-Answer

The Device-Watchdog-Answer (DWA), indicated by the Command-Code set to 280 and the Command Flags' 'R' bit cleared, is sent as a response to the Device-Watchdog-Request message.

Message Format

```
   <DWA>  ::= < Diameter Header: 280 >
              { Result-Code }
              { Origin-Host }
              { Origin-Realm }
              [ Error-Message ]
              [ Failed-AVP ]
              [ Origin-State-Id ]
            * [ AVP ]
```

5.5.3. Transport Failure Algorithm

The transport failure algorithm is defined in [RFC3539]. All Diameter implementations MUST support the algorithm defined in the specification in order to be compliant to the Diameter base protocol.

5.5.4. Failover and Failback Procedures

In the event that a transport failure is detected with a peer, it is necessary for all pending request messages to be forwarded to an alternate agent, if possible. This is commonly referred to as failover.

In order for a Diameter node to perform failover procedures, it is necessary for the node to maintain a pending message queue for a given peer. When an answer message is received, the corresponding request is removed from the queue. The Hop-by-Hop Identifier field is used to match the answer with the queued request.

When a transport failure is detected, if possible all messages in the queue are sent to an alternate agent with the T flag set. On booting a Diameter client or agent, the T flag is also set on any records still remaining to be transmitted in non-volatile storage. An example of a case where it is not possible to forward the message to an alternate server is when the message has a fixed destination, and the unavailable peer is the message's final destination (see Destination-Host AVP). Such an error requires that the agent return an answer message with the 'E' bit set and the Result-Code AVP set to DIAMETER_UNABLE_TO_DELIVER.

It is important to note that multiple identical requests or answers MAY be received as a result of a failover. The End-to-End Identifier field in the Diameter header along with the Origin-Host AVP MUST be used to identify duplicate messages.

As described in Section 2.1, a connection request should be periodically attempted with the failed peer in order to re-establish the transport connection. Once a connection has been successfully established, messages can once again be forwarded to the peer. This is commonly referred to as failback.

5.6. Peer State Machine

This section contains a finite state machine that MUST be observed by all Diameter implementations. Each Diameter node MUST follow the state machine described below when communicating with each peer. Multiple actions are separated by commas, and may continue on succeeding lines, as space requires. Similarly, state and next state may also span multiple lines, as space requires.

This state machine is closely coupled with the state machine described in [RFC3539], which is used to open, close, failover, probe, and reopen transport connections. Note in particular that [RFC3539] requires the use of watchdog messages to probe connections. For Diameter, DWR and DWA messages are to be used.

I- is used to represent the initiator (connecting) connection, while the R- is used to represent the responder (listening) connection. The lack of a prefix indicates that the event or action is the same regardless of the connection on which the event occurred.

The stable states that a state machine may be in are Closed, I-Open and R-Open; all other states are intermediate. Note that I-Open and R-Open are equivalent except for whether the initiator or responder transport connection is used for communication.

A CER message is always sent on the initiating connection immediately after the connection request is successfully completed. In the case of an election, one of the two connections will shut down. The responder connection will survive if the Origin-Host of the local Diameter entity is higher than that of the peer; the initiator connection will survive if the peer’s Origin-Host is higher. All subsequent messages are sent on the surviving connection. Note that the results of an election on one peer are guaranteed to be the inverse of the results on the other.

For TLS/TCP and DTLS/SCTP usage, TLS/TCP and DTLS/SCTP handshake SHOULD begin when both ends are in the closed state prior to any

Diameter message exchanges. The TLS/TCP and DTLS/SCTP connection SHOULD be established before sending any CER or CEA message to secure and protect the capabilities information of both peers. The TLS/TCP and DTLS/SCTP connection SHOULD be disconnected when the state machine moves to the closed state. When connecting to responders that do not conform to this document (i.e. older Diameter implementations that are not prepared to received TLS/TCP and DTLS/SCTP connections in the closed state), the initial TLS/TCP and DTLS/SCTP connection attempt will fail. The initiator MAY then attempt to connect via TCP or SCTP and initiate the TLS/TCP and DTLS/SCTP handshake when both ends are in the open state. If the handshake is successful, all further messages will be sent via TLS/TCP and DTLS/SCTP. If the handshake fails, both ends move to the closed state.

The state machine constrains only the behavior of a Diameter implementation as seen by Diameter peers through events on the wire.

Any implementation that produces equivalent results is considered compliant.

state	event	action	next state
Closed	Start	I-Snd-Conn-Req	Wait-Conn-Ack
	R-Conn-CER	R-Accept, Process-CER, R-Snd-CEA	R-Open
Wait-Conn-Ack	I-Rcv-Conn-Ack	I-Snd-CER	Wait-I-CEA
	I-Rcv-Conn-Nack	Cleanup	Closed
	R-Conn-CER	R-Accept, Process-CER	Wait-Conn-Ack/ Elect
	Timeout	Error	Closed
Wait-I-CEA	I-Rcv-CEA	Process-CEA	I-Open
	R-Conn-CER	R-Accept, Process-CER, Elect	Wait-Returns
	I-Peer-Disc	I-Disc	Closed
	I-Rcv-Non-CEA	Error	Closed
	Timeout	Error	Closed
Wait-Conn-Ack/ Elect	I-Rcv-Conn-Ack	I-Snd-CER,Elect	Wait-Returns
	I-Rcv-Conn-Nack	R-Snd-CEA	R-Open
	R-Peer-Disc	R-Disc	Wait-Conn-Ack
	R-Conn-CER	R-Reject	Wait-Conn-Ack/ Elect
	Timeout	Error	Closed

Wait-Returns	Win-Election	I-Disc,R-Snd-CEA	R-Open
	I-Peer-Disc	I-Disc, R-Snd-CEA	R-Open
	I-Rcv-CEA	R-Disc	I-Open
	R-Peer-Disc	R-Disc	Wait-I-CEA
	R-Conn-CER	R-Reject	Wait-Returns
	Timeout	Error	Closed
R-Open	Send-Message	R-Snd-Message	R-Open
	R-Rcv-Message	Process	R-Open
	R-Rcv-DWR	Process-DWR, R-Snd-DWA	R-Open
	R-Rcv-DWA	Process-DWA	R-Open
	R-Conn-CER	R-Reject	R-Open
	Stop	R-Snd-DPR	Closing
	R-Rcv-DPR	R-Snd-DPA, R-Disc	Closed
	R-Peer-Disc	R-Disc	Closed
I-Open	Send-Message	I-Snd-Message	I-Open
	I-Rcv-Message	Process	I-Open
	I-Rcv-DWR	Process-DWR, I-Snd-DWA	I-Open
	I-Rcv-DWA	Process-DWA	I-Open
	R-Conn-CER	R-Reject	I-Open
	Stop	I-Snd-DPR	Closing
	I-Rcv-DPR	I-Snd-DPA, I-Disc	Closed
	I-Peer-Disc	I-Disc	Closed
Closing	I-Rcv-DPA	I-Disc	Closed
	R-Rcv-DPA	R-Disc	Closed
	Timeout	Error	Closed
	I-Peer-Disc	I-Disc	Closed
	R-Peer-Disc	R-Disc	Closed

5.6.1. Incoming connections

When a connection request is received from a Diameter peer, it is not, in the general case, possible to know the identity of that peer until a CER is received from it. This is because host and port determine the identity of a Diameter peer; and the source port of an incoming connection is arbitrary. Upon receipt of CER, the identity of the connecting peer can be uniquely determined from Origin-Host.

For this reason, a Diameter peer must employ logic separate from the state machine to receive connection requests, accept them, and await CER. Once CER arrives on a new connection, the Origin-Host that

identifies the peer is used to locate the state machine associated with that peer, and the new connection and CER are passed to the state machine as an R-Conn-CER event.

The logic that handles incoming connections SHOULD close and discard the connection if any message other than CER arrives, or if an implementation-defined timeout occurs prior to receipt of CER.

Because handling of incoming connections up to and including receipt of CER requires logic, separate from that of any individual state machine associated with a particular peer, it is described separately in this section rather than in the state machine above.

5.6.2. Events

Transitions and actions in the automaton are caused by events. In this section, we will ignore the -I and -R prefix, since the actual event would be identical, but would occur on one of two possible connections.

Start	The Diameter application has signaled that a connection should be initiated with the peer.
R-Conn-CER	An acknowledgement is received stating that the transport connection has been established, and the associated CER has arrived.
Rcv-Conn-Ack	A positive acknowledgement is received confirming that the transport connection is established.
Rcv-Conn-Nack	A negative acknowledgement was received stating that the transport connection was not established.
Timeout	An application-defined timer has expired while waiting for some event.
Rcv-CER	A CER message from the peer was received.
Rcv-CEA	A CEA message from the peer was received.
Rcv-Non-CEA	A message other than CEA from the peer was received.
Peer-Disc	A disconnection indication from the peer was received.
Rcv-DPR	A DPR message from the peer was received.
Rcv-DPA	A DPA message from the peer was received.
Win-Election	An election was held, and the local node was the winner.
Send-Message	A message is to be sent.
Rcv-Message	A message other than CER, CEA, DPR, DPA, DWR or DWA was received.
Stop	The Diameter application has signaled that a connection should be terminated (e.g., on system shutdown).

5.6.3. Actions

Actions in the automaton are caused by events and typically indicate the transmission of packets and/or an action to be taken on the connection. In this section we will ignore the I- and R-prefix, since the actual action would be identical, but would occur on one of two possible connections.

Snd-Conn-Req	A transport connection is initiated with the peer.
Accept	The incoming connection associated with the R-Conn-CER is accepted as the responder connection.
Reject	The incoming connection associated with the R-Conn-CER is disconnected.
Process-CER	The CER associated with the R-Conn-CER is processed.
Snd-CER	A CER message is sent to the peer.
Snd-CEA	A CEA message is sent to the peer.
Cleanup	If necessary, the connection is shutdown, and any local resources are freed.
Error	The transport layer connection is disconnected, either politely or abortively, in response to an error condition. Local resources are freed.
Process-CEA	A received CEA is processed.
Snd-DPR	A DPR message is sent to the peer.
Snd-DPA	A DPA message is sent to the peer.
Disc	The transport layer connection is disconnected, and local resources are freed.
Elect	An election occurs (see Section 5.6.4 for more information).
Snd-Message	A message is sent.
Snd-DWR	A DWR message is sent.
Snd-DWA	A DWA message is sent.
Process-DWR	The DWR message is serviced.
Process-DWA	The DWA message is serviced.
Process	A message is serviced.

5.6.4. The Election Process

The election is performed on the responder. The responder compares the Origin-Host received in the CER with its own Origin-Host as two streams of octets. If the local Origin-Host lexicographically succeeds the received Origin-Host a Win-Election event is issued locally. Diameter identities are in ASCII form therefore the lexical comparison is consistent with DNS case insensitivity where octets that fall in the ASCII range 'a' through 'z' MUST compare equally to their upper-case counterparts between 'A' and 'Z'. See Appendix D for interactions between the Diameter protocol and Internationalized Domain Name (IDNs).

The winner of the election MUST close the connection it initiated. Historically, maintaining the responder side of a connection was more efficient than maintaining the initiator side. However, current practices makes this distinction irrelevant.

6. Diameter Message Processing

This section describes how Diameter requests and answers are created and processed.

6.1. Diameter Request Routing Overview

A request is sent towards its final destination using a combination of the Destination-Realm and Destination-Host AVPs, in one of these three combinations:

- a request that is not able to be proxied (such as CER) MUST NOT contain either Destination-Realm or Destination-Host AVPs.
- a request that needs to be sent to a home server serving a specific realm, but not to a specific server (such as the first request of a series of round-trips), MUST contain a Destination-Realm AVP, but MUST NOT contain a Destination-Host AVP. For Diameter clients, the value of the Destination-Realm AVP MAY be extracted from the User-Name AVP, or other methods.
- otherwise, a request that needs to be sent to a specific home server among those serving a given realm, MUST contain both the Destination-Realm and Destination-Host AVPs.

The Destination-Host AVP is used as described above when the destination of the request is fixed, which includes:

- Authentication requests that span multiple round trips
- A Diameter message that uses a security mechanism that makes use of a pre-established session key shared between the source and the final destination of the message.
- Server initiated messages that MUST be received by a specific Diameter client (e.g., access device), such as the Abort-Session-Request message, which is used to request that a particular user's session be terminated.

Note that an agent can forward a request to a host described in the Destination-Host AVP only if the host in question is included in its peer table (see Section 2.6). Otherwise, the request is routed based on the Destination-Realm only (see Section 6.1.6).

When a message is received, the message is processed in the following order:

- If the message is destined for the local host, the procedures listed in Section 6.1.4 are followed.
- If the message is intended for a Diameter peer with whom the local host is able to directly communicate, the procedures listed in Section 6.1.5 are followed. This is known as Request Forwarding.
- The procedures listed in Section 6.1.6 are followed, which is known as Request Routing.
- If none of the above is successful, an answer is returned with the Result-Code set to DIAMETER_UNABLE_TO_DELIVER, with the 'E' bit set.

For routing of Diameter messages to work within an administrative domain, all Diameter nodes within the realm MUST be peers.

The overview contained in this section (6.1) is intended to provide general guidelines to Diameter developers. Implementations are free to use different methods than the ones described here as long as they conform to the requirements specified in Sections 6.1.1 through 6.1.9. See Section 7 for more detail on error handling.

6.1.1. Originating a Request

When creating a request, in addition to any other procedures described in the application definition for that specific request, the following procedures MUST be followed:

- the Command-Code is set to the appropriate value
- the 'R' bit is set
- the End-to-End Identifier is set to a locally unique value
- the Origin-Host and Origin-Realm AVPs MUST be set to the appropriate values, used to identify the source of the message
- the Destination-Host and Destination-Realm AVPs MUST be set to the appropriate values as described in Section 6.1.

6.1.2. Sending a Request

When sending a request, originated either locally, or as the result of a forwarding or routing operation, the following procedures SHOULD be followed:

- The Hop-by-Hop Identifier SHOULD be set to a locally unique value.
- The message SHOULD be saved in the list of pending requests.

Other actions to perform on the message based on the particular role the agent is playing are described in the following sections.

6.1.3. Receiving Requests

A relay or proxy agent MUST check for forwarding loops when receiving requests. A loop is detected if the server finds its own identity in a Route-Record AVP. When such an event occurs, the agent MUST answer with the Result-Code AVP set to DIAMETER_LOOP_DETECTED.

6.1.4. Processing Local Requests

A request is known to be for local consumption when one of the following conditions occur:

- The Destination-Host AVP contains the local host's identity,
- The Destination-Host AVP is not present, the Destination-Realm AVP contains a realm the server is configured to process locally, and the Diameter application is locally supported, or
- Both the Destination-Host and the Destination-Realm are not present.

When a request is locally processed, the rules in Section 6.2 should be used to generate the corresponding answer.

6.1.5. Request Forwarding

Request forwarding is done using the Diameter Peer Table. The Diameter peer table contains all of the peers that the local node is able to directly communicate with.

When a request is received, and the host encoded in the Destination-Host AVP is one that is present in the peer table, the message SHOULD be forwarded to the peer.

6.1.6. Request Routing

Diameter request message routing is done via realms and application identifiers. A Diameter message that may be forwarded by Diameter agents (proxies, redirect or relay agents) MUST include the target realm in the Destination-Realm AVP. Request routing SHOULD rely on the Destination-Realm AVP and the Application Id present in the request message header to aid in the routing decision. The realm MAY be retrieved from the User-Name AVP, which is in the form of a Network Access Identifier (NAI). The realm portion of the NAI is inserted in the Destination-Realm AVP.

Diameter agents MAY have a list of locally supported realms and applications, and MAY have a list of externally supported realms and applications. When a request is received that includes a realm and/or application that is not locally supported, the message is routed to the peer configured in the Routing Table (see Section 2.7).

Realm names and Application Ids are the minimum supported routing criteria, additional information may be needed to support redirect semantics.

6.1.7. Predictive Loop Avoidance

Before forwarding or routing a request Diameter agents, in addition to performing the processing described in Section 6.1.3, SHOULD check for the presence of candidate route's peer identity in any of the Route-Record AVPs. In an event of the agent detecting the presence of a candidate route's peer identity in a Route-Record AVP, the agent MUST ignore such route for the Diameter request message and attempt alternate routes if any. In case all the candidate routes are eliminated by the above criteria, the agent SHOULD return DIAMETER_UNABLE_TO_DELIVER message.

6.1.8. Redirecting Requests

When a redirect agent receives a request whose routing entry is set to REDIRECT, it MUST reply with an answer message with the 'E' bit

set, while maintaining the Hop-by-Hop Identifier in the header, and include the Result-Code AVP to DIAMETER_REDIRECT_INDICATION. Each of the servers associated with the routing entry are added in separate Redirect-Host AVP.

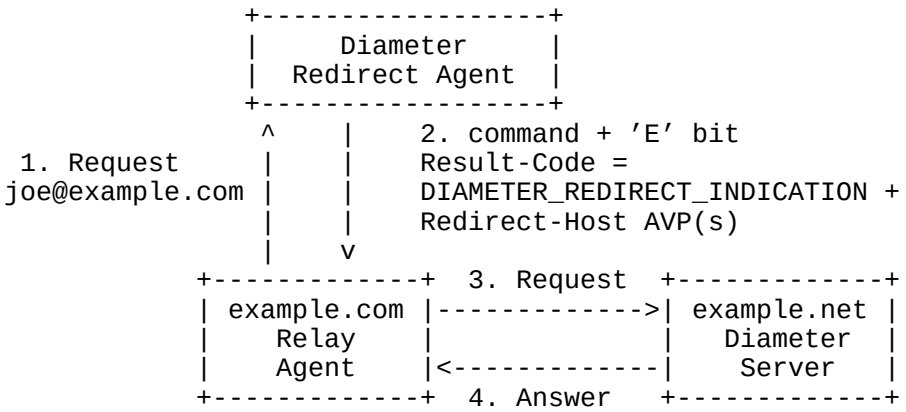


Figure 5: Diameter Redirect Agent

The receiver of an answer message with the 'E' bit set and the Result-Code AVP set to DIAMETER_REDIRECT_INDICATION uses the Hop-by-Hop Identifier in the Diameter header to identify the request in the pending message queue (see Section 5.5.4) that is to be redirected. If no transport connection exists with the new agent, one is created, and the request is sent directly to it.

Multiple Redirect-Host AVPs are allowed. The receiver of the answer message with the 'E' bit set selects exactly one of these hosts as the destination of the redirected message.

When the Redirect-Host-Usage AVP included in the answer message has a non-zero value, a route entry for the redirect indications is created and cached by the receiver. The redirect usage for such route entry is set by the value of Redirect-Host-Usage AVP and the lifetime of the cached route entry is set by Redirect-Max-Cache-Time AVP value.

It is possible that multiple redirect indications can create multiple cached route entries differing only in their redirect usage and the peer to forward messages to. As an example, two(2) route entries that are created by two(2) redirect indications results in two(2) cached routes for the same realm and Application Id. However, one has a redirect usage of ALL_SESSION where matching request will be forwarded to one peer and the other has a redirect usage of ALL_REALM where request are forwarded to another peer. Therefore, an incoming request that matches the realm and Application Id of both routes will

need additional resolution. In such a case, a routing precedence rule MUST be used against the redirect usage value to resolve the contention. The precedence rule can be found in Section 6.13.

6.1.9. Relaying and Proxying Requests

A relay or proxy agent MUST append a Route-Record AVP to all requests forwarded. The AVP contains the identity of the peer the request was received from.

The Hop-by-Hop identifier in the request is saved, and replaced with a locally unique value. The source of the request is also saved, which includes the IP address, port and protocol.

A relay or proxy agent MAY include the Proxy-Info AVP in requests if it requires access to any local state information when the corresponding response is received. The Proxy-Info AVP has security implications as state information is distributed to other entities. As such, it is RECOMMENDED that the content of the Proxy-Info AVP be protected with cryptographic mechanisms, for example by using a keyed message digest such as HMAC-SHA1 [RFC2104]. Such a mechanism, however, requires the management of keys, although only locally at the Diameter server. Still, a full description of the management of the keys used to protect the Proxy-Info AVP is beyond the scope of this document. Below is a list of common recommendations:

- The keys should be generated securely following the randomness recommendations in [RFC4086].
- The keys and cryptographic protection algorithms should be at least 128 bits in strength.
- The keys should not be used for any other purpose than generating and verifying tickets.
- The keys should be changed regularly.
- The keys should be changed if the ticket format or cryptographic protection algorithms change.

The message is then forwarded to the next hop, as identified in the Routing Table.

Figure 6 provides an example of message routing using the procedures listed in these sections.

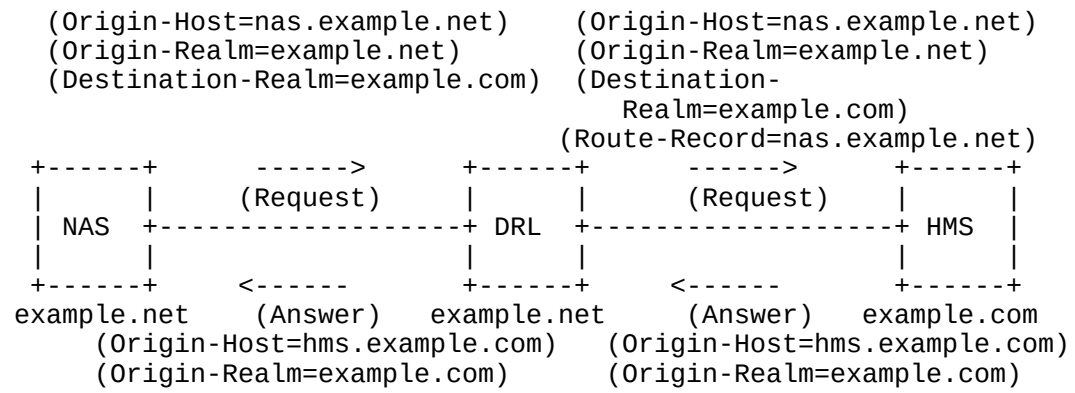


Figure 6: Routing of Diameter messages

Relay and proxy agents are not required to perform full inspection of incoming messages. At a minimum, validation of the message header and relevant routing AVPs has to be done when relaying messages. Proxy agents may optionally perform more in-depth message validation for applications it is interested in.

6.2. Diameter Answer Processing

When a request is locally processed, the following procedures MUST be applied to create the associated answer, in addition to any additional procedures that MAY be discussed in the Diameter application defining the command:

- The same Hop-by-Hop identifier in the request is used in the answer.
- The local host’s identity is encoded in the Origin-Host AVP.
- The Destination-Host and Destination-Realm AVPs MUST NOT be present in the answer message.
- The Result-Code AVP is added with its value indicating success or failure.
- If the Session-Id is present in the request, it MUST be included in the answer.
- Any Proxy-Info AVPs in the request MUST be added to the answer message, in the same order they were present in the request.
- The ’P’ bit is set to the same value as the one in the request.

- The same End-to-End identifier in the request is used in the answer.

Note that the error messages (see Section 7) are also subjected to the above processing rules.

6.2.1. Processing Received Answers

A Diameter client or proxy MUST match the Hop-by-Hop Identifier in an answer received against the list of pending requests. The corresponding message should be removed from the list of pending requests. It SHOULD ignore answers received that do not match a known Hop-by-Hop Identifier.

6.2.2. Relaying and Proxying Answers

If the answer is for a request which was proxied or relayed, the agent MUST restore the original value of the Diameter header's Hop-by-Hop Identifier field.

If the last Proxy-Info AVP in the message is targeted to the local Diameter server, the AVP MUST be removed before the answer is forwarded.

If a relay or proxy agent receives an answer with a Result-Code AVP indicating a failure, it MUST NOT modify the contents of the AVP. Any additional local errors detected SHOULD be logged, but not reflected in the Result-Code AVP. If the agent receives an answer message with a Result-Code AVP indicating success, and it wishes to modify the AVP to indicate an error, it MUST modify the Result-Code AVP to contain the appropriate error in the message destined towards the access device as well as include the Error-Reporting-Host AVP and it MUST issue an STR on behalf of the access device towards the Diameter server.

The agent MUST then send the answer to the host that it received the original request from.

6.3. Origin-Host AVP

The Origin-Host AVP (AVP Code 264) is of type DiameterIdentity, and MUST be present in all Diameter messages. This AVP identifies the endpoint that originated the Diameter message. Relay agents MUST NOT modify this AVP.

The value of the Origin-Host AVP is guaranteed to be unique within a single host.

Note that the Origin-Host AVP may resolve to more than one address as the Diameter peer may support more than one address.

This AVP SHOULD be placed as close to the Diameter header as possible.

6.4. Origin-Realm AVP

The Origin-Realm AVP (AVP Code 296) is of type DiameterIdentity. This AVP contains the Realm of the originator of any Diameter message and MUST be present in all messages.

This AVP SHOULD be placed as close to the Diameter header as possible.

6.5. Destination-Host AVP

The Destination-Host AVP (AVP Code 293) is of type DiameterIdentity. This AVP MUST be present in all unsolicited agent initiated messages, MAY be present in request messages, and MUST NOT be present in Answer messages.

The absence of the Destination-Host AVP will cause a message to be sent to any Diameter server supporting the application within the realm specified in Destination-Realm AVP.

This AVP SHOULD be placed as close to the Diameter header as possible.

6.6. Destination-Realm AVP

The Destination-Realm AVP (AVP Code 283) is of type DiameterIdentity, and contains the realm the message is to be routed to. The Destination-Realm AVP MUST NOT be present in Answer messages. Diameter Clients insert the realm portion of the User-Name AVP. Diameter servers initiating a request message use the value of the Origin-Realm AVP from a previous message received from the intended target host (unless it is known a priori). When present, the Destination-Realm AVP is used to perform message routing decisions.

The CCF for a request message that includes the Destination-Realm AVP SHOULD list the Destination-Realm AVP as a required AVP (an AVP indicated as {AVP}) otherwise the message is inherently a non-routable message.

This AVP SHOULD be placed as close to the Diameter header as possible.

6.7. Routing AVPs

The AVPs defined in this section are Diameter AVPs used for routing purposes. These AVPs change as Diameter messages are processed by agents.

6.7.1. Route-Record AVP

The Route-Record AVP (AVP Code 282) is of type DiameterIdentity. The identity added in this AVP MUST be the same as the one received in the Origin-Host of the Capabilities Exchange message.

6.7.2. Proxy-Info AVP

The Proxy-Info AVP (AVP Code 284) is of type Grouped. This AVP contains the identity and local state information of the Diameter node that creates and adds it to a message. The Grouped Data field has the following CCF grammar:

```
Proxy-Info ::= < AVP Header: 284 >
               { Proxy-Host }
               { Proxy-State }
             * [ AVP ]
```

6.7.3. Proxy-Host AVP

The Proxy-Host AVP (AVP Code 280) is of type DiameterIdentity. This AVP contains the identity of the host that added the Proxy-Info AVP.

6.7.4. Proxy-State AVP

The Proxy-State AVP (AVP Code 33) is of type OctetString. It contains state information that would otherwise be stored at the Diameter entity that created it. As such, this AVP MUST be treated as opaque data by other Diameter entities.

6.8. Auth-Application-Id AVP

The Auth-Application-Id AVP (AVP Code 258) is of type Unsigned32 and is used in order to advertise support of the Authentication and Authorization portion of an application (see Section 2.4). If present in a message other than CER and CEA, the value of the Auth-Application-Id AVP MUST match the Application Id present in the Diameter message header.

6.9. Acct-Application-Id AVP

The Acct-Application-Id AVP (AVP Code 259) is of type Unsigned32 and is used in order to advertise support of the Accounting portion of an application (see Section 2.4). If present in a message other than CER and CEA, the value of the Acct-Application-Id AVP MUST match the Application Id present in the Diameter message header.

6.10. Inband-Security-Id AVP

The Inband-Security-Id AVP (AVP Code 299) is of type Unsigned32 and is used in order to advertise support of the security portion of the application. The use of this AVP in CER and CEA messages is NOT RECCOMENDED. Instead, discovery of a Diameter entities security capabilities can be done either through static configuration or via Diameter Peer Discovery as described in Section 5.2.

The following values are supported:

NO_INBAND_SECURITY 0

This peer does not support TLS/TCP and DTLS/SCTP. This is the default value, if the AVP is omitted.

TLS 1

This node supports TLS/TCP [RFC5246] and DTLS/SCTP [RFC6083] security.

6.11. Vendor-Specific-Application-Id AVP

The Vendor-Specific-Application-Id AVP (AVP Code 260) is of type Grouped and is used to advertise support of a vendor-specific Diameter Application. Exactly one instance of either Auth-Application-Id or Acct-Application-Id AVP MUST be present. The Application Id carried by either Auth-Application-Id or Acct-Application-Id AVP MUST comply with vendor specific Application Id assignment described in Sec 11.3. It MUST also match the Application Id present in the Diameter header except when used in a CER or CEA message.

The Vendor-Id AVP is an informational AVP pertaining to the vendor who may have authorship of the vendor-specific Diameter application. It MUST NOT be used as a means of defining a completely separate vendor-specific Application Id space.

The Vendor-Specific-Application-Id AVP SHOULD be placed as close to

the Diameter header as possible.

AVP Format

```
<Vendor-Specific-Application-Id> ::= < AVP Header: 260 >
                                     { Vendor-Id }
                                     [ Auth-Application-Id ]
                                     [ Acct-Application-Id ]
```

A Vendor-Specific-Application-Id AVP MUST contain exactly one of either Auth-Application-Id or Acct-Application-Id. If a Vendor-Specific-Application-Id is received without any of these two AVPs, then the recipient SHOULD issue an answer with a Result-Code set to DIAMETER_MISSING_AVP. The answer SHOULD also include a Failed-AVP which MUST contain an example of an Auth-Application-Id AVP and an Acct-Application-Id AVP.

If a Vendor-Specific-Application-Id is received that contains both Auth-Application-Id and Acct-Application-Id, then the recipient MUST issue an answer with Result-Code set to DIAMETER_AVP_OCCURS_TOO_MANY_TIMES. The answer MUST also include a Failed-AVP which MUST contain the received Auth-Application-Id AVP and Acct-Application-Id AVP.

6.12. Redirect-Host AVP

The Redirect-Host AVP (AVP Code 292) is of type DiameterURI. One or more of instances of this AVP MUST be present if the answer message's 'E' bit is set and the Result-Code AVP is set to DIAMETER_REDIRECT_INDICATION.

Upon receiving the above, the receiving Diameter node SHOULD forward the request directly to one of the hosts identified in these AVPs. The server contained in the selected Redirect-Host AVP SHOULD be used for all messages matching the criteria set by the Redirect-Host-Usage AVP.

6.13. Redirect-Host-Usage AVP

The Redirect-Host-Usage AVP (AVP Code 261) is of type Enumerated. This AVP MAY be present in answer messages whose 'E' bit is set and the Result-Code AVP is set to DIAMETER_REDIRECT_INDICATION.

When present, this AVP provides a hints about how the routing entry resulting from the Redirect-Host is to be used. The following values are supported:

DONT_CACHE 0

The host specified in the Redirect-Host AVP SHOULD NOT be cached. This is the default value.

ALL_SESSION 1

All messages within the same session, as defined by the same value of the Session-ID AVP SHOULD be sent to the host specified in the Redirect-Host AVP.

ALL_REALM 2

All messages destined for the realm requested SHOULD be sent to the host specified in the Redirect-Host AVP.

REALM_AND_APPLICATION 3

All messages for the application requested to the realm specified SHOULD be sent to the host specified in the Redirect-Host AVP.

ALL_APPLICATION 4

All messages for the application requested SHOULD be sent to the host specified in the Redirect-Host AVP.

ALL_HOST 5

All messages that would be sent to the host that generated the Redirect-Host SHOULD be sent to the host specified in the Redirect-Host AVP.

ALL_USER 6

All messages for the user requested SHOULD be sent to the host specified in the Redirect-Host AVP.

When multiple cached routes are created by redirect indications and they differ only in redirect usage and peers to forward requests to (see Section 6.1.8, a precedence rule MUST be applied to the redirect usage values of the cached routes during normal routing to resolve contentions that may occur. The precedence rule is the order that

dictate which redirect usage should be considered before any other as they appear. The order is as follows:

1. ALL_SESSION
2. ALL_USER
3. REALM_AND_APPLICATION
4. ALL_REALM
5. ALL_APPLICATION
6. ALL_HOST

6.14. Redirect-Max-Cache-Time AVP

The Redirect-Max-Cache-Time AVP (AVP Code 262) is of type Unsigned32. This AVP MUST be present in answer messages whose 'E' bit is set, the Result-Code AVP is set to DIAMETER_REDIRECT_INDICATION and the Redirect-Host-Usage AVP set to a non-zero value.

This AVP contains the maximum number of seconds the peer and route table entries, created as a result of the Redirect-Host, SHOULD be cached. Note that once a host is no longer reachable, any associated cache, peer and routing table entries MUST be deleted.

7. Error Handling

There are two different types of errors in Diameter; protocol and application errors. A protocol error is one that occurs at the base protocol level, and MAY require per hop attention (e.g., message routing error). Application errors, on the other hand, generally occur due to a problem with a function specified in a Diameter application (e.g., user authentication, missing AVP).

Result-Code AVP values that are used to report protocol errors MUST only be present in answer messages whose 'E' bit is set. When a request message is received that causes a protocol error, an answer message is returned with the 'E' bit set, and the Result-Code AVP is set to the appropriate protocol error value. As the answer is sent back towards the originator of the request, each proxy or relay agent MAY take action on the message.

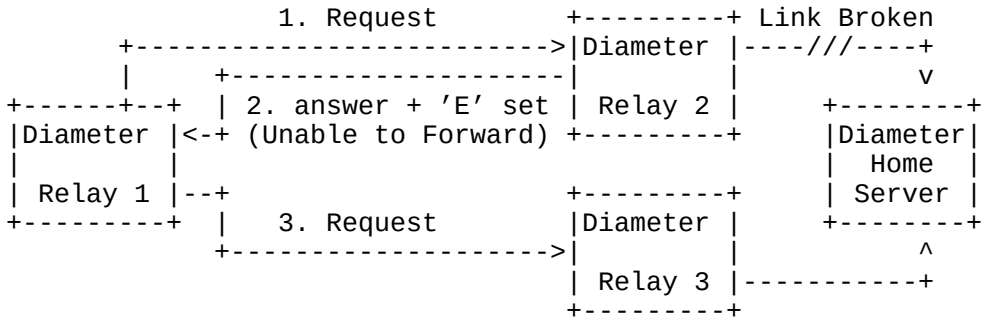


Figure 7: Example of Protocol Error causing answer message

Figure 7 provides an example of a message forwarded upstream by a Diameter relay. When the message is received by Relay 2, and it detects that it cannot forward the request to the home server, an answer message is returned with the 'E' bit set and the Result-Code AVP set to DIAMETER_UNABLE_TO_DELIVER. Given that this error falls within the protocol error category, Relay 1 would take special action, and given the error, attempt to route the message through its alternate Relay 3.

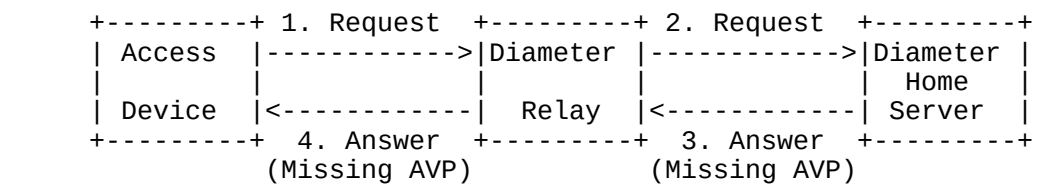


Figure 8: Example of Application Error Answer message

Figure 8 provides an example of a Diameter message that caused an application error. When application errors occur, the Diameter entity reporting the error clears the 'R' bit in the Command Flags, and adds the Result-Code AVP with the proper value. Application errors do not require any proxy or relay agent involvement, and therefore the message would be forwarded back to the originator of the request.

In the case where the answer message itself contains errors, any related session SHOULD be terminated by sending an STR or ASR message. The Termination-Cause AVP in the STR MAY be filled with the appropriate value to indicate the cause of the error. An application MAY also send an application-specific request instead of STR or ASR to signal the error in the case where no state is maintained or to allow for some form of error recovery with the corresponding Diameter entity.

There are certain Result-Code AVP application errors that require additional AVPs to be present in the answer. In these cases, the Diameter node that sets the Result-Code AVP to indicate the error MUST add the AVPs. Examples are:

- A request with an unrecognized AVP is received with the 'M' bit (Mandatory bit) set, causes an answer to be sent with the Result-Code AVP set to DIAMETER_AVP_UNSUPPORTED, and the Failed-AVP AVP containing the offending AVP.

- A request with an AVP that is received with an unrecognized value causes an answer to be returned with the Result-Code AVP set to DIAMETER_INVALID_AVP_VALUE, with the Failed-AVP AVP containing the AVP causing the error.

- A received command which is missing AVP(s) that are defined as required in the commands CCF; examples are AVPs indicated as {AVP}. The receiver issues an answer with the Result-Code set to DIAMETER_MISSING_AVP, and creates an AVP with the AVP Code and other fields set as expected in the missing AVP. The created AVP is then added to the Failed-AVP AVP.

The Result-Code AVP describes the error that the Diameter node encountered in its processing. In case there are multiple errors, the Diameter node MUST report only the first error it encountered (detected possibly in some implementation dependent order). The specific errors that can be described by this AVP are described in the following section.

7.1. Result-Code AVP

The Result-Code AVP (AVP Code 268) is of type Unsigned32 and indicates whether a particular request was completed successfully or whether an error occurred. All Diameter answer messages in IETF defined Diameter application specification MUST include one Result-Code AVP. A non-successful Result-Code AVP (one containing a non 2xxx value other than DIAMETER_REDIRECT_INDICATION) MUST include the Error-Reporting-Host AVP if the host setting the Result-Code AVP is different from the identity encoded in the Origin-Host AVP.

The Result-Code data field contains an IANA-managed 32-bit address space representing errors (see Section 11.3.2). Diameter provides the following classes of errors, all identified by the thousands digit in the decimal notation:

- o 1xxx (Informational)
- o 2xxx (Success)
- o 3xxx (Protocol Errors)
- o 4xxx (Transient Failures)
- o 5xxx (Permanent Failure)

A non-recognized class (one whose first digit is not defined in this section) MUST be handled as a permanent failure.

7.1.1. Informational

Errors that fall within this category are used to inform the requester that a request could not be satisfied, and additional action is required on its part before access is granted.

DIAMETER_MULTI_ROUND_AUTH 1001

This informational error is returned by a Diameter server to inform the access device that the authentication mechanism being used requires multiple round trips, and a subsequent request needs to be issued in order for access to be granted.

7.1.2. Success

Errors that fall within the Success category are used to inform a peer that a request has been successfully completed.

DIAMETER_SUCCESS 2001

The request was successfully completed.

DIAMETER_LIMITED_SUCCESS 2002

When returned, the request was successfully completed, but additional processing is required by the application in order to provide service to the user.

7.1.3. Protocol Errors

Errors that fall within the Protocol Error category SHOULD be treated on a per-hop basis, and Diameter proxies MAY attempt to correct the error, if it is possible. Note that these errors MUST only be used

in answer messages whose 'E' bit is set.

DIAMETER_COMMAND_UNSUPPORTED 3001

This error code is used when a Diameter entity receives a message with a Command Code that it does not support.

DIAMETER_UNABLE_TO_DELIVER 3002

This error is given when Diameter can not deliver the message to the destination, either because no host within the realm supporting the required application was available to process the request, or because Destination-Host AVP was given without the associated Destination-Realm AVP.

DIAMETER_REALM_NOT_SERVED 3003

The intended realm of the request is not recognized.

DIAMETER_TOO_BUSY 3004

When returned, a Diameter node SHOULD attempt to send the message to an alternate peer. This error MUST only be used when a specific server is requested, and it cannot provide the requested service.

DIAMETER_LOOP_DETECTED 3005

An agent detected a loop while trying to get the message to the intended recipient. The message MAY be sent to an alternate peer, if one is available, but the peer reporting the error has identified a configuration problem.

DIAMETER_REDIRECT_INDICATION 3006

A redirect agent has determined that the request could not be satisfied locally and the initiator of the request SHOULD direct the request directly to the server, whose contact information has been added to the response. When set, the Redirect-Host AVP MUST be present.

DIAMETER_APPLICATION_UNSUPPORTED 3007

A request was sent for an application that is not supported.

DIAMETER_INVALID_HDR_BITS 3008

A request was received whose bits in the Diameter header were either set to an invalid combination, or to a value that is inconsistent with the command code's definition.

DIAMETER_INVALID_AVP_BITS 3009

A request was received that included an AVP whose flag bits are set to an unrecognized value, or that is inconsistent with the AVP's definition.

DIAMETER_UNKNOWN_PEER 3010

A CER was received from an unknown peer.

7.1.4. Transient Failures

Errors that fall within the transient failures category are used to inform a peer that the request could not be satisfied at the time it was received, but MAY be able to satisfy the request in the future. Note that these errors MUST be used in answer messages whose 'E' bit is not set.

DIAMETER_AUTHENTICATION_REJECTED 4001

The authentication process for the user failed, most likely due to an invalid password used by the user. Further attempts MUST only be tried after prompting the user for a new password.

DIAMETER_OUT_OF_SPACE 4002

A Diameter node received the accounting request but was unable to commit it to stable storage due to a temporary lack of space.

ELECTION_LOST 4003

The peer has determined that it has lost the election process and has therefore disconnected the transport connection.

7.1.5. Permanent Failures

Errors that fall within the permanent failures category are used to inform the peer that the request failed, and should not be attempted again. Note that these errors SHOULD be used in answer messages whose 'E' bit is not set. In error conditions where it is not possible or efficient to compose application-specific answer grammar then answer messages with E-bit set and complying to the grammar described in 7.2 MAY also be used for permanent errors.

DIAMETER_AVP_UNSUPPORTED 5001

The peer received a message that contained an AVP that is not recognized or supported and was marked with the Mandatory bit. A Diameter message with this error MUST contain one or more Failed-AVP AVP containing the AVPs that caused the failure.

DIAMETER_UNKNOWN_SESSION_ID 5002

The request contained an unknown Session-Id.

DIAMETER_AUTHORIZATION_REJECTED 5003

A request was received for which the user could not be authorized. This error could occur if the service requested is not permitted to the user.

DIAMETER_INVALID_AVP_VALUE 5004

The request contained an AVP with an invalid value in its data portion. A Diameter message indicating this error MUST include the offending AVPs within a Failed-AVP AVP.

DIAMETER_MISSING_AVP 5005

The request did not contain an AVP that is required by the Command Code definition. If this value is sent in the Result-Code AVP, a Failed-AVP AVP SHOULD be included in the message. The Failed-AVP

AVP MUST contain an example of the missing AVP complete with the Vendor-Id if applicable. The value field of the missing AVP should be of correct minimum length and contain zeroes.

DIAMETER_RESOURCES_EXCEEDED 5006

A request was received that cannot be authorized because the user has already expended allowed resources. An example of this error condition is a user that is restricted to one dial-up PPP port, attempts to establish a second PPP connection.

DIAMETER_CONTRADICTING_AVPS 5007

The Home Diameter server has detected AVPs in the request that contradicted each other, and is not willing to provide service to the user. The Failed-AVP AVPs MUST be present which contains the AVPs that contradicted each other.

DIAMETER_AVP_NOT_ALLOWED 5008

A message was received with an AVP that MUST NOT be present. The Failed-AVP AVP MUST be included and contain a copy of the offending AVP.

DIAMETER_AVP_OCCURS_TOO_MANY_TIMES 5009

A message was received that included an AVP that appeared more often than permitted in the message definition. The Failed-AVP AVP MUST be included and contain a copy of the first instance of the offending AVP that exceeded the maximum number of occurrences

DIAMETER_NO_COMMON_APPLICATION 5010

This error is returned by a Diameter node that receives a CER whereby no applications are common between the CER sending peer and the CER receiving peer.

DIAMETER_UNSUPPORTED_VERSION 5011

This error is returned when a request was received, whose version number is unsupported.

DIAMETER_UNABLE_TO_COMPLY 5012

This error is returned when a request is rejected for unspecified reasons.

DIAMETER_INVALID_BIT_IN_HEADER 5013

This error is returned when a reserved bit in the Diameter header is set to one (1) or the bits in the Diameter header are set incorrectly.

DIAMETER_INVALID_AVP_LENGTH 5014

The request contained an AVP with an invalid length. A Diameter message indicating this error MUST include the offending AVPs within a Failed-AVP AVP. In cases where the erroneous AVP length value exceeds the message length or is less than the minimum AVP header length, it is sufficient to include the offending AVP header and a zero filled payload of the minimum required length for the payloads data type. If the AVP is a grouped AVP, the grouped AVP header with an empty payload would be sufficient to indicate the offending AVP. In the case where the offending AVP header cannot be fully decoded when the AVP length is less than the minimum AVP header length, it is sufficient to include an offending AVP header that is formulated by padding the incomplete AVP header with zero up to the minimum AVP header length.

DIAMETER_INVALID_MESSAGE_LENGTH 5015

This error is returned when a request is received with an invalid message length.

DIAMETER_INVALID_AVP_BIT_COMBO 5016

The request contained an AVP with which is not allowed to have the given value in the AVP Flags field. A Diameter message indicating this error MUST include the offending AVPs within a Failed-AVP AVP.

DIAMETER_NO_COMMON_SECURITY 5017

This error is returned when a CER message is received, and there are no common security mechanisms supported between the peers. A

Capabilities-Exchange-Answer (CEA) MUST be returned with the Result-Code AVP set to DIAMETER_NO_COMMON_SECURITY.

7.2. Error Bit

The ’E’ (Error Bit) in the Diameter header is set when the request caused a protocol-related error (see Section 7.1.3). A message with the ’E’ bit MUST NOT be sent as a response to an answer message. Note that a message with the ’E’ bit set is still subjected to the processing rules defined in Section 6.2. When set, the answer message will not conform to the CCF specification for the command, and will instead conform to the following CCF:

Message Format

```
<answer-message> ::= < Diameter Header: code, ERR [, PXY] >
                    0*1< Session-Id >
                       { Origin-Host }
                       { Origin-Realm }
                       { Result-Code }
                       [ Origin-State-Id ]
                       [ Error-Message ]
                       [ Error-Reporting-Host ]
                       [ Failed-AVP ]
                       [ Experimental-Result ]
                     * [ Proxy-Info ]
                     * [ AVP ]
```

Note that the code used in the header is the same than the one found in the request message, but with the ’R’ bit cleared and the ’E’ bit set. The ’P’ bit in the header is set to the same value as the one found in the request message.

7.3. Error-Message AVP

The Error-Message AVP (AVP Code 281) is of type UTF8String. It MAY accompany a Result-Code AVP as a human readable error message. The Error-Message AVP is not intended to be useful in an environment where error messages are processed automatically. It SHOULD NOT be expected that the content of this AVP is parsed by network entities.

7.4. Error-Reporting-Host AVP

The Error-Reporting-Host AVP (AVP Code 294) is of type DiameterIdentity. This AVP contains the identity of the Diameter host that sent the Result-Code AVP to a value other than 2001 (Success), only if the host setting the Result-Code is different from the one encoded in the Origin-Host AVP. This AVP is intended to be

used for troubleshooting purposes, and MUST be set when the Result-Code AVP indicates a failure.

7.5. Failed-AVP AVP

The Failed-AVP AVP (AVP Code 279) is of type Grouped and provides debugging information in cases where a request is rejected or not fully processed due to erroneous information in a specific AVP. The value of the Result-Code AVP will provide information on the reason for the Failed-AVP AVP. A Diameter answer message SHOULD contain only one Failed-AVP that corresponds to the error indicated by the Result-Code AVP. For practical purposes, this Failed-AVP would typically refer to the first AVP processing error that a Diameter node encounters.

The possible reasons for this AVP are the presence of an improperly constructed AVP, an unsupported or unrecognized AVP, an invalid AVP value, the omission of a required AVP, the presence of an explicitly excluded AVP (see tables in Section 10) or the presence of two or more occurrences of an AVP which is restricted to 0, 1, or 0-1 occurrences.

A Diameter message SHOULD contain one Failed-AVP AVP, containing the entire AVP that could not be processed successfully. If the failure reason is omission of a required AVP, an AVP with the missing AVP code, the missing vendor id, and a zero filled payload of the minimum required length for the omitted AVP will be added. If the failure reason is an invalid AVP length where the reported length is less than the minimum AVP header length or greater than the reported message length, a copy of the offending AVP header and a zero filled payload of the minimum required length SHOULD be added.

In the case where the offending AVP is embedded within a grouped AVP, the Failed-AVP MAY contain the grouped AVP which in turn contains the single offending AVP. The same method MAY be employed if the grouped AVP itself is embedded in yet another grouped AVP and so on. In this case, the Failed-AVP MAY contain the grouped AVP hierarchy up to the single offending AVP. This enables the recipient to detect the location of the offending AVP when embedded in a group.

AVP Format

```
<Failed-AVP> ::= < AVP Header: 279 >
                 1* {AVP}
```

7.6. Experimental-Result AVP

The Experimental-Result AVP (AVP Code 297) is of type Grouped, and indicates whether a particular vendor-specific request was completed successfully or whether an error occurred. This AVP has the following structure:

AVP Format

```
Experimental-Result ::= < AVP Header: 297 >
                        { Vendor-Id }
                        { Experimental-Result-Code }
```

The Vendor-Id AVP (see Section 5.3.3 in this grouped AVP identifies the vendor responsible for the assignment of the result code which follows. All Diameter answer messages defined in vendor-specific applications MUST include either one Result-Code AVP or one Experimental-Result AVP.

7.7. Experimental-Result-Code AVP

The Experimental-Result-Code AVP (AVP Code 298) is of type Unsigned32 and contains a vendor-assigned value representing the result of processing the request.

It is recommended that vendor-specific result codes follow the same conventions given for the Result-Code AVP regarding the different types of result codes and the handling of errors (for non 2xxx values).

8. Diameter User Sessions

In general, Diameter can provide two different types of services to applications. The first involves authentication and authorization, and can optionally make use of accounting. The second only makes use of accounting.

When a service makes use of the authentication and/or authorization portion of an application, and a user requests access to the network, the Diameter client issues an auth request to its local server. The auth request is defined in a service-specific Diameter application (e.g., NASREQ). The request contains a Session-Id AVP, which is used in subsequent messages (e.g., subsequent authorization, accounting, etc) relating to the user's session. The Session-Id AVP is a means for the client and servers to correlate a Diameter message with a user session.

When a Diameter server authorizes a user to use network resources for a finite amount of time, and it is willing to extend the authorization via a future request, it MUST add the Authorization-Lifetime AVP to the answer message. The Authorization-Lifetime AVP defines the maximum number of seconds a user MAY make use of the resources before another authorization request is expected by the server. The Auth-Grace-Period AVP contains the number of seconds following the expiration of the Authorization-Lifetime, after which the server will release all state information related to the user's session. Note that if payment for services is expected by the serving realm from the user's home realm, the Authorization-Lifetime AVP, combined with the Auth-Grace-Period AVP, implies the maximum length of the session the home realm is willing to be fiscally responsible for. Services provided past the expiration of the Authorization-Lifetime and Auth-Grace-Period AVPs are the responsibility of the access device. Of course, the actual cost of services rendered is clearly outside the scope of the protocol.

An access device that does not expect to send a re-authorization or a session termination request to the server MAY include the Auth-Session-State AVP with the value set to NO_STATE_MAINTAINED as a hint to the server. If the server accepts the hint, it agrees that since no session termination message will be received once service to the user is terminated, it cannot maintain state for the session. If the answer message from the server contains a different value in the Auth-Session-State AVP (or the default value if the AVP is absent), the access device MUST follow the server's directives. Note that the value NO_STATE_MAINTAINED MUST NOT be set in subsequent re-authorization requests and answers.

The base protocol does not include any authorization request messages, since these are largely application-specific and are defined in a Diameter application document. However, the base protocol does define a set of messages that are used to terminate user sessions. These are used to allow servers that maintain state information to free resources.

When a service only makes use of the Accounting portion of the Diameter protocol, even in combination with an application, the Session-Id is still used to identify user sessions. However, the session termination messages are not used, since a session is signaled as being terminated by issuing an accounting stop message.

Diameter may also be used for services that cannot be easily categorized as authentication, authorization or accounting (e.g., certain 3GPP IMS interfaces). In such cases, the finite state machine defined in subsequent sections may not be applicable. Therefore, the applications itself MAY need to define its own finite

state machine. However, such application-specific state machines SHOULD follow the general state machine framework outlined in this document such as the use of Session-Id AVPs and the use of STR/STA, ASR/ASA messages for stateful sessions.

8.1. Authorization Session State Machine

This section contains a set of finite state machines, representing the life cycle of Diameter sessions, and which MUST be observed by all Diameter implementations that make use of the authentication and/or authorization portion of a Diameter application. The term Service-Specific below refers to a message defined in a Diameter application (e.g., Mobile IPv4, NASREQ).

There are four different authorization session state machines supported in the Diameter base protocol. The first two describe a session in which the server is maintaining session state, indicated by the value of the Auth-Session-State AVP (or its absence). One describes the session from a client perspective, the other from a server perspective. The second two state machines are used when the server does not maintain session state. Here again, one describes the session from a client perspective, the other from a server perspective.

When a session is moved to the Idle state, any resources that were allocated for the particular session must be released. Any event not listed in the state machines MUST be considered as an error condition, and an answer, if applicable, MUST be returned to the originator of the message.

In the case that an application does not support re-auth, the state transitions related to server-initiated re-auth when both client and server session maintains state (e.g., Send RAR, Pending, Receive RAA) MAY be ignored.

In the state table, the event 'Failure to send X' means that the Diameter agent is unable to send command X to the desired destination. This could be due to the peer being down, or due to the peer sending back a transient failure or temporary protocol error notification DIAMETER_TOO_BUSY or DIAMETER_LOOP_DETECTED in the Result-Code AVP of the corresponding Answer command. The event 'X successfully sent' is the complement of 'Failure to send X'.

The following state machine is observed by a client when state is maintained on the server:

CLIENT, STATEFUL

State	Event	Action	New State

Idle	Client or Device Requests access	Send service specific auth req	Pending
Idle	ASR Received for unknown session	Send ASA with Result-Code = UNKNOWN_ SESSION_ID	Idle
Idle	RAR Received for unknown session	Send RAA with Result-Code = UNKNOWN_ SESSION_ID	Idle
Pending	Successful Service-specific authorization answer received with default Auth-Session-State value	Grant Access	Open
Pending	Successful Service-specific authorization answer received but service not provided	Sent STR	Discon
Pending	Error processing successful Service-specific authorization answer	Sent STR	Discon
Pending	Failed Service-specific authorization answer received	Cleanup	Idle
Open	User or client device requests access to service	Send service specific auth req	Open
Open	Successful Service-specific authorization answer received	Provide Service	Open
Open	Failed Service-specific authorization answer received.	Discon. user/device	Idle
Open	RAR received and client will perform subsequent re-auth	Send RAA with	Open

		Result-Code = SUCCESS	
Open	RAR received and client will not perform subsequent re-auth	Send RAA with Result-Code != SUCCESS, Discon. user/device	Idle
Open	Session-Timeout Expires on Access Device	Send STR	Discon
Open	ASR Received, client will comply with request to end the session	Send ASA with Result-Code = = SUCCESS, Send STR.	Discon
Open	ASR Received, client will not comply with request to end the session	Send ASA with Result-Code != != SUCCESS	Open
Open	Authorization-Lifetime + Auth-Grace-Period expires on access device	Send STR	Discon
Discon	ASR Received	Send ASA	Discon
Discon	STA Received	Discon. user/device	Idle

The following state machine is observed by a server when it is maintaining state for the session:

SERVER, STATEFUL

State	Event	Action	New State
Idle	Service-specific authorization request received, and user is authorized	Send successful serv. specific answer	Open
Idle	Service-specific authorization request received, and user is not authorized	Send failed serv. specific	Idle

		answer	
Open	Service-specific authorization request received, and user is authorized	Send successful serv. specific answer	Open
Open	Service-specific authorization request received, and user is not authorized	Send failed serv. specific answer, Cleanup	Idle
Open	Home server wants to confirm authentication and/or authorization of the user	Send RAR	Pending
Pending	Received RAA with a failed Result-Code	Cleanup	Idle
Pending	Received RAA with Result-Code = SUCCESS	Update session	Open
Open	Home server wants to terminate the service	Send ASR	Discon
Open	Authorization-Lifetime (and Auth-Grace-Period) expires on home server.	Cleanup	Idle
Open	Session-Timeout expires on home server	Cleanup	Idle
Discon	Failure to send ASR	Wait, resend ASR	Discon
Discon	ASR successfully sent and ASA Received with Result-Code	Cleanup	Idle
Not Discon	ASA Received	None	No Change.
Any	STR Received	Send STA, Cleanup.	Idle

The following state machine is observed by a client when state is not maintained on the server:

CLIENT, STATELESS

State	Event	Action	New State
Idle	Client or Device Requests access	Send service specific auth req	Pending
Pending	Successful Service-specific authorization answer received with Auth-Session-State set to NO_STATE_MAINTAINED	Grant Access	Open
Pending	Failed Service-specific authorization answer received	Cleanup	Idle
Open	Session-Timeout Expires on Access Device	Discon. user/device	Idle
Open	Service to user is terminated	Discon. user/device	Idle

The following state machine is observed by a server when it is not maintaining state for the session:

SERVER, STATELESS

State	Event	Action	New State
Idle	Service-specific authorization request received, and successfully processed	Send serv. specific answer	Idle

8.2. Accounting Session State Machine

The following state machines MUST be supported for applications that have an accounting portion or that require only accounting services. The first state machine is to be observed by clients.

See Section 9.7 for Accounting Command Codes and Section 9.8 for Accounting AVPs.

The server side in the accounting state machine depends in some cases on the particular application. The Diameter base protocol defines a default state machine that MUST be followed by all applications that have not specified other state machines. This is the second state machine in this section described below.

The default server side state machine requires the reception of accounting records in any order and at any time, and does not place any standards requirement on the processing of these records. Implementations of Diameter may perform checking, ordering, correlation, fraud detection, and other tasks based on these records. AVPs may need to be inspected as a part of these tasks. The tasks can happen either immediately after record reception or in a post-processing phase. However, as these tasks are typically application or even policy dependent, they are not standardized by the Diameter specifications. Applications MAY define requirements on when to accept accounting records based on the used value of Accounting-Realtime-Required AVP, credit limits checks, and so on.

However, the Diameter base protocol defines one optional server side state machine that MAY be followed by applications that require keeping track of the session state at the accounting server. Note that such tracking is incompatible with the ability to sustain long duration connectivity problems. Therefore, the use of this state machine is recommended only in applications where the value of the Accounting-Realtime-Required AVP is DELIVER_AND_GRANT, and hence accounting connectivity problems are required to cause the serviced user to be disconnected. Otherwise, records produced by the client may be lost by the server which no longer accepts them after the connectivity is re-established. This state machine is the third state machine in this section. The state machine is supervised by a supervision session timer Ts, which the value should be reasonably higher than the Acct_Interim_Interval value. Ts MAY be set to two times the value of the Acct_Interim_Interval so as to avoid the accounting session in the Diameter server to change to Idle state in case of short transient network failure.

Any event not listed in the state machines MUST be considered as an error condition, and a corresponding answer, if applicable, MUST be returned to the originator of the message.

In the state table, the event ’Failure to send’ means that the Diameter client is unable to communicate with the desired destination. This could be due to the peer being down, or due to the peer sending back a transient failure or temporary protocol error notification DIAMETER_OUT_OF_SPACE, DIAMETER_TOO_BUSY, or DIAMETER_LOOP_DETECTED in the Result-Code AVP of the Accounting Answer command.

The event ’Failed answer’ means that the Diameter client received a non-transient failure notification in the Accounting Answer command.

Note that the action ’Disconnect user/dev’ MUST have an effect also to the authorization session state table, e.g., cause the STR message

to be sent, if the given application has both authentication/ authorization and accounting portions.

The states PendingS, PendingI, PendingL, PendingE and PendingB stand for pending states to wait for an answer to an accounting request related to a Start, Interim, Stop, Event or buffered record, respectively.

CLIENT, ACCOUNTING

State	Event	Action	New State
Idle	Client or device requests access	Send accounting start req.	PendingS
Idle	Client or device requests a one-time service	Send accounting event req	PendingE
Idle	Records in storage	Send record	PendingB
PendingS	Successful accounting start answer received		Open
PendingS	Failure to send and buffer space available and realtime not equal to DELIVER_AND_GRANT	Store Start Record	Open
PendingS	Failure to send and no buffer space available and realtime equal to GRANT_AND_LOSE		Open
PendingS	Failure to send and no buffer space available and realtime not equal to GRANT_AND_LOSE	Disconnect user/dev	Idle
PendingS	Failed accounting start answer received and realtime equal to GRANT_AND_LOSE		Open
PendingS	Failed accounting start answer received and realtime not equal to GRANT_AND_LOSE	Disconnect user/dev	Idle
PendingS	User service terminated	Store stop	PendingS

		record	
Open	Interim interval elapses	Send accounting interim record	PendingI
Open	User service terminated	Send accounting stop req.	PendingL
PendingI	Successful accounting interim answer received		Open
PendingI	Failure to send and (buffer space available or old record can be overwritten) and realtime not equal to DELIVER_AND_GRANT	Store interim record	Open
PendingI	Failure to send and no buffer space available and realtime equal to GRANT_AND_LOSE		Open
PendingI	Failure to send and no buffer space available and realtime not equal to GRANT_AND_LOSE	Disconnect user/dev	Idle
PendingI	Failed accounting interim answer received and realtime equal to GRANT_AND_LOSE		Open
PendingI	Failed accounting interim answer received and realtime not equal to GRANT_AND_LOSE	Disconnect user/dev	Idle
PendingI	User service terminated	Store stop record	PendingI
PendingE	Successful accounting event answer received		Idle
PendingE	Failure to send and buffer space available	Store event record	Idle

PendingE	Failure to send and no buffer space available		Idle
PendingE	Failed accounting event answer received		Idle
PendingB	Successful accounting answer received	Delete record	Idle
PendingB	Failure to send		Idle
PendingB	Failed accounting answer received	Delete record	Idle
PendingL	Successful accounting stop answer received		Idle
PendingL	Failure to send and buffer space available	Store stop record	Idle
PendingL	Failure to send and no buffer space available		Idle
PendingL	Failed accounting stop answer received		Idle

SERVER, STATELESS ACCOUNTING

State	Event	Action	New State
Idle	Accounting start request received, and successfully processed.	Send accounting start answer	Idle
Idle	Accounting event request received, and successfully processed.	Send accounting event answer	Idle
Idle	Interim record received, and successfully processed.	Send accounting interim answer	Idle
Idle	Accounting stop request	Send	Idle

	received, and successfully processed	accounting stop answer	
Idle	Accounting request received, no space left to store records	Send accounting answer, Result-Code = OUT_OF_ SPACE	Idle

SERVER, STATEFUL ACCOUNTING

State	Event	Action	New State
Idle	Accounting start request received, and successfully processed.	Send accounting start answer, Start Ts	Open
Idle	Accounting event request received, and successfully processed.	Send accounting event answer	Idle
Idle	Accounting request received, no space left to store records	Send accounting answer, Result-Code = OUT_OF_ SPACE	Idle
Open	Interim record received, and successfully processed.	Send accounting interim answer, Restart Ts	Open
Open	Accounting stop request received, and successfully processed	Send accounting stop answer, Stop Ts	Idle
Open	Accounting request received, no space left to store records	Send accounting answer, Result-Code = OUT_OF_	Idle

		SPACE, Stop Ts	
Open	Session supervision timer Ts expired	Stop Ts	Idle

8.3. Server-Initiated Re-Auth

A Diameter server may initiate a re-authentication and/or re-authorization service for a particular session by issuing a Re-Auth-Request (RAR).

For example, for pre-paid services, the Diameter server that originally authorized a session may need some confirmation that the user is still using the services.

An access device that receives a RAR message with Session-Id equal to a currently active session MUST initiate a re-auth towards the user, if the service supports this particular feature. Each Diameter application MUST state whether server-initiated re-auth is supported, since some applications do not allow access devices to prompt the user for re-auth.

8.3.1. Re-Auth-Request

The Re-Auth-Request (RAR), indicated by the Command-Code set to 258 and the message flags' 'R' bit set, may be sent by any server to the access device that is providing session service, to request that the user be re-authenticated and/or re-authorized.

Message Format

```
     <RAR>  ::= < Diameter Header: 258, REQ, PXY >
                < Session-Id >
                { Origin-Host }
                { Origin-Realm }
                { Destination-Realm }
                { Destination-Host }
                { Auth-Application-Id }
                { Re-Auth-Request-Type }
                [ User-Name ]
                [ Origin-State-Id ]
              * [ Proxy-Info ]
              * [ Route-Record ]
              * [ AVP ]
```

8.3.2. Re-Auth-Answer

The Re-Auth-Answer (RAA), indicated by the Command-Code set to 258 and the message flags' 'R' bit clear, is sent in response to the RAR. The Result-Code AVP MUST be present, and indicates the disposition of the request.

A successful RAA message MUST be followed by an application-specific authentication and/or authorization message.

Message Format

```
     <RAA>  ::= < Diameter Header: 258, PXY >
                < Session-Id >
                { Result-Code }
                { Origin-Host }
                { Origin-Realm }
                [ User-Name ]
                [ Origin-State-Id ]
                [ Error-Message ]
                [ Error-Reporting-Host ]
                [ Failed-AVP ]
              * [ Redirect-Host ]
                [ Redirect-Host-Usage ]
                [ Redirect-Max-Cache-Time ]
              * [ Proxy-Info ]
              * [ AVP ]
```

8.4. Session Termination

It is necessary for a Diameter server that authorized a session, for which it is maintaining state, to be notified when that session is no longer active, both for tracking purposes as well as to allow stateful agents to release any resources that they may have provided for the user's session. For sessions whose state is not being maintained, this section is not used.

When a user session that required Diameter authorization terminates, the access device that provided the service MUST issue a Session-Termination-Request (STR) message to the Diameter server that authorized the service, to notify it that the session is no longer active. An STR MUST be issued when a user session terminates for any reason, including user logoff, expiration of Session-Timeout, administrative action, termination upon receipt of an Abort-Session-Request (see below), orderly shutdown of the access device, etc.

The access device also MUST issue an STR for a session that was

authorized but never actually started. This could occur, for example, due to a sudden resource shortage in the access device, or because the access device is unwilling to provide the type of service requested in the authorization, or because the access device does not support a mandatory AVP returned in the authorization, etc.

It is also possible that a session that was authorized is never actually started due to action of a proxy. For example, a proxy may modify an authorization answer, converting the result from success to failure, prior to forwarding the message to the access device. If the answer did not contain an Auth-Session-State AVP with the value NO_STATE_MAINTAINED, a proxy that causes an authorized session not to be started MUST issue an STR to the Diameter server that authorized the session, since the access device has no way of knowing that the session had been authorized.

A Diameter server that receives an STR message MUST clean up resources (e.g., session state) associated with the Session-Id specified in the STR, and return a Session-Termination-Answer.

A Diameter server also MUST clean up resources when the Session-Timeout expires, or when the Authorization-Lifetime and the Auth-Grace-Period AVPs expires without receipt of a re-authorization request, regardless of whether an STR for that session is received. The access device is not expected to provide service beyond the expiration of these timers; thus, expiration of either of these timers implies that the access device may have unexpectedly shut down.

8.4.1. Session-Termination-Request

The Session-Termination-Request (STR), indicated by the Command-Code set to 275 and the Command Flags' 'R' bit set, is sent by a Diameter client or by a Diameter proxy to inform the Diameter Server that an authenticated and/or authorized session is being terminated.

Message Format

```
    <STR>  ::= < Diameter Header: 275, REQ, PXY >
               < Session-Id >
               { Origin-Host }
               { Origin-Realm }
               { Destination-Realm }
               { Auth-Application-Id }
               { Termination-Cause }
               [ User-Name ]
               [ Destination-Host ]
             * [ Class ]
               [ Origin-State-Id ]
             * [ Proxy-Info ]
             * [ Route-Record ]
             * [ AVP ]
```

8.4.2. Session-Termination-Answer

The Session-Termination-Answer (STA), indicated by the Command-Code set to 275 and the message flags' 'R' bit clear, is sent by the Diameter Server to acknowledge the notification that the session has been terminated. The Result-Code AVP MUST be present, and MAY contain an indication that an error occurred while servicing the STR.

Upon sending or receipt of the STA, the Diameter Server MUST release all resources for the session indicated by the Session-Id AVP. Any intermediate server in the Proxy-Chain MAY also release any resources, if necessary.

Message Format

```
     <STA> ::= < Diameter Header: 275, PXY >
               < Session-Id >
               { Result-Code }
               { Origin-Host }
               { Origin-Realm }
               [ User-Name ]
             * [ Class ]
               [ Error-Message ]
               [ Error-Reporting-Host ]
               [ Failed-AVP ]
               [ Origin-State-Id ]
             * [ Redirect-Host ]
               [ Redirect-Host-Usage ]
               [ Redirect-Max-Cache-Time ]
             * [ Proxy-Info ]
```

```
                  * [ AVP ]
```

8.5. Aborting a Session

A Diameter server may request that the access device stop providing service for a particular session by issuing an Abort-Session-Request (ASR).

For example, the Diameter server that originally authorized the session may be required to cause that session to be stopped for lack of credit or other reasons that were not anticipated when the session was first authorized.

An access device that receives an ASR with Session-ID equal to a currently active session MAY stop the session. Whether the access device stops the session or not is implementation- and/or configuration-dependent. For example, an access device may honor ASRs from certain agents only. In any case, the access device MUST respond with an Abort-Session-Answer, including a Result-Code AVP to indicate what action it took.

8.5.1. Abort-Session-Request

The Abort-Session-Request (ASR), indicated by the Command-Code set to 274 and the message flags' 'R' bit set, may be sent by any Diameter server or any Diameter proxy to the access device that is providing session service, to request that the session identified by the Session-Id be stopped.

Message Format

```
     <ASR>  ::= < Diameter Header: 274, REQ, PXY >
                < Session-Id >
                { Origin-Host }
                { Origin-Realm }
                { Destination-Realm }
                { Destination-Host }
                { Auth-Application-Id }
                [ User-Name ]
                [ Origin-State-Id ]
              * [ Proxy-Info ]
              * [ Route-Record ]
              * [ AVP ]
```

8.5.2. Abort-Session-Answer

The Abort-Session-Answer (ASA), indicated by the Command-Code set to 274 and the message flags' 'R' bit clear, is sent in response to the ASR. The Result-Code AVP MUST be present, and indicates the disposition of the request.

If the session identified by Session-Id in the ASR was successfully terminated, Result-Code is set to DIAMETER_SUCCESS. If the session is not currently active, Result-Code is set to DIAMETER_UNKNOWN_SESSION_ID. If the access device does not stop the session for any other reason, Result-Code is set to DIAMETER_UNABLE_TO_COMPLY.

Message Format

```
<ASA>  ::= < Diameter Header: 274, PXY >
           < Session-Id >
           { Result-Code }
           { Origin-Host }
           { Origin-Realm }
           [ User-Name ]
           [ Origin-State-Id ]
           [ Error-Message ]
           [ Error-Reporting-Host ]
           [ Failed-AVP ]
         * [ Redirect-Host ]
           [ Redirect-Host-Usage ]
           [ Redirect-Max-Cache-Time ]
         * [ Proxy-Info ]
         * [ AVP ]
```

8.6. Inferring Session Termination from Origin-State-Id

The Origin-State-Id is used to allow detection of terminated sessions for which no STR would have been issued, due to unanticipated shutdown of an access device.

A Diameter client or access device increments the value of the Origin-State-Id every time it is started or powered-up. The new Origin-State-Id is then sent in the CER/CEA message immediately upon connection to the server. The Diameter server receiving the new Origin-State-Id can determine whether the sending Diameter client had abruptly shutdown by comparing the old value of the Origin-State-Id it has kept for that specific client is less than the new value and whether it has un-terminated sessions originating from that client.

An access device can also include the Origin-State-Id in request messages other than CER if there are relays or proxies in between the access device and the server. In this case, however, the server cannot discover that the access device has been restarted unless and until it receives a new request from it. Therefore this mechanism is more opportunistic across proxies and relays.

The Diameter server may assume that all sessions that were active prior to detection of a client restart have been terminated. The Diameter server MAY clean up all session state associated with such lost sessions, and MAY also issues STRs for all such lost sessions that were authorized on upstream servers, to allow session state to be cleaned up globally.

8.7. Auth-Request-Type AVP

The Auth-Request-Type AVP (AVP Code 274) is of type Enumerated and is included in application-specific auth requests to inform the peers whether a user is to be authenticated only, authorized only or both. Note any value other than both MAY cause RADIUS interoperability issues. The following values are defined:

AUTHENTICATE_ONLY 1

The request being sent is for authentication only, and MUST contain the relevant application specific authentication AVPs that are needed by the Diameter server to authenticate the user.

AUTHORIZE_ONLY 2

The request being sent is for authorization only, and MUST contain the application-specific authorization AVPs that are necessary to identify the service being requested/offered.

AUTHORIZE_AUTHENTICATE 3

The request contains a request for both authentication and authorization. The request MUST include both the relevant application-specific authentication information, and authorization information necessary to identify the service being requested/ offered.

8.8. Session-Id AVP

The Session-Id AVP (AVP Code 263) is of type UTF8String and is used to identify a specific session (see Section 8). All messages pertaining to a specific session MUST include only one Session-Id AVP and the same value MUST be used throughout the life of a session. When present, the Session-Id SHOULD appear immediately following the Diameter Header (see Section 3).

The Session-Id MUST be globally and eternally unique, as it is meant to uniquely identify a user session without reference to any other information, and may be needed to correlate historical authentication information with accounting information. The Session-Id includes a mandatory portion and an implementation-defined portion; a recommended format for the implementation-defined portion is outlined below.

The Session-Id MUST begin with the sender's identity encoded in the DiameterIdentity type (see Section 4.3.1). The remainder of the Session-Id is delimited by a ";" character, and MAY be any sequence that the client can guarantee to be eternally unique; however, the following format is recommended, (square brackets [] indicate an optional element):

<DiameterIdentity>;<high 32 bits>;<low 32 bits>[;<optional value>]

<high 32 bits> and <low 32 bits> are decimal representations of the high and low 32 bits of a monotonically increasing 64-bit value. The 64-bit value is rendered in two part to simplify formatting by 32-bit processors. At startup, the high 32 bits of the 64-bit value MAY be initialized to the time in NTP format [RFC5905], and the low 32 bits MAY be initialized to zero. This will for practical purposes eliminate the possibility of overlapping Session-Ids after a reboot, assuming the reboot process takes longer than a second. Alternatively, an implementation MAY keep track of the increasing value in non-volatile memory.

<optional value> is implementation specific but may include a modem's device Id, a layer 2 address, timestamp, etc.

Example, in which there is no optional value:

accesspoint7.example.com;1876543210;523

Example, in which there is an optional value:

accesspoint7.example.com;1876543210;523;mobile@200.1.1.88

The Session-Id is created by the Diameter application initiating the session, which in most cases is done by the client. Note that a Session-Id MAY be used for both the authentication, authorization and accounting commands of a given application.

8.9. Authorization-Lifetime AVP

The Authorization-Lifetime AVP (AVP Code 291) is of type Unsigned32 and contains the maximum number of seconds of service to be provided to the user before the user is to be re-authenticated and/or re-authorized. Care should be taken when the Authorization-Lifetime value is determined, since a low, non-zero, value could create significant Diameter traffic, which could congest both the network and the agents.

A value of zero (0) means that immediate re-auth is necessary by the access device. The absence of this AVP, or a value of all ones (meaning all bits in the 32 bit field are set to one) means no re-auth is expected.

If both this AVP and the Session-Timeout AVP are present in a message, the value of the latter MUST NOT be smaller than the Authorization-Lifetime AVP.

An Authorization-Lifetime AVP MAY be present in re-authorization messages, and contains the number of seconds the user is authorized to receive service from the time the re-auth answer message is received by the access device.

This AVP MAY be provided by the client as a hint of the maximum lifetime that it is willing to accept. The server MUST return a value that is equal to, or smaller, than the one provided by the client.

8.10. Auth-Grace-Period AVP

The Auth-Grace-Period AVP (AVP Code 276) is of type Unsigned32 and contains the number of seconds the Diameter server will wait following the expiration of the Authorization-Lifetime AVP before cleaning up resources for the session.

8.11. Auth-Session-State AVP

The Auth-Session-State AVP (AVP Code 277) is of type Enumerated and specifies whether state is maintained for a particular session. The client MAY include this AVP in requests as a hint to the server, but the value in the server’s answer message is binding. The following values are supported:

STATE_MAINTAINED 0

This value is used to specify that session state is being maintained, and the access device MUST issue a session termination message when service to the user is terminated. This is the default value.

NO_STATE_MAINTAINED 1

This value is used to specify that no session termination messages will be sent by the access device upon expiration of the Authorization-Lifetime.

8.12. Re-Auth-Request-Type AVP

The Re-Auth-Request-Type AVP (AVP Code 285) is of type Enumerated and is included in application-specific auth answers to inform the client of the action expected upon expiration of the Authorization-Lifetime. If the answer message contains an Authorization-Lifetime AVP with a positive value, the Re-Auth-Request-Type AVP MUST be present in an answer message. The following values are defined:

AUTHORIZE_ONLY 0

An authorization only re-auth is expected upon expiration of the Authorization-Lifetime. This is the default value if the AVP is not present in answer messages that include the Authorization-Lifetime.

AUTHORIZE_AUTHENTICATE 1

An authentication and authorization re-auth is expected upon expiration of the Authorization-Lifetime.

8.13. Session-Timeout AVP

The Session-Timeout AVP (AVP Code 27) [RFC2865] is of type Unsigned32 and contains the maximum number of seconds of service to be provided to the user before termination of the session. When both the Session-Timeout and the Authorization-Lifetime AVPs are present in an answer message, the former MUST be equal to or greater than the value of the latter.

A session that terminates on an access device due to the expiration of the Session-Timeout MUST cause an STR to be issued, unless both the access device and the home server had previously agreed that no session termination messages would be sent (see Section 8).

A Session-Timeout AVP MAY be present in a re-authorization answer message, and contains the remaining number of seconds from the beginning of the re-auth.

A value of zero, or the absence of this AVP, means that this session has an unlimited number of seconds before termination.

This AVP MAY be provided by the client as a hint of the maximum timeout that it is willing to accept. However, the server MAY return a value that is equal to, or smaller, than the one provided by the client.

8.14. User-Name AVP

The User-Name AVP (AVP Code 1) [RFC2865] is of type UTF8String, which contains the User-Name, in a format consistent with the NAI specification [RFC4282].

8.15. Termination-Cause AVP

The Termination-Cause AVP (AVP Code 295) is of type Enumerated, and is used to indicate the reason why a session was terminated on the access device. The following values are defined:

DIAMETER_LOGOUT 1

The user initiated a disconnect

DIAMETER_SERVICE_NOT_PROVIDED 2

This value is used when the user disconnected prior to the receipt of the authorization answer message.

DIAMETER_BAD_ANSWER 3

This value indicates that the authorization answer received by the access device was not processed successfully.

DIAMETER_ADMINISTRATIVE 4

The user was not granted access, or was disconnected, due to administrative reasons, such as the receipt of a Abort-Session-Request message.

DIAMETER_LINK_BROKEN 5

The communication to the user was abruptly disconnected.

DIAMETER_AUTH_EXPIRED 6

The user's access was terminated since its authorized session time has expired.

DIAMETER_USER_MOVED 7

The user is receiving services from another access device.

DIAMETER_SESSION_TIMEOUT 8

The user's session has timed out, and service has been terminated.

8.16. Origin-State-Id AVP

The Origin-State-Id AVP (AVP Code 278), of type Unsigned32, is a monotonically increasing value that is advanced whenever a Diameter entity restarts with loss of previous state, for example upon reboot. Origin-State-Id MAY be included in any Diameter message, including CER.

A Diameter entity issuing this AVP MUST create a higher value for this AVP each time its state is reset. A Diameter entity MAY set Origin-State-Id to the time of startup, or it MAY use an incrementing counter retained in non-volatile memory across restarts.

The Origin-State-Id, if present, MUST reflect the state of the entity indicated by Origin-Host. If a proxy modifies Origin-Host, it MUST either remove Origin-State-Id or modify it appropriately as well. Typically, Origin-State-Id is used by an access device that always starts up with no active sessions; that is, any session active prior to restart will have been lost. By including Origin-State-Id in a message, it allows other Diameter entities to infer that sessions

associated with a lower Origin-State-Id are no longer active. If an access device does not intend for such inferences to be made, it MUST either not include Origin-State-Id in any message, or set its value to 0.

8.17. Session-Binding AVP

The Session-Binding AVP (AVP Code 270) is of type Unsigned32, and MAY be present in application-specific authorization answer messages. If present, this AVP MAY inform the Diameter client that all future application-specific re-auth and Session-Termination-Request messages for this session MUST be sent to the same authorization server.

This field is a bit mask, and the following bits have been defined:

RE_AUTH 1

When set, future re-auth messages for this session MUST NOT include the Destination-Host AVP. When cleared, the default value, the Destination-Host AVP MUST be present in all re-auth messages for this session.

STR 2

When set, the STR message for this session MUST NOT include the Destination-Host AVP. When cleared, the default value, the Destination-Host AVP MUST be present in the STR message for this session.

ACCOUNTING 4

When set, all accounting messages for this session MUST NOT include the Destination-Host AVP. When cleared, the default value, the Destination-Host AVP, if known, MUST be present in all accounting messages for this session.

8.18. Session-Server-Failover AVP

The Session-Server-Failover AVP (AVP Code 271) is of type Enumerated, and MAY be present in application-specific authorization answer messages that either do not include the Session-Binding AVP or include the Session-Binding AVP with any of the bits set to a zero value. If present, this AVP MAY inform the Diameter client that if a re-auth or STR message fails due to a delivery problem, the Diameter

client SHOULD issue a subsequent message without the Destination-Host AVP. When absent, the default value is REFUSE_SERVICE.

The following values are supported:

REFUSE_SERVICE 0

If either the re-auth or the STR message delivery fails, terminate service with the user, and do not attempt any subsequent attempts.

TRY_AGAIN 1

If either the re-auth or the STR message delivery fails, resend the failed message without the Destination-Host AVP present.

ALLOW_SERVICE 2

If re-auth message delivery fails, assume that re-authorization succeeded. If STR message delivery fails, terminate the session.

TRY_AGAIN_ALLOW_SERVICE 3

If either the re-auth or the STR message delivery fails, resend the failed message without the Destination-Host AVP present. If the second delivery fails for re-auth, assume re-authorization succeeded. If the second delivery fails for STR, terminate the session.

8.19. Multi-Round-Time-Out AVP

The Multi-Round-Time-Out AVP (AVP Code 272) is of type Unsigned32, and SHOULD be present in application-specific authorization answer messages whose Result-Code AVP is set to DIAMETER_MULTI_ROUND_AUTH. This AVP contains the maximum number of seconds that the access device MUST provide the user in responding to an authentication request.

8.20. Class AVP

The Class AVP (AVP Code 25) is of type OctetString and is used by Diameter servers to return state information to the access device. When one or more Class AVPs are present in application-specific authorization answer messages, they MUST be present in subsequent re-

authorization, session termination and accounting messages. Class AVPs found in a re-authorization answer message override the ones found in any previous authorization answer message. Diameter server implementations SHOULD NOT return Class AVPs that require more than 4096 bytes of storage on the Diameter client. A Diameter client that receives Class AVPs whose size exceeds local available storage MUST terminate the session.

8.21. Event-Timestamp AVP

The Event-Timestamp (AVP Code 55) is of type Time, and MAY be included in an Accounting-Request and Accounting-Answer messages to record the time that the reported event occurred, in seconds since January 1, 1900 00:00 UTC.

9. Accounting

This accounting protocol is based on a server directed model with capabilities for real-time delivery of accounting information. Several fault resilience methods [RFC2975] have been built in to the protocol in order minimize loss of accounting data in various fault situations and under different assumptions about the capabilities of the used devices.

9.1. Server Directed Model

The server directed model means that the device generating the accounting data gets information from either the authorization server (if contacted) or the accounting server regarding the way accounting data shall be forwarded. This information includes accounting record timeliness requirements.

As discussed in [RFC2975], real-time transfer of accounting records is a requirement, such as the need to perform credit limit checks and fraud detection. Note that batch accounting is not a requirement, and is therefore not supported by Diameter. Should batched accounting be required in the future, a new Diameter application will need to be created, or it could be handled using another protocol. Note, however, that even if at the Diameter layer accounting requests are processed one by one, transport protocols used under Diameter typically batch several requests in the same packet under heavy traffic conditions. This may be sufficient for many applications.

The authorization server (chain) directs the selection of proper transfer strategy, based on its knowledge of the user and relationships of roaming partnerships. The server (or agents) uses the Acct-Interim-Interval and Accounting-Realtime-Required AVPs to

control the operation of the Diameter peer operating as a client. The Acct-Interim-Interval AVP, when present, instructs the Diameter node acting as a client to produce accounting records continuously even during a session. Accounting-Realtime-Required AVP is used to control the behavior of the client when the transfer of accounting records from the Diameter client is delayed or unsuccessful.

The Diameter accounting server MAY override the interim interval or the realtime requirements by including the Acct-Interim-Interval or Accounting-Realtime-Required AVP in the Accounting-Answer message. When one of these AVPs is present, the latest value received SHOULD be used in further accounting activities for the same session.

9.2. Protocol Messages

A Diameter node that receives a successful authentication and/or authorization messages from the Diameter server SHOULD collect accounting information for the session. The Accounting-Request message is used to transmit the accounting information to the Diameter server, which MUST reply with the Accounting-Answer message to confirm reception. The Accounting-Answer message includes the Result-Code AVP, which MAY indicate that an error was present in the accounting message. The value of the Accounting-Realtime-Required AVP received earlier for the session in question may indicate that the user’s session has to be terminated when a rejected Accounting-Request message was received.

9.3. Accounting Application Extension and Requirements

Each Diameter application (e.g., NASREQ, MobileIP), SHOULD define their Service-Specific AVPs that MUST be present in the Accounting-Request message in a section entitled "Accounting AVPs". The application MUST assume that the AVPs described in this document will be present in all Accounting messages, so only their respective service-specific AVPs need to be defined in that section.

Applications have the option of using one or both of the following accounting application extension models:

Split Accounting Service

The accounting message will carry the Application Id of the Diameter base accounting application (see Section 2.4). Accounting messages may be routed to Diameter nodes other than the corresponding Diameter application. These nodes might be centralized accounting servers that provide accounting service for multiple different Diameter applications. These nodes MUST advertise the Diameter base accounting Application Id during

capabilities exchange.

Coupled Accounting Service

The accounting messages will carry the Application Id of the application that is using it. The application itself will process the received accounting records or forward them to an accounting server. There is no accounting application advertisement required during capabilities exchange and the accounting messages will be routed the same as any of the other application messages.

In cases where an application does not define its own accounting service, it is preferred that the split accounting model be used.

9.4. Fault Resilience

Diameter Base protocol mechanisms are used to overcome small message loss and network faults of temporary nature.

Diameter peers acting as clients MUST implement the use of failover to guard against server failures and certain network failures. Diameter peers acting as agents or related off-line processing systems MUST detect duplicate accounting records caused by the sending of the same record to several servers and duplication of messages in transit. This detection MUST be based on the inspection of the Session-Id and Accounting-Record-Number AVP pairs. Appendix C discusses duplicate detection needs and implementation issues.

Diameter clients MAY have non-volatile memory for the safe storage of accounting records over reboots or extended network failures, network partitions, and server failures. If such memory is available, the client SHOULD store new accounting records there as soon as the records are created and until a positive acknowledgement of their reception from the Diameter Server has been received. Upon a reboot, the client MUST starting sending the records in the non-volatile memory to the accounting server with appropriate modifications in termination cause, session length, and other relevant information in the records.

A further application of this protocol may include AVPs to control how many accounting records may at most be stored in the Diameter client without committing them to the non-volatile memory or transferring them to the Diameter server.

The client SHOULD NOT remove the accounting data from any of its memory areas before the correct Accounting-Answer has been received. The client MAY remove oldest, undelivered or yet unacknowledged

accounting data if it runs out of resources such as memory. It is an implementation dependent matter for the client to accept new sessions under this condition.

9.5. Accounting Records

In all accounting records, the Session-Id AVP MUST be present; the User-Name AVP MUST be present if it is available to the Diameter client.

Different types of accounting records are sent depending on the actual type of accounted service and the authorization server's directions for interim accounting. If the accounted service is a one-time event, meaning that the start and stop of the event are simultaneous, then the Accounting-Record-Type AVP MUST be present and set to the value EVENT_RECORD.

If the accounted service is of a measurable length, then the AVP MUST use the values START_RECORD, STOP_RECORD, and possibly, INTERIM_RECORD. If the authorization server has not directed interim accounting to be enabled for the session, two accounting records MUST be generated for each service of type session. When the initial Accounting-Request for a given session is sent, the Accounting-Record-Type AVP MUST be set to the value START_RECORD. When the last Accounting-Request is sent, the value MUST be STOP_RECORD.

If the authorization server has directed interim accounting to be enabled, the Diameter client MUST produce additional records between the START_RECORD and STOP_RECORD, marked INTERIM_RECORD. The production of these records is directed by Acct-Interim-Interval as well as any re-authentication or re-authorization of the session. The Diameter client MUST overwrite any previous interim accounting records that are locally stored for delivery, if a new record is being generated for the same session. This ensures that only one pending interim record can exist on an access device for any given session.

A particular value of Accounting-Sub-Session-Id MUST appear only in one sequence of accounting records from a Diameter client, except for the purposes of retransmission. The one sequence that is sent MUST be either one record with Accounting-Record-Type AVP set to the value EVENT_RECORD, or several records starting with one having the value START_RECORD, followed by zero or more INTERIM_RECORD and a single STOP_RECORD. A particular Diameter application specification MUST define the type of sequences that MUST be used.

9.6. Correlation of Accounting Records

If an application uses accounting messages, it can correlate accounting records with a specific application session by using the Session-Id of the particular application session in the accounting messages. Accounting messages MAY also use a different Session-Id from that of the application sessions in which case other session related information is needed to perform correlation.

In cases where an application requires multiple accounting sub-session, an Accounting-Sub-Session-Id AVP is used to differentiate each sub-session. The Session-Id would remain constant for all sub-sessions and is be used to correlate all the sub-sessions to a particular application session. Note that receiving a STOP_RECORD with no Accounting-Sub-Session-Id AVP when sub-sessions were originally used in the START_RECORD messages implies that all sub-sessions are terminated.

There are also cases where an application needs to correlate multiple application sessions into a single accounting record; the accounting record may span multiple different Diameter applications and sessions used by the same user at a given time. In such cases, the Acct-Multi-Session-Id AVP is used. The Acct-Multi-Session-Id AVP SHOULD be signaled by the server to the access device (typically during authorization) when it determines that a request belongs to an existing session. The access device MUST then include the Acct-Multi-Session-Id AVP in all subsequent accounting messages.

The Acct-Multi-Session-Id AVP MAY include the value of the original Session-Id. It’s contents are implementation specific, but MUST be globally unique across other Acct-Multi-Session-Id, and MUST NOT change during the life of a session.

A Diameter application document MUST define the exact concept of a session that is being accounted, and MAY define the concept of a multi-session. For instance, the NASREQ DIAMETER application treats a single PPP connection to a Network Access Server as one session, and a set of Multilink PPP sessions as one multi-session.

9.7. Accounting Command-Codes

This section defines Command-Code values that MUST be supported by all Diameter implementations that provide Accounting services.

9.7.1. Accounting-Request

The Accounting-Request (ACR) command, indicated by the Command-Code field set to 271 and the Command Flags’ ’R’ bit set, is sent by a

Diameter node, acting as a client, in order to exchange accounting information with a peer.

In addition to the AVPs listed below, Accounting-Request messages SHOULD include service-specific accounting AVPs.

Message Format

```
<ACR> ::= < Diameter Header: 271, REQ, PXY >
          < Session-Id >
          { Origin-Host }
          { Origin-Realm }
          { Destination-Realm }
          { Accounting-Record-Type }
          { Accounting-Record-Number }
          [ Acct-Application-Id ]
          [ Vendor-Specific-Application-Id ]
          [ User-Name ]
          [ Destination-Host ]
          [ Accounting-Sub-Session-Id ]
          [ Acct-Session-Id ]
          [ Acct-Multi-Session-Id ]
          [ Acct-Interim-Interval ]
          [ Accounting-Realtime-Required ]
          [ Origin-State-Id ]
          [ Event-Timestamp ]
        * [ Proxy-Info ]
        * [ Route-Record ]
        * [ AVP ]
```

9.7.2. Accounting-Answer

The Accounting-Answer (ACA) command, indicated by the Command-Code field set to 271 and the Command Flags' 'R' bit cleared, is used to acknowledge an Accounting-Request command. The Accounting-Answer command contains the same Session-Id as the corresponding request.

Only the target Diameter Server, known as the home Diameter Server, SHOULD respond with the Accounting-Answer command.

In addition to the AVPs listed below, Accounting-Answer messages SHOULD include service-specific accounting AVPs.

Message Format

```
<ACA> ::= < Diameter Header: 271, PXY >
          < Session-Id >
          { Result-Code }
          { Origin-Host }
          { Origin-Realm }
          { Accounting-Record-Type }
          { Accounting-Record-Number }
          [ Acct-Application-Id ]
          [ Vendor-Specific-Application-Id ]
          [ User-Name ]
          [ Accounting-Sub-Session-Id ]
          [ Acct-Session-Id ]
          [ Acct-Multi-Session-Id ]
          [ Error-Message ]
          [ Error-Reporting-Host ]
          [ Failed-AVP ]
          [ Acct-Interim-Interval ]
          [ Accounting-Realtime-Required ]
          [ Origin-State-Id ]
          [ Event-Timestamp ]
        * [ Proxy-Info ]
        * [ AVP ]
```

9.8. Accounting AVPs

This section contains AVPs that describe accounting usage information related to a specific session.

9.8.1. Accounting-Record-Type AVP

The Accounting-Record-Type AVP (AVP Code 480) is of type Enumerated and contains the type of accounting record being sent. The following values are currently defined for the Accounting-Record-Type AVP:

EVENT_RECORD 1

An Accounting Event Record is used to indicate that a one-time event has occurred (meaning that the start and end of the event are simultaneous). This record contains all information relevant to the service, and is the only record of the service.

START_RECORD 2

An Accounting Start, Interim, and Stop Records are used to indicate that a service of a measurable length has been given. An Accounting Start Record is used to initiate an accounting session, and contains accounting information that is relevant to the initiation of the session.

INTERIM_RECORD 3

An Interim Accounting Record contains cumulative accounting information for an existing accounting session. Interim Accounting Records SHOULD be sent every time a re-authentication or re-authorization occurs. Further, additional interim record triggers MAY be defined by application-specific Diameter applications. The selection of whether to use INTERIM_RECORD records is done by the Acct-Interim-Interval AVP.

STOP_RECORD 4

An Accounting Stop Record is sent to terminate an accounting session and contains cumulative accounting information relevant to the existing session.

9.8.2. Acct-Interim-Interval AVP

The Acct-Interim-Interval AVP (AVP Code 85) is of type Unsigned32 and is sent from the Diameter home authorization server to the Diameter client. The client uses information in this AVP to decide how and when to produce accounting records. With different values in this AVP, service sessions can result in one, two, or two+N accounting records, based on the needs of the home-organization. The following accounting record production behavior is directed by the inclusion of this AVP:

1. The omission of the Acct-Interim-Interval AVP or its inclusion with Value field set to 0 means that EVENT_RECORD, START_RECORD, and STOP_RECORD are produced, as appropriate for the service.

2. The inclusion of the AVP with Value field set to a non-zero value means that INTERIM_RECORD records MUST be produced between the START_RECORD and STOP_RECORD records. The Value field of this AVP is the nominal interval between these records in seconds.

The Diameter node that originates the accounting information, known as the client, MUST produce the first INTERIM_RECORD record roughly at the time when this nominal interval has elapsed from the START_RECORD, the next one again as the interval has elapsed once more, and so on until the session ends and a STOP_RECORD record is produced.

The client MUST ensure that the interim record production times are randomized so that large accounting message storms are not created either among records or around a common service start time.

9.8.3. Accounting-Record-Number AVP

The Accounting-Record-Number AVP (AVP Code 485) is of type Unsigned32 and identifies this record within one session. As Session-Id AVPs are globally unique, the combination of Session-Id and Accounting-Record-Number AVPs is also globally unique, and can be used in matching accounting records with confirmations. An easy way to produce unique numbers is to set the value to 0 for records of type EVENT_RECORD and START_RECORD, and set the value to 1 for the first INTERIM_RECORD, 2 for the second, and so on until the value for STOP_RECORD is one more than for the last INTERIM_RECORD.

9.8.4. Acct-Session-Id AVP

The Acct-Session-Id AVP (AVP Code 44) is of type OctetString is only used when RADIUS/Diameter translation occurs. This AVP contains the contents of the RADIUS Acct-Session-Id attribute.

9.8.5. Acct-Multi-Session-Id AVP

The Acct-Multi-Session-Id AVP (AVP Code 50) is of type UTF8String, following the format specified in Section 8.8. The Acct-Multi-Session-Id AVP is used to link together multiple related accounting sessions, where each session would have a unique Session-Id, but the same Acct-Multi-Session-Id AVP. This AVP MAY be returned by the Diameter server in an authorization answer, and MUST be used in all accounting messages for the given session.

9.8.6. Accounting-Sub-Session-Id AVP

The Accounting-Sub-Session-Id AVP (AVP Code 287) is of type Unsigned64 and contains the accounting sub-session identifier. The combination of the Session-Id and this AVP MUST be unique per sub-session, and the value of this AVP MUST be monotonically increased by one for all new sub-sessions. The absence of this AVP implies no sub-sessions are in use, with the exception of an Accounting-Request

whose Accounting-Record-Type is set to STOP_RECORD. A STOP_RECORD message with no Accounting-Sub-Session-Id AVP present will signal the termination of all sub-sessions for a given Session-Id.

9.8.7. Accounting-Realtime-Required AVP

The Accounting-Realtime-Required AVP (AVP Code 483) is of type Enumerated and is sent from the Diameter home authorization server to the Diameter client or in the Accounting-Answer from the accounting server. The client uses information in this AVP to decide what to do if the sending of accounting records to the accounting server has been temporarily prevented due to, for instance, a network problem.

DELIVER_AND_GRANT 1

The AVP with Value field set to DELIVER_AND_GRANT means that the service MUST only be granted as long as there is a connection to an accounting server. Note that the set of alternative accounting servers are treated as one server in this sense. Having to move the accounting record stream to a backup server is not a reason to discontinue the service to the user.

GRANT_AND_STORE 2

The AVP with Value field set to GRANT_AND_STORE means that service SHOULD be granted if there is a connection, or as long as records can still be stored as described in Section 9.4.

This is the default behavior if the AVP isn’t included in the reply from the authorization server.

GRANT_AND_LOSE 3

The AVP with Value field set to GRANT_AND_LOSE means that service SHOULD be granted even if the records cannot be delivered or stored.

10. AVP Occurrence Tables

The following tables presents the AVPs defined in this document, and specifies in which Diameter messages they MAY be present or not. AVPs that occur only inside a Grouped AVP are not shown in this table.

The table uses the following symbols:

0 The AVP MUST NOT be present in the message.

0+ Zero or more instances of the AVP MAY be present in the message.

0-1 Zero or one instance of the AVP MAY be present in the message. It is considered an error if there are more than one instance of the AVP.

1 One instance of the AVP MUST be present in the message.

1+ At least one instance of the AVP MUST be present in the message.

10.1. Base Protocol Command AVP Table

The table in this section is limited to the non-accounting Command Codes defined in this specification.

	Command-Code											
Attribute Name	CER	CEA	DPR	DPA	DWR	DWA	RAR	RAA	ASR	ASA	STR	STA
Acct-Interim-Interval	0	0	0	0	0	0	0-1	0	0	0	0	0
Accounting-Realtime-Required	0	0	0	0	0	0	0-1	0	0	0	0	0
Acct-Application-Id	0+	0+	0	0	0	0	0	0	0	0	0	0
Auth-Application-Id	0+	0+	0	0	0	0	1	0	1	0	1	0
Auth-Grace-Period	0	0	0	0	0	0	0	0	0	0	0	0
Auth-Request-Type	0	0	0	0	0	0	0	0	0	0	0	0
Auth-Session-State	0	0	0	0	0	0	0	0	0	0	0	0
Authorization-Lifetime	0	0	0	0	0	0	0	0	0	0	0	0
Class	0	0	0	0	0	0	0	0	0	0	0+	0+
Destination-Host	0	0	0	0	0	0	1	0	1	0	0-1	0
Destination-Realm	0	0	0	0	0	0	1	0	1	0	1	0
Disconnect-Cause	0	0	1	0	0	0	0	0	0	0	0	0
Error-Message	0	0-1	0	0-1	0	0-1	0	0-1	0	0-1	0	0-1
Error-Reporting-Host	0	0	0	0	0	0	0	0-1	0	0-1	0	0-1
Failed-AVP	0	0+	0	0+	0	0+	0	0+	0	0+	0	0+
Firmware-Revision	0-1	0-1	0	0	0	0	0	0	0	0	0	0
Host-IP-Address	1+	1+	0	0	0	0	0	0	0	0	0	0
Inband-Security-Id	0	0	0	0	0	0	0	0	0	0	0	0

Multi-Round-Time-Out	0	0	0	0	0	0	0	0	0	0	0	0
Origin-Host	1	1	1	1	1	1	1	1	1	1	1	1
Origin-Realm	1	1	1	1	1	1	1	1	1	1	1	1
Origin-State-Id	0-1	0-1	0	0	0-1	0-1	0-1	0-1	0-1	0-1	0-1	0-1
Product-Name	1	1	0	0	0	0	0	0	0	0	0	0
Proxy-Info	0	0	0	0	0	0	0+	0+	0+	0+	0+	0+
Redirect-Host	0	0	0	0	0	0	0	0+	0	0+	0	0+
Redirect-Host-Usage	0	0	0	0	0	0	0	0-1	0	0-1	0	0-1
Redirect-Max-Cache-Time	0	0	0	0	0	0	0	0-1	0	0-1	0	0-1
Result-Code	0	1	0	1	0	1	0	1	0	1	0	1
Re-Auth-Request-Type	0	0	0	0	0	0	1	0	0	0	0	0
Route-Record	0	0	0	0	0	0	0+	0	0+	0	0+	0
Session-Binding	0	0	0	0	0	0	0	0	0	0	0	0
Session-Id	0	0	0	0	0	0	1	1	1	1	1	1
Session-Server-Failover	0	0	0	0	0	0	0	0	0	0	0	0
Session-Timeout	0	0	0	0	0	0	0	0	0	0	0	0
Supported-Vendor-Id	0+	0+	0	0	0	0	0	0	0	0	0	0
Termination-Cause	0	0	0	0	0	0	0	0	0	0	1	0
User-Name	0	0	0	0	0	0	0-1	0-1	0-1	0-1	0-1	0-1
Vendor-Id	1	1	0	0	0	0	0	0	0	0	0	0
Vendor-Specific-Application-Id	0+	0+	0	0	0	0	0	0	0	0	0	0

10.2. Accounting AVP Table

The table in this section is used to represent which AVPs defined in this document are to be present in the Accounting messages. These AVP occurrence requirements are guidelines, which may be expanded, and/or overridden by application-specific requirements in the Diameter applications documents.

Attribute Name	Command Code ACR	Command Code ACA
Acct-Interim-Interval	0-1	0-1
Acct-Multi-Session-Id	0-1	0-1
Accounting-Record-Number	1	1
Accounting-Record-Type	1	1
Acct-Session-Id	0-1	0-1
Accounting-Sub-Session-Id	0-1	0-1
Accounting-Realtime-Required	0-1	0-1
Acct-Application-Id	0-1	0-1
Auth-Application-Id	0	0
Class	0+	0+
Destination-Host	0-1	0
Destination-Realm	1	0
Error-Reporting-Host	0	0+
Event-Timestamp	0-1	0-1
Origin-Host	1	1
Origin-Realm	1	1
Proxy-Info	0+	0+
Route-Record	0+	0
Result-Code	0	1
Session-Id	1	1
Termination-Cause	0	0
User-Name	0-1	0-1
Vendor-Specific-Application-Id	0-1	0-1

11. IANA Considerations

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the Diameter protocol, in accordance with [RFC5226]. Existing IANA registries and assignments put in place by [RFC3588] remain the same unless explicitly updated or deprecated in this section.

11.1. AVP Header

As defined in Section 4, the AVP header contains three fields that requires IANA namespace management; the AVP Code, Vendor-ID and Flags field.

11.1.1. AVP Codes

There are multiple namespaces. Vendors can have their own AVP Codes namespace which will be identified by their Vendor-ID (also known as Enterprise-Number) and they control the assignments of their vendor-specific AVP codes within their own namespace. The absence of a Vendor-ID or a Vendor-ID value of zero (0) identifies the IETF IANA controlled AVP Codes namespace. The AVP Codes and sometimes also possible values in an AVP are controlled and maintained by IANA. AVP Code 0 is not used. AVP Codes 1-255 are managed separately as RADIUS Attribute Types. Where a Vendor-Specific AVP is implemented by more than one vendor, allocation of global AVPs should be encouraged instead.

AVPs may be allocated following Expert Review (or Designated Expert) with Specification Required [RFC5226]. A block allocation (release of more than 3 AVPs at a time for a given purpose) requires IETF Review.

11.1.2. AVP Flags

Section 4.1 describes the existing AVP Flags. The remaining bits can only be assigned via a Standards Action [RFC5226].

11.2. Diameter Header

11.2.1. Command Codes

For the Diameter Header, the command code namespace allocation has changed. The new allocation rules are as follows:

The command code values 256 - 8,388,607 (0x100 to 0x7fffff) are for permanent, standard commands, allocated by IETF Review [RFC5226].

The values 8,388,608 - 16,777,213 (0x800000 - 0xfffffd) are reserved for vendor-specific command codes, to be allocated on a First Come, First Served basis by IANA [RFC5226]. The request to IANA for a Vendor-Specific Command Code SHOULD include a reference to a publicly available specification which documents the command in sufficient detail to aid in interoperability between independent implementations. If the specification cannot be made publicly available, the request for a vendor-specific command code MUST include the contact information of persons and/or entities responsible for authoring and maintaining the command.

The values 16,777,214 and 16,777,215 (hexadecimal values 0xfffffe - 0xffffff) are reserved for experimental commands. As these

codes are only for experimental and testing purposes, no guarantee is made for interoperability between Diameter peers using experimental commands.

11.2.2. Command Flags

Section 3 describes the existing Command Flag field. The remaining bits can only be assigned via a Standards Action [RFC5226].

11.3. AVP Values

For AVP values, the Experimental-Result-Code AVP value allocation has been added, see Section 11.3.1. The old AVP value allocation rule IETF Consensus has been updated to IETF Review as per [RFC5226] and affected AVPs are listed as reminders.

11.3.1. Experimental-Result-Code AVP

Values for this AVP are purely local to the indicated vendor, and no IANA registry is maintained for them.

11.3.2. Result-Code AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.3. Accounting-Record-Type AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.4. Termination-Cause AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.5. Redirect-Host-Usage AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.6. Session-Server-Failover AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.7. Session-Binding AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.8. Disconnect-Cause AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.9. Auth-Request-Type AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.10. Auth-Session-State AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.11. Re-Auth-Request-Type AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.12. Accounting-Realtime-Required AVP Values

New values are available for assignment via IETF Review [RFC5226].

11.3.13. Inband-Security-Id AVP (code 299)

The use of this AVP has been deprecated.

11.4. _diameters Service Name and Port Number Registration

This section requests the IANA to register the "_diameters" service name and assign port numbers for TLS/TCP and DTLS/SCTP according to the guidelines given in Cotton, et al. [RFC6335].

Service Name:	_diameters
Transport Protocols:	TCP, SCTP
Assignee:	IESG <iesg@ietf.org>
Contact:	IETF Chair <chair@ietf.org>
Description:	Diameter over TLS/TCP and DTLS/SCTP
Reference:	draft-ietf-dime-rfc3588bis
Port Number:	<TBD>, from the User Range

11.5. SCTP Payload Protocol Identifiers

Two SCTP payload protocol identifiers are registered in SCTP Payload Protocol Identifier registry:

Value	SCTP Payload Protocol Identifier
<TBD2>	Diameter in a SCTP DATA chunk
<TBD3>	Diameter in a DTLS/SCTP DATA chunk

11.6. S-NAPTR Parameters

This document also registers the following S-NAPTR Application Protocol Tags registry:

Tag	Protocol
diameter.dtls.sctp	DTLS/SCTP

12. Diameter Protocol-related Configurable Parameters

This section contains the configurable parameters that are found throughout this document:

Diameter Peer

A Diameter entity MAY communicate with peers that are statically configured. A statically configured Diameter peer would require that either the IP address or the fully qualified domain name (FQDN) be supplied, which would then be used to resolve through DNS.

Routing Table

A Diameter proxy server routes messages based on the realm portion of a Network Access Identifier (NAI). The server MUST have a table of Realm Names, and the address of the peer to which the message must be forwarded to. The routing table MAY also include a "default route", which is typically used for all messages that cannot be locally processed.

Tc timer

The Tc timer controls the frequency that transport connection attempts are done to a peer with whom no active transport

connection exists. The recommended value is 30 seconds.

13. Security Considerations

The Diameter base protocol messages SHOULD be secured by using TLS [RFC5246] or DTLS/SCTP [RFC6083]. Additional security mechanisms such as IPsec [RFC4301] MAY also be deployed to secure connections between peers. However, all Diameter base protocol implementations MUST support the use of TLS/TCP and DTLS/SCTP and the Diameter protocol MUST NOT be used without one of TLS, DTLS or IPsec.

If a Diameter connection is to be protected via TLS/TCP and DTLS/SCTP or IPsec, then TLS/TCP and DTLS/SCTP or IPsec/IKE SHOULD begin prior to any Diameter message exchange. All security parameters for TLS/TCP and DTLS/SCTP or IPsec are configured independent of the Diameter protocol. All Diameter messages will be sent through the TLS/TCP and DTLS/SCTP or IPsec connection after a successful setup.

For TLS/TCP and DTLS/SCTP connections to be established in the open state, the CER/CEA exchange MUST include an Inband-Security-ID AVP with a value of TLS/TCP and DTLS/SCTP. The TLS/TCP and DTLS/SCTP handshake will begin when both ends successfully reached the open state, after completion of the CER/CEA exchange. If the TLS/TCP and DTLS/SCTP handshake is successful, all further messages will be sent via TLS/TCP and DTLS/SCTP. If the handshake fails, both ends MUST move to the closed state. See Section 13.1 for more details.

13.1. TLS/TCP and DTLS/SCTP Usage

Diameter nodes using TLS/TCP and DTLS/SCTP for security MUST mutually authenticate as part of TLS/TCP and DTLS/SCTP session establishment. In order to ensure mutual authentication, the Diameter node acting as the TLS/TCP and DTLS/SCTP server MUST request a certificate from the Diameter node acting as TLS/TCP and DTLS/SCTP client, and the Diameter node acting as the TLS/TCP and DTLS/SCTP client MUST be prepared to supply a certificate on request.

Diameter nodes MUST be able to negotiate the following TLS/TCP and DTLS/SCTP cipher suites:

```
TLS_RSA_WITH_RC4_128_MD5
TLS_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA
```

Diameter nodes SHOULD be able to negotiate the following TLS/TCP and DTLS/SCTP cipher suite:

TLS_RSA_WITH_AES_128_CBC_SHA

Note that that it is quite possible that support for the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite will be REQUIRED at some future date. Diameter nodes MAY negotiate other TLS/TCP and DTLS/SCTP cipher suites.

If public key certificates are used for Diameter security (for example, with TLS), the value of the expiration times in the routing and peer tables MUST NOT be greater than the expiry time in the relevant certificates.

13.2. Peer-to-Peer Considerations

As with any peer-to-peer protocol, proper configuration of the trust model within a Diameter peer is essential to security. When certificates are used, it is necessary to configure the root certificate authorities trusted by the Diameter peer. These root CAs are likely to be unique to Diameter usage and distinct from the root CAs that might be trusted for other purposes such as Web browsing. In general, it is expected that those root CAs will be configured so as to reflect the business relationships between the organization hosting the Diameter peer and other organizations. As a result, a Diameter peer will typically not be configured to allow connectivity with any arbitrary peer. With certificate authentication, Diameter peers may not be known beforehand and therefore peer discovery may be required.

13.3. AVP Considerations

Diameter AVPs often contain security-sensitive data; for example, user passwords and location data, network addresses and cryptographic keys. The Diameter messages containing such AVPs MUST only be sent protected via mutually authenticated TLS or IPsec. In addition, those messages SHOULD NOT be sent via intermediate nodes that would expose the sensitive data at those nodes except in cases where an intermediary is known to be operated as part of the same administrative domain as the endpoints so that an ability to successfully compromise the intermediary would imply a high probability of being able to compromise the endpoints as well.

14. References

14.1. Normative References

[FLOATPOINT]
Institute of Electrical and Electronics Engineers, "IEEE

Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Standard 754-1985", August 1985.

[IANAADFAM]
IANA,, "Address Family Numbers", http://www.iana.org/assignments/address-family-numbers.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, March 2003.

[RFC3539] Aboba, B. and J. Wood, "Authentication, Authorization and Accounting (AAA) Transport Profile", RFC 3539, June 2003.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, January 2005.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4004] Calhoun, P., Johansson, T., Perkins, C., Hiller, T., and P. McCann, "Diameter Mobile IPv4 Application", RFC 4004, August 2005.

[RFC4005] Calhoun, P., Zorn, G., Spence, D., and D. Mitton, "Diameter Network Access Server Application", RFC 4005, August 2005.

[RFC4006] Hakala, H., Mattila, L., Koskinen, J-P., Stura, M., and J. Loughney, "Diameter Credit-Control Application", RFC 4006, August 2005.

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, December 2005.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5729] Korhonen, J., Jones, M., Morand, L., and T. Tsou, "Clarifications on the Routing of Diameter Requests Based on the Username and the Realm", RFC 5729, December 2009.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010.

[RFC5891] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, August 2010.

[RFC6083] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)", RFC 6083, January 2011.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

[RFC6408] Jones, M., Korhonen, J., and L. Morand, "Diameter Straightforward-Naming Authority Pointer (S-NAPTR) Usage", RFC 6408, November 2011.

[RFC791] Postel, J., "Internet Protocol", RFC 791, September 1981.

[RFC793] Postel, J., "Transmission Control Protocol", RFC 793, January 1981.

14.2. Informational References

[ENTERPRISE]
IANA, "SMI Network Management Private Enterprise Codes", http://www.iana.org/assignments/enterprise-numbers.

[RFC1492] Finseth, C., "An Access Control Protocol, Sometimes Called TACACS", RFC 1492, July 1993.

[RFC1661] Simpson, W., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC2866] Rigney, C., "RADIUS Accounting", RFC 2866, June 2000.

[RFC2869] Rigney, C., Willats, W., and P. Calhoun, "RADIUS Extensions", RFC 2869, June 2000.

[RFC2975] Aboba, B., Arkko, J., and D. Harrington, "Introduction to Accounting Management", RFC 2975, October 2000.

[RFC2989] Aboba, B., Calhoun, P., Glass, S., Hiller, T., McCann, P., Shiino, H., Walsh, P., Zorn, G., Dommety, G., Perkins, C., Patil, B., Mitton, D., Manning, S., Beadles, M., Chen, X., Sivalingham, S., Hameed, A., Munson, M., Jacobs, S., Lim, B., Hirschman, B., Hsu, R., Koo, H., Lipford, M., Campbell, E., Xu, Y., Baba, S., and E. Jaques, "Criteria for Evaluating AAA Protocols for Network Access", RFC 2989, November 2000.

[RFC3162] Aboba, B., Zorn, G., and D. Mitton, "RADIUS and IPv6", RFC 3162, August 2001.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H.

Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4690] Klensin, J., Faltstrom, P., Karp, C., and IAB, "Review and Recommendations for Internationalized Domain Names (IDNs)", RFC 4690, September 2006.

[RFC5176] Chiba, M., Dommety, G., Eklund, M., Mitton, D., and B. Aboba, "Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS)", RFC 5176, January 2008.

[RFC5461] Gont, F., "TCP's Reaction to Soft Errors", RFC 5461, February 2009.

[RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC5927] Gont, F., "ICMP Attacks against TCP", RFC 5927, July 2010.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, August 2011.

Appendix A. Acknowledgements

A.1. RFC3588bis

The authors would like to thank the following people that have provided proposals and contributions to this document:

To Vishnu Ram and Satendra Gera for their contributions on Capabilities Updates, and Predictive Loop Avoidance as well as many other technical proposals. To Tolga Asveren for his insights and contributions on almost all of the proposed solutions incorporated into this document. To Timothy Smith for helping on the Capabilities Update and other topics. To Tony Zhang for providing fixes to loop holes on composing Failed-AVPs as well as many other issues and topics. To Jan Nordqvist for clearly stating the usage of Application Ids. To Anders Kristensen for providing needed technical opinions. To David Frascone for providing invaluable review of the

document. To Mark Jones for providing clarifying text on vendor command codes and other vendor specific indicators. To Jouni Korhonen for taking over the editing task and resolving last bits from -27 through -29.

Special thanks to the Diameter extensibility design team which helped resolve the tricky question of mandatory AVPs and ABNF semantics. The members of this team are as follows:

Avi Lior, Jari Arkko, Glen Zorn, Lionel Morand, Mark Jones, Tolga Asveren Jouni Korhonen, Glenn McGregor.

Special thanks also to people who have provided invaluable comments and inputs especially in resolving controversial issues:

Glen Zorn, Yoshihiro Ohba, Marco Stura, Stephen Farrel, Pete Resnick, Peter Saint-Andre, Robert Sparks, Krishna Prasad, Sean Turner, Barry Leiba and Pasi Eronen.

Finally, we would like to thank the original authors of this document:

Pat Calhoun, John Loughney, Jari Arkko, Erik Guttman and Glen Zorn.

Their invaluable knowledge and experience has given us a robust and flexible AAA protocol that many people have seen great value in adopting. We greatly appreciate their support and stewardship for the continued improvements of Diameter as a protocol. We would also like to extend our gratitude to folks aside from the authors who have assisted and contributed to the original version of this document. Their efforts significantly contributed to the success of Diameter.

A.2. RFC3588

The authors would like to thank Nenad Trifunovic, Tony Johansson and Pankaj Patel for their participation in the pre-IETF Document Reading Party. Allison Mankin, Jonathan Wood and Bernard Aboba provided invaluable assistance in working out transport issues, and similarly with Steven Bellovin in the security area.

Paul Funk and David Mitton were instrumental in getting the Peer State Machine correct, and our deep thanks go to them for their time.

Text in this document was also provided by Paul Funk, Mark Eklund, Mark Jones and Dave Spence. Jacques Caron provided many great comments as a result of a thorough review of the spec.

The authors would also like to acknowledge the following people for

their contribution in the development of the Diameter protocol:

Allan C. Rubens, Haseeb Akhtar, William Bulley, Stephen Farrell, David Frascone, Daniel C. Fox, Lol Grant, Ignacio Goyret, Nancy Greene, Peter Heitman, Fredrik Johansson, Mark Jones, Martin Julien, Bob Kopacz, Paul Krumviede, Fergal Ladley, Ryan Moats, Victor Muslin, Kenneth Peirce, John Schnizlein, Sumit Vakil, John R. Vollbrecht and Jeff Weisberg.

Finally, Pat Calhoun would like to thank Sun Microsystems since most of the effort put into this document was done while he was in their employ.

Appendix B. S-NAPTR Example

As an example, consider a client that wishes to resolve aaa: ex1.example.com. The client performs a NAPTR query for that domain, and the following NAPTR records are returned:

```
 ;;        order pref flags service   regexp replacement
 IN NAPTR  50    50   "s"   "aaa:diameter.tls.tcp" ""
              _diameter._tls.ex1.example.com
 IN NAPTR  100   50   "s"   "aaa:diameter.tcp"     ""
              _aaa._tcp.ex1.example.com
 IN NAPTR  150   50   "s"   "aaa:diameter.sctp"    ""
              _diameter._sctp.ex1.example.com
```

This indicates that the server supports TLS, TCP and SCTP in that order. If the client supports TLS, TLS will be used, targeted to a host determined by an SRV lookup of _diameter._tls.ex1.example.com. That lookup would return:

```
 ;;       Priority  Weight  Port    Target
 IN SRV   0         1       5060    server1.ex1.example.com
 IN SRV   0         2       5060    server2.ex1.example.com
```

As an alternative example, a client that wishes to resolve aaa: ex2.example.com. The client performs a NAPTR query for that domain, and the following NAPTR records are returned:

```
 ;;        order pref flags service   regexp replacement
 IN NAPTR  150   50   "a"   "aaa:diameter.tls.tcp"  ""
              server1.ex2.example.com
 IN NAPTR  150   50   "a"   "aaa:diameter.tls.tcp"  ""
              server2.ex2.example.com
```

This indicates that the server supports TCP available at the returned

host names.

Appendix C. Duplicate Detection

As described in Section 9.4, accounting record duplicate detection is based on session identifiers. Duplicates can appear for various reasons:

- Failover to an alternate server. Where close to real-time performance is required, failover thresholds need to be kept low and this may lead to an increased likelihood of duplicates. Failover can occur at the client or within Diameter agents.
- Failure of a client or agent after sending of a record from non-volatile memory, but prior to receipt of an application layer ACK and deletion of the record. record to be sent. This will result in retransmission of the record soon after the client or agent has rebooted.
- Duplicates received from RADIUS gateways. Since the retransmission behavior of RADIUS is not defined within [RFC2865], the likelihood of duplication will vary according to the implementation.
- Implementation problems and misconfiguration.

The T flag is used as an indication of an application layer retransmission event, e.g., due to failover to an alternate server. It is defined only for request messages sent by Diameter clients or agents. For instance, after a reboot, a client may not know whether it has already tried to send the accounting records in its non-volatile memory before the reboot occurred. Diameter servers MAY use the T flag as an aid when processing requests and detecting duplicate messages. However, servers that do this MUST ensure that duplicates are found even when the first transmitted request arrives at the server after the retransmitted request. It can be used only in cases where no answer has been received from the Server for a request and the request is sent again, (e.g., due to a failover to an alternate peer, due to a recovered primary peer or due to a client re-sending a stored record from non-volatile memory such as after reboot of a client or agent).

In some cases the Diameter accounting server can delay the duplicate detection and accounting record processing until a post-processing phase takes place. At that time records are likely to be sorted according to the included User-Name and duplicate elimination is easy in this case. In other situations it may be necessary to perform

real-time duplicate detection, such as when credit limits are imposed or real-time fraud detection is desired.

In general, only generation of duplicates due to failover or re-sending of records in non-volatile storage can be reliably detected by Diameter clients or agents. In such cases the Diameter client or agents can mark the message as possible duplicate by setting the T flag. Since the Diameter server is responsible for duplicate detection, it can choose to make use of the T flag or not, in order to optimize duplicate detection. Since the T flag does not affect interoperability, and may not be needed by some servers, generation of the T flag is REQUIRED for Diameter clients and agents, but MAY be implemented by Diameter servers.

As an example, it can be usually be assumed that duplicates appear within a time window of longest recorded network partition or device fault, perhaps a day. So only records within this time window need to be looked at in the backward direction. Secondly, hashing techniques or other schemes, such as the use of the T flag in the received messages, may be used to eliminate the need to do a full search even in this set except for rare cases.

The following is an example of how the T flag may be used by the server to detect duplicate requests.

> A Diameter server MAY check the T flag of the received message to determine if the record is a possible duplicate. If the T flag is set in the request message, the server searches for a duplicate within a configurable duplication time window backward and forward. This limits database searching to those records where the T flag is set. In a well run network, network partitions and device faults will presumably be rare events, so this approach represents a substantial optimization of the duplicate detection process. During failover, it is possible for the original record to be received after the T flag marked record, due to differences in network delays experienced along the path by the original and duplicate transmissions. The likelihood of this occurring increases as the failover interval is decreased. In order to be able to detect out of order duplicates, the Diameter server should use backward and forward time windows when performing duplicate checking for the T flag marked request. For example, in order to allow time for the original record to exit the network and be recorded by the accounting server, the Diameter server can delay processing records with the T flag set until a time period TIME_WAIT + RECORD_PROCESSING_TIME has elapsed after the closing of the original transport connection. After this time period has expired, then it may check the T flag marked records against the

database with relative assurance that the original records, if sent, have been received and recorded.

Appendix D. Internationalized Domain Names

To be compatible with the existing DNS infrastructure and simplify host and domain name comparison, Diameter identities (FQDNs) are represented in ASCII form. This allows the Diameter protocol to fall in-line with the DNS strategy of being transparent from the effects of Internationalized Domain Names (IDNs) by following the recommendations in [RFC4690] and [RFC5890]. Applications that provide support for IDNs outside of the Diameter protocol but interacting with it SHOULD use the representation and conversion framework described in [RFC5890], [RFC5891] and [RFC3492].

Authors' Addresses

Victor Fajardo (editor)
Telcordia Technologies
One Telcordia Drive, 1S-222
Piscataway, NJ 08854
USA

Phone: +1-908-421-1845
Email: vf0213@gmail.com

Jari Arkko
Ericsson Research
02420 Jorvas
Finland

Phone: +358 40 5079256
Email: jari.arkko@ericsson.com

John Loughney
Nokia Research Center
955 Page Mill Road
Palo Alto, CA 94304
US

Phone: +1-650-283-8068
Email: john.loughney@nokia.com

Glen Zorn (editor)
Network Zen
227/358 Thanon Sanphawut
Bang Na, Bangkok 10260
Thailand

Phone: +66 (0) 87-0404617
Email: glenzorn@gmail.com

Network Working Group
Internet-Draft
Obsoletes: 4005 (if approved)
Intended status: Standards Track
Expires: June 1, 2014

G. Zorn, Ed.
Network Zen

November 28, 2013

Diameter Network Access Server Application
draft-ietf-dime-rfc4005bis-14

Abstract

This document describes the Diameter protocol application used for Authentication, Authorization, and Accounting (AAA) services in the Network Access Server (NAS) environment; it obsoletes RFC 4005. When combined with the Diameter Base protocol, Transport Profile, and Extensible Authentication Protocol specifications, this application specification satisfies typical network access services requirements.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 1, 2014.

Copyright Notice

Table of Contents

1. Introduction

This document describes the Diameter protocol application used for AAA in the Network Access Server (NAS) environment. When combined with the Diameter Base protocol [RFC6733], Transport Profile [RFC3539], and EAP [RFC4072] specifications, this specification

satisfies the NAS-related requirements defined in Aboba, et al. [RFC2989] and Beadles & Mitton [RFC3169].

First, this document describes the operation of a Diameter NAS application. Then it defines the Diameter message Command-Codes. The following sections list the AVPs used in these messages, grouped by common usage. These are session identification, authentication, authorization, tunneling, and accounting. The authorization AVPs are further broken down by service type.

1.1. Changes from RFC 4005

This document obsoletes RFC 4005 and is not backward compatible with that document. An overview of some of the major changes is given below.

- All of the material regarding RADIUS/Diameter protocol interactions has been removed; however, where AVPs are derived from RADIUS Attributes, the range and format of those Attribute values have been retained for ease of transition.

- The Command Code Format (CCF) [RFC6733] for the Accounting-Request and Accounting-Answer messages has been changed to explicitly require the inclusion of the Acct-Application-Id AVP and exclude the Vendor-Specific-Application-Id AVP. Normally, this type of change would require the allocation of a new command code and consequently, a new application-id (See Section 1.3.3 of [RFC6733]). However, the presence of an instance of the Acct-Application-Id AVP was required in RFC 4005, as well:

  > The ACR message [BASE] is sent by the NAS to report its session information to a target server downstream.
  >
  > Either of Acct-Application-Id or Vendor-Specific-Application-Id AVPs MUST be present. If the Vendor-Specific-Application-Id grouped AVP is present, it must have an Acct-Application-Id inside.

  Thus, though the syntax of the commands has changed, the semantics have not (with the caveat that the Acct-Application-Id AVP can no longer be contained in the Vendor-Specific-Application-Id AVP).

- The lists of RADIUS attribute values have been deleted in favor of references to the appropriate IANA registries.

- The accounting model to be used is now specified (see Section 1.6).

There are many other miscellaneous fixes that have been introduced in this document that may not be considered significant but they are useful nonetheless. Examples are fixes to example IP addresses, addition of clarifying references, etc. All of the errata previously filed against RFC 4005 have been fixed. A comprehensive list of changes is not shown here for practical reasons.

1.2. Terminology

Section 1.2 of the Diameter base protocol specification [RFC6733] defines most of the terminology used in this document. Additionally, the following terms and acronyms are used in this application:

NAS (Network Access Server)

A device that provides an access service for a user to a network. The service may be a network connection or a value-added service such as terminal emulation [RFC2881].

PPP (Point-to-Point Protocol)

A multiprotocol serial datalink. PPP is the primary IP datalink used for dial-in NAS connection service [RFC1661].

CHAP (Challenge Handshake Authentication Protocol)

An authentication process used in PPP [RFC1994].

PAP (Password Authentication Protocol)

A deprecated PPP authentication process, but often used for backward compatibility [RFC1334].

SLIP (Serial Line Interface Protocol)

A serial datalink that only supports IP. A design prior to PPP.

ARAP (Appletalk Remote Access Protocol)

A serial datalink for accessing Appletalk networks [ARAP].

IPX (Internet Packet Exchange)

The network protocol used by NetWare networks [IPX].

L2TP (Layer Two Tunneling Protocol)

L2TP [RFC3931] provides a dynamic mechanism for tunneling Layer 2 "circuits" across a packet-oriented data network.

LAC (L2TP Access Concentrator)

An L2TP Control Connection Endpoint being used to cross-connect an L2TP session directly to a data link [RFC3931].

LAT (Local Area Transport)

A Digital Equipment Corp. LAN protocol for terminal services [LAT].

LCP (Link Control Protocol)

One of the three major components of PPP [RFC1661]. LCP is used to automatically agree upon encapsulation format options, handle varying limits on sizes of packets, detect a looped-back link and other common misconfiguration errors, and terminate the link. Other optional facilities provided are authentication of the identity of its peer on the link, and determination when a link is functioning properly and when it is failing.

PPTP (Point-to-Point Tunneling Protocol)

A protocol which allows PPP to be tunneled through an IP network [RFC2637].

VPN (Virtual Private Network)

In this document, this term is used to describe access services that use tunneling methods.

1.3. Requirements Language

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

The use of "MUST" and "MUST NOT" in the AVP Flag rules columns of AVP Tables in this document refers to AVP flags ([RFC6733], Section 4.1) that:

- MUST be set to 1 in the AVP Header ("MUST" column) and
- MUST NOT be set to 1 ("MUST NOT" column)

1.4. Advertising Application Support

Diameter nodes conforming to this specification MUST advertise support by including the value of one (1) in the Auth-Application-Id of the Capabilities-Exchange-Request (CER) message [RFC6733].

1.5. Application Identification

When used in this application, the Auth-Application-Id AVP MUST be set to the value one (1) in the following messages

- AA-Request (Section 3.1)
- Re-Auth-Request(Section 3.3)
- Session-Termination-Request (Section 3.5)
- Abort-Session-Request (Section 3.7)

1.6. Accounting Model

It is RECOMMENDED that the coupled accounting model (RFC 6733, Section 9.3) be used with this application; therefore, the value of the Acct-Application-Id AVP in the Accounting-Request (Section 3.10) and Accounting-Answer (Section 3.9) messages SHOULD be set to one (1).

2. NAS Calls, Ports, and Sessions

The arrival of a new call or service connection at a port of a Network Access Server (NAS) starts a Diameter NAS Application message exchange. Information about the call, the identity of the user, and the user's authentication information are packaged into a Diameter AA-Request (AAR) message and sent to a server.

The server processes the information and responds with a Diameter AA-Answer (AAA) message that contains authorization information for the NAS, or a failure code (Result-Code AVP). A value of DIAMETER_MULTI_ROUND_AUTH indicates an additional authentication exchange, and several AAR and AAA messages may be exchanged until the transaction completes.

2.1. Diameter Session Establishment

When the authentication or authorization exchange completes successfully, the NAS application SHOULD start a session context. If the Result-Code of DIAMETER_MULTI_ROUND_AUTH is returned, the exchange continues until a success or error is returned.

If accounting is active, the application MUST also send an Accounting message [RFC6733]. An Accounting-Record-Type of START_RECORD is sent for a new session. If a session fails to start, the EVENT_RECORD message is sent with the reason for the failure described.

Note that the return of an unsupportable Accounting-Realtime-Required value [RFC6733] would result in a failure to establish the session.

2.2. Diameter Session Reauthentication or Reauthorization

The Diameter Base protocol allows users to be periodically reauthenticated and/or reauthorized. In such instances, the Session-Id AVP in the AAR message MUST be the same as the one present in the original authentication/authorization message.

A Diameter server informs the NAS of the maximum time allowed before reauthentication or reauthorization via the Authorization-Lifetime AVP [RFC6733]. A NAS MAY reauthenticate and/or reauthorize before the end, but A NAS MUST reauthenticate and/or reauthorize at the end of the period provided by the Authorization-Lifetime AVP. The failure of a reauthentication exchange will terminate the service.

Furthermore, it is possible for Diameter servers to issue an unsolicited reauthentication and/or reauthorization request (e.g., Re-Auth-Request (RAR) message [RFC6733]) to the NAS. Upon receipt of such a message, the NAS MUST respond to the request with a Re-Auth-Answer (RAA) message [RFC6733].

If the RAR properly identifies an active session, the NAS will initiate a new local reauthentication or authorization sequence as indicated by the Re-Auth-Request-Type value. This will cause the NAS to send a new AAR message using the existing Session-Id. The server will respond with an AAA message to specify the new service parameters.

If accounting is active, every change of authentication or authorization SHOULD generate an accounting message. If the NAS service is a continuation of the prior user context, then an Accounting-Record-Type of INTERIM_RECORD indicating the new session attributes and cumulative status would be appropriate. If a new user or a significant change in authorization is detected by the NAS, then the service may send two messages of the types STOP_RECORD and START_RECORD. Accounting may change the subsession identifiers (Acct-Session-ID, or Acct-Sub-Session-Id) to indicate such sub-sessions. A service may also use a different Session-Id value for accounting (see Section 9.6 of [RFC6733]).

However, the Diameter Session-ID AVP value used for the initial authorization exchange MUST be used to generate an STR message when the session context is terminated.

2.3. Diameter Session Termination

When a NAS receives an indication that a user's session is being disconnected by the client (e.g., an LCP Terminate-Request message [RFC1661] is received) or an administrative command, the NAS MUST issue a Session-Termination-Request (STR) [RFC6733] to its Diameter Server. This will ensure that any resources maintained on the servers are freed appropriately.

Furthermore, a NAS that receives an Abort-Session-Request (ASR) [RFC6733] MUST issue an Abort-Session-Answer (ASA) if the session identified is active and disconnect the PPP (or tunneling) session.

If accounting is active, an Accounting STOP_RECORD message [RFC6733] MUST be sent upon termination of the session context.

More information on Diameter Session Termination can be found in Sections 8.4 and 8.5 of [RFC6733].

3. Diameter NAS Application Messages

This section defines the Diameter message Command-Code [RFC6733] values that MUST be supported by all Diameter implementations conforming to this specification. The Command Codes are as follows:

Command Name	Abbrev.	Code	Reference
AA-Request	AAR	265	Section 3.1
AA-Answer	AAA	265	Section 3.2
Re-Auth-Request	RAR	258	Section 3.3
Re-Auth-Answer	RAA	258	Section 3.4
Session-Termination-Request	STR	275	Section 3.5
Session-Termination-Answer	STA	275	Section 3.6
Abort-Session-Request	ASR	274	Section 3.7
Abort-Session-Answer	ASA	274	Section 3.8
Accounting-Request	ACR	271	Section 3.9
Accounting-Answer	ACA	271	Section 3.10

Note that the message formats in the following sub-sections use the standard Diameter Command Code Format ([RFC6733], Section 3.2).

3.1. AA-Request (AAR) Command

The AA-Request (AAR), which is indicated by setting the Command-Code field to 265 and the 'R' bit in the Command Flags field, is used to request authentication and/or authorization for a given NAS user. The type of request is identified through the Auth-Request-Type AVP [RFC6733]. The recommended value for most situations is AUTHORIZE_AUTHENTICATE.

If Authentication is requested, the User-Name attribute SHOULD be present, as well as any additional authentication AVPs that would carry the password information. A request for authorization SHOULD only include the information from which the authorization will be performed, such as the User-Name, Called-Station-Id, or Calling-Station-Id AVPs. All requests SHOULD contain AVPs uniquely identifying the source of the call, such as Origin-Host and NAS-Port. Certain networks MAY use different AVPs for authorization purposes. A request for authorization will include some AVPs defined in Section 4.4.

It is possible for a single session to be authorized first and then for an authentication request to follow.

This AA-Request message MAY be the result of a multi-round authentication exchange, which occurs when the AA-Answer message is received with the Result-Code AVP set to DIAMETER_MULTI_ROUND_AUTH. A subsequent AAR message SHOULD be sent, with the User-Password AVP that includes the user's response to the prompt, and MUST include any State AVPs that were present in the AAA message.

Message Format

```
<AA-Request> ::= < Diameter Header: 265, REQ, PXY >
                 < Session-Id >
                 { Auth-Application-Id }
                 { Origin-Host }
                 { Origin-Realm }
                 { Destination-Realm }
                 { Auth-Request-Type }
                 [ Destination-Host ]
                 [ NAS-Identifier ]
                 [ NAS-IP-Address ]
                 [ NAS-IPv6-Address ]
                 [ NAS-Port ]
                 [ NAS-Port-Id ]
                 [ NAS-Port-Type ]
                 [ Origin-AAA-Protocol ]
                 [ Origin-State-Id ]
```

```
                          [ Port-Limit ]
                          [ User-Name ]
                          [ User-Password ]
                          [ Service-Type ]
                          [ State ]
                          [ Authorization-Lifetime ]
                          [ Auth-Grace-Period ]
                          [ Auth-Session-State ]
                          [ Callback-Number ]
                          [ Called-Station-Id ]
                          [ Calling-Station-Id ]
                          [ Originating-Line-Info ]
                          [ Connect-Info ]
                          [ CHAP-Auth ]
                          [ CHAP-Challenge ]
                        * [ Framed-Compression ]
                          [ Framed-Interface-Id ]
                          [ Framed-IP-Address ]
                        * [ Framed-IPv6-Prefix ]
                          [ Framed-IP-Netmask ]
                          [ Framed-MTU ]
                          [ Framed-Protocol ]
                          [ ARAP-Password ]
                          [ ARAP-Security ]
                        * [ ARAP-Security-Data ]
                        * [ Login-IP-Host ]
                        * [ Login-IPv6-Host ]
                          [ Login-LAT-Group ]
                          [ Login-LAT-Node ]
                          [ Login-LAT-Port ]
                          [ Login-LAT-Service ]
                        * [ Tunneling ]
                        * [ Proxy-Info ]
                        * [ Route-Record ]
                        * [ AVP ]
```

Figure 1

3.2. AA-Answer (AAA) Command

The AA-Answer (AAA) message is indicated by setting the Command-Code field to 265 and clearing the 'R' bit in the Command Flags field. It is sent in response to the AA-Request (AAR) message. If authorization was requested, a successful response will include the authorization AVPs appropriate for the service being provided, as defined in Section 4.4.

For authentication exchanges requiring more than a single round trip, the server MUST set the Result-Code AVP to DIAMETER_MULTI_ROUND_AUTH. An AAA message with this result code MAY include one Reply-Message or more and MAY include zero or one State AVPs.

If the Reply-Message AVP was present, the network access server SHOULD send the text to the user's client to display to the user, instructing the client to prompt the user for a response. For example, this can be achieved in PPP via PAP. If it is impossible to deliver the text prompt to the user, the Diameter NAS Application client MUST treat the AA-Answer (AAA) with the Reply-Message AVP as an error and deny access.

Message Format

```
<AA-Answer> ::= < Diameter Header: 265, PXY >
                < Session-Id >
                { Auth-Application-Id }
                { Auth-Request-Type }
                { Result-Code }
                { Origin-Host }
                { Origin-Realm }
                [ User-Name ]
                [ Service-Type ]
              * [ Class ]
              * [ Configuration-Token ]
                [ Acct-Interim-Interval ]
                [ Error-Message ]
                [ Error-Reporting-Host ]
              * [ Failed-AVP ]
                [ Idle-Timeout ]
                [ Authorization-Lifetime ]
                [ Auth-Grace-Period ]
                [ Auth-Session-State ]
                [ Re-Auth-Request-Type ]
                [ Multi-Round-Time-Out ]
                [ Session-Timeout ]
                [ State ]
              * [ Reply-Message ]
                [ Origin-AAA-Protocol ]
                [ Origin-State-Id ]
              * [ Filter-Id ]
                [ Password-Retry ]
                [ Port-Limit ]
                [ Prompt ]
                [ ARAP-Challenge-Response ]
                [ ARAP-Features ]
                [ ARAP-Security ]
```

```
                        * [ ARAP-Security-Data ]
                          [ ARAP-Zone-Access ]
                          [ Callback-Id ]
                          [ Callback-Number ]
                          [ Framed-Appletalk-Link ]
                        * [ Framed-Appletalk-Network ]
                          [ Framed-Appletalk-Zone ]
                        * [ Framed-Compression ]
                          [ Framed-Interface-Id ]
                          [ Framed-IP-Address ]
                        * [ Framed-IPv6-Prefix ]
                          [ Framed-IPv6-Pool ]
                        * [ Framed-IPv6-Route ]
                          [ Framed-IP-Netmask ]
                        * [ Framed-Route ]
                          [ Framed-Pool ]
                          [ Framed-IPX-Network ]
                          [ Framed-MTU ]
                          [ Framed-Protocol ]
                          [ Framed-Routing ]
                        * [ Login-IP-Host ]
                        * [ Login-IPv6-Host ]
                          [ Login-LAT-Group ]
                          [ Login-LAT-Node ]
                          [ Login-LAT-Port ]
                          [ Login-LAT-Service ]
                          [ Login-Service ]
                          [ Login-TCP-Port ]
                        * [ NAS-Filter-Rule ]
                        * [ QoS-Filter-Rule ]
                        * [ Tunneling ]
                        * [ Redirect-Host ]
                          [ Redirect-Host-Usage ]
                          [ Redirect-Max-Cache-Time ]
                        * [ Proxy-Info ]
                        * [ AVP ]
```

Figure 2

3.3. Re-Auth-Request (RAR) Command

A Diameter server can initiate re-authentication and/or re-authorization for a particular session by issuing a Re-Auth-Request (RAR) message [RFC6733].

For example, for pre-paid services, the Diameter server that originally authorized a session may need some confirmation that the user is still using the services.

If a NAS receives an RAR message with Session-Id equal to a currently active session and a Re-Auth-Type that includes authentication, it MUST initiate a re-authentication toward the user, if the service supports this particular feature.

Message Format

```
<RA-Request>  ::= < Diameter Header: 258, REQ, PXY >
                  < Session-Id >
                  { Origin-Host }
                  { Origin-Realm }
                  { Destination-Realm }
                  { Destination-Host }
                  { Auth-Application-Id }
                  { Re-Auth-Request-Type }
                  [ User-Name ]
                  [ Origin-AAA-Protocol ]
                  [ Origin-State-Id ]
                  [ NAS-Identifier ]
                  [ NAS-IP-Address ]
                  [ NAS-IPv6-Address ]
                  [ NAS-Port ]
                  [ NAS-Port-Id ]
                  [ NAS-Port-Type ]
                  [ Service-Type ]
                  [ Framed-IP-Address ]
                  [ Framed-IPv6-Prefix ]
                  [ Framed-Interface-Id ]
                  [ Called-Station-Id ]
                  [ Calling-Station-Id ]
                  [ Originating-Line-Info ]
                  [ Acct-Session-Id ]
                  [ Acct-Multi-Session-Id ]
                  [ State ]
                * [ Class ]
                  [ Reply-Message ]
                * [ Proxy-Info ]
                * [ Route-Record ]
                * [ AVP ]
```

Figure 3

3.4. Re-Auth-Answer (RAA) Command

The Re-Auth-Answer (RAA) message [RFC6733] is sent in response to the RAR. The Result-Code AVP MUST be present and indicates the disposition of the request.

A successful RAA transaction MUST be followed by an AAR message.

Message Format

```
<RA-Answer>  ::= < Diameter Header: 258, PXY >
                 < Session-Id >
                 { Result-Code }
                 { Origin-Host }
                 { Origin-Realm }
                 [ User-Name ]
                 [ Origin-AAA-Protocol ]
                 [ Origin-State-Id ]
                 [ Error-Message ]
                 [ Error-Reporting-Host ]
               * [ Failed-AVP ]
               * [ Redirected-Host ]
                 [ Redirected-Host-Usage ]
                 [ Redirected-Host-Cache-Time ]
                 [ Service-Type ]
               * [ Configuration-Token ]
                 [ Idle-Timeout ]
                 [ Authorization-Lifetime ]
                 [ Auth-Grace-Period ]
                 [ Re-Auth-Request-Type ]
                 [ State ]
               * [ Class ]
               * [ Reply-Message ]
                 [ Prompt ]
               * [ Proxy-Info ]
               * [ AVP ]
```

Figure 4

3.5. Session-Termination-Request (STR) Command

The Session-Termination-Request (STR) message [RFC6733] is sent by the NAS to inform the Diameter Server that an authenticated and/or authorized session is being terminated.

Message Format

```
<ST-Request> ::= < Diameter Header: 275, REQ, PXY >
                 < Session-Id >
                 { Origin-Host }
                 { Origin-Realm }
                 { Destination-Realm }
                 { Auth-Application-Id }
                 { Termination-Cause }
```

```
                    [ User-Name ]
                    [ Destination-Host ]
                  * [ Class ]
                    [ Origin-AAA-Protocol ]
                    [ Origin-State-Id ]
                  * [ Proxy-Info ]
                  * [ Route-Record ]
                  * [ AVP ]
```

Figure 5

3.6. Session-Termination-Answer (STA) Command

The Session-Termination-Answer (STA) message [RFC6733] is sent by the Diameter Server to acknowledge the notification that the session has been terminated. The Result-Code AVP MUST be present and MAY contain an indication that an error occurred while the STR was being serviced.

Upon sending the STA, the Diameter Server MUST release all resources for the session indicated by the Session-Id AVP. Any intermediate server in the Proxy-Chain MAY also release any resources, if necessary.

Message Format

```
   <ST-Answer>  ::= < Diameter Header: 275, PXY >
                    < Session-Id >
                    { Result-Code }
                    { Origin-Host }
                    { Origin-Realm }
                    [ User-Name ]
                  * [ Class ]
                    [ Error-Message ]
                    [ Error-Reporting-Host ]
                  * [ Failed-AVP ]
                    [ Origin-AAA-Protocol ]
                    [ Origin-State-Id ]
                  * [ Redirect-Host ]
                    [ Redirect-Host-Usase ]
                    [ Redirect-Max-Cache-Time ]
                  * [ Proxy-Info ]
                  * [ AVP ]
```

Figure 6

3.7. Abort-Session-Request (ASR) Command

The Abort-Session-Request (ASR) message [RFC6733] can be sent by any Diameter server to the NAS providing session service to request that the session identified by the Session-Id be stopped.

Message Format

```
<AS-Request>  ::= < Diameter Header: 274, REQ, PXY >
                  < Session-Id >
                  { Origin-Host }
                  { Origin-Realm }
                  { Destination-Realm }
                  { Destination-Host }
                  { Auth-Application-Id }
                  [ User-Name ]
                  [ Origin-AAA-Protocol ]
                  [ Origin-State-Id ]
                  [ NAS-Identifier ]
                  [ NAS-IP-Address ]
                  [ NAS-IPv6-Address ]
                  [ NAS-Port ]
                  [ NAS-Port-Id ]
                  [ NAS-Port-Type ]
                  [ Service-Type ]
                  [ Framed-IP-Address ]
                  [ Framed-IPv6-Prefix ]
                  [ Framed-Interface-Id ]
                  [ Called-Station-Id ]
                  [ Calling-Station-Id ]
                  [ Originating-Line-Info ]
                  [ Acct-Session-Id ]
                  [ Acct-Multi-Session-Id ]
                  [ State ]
                * [ Class ]
                * [ Reply-Message ]
                * [ Proxy-Info ]
                * [ Route-Record ]
                * [ AVP ]
```

Figure 7

3.8. Abort-Session-Answer (ASA) Command

The ASA message [RFC6733] is sent in response to the ASR. The Result-Code AVP MUST be present and indicates the disposition of the request.

If the session identified by Session-Id in the ASR was successfully terminated, Result-Code is set to DIAMETER_SUCCESS. If the session

is not currently active, the Result-Code AVP is set to DIAMETER_UNKNOWN_SESSION_ID. If the access device does not stop the session for any other reason, the Result-Code AVP is set to DIAMETER_UNABLE_TO_COMPLY.

Message Format

```
<AS-Answer>  ::= < Diameter Header: 274, PXY >
                 < Session-Id >
                 { Result-Code }
                 { Origin-Host }
                 { Origin-Realm }
                 [ User-Name ]
                 [ Origin-AAA-Protocol ]
                 [ Origin-State-Id ]
                 [ State]
                 [ Error-Message ]
                 [ Error-Reporting-Host ]
               * [ Failed-AVP ]
               * [ Redirected-Host ]
                 [ Redirected-Host-Usage ]
                 [ Redirected-Max-Cache-Time ]
               * [ Proxy-Info ]
               * [ AVP ]
```

Figure 8

3.9. Accounting-Request (ACR) Command

The ACR message [RFC6733] is sent by the NAS to report its session information to a target server downstream.

The Acct-Application-Id AVP MUST be present.

The AVPs listed in the Base protocol specification [RFC6733] MUST be assumed to be present, as appropriate. NAS service-specific accounting AVPs SHOULD be present as described in Section 4.6 and the rest of this specification.

Message Format

```
<AC-Request> ::= < Diameter Header: 271, REQ, PXY >
                 < Session-Id >
                 { Origin-Host }
                 { Origin-Realm }
                 { Destination-Realm }
                 { Accounting-Record-Type }
                 { Accounting-Record-Number }
```

```
                    { Acct-Application-Id }
                    [ User-Name ]
                    [ Accounting-Sub-Session-Id ]
                    [ Acct-Session-Id ]
                    [ Acct-Multi-Session-Id ]
                    [ Origin-AAA-Protocol ]
                    [ Origin-State-Id ]
                    [ Destination-Host ]
                    [ Event-Timestamp ]
                    [ Acct-Delay-Time ]
                    [ NAS-Identifier ]
                    [ NAS-IP-Address ]
                    [ NAS-IPv6-Address ]
                    [ NAS-Port ]
                    [ NAS-Port-Id ]
                    [ NAS-Port-Type ]
                  * [ Class ]
                    [ Service-Type ]
                    [ Termination-Cause ]
                    [ Accounting-Input-Octets ]
                    [ Accounting-Input-Packets ]
                    [ Accounting-Output-Octets ]
                    [ Accounting-Output-Packets ]
                    [ Acct-Authentic ]
                    [ Accounting-Auth-Method ]
                    [ Acct-Link-Count ]
                    [ Acct-Session-Time ]
                    [ Acct-Tunnel-Connection ]
                    [ Acct-Tunnel-Packets-Lost ]
                    [ Callback-Id ]
                    [ Callback-Number ]
                    [ Called-Station-Id ]
                    [ Calling-Station-Id ]
                  * [ Connection-Info ]
                    [ Originating-Line-Info ]
                    [ Authorization-Lifetime ]
                    [ Session-Timeout ]
                    [ Idle-Timeout ]
                    [ Port-Limit ]
                    [ Accounting-Realtime-Required ]
                    [ Acct-Interim-Interval ]
                  * [ Filter-Id ]
                  * [ NAS-Filter-Rule ]
                  * [ QoS-Filter-Rule ]
                    [ Framed-AppleTalk-Link ]
                    [ Framed-AppleTalk-Network ]
                    [ Framed-AppleTalk-Zone ]
                    [ Framed-Compression ]
```

```
                        [ Framed-Interface-Id ]
                        [ Framed-IP-Address ]
                        [ Framed-IP-Netmask ]
                      * [ Framed-IPv6-Prefix ]
                        [ Framed-IPv6-Pool ]
                      * [ Framed-IPv6-Route ]
                        [ Framed-IPX-Network ]
                        [ Framed-MTU ]
                        [ Framed-Pool ]
                        [ Framed-Protocol ]
                      * [ Framed-Route ]
                        [ Framed-Routing ]
                      * [ Login-IP-Host ]
                      * [ Login-IPv6-Host ]
                        [ Login-LAT-Group ]
                        [ Login-LAT-Node ]
                        [ Login-LAT-Port ]
                        [ Login-LAT-Service ]
                        [ Login-Service ]
                        [ Login-TCP-Port ]
                      * [ Tunneling ]
                      * [ Proxy-Info ]
                      * [ Route-Record ]
                      * [ AVP ]
```

Figure 9

3.10. Accounting-Answer (ACA) Command

The ACA message [RFC6733] is used to acknowledge an Accounting-Request command. The Accounting-Answer command contains the same Session-Id as the Request.

Only the target Diameter Server or home Diameter Server SHOULD respond with the Accounting-Answer command.

The Acct-Application-Id AVP MUST be present.

The AVPs listed in the Base protocol specification [RFC6733] MUST be assumed to be present, as appropriate. NAS service-specific accounting AVPs SHOULD be present as described in Section 4.6 and the rest of this specification.

Message Format

```
<AC-Answer> ::= < Diameter Header: 271, PXY >
                < Session-Id >
                { Result-Code }
```

```
                          { Origin-Host }
                          { Origin-Realm }
                          { Accounting-Record-Type }
                          { Accounting-Record-Number }
                          { Acct-Application-Id }
                          [ User-Name ]
                          [ Accounting-Sub-Session-Id ]
                          [ Acct-Session-Id ]
                          [ Acct-Multi-Session-Id ]
                          [ Event-Timestamp ]
                          [ Error-Message ]
                          [ Error-Reporting-Host ]
                        * [ Failed-AVP ]
                          [ Origin-AAA-Protocol ]
                          [ Origin-State-Id ]
                          [ NAS-Identifier ]
                          [ NAS-IP-Address ]
                          [ NAS-IPv6-Address ]
                          [ NAS-Port ]
                          [ NAS-Port-Id ]
                          [ NAS-Port-Type ]
                          [ Service-Type ]
                          [ Termination-Cause ]
                          [ Accounting-Realtime-Required ]
                          [ Acct-Interim-Interval ]
                        * [ Class ]
                        * [ Proxy-Info ]
                        * [ AVP ]
```

Figure 10

4. Diameter NAS Application AVPs

The following sections define a new derived AVP data format, a set of application-specific AVPs and describe the use of AVPs defined in other documents by the Diameter NAS Application.

4.1. Derived AVP Data Formats

4.1.1. QoSFilterRule

The QosFilterRule format is derived from the OctetString AVP Base Format. It uses the ASCII charset. Packets may be marked or metered based on the following information:

- Direction (in or out)
- Source and destination IP address (possibly masked)

- Protocol
- Source and destination port (lists or ranges)
- DSCP values (no mask or range)

Rules for the appropriate direction are evaluated in order; the first matched rule terminates the evaluation. Each packet is evaluated once. If no rule matches, the packet is treated as best effort. An access device unable to interpret or apply a QoS rule SHOULD NOT terminate the session.

QoSFilterRule filters MUST follow the following format:

```
action dir proto from src to dst [options]

where


action

            tag      Mark packet with a specific DSCP [RFC2474]

            meter    Meter traffic


dir         The format is as described under IPFilterRule
            [RFC6733]


proto       The format is as described under IPFilterRule
            [RFC6733]


src and dst The format is as described under IPFilterRule
            [RFC6733]
```

The options are described in Section 4.4.9.

The rule syntax is a modified subset of ipfw(8) from FreeBSD, and the ipfw.c code may provide a useful base for implementations.

4.2. NAS Session AVPs

Diameter reserves the AVP Codes 0 - 255 for RADIUS Attributes that are implemented in Diameter.

4.2.1. Call and Session Information

This section describes the AVPs specific to Diameter applications that are needed to identify the call and session context and status information. On a request, this information allows the server to qualify the session.

These AVPs are used in addition to the following AVPs from the base protocol specification [RFC6733]:

```
Session-Id
Auth-Application-Id
Origin-Host
Origin-Realm
Auth-Request-Type
Termination-Cause
```

The following table gives the possible flag values for the session level AVPs.

Attribute Name	Section Defined	AVP Flag Rules: MUST	AVP Flag Rules: MUST NOT
NAS-Port	4.2.2	M	V
NAS-Port-Id	4.2.3	M	V
NAS-Port-Type	4.2.4	M	V
Called-Station-Id	4.2.5	M	V
Calling-Station-Id	4.2.6	M	V
Connect-Info	4.2.7	M	V
Originating-Line-Info	4.2.8	M	V
Reply-Message	4.2.9	M	V

4.2.2. NAS-Port AVP

The NAS-Port AVP (AVP Code 5) is of type Unsigned32 and contains the physical or virtual port number of the NAS which is authenticating the user. Note that "port" is meant in its sense as a service connection on the NAS, not as an IP protocol identifier, and hence the format and contents of the string that identifies the port are specific to the NAS implementation.

Either the NAS-Port AVP or the NAS-Port-Id AVP (Section 4.2.3) SHOULD be present in the AA-Request (AAR, Section 3.1) command if the NAS differentiates among its ports.

4.2.3. NAS-Port-Id AVP

The NAS-Port-Id AVP (AVP Code 87) is of type UTF8String and consists of 7-bit ASCII text identifying the port of the NAS authenticating the user. Note that "port" is meant in its sense as a service connection on the NAS, not as an IP protocol identifier.

Either the NAS-Port-Id AVP or the NAS-Port AVP (Section 4.2.2) SHOULD be present in the AA-Request (AAR, Section 3.1) command if the NAS differentiates among its ports. NAS-Port-Id is intended for use by NASes that cannot conveniently number their ports.

4.2.4. NAS-Port-Type AVP

The NAS-Port-Type AVP (AVP Code 61) is of type Enumerated and contains the type of the port on which the NAS is authenticating the user. This AVP SHOULD be present if the NAS uses the same NAS-Port number ranges for different service types concurrently.

The currently supported values of the NAS-Port-Type AVP are listed in [RADIUSAttrVals].

4.2.5. Called-Station-Id AVP

The Called-Station-Id AVP (AVP Code 30) is of type UTF8String contains a 7-bit ASCII string sent by the NAS to describe the Layer 2 address the user contacted in the request. For dialup access, this can be a phone number obtained by using the Dialed Number Identification Service (DNIS) or a similar technology. Note that this may be different from the phone number the call comes in on. For use with IEEE 802 access, the Called-Station-Id MAY contain a MAC address formatted as described in Congdon, et al. [RFC3580].

If the Called-Station-Id AVP is present in an AAR message, Auth-Request-Type AVP is set to AUTHORIZE_ONLY and the User-Name AVP is absent, the Diameter Server MAY perform authorization based on this AVP. This can be used by a NAS to request whether a call should be answered based on the DNIS result.

Further codification of this field’s allowed content and usage is outside the scope of this specification.

4.2.6. Calling-Station-Id AVP

The Calling-Station-Id AVP (AVP Code 31) is of type UTF8String and contains a 7-bit ASCII string sent by the NAS to describe the Layer 2 address from which the user connected in the request. For dialup access, this is the phone number the call came from, using Automatic Number Identification (ANI) or a similar technology. For use with IEEE 802 access, the Calling-Station-Id AVP MAY contain a MAC address, formated as described in RFC 3580.

If the Calling-Station-Id AVP is present in an AAR message, the Auth-Request-Type AVP is set to AUTHORIZE_ONLY and the User-Name AVP is absent, the Diameter Server MAY perform authorization based on the value of this AVP. This can be used by a NAS to request whether a call should be answered based on the Layer 2 address (ANI, MAC Address, etc.)

Further codification of this field’s allowed content and usage is outside the scope of this specification.

4.2.7. Connect-Info AVP

The Connect-Info AVP (AVP Code 77) is of type UTF8String and is sent in the AA-Request message or an ACR message with the value of the Accounting-Record-Type AVP set to STOP. When sent in the AA-Request, it indicates the nature of the user’s connection. The connection speed SHOULD be included at the beginning of the first Connect-Info AVP in the message. If the transmit and receive connection speeds differ, both may be included in the first AVP with the transmit speed listed first (the speed at which the NAS modem transmits), then a slash (/), then the receive speed, and then other optional information.

For example: "28800 V42BIS/LAPM" or "52000/31200 V90"

If sent in an ACR message with the value of the Accounting-Record-Type AVP set to STOP, this attribute may summarize statistics relating to session quality. For example, in IEEE 802.11, the Connect-Info AVP may contain information on the number of link layer retransmissions. The exact format of this attribute is implementation specific.

4.2.8. Originating-Line-Info AVP

The Originating-Line-Info AVP (AVP Code 94) is of type OctetString and is sent by the NAS system to convey information about the origin of the call from an SS7 system.

The Originating Line Information (OLI) element indicates the nature and/or characteristics of the line from which a call originated

(e.g., pay phone, hotel, cellular). Telephone companies are starting to offer OLI to their customers as an option over Primary Rate Interface (PRI). Internet Service Providers (ISPs) can use OLI in addition to Called-Station-Id and Calling-Station-Id attributes to differentiate customer calls and to define different services.

The Value field contains two octets (00 - 99). ANSI T1.113 and BELLCORE 394 can be used for additional information about these values and their use. For information on the currently assigned values, see [ANITypes].

4.2.9. Reply-Message AVP

The Reply-Message AVP (AVP Code 18) is of type UTF8String and contains text that MAY be displayed to the user. When used in an AA-Answer message with a successful Result-Code AVP, it indicates success. When found in an AAA message with a Result-Code other than DIAMETER_SUCCESS, the AVP contains a failure message.

The Reply-Message AVP MAY contain text to prompt the user before another AA-Request attempt. When used in an AA-Answer message containing a Result-Code AVP with the value DIAMETER_MULTI_ROUND_AUTH or in an Re-Auth-Request message, it MAY contain text to prompt the user for a response.

4.3. NAS Authentication AVPs

This section defines the AVPs necessary to carry the authentication information in the Diameter protocol. The functionality defined here provides a RADIUS-like AAA service [RFC2865] over a more reliable and secure transport, as defined in the base protocol [RFC6733].

The following table gives the possible flag values for the session level AVPs.

		AVP Flag rules	
Attribute Name	Section Defined	MUST	MUST NOT
User-Password	4.3.1	M	V
Password-Retry	4.3.2	M	V
Prompt	4.3.3	M	V
CHAP-Auth	4.3.4	M	V
CHAP-Algorithm	4.3.5	M	V
CHAP-Ident	4.3.6	M	V

CHAP-Response	4.3.7	M	V
CHAP-Challenge	4.3.8	M	V
ARAP-Password	4.3.9	M	V
ARAP-Challenge-Response	4.3.10	M	V
ARAP-Security	4.3.11	M	V
ARAP-Security-Data	4.3.12	M	V

4.3.1. User-Password AVP

The User-Password AVP (AVP Code 2) is of type OctetString and contains the password of the user to be authenticated, or the user's input in a multi-round authentication exchange.

The User-Password AVP contains a user password or one-time password and therefore represents sensitive information. As required by Fajardo, et al. [RFC6733], Diameter messages are encrypted by using IPsec [RFC4301] or TLS [RFC5246]. Unless this AVP is used for one-time passwords, the User-Password AVP SHOULD NOT be used in untrusted proxy environments without encrypting it by using end-to-end security techniques.

The clear-text password (prior to encryption) MUST NOT be longer than 128 bytes in length.

4.3.2. Password-Retry AVP

The Password-Retry AVP (AVP Code 75) is of type Unsigned32 and MAY be included in the AA-Answer if the Result-Code indicates an authentication failure. The value of this AVP indicates how many authentication attempts a user is permitted before being disconnected. This AVP is primarily intended for use when the Framed-Protocol AVP (Section 4.4.10.1) is set to ARAP.

4.3.3. Prompt AVP

The Prompt AVP (AVP Code 76) is of type Enumerated and MAY be present in the AA-Answer message. When present, it is used by the NAS to determine whether the user's response, when entered, should be echoed.

The supported values are listed in [RADIUSAttrVals]

4.3.4. CHAP-Auth AVP

The CHAP-Auth AVP (AVP Code 402) is of type Grouped and contains the information necessary to authenticate a user using the PPP Challenge-Handshake Authentication Protocol (CHAP) [RFC1994]. If the CHAP-Auth AVP is found in a message, the CHAP-Challenge AVP (Section 4.3.8) MUST be present as well. The optional AVPs containing the CHAP response depend upon the value of the CHAP-Algorithm AVP (Section 4.3.8). The grouped AVP has the following ABNF grammar:

```
CHAP-Auth  ::= < AVP Header: 402 >
               { CHAP-Algorithm }
               { CHAP-Ident }
               [ CHAP-Response ]
             * [ AVP ]
```

4.3.5. CHAP-Algorithm AVP

The CHAP-Algorithm AVP (AVP Code 403) is of type Enumerated and contains the algorithm identifier used in the computation of the CHAP response [RFC1994]. The following values are currently supported:

CHAP with MD5 5

The CHAP response is computed by using the procedure described in [RFC1994] This algorithm requires that the CHAP-Response AVP (Section 4.3.7) MUST be present in the CHAP-Auth AVP (Section 4.3.4).

4.3.6. CHAP-Ident AVP

The CHAP-Ident AVP (AVP Code 404) is of type OctetString and contains the 1 octet CHAP Identifier used in the computation of the CHAP response [RFC1994]

4.3.7. CHAP-Response AVP

The CHAP-Response AVP (AVP Code 405) is of type OctetString and contains the 16 octet authentication data provided by the user in response to the CHAP challenge [RFC1994].

4.3.8. CHAP-Challenge AVP

The CHAP-Challenge AVP (AVP Code 60) is of type OctetString and contains the CHAP Challenge sent by the NAS to the CHAP peer [RFC1994].

4.3.9. ARAP-Password AVP

The ARAP-Password AVP (AVP Code 70) is of type OctetString and is only present when the Framed-Protocol AVP (Section 4.4.10.1) is included in the message and is set to ARAP. This AVP MUST NOT be present if either the User-Password or the CHAP-Auth AVP is present. See Rigney, et al. [RFC2869] for more information on the contents of this AVP.

4.3.10. ARAP-Challenge-Response AVP

The ARAP-Challenge-Response AVP (AVP Code 84) is of type OctetString and is only present when the Framed-Protocol AVP (Section 4.4.10.1) is included in the message and is set to ARAP. This AVP contains an 8 octet response to the dial-in client's challenge. The Diameter server calculates this value by taking the dial-in client's challenge from the high-order 8 octets of the ARAP-Password AVP and performing DES encryption on this value with the authenticating user's password as the key. If the user's password is fewer than 8 octets in length, the password is padded at the end with NULL octets to a length of 8 before it is used as a key.

4.3.11. ARAP-Security AVP

The ARAP-Security AVP (AVP Code 73) is of type Unsigned32 and MAY be present in the AA-Answer message if the Framed-Protocol AVP (Section 4.4.10.1) is set to the value of ARAP, and the Result-Code AVP ([RFC6733], Section 7.1) is set to DIAMETER_MULTI_ROUND_AUTH. See RFC 2869 for more information on the contents of this AVP.

4.3.12. ARAP-Security-Data AVP

The ARAP-Security-Data AVP (AVP Code 74) is of type OctetString and MAY be present in the AA-Request or AA-Answer message if the Framed-Protocol AVP (Section 4.4.10.1) is set to the value of ARAP and the Result-Code AVP ([RFC6733], Section 7.1) is set to DIAMETER_MULTI_ROUND_AUTH. This AVP contains the security module challenge or response associated with the ARAP Security Module specified in the ARAP-Security AVP (Section 4.3.11).

4.4. NAS Authorization AVPs

This section contains the authorization AVPs supported in the NAS Application. The Service-Type AVP SHOULD be present in all messages and, based on its value, additional AVPs defined in this section and Section 4.5 MAY be present.

The following table gives the possible flag values for the session-level AVPs.

Attribute Name	Section Defined	AVP Flag rules: MUST	AVP Flag rules: MUST NOT
Service-Type	4.4.1	M	V
Callback-Number	4.4.2	M	V
Callback-Id	4.4.3	M	V
Idle-Timeout	4.4.4	M	V
Port-Limit	4.4.5	M	V
NAS-Filter-Rule	4.4.6	M	V
Filter-Id	4.4.7	M	V
Configuration-Token	4.4.8	M	V
QoS-Filter-Rule	4.4.9		
Framed-Protocol	4.4.10.1	M	V
Framed-Routing	4.4.10.2	M	V
Framed-MTU	4.4.10.3	M	V
Framed-Compression	4.4.10.4	M	V
Framed-IP-Address	4.4.10.5.1	M	V
Framed-IP-Netmask	4.4.10.5.2	M	V
Framed-Route	4.4.10.5.3	M	V
Framed-Pool	4.4.10.5.4	M	V
Framed-Interface-Id	4.4.10.5.5	M	V
Framed-IPv6-Prefix	4.4.10.5.6	M	V
Framed-IPv6-Route	4.4.10.5.7	M	V
Framed-IPv6-Pool	4.4.10.5.8	M	V
Framed-IPX-Network	4.4.10.6.1	M	V
Framed-Appletalk-Link	4.4.10.7.1	M	V
Framed-Appletalk-Network	4.4.10.7.2	M	V
Framed-Appletalk-Zone	4.4.10.7.3	M	V
ARAP-Features	4.4.10.8.1	M	V
ARAP-Zone-Access	4.4.10.8.2	M	V
Login-IP-Host	4.4.11.1	M	V
Login-IPv6-Host	4.4.11.2	M	V
Login-Service	4.4.11.3	M	V
Login-TCP-Port	4.4.11.4.1	M	V
Login-LAT-Service	4.4.11.5.1	M	V
Login-LAT-Node	4.4.11.5.2	M	V
Login-LAT-Group	4.4.11.5.3	M	V
Login-LAT-Port	4.4.11.5.4	M	V

4.4.1. Service-Type AVP

The Service-Type AVP (AVP Code 6) is of type Enumerated and contains the type of service the user has requested or the type of service to be provided. One such AVP MAY be present in an authentication and/or authorization request or response. A NAS is not required to implement all of these service types. It MUST treat unknown or unsupported Service-Types received in a response as a failure and end the session with a DIAMETER_INVALID_AVP_VALUE Result-Code.

When used in a request, the Service-Type AVP SHOULD be considered a hint to the server that the NAS believes the user would prefer the kind of service indicated. The server is not required to honor the hint. Furthermore, if the service specified by the server is supported, but not compatible with the current mode of access, the NAS MUST fail to start the session. The NAS MUST also generate the appropriate error message(s).

The complete list of defined values that the Service-Type AVP can take can be found in Rigney, et al. [RFC2865] and and the relevant IANA registry [RADIUSAttrVals], but the following values require further qualification here:

Login (1)

> The user should be connected to a host. The message MAY include additional AVPs as defined in Section 4.4.11.4 or Section 4.4.11.5.

Framed (2)

> A Framed Protocol, such as PPP or SLIP, should be started for the User. The message MAY include additional AVPs defined in Section 4.4.10, or Section 4.5 for tunneling services.

Callback Login (3)

> The user should be disconnected and called back, then connected to a host. The message MAY include additional AVPs defined in this Section.

Callback Framed (4)

The user should be disconnected and called back, and then a Framed Protocol, such as PPP or SLIP, should be started for the user. The message MAY include additional AVPs defined in Section 4.4.10, or Section 4.5 for tunneling services.

4.4.2. Callback-Number AVP

The Callback-Number AVP (AVP Code 19) is of type UTF8String and contains a dialing string to be used for callback, the format of which is deployment-specific. The Callback-Number AVP MAY be used in an authentication and/or authorization request as a hint to the server that a callback service is desired, but the server is not required to honor the hint in the corresponding response.

Any further codification of this field's allowed usage range is outside the scope of this specification.

4.4.3. Callback-Id AVP

The Callback-Id AVP (AVP Code 20) is of type UTF8String and contains the name of a place to be called, to be interpreted by the NAS. This AVP MAY be present in an authentication and/or authorization response.

This AVP is not roaming-friendly as it assumes that the Callback-Id is configured on the NAS. Using the Callback-Number AVP (Section 4.4.2) is therefore RECOMMENDED.

4.4.4. Idle-Timeout AVP

The Idle-Timeout AVP (AVP Code 28) is of type Unsigned32 and sets the maximum number of consecutive seconds of idle connection allowable to the user before termination of the session or before a prompt is issued. The default is none, or system specific.

4.4.5. Port-Limit AVP

The Port-Limit AVP (AVP Code 62) is of type Unsigned32 and sets the maximum number of ports the NAS provides to the user. It MAY be used in an authentication and/or authorization request as a hint to the server that multilink PPP [RFC1990] service is desired, but the server is not required to honor the hint in the corresponding response.

4.4.6. NAS-Filter-Rule AVP

The NAS-Filter-Rule AVP (AVP Code 400) is of type IPFilterRule and provides filter rules that need to be configured on the NAS for the user. One or more of these AVPs MAY be present in an authorization response.

4.4.7. Filter-Id AVP

The Filter-Id AVP (AVP Code 11) is of type UTF8String and contains the name of the filter list for this user. It is intended to be human-readable. Zero or more Filter-Id AVPs MAY be sent in an authorization answer message.

Identifying a filter list by name allows the filter to be used on different NASes without regard to filter-list implementation details. However, this AVP is not roaming-friendly, as filter naming differs from one service provider to another.

In environments where backward compatibility with RADIUS is not required, it is RECOMMENDED that the NAS-Filter-Rule AVP (Section 4.4.6) be used instead.

4.4.8. Configuration-Token AVP

The Configuration-Token AVP (AVP Code 78) is of type OctetString and is sent by a Diameter Server to a Diameter Proxy Agent in an AA-Answer command to indicate a type of user profile to be used. It should not be sent to a Diameter Client (NAS).

The format of the Data field of this AVP is site specific.

4.4.9. QoS-Filter-Rule AVP

The QoS-Filter-Rule AVP (AVP Code 407) is of type QoSFilterRule (Section 4.1.1) and provides QoS filter rules that need to be configured on the NAS for the user. One or more such AVPs MAY be present in an authorization response.

The use of this AVP is NOT RECOMMENDED; the AVPs defined by Korhonen, et al. [RFC5777] SHOULD be used instead.

The following options are defined for the QoSFilterRule filters:

DSCP <color>

> If action is set to tag (Section 4.1.1) this option MUST be included in the rule.

Color values are defined in Nichols, et al. [RFC2474]. Exact matching of DSCP values is required (no masks or ranges).

metering <rate> <color_under> <color_over>

The metering option provides Assured Forwarding, as defined in Heinanen, et al. [RFC2597]. and MUST be present if the action is set to meter (Section 4.1.1) The rate option is the throughput, in bits per second, used by the access device to mark packets. Traffic over the rate is marked with the color_over codepoint, and traffic under the rate is marked with the color_under codepoint. The color_under and color_over options contain the drop preferences and MUST conform to the recommended codepoint keywords described in RFC 2597 (e.g., AF13).

The metering option also supports the strict limit on traffic required by Expedited Forwarding, as defined in Davie, et al. [RFC3246]. The color_over option may contain the keyword "drop" to prevent forwarding of traffic that exceeds the rate parameter.

4.4.10. Framed Access Authorization AVPs

This section lists the authorization AVPs necessary to support framed access, such as PPP and SLIP. AVPs defined in this section MAY be present in a message if the Service-Type AVP was set to "Framed" or "Callback Framed".

4.4.10.1. Framed-Protocol AVP

The Framed-Protocol AVP (AVP Code 7) is of type Enumerated and contains the framing to be used for framed access. This AVP MAY be present in both requests and responses. The supported values are listed in [RADIUSAttrVals].

4.4.10.2. Framed-Routing AVP

The Framed-Routing AVP (AVP Code 10) is of type Enumerated and contains the routing method for the user when the user is a router to a network. This AVP SHOULD only be present in authorization responses. The supported values are listed in [RADIUSAttrVals].

4.4.10.3. Framed-MTU AVP

The Framed-MTU AVP (AVP Code 12) is of type Unsigned32 and contains the Maximum Transmission Unit (MTU) to be configured for the user, when it is not negotiated by some other means (such as PPP). This AVP SHOULD only be present in authorization responses. The MTU value MUST be in the range from 64 to 65535.

4.4.10.4. Framed-Compression AVP

The Framed-Compression AVP (AVP Code 13) is of type Enumerated and contains the compression protocol to be used for the link. It MAY be used in an authorization request as a hint to the server that a specific compression type is desired, but the server is not required to honor the hint in the corresponding response.

More than one compression protocol AVP MAY be sent. The NAS is responsible for applying the proper compression protocol to the appropriate link traffic.

The supported values are listed in [RADIUSAttrVals].

4.4.10.5. IP Access Authorization AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access service to IP.

4.4.10.5.1. Framed-IP-Address AVP

The Framed-IP-Address AVP (AVP Code 8) [RFC2865] is of type OctetString and contains an IPv4 address of the type specified in the attribute value to be configured for the user. It MAY be used in an authorization request as a hint to the server that a specific address is desired, but the server is not required to honor the hint in the corresponding response.

Two values have special significance: 0xFFFFFFFF and 0xFFFFFFFE. The value 0xFFFFFFFF indicates that the NAS should allow the user to select an address (i.e., negotiated). The value 0xFFFFFFFE indicates that the NAS should select an address for the user (e.g., assigned from a pool of addresses kept by the NAS).

4.4.10.5.2. Framed-IP-Netmask AVP

The Framed-IP-Netmask AVP (AVP Code 9) is of type OctetString and contains the four octets of the IPv4 netmask to be configured for the user when the user is a router to a network. It MAY be used in an authorization request as a hint to the server that a specific netmask

is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST be present in a response if the request included this AVP with a value of 0xFFFFFFFF.

4.4.10.5.3. Framed-Route AVP

The Framed-Route AVP (AVP Code 22) is of type UTF8String and contains the 7-bit ASCII routing information to be configured for the user on the NAS. Zero or more of these AVPs MAY be present in an authorization response.

The string MUST contain a destination prefix in dotted quad form optionally followed by a slash and a decimal length specifier stating how many high-order bits of the prefix should be used. This is followed by a space, a gateway address in dotted quad form, a space, and one or more metrics separated by spaces; for example,

"192.0.2.0/24 192.0.2.1 1"

The length specifier may be omitted, in which case it should default to 8 bits for class A prefixes, to 16 bits for class B prefixes, and to 24 bits for class C prefixes; for example,

"192.0.2.0 192.0.2.1 1"

Whenever the gateway address is specified as "0.0.0.0" the IP address of the user SHOULD be used as the gateway address.

4.4.10.5.4. Framed-Pool AVP

The Framed-Pool AVP (AVP Code 88) is of type OctetString and contains the name of an assigned address pool that SHOULD be used to assign an address for the user. If a NAS does not support multiple address pools, the NAS SHOULD ignore this AVP. Address pools are usually used for IP addresses but can be used for other protocols if the NAS supports pools for those protocols.

Although specified as type OctetString for compatibility with RADIUS [RFC2869], the encoding of the Data field SHOULD also conform to the rules for the UTF8String Data Format.

4.4.10.5.5. Framed-Interface-Id AVP

The Framed-Interface-Id AVP (AVP Code 96) is of type Unsigned64 and contains the IPv6 interface identifier to be configured for the user. It MAY be used in authorization requests as a hint to the server that a specific interface id is desired, but the server is not required to honor the hint in the corresponding response.

4.4.10.5.6. Framed-IPv6-Prefix AVP

The Framed-IPv6-Prefix AVP (AVP Code 97) is of type OctetString and contains the IPv6 prefix to be configured for the user. One or more AVPs MAY be used in authorization requests as a hint to the server that specific IPv6 prefixes are desired, but the server is not required to honor the hint in the corresponding response.

4.4.10.5.7. Framed-IPv6-Route AVP

The Framed-IPv6-Route AVP (AVP Code 99) is of type UTF8String and contains the ASCII routing information to be configured for the user on the NAS. Zero or more of these AVPs MAY be present in an authorization response.

The string MUST contain an IPv6 address prefix followed by a slash and a decimal length specifier stating how many high order bits of the prefix should be used. This is followed by a space, a gateway address in hexadecimal notation, a space, and one or more metrics separated by spaces; for example,

"2001:db8::/32 2001:db8:106:a00:20ff:fe99:a998 1"

Whenever the gateway address is the IPv6 unspecified address, the IP address of the user SHOULD be used as the gateway address, such as in:

"2001:db8::/32 :: 1"

4.4.10.5.8. Framed-IPv6-Pool AVP

The Framed-IPv6-Pool AVP (AVP Code 100) is of type OctetString and contains the name of an assigned pool that SHOULD be used to assign an IPv6 prefix for the user. If the access device does not support multiple prefix pools, it MUST ignore this AVP.

Although specified as type OctetString for compatibility with RADIUS [RFC3162], the encoding of the Data field SHOULD also conform to the rules for the UTF8String Data Format.

4.4.10.6. IPX Access AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access to an IPX network service [IPX].

4.4.10.6.1. Framed-IPX-Network AVP

The Framed-IPX-Network AVP (AVP Code 23) is of type Unsigned32 and contains the IPX Network number to be configured for the user. It MAY be used in an authorization request as a hint to the server that a specific address is desired, but the server is not required to honor the hint in the corresponding response.

Two addresses have special significance: 0xFFFFFFFF and 0xFFFFFFFE. The value 0xFFFFFFFF indicates that the NAS should allow the user to select an address (i.e., Negotiated). The value 0xFFFFFFFE indicates that the NAS should select an address for the user (e.g., assign it from a pool of one or more IPX networks kept by the NAS).

4.4.10.7. AppleTalk Network Access AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access to an AppleTalk network [AppleTalk].

4.4.10.7.1. Framed-AppleTalk-Link AVP

The Framed-AppleTalk-Link AVP (AVP Code 37) is of type Unsigned32 and contains the AppleTalk network number that should be used for the serial link to the user, which is another AppleTalk router. This AVP MUST only be present in an authorization response and is never used when the user is not another router.

Despite the size of the field, values range from 0 to 65,535. The special value of 0 indicates an unnumbered serial link. A value of 1 to 65,535 means that the serial line between the NAS and the user should be assigned that value as an AppleTalk network number.

4.4.10.7.2. Framed-AppleTalk-Network AVP

The Framed-AppleTalk-Network AVP (AVP Code 38) is of type Unsigned32 and contains the AppleTalk Network number that the NAS should probe to allocate an AppleTalk node for the user. This AVP MUST only be present in an authorization response and is never used when the user is not another router. Multiple instances of this AVP indicate that the NAS may probe, using any of the network numbers specified.

Despite the size of the field, values range from 0 to 65,535. The special value 0 indicates that the NAS should assign a network for the user, using its default cable range. A value between 1 and 65,535 (inclusive) indicates to the AppleTalk Network that the NAS should probe to find an address for the user.

4.4.10.7.3. Framed-AppleTalk-Zone AVP

The Framed-AppleTalk-Zone AVP (AVP Code 39) is of type OctetString and contains the AppleTalk Default Zone to be used for this user. This AVP MUST only be present in an authorization response. Multiple instances of this AVP in the same message are not allowed.

The codification of this field's allowed range is outside the scope of this specification.

4.4.10.8. AppleTalk Remote Access AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access to the AppleTalk network via the AppleTalk Remote Access Protocol [ARAP]. They are only present if the Framed-Protocol AVP (Section 4.4.10.1) is set to ARAP. Section 2.2 of RFC 2869 describes the operational use of these attributes.

4.4.10.8.1. ARAP-Features AVP

The ARAP-Features AVP (AVP Code 71) is of type OctetString and MAY be present in the AA-Accept message if the Framed-Protocol AVP is set to the value of ARAP. See RFC 2869 for more information about the format of this AVP.

4.4.10.8.2. ARAP-Zone-Access AVP

The ARAP-Zone-Access AVP (AVP Code 72) is of type Enumerated and MAY be present in the AA-Accept message if the Framed-Protocol AVP is set to the value of ARAP.

The supported values are listed in [RADIUSAttrVals] and defined in RFC 2869.

4.4.11. Non-Framed Access Authorization AVPs

This section contains the authorization AVPs that are needed to support terminal server functionality. AVPs defined in this section MAY be present in a message if the Service-Type AVP was set to "Login" or "Callback Login".

4.4.11.1. Login-IP-Host AVP

The Login-IP-Host AVP (AVP Code 14) [RFC2865] is of type OctetString and contains the IPv4 address of a host with which to connect the user when the Login-Service AVP is included. It MAY be used in an AA-Request command as a hint to the Diameter Server that a specific host is desired, but the Diameter Server is not required to honor the hint in the AA-Answer.

Two addresses have special significance: all ones and 0. The value of all ones indicates that the NAS SHOULD allow the user to select an address. The value 0 indicates that the NAS SHOULD select a host to connect the user to.

4.4.11.2. Login-IPv6-Host AVP

The Login-IPv6-Host AVP (AVP Code 98) [RFC3162] is of type OctetString and contains the IPv6 address of a host with which to connect the user when the Login-Service AVP is included. It MAY be used in an AA-Request command as a hint to the Diameter Server that a specific host is desired, but the Diameter Server is not required to honor the hint in the AA-Answer.

Two addresses have special significance, 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF and 0. The value 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF indicates that the NAS SHOULD allow the user to select an address. The value 0 indicates that the NAS SHOULD select a host to connect the user to.

4.4.11.3. Login-Service AVP

The Login-Service AVP (AVP Code 15) is of type Enumerated and contains the service that should be used to connect the user to the login host. This AVP SHOULD only be present in authorization responses. The supported values are listed in RFC 2869.

4.4.11.4. TCP Services

The AVP described in the following section MAY be present if the Login-Service AVP is set to Telnet, Rlogin, TCP Clear, or TCP Clear Quiet.

4.4.11.4.1. Login-TCP-Port AVP

The Login-TCP-Port AVP (AVP Code 16) is of type Unsigned32 and contains the TCP port with which the user is to be connected when the Login-Service AVP is also present. This AVP SHOULD only be present in authorization responses. The value MUST NOT be greater than 65,535.

4.4.11.5. LAT Services

The AVPs described in this section MAY be present if the Login-Service AVP is set to LAT [LAT].

4.4.11.5.1. Login-LAT-Service AVP

The Login-LAT-Service AVP (AVP Code 34) is of type OctetString and contains the system with which the user is to be connected by LAT. It MAY be used in an authorization request as a hint to the server that a specific service is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in the response if the Login-Service AVP states that LAT is desired.

Administrators use this service attribute when dealing with clustered systems. In these environments, several different time-sharing hosts share the same resources (disks, printers, etc.), and administrators often configure each host to offer access (service) to each of the shared resources. In this case, each host in the cluster advertises its services through LAT broadcasts.

Sophisticated users often know which service providers (machines) are faster and tend to use a node name when initiating a LAT connection. Some administrators want particular users to use certain machines as a primitive form of load balancing (although LAT knows how to do load balancing itself).

The String field contains the identity of the LAT service to use. The LAT Architecture allows this string to contain $ (dollar), - (hyphen), . (period), _ (underscore), numerics, upper- and lowercase alphabetics, and the ISO Latin-1 character set extension [ISO.8859-1.1987]. All LAT string comparisons are case insensitive.

4.4.11.5.2. Login-LAT-Node AVP

The Login-LAT-Node AVP (AVP Code 35) is of type OctetString and contains the Node with which the user is to be automatically connected by LAT. It MAY be used in an authorization request as a hint to the server that a specific LAT node is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in a response if the Login-Service-Type AVP is set to LAT.

The String field contains the identity of the LAT service to use. The LAT Architecture allows this string to contain $ (dollar), - (hyphen), . (period), _ (underscore), numerics, upper- and lowercase alphabetics, and the ISO Latin-1 character set extension [ISO.8859-1.1987]. All LAT string comparisons are case insensitive.

4.4.11.5.3. Login-LAT-Group AVP

The Login-LAT-Group AVP (AVP Code 36) is of type OctetString and contains a string identifying the LAT group codes this user is authorized to use. It MAY be used in an authorization request as a hint to the server that a specific group is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in a response if the Login-Service-Type AVP is set to LAT.

LAT supports 256 different group codes, which LAT uses as a form of access rights. LAT encodes the group codes as a 256-bit bitmap.

Administrators can assign one or more of the group code bits at the LAT service provider; it will only accept LAT connections that have these group codes set in the bitmap. The administrators assign a bitmap of authorized group codes to each user. LAT gets these from the operating system and uses them in its requests to the service providers.

The codification of the range of allowed usage of this field is outside the scope of this specification.

4.4.11.5.4. Login-LAT-Port AVP

The Login-LAT-Port AVP (AVP Code 63) is of type OctetString and contains the Port with which the user is to be connected by LAT. It MAY be used in an authorization request as a hint to the server that a specific port is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in a response if the Login-Service-Type AVP is set to LAT.

The String field contains the identity of the LAT service to use. The LAT Architecture allows this string to contain $ (dollar), - (hyphen), . (period), _ (underscore), numerics, upper- and lower-case alphabetics, and the ISO Latin-1 character set extension [ISO.8859-1.1987].

All LAT string comparisons are case insensitive.

4.5. NAS Tunneling AVPs

Some NASes support compulsory tunnel services in which the incoming connection data is conveyed by an encapsulation method to a gateway elsewhere in the network. This is typically transparent to the service user, and the tunnel characteristics may be described by the remote AAA server, based on the user’s authorization information. Several tunnel characteristics may be returned, and the NAS

implementation may choose one. See Zorn, et al. [RFC2868] and Zorn, Aboba & Mitton [RFC2867] for further information.

The following table gives the possible flag values for the session level AVPs and specifies whether the AVP MAY be encrypted.

Attribute Name	Section Defined	AVP Flag rules	
		MUST	MUST NOT
Tunneling	4.5.1	M	V
Tunnel-Type	4.5.2	M	V
Tunnel-Medium-Type	4.5.3	M	V
Tunnel-Client-Endpoint	4.5.4	M	V
Tunnel-Server-Endpoint	4.5.5	M	V
Tunnel-Password	4.5.6	M	V
Tunnel-Private-Group-Id	4.5.7	M	V
Tunnel-Assignment-Id	4.5.8	M	V
Tunnel-Preference	4.5.9	M	V
Tunnel-Client-Auth-Id	4.5.10	M	V
Tunnel-Server-Auth-Id	4.5.11	M	V

4.5.1. Tunneling AVP

The Tunneling AVP (AVP Code 401) is of type Grouped and contains the following AVPs, used to describe a compulsory tunnel service ([RFC2868], [RFC2867]). Its data field has the following ABNF grammar:

```
Tunneling     ::= < AVP Header: 401 >
                  { Tunnel-Type }
                  { Tunnel-Medium-Type }
                  { Tunnel-Client-Endpoint }
                  { Tunnel-Server-Endpoint }
                  [ Tunnel-Preference ]
                  [ Tunnel-Client-Auth-Id ]
                  [ Tunnel-Server-Auth-Id ]
                  [ Tunnel-Assignment-Id ]
                  [ Tunnel-Password ]
                  [ Tunnel-Private-Group-Id ]
```

4.5.2. Tunnel-Type AVP

The Tunnel-Type AVP (AVP Code 64) is of type Enumerated and contains the tunneling protocol(s) to be used (in the case of a tunnel initiator) or in use (in the case of a tunnel terminator). It MAY be used in an authorization request as a hint to the server that a specific tunnel type is desired, but the server is not required to honor the hint in the corresponding response.

The Tunnel-Type AVP SHOULD also be included in ACR messages.

A tunnel initiator is not required to implement any of these tunnel types. If a tunnel initiator receives a response that contains only unknown or unsupported Tunnel-Types, the tunnel initiator MUST behave as though a response were received with the Result-Code indicating a failure.

The supported values are listed in [RADIUSAttrVals].

4.5.3. Tunnel-Medium-Type AVP

The Tunnel-Medium-Type AVP (AVP Code 65) is of type Enumerated and contains the transport medium to use when creating a tunnel for protocols (such as L2TP [RFC3931]) that can operate over multiple transports. It MAY be used in an authorization request as a hint to the server that a specific medium is desired, but the server is not required to honor the hint in the corresponding response.

The supported values are listed in [RADIUSAttrVals].

4.5.4. Tunnel-Client-Endpoint AVP

The Tunnel-Client-Endpoint AVP (AVP Code 66) is of type UTF8String and contains the address of the initiator end of the tunnel. It MAY be used in an authorization request as a hint to the server that a specific endpoint is desired, but the server is not required to honor the hint in the corresponding response. This AVP SHOULD be included in the corresponding ACR messages, in which case it indicates the address from which the tunnel was initiated. This AVP, along with the Tunnel-Server-Endpoint (Section 4.5.5) and Session-Id AVPs ([RFC6733], Section 8.8), can be used to provide a globally unique means to identify a tunnel for accounting and auditingpurposes.

If the value of the Tunnel-Medium-Type AVP (Section 4.5.3) is IPv4 (1), then this string is either the fully qualified domain name (FQDN) of the tunnel client machine, or a "dotted-decimal" IP address. Implementations MUST support the dotted-decimal format and SHOULD support the FQDN format for IP addresses.

If Tunnel-Medium-Type is IPv6 (2), then this string is either the FQDN of the tunnel client machine, or a text representation of the address in either the preferred or alternate form [RFC3516]. Conforming implementations MUST support the preferred form and SHOULD support both the alternate text form and the FQDN format for IPv6 addresses.

If Tunnel-Medium-Type is neither IPv4 nor IPv6, then this string is a tag referring to configuration data local to the Diameter client that describes the interface or medium-specific client address to use.

Note that this application handles internationalized domain names in the same way as the Diameter base protocol (see Appendix D of RFC 6733 for details).

4.5.5. Tunnel-Server-Endpoint AVP

The Tunnel-Server-Endpoint AVP (AVP Code 67) is of type UTF8String and contains the address of the server end of the tunnel. It MAY be used in an authorization request as a hint to the server that a specific endpoint is desired, but the server is not required to honor the hint in the corresponding response.

This AVP SHOULD be included in the corresponding ACR messages, in which case it indicates the address from which the tunnel was initiated. This AVP, along with the Tunnel-Client-Endpoint (Section 4.5.4) and Session-Id AVP ([RFC6733], Section 8.8), can be used to provide a globally unique means to identify a tunnel for accounting and auditing purposes.

If Tunnel-Medium-Type is IPv4 (1), then this string is either the fully qualified domain name (FQDN) of the tunnel server machine, or a "dotted-decimal" IP address. Implementations MUST support the dotted-decimal format and SHOULD support the FQDN format for IP addresses.

If Tunnel-Medium-Type is IPv6 (2), then this string is either the FQDN of the tunnel server machine, or a text representation of the address in either the preferred or alternate form [RFC3516]. Implementations MUST support the preferred form and SHOULD support both the alternate text form and the FQDN format for IPv6 addresses.

If Tunnel-Medium-Type is not IPv4 or IPv6, this string is a tag referring to configuration data local to the Diameter client that describes the interface or medium-specific server address to use.

Note that this application handles internationalized domain names in the same way as the Diameter base protocol (see Appendix D of RFC 6733 for details).

4.5.6. Tunnel-Password AVP

The Tunnel-Password AVP (AVP Code 69) is of type OctetString and may contain a password to be used to authenticate to a remote server.

The Tunnel-Password AVP SHOULD NOT be used in untrusted proxy environments without encrypting it by using end-to-end security techniques.

4.5.7. Tunnel-Private-Group-Id AVP

The Tunnel-Private-Group-Id AVP (AVP Code 81) is of type OctetString and contains the group Id for a particular tunneled session. The Tunnel-Private-Group-Id AVP MAY be included in an authorization request if the tunnel initiator can predetermine the group resulting from a particular connection. It SHOULD be included in the authorization response if this tunnel session is to be treated as belonging to a particular private group. Private groups may be used to associate a tunneled session with a particular group of users. For example, it MAY be used to facilitate routing of unregistered IP addresses through a particular interface. This AVP SHOULD be included in the ACR messages that pertain to the tunneled session.

4.5.8. Tunnel-Assignment-Id AVP

The Tunnel-Assignment-Id AVP (AVP Code 82) is of type OctetString and is used to indicate to the tunnel initiator the particular tunnel to which a session is to be assigned. Some tunneling protocols, such as PPTP [RFC2637] and L2TP [RFC3931], allow for sessions between the same two tunnel endpoints to be multiplexed over the same tunnel and also for a given session to use its own dedicated tunnel. This attribute provides a mechanism for Diameter to inform the tunnel initiator (for example, a LAC) whether to assign the session to a multiplexed tunnel or to a separate tunnel. Furthermore, it allows for sessions sharing multiplexed tunnels to be assigned to different multiplexed tunnels.

A particular tunneling implementation may assign differing characteristics to particular tunnels. For example, different tunnels may be assigned different QoS parameters. Such tunnels may be used to carry either individual or multiple sessions. The Tunnel-Assignment-Id attribute thus allows the Diameter server to indicate that a particular session is to be assigned to a tunnel providing an appropriate level of service. It is expected that any QoS-related

Diameter tunneling attributes defined in the future accompanying this one will be associated by the tunnel initiator with the Id given by this attribute. In the meantime, any semantic given to a particular Id string is a matter left to local configuration in the tunnel initiator.

The Tunnel-Assignment-Id AVP is of significance only to Diameter and the tunnel initiator. The Id it specifies is only intended to be of local use to Diameter and the tunnel initiator. The Id assigned by the tunnel initiator is not conveyed to the tunnel peer.

This attribute MAY be included in authorization responses. The tunnel initiator receiving this attribute MAY choose to ignore it and to assign the session to an arbitrary multiplexed or non-multiplexed tunnel between the desired endpoints. This AVP SHOULD also be included in the Accounting-Request messages pertaining to the tunneled session.

If a tunnel initiator supports the Tunnel-Assignment-Id AVP, then it should assign a session to a tunnel in the following manner:

- If this AVP is present and a tunnel exists between the specified endpoints with the specified Id, then the session should be assigned to that tunnel.

- If this AVP is present and no tunnel exists between the specified endpoints with the specified Id, then a new tunnel should be established for the session and the specified Id should be associated with the new tunnel.

- If this AVP is not present, then the session is assigned to an unnamed tunnel. If an unnamed tunnel does not yet exist between the specified endpoints, then it is established and used for this session and for subsequent ones established without the Tunnel-Assignment-Id attribute. A tunnel initiator MUST NOT assign a session for which a Tunnel-Assignment-Id AVP was not specified to a named tunnel (i.e., one that was initiated by a session specifying this AVP).

Note that the same Id may be used to name different tunnels if these tunnels are between different endpoints.

4.5.9. Tunnel-Preference AVP

The Tunnel-Preference AVP (AVP Code 83) is of type Unsigned32 and is used to identify the relative preference assigned to each tunnel when more than one set of tunneling AVPs is returned within separate Grouped-AVP AVPs. It MAY be used in an authorization request as a

hint to the server that a specific preference is desired, but the server is not required to honor the hint in the corresponding response.

For example, suppose that AVPs describing two tunnels are returned by the server, one with a Tunnel-Type of PPTP and the other with a Tunnel-Type of L2TP. If the tunnel initiator supports only one of the Tunnel-Types returned, it will initiate a tunnel of that type. If, however, it supports both tunnel protocols, it SHOULD use the value of the Tunnel-Preference AVP to decide which tunnel should be started. The tunnel with the lowest numerical value in the Value field of this AVP SHOULD be given the highest preference. The values assigned to two or more instances of the Tunnel-Preference AVP within a given authorization response MAY be identical. In this case, the tunnel initiator SHOULD use locally configured metrics to decide which set of AVPs to use.

4.5.10. Tunnel-Client-Auth-Id AVP

The Tunnel-Client-Auth-Id AVP (AVP Code 90) is of type UTF8String and specifies the 7-bit US-ASCII name used by the tunnel initiator during the authentication phase of tunnel establishment. It MAY be used in an authorization request as a hint to the server that a specific preference is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST be present in the authorization response if an authentication name other than the default is desired. This AVP SHOULD be included in the ACR messages pertaining to the tunneled session.

4.5.11. Tunnel-Server-Auth-Id AVP

The Tunnel-Server-Auth-Id AVP (AVP Code 91) is of type UTF8String and specifies the 7-bit US-ASCII name used by the tunnel terminator during the authentication phase of tunnel establishment. It MAY be used in an authorization request as a hint to the server that a specific preference is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST be present in the authorization response if an authentication name other than the default is desired. This AVP SHOULD be included in the ACR messages pertaining to the tunneled session.

4.6. NAS Accounting AVPs

Applications implementing this specification use Diameter Accounting (as defined in [RFC6733]) and the AVPs in the following section. Service-specific AVP usage is defined in the tables in Section 5.

If accounting is active, Accounting Request (ACR) messages SHOULD be sent after the completion of any Authentication or Authorization transaction and at the end of a Session. The value of the Accounting-Record-Type AVP [RFC6733] indicates the type of event. All other AVPs identify the session and provide additional information relevant to the event.

The successful completion of the first Authentication or Authorization transaction SHOULD cause a START_RECORD to be sent. If additional Authentications or Authorizations occur in later transactions, the first exchange should generate a START_RECORD, and the later an INTERIM_RECORD. For a given session, there MUST only be one set of matching START and STOP records, with any number of INTERIM_RECORDS in between, or one EVENT_RECORD indicating the reason a session wasn't started.

The following table gives the possible flag values for the session level AVPs and specifies whether the AVP MAY be encrypted.

		AVP Flag rules	
Attribute Name	Section Defined	MUST	MUST NOT
Accounting-Input-Octets	4.6.1	M	V
Accounting-Output-Octets	4.6.2	M	V
Accounting-Input-Packets	4.6.3	M	V
Accounting-Output-Packets	4.6.4	M	V
Acct-Session-Time	4.6.5	M	V
Acct-Authentic	4.6.6	M	V
Accounting-Auth-Method	4.6.7	M	V
Acct-Delay-Time	4.6.8	M	V
Acct-Link-Count	4.6.9	M	V
Acct-Tunnel-Connection	4.6.10	M	V
Acct-Tunnel-Packets-Lost	4.6.11	M	V

4.6.1. Accounting-Input-Octets AVP

The Accounting-Input-Octets AVP (AVP Code 363) is of type Unsigned64 and contains the number of octets received from the user.

For NAS usage, this AVP indicates how many octets have been received from the port in the course of this session. It can only be present in ACR messages with an Accounting-Record-Type [RFC6733] of INTERIM_RECORD or STOP_RECORD.

4.6.2. Accounting-Output-Octets AVP

The Accounting-Output-Octets AVP (AVP Code 364) is of type Unsigned64 and contains the number of octets sent to the user.

For NAS usage, this AVP indicates how many octets have been sent to the port in the course of this session. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.3. Accounting-Input-Packets AVP

The Accounting-Input-Packets (AVP Code 365) is of type Unsigned64 and contains the number of packets received from the user.

For NAS usage, this AVP indicates how many packets have been received from the port over the course of a session being provided to a Framed User. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.4. Accounting-Output-Packets AVP

The Accounting-Output-Packets (AVP Code 366) is of type Unsigned64 and contains the number of IP packets sent to the user.

For NAS usage, this AVP indicates how many packets have been sent to the port over the course of a session being provided to a Framed User. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.5. Acct-Session-Time AVP

The Acct-Session-Time AVP (AVP Code 46) is of type Unsigned32 and indicates the length of the current session in seconds. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.6. Acct-Authentic AVP

The Acct-Authentic AVP (AVP Code 45) is of type Enumerated and specifies how the user was authenticated. The supported values are listed in [RADIUSAttrVals].

4.6.7. Accounting-Auth-Method AVP

The Accounting-Auth-Method AVP (AVP Code 406) is of type Enumerated. A NAS MAY include this AVP in an Accounting-Request message to indicate the method used to authenticate the user. (Note that this AVP is semantically equivalent, and the supported values are identical, to the Microsoft MS-Acct-Auth-Type vendor-specific RADIUS attribute [RFC2548]).

4.6.8. Acct-Delay-Time AVP

The Acct-Delay-Time AVP (AVP Code 41) is of type Unsigned32 and indicates the number of seconds the Diameter client has been trying to send the Accounting-Request (ACR). The accounting server may subtract this value from the time when the ACR arrives at the server to calculate the approximate time of the event that caused the ACR to be generated.

This AVP is not used for retransmissions at the transport level (TCP or SCTP). Rather, it may be used when an ACR command cannot be transmitted because there is no appropriate peer to transmit it to or was rejected because it could not be delivered. In these cases, the command MAY be buffered and transmitted later, when an appropriate peer-connection is available or after sufficient time has passed that the destination-host may be reachable and operational. If the ACR is re-sent in this way, the Acct-Delay-Time AVP SHOULD be included. The value of this AVP indicates the number of seconds that elapsed between the time of the first attempt at transmission and the current attempt.

4.6.9. Acct-Link-Count AVP

The Acct-Link-Count AVP (AVP Code 51) is of type Unsigned32 and indicates the total number of links that have been active (current or closed) in a given multilink session at the time the accounting record is generated. This AVP MAY be included in Accounting-Requests for any session that may be part of a multilink service.

The Acct-Link-Count AVP may be used to make it easier for an accounting server to know when it has all the records for a given multilink service. When the number of Accounting-Requests received with Accounting-Record-Type = STOP_RECORD and with the same Acct-Multi-Session-Id and unique Session-Ids equals the largest value of Acct-Link-Count seen in those Accounting-Requests, all STOP_RECORD Accounting-Requests for that multilink service have been received.

The following example, showing eight Accounting-Requests, illustrates how the Acct-Link-Count AVP is used. In the table below, only the

relevant AVPs are shown, although additional AVPs containing accounting information will be present in the Accounting-Requests.

Acct-Multi-Session-Id	Session-Id	Accounting-Record-Type	Acct-Link-Count
"...10"	"...10"	START_RECORD	1
"...10"	"...11"	START_RECORD	2
"...10"	"...11"	STOP_RECORD	2
"...10"	"...12"	START_RECORD	3
"...10"	"...13"	START_RECORD	4
"...10"	"...12"	STOP_RECORD	4
"...10"	"...13"	STOP_RECORD	4
"...10"	"...10"	STOP_RECORD	4

4.6.10. Acct-Tunnel-Connection AVP

The Acct-Tunnel-Connection AVP (AVP Code 68) is of type OctetString and contains the identifier assigned to the tunnel session. This AVP, along with the Tunnel-Client-Endpoint (Section 4.5.4) and Tunnel-Server-Endpoint (Section 4.5.5) AVPs, may be used to provide a means to uniquely identify a tunnel session for auditing purposes.

The format of the identifier in this AVP depends upon the value of the Tunnel-Type AVP (Section 4.5.2). For example, to identify an L2TP tunnel connection fully, the L2TP Tunnel Id and Call Id might be encoded in this field. The exact encoding of this field is implementation dependent.

4.6.11. Acct-Tunnel-Packets-Lost AVP

The Acct-Tunnel-Packets-Lost AVP (AVP Code 86) is of type Unsigned32 and contains the number of packets lost on a given tunnel.

5. AVP Occurrence Tables

The following tables present the AVPs used by NAS applications in NAS messages and specify in which Diameter messages they may or may not be present. Messages and AVPs defined in the base Diameter protocol [RFC6733] are not described in this document. Note that AVPs that can only be present within a Grouped AVP are not represented in this table.

The tables use the following symbols:

0 The AVP MUST NOT be present in the message.

0+ Zero or more instances of the AVP MAY be present in the message.

0-1 Zero or one instance of the AVP MAY be present in the message.

1 Exactly one instance of the AVP MUST be present in the message.

5.1. AA-Request/Answer AVP Table

The table in this section is limited to the Command Codes defined in this specification.

AVP Name	Command AAR	Command AAA
Acct-Interim-Interval	0	0-1
ARAP-Challenge-Response	0	0-1
ARAP-Features	0	0-1
ARAP-Password	0-1	0
ARAP-Security	0-1	0-1
ARAP-Security-Data	0+	0+
ARAP-Zone-Access	0	0-1
Auth-Application-Id	1	1
Auth-Grace-Period	0-1	0-1
Auth-Request-Type	1	1
Auth-Session-State	0-1	0-1
Authorization-Lifetime	0-1	0-1

Attribute Name	Command AAR	Command AAA
Callback-Id	0	0-1
Callback-Number	0-1	0-1
Called-Station-Id	0-1	0
Calling-Station-Id	0-1	0
CHAP-Auth	0-1	0
CHAP-Challenge	0-1	0
Class	0	0+
Configuration-Token	0	0+
Connect-Info	0+	0
Destination-Host	0-1	0

Destination-Realm	1	0
Error-Message	0	0-1
Error-Reporting-Host	0	0-1
Failed-AVP	0+	0+
Filter-Id	0	0+
Framed-Appletalk-Link	0	0-1
Framed-Appletalk-Network	0	0+
Framed-Appletalk-Zone	0	0-1
Framed-Compression	0+	0+
Framed-Interface-Id	0-1	0-1
Framed-IP-Address	0-1	0-1
Framed-IP-Netmask	0-1	0-1
Framed-IPv6-Prefix	0+	0+
Framed-IPv6-Pool	0	0-1
Framed-IPv6-Route	0	0+
Framed-IPX-Network	0	0-1
Framed-MTU	0-1	0-1
Framed-Pool	0	0-1
Framed-Protocol	0-1	0-1
Framed-Route	0	0+
Framed-Routing	0	0-1
Idle-Timeout	0	0-1
Login-IP-Host	0+	0+
Login-IPv6-Host	0+	0+
Login-LAT-Group	0-1	0-1
Login-LAT-Node	0-1	0-1
Login-LAT-Port	0-1	0-1
Login-LAT-Service	0-1	0-1
Login-Service	0	0-1
Login-TCP-Port	0	0-1
Multi-Round-Time-Out	0	0-1

	Command	
Attribute Name	AAR	AAA
NAS-Filter-Rule	0	0+
NAS-Identifier	0-1	0
NAS-IP-Address	0-1	0
NAS-IPv6-Address	0-1	0
NAS-Port	0-1	0
NAS-Port-Id	0-1	0
NAS-Port-Type	0-1	0
Origin-AAA-Protocol	0-1	0-1
Origin-Host	1	1
Origin-Realm	1	1

Origin-State-Id	0-1	0-1
Originating-Line-Info	0-1	0
Password-Retry	0	0-1
Port-Limit	0-1	0-1
Prompt	0	0-1
Proxy-Info	0+	0+
QoS-Filter-Rule	0	0+
Re-Auth-Request-Type	0	0-1
Redirect-Host	0	0+
Redirect-Host-Usage	0	0-1
Redirect-Max-Cache-Time	0	0-1
Reply-Message	0	0+
Result-Code	0	1
Route-Record	0+	0
Service-Type	0-1	0-1
Session-Id	1	1
Session-Timeout	0	0-1
State	0-1	0-1
Tunneling	0+	0+
User-Name	0-1	0-1
User-Password	0-1	0

5.2. Accounting AVP Tables

The tables in this section are used to show which AVPs defined in this document are to be present and used in NAS application Accounting messages. These AVPs are defined in this document, as well as in [RFC6733] and [RFC2866].

5.2.1. Framed Access Accounting AVP Table

The table in this section is used when the Service-Type AVP (Section 4.4.1) specifies Framed Access.

	Command	
Attribute Name	ACR	ACA
Accounting-Auth-Method	0-1	0
Accounting-Input-Octets	1	0
Accounting-Input-Packets	1	0
Accounting-Output-Octets	1	0
Accounting-Output-Packets	1	0
Accounting-Record-Number	0-1	0-1
Accounting-Record-Type	1	1
Accounting-Realtime-Required	0-1	0-1

Attribute Name	Command ACR	Command ACA
Accounting-Sub-Session-Id	0-1	0-1
Acct-Application-Id	0-1	0-1
Acct-Session-Id	1	0-1
Acct-Multi-Session-Id	0-1	0-1
Acct-Authentic	1	0
Acct-Delay-Time	0-1	0
Acct-Interim-Interval	0-1	0-1
Acct-Link-Count	0-1	0
Acct-Session-Time	1	0
Acct-Tunnel-Connection	0-1	0
Acct-Tunnel-Packets-Lost	0-1	0
Authorization-Lifetime	0-1	0
Callback-Id	0-1	0
Callback-Number	0-1	0
Called-Station-Id	0-1	0
Calling-Station-Id	0-1	0
Class	0+	0+
Connection-Info	0+	0
Destination-Host	0-1	0
Destination-Realm	1	0
Event-Timestamp	0-1	0-1
Error-Message	0	0-1
Error-Reporting-Host	0	0-1
Failed-AVP	0	0+

Attribute Name	Command ACR	Command ACA
Framed-AppleTalk-Link	0-1	0
Framed-AppleTalk-Network	0-1	0
Framed-AppleTalk-Zone	0-1	0
Framed-Compression	0-1	0
Framed-IP-Address	0-1	0
Framed-IP-Netmask	0-1	0
Framed-IPv6-Prefix	0+	0
Framed-IPv6-Pool	0-1	0
Framed-IPX-Network	0-1	0
Framed-MTU	0-1	0
Framed-Pool	0-1	0
Framed-Protocol	0-1	0
Framed-Route	0-1	0
Framed-Routing	0-1	0
NAS-Filter-Rule	0+	0
NAS-Identifier	0-1	0-1
NAS-IP-Address	0-1	0-1

NAS-IPv6-Address	0-1	0-1
NAS-Port	0-1	0-1
NAS-Port-Id	0-1	0-1
NAS-Port-Type	0-1	0-1
Origin-AAA-Protocol	0-1	0-1
Origin-Host	1	1
Origin-Realm	1	1
Origin-State-Id	0-1	0-1
Originating-Line-Info	0-1	0
Proxy-Info	0+	0+
QoS-Filter-Rule	0+	0
Route-Record	0+	0
Result-Code	0	1
Service-Type	0-1	0-1
Session-Id	1	1
Termination-Cause	0-1	0-1
Tunnel-Assignment-Id	0-1	0
Tunnel-Client-Endpoint	0-1	0
Tunnel-Medium-Type	0-1	0
Tunnel-Private-Group-Id	0-1	0
Tunnel-Server-Endpoint	0-1	0
Tunnel-Type	0-1	0
User-Name	0-1	0-1

5.2.2. Non-Framed Access Accounting AVP Table

The table in this section is used when the Service-Type AVP (Section 4.4.1) specifies Non-Framed Access.

	Command	
Attribute Name	ACR	ACA
Accounting-Auth-Method	0-1	0
Accounting-Input-Octets	1	0
Accounting-Output-Octets	1	0
Accounting-Record-Type	1	1
Accounting-Record-Number	0-1	0-1
Accounting-Realtime-Required	0-1	0-1
Accounting-Sub-Session-Id	0-1	0-1
Acct-Application-Id	0-1	0-1
Acct-Session-Id	1	0-1
Acct-Multi-Session-Id	0-1	0-1
Acct-Authentic	1	0
Acct-Delay-Time	0-1	0
Acct-Interim-Interval	0-1	0-1

Acct-Link-Count	0-1	0
Acct-Session-Time	1	0
Authorization-Lifetime	0-1	0
Callback-Id	0-1	0
Callback-Number	0-1	0
Called-Station-Id	0-1	0
Calling-Station-Id	0-1	0
Class	0+	0+
Connection-Info	0+	0
Destination-Host	0-1	0
Destination-Realm	1	0
Event-Timestamp	0-1	0-1
Error-Message	0	0-1
Error-Reporting-Host	0	0-1
Failed-AVP	0	0+
Login-IP-Host	0+	0
Login-IPv6-Host	0+	0
Login-LAT-Service	0-1	0
Login-LAT-Node	0-1	0
Login-LAT-Group	0-1	0
Login-LAT-Port	0-1	0
Login-Service	0-1	0
Login-TCP-Port	0-1	0

	Command	
Attribute Name	ACR	ACA
NAS-Identifier	0-1	0-1
NAS-IP-Address	0-1	0-1
NAS-IPv6-Address	0-1	0-1
NAS-Port	0-1	0-1
NAS-Port-Id	0-1	0-1
NAS-Port-Type	0-1	0-1
Origin-AAA-Protocol	0-1	0-1
Origin-Host	1	1
Origin-Realm	1	1
Origin-State-Id	0-1	0-1
Originating-Line-Info	0-1	0
Proxy-Info	0+	0+
QoS-Filter-Rule	0+	0
Route-Record	0+	0
Result-Code	0	1
Session-Id	1	1
Service-Type	0-1	0-1
Termination-Cause	0-1	0-1

User-Name	0-1	0-1

6. Unicode Considerations

A number of the AVPs in this RFC use the UTF8String type specified in the Diameter Base protocol [RFC6733]. Implementation differences in Unicode input processing may result in the same Unicode input characters generating different UTF-8 strings that fail to match when compared for equality. This may result in interoperability problems between a network access server and a Diameter server when a UTF-8 string entered locally is compared with one received via Diameter. Many of the uses of UTF8String in this RFC are limited to the 7-bit ASCII-compatible subset of UTF-8 where this class of Unicode string comparison problems does not arise.

Careful preparation of Unicode strings can increase the likelihood that string comparison will work in ways that make sense for typical users throughout the world; [RFC3454] is an example a framework for such Unicode string preparation. The Diameter application specified in this RFC has been deployed with use of Unicode in accordance with [RFC4005], which does not require any Unicode string preparation. As a result, additional requirements for Unicode string preparation in this RFC would not be backwards compatible with existing usage.

The Diameter server and the network access servers that it serves can be assumed to be under common administrative control, and all of the UTF-8 strings involved are part of the configuration of these servers. Therefore administrative interfaces for implementations of this RFC:

a. SHOULD accept direct UTF-8 input of all configuration strings for AVPs that allow Unicode characters beyond the 7-bit ASCII-compatible subset of Unicode (in addition to any provisions for accepting Unicode characters for processing into UTF-8), and

b. SHOULD make all such configuration strings available as UTF-8 strings

This functionality enables an administrator who encounters Unicode string comparison problems to copy one instance of aproblematic UTF-8 string from one server to the other, after which the two (now identical) copies should compare as expected.

7. IANA Considerations

Several of the namespaces used in this document are managed by the Internet Assigned Numbers Authority [IANA], including the AVP Codes

[AVP-Codes], AVP Specific Values [AVP-Vals], Application IDs [App-Ids], Command Codes [Command-Codes] and RADIUS Attribute Values [RADIUSAttrVals].

For the current values allocated, and the policies governing allocation in those namespaces, please see the above-referenced registries.

IANA Note: Please change all the references in the registries listed above that are currently pointing to RFC 4005 to point to this document instead; please change the reference for for the value '1' in the "Application IDs" sub-registry of the "Authentication, Authorization, and Accounting (AAA) Parameters" registry to point to this document, as well.

RFC Editor: Please remove both this note and the IANA note above before publication.

8. Security Considerations

This document describes the extension of Diameter for the NAS application. Security considerations regarding the Diameter protocol itself are discussed in [RFC6733]. Use of this application of Diameter MUST take into consideration the security issues and requirements of the Base protocol.

8.1. Authentication Considerations

This document does not contain a security protocol but does discuss how PPP authentication protocols can be carried within the Diameter protocol. The PPP authentication protocols described are PAP and CHAP.

The use of PAP SHOULD be discouraged, as it exposes users' passwords to possibly non-trusted entities. However, PAP is also frequently used for use with One-Time Passwords, which do not expose a security risk.

This document also describes how CHAP can be carried within the Diameter protocol, which is required for RADIUS backward compatibility. The CHAP protocol, as used in a RADIUS environment, facilitates authentication replay attacks.

The use of the EAP authentication protocols [RFC4072] can offer better security, given a method suitable for the circumstances.

Depending on the value of the Auth-Request-Type AVP, the Diameter protocol allows authorization-only requests that contain no

authentication information from the client. This capability goes beyond the Call Check capabilities provided by RADIUS (Section 5.6 of [RFC2865]) in that no access decision is requested. As a result, a new session cannot be started as a result of a response to an authorization-only request without introducing a significant security vulnerability.

8.2. AVP Considerations

Diameter AVPs often contain security-sensitive data; for example, user passwords and location data, network addresses and cryptographic keys. With the exception of the Configuration-Token (Section 4.4.8), QoS-Filter-Rule (Section 4.4.9) and Tunneling (Section 4.5.1) AVPs, all of the AVPs defined in this document are considered to be security-sensitive.

Diameter messages containing any AVPs considered to be security-sensitive MUST only be sent protected via mutually authenticated TLS or IPsec. In addition, those messages MUST NOT be sent via intermediate nodes unless there is end-to-end security between the originator and recipient or the originator has locally trusted configuration that indicates that end-to-end security is not needed. For example, end-to-end security may not be required in the case where an intermediary node is known to be operated as part of the same administrative domain as the endpoints so that an ability to successfully compromise the intermediary would imply a high probability of being able to compromise the endpoints as well. Note that no end-to-end security mechanism is specified in this document.

9. References

9.1. Normative References

[ANITypes] NANPA Number Resource Info, "ANI Assignments", <http://www.nanpa.com/number_resource_info/ani_ii_assignments.html>.

[RFC1994] Simpson, W., "PPP Challenge Handshake Authentication Protocol (CHAP)", RFC 1994, August 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC3162] Aboba, B., Zorn, G., and D. Mitton, "RADIUS and IPv6", RFC 3162, August 2001.

[RFC3516] Nerenberg, L., "IMAP4 Binary Content Extension", RFC 3516, April 2003.

[RFC3539] Aboba, B. and J. Wood, "Authentication, Authorization and Accounting (AAA) Transport Profile", RFC 3539, June 2003.

[RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Jones, M., and A. Lior, "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, February 2010.

[RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

9.2. Informative References

[ARAP] Apple Computer, "Apple Remote Access Protocol (ARAP) Version 2.0 External Reference Specification", R0612LL/B , September 1994.

[AVP-Codes] IANA, "IANA AAA AVP Codes Registry", <http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#aaa-parameters-1>.

[AVP-Vals] IANA, "IANA AAA AVP Specific Values", <http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#aaa-parameters-2>.

[App-Ids] IANA, "IANA AAA Application IDs Registry", <http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#aaa-parameters-1>.

[AppleTalk] Sidhu, G., Andrews, R., and A. Oppenheimer, "Inside AppleTalk", Second Edition Apple Computer, 1990.

[BASE] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[Command-Codes] IANA, "IANA AAA Command Codes Registry", <http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#command-code-rules>.

[IANA] IANA, "Internet Assigned Numbers Authority", <http://www.iana.org/>.

[IPX] Novell, Inc., "NetWare System Technical Interface Overview", #883-000780-001, June 1989.

[ISO.8859-1.1987] International Organization for Standardization, "Information technology - 8-bit single byte coded graphic - character sets - Part 1: Latin alphabet No. 1, JTC1/ SC2", ISO Standard 8859-1, 1987.

[LAT] Digital Equipment Corp., "Local Area Transport (LAT) Specification V5.0", AA-NL26A-TE, June 1989.

[RADIUSAttrVals] IANA, "IANA Radius Attribute Values Registry", <http:// www.iana.org/assignments/radius-types/radius-types.xml #radius-types-3>.

[RFC1334] Lloyd, B. and W. Simpson, "PPP Authentication Protocols", RFC 1334, October 1992.

[RFC1661] Simpson, W., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994.

[RFC1990] Sklower, K., Lloyd, B., McGregor, G., Carr, D., and T. Coradetti, "The PPP Multilink Protocol (MP)", RFC 1990, August 1996.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[RFC2548] Zorn, G., "Microsoft Vendor-specific RADIUS Attributes", RFC 2548, March 1999.

[RFC2597] Heinanen, J., Baker, F., Weiss, W., and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597, June 1999.

[RFC2637] Hamzeh, K., Pall, G., Verthein, W., Taarud, J., Little, W., and G. Zorn, "Point-to-Point Tunneling Protocol", RFC 2637, July 1999.

[RFC2866] Rigney, C., "RADIUS Accounting", RFC 2866, June 2000.

[RFC2867] Zorn, G., Aboba, B., and D. Mitton, "RADIUS Accounting Modifications for Tunnel Protocol Support", RFC 2867, June 2000.

[RFC2868] Zorn, G., Leifer, D., Rubens, A., Shriver, J., Holdrege, M., and I. Goyret, "RADIUS Attributes for Tunnel Protocol Support", RFC 2868, June 2000.

[RFC2869] Rigney, C., Willats, W., and P. Calhoun, "RADIUS Extensions", RFC 2869, June 2000.

[RFC2881] Mitton, D. and M. Beadles, "Network Access Server Requirements Next Generation (NASREQNG) NAS Model", RFC 2881, July 2000.

[RFC2989] Aboba, B., Calhoun, P., Glass, S., Hiller, T., McCann, P., Shiino, H., Walsh, P., Zorn, G., Dommety, G., Perkins, C., Patil, B., Mitton, D., Manning, S., Beadles, M., Chen, X., Sivalingham, S., Hameed, A., Munson, M., Jacobs, S., Lim, B., Hirschman, B., Hsu, R., Koo, H., Lipford, M., Campbell, E., Xu, Y., Baba, S., and E. Jaques, "Criteria for Evaluating AAA Protocols for Network Access", RFC 2989, November 2000.

[RFC3169] Beadles, M. and D. Mitton, "Criteria for Evaluating Network Access Server Protocols", RFC 3169, September 2001.

[RFC3246] Davie, B., Charny, A., Bennet, J., Benson, K., Le Boudec, J., Courtney, W., Davari, S., Firoiu, V., and D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)", RFC 3246, March 2002.

[RFC3454] , .

[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.

[RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

[RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

Appendix A. Acknowledgements

A.1. This Document

The vast majority of the text in this document was taken directly from RFC 4005; the editor owes a debt of gratitude to the authors thereof (especially Dave Mitton, who somehow managed to make nroff paginate the AVP Occurance Tables correctly!).

Thanks (in no particular order) to Jai-Jin Lim, Liu Hans, Sebastien Decugis, Jouni Korhonen, Mark Jones, Hannes Tschofenig, Dave Crocker, David Black, Barry Leiba, Peter Saint-Andre, Stefan Winter and Lionel Morand for their useful reviews and helpful comments.

A.2. RFC 4005

The authors would like to thank Carl Rigney, Allan C. Rubens, William Allen Simpson, and Steve Willens for their work on the original RADIUS protocol, from which many of the concepts in this specification were derived. Thanks, also, to Carl Rigney for [RFC2866] and [RFC2869]; Ward Willats for [RFC2869]; Glen Zorn, Bernard Aboba, and Dave Mitton for [RFC2867] and [RFC3162]; and Dory Leifer, John Shriver, Matt Holdrege, Allan Rubens, Glen Zorn and Ignacio Goyret for their work on [RFC2868]. This document stole text and concepts from both [RFC2868] and [RFC2869]. Thanks go to Carl Williams for providing IPv6-specific text.

The authors would also like to acknowledge the following people for their contributions in the development of the Diameter protocol: Bernard Aboba, Jari Arkko, William Bulley, Kuntal Chowdhury, Daniel C. Fox, Lol Grant, Nancy Greene, Jeff Hagg, Peter Heitman, Paul Krumviede, Fergal Ladley, Ryan Moats, Victor Muslin, Kenneth Peirce, Sumit Vakil, John R. Vollbrecht, and Jeff Weisberg.

Finally, Pat Calhoun would like to thank Sun Microsystems, as most of the effort put into this document was done while he was in their employ.

Author’s Address

Glen Zorn (editor)
Network Zen
227/358 Thanon Sanphawut
Bang Na, Bangkok 10260
Thailand

Phone: +66 (0)8-1000-4155
EMail: glenzorn@gmail.com

This Internet-Draft, draft-liebsch-dime-diameter-gps-01.txt, has expired, and has been deleted from the Internet-Drafts directory. An Internet-Draft expires 185 days from the date that it is posted unless it is replaced by an updated version, or the Secretariat has been notified that the document is under official review by the IESG or has been passed to the RFC Editor for review and/or publication as an RFC. This Internet-Draft was not published as an RFC.

Internet-Drafts are not archival documents, and copies of Internet-Drafts that have been deleted from the directory are not available. The Secretariat does not have any information regarding the future plans of the authors or working group, if applicable, with respect to this deleted Internet-Draft. For more information, or to request a copy of the document, please contact the authors directly.

Draft Authors:
Marco Liebsch<liebsch@neclab.eu>
Gottfried Punz<punz@neclab.eu>