AAA Working Group
Internet-Draft
Category: Standards Track
<draft-ietf-aaa-diameter-09.txt>

Pat R. Calhoun
Black Storm Networks
Jari Arkko
Oy LM Ericsson Ab
Erik Guttman
Sun Microsystems, Inc.
Glen Zorn
Cisco Systems, Inc.
John Loughney
Nokia
March 2002

Diameter Base Protocol

Status of this Memo

Abstract

The Diameter base protocol is intended to provide an AAA framework for Mobile-IP, NASREQ and ROAMOPS. This draft specifies the message format, transport, error reporting and security services to be used by all Diameter applications and MUST be supported by all Diameter implementations.

Table of Contents

1.0 Introduction

Historically, the RADIUS protocol has been used to provide AAA services for dial-up PPP [PPP] and terminal server access. Over time, routers and network access servers (NAS) have increased in complexity and density, making the RADIUS protocol increasingly unsuitable for use in such networks.

The Roaming Operations Working Group (ROAMOPS) has published a set of specifications [ROAMCRIT, ROAMREV, PROXYCHAIN] that define how a PPP user can gain access to the Internet without having to dial into his/her home service provider's modem pool. This is achieved by allowing service providers to cross-authenticate their users. Effectively, a user can dial into any service provider's point of presence (POP) that has a roaming agreement with his/her home Internet service provider (ISP), the benefit being that the user does not have to incur a long distance charge while traveling, which can sometimes be quite expensive.

Given the number of ISPs today, ROAMOPS realized that requiring each ISP to set up roaming agreements with all other ISPs did not scale. Therefore, the working group defined a "broker", which acts as an intermediate server, whose sole purpose is to set up these roaming agreements. A collection of ISPs and a broker is called a "roaming consortium". There are many such brokers in existence today; many also provide settlement services for member ISPs.

The Mobile-IP Working Group has recently changed its focus to inter-administrative domain mobility, which is a requirement for cellular carriers wishing to deploy IETF-based mobility protocols. The current cellular carriers requirements [CDMA2000REQ, MIPREQ] are very similar to the ROAMOPS model, with the exception that the access protocol is Mobile-IP [MIPV4] instead of PPP.

The Network Access Server Requirements (NASREQ) working group has focused on proving next generation Authentication, Authorization and usage Accounting for simple dial-in access and beyond; such as Virtual Private Network support, smart authentication methods, and roaming concerns. The Working Group has published number of documents the requirements for NAS user authorization as well as criteria for evaluating NAS protocols [NASCRIT].

The basic concept behind Diameter is to provide a base protocol that can be extended in order to provide AAA services to new access technologies. Currently, the protocol only concerns itself with Internet access, both in the traditional PPP sense as well as taking into account the ROAMOPS model, and Mobile-IP.

Although Diameter could be used to solve a wider set of AAA problems,

we are currently limiting the scope of the protocol in order to ensure that the effort remains focused on satisfying the requirements of network access. Note that 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.

1.1 Diameter Protocol

The Diameter protocol allows peers to exchange a variety of messages. The base protocol provides the following facilities:

- Delivery of AVPs (attribute value pairs)
- Capabilities negotiation, as required in [ROAMCRIT]
- Error notification
- Extensibility, through addition of new commands and AVPs, as required in [NASCRIT]

All data delivered by the protocol is in the form of an AVP. 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 added arbitrarily to Diameter messages, so long as the required AVPs are included and AVPs that are explicitly excluded are not included. 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.
- Relaying, proxying and redirecting of Diameter messages through a server hierarchy.

The Diameter base protocol provides the minimum requirements needed for an AAA transport protocol, as required by NASREQ [NASCRIT], Mobile IP [CDMA2000REQ, MIPREQ], and ROAMOPS [ROAMCRIT]. The base protocol is not intended to be used by itself, and must be used with a Diameter application, such as Mobile IP [DIAMMIP]. The Diameter protocol was heavily inspired by and builds upon the tradition of the RADIUS [RADIUS] protocol. See section 2.4 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 an access device that initiates a request for authentication and/or authorization of a given user. A Diameter agent is a node that does not authenticate and/or authorize messages locally. Examples of agents are proxies and relay agents. A Diameter server is one that performs authentication and/or authorization of the user based on some profile. 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 towards access devices, such as a request to abort service to a particular user.

1.1.1 Differences from Radius

The Diameter protocol was not designed from the ground up. Instead, the basic RADIUS model was retained while fixing the flaws in the RADIUS protocol itself. Diameter does not share a common protocol data unit (PDU) with RADIUS, but does borrow sufficiently from the protocol to ease migration. The major differences include:

- Peer-to-peer nature
- Explicit support for intermediaries
- Connection-oriented versus connectionless
- Extensibility [see section 1.2]
- Built-in failover support
- Larger attribute space
- Integrated accounting
- Mandatory bit
- Application-layer ACKs and error messages
- Unsolicited server messages
- Peer discovery
- Capabilities negotiation

1.1.2 Description of the Document Set

Currently, the Diameter specification consists of a base specification (this document), Transport Issues [AAATRANS] and a number of applications: Mobile IPv4 [DIAMMIP], NASREQ [NASREQ] and CMS Security [CMS].

The Transport Issues document [AAATRANS] discusses transport layer issues that arise with AAA protocols and recommendations on how to overcome these issues.

The Mobile IPv4 [DIAMMIP] application defines a Diameter application that allows a Diameter server to perform AAA functions for Mobile

IPv4 services to a mobile node.

The NASREQ [NASREQ] application defines a Diameter Application that allows a Diameter server to be used in a PPP/SLIP Dial-Up and Terminal Server Access environment. Consideration was given for servers that need to perform protocol conversion between Diameter and RADIUS.

The CMS Security [CMS] application defines how security associations are established between two peers and how authentication, integrity, confidentiality and non-repudiation can be achieved.

In summary, this document defines the base protocol specification for AAA. The MIPv4 and the NASREQ documents describe applications that use this base specification to achieve Authentication, Authorization and Accounting. The CMS Application describes a security application for providing secure communication in the presence of relay and peer agents.

1.2 Approach to Extensibility

The Diameter protocol is designed to be extensible. However, it is strongly encouraged to reuse existing mechanism before attempting any Diameter extensions. The extensibility includes:

- Defining new AVP values.
- Creating new AVPs
- Creating new authentication applications
- Creating a new Accounting Application
- Application Authentication Procedures

1.2.1 Defining New AVP Values

New applications should attempt to reuse AVPs defined in existing application when possible, as opposed to creating a new AVP. For AVPs of type Enumerated, it is possible the application requires a new value to communicate some service-specific information.

In order to allocate a new AVP value, a request MUST be sent to IANA

[47], with a detailed explanation of the value. If the new AVP value changes an existing command code's ABNF, the IANA AVP value request MUST include the new ABNF itself.

1.2.2 Creating New AVPs

When no existing AVP can be used appropriately to communicate some service-specific information, a new AVP should be created. The new AVP being defined MUST follow one of the types listed in section 4.3.

In the event that a logical grouping of AVPs is necessary, and multiple "groups" are possible in a given command, it is highly recommended that a Grouped AVP be used (see Section 4.5).

In order to create a new AVP, a request MUST be sent to IANA, with a detailed explanation of the AVP, its type and possible values. Furthermore, the request MUST include the commands that would make use of the AVP.

2.3.3 Creating New Auth Applications

Should a new application require Diameter support, but it cannot fit within an existing application without requiring major changes to the specification, it may be desirable to create a new Diameter application. Major changes to an application include:

- Requiring a whole different set of mandatory AVPs to a command
- Requiring a command that has a different number of round trips to satisfy a request (e.g. application foo has a command that requires one round trip, but new application bar has a command that requires two round trips to complete).
- The method used to authenticate the user is drastically different from any existing application, and the authentication information cannot be carried within the AVPs defined in the application.

Note that the creation of a new application should be viewed as a last resort.

New Diameter applications MUST define at least one Command Code, the expected AVPs in an ABNF [ABNF] grammar (see section 3.2), and MAY also define new AVPs. If the Diameter application has any accounting requirements, it MUST also specify the AVPs that are to be present in

the Diameter Accounting messages (see section 9.3).

When possible, a new Diameter application SHOULD attempt to re-use any existing Diameter AVP, in order to reduce the possibility of having multiple AVPs that carry similar information.

Every Diameter application specification MUST have an IANA assigned Application Identifier (see section 2.4).

1.2.4 Creating New Accounting Applications

There are services that only require the use of Diameter accounting. Since such services need to define the service specific AVPs that must be carried in the Accounting-Request/Answer messages, but do not need to define command codes, the rules on allocation of Accounting Application Identifiers is different from the ones defined in section

2.3.3.

When possible, a new Diameter accounting application SHOULD attempt to re-use any existing Diameter AVP, in order to reduce the possibility of having multiple AVPs that carry similar information.

Every Diameter accounting application specification MUST have an IANA assigned Application Identifier (see section 2.4).

1.2.5 Application Authentication Procedures

When possible, applications SHOULD be designed such that new authentication methods MAY be added without requiring changes to the application. This MAY require that new AVP values be assigned to represent the new authentication transform, or any other scheme that produces similar results. When possible, authentication frameworks, such as Extensible Authentication Protocol [EAP], SHOULD be used.

1.3 Requirements Language

In this document, the key words "MAY", "MUST", "MUST NOT", "OPTIONAL", "RECOMMENDED", "SHOULD", and "SHOULD NOT", are to be interpreted as described in [KEYWORDS].

1.4 Terminology

AAA
: Authentication, Authorization and Accounting.

Accounting
: The act of collecting information on resource usage for the purpose of trend analysis, auditing, billing or cost allocation.

Accounting record
: A session record represents a summary of the resource consumption of a user over the entire session. Accounting gateways creating the session 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).

AVP
: The Diameter protocol consists of a header followed by one or more Attribute-Value-Pair (AVP). The AVP includes a header and is used to encapsulate protocol-specific data (e.g. routing information) as well as authentication, authorization or accounting information.

Broker
: A broker is a business term commonly used in AAA infrastructures. A broker is either a relay, proxy or redirect agent, and MAY be operated by roaming consortiums.

Diameter Agent
: A Diameter Agent is a host that is providing either relay, proxy, redirect or translation services.

Diameter Client
: A Diameter Client is a device at the edge of the network that

performs access control. An example of a Diameter client is a Network Access Server (NAS) or a Foreign Agent (FA).

Diameter Node
A Diameter node is a host that implements the Diameter protocol, and acts either as a Client, Agent or Server.

Diameter Peer
A Diameter Peer is a Diameter Node to which a given Diameter Node has a direct transport connection.

Diameter Server
A Diameter Server is one that handles authentication, authorization and accounting requests for a particular realm. By its very nature, a Diameter Server MUST support Diameter 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 access device.

Home Realm
A Home Realm is the administrative domain with which the user maintains an account relationship.

Home Server
See Diameter Server.

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 event of a device reboot or other network problem that prevents the reception 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 Accounting-Multi-Session-Id. An example of a multi-session would be a MLP 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 [NAI], 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
In addition to forwarding requests and responses, proxies enforce policies 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.

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.

Relay

Relays forward requests and responses based on routing-related AVPs and realm routing table entries. Since relays do not enforce policies, 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 conversations 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 Routing or Policy proxies for other types. As with Routing proxies, redirect agents do not keep state with respect to conversations or NAS resources.

Roaming Relationships
: Roaming relationships include relationships between companies and ISPs, relationships among peer ISPs within a roaming consortium, and relationships between an ISP and a roaming consortia.

Session
: A session is a related progression of events devoted to a particular activity. Each application SHOULD provide guidelines as to when a session begins and ends. All Diameter packets with the same Session-Identifier are considered to be part of the same session.

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. An example of a session divided into several sub-sessions would be a dial-up connection in which the pre-authentication activity (call setup, resource allocation, etc.), interactive login, and PPP communication would all be sub-sessions.

Translation Agent
: A translation agent is a stateful Diameter node that performs protocol translation between Diameter and another AAA protocol.

Upstream
Upstream is used to identify the direction of a particular Diameter message from the access device towards the home server.

2.0 Protocol Overview

The base Diameter protocol is never used on its own. It is always extended for a particular application. Three Diameter applications are defined by companion documents: NASREQ [NASREQ], Mobile IP [DIAMMIP], CMS Security [CMS]. These applications are introduced in this document but specified elsewhere. Additional Diameter applications MAY be defined in the future (see Section 11.3).

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 IP. A Diameter Client that does not support both NASREQ and Mobile IP, 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 IP. A Diameter Server that does not support both NASREQ and Mobile IP, 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, by definition, protocol transparent, and MUST transparently 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 IP. A Diameter Proxy which does not support also both NASREQ and Mobile IP, MUST be referred to as "Diameter X Proxy" where X is the application which it supports, and not a "Diameter Proxy."

The Diameter CMS security application [CMS] contains two features:

1. A set of messages that allows a Diameter node to establish a security association, which is used to secure AVPs within a Diameter message, even though the message may traverse intermediate Diameter agents. A set of AVPs is also defined to sign and encrypt AVPs, as well as to transport certificates.

This feature MUST be supported by Diameter server and proxy

agents, SHOULD be supported by Diameter clients, and MAY be supported by relay and redirect agents.

2. A set of messages, known as PDSR and PDSA, allows a Diameter client to request that an agent establish a Diameter security association with a server in a specific realm. This feature MUST be supported by Diameter clients and Proxy agents, and MAY be supported by Diameter servers, relay and redirect agents.

The base Diameter protocol concerns itself with capabilities negotiation, how messages are sent and how peers may eventually be abandoned. The base protocol also defines certain rules that apply to all exchanges of messages 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. The Session-Id is then used in all subsequent messages to identify the user's session (see section 8.0 for more information). The communicating party may accept the request, or reject it by returning an answer message with 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.

2.1 Transport

The base Diameter protocol is run on port TBD of both TCP [TCP] and SCTP [SCTP] transport protocols (for interoperability test purposes port 1812 will be used until IANA assigns a port to the protocol). When used with TLS [TLS], the Diameter protocol is run on port TBD of

both TCP and SCTP.

Diameter clients MUST support either TCP or SCTP, while agents and servers MUST support both. Future versions of this specification MAY mandate that clients support SCTP.

A Diameter node MAY initiate connections from a source port other than the one that it declares it accepts incoming connections on, and

MUST be prepared to receive connections on port TBD. A given Diameter process MUST NOT use more than one transport connection to communicate with a given peer, unless multiple processes 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 attempted. This behavior is handled via the Tc timer, who's 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, SCTP SHOULD be tried first, followed by TCP. 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, in addition to interpreting ECONNREFUSED (a reset from the transport) and timed-out connection attempts.

If Diameter receives data up from TCP 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 (graceful closure is also compromised).

2.1.1 SCTP Guidelines

The following are guidelines for Diameter implementations that support SCTP:

1. For interoperability: All Diameter nodes MUST be prepared to

receive Diameter messages on any SCTP stream in the association.

2. To prevent blocking: All Diameter nodes SHOULD utilize all SCTP streams available to the association to prevent head-of-the-line blocking.

2.2 Securing Diameter Messages

Diameter clients, such as Network Access Servers (NASes) and Mobility Agents MUST support IP Security [SEC ARCH], and MAY support TLS [TLS]. Diameter servers MUST support TLS and IPsec. Operating the Diameter protocol without any security mechanism is not recommended.

It is suggested that IPsec can be used primarily at the edges and in

intra-domain traffic, such as using pre-shared keys between a NAS a local AAA proxy. This also eases the requirements on the NAS to support certificates. It is also suggested that inter-domain traffic would primarily use TLS. See sections 13.1 and 13.2 for more details on IPsec and TLS usage.

2.3 Diameter Application Compliance

Application Identifiers are advertised during the capabilities exchange phase (see section 2.5). For a given application, there are two different ways of advertising support. First, advertising support of the application via the Auth-Application-Id implies that the sender supports all authentication and authorization command codes, and the AVPs specified in the associated ABNFs, described in the specification. Second, advertising support of the application via the Acct-Application-Id implies that the sender supports the Accounting command codes defined in this specification, as well as the accounting AVPs defined in the application's specification.

An implementation MAY add arbitrary AVPs to any command defined in an application, including vendor-specific AVPs. Please refer to section 4.6 for details.

2.4 Application Identifiers

Each Diameter application MUST have an IANA assigned Application Identifier (see section 11.3). The base protocol does not require an Application Identifier 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 Identifier, which is used in the message forwarding process.

The following Application Identifier values are defined:

NASREQ	1 [NASREQ]
CMS Security	2 [CMS]
Mobile-IP	4 [DIAMMIP]
Relay	0xffffffff

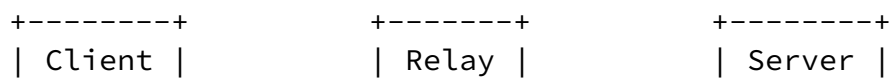
Relay and redirect agents MUST advertise the Relay Application Identifier, while all other Diameter nodes MUST advertise locally supported applications. The receiver of a Capabilities Exchange message advertising Relay service MUST assume that the sender supports all current and future applications.

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 is a transport level connection between two peers, used to send and receive Diameter messages. A session is a logical concept at the application layer, and is shared between an access device and a server, and is identified via the Session-Id AVP



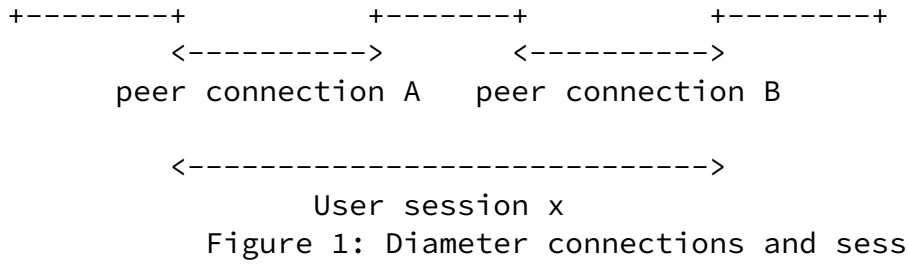


Figure 1: Diameter connections and sessions

In the example provided in figure 1, peer connection A is established between the Client and its local 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.

2.6 Peer Table

The Diameter Peer Table is used in message forwarding, and referenced by the Realm 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 4.4. This field contains the contents of the Origin-Host AVP found in the CER or CEA message.
- Status. 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.
- TLS Enabled. Specifies whether TLS is to be used when communicating with the peer.
- Additional security information, when needed (e.g. keys, certificates)

2.7 Realm-Based Routing Table

All Realm-Based routing lookups are performed against what is commonly known as the Realm Routing Table (see section 12.0). A Realm Routing Table Entry contains the following fields:

- Realm Name. This is the field that is typically 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. A route entry can have a different destination based on the Acct-Application-Id for accounting messages) or Auth-Application-Id (for non-accounting messages) of the message. 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 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 server.
    2. RELAY - All Diameter messages that fall within this category MUST be routed to a next hop server, without modifying any non-routing AVPs. See section 6.1.8 for relaying guidelines
    3. PROXY - All Diameter messages that fall within this category MUST be routed to a next hop server. The local server MAY apply its local policies to the message by including new AVPs to the message prior to routing. See section 6.1.8 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 6.1.7 for redirect guidelines.
- Server Identifier. One or more servers the message is to be routed to. These servers MUST also be present in the Peer table. When the Local Action is set to RELAY or PROXY, this field

contains the identity of the server(s) the message must be routed to. When the Local Action field is set to REDIRECT, this field contains the identity of one or more servers the message should be redirected to.

- Static or Dynamic. Specifies whether a route entry was

statically configured, or dynamically discovered.
- Expiration time. Specifies the time which a dynamically discovered route table entry expires.

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 2.0. Relay agents MUST NOT reorder AVPs, 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 Identifier (see section 2.5) 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 client and servers, the Diameter protocol introduces relay, proxy, redirect, and translation agents, each of which is defined in Section 1.3. These 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 it is notified otherwise, or by expiration. 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 Realm Routing Table, as is defined further in section 2.8.

Relays MAY 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. Further, Relays' inherent simplicity implies that they are stateless and therefore SHOULD NOT maintain session state but MUST maintain

transaction state.

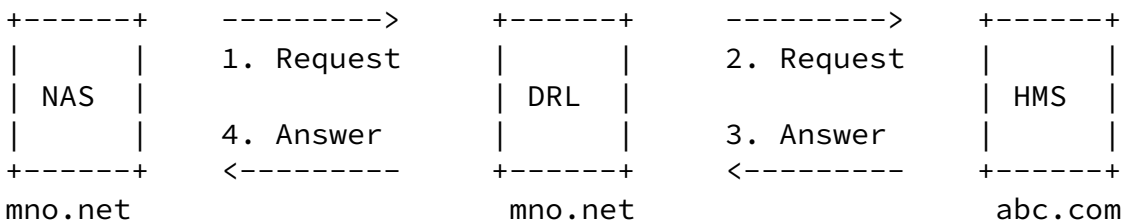


Figure 2: Relaying of Diameter messages

The example provided in Figure 2 depicts a request issued from NAS, which is an access device, for the user bob@abc.com. Prior to issuing the request, NAS performs a Diameter route lookup, using "abc.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 abc.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 Identifier.

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.

It is important to note that although proxies MAY provide a value-add function for NASes, they do not allow access devices to use the Diameter CMS Security application, since modifying messages breaks authentication.

Proxies MAY 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.

Proxies that wish to limit resources MUST be stateful, and all Proxies MUST maintain transaction state.

Proxy agents MUST NOT allow CMS security to be established between

two peers if it expects to modify ANY non-routing AVP in messages exchanged between the peers. See [CMS] for more information.

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 provide Realm to Server address resolution and MAY also provide User to Server address resolution. These redirect agents would make use of the Diameter routing table or optionally, a user table to determine where a given request should be forwarded. When a request is received by a redirect agent, a special answer is created, which includes the identity of the Diameter server(s) the originator of the request should contact directly.

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. Further, since redirect agents never relay requests, they are not required to maintain transaction state.

The example provided in Figure 3 depicts a request issued from the access device, NAS, for the user bob@abc.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 abc.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.

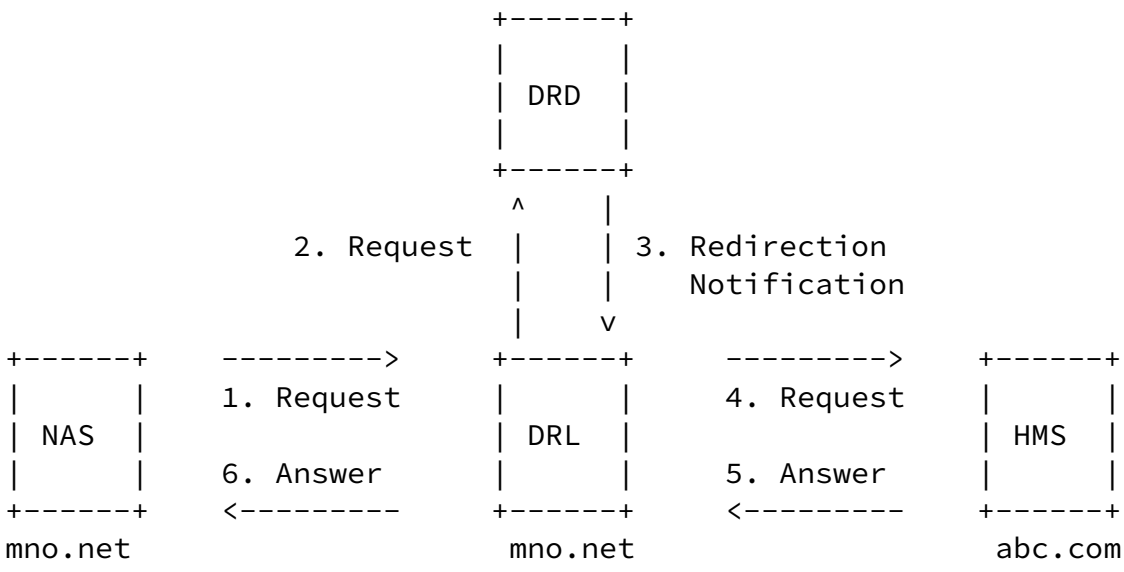


Figure 3: Redirecting a Diameter Message

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.

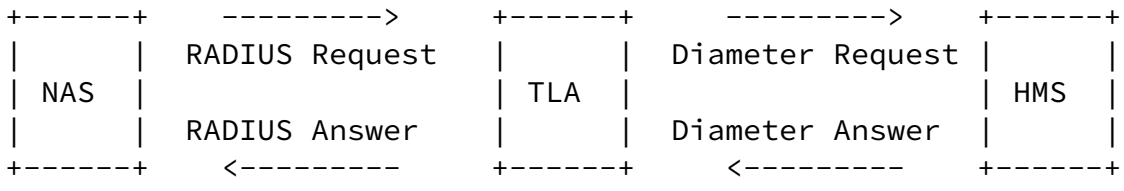
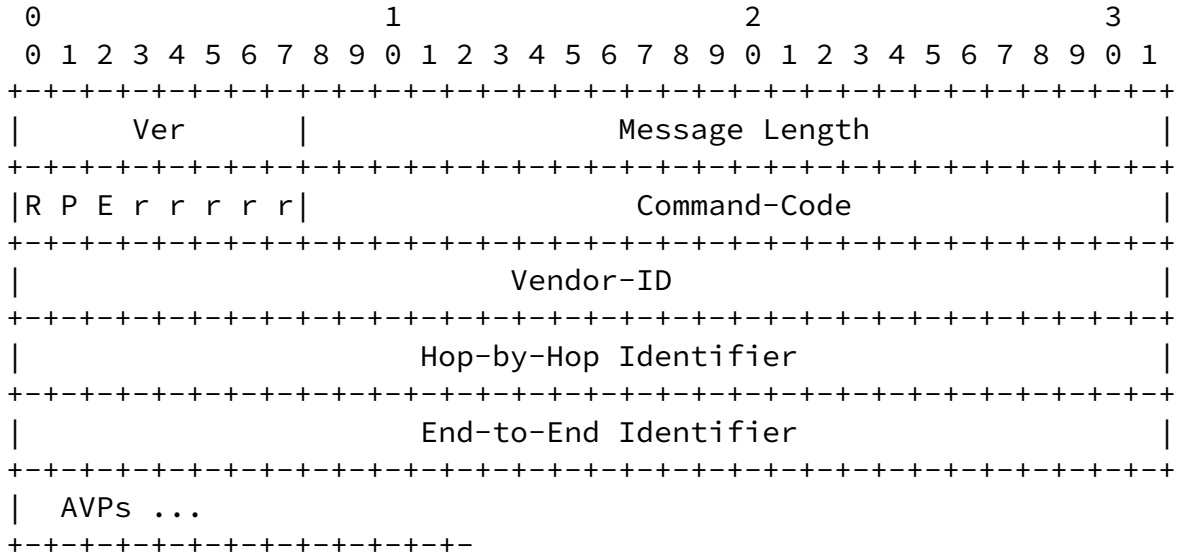


Figure 4: Translation of RADIUS to Diameter

3.0 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.

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 ABNF described for this command. Messages with the 'E' bit set are commonly referred to as an error messages. This bit MUST NOT be set in request messages. See section 7.2.
r(eserved) - these flag bits are reserved for future use, and MUST be set to zero, otherwise an error MUST be sent to the sender.

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 11.2).

Vendor-ID
In the event that the Command-Code field contains a vendor specific command, the four-octet Vendor-ID field contains the IANA assigned "SMI Network Management Private Enterprise Codes" [ASSIGN NO] value. If the Command-Code field contains an IETF standard Command, the Vendor-ID field MUST be set to zero (0). Any vendor wishing to implement a vendor-specific Diameter command MUST use their own Vendor-ID along with their privately managed Command-Code address space, guaranteeing that they will not collide with

any other vendor's vendor-specific command, nor with future IETF applications. All vendor-specific Diameter commands require Information RFCs documenting the command.

Hop-by-Hop Identifier
: The Hop-by-Hop Identifier is an Unsigned32 field 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 Unsigned32 field 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 relay agents. The combination of the Origin-Host 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 ABNF specification

Every Command Code defined MUST include a corresponding ABNF specification, which is used to define the AVPs that MUST or MAY be present. The following format is used in the definition:

```
command-def      = command-name "::=" diameter-message

command-name     = diameter-name
                   ; The command-name has to be Command name,
```

```
                    ; defined in the base or extended Diameter
                    ; specifications.

diameter-name     = ALPHA *(ALPHA / DIGIT / "-")

diameter-message = header  [ *fixed] [ *required] [ *optional]
                    [ *fixed]

header            = "< Diameter-Header:" [vendor-id] command-id
                    [r-bit] [p-bit] [e-bit] ">"

vendor-id         = 1*DIGIT ":"
                    ; The optional vendor-id is used to define
                    ; vendor specific commands

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.

qual              = [min] "*" [max]
```

```
                        ; See ABNF conventions, RFC 2234 section 6.6.
                        ; 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.
                        ;
                        ; NOTE:  "[" and "]" have a different meaning
                        ; than in ABNF (see the optional rule, above).
                        ; 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'.

      min               = 1*DIGIT
                        ; The minimum number of times the element may
                        ; be present. The default value is zero.

      max               = 1*DIGIT
                        ; The maximum number of times the element may
                        ; be present. 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, which does not conflict with the
                        ; required or fixed position AVPs defined in
                        ; the command code definition.
```

The following is a definition of a fictitious command code:

```
   Example-Request ::= < Diameter-Header: 9999999, REQ, PXY >
                         { User-Name }
                       * { Origin-Host }
                       * [ AVP ]
```

3.3 Diameter Command Naming Conventions

Diameter commands 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 must 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.

4.0 Diameter AVPs

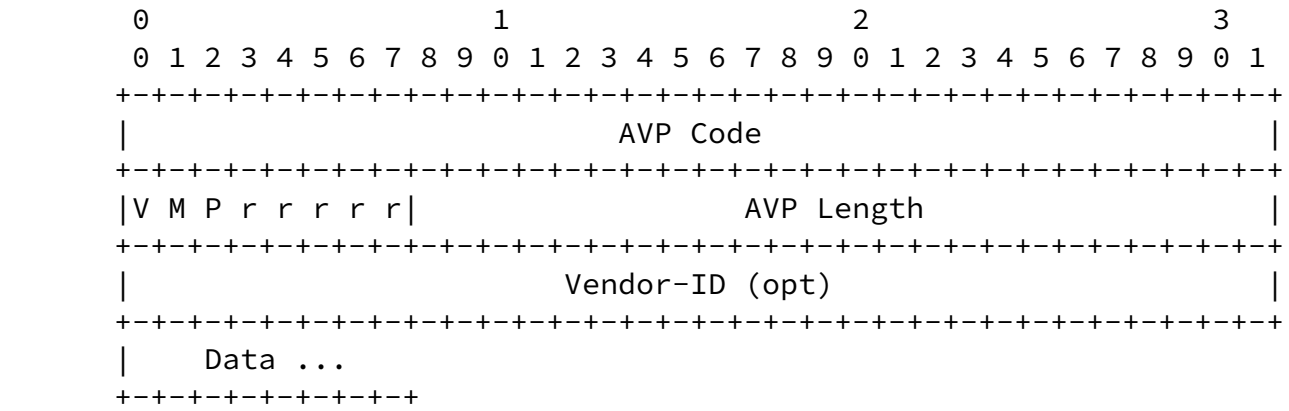
Diameter AVPs carry specific authentication, accounting, authorization, routing and security information as well as configuration details for the request and reply.

Some AVPs MAY be listed more than once. The effect of such an AVP is specific, and is specified in each case by the AVP description.

Each AVP of type OctetString MUST be padded to align on a 32-bit boundary, while other AVP types align naturally. NULL 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 0 through 255, with the Vendor-Id set to zero (0) are reserved for backward compatibility with RADIUS. AVP numbers 256 and above are used for Diameter, which are allocated by IANA (see section 11.1).

AVP Flags
The AVP Flags field informs the receiver how each attribute must be handled. The 'r' (reserved) bits are unused and SHOULD be set to 0. Note that subsequent Diameter applications MAY define additional bits within the AVP Header, and an unrecognized bit SHOULD be considered an error. The 'P' bit is defined in [CMS].

The 'M' Bit, known as the Mandatory bit, indicates whether support of the AVP is required. If an AVP with the 'M' bit set is received by a Diameter client, server, proxy, or translation agent and

either the AVP or its value is unrecognized, the message MUST be rejected. Diameter Relay and Redirect agents MUST NOT reject messages with unrecognized AVPs.

The 'M' bit MUST be set according to the rules defined for the AVP containing it. In order to preserve interoperability, a Diameter implementation MUST be able to exclude from a Diameter message any Mandatory AVP which is neither defined in the base Diameter standard nor in any of the Diameter Application specifications governing the message in which it appears. It MAY do this in one of the following ways:

1) If a message is rejected because it contains a Mandatory AVP which is neither defined in the base Diameter standard nor in any of the Diameter Application specifications governing the message in which it appears, the implementation may resend the message without the AVP, possibly inserting additional standard AVPs instead.

2) A configuration option may be provided on a system wide, per peer, or per realm basis that would allow/prevent particular Mandatory AVPs to be sent. Thus an administrator could change the configuration to avoid interoperability problems.

Diameter implementations are required to support all Mandatory AVPs which are allowed by the message's formal syntax and defined either in the base Diameter standard or in one of the Diameter Application specifications governing the message.

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.

Unless otherwise noted, AVPs will have the following default AVP Flags field settings:

The 'M' bit MUST be set. The 'V' bit MUST NOT be set.

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 SHOULD be rejected.

4.2 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" [ASSIGN NO] value, encoded in network byte order. Any vendor 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, nor with future IETF applications.

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 SHOULD NOT use the zero (0) vendor ID.

4.3 AVP Base Data Format

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 AVP Base 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 4 octets in length be 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.

Derived AVP Data Formats

In addition to using the AVP Base Data Formats, applications may define data formats derived from the AVP Base Data Formats. An application that defines new AVP Derived Data Formats MUST include them in a section entitled "AVP Derived 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.

The below AVP Derived Data Formats are commonly used by applications.

IPAddress
The IPAddress format is derived from the OctetString AVP Base Format. It represents 32 bit (IPv4) [IPV4] or 128-bit (IPv6) [IPV6] address, most significant octet first. The format of the address (IPv4 or IPv6) is determined by the length. If the attribute value is an IPv4 address, the AVP Length field MUST be 12 (16 if 'V' bit is enabled); otherwise, the AVP Length field MUST be set to 24 (28 if the 'V' bit is enabled) for IPv6 addresses.

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 [SNTP].

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 [SNTP] describes a procedure to extend the time to 2104. This procedure MUST be used 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 [UFT8] transformation format described in RFC 2279.

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 UTF-8 encoding of a code point or are outside this range are prohibited.

The use of control codes SHOULD be avoided. When it is necessary to represent a newline, 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 encoding is identical to the US-ASCII encoding.

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 size of an UTF8String is measured in octets, not characters.

DiameterIdentity

The DiameterIdentity format is derived from the OctetString AVP Base Format. It uses the UTF-8 encoding and has the same requirements as the UTF8String:

```
DiameterIdentity  = fqdn
```

A Diameter node must be uniquely identified by its DiameterIdentity, which contains 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, one single FQDN should be picked at startup, and used as the only DiameterIdentity for that node, whatever the connection it is sent on. The

DiameterIdentity value is used to uniquely identify a Diameter node for purposes of duplicate connection and routing loop detection.

DiameterURI

The DiameterURI MUST follow the Uniform Resource Identifiers (URI) syntax [URI] rules specified below:

```
"aaa://" fqdn [ port ] [ transport ] [ protocol ]

                    ; No transport security

"aaas://" fqdn [ port ] [ transport ] [ protocol ]

                    ; Transport security used

fqdn                = Fully Qualified Host Name

port                = ":" 1*DIGIT

                    ; One of the ports used to listen for ;
                    incoming connections. ; If absent, ; the
                    default Diameter port (TBD) is ; assumed.

transport           = ";transport=" transport-protocol

                    ; One of the transports used to listen ; for
                    incoming connections. If absent, ; the default
                    SCTP [SCTP] protocol is ; assumed. 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.abc.com;transport=tcp
aaa://host.abc.com:6666;transport=tcp
```

```
aaa://host.abc.com;protocol=diameter
aaa://host.abc.com:6666;protocol=diameter
```



```
aaa://host.abc.com:6666;transport=tcp;protocol=diameter
aaa://host.abc.com:1813;transport=udp;protocol=radius
aaas://host.abc.com;transport=tcp
aaas://host.abc.com;protocol=diameter
```

Enumerated

Enumerated is derived from the Integer32 AVP Base Format. This 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. It uses the UTF-8 encoding and has the same requirements as the UTF8String. 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 1.2.3.4/24. In this case all IP numbers from 1.2.3.0 to 1.2.3.255 will match. The bit width MUST be valid for the IP version and the IP number MUST NOT have bits set beyond the mask.

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.

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. The first rule SHOULD be "deny in ip !assigned".

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. 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.

The rule syntax is a modified subset of ipfw(8) from FreeBSD,

and the ipfw.c code may provide a useful base for implementations.

QoSFilterRule

The QosFilterRule format is derived from the OctetString AVP Base Format. It uses the UTF-8 encoding and has the same requirements as the UTF8String. Packets may be marked or metered 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)
DSCP values	(no mask or range)

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 treated as best effort.

QoSFilterRule filters MUST follow the format:

action dir proto from src to dst [options]

tag - Mark packet with a specific DSCP [DIFFSERV]. The DSCP option MUST be included.
meter - Meter traffic. The metering options MUST be included.

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 1.2.3.4/24. In this case all IP numbers from 1.2.3.0 to 1.2.3.255 will match. The bit width MUST be valid for the IP version and the IP number MUST NOT have bits set beyond the mask.

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.

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. The first rule SHOULD be "deny in ip !assigned".

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).

options:

DSCP <color>
color values as defined in [DIFFSERV]. 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 [DIFFSERVAF], and MUST be present if the action is set to meter. The rate option is

the throughput, in bits per second, which is used by the access device to mark packets. Traffic above the rate is marked with the color_over codepoint, while 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 [DIFFSERVAF] (e.g. AF13).

The metering option also supports the strict limit on traffic required by Expedited Forwarding, as defined in [DIFFSERVEF]. The color_over option may contain the keyword "drop" to prevent forwarding of traffic that exceeds the rate parameter.

The rule syntax is a modified subset of ipfw(8) from FreeBSD, and the ipfw.c code may provide a useful base for implementations.

4.5 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 4.0.

The AVP Code numbering space of all AVPs included in a Grouped AVP is the same as for non-grouped AVPs. Further, if any of the AVPs encapsulated within a Grouped AVP has the 'M' (mandatory) bit set, the Grouped AVP itself MUST also include the 'M' bit set.

Every Grouped AVP defined MUST include a corresponding grammar, using ABNF [ABNF] (with modifications), as defined below.

```
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]
                        [ *fixed]

      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.

      fixed             = [qual] "<" avp-spec ">"

      required          = [qual] "{" avp-spec "}"

      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.

      qual              = [min] "*" [max]
                        ; See ABNF conventions, RFC 2234 section 6.6.
                        ; The absence of any qualifiers implies that
                        ; one and only one such AVP MUST be present.
                        ;
                        ; NOTE:  "[" and "]" have a different meaning
                        ; than in ABNF (see the optional rule, above).
                        ; 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'.

      min               = 1*DIGIT
                        ; The minimum number of times the element may
                        ; be present.

      max               = 1*DIGIT
                        ; The maximum number of times the element may
                        ; be present.

      avp-spec          = name-fmt
```

```
                        ; 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, which does not conflict with the
                        ; required or fixed position AVPs defined in
                        ; the command code definition.
```

4.5.1 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 ABNF grammar:

```
   Example-AVP  ::= < AVP Header: 999999 >
                    { Origin-Host }
                  1*{ Session-Id }
                   *[ AVP ]
```

An Example-AVP with Grouped Data follows.

The Origin-Host AVP is required. In this case:

```
   Origin-Host = "abc.com".
```

One or more Session-Ids must follow. Here there are two:

```
   Session-Id =
     "grump.abc.com:33041;23432;893;0AF3B81"

   Session-Id =
     "grump.abc.com:33054;23561;2358;0AF3B82"
```

optional AVPs included are

```
   Recovery-Policy = <binary>
      2163bc1d0ad82371f6bc09484133c3f09ad74a0dd5346d54195a7cf0b35
      2cabc881839a4fdcfbc1769e2677a4c1fb499284c5f70b48f58503a45c5
```

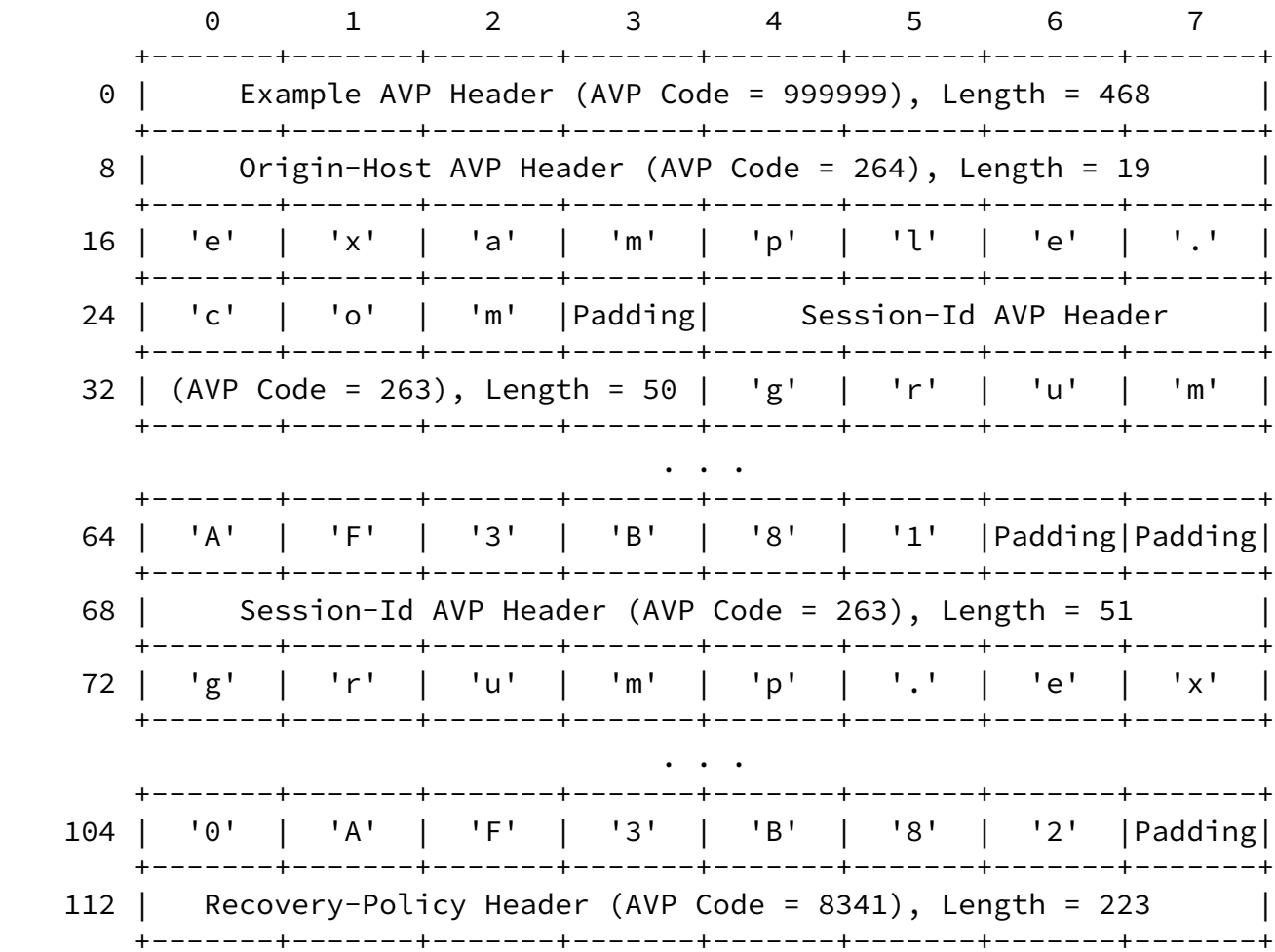
```
      c2d6943f82d5930f2b7c1da640f476f0e9c9572a50db8ea6e51e1c2c7bd
      f8bb43dc995144b8dbe297ac739493946803e1cee3e15d9b765008a1b2a
      cf4ac777c80041d72c01e691cf751dbf86e85f509f3988e5875dc905119
      26841f00f0e29a6d1ddc1a842289d440268681e052b30fb638045f7779c
      1d873c784f054f688f5001559ecff64865ef975f3e60d2fd7966b8c7f92

   Futuristic-Acct-Record = <binary>
      fe19da5802acd98b07a5b86cb4d5d03f0314ab9ef1ad0b67111ff3b90a0
```

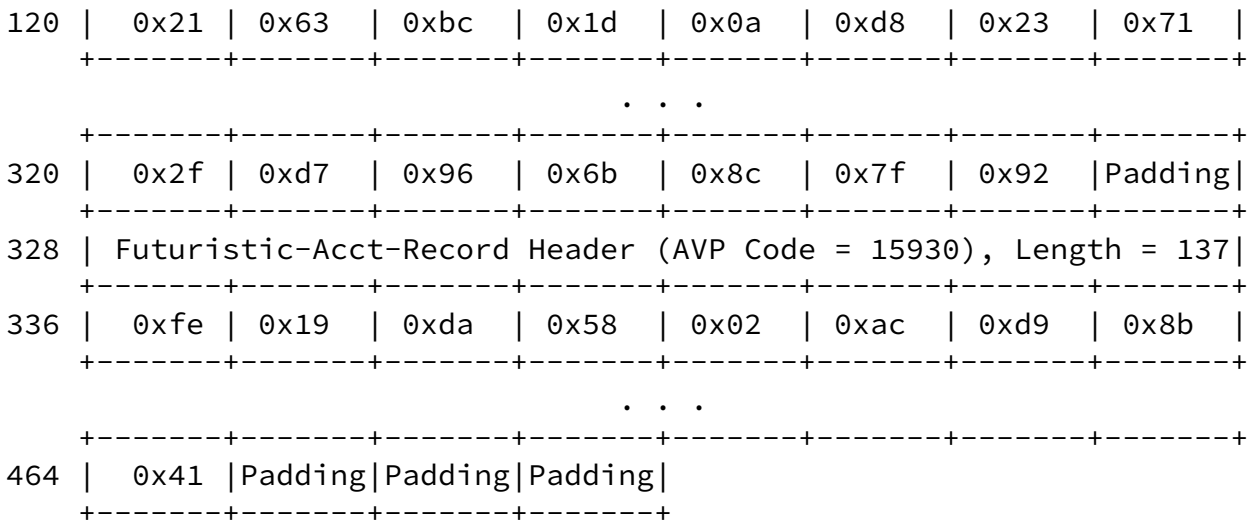


```
      57fe29620bf3585fd2dd9fcc38ce62f6cc208c6163c008f4258d1bc88b8
      17694a74ccad3ec69269461b14b2e7a4c111fb239e33714da207983f58c
      41d018d56fe938f3cbf089aac12a912a2f0d1923a9390e5f789cb2e5067
      d3427475e49968f841
```

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).

This AVP would be encoded as follows:


0 1 2 3 4 5 6 7
0 Example AVP Header (AVP Code = 999999), Length = 468
8 Origin-Host AVP Header (AVP Code = 264), Length = 19
16 'e' 'x' 'a' 'm' 'p' 'l' 'e' '.'
24 'c' 'o' 'm' Padding Session-Id AVP Header
32 (AVP Code = 263), Length = 50 'g' 'r' 'u' 'm'
. . .
64 'A' 'F' '3' 'B' '8' '1' Padding Padding
68 Session-Id AVP Header (AVP Code = 263), Length = 51
72 'g' 'r' 'u' 'm' 'p' '.' 'e' 'x'
. . .
104 '0' 'A' 'F' '3' 'B' '8' '2' Padding
112 Recovery-Policy Header (AVP Code = 8341), Length = 223



4.6 Diameter Base Protocol AVPs

The following table describes the Diameter AVPs defined in the base protocol, their AVP Code values, types, possible flag values and whether the AVP MAY be encrypted. For the originator of a Diameter message, "MAY Encr" means that if a message containing that AVP is

to be sent via a proxy/agent then the message MUST NOT be sent unless there is a DSA between the originator and the recipient OR the originator has locally trusted configuration that indicates that CMS need not be used.

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	MAY	SHLD NOT	MUST NOT	MAY Encr
Accounting-Interim-Interval	482	9.8.2	Unsigned32	M	P		V	Y
Accounting-Realtime-Required	483	9.8.7	Unsigned32	M	P		V	Y
Accounting-Multi-Session-Id	50	9.8.5	UTF8String	M	P		V	Y

Accounting-Record-Number	485	9.8.3	Unsigned32	M	P		V	Y
Accounting-Record-Type	480	9.8.1	Enumerated	M	P		V	Y
Accounting-RADIUS-Session-Id	44	9.8.4	OctetString	M	P		V	Y
Accounting-Sub-Session-Id	287	9.8.6	Unsigned64	M	P		V	Y
Acct-Application-Id	259	6.9	Integer32	M	P		V	N
Auth-Application-Id	258	6.8	Integer32	M	P		V	N
Auth-Request-Type	274	8.7	Enumerated	M	P		V	N
Authorization-Lifetime	291	8.9	Unsigned32	M	P		V	N
Auth-Grace-Period	276	8.10	Unsigned32	M	P		V	N
Auth-Session-State	277	8.11	Enumerated	M	P		V	N
Re-Auth-Request-Type	285	8.12	Enumerated	M	P		V	N
Class	25	8.20	OctetString	M	P		V	Y
Destination-Host	293	6.5	DiamIdent	M	P		V	N
Destination-Realm	283	6.6	UTF8String	M	P		V	N
Disconnect-Cause	273	5.4.3	Enumerated	M	P		V	N
Error-Message	281	7.3	OctetString		P		V,M	N
Error-Reporting-Host	294	7.4	UTF8String		P		V,M	N
Failed-AVP	279	7.5	Grouped	M	P		V	N
Firmware-Revision	267	5.3.4	Unsigned32				P,V,M	N
Host-IP-Address	257	5.3.5	IPAddress	M	P		V	N

				AVP Flag rules				
Attribute Name	AVP Code	Section Defined	Data Type	MUST	MAY	SHLD NOT	MUST NOT	MAY Encr

Multi-Round-Time-Out	272	8.19	Unsigned32	M	P		V	Y
Origin-Host	264	6.3	DiamIdent	M	P		V	N
Origin-Realm	296	6.4	UTF8String	M	P		V	N
Origin-State-Id	278	8.16	Unsigned32	M	P		V	N
Product-Name	269	5.3.7	UTF8String				P,V,M	N
Proxy-Host	280	6.7.3	IPAddress	M			P,V	N
Proxy-Info	284	6.7.2	Grouped	M			P,V	N
Proxy-State	33	6.7.4	OctetString	M			P,V	N
Redirect-Host	292	6.11	DiamURI	M	P		V	N
Redirect-Host-Usage	261	6.12	Enumerated	M	P		V	N
Redirect-Max-Cache-Time	262	6.13	Unsigned32	M	P		V	N
Result-Code	268	7.1	Unsigned32	M	P		V	N
Route-Record	282	6.7.1	DiamIdent	M			P,V	N
Session-Id	263	8.8	UTF8String	M	P		V	Y
Session-Timeout	27	8.13	Unsigned32	M	P		V	N
Session-Binding	270	8.17	Unsigned32	M	P		V	Y
Session-Server-Failover	271	8.18	Enumerated	M	P		V	Y
Supported-Vendor-Id	265	5.3.6	Unsigned32	M	P		V	N
Termination-Cause	295	8.15	Enumerated	M	P		V	N
User-Name	1	8.14	UTF8String	M	P		V	Y
Vendor-Id	266	5.3.3	Unsigned32	M	P		V	N
Vendor-Specific-Application-Id	260	6.10	Grouped	M	P		V	N

5.0 Diameter Peers

This section describes how Diameter nodes establish connections and communicate with peers.

5.1 Peer Connections

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 not 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 will make it possible for simpler and more robust deployment of Diameter services. In order to promote interoperable implementations of Diameter peer discovery, the following mechanisms are described. These are based on existing IETF standards. The first option (manual configuration) MUST be supported by all DIAMETER nodes, while the latter two options (SRVLOC and DNS) MAY be supported.

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 (manual) configured Diameter agent locations. These will be used if they exist and respond.

2. The Diameter implementation uses SLPv2 [SLP] to discover Diameter services. The Diameter service template [TEMPLATE] is included in Appendix A. It is recommended that SLPv2 security be deployed (this requires distributing keys to SLPv2 agents). This is discussed further in Appendix A.

   SLPv2 will allow Diameter implementations to discover the location of Diameter agents in the local site, as well as their characteristics. Diameter agents with specific capabilities (say support for the Mobile IP application) can be requested, and only those will be discovered.

3. 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. This could be deduced, for example, from the 'realm' in a NAI that a Diameter implementation needed to perform a Diameter operation on.

   3.1 The services relevant for the task of transport protocol selection are those with NAPTR service fields with values "AAA+D2x" and "AAAS+D2X", where x is a letter that corresponds to a transport protocol supported by the domain. This specification defines D2T for TCP and D2S for SCTP. We also establish an IANA registry for NAPTR service name to transport protocol mappings.

   These NAPTR records provide a mapping from a domain, to the SRV record for contacting a server with the specific transport protocol in the NAPTR services field. The resource record will contain an empty regular expression and a replacement value, which is the SRV record for that particular transport protocol. If the server supports multiple transport protocols, there will be multiple NAPTR records, each with a different service value. As per RFC 2915 [NAPTR], the client discards any records whose services fields are not applicable. For the purposes of this specification, several rules are defined.

3.2 First, a client resolving a AAAS URI MUST discard any services that do not contain "AAAS" as the protocol in the service field. The converse is not true, however. A client resolving an AAA URI SHOULD retain records with "AAAS" as the protocol, if the client supports TLS. Second, a client

MUST discard any service fields that identify a resolution service whose value is not "D2X", for values of X that indicate transport protocols supported by the client. The NAPTR processing as described in RFC 2915 will result in discovery of the most preferred transport protocol of the server that is supported by the client, as well as an SRV record for the server. It will also allow the client to discover if TLS is available and its preference for its usage.

The domain suffixes in the NAPTR replacement field SHOULD match the domain of the original query. It is not necessary for the domain suffixes in the NAPTR replacement field to match the domain of the original query.

3.3 If no NAPTR records are found, the requester queries for those address records for the destination address, '_diameters._sctp'.realm or '_diameters._tcp'.realm when using TLS or '_diameter._sctp'.realm or '_diameter._tcp'.realm when not using TLS. Address records include A RR's, AAAA RR's or other similar records, chosen according to the requestor's network protocol capabilities. If the DNS server returns no address records, the requestor gives up.

For NAPTR records with AAAS protocol fields, if the server is using a site certificate, the domain name in the 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 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.

A dynamically discovered peer causes an entry in the Peer Table (see section 2.7) 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 2.8) 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, supported Diameter applications, 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 applications 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 2.5) MUST be interpreted as having common applications with the peer.

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.

The CER and CEA messages MUST NOT be proxied, or redirected.

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 (see Section 7.2) 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 Identifier).

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 [SCTP], which allows for connections to span multiple interfaces, hence, 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 ]
            * [ 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 [SCTP], which allows 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 ]
         * [ 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" [ASSIGNNO] value assigned to the vendor of the Diameter device. In combination with the Supported-Vendor-Id AVP (section 5.3.6), this MAY be used in order to know which vendor specific attributes may be sent to the peer. It is also 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 very useful debugging information.

A Vendor-Id value of zero in the CER or CEA messages is reserved and indicates that the Diameter peer is in the experimental or concept stage and that an IANA Private Enterprise Number has yet to be obtained by the implementer.

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 IPAddress 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 [SCTP] MUST be advertised in the CER and CEA messages by including a Host-IP-Address AVP for each address. This AVP MUST ONLY be used in the CER and CEA messages.

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" [ASSIGN NO] value assigned to a vendor other than the device 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 commands and/or AVPs defined by the vendor identified in this AVP.

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 be forwarded, and are likely in flight, which would otherwise cause a race condition.

The receiver of the Disconnect-Peer-Answer initiates the transport disconnect.

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 }
```

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 ]
```

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.

BUSY                            1
```

The peer's internal resources are constrained, and it has determined that the transport connection needs to be shutdown.

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.

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 ]
```

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 ]
                 [ Original-State-Id ]
```

5.5.3 Transport Failure Algorithm

The transport failure algorithm is defined in [AAATRANS]. 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/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, all messages in the queue are sent to an alternate agent, if possible. 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 [AAATRANS], which is used to open, close, failover, probe, and reopen transport connections. Note in particular that [AAATRANS] 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.

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-Snd-CEA R-Reject	R-Open
	Stop	R-Snd-DPR	Closing
	R-Rcv-DPR	R-Snd-DPA, R-Disc	Closed
	R-Peer-Disc	R-Disc	Closed
	R-Rcv-CER	R-Snd-CEA	R-Open
	R-Rcv-CEA	Process-CEA	R-Open
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
	I-Rcv-CER	I-Snd-CEA	I-Open
	I-Rcv-CEA	Process-CEA	I-Open
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-Conn-Ack	an acknowledgement is received confirming that the transport connection is established.
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 sent by its peer with its own Origin-Host. If the local Diameter entity's Origin-Host is higher than the peer's, a Win-Election event is issued locally.

The comparison proceeds by considering the shorter OctetString to be null-padded to the length of the longer, then performing an octet-by-octet unsigned comparison with the first octet being most significant. Hanging octets are assumed to have value 0x80, but dimpled octets are ignored.

6.0 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.
- 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 2.7). Otherwise, the request is routed based on the Destination-Realm only (see sections 6.1.6).

The Destination-Realm AVP MUST be present if the message is proxiable. Proxiable request messages MUST also contain either an Acct-Application-Id AVP or an Auth-Application-Id AVP. A message that MUST NOT be relayed, proxied or redirected MUST NOT include the Destination-Realm in its ABNF. The value of the Destination-Realm AVP MAY be extracted from the User-Name AVP, or other application-specific methods.

When a message is received, the message is processed in the following order:

1. If the message is destined for the local host, the procedures listed in section 6.1.4 are followed.
2. 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.
3. The procedures listed in section 6.1.6 are followed, which is known as Request Routing.
4. If none of the above is successful, an answer is returned with the Result-Code set to DIAMETER_UNABLE_TO_DELIVER.

For routing of Diameter messages to work within an administrative domain, all Diameter nodes within the realm MUST be peers. If intermediate nodes are desired (see Figure 5), the destination node MUST be in a subrealm and routes to that subrealm MUST exist in the routing table on the sending node and all intermediate nodes. Figure 5 shows an example of a hierarchical network that requires the use of subrealms. In such a network, routing must be performed with longest match from right.

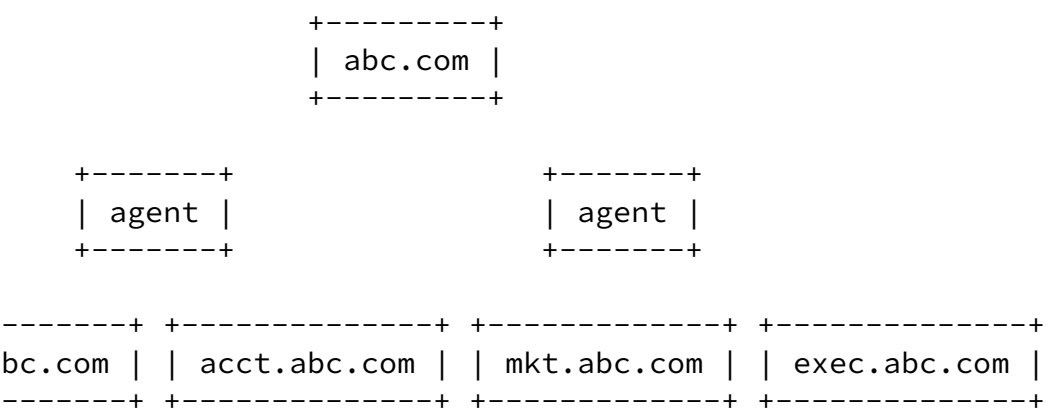


Figure 5: Hierarchical administrative domain

Note the processing rules contained in this section are intended to be used as general guidelines to Diameter developers. Certain implementations MAY use different methods than the ones described here, and still comply with the protocol specification.

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 should be set to the appropriate value
- the 'R' bit should be set
- the End-to-End Identifier should be 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.

- either an Acct-Application-Id AVP or an Auth-Application-Id AVP must be included if the request is proxiable.

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 MUST 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.

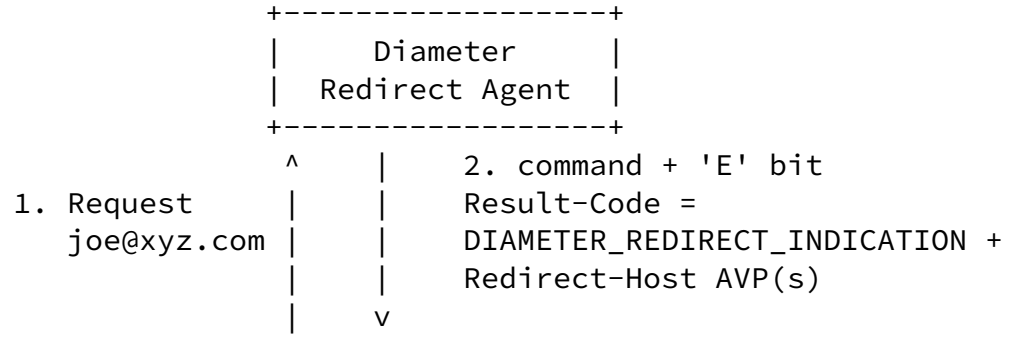
6.1.6 Request Routing

Diameter request message routing is done via realms. A Diameter message that is able to be proxied MUST include the target realm in the Destination-Realm AVP. 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 MAY have a list of externally supported realms. When a request is received that includes a realm that is not locally supported, the message is routed to the peer configured in the Realm Routing Table table (see section 2.8).

6.1.7 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.



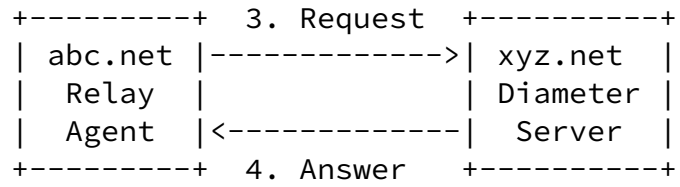


Figure 6: Diameter Redirect Agent

Redirect agents MAY also include the certificate of the servers in the Redirect-Host AVP(s). These certificates are encapsulated in a AAA-Node-Cert AVP [CMS].

The receiver of the answer message with the 'E' bit set, and the Result-Code AVP set to DIAMETER_REDIRECT_INDICATION uses the hop-by-hop field in the Diameter header to identify the request in the pending message queue (see Section 5.3) 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.

6.1.8 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.

Relay and Proxy agents MAY include the Proxy-Info AVP in requests if

it requires access any local state information when the corresponding response is received. Alternatively, it MAY simply use local storage to store state information.

The message is then forwarded to the next hop, as identified in the Realm Routing Table.

Figure 7 provides an example of message routing using the procedures listed in these sections.

(Origin-Host=nas.mno.net) (Origin-Host=nas.mno.net)

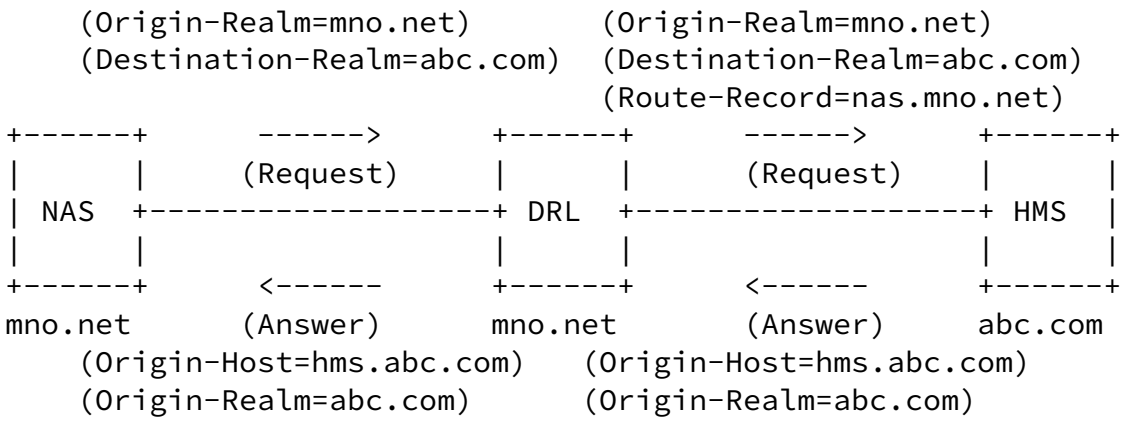


Figure 7: Routing of Diameter messages

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 7.2) 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.

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 UTF8String. 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 UTF8String, 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.

Request messages whose ABNF does not list the Destination-Realm AVP as a mandatory AVP are inherently non-routable messages.

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, and therefore MUST NOT be protected using the Diameter CMS Security application [CMS].

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. The Grouped Data field has the following ABNF 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, and contains state local information, and MUST be treated as opaque data.

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 2.5). The Auth-Application-Id MUST also be present in all Authentication and/or Authorization messages that are defined in a separate Diameter specification and have an Application ID assigned.

This AVP SHOULD be placed as close to the Diameter header as possible.

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 2.5). The Acct-Application-Id MUST also be present in all Accounting messages that are defined in a separate Diameter specification and have an Application ID assigned.

This AVP SHOULD be placed as close to the Diameter header as possible.

6.10 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. Either the Auth-Application-Id or the Acct-Application-Id AVP MAY be present. Both AVPs MAY be present if they both contain the same value.

This AVP MUST also be present in all vendor-specific commands defined in the vendor-specific application.

This AVP SHOULD be placed as close to the Diameter header as possible.

AVP Format

```
<Vendor-Specific-Application-Id> ::= < AVP Header: 260 >
                                     1* [ Vendor-Id ]
                                     0*1{ Auth-Application-Id }
                                     0*1{ Acct-Application-Id }
```

6.11 Redirect-Host AVP

The Redirect-Host AVP (AVP Code 292) is of type DiameterURI. 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 the host identified in this AVP. The server contained in the Redirect-Host SHOULD be used for all messages pertaining to this session.

6.12 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 dictates 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 MAY be sent to the host specified in the Redirect-Host AVP.

ALL_REALM 2
All messages destined for the realm requested MAY 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 MAY be sent to the host specified in the Redirect-Host AVP.

ALL_APPLICATION 4

All messages for the application requested MAY 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 MAY be sent to the host specified in the Redirect-Host AVP.

ALL_USER 6
All messages for the user requested MAY be sent to the host specified in the Redirect-Host AVP.

6.13 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, will be cached. Note that once a host created due to a redirect indication is

no longer reachable, any associated peer and routing table entries MUST be deleted.

7.0 Error Handling

There are two different types of errors in Diameter; protocol and applications. 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, are 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.

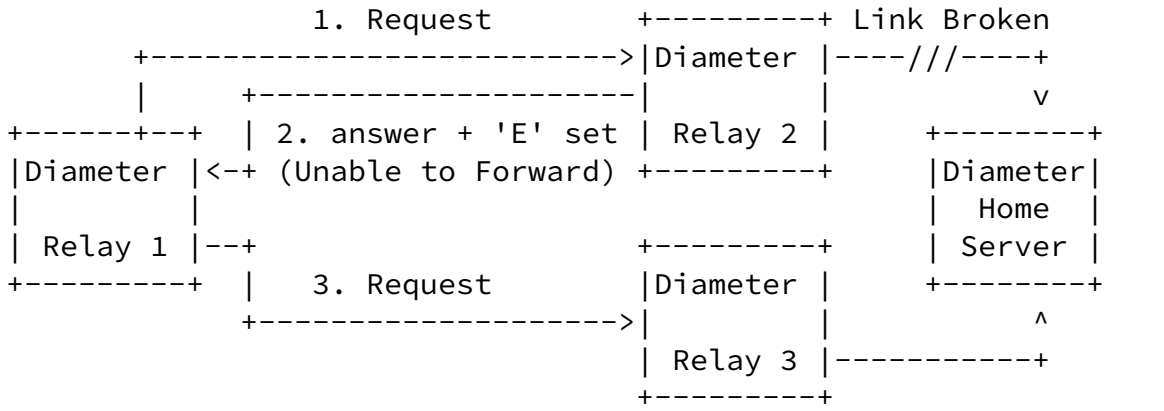


Figure 8: Example of Protocol Error causing answer message

Figure 8 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.

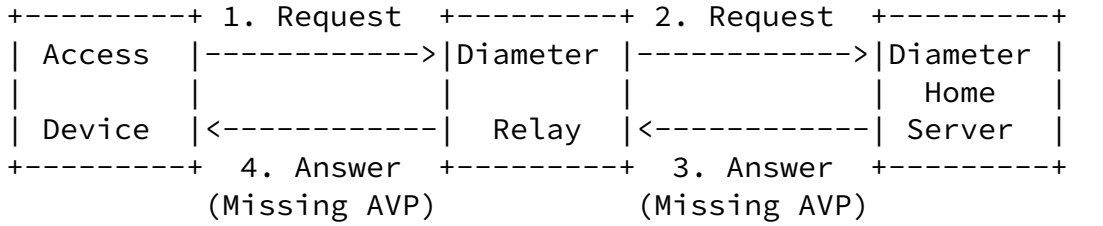


Figure 9: Example of Application Error Answer message

Figure 9 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.

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:

- 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.
- 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 command is received with an AVP that is omitted, yet is mandatory according to the command's ABNF. 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 to the missing AVP's. The created AVP is then added to the Failed-AVP AVP.

The Result-Code AVP contains additional errors conditions, and defines the expected behavior of each.

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 MUST include one Result-Code AVP. A non-successful Result-Code AVP (one containing a non 2xxx value) 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 11.4). Diameter provides the following classes of errors, all identified by the thousands digit:

- 1xxx (Informational)
- 2xxx (Success)

- 3xxx (Protocol Errors)
- 4xxx (Transient Failures)
- 5xxx (Permanent Failure)

A non-recognize 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 required 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

The Request contained a Command-Code that the receiver did not recognize or 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 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.

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.

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.

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. One or more Failed-AVP AVPs MUST be present, containing 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_UNSUPPORTED_TRANSFORM 5010
A message was received that included a CMS-Data AVP [CMS] that made use of an unsupported transform.

DIAMETER_NO_COMMON_APPLICATION 5011
This error is returned when a CER message is received, and there are no common applications supported between the peers.

DIAMETER_UNSUPPORTED_VERSION 5012
This error is returned when a request was received, whose version number is unsupported.

DIAMETER_UNABLE_TO_COMPLY 5013
This error is returned when a request is rejected for unspecified reasons.

DIAMETER_INVALID_BIT_IN_HEADER 5014
This error is returned when an unrecognized bit in the Diameter header is set to one (1).

DIAMETER_INVALID_AVP_LENGTH 5015

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.

DIAMETER_INVALID_MESSAGE_LENGTH 5016
This error is returned when a request is received with an invalid message length.

DIAMETER_INVALID_AVP_BIT_COMBO 5017
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.

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 ABNF specification for the command, and will instead conform to the following ABNF:

Message Format

<answer-message> ::= < Diameter Header: code, ERR [PXY] >

```
                        0*1< Session-Id >
                           { Origin-Host }
                           { Origin-Realm }
                           { Result-Code }
                           [ Origin-State-Id ]
                           [ Error-Reporting-Host ]
                           [ Proxy-Info ]
                         * [ AVP ]
```

Note that the code used in the header is the same that 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 real-time, and

SHOULD NOT be expected to be 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.

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 10.0), or the presence of two or more occurrences of an AVP which is restricted to 0, 1, or 0-1 occurrences.

A Diameter message MAY 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.

AVP Format

```
<Failed-AVP> ::= < AVP Header: 279 >
                 1* {AVP}
```

8.0 Diameter User Sessions

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 is 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.

8.1 Authorization Session State Machine

This section contains a finite state machine, representing the life cycle of Diameter sessions, and 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 IP, NASREQ).

There are two different session state machines supported in the Diameter base protocol. The first consists of a session in which the server is maintaining session state, indicated by the value of the Auth-Session-State AVP (or its absence). The second state machine is used when the server does not maintain session state.

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.

The following state machine is used when state is maintained on the server:

State	Event	Action	New State
Idle	Client or Device Requests access	send service specific auth req	Pending
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 answer	Idle
Pending	Successful Service-specific authorization answer	Grant Access	Open

	received with default Auth-Session-State value		
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	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	Successful Service-specific authorization answer received	Extend Answer	Open
Open	Accounting message sent or received	process	Open
Open	Failed Service-specific authorization answer received.	Discon. user/device	Idle
Open	Session-Timeout Expires on Access Device	send STR	Discon
Open	Home server wants to terminate the service	send ASR	Open
Open	ASA Received	Cleanup	Idle

	with Result-Code = UNKNOWN-SESSION-ID		
Open	ASA Received with Result-Code not = UNKNOWN-SESSION-ID	None (ignore)	Open
Open	ASR Received	send ASA, STR	Discon
Open	Authorization-Lifetime + Auth-Grace-Period expires on access device	send STR	Discon
Open	Authorization-Lifetime (and Auth-Grace-Period) expires on home server.	Cleanup	Discon
Open	Session-Timeout expires on home server	Cleanup	Discon
Open	STR Received	Send STA	Idle
Not Open	ASA Received	None	No Change.
Discon	ASR Received	None	Discon
Discon	STR Received	Send STA	Idle
Discon	STA Received	Discon. user/device	Idle

The following state machine is used when state is not maintained on the server:

State	Event	Action	New State
Idle	Client or Device Requests access	send service specific auth req	Pending

Idle	Service-specific authorization request received, and successfully processed	send serv. specific answer	Open
Pending	Successful Service-specific	Grant Access	Open

	authorization answer received with Auth-Session-State set to NO_STATE_MAINTAINED		
Pending	Failed Service-specific authorization answer received	Cleanup	Idle
Open	Accounting message sent or received	process	Open
Open	Session-Timeout Expires on Access Device	Discon. user/device	Idle
Open	Service to user is terminated	Discon. user/device	Idle

8.2 Accounting Session State Machine

For applications that only require accounting services, the following state machine MUST be supported.

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.

State	Event	Action	New State
Idle	Client or device requests access	send accounting start req.	PendingS
Idle	Accounting start request received, and successfully processed.	send accounting start answer	Open
Idle	Client or device requests a one-time service	send accounting event req	PendingE
Idle	Accounting event request received, and successfully processed.	send accounting event answer	Idle
Idle	Records in storage	Send record	PendingB
Open	Receive Interim Record	send accounting answer	Open
Open	User service terminated	send accounting	PendingL

		stop req.	
Open	Accounting stop request received, and successfully processed	send accounting stop answer	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
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
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

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
PendingB	Successful accounting answer received	Delete record	Idle
PendingB	Failure to send		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 service-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 ]
         * [ AVP ]
         * [ Proxy-Info ]
         * [ Route-Record ]
```

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 ]
                * [ Redirected-Host ]
                  [ Redirected-Host-Usage ]
                  [ Redirected-Host-Cache-Time ]
                * [ AVP ]
                * [ Proxy-Info ]
```

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. 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 the access device 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 ]
           * [ AVP ]
           * [ Proxy-Info ]
           * [ Route-Record ]
```

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-Usase ]
              [ Redirect-Max-Cache-Time ]
            * [ AVP ]
            * [ Proxy-Info ]
```

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 credit or other reasons that were not anticipated when the session was first authorized. On the other hand, an operator may maintain a management server for the purpose of issuing ASRs to administratively remove users from the network.

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.

Note that if the access device does stop the session upon receipt of an ASR, it issues an STR to the authorizing server (which may or may not be the agent issuing the ASR) just as it would if the session were terminated for any other reason.

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 server 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 ]
            * [ AVP ]
            * [ Proxy-Info ]
            * [ Route-Record ]
```

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 ]
         * [ Redirected-Host ]
           [ Redirected-Host-Usage ]
           [ Redirected-Max-Cache-Time ]
         * [ AVP ]
         * [ Proxy-Info ]
```

8.6 Inferring Session Termination from Origin-State-Id

Origin-State-Id is used to allow rapid detection of terminated sessions for which no STR would have been issued, due to unanticipated shutdown of an access device.

By including Origin-State-Id in CER/CAA messages, an access device allows a next-hop server to determine immediately upon connection whether the device has lost its sessions since the last connection.

By including Origin-State-Id in request messages, an access device also allows a server with which it communicates via proxy to make such a determination. However, a server that is not directly connected with the access device will not discover that the access device has been restarted unless and until it receives a new request

from the access device. Thus, use of this mechanism across proxies is opportunistic rather than reliable, but useful nonetheless.

When a Diameter server receives an Origin-State-Id that is greater than the Origin-State-Id previously received from the same issuer, it may assume that the issuer has lost state since the previous message and that all sessions that were active under the lower Origin-State-Id 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 8.0). 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 3.0).

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 4.4). The remainder of the Session-Id 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, 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.acme.com;1876543210;523

Example, in which there is an optional value:
accesspoint7.acme.com;1876543210;523;mobile@200.1.1.88

The Session-Id is created by the Diameter device initiating the session, which in most cases is done by the client. Note that a Session-Id MAY be used for both the 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. Great 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. This is typically used in cases where multiple authentication methods are used, and a successful auth response with this AVP set to zero is used to signal that the next authentication method is to be immediately initiated. 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. However, the server MAY 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. 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) [RADIUS] 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 8.9).

A Session-Timeout AVP MAY be present in a re-authorization message, and contains the 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) [RADIUS] is of type UTF8String, which contains the User-Name, in a format consistent with the NAI specification [NAI].

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 messages for this session MUST be sent to the same authorization server. This AVP MAY also specify that a Session-Termination-Request message for this session MUST be sent to the same authorizing 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 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 to 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.

9.0 Accounting

This accounting protocol is based on a server directed model with capabilities for real-time delivery of accounting information. Several fault resilience methods [ACCMGMT] 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 [ACCMGMT], 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.

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 Accounting-Interim-Interval AVP to control the operation of the Diameter peer operating as a client. The Accounting-Interim-Interval AVP, when present, instructs the Diameter node acting as a client to produce accounting records continuously even during a session.

The Diameter accounting server MAY override the interim interval by including an Accounting-Interim-Interval AVP in the Accounting-Answer message. When the AVP is present, the latest value received SHOULD be used in the generation of interim accounting messages.

9.2 Protocol Messages

A Diameter node that receives a successful authentication and/or authorization messages from the Home AAA server MUST collect

accounting information for the session. The Accounting-Request message is used to transmit the accounting information to the Home AAA 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. A rejected Accounting-Request message SHOULD cause the user's session to be terminated.

Each Diameter Accounting protocol message MAY be compressed using IPComp [IPComp] in order to reduce the used network bandwidth, which MAY use IKE [IKE] to negotiate the compression parameters.

9.3 Application document requirements

Each Diameter application (e.g. NASREQ, MobileIP), MUST 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 this section.

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 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 need 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. If strong authentication across agents is required, as described in [CMS], the CMS-Signed-Data AVP may be used to authenticate the Accounting Data and Service Specific AVPs. It is not typically necessary that the CMS-Signed-Data AVP cover any additional AVPs, but it is permitted as long as the AVPs protected are defined to have their 'P' bit set.

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 directed interim accounting to be enabled for the session, but no interim interval was specified, 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 a specified interim interval exists, 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 both Accounting-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

The Diameter protocol's Session-Id AVP, which is globally unique (see section 8.8), is used during the authorization phase to identify a particular session. Services that do not require any authorization still use the Session-Id AVP to identify sessions.

However, there are certain applications that require multiple accounting sub-sessions. Such applications would send messages with a constant Session-Id AVP, but a different Accounting-Sub-Session-Id AVP. In these cases, correlation is performed using the Session-Id. It is important to 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.

Furthermore, there are certain applications where a user receives service from different access devices (e.g. Mobile IP), each with their own unique Session-Id. In such cases, the Accounting-Multi-Session-Id AVP is used for correlation. During authorization, a server that determines that a request is for an existing session SHOULD include the Accounting-Multi-Session-Id AVP, which the access device MUST include in all subsequent accounting messages.

The Accounting-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 Accounting-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 new 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.

When the Accounting-Request is being submitted to a third party (e.g. settlement service), and includes the CMS-Signed-Data AVP [CMS], the CMS-Signed-Data AVP MUST be signed by both the local and home Diameter server using the countersignature procedures described in [CMS].

The AVP listed below SHOULD include service specific accounting AVPs, as described in section 9.3.

Message Format

```
<ACR> ::= < Diameter Header: 271, REQ, PXY >
          < Session-Id >
          { Acct-Application-Id }
          { Origin-Host }
          { Origin-Realm }
          { Destination-Realm }
          { Accounting-Record-Type }
          { Accounting-Record-Number }
          [ User-Name ]
          [ Accounting-Sub-Session-Id ]
          [ Accounting-RADIUS-Session-Id ]
          [ Accounting-Multi-Session-Id ]
          [ Accounting-Interim-Interval ]
          [ Origin-State-Id ]
        * [ AVP ]
        * [ Proxy-Info ]
        * [ Route-Record ]
```

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 and MAY contains the same Accounting Description and Usage AVPs that were sent in the Accounting-Request command. If the CMS-Data AVP was present in the Accounting-Request, the corresponding ACA message MUST include the CMS-Data AVP signed by the responder to provide strong AVP authentication, which MAY be used for the purposes of repudiation.

Only the target Diameter Server, known as the home Diameter Server, SHOULD respond with the Accounting-Answer command.

The AVP listed below SHOULD include service specific accounting AVPs, as described in section 9.3.

Message Format

```
<ACA> ::= < Diameter Header: 271, PXY >
          < Session-Id >
          { Acct-Application-Id }
          { Result-Code }
          { Origin-Host }
          { Origin-Realm }
          { Accounting-Record-Type }
          { Accounting-Record-Number }
          [ User-Name ]
          [ Accounting-Sub-Session-Id ]
          [ Accounting-RADIUS-Session-Id ]
          [ Accounting-Multi-Session-Id ]
          [ Error-Reporting-Host ]
          [ Accounting-Interim-Interval ]
          [ Origin-State-Id ]
        * [ AVP ]
        * [ Proxy-Info ]
```

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 directed by the Accounting-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 Accounting-Interim-Interval AVP

The Accounting-Interim-Interval AVP (AVP Code 482) 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 Accounting-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 Accounting-RADIUS-Session-Id AVP

The Accounting-RADIUS-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 Accounting-Session-Id attribute.

9.8.5 Accounting-Multi-Session-Id AVP

The Accounting-Multi-Session-Id AVP (AVP Code 50) is of type UTF8String, following the format specified in section 8.8. The Accounting-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 Accounting-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 TBD) 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 behaviour 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 can not be delivered or stored.

10.0 AVP Occurrence Table

The following tables presents the AVPs defined in this document, and specifies in which Diameter messages they MAY, or MAY NOT be present. 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. It is considered an error if there are more than once 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
Accounting-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
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	0	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	1	1	1	1
Vendor-Id	1	1	0	0	0	0	0	0	0	0	0	0
Vendor-Specific-	0+	0+	0	0	0	0	0	0	0	0	0	0

Application-Id												

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.

	Command Code	
Attribute Name	ACR	ACA
Accounting-Interim-Interval	0-1	0-1
Accounting-Multi-Session-Id	0-1	0-1
Accounting-Record-Number	1	1
Accounting-Record-Type	1	1
Accounting-RADIUS-Session-Id	0-1	0-1
Accounting-Sub-Session-Id	0-1	0-1
Accounting-Realtime-Required	0	0-1
Acct-Application-Id	1	1
Class	0+	0+
Destination-Host	0-1	0
Destination-Realm	1	0
Error-Reporting-Host	0	0+
Origin-Host	1	1
Origin-Realm	1	1
Proxy-Info	0+	0+
Route-Record	0+	0+
Result-Code	0	1
Session-Id	1	1
User-Name	0+	0+

11.0 IANA Considerations

This document defines a number of assigned numbers to be maintained by the IANA. This section explains the criteria to be used by the IANA to assign additional numbers in each of these lists. The following subsections describe the assignment policy for the namespaces defined elsewhere in this document.

11.1 AVP

As defined in section 4.0, the AVP header contains two fields that

requires IANA namespace management; the AVP Code and Flags field.

11.1.1 AVP Code

The AVP Code namespace is used to identify attributes. When the Vendor ID value is set to zero (0), IANA will maintain a registry of assigned AVP codes and in some cases also their values. AVP Codes 0-254 are managed separately as RADIUS Attribute Types [RAD TYPE], while the remaining namespace is available for assignment via Specification Required [IANA].

Vendor-Specific AVP Codes, where the Vendor-Id field in the AVP header is set to a non-zero value, are for Private Use.

This document defines the AVP Codes 257-274, 276-285, 287, 291-297, 480, 482 and 485-486. See section 4.6 for the assignment of the namespace in this specification.

11.1.2 AVP Flags

There are 8 bits in the AVP Flags field of the AVP header, defined in section 4.0. This document assigns bit 8 ('V'endor Specific), bit 7 ('M'andatory) and bit 6 ('P'rotected). The remaining bits should only be assigned via a Standards Action [IANA].

11.2 Diameter Header

As defined in section 3.0, the Diameter header contains two fields that require IANA namespace management; Command Code and Command Flags.

11.2.1 Command Codes

The Command Code namespace is used to identify Diameter commands. The values 0-255 are reserved for RADIUS backward compatibility, and are defined as "RADIUS Packet Type Codes" in [RADTYPE]. The remaining values are available via Standards Action [IANA].

Vendor-Specific Command Codes, where the Vendor-Id field in the

Diameter header is set to a non-zero value, are for Private Use.

This document defines the Command Codes 257, 258, 271, 274-275, 280 and 282. See section 3.1 for the assignment of the namespace in this specification.

11.2.2 Command Flags

There are eight bits in the Command Flags field of the Diameter header. This document assigns bit 8 ('R'equest), bit 7 ('P'roxy) and bit 6 ('E'rror). Bits 1 through 5 MUST only be assigned via a Standards Action [IANA].

11.3 Application Identifiers

As defined in section 2.5, the Application Identifier is used to identify a specific Diameter Application. All values except zero (0) are available for assignment via Standards Action [IANA].

Vendor-Specific Application Identifiers, encoded in the Vendor-Specific-Application-Id Grouped AVP, with the Vendor-Id AVP set to the vendor's enterprise number, is for Private Use.

Note that the Diameter protocol is not intended to be extended for any purpose. Any applications defined MUST ensure that they fit within the existing framework, and that no changes to the base protocol are required.

11.4 Result-Code AVP Values

As defined in Section 7.1, the Result-Code AVP (AVP Code 268) defines the values 1001, 2001-2002, 3001-3009, 4001-4002 and 5001-5017.

All remaining values are available for assignment via IETF Consensus [IANA].

11.5 Accounting-Record-Type AVP Values

As defined in Section 9.8.1, the Accounting-Record-Type AVP (AVP Code

480) defines the values 1-4. All remaining values are available for assignment via IETF Consensus [IANA].

11.6 Termination-Cause AVP Values

As defined in Section 8.15, the Termination-Cause AVP (AVP Code 295) defines the values 1-8. All remaining values are available for assignment via IETF Consensus [IANA].

11.7 Redirect-Host-Usage AVP Values

As defined in Section 6.12, the Redirect-Host-Usage AVP (AVP Code 261) defines the values 0-5. All remaining values are available for assignment via IETF Consensus [IANA].

11.8 Session-Server-Failover AVP Values

As defined in Section 8.18, the Session-Server-Failover AVP (AVP Code 271) defines the values 0-3. All remaining values are available for assignment via IETF Consensus [IANA].

11.9 Session-Binding AVP Values

As defined in Section 8.17, the Session-Binding AVP (AVP Code 270) defines the bits 1-4. All remaining bits are available for assignment via IETF Consensus [IANA].

11.10 Diameter TCP/SCTP Port Numbers

An IANA request has been placed for TCP and SCTP port numbers. The IANA has informed the authors that "TBD" should be used in section 2.1 and throughout this document, and will be updated by the RFC editor during the RFC publication process.

IANA should also replace "TBD" in sections 4.4 and 5.2 with the port number assigned in section 2.1.

11.11 Disconnect-Cause AVP Values

As defined in Section 5.4.3, the Disconnect-Cause AVP (AVP Code 273) defines the values 0-2. All remaining values are available for assignment via IETF Consensus [IANA].

11.12 Auth-Request-Type AVP Values

As defined in Section 8.7, the Auth-Request-Type AVP (AVP Code 274) defines the values 1-3. All remaining values are available for assignment via IETF Consensus [TCP].

11.13 Auth-Session-State AVP Values

As defined in Section 8.11, the Auth-Session-State AVP (AVP Code 277) defines the values 0-1. All remaining values are available for

assignment via IETF Consensus [TCP].

11.14 Re-Auth-Request-Type AVP Values

As defined in Section 8.12, the Re-Auth-Request-Type AVP (AVP Code 285) defines the values 0-1. All remaining values are available for assignment via IETF Consensus [TCP].

11.15 NAPTR Service Fields

The registration in the RFC MUST include the following information:

Service Field: The service field being registered. An example for a new fictitious transport protocol called NCTP might be "AAA+D2N".

Protocol: The specific transport protocol associated with that service field. This MUST include the name and acronym for the protocol, along with reference to a document that describes the transport protocol. For example - "New Connectionless Transport Protocol (NCTP), RFC 5766".

Name and Contact Information: The name, address, email address and telephone number for the person performing the registration.

The following values are to be placed into the registry:

Services Field	Protocol AAA+D2T
TCP AAAS+D2T	TLS over TCP AAA+D2S
SCTP AAAS+D2S	TLS over SCTP

12.0 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.

Realm 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 Realms 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.

13.0 Security Considerations

The Diameter base protocol assumes that messages are secured by using either IP Security, or TLS. This security model is acceptable in environments where there is no untrusted third party relay, proxy, or redirect agent.

When third party brokers or redirect agents are used, strong application level security SHOULD be required, such as non-repudiation. When the communicating peers do require this level of security either for legal or business purposes, the Diameter application defined in [CMS] MAY be used. This security model provides AVP-level authentication, and the encryption mechanism is designed such that only the target host has the keying information required to decrypt the information.

13.1 IPsec Usage

All Diameter implementations MUST support IPsec ESP [IPsec] in transport mode with with non-null encryption and authentication algorithms to provide per-packet authentication, integrity protection and confidentiality, and MUST support the replay protection mechanisms of IPsec.

Diameter implementations MUST support IKE for peer authentication, negotiation of security associations, and key management, using the IPsec DOI [IPSECDOI]. Diameter implementations MUST support peer authentication using a pre-shared key, and MAY support certificate-based peer authentication using digital signatures. Peer authentication using the public key encryption methods outlined in IKE's sections 5.2 and 5.3 [IKE] SHOULD NOT be used.

Conformant implementations MUST support both IKE Main Mode and Aggressive Mode. When pre-shared keys are used for authentication, IKE Aggressive Mode SHOULD be used, and IKE Main Mode SHOULD NOT be used. When digital signatures are used for authentication, either IKE Main Mode or IKE Aggressive Mode MAY be used.

When digital signatures are used to achieve authentication, an IKE negotiator SHOULD use IKE Certificate Request Payload(s) to specify

the certificate authority (or authorities) that are trusted in accordance with its local policy. IKE negotiators SHOULD use pertinent certificate revocation checks before accepting a PKI certificate for use in IKE's authentication procedures.

The Phase 2 Quick Mode exchanges used to negotiate protection for Diameter connections MUST explicitly carry the Identity Payload fields (IDci and IDcr). The DOI provides for several types of

identification data. However, when used in conformant implementations, each ID Payload MUST carry a single IP address and a single non-zero port number, and MUST NOT use the IP Subnet or IP Address Range formats. This allows the Phase 2 security association to correspond to specific TCP and SCTP connections.

Since IPsec acceleration hardware may only be able to handle a limited number of active IKE Phase 2 SAs, Phase 2 delete messages may be sent for idle SAs, as a means of keeping the number of active Phase 2 SAs to a minimum. The receipt of an IKE Phase 2 delete message SHOULD NOT be interpreted as a reason for tearing down a Diameter connection. Rather, it is preferable to leave the connection up, and if additional traffic is sent on it, to bring up another IKE Phase 2 SA to protect it. This avoids the potential for continually bringing connections up and down.

13.2 TLS Usage

A Diameter node that initiates a connection to another Diameter node acts as a TLS client according to [TLS], and a Diameter node that accepts a connection acts as a TLS server. Diameter nodes implementing TLS for security MUST mutually authenticate as part of TLS session establishment. In order to ensure mutual authentication, the Diameter node acting as TLS server must request a certificate from the Diameter node acting as TLS client, and the Diameter node acting as TLS client MUST be prepared to supply a certificate on request.

When Diameter uses TLS, it MUST have the same dNSName field requirements as the Diameter CMS Security Application [CMS] listed in section 3.2.

Diameter nodes MUST be able to negotiate the following TLS cipher suites:

```
TLS_RSA_WITH_RC4_128_MD5
TLS_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA
```

Diameter nodes MAY negotiate other TLS cipher suites.

14.0 References

14.1 Normative

[AAATRANS] B. Aboba, J. Wood, "Authentication, Authorization and Accounting (AAA) Transport Profile", draft-ietf-aaa-transport-04.txt, IETF Work in Progress, June 2001.

[ASSIGNNO] Reynolds, Postel, "Assigned Numbers", RFC 1700, October 1994.

[CMS] P. Calhoun, W. Bulley, S. Farrell, "Diameter CMS Security application", draft-ietf-aaa-diameter-cms-sec-03.txt, IETF work in progress, November 2001.

[DIFFSERV] K. Nichols, S. Blake, F. Baker, D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," RFC 2474, December 1998.

[DIFFSERVAF] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, "Assured Forwarding PHB Group," RFC 2597, June 1999.

[DIFFSERVEF] V. Jacobson, K. Nichols, K. Poduri, "An Expedited For¡warding PHB", RFC 2598, June 1999.

[DNSSRV] A. Gulbrandsen, P. Vixie, L. Esibov, "A DNS RR for speci¡fying the location of services (DNS SRV)", RFC 2782, February 2000.

[EAP] L. J. Blunk, J. R. Vollbrecht, "PPP Extensible Authenti¡cation Protocol (EAP)." RFC 2284, March 1998.

[FLOATPOINT] Institute of Electrical and Electronics Engineers, "IEEE Standard for Binary Floating-Point Arithmetic", ANSI/IEEE Standard 754-1985, August 1985.

[IANA] Narten, Alvestrand, "Guidelines for Writing an IANA Con¡siderations Section in RFCs", BCP 26, RFC 2434, October 1998

[IANAWEB] IANA, "Number assignment", http://www.iana.org

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

[IPSECDOI] D. Piper, "The Internet IP Security Domain of Interpreta¡tion for ISAKMP", RFC 2407, November 1998.

[IPV4] ISI, "Internet Protocol", RFC 791, September 1981.

[IPV6] Hinden, Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

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

[NAI] Aboba, Beadles "The Network Access Identifier." RFC 2486. January 1999.

[NAPTR] M. Mealling and R. Daniel, "The naming authority pointer (NAPTR) DNS resource record," Request for Comments 2915, Internet Engineering Task Force, Sept. 2000.

[RADTYPE] IANA, "RADIUS Types", http://www.isi.edu/in-notes/iana/assignments/radius-types

[SCTP] R. Stewart et al., "Stream Control Transmission Proto¡ col". RFC 2960. October 2000.

[SLP] E. Guttman, C. Perkins, J. Veizades, M. Day. "Service Location Protocol, Version 2", RFC 2165, June 1999.

[SNTP] Mills, "Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI, RFC 2030, October 1996.

[TCP] Postel, J. "Transmission Control Protocol", RFC 793, Jan¡ uary 1981.

[TEMPLATE] E. Guttman, C. Perkins, J. Kempf, "Service Templates and Service: Schemes", RFC 2609, June 1999.

[TLS] T. Dierks, C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[TLSSCTP] M. Tuexen, et al. "TLS over SCTP" IETF Work in Progress, November 2001.

[URI] T. Berners-Lee, R. Fielding, U.C. Irvine, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax". RFC 2396, August 1998.

[UTF8] F. Yergeau, "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998.

14.2 Non-Normative

[ABNF] D. Crocker, P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[ACCMGMT] B. Aboba, J. Arkko, D. Harrington. "Introduction to Accounting Management", RFC 2975, October 2000.

[CDMA2000] T. Hiller and al, "CDMA2000 Wireless Data Requirements for AAA", RFC 3141, June 2001.

[DIAMMIP] P. Calhoun, C. Perkins, "Diameter Mobile IP Application", draft-ietf-aaa-diameter-mobileip-08.txt, IETF work in progress, November 2001.

[IPComp] A. Shacham, R. Monsour, R. Pereira, M. Thomas, "IP Payload Compression Protocol (IPComp)", RFC 2393, December 1998.

[MIPV4] C. Perkins, Editor. IP Mobility Support. RFC 2002, October 1996.

[MIPREQ] S. Glass, S. Jacobs, C. Perkins, "Mobile IP Authentication, Authorization, and Accounting Requirements". RFC 2977. October 2000.

[NASREQ] P. Calhoun, W. Bulley, A. Rubens, J. Haag, "Diameter NASREQ Application", draft-ietf-aaa-diameter-nasreq-08.txt, IETF work in progress, November 2001.

[NASCRIT] M. Beadles, D. Mitton, "Criteria for Evaluating Network Access Server Protocols", RFC 3169, September 2001.

[PPP] W. Simpson, "The Point-to-Point Protocol (PPP)", RFC 1661, STD 51, July 1994.

[PROXYCHAIN] B. Aboba, J. Vollbrecht, "Proxy Chaining and Policy Implementation in Roaming", RFC 2607, June 1999.

[RADIUS] C. Rigney, A. Rubens, W. Simpson, S. Willens, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865,

June 2000.

[ROAMCRIT] B. Aboba, G. Zorn, "Criteria for Evaluating Roaming Pro¡ tocols", RFC 2477, January 1999.

[SECARCH] S. Kent, R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

15.0 Acknowledgements

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 com¡ ments 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, 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.

16.0 Authors' Addresses

Questions about this memo can be directed to:

Pat R. Calhoun

Black Storm Networks
250 Cambridge Avenue, Suite 200
Palo Alto, California, 94306
USA

 Phone: +1 650-617-2932
   Fax: +1 650-786-6445
E-mail: pcalhoun@diameter.org

Jari Arkko
Oy LM Ericsson Ab
02420 Jorvas
Finland

 Phone: +358 40 5079256
E-Mail: Jari.Arkko@ericsson.com

Erik Guttman
Solaris Advanced Development
Sun Microsystems, Inc.
Eichhoelzelstr. 7
74915 Waibstadt
Germany

 Phone: +49-7263-911-701
E-mail: erik.guttman@germany.sun.com

Glen Zorn
Cisco Systems, Inc.
500 108th Avenue N.E., Suite 500
Bellevue, WA 98004
USA

Phone: +1 425 438 8218

John Loughney
Nokia Research Center
Itämerenkatu 11-13
00180 Helsinki
Finland

Phone: +358 50 483 6242
E-mail: john.Loughney@nokia.com

17.0 Full Copyright Statement

18.0 Expiration Date

This memo is filed as <draft-ietf-aaa-diameter-10.txt> and expires in September 2002.

Appendix A. Diameter Service Template

The following service template describes the attributes used by Diam¡ eter servers to advertise themselves. This simplifies the process of selecting an appropriate server to communicate with. A Diameter client can request specific Diameter servers based on characteristics of the Diameter service desired (for example, an AAA server to use for accounting.)

Name of submitter: "Erik Guttman" <Erik.Guttman@sun.com>
Language of service template: en

Security Considerations:
   Diameter clients and servers use various cryptographic mechanisms
   to protect communication integrity, confidentiality as well as
   perform end-point authentication. It would thus be difficult if
   not impossible for an attacker to advertise itself using SLPv2 and
   pose as a legitimate Diameter peer without proper preconfigured
   secrets or cryptographic keys. Still, as Diameter services are
   vital for network operation it is important to use SLPv2 authenti¡
   cation to prevent an attacker from modifying or eliminating ser¡
   vice advertisements for legitimate Diameter servers.

Template text:
```
-------------------------template begins here----------------------
template-type=service:diameter

template-version=0.0

template-description=
  The Diameter protocol is defined by draft-ietf-aaa-diameter-10.txt

template-url-syntax=
  url-path= ; The Diameter URL format is described in section 2.9.
            ; Example: 'aaa://aaa.abc.com:1812;transport=tcp
```



```
   supported-auth-applications= string L M
   # This attribute lists the Diameter applications supported by the
   # AAA implementation.  The applications currently defined are:
   #  Application Name     Defined by
```

```
#  ----------------     ----------------------------------
#  NASREQ               draft-ietf-aaa-diameter-nasreq-08.txt
#  MobileIP             draft-ietf-aaa-diameter-mobileip-08.txt
#  CMS Security         draft-ietf-aaa-diameter-cms-sec-03.txt
#
# Notes:
#    . Diameter implementations support one or more applications.
#    . Additional applications may be defined in the future.
#      An updated service template will be created at that time.
#
NASREQ,MobileIP,CMS Security

supported-acct-applications= string L M
# This attribute lists the Diameter applications supported by the
# AAA implementation.  The applications currently defined are:
#  Application Name     Defined by
#  ----------------     ----------------------------------
#  NASREQ               draft-ietf-aaa-diameter-nasreq-08.txt
#  MobileIP             draft-ietf-aaa-diameter-mobileip-08.txt
#  CMS Security         draft-ietf-aaa-diameter-cms-sec-03.txt
#
# Notes:
#    . Diameter implementations support one or more applications.
#    . Additional applications may be defined in the future.
#      An updated service template will be created at that time.
#
NASREQ,MobileIP,CMS Security

supported-transports= string L M
SCTP
# This attribute lists the supported transports that the Diameter
# implementation accepts.  Note that a compliant Diameter
# implementation MUST support SCTP, though it MAY support other
# transports, too.
SCTP,TCP
```

```
-------------------------template ends here-----------------------
```

Appendix B. NAPTR Example

As an example, consider a client that wishes to resolve aaa:ex.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 20   50  "s"  "AAAS+D2S"          ""  _diame¡
   ters._sctp.ex.com.  IN NAPTR 50   50  "s"  "AAA+D2S"           ""
   _diameter._sctp.ex.com.  IN NAPTR 90   50  "s"  "AAAS+D2T"
   ""  _diameters._tcp.ex.com.  IN NAPTR 100  50  "s"  "AAA+D2T"
   ""  _aaa._tcp.ex.com
```

This indicates that the server supports TLS over SCTP, SCTP, TLS over TCP, and TCP, in that order. If the client supports TLS over SCTP, SCTP will be used, targeted to a host determined by an SRV lookup of _diameters._sctp.ex.com. That lookup would return:

```
;;          Priority Weight Port   Target
   IN SRV  0        1      5060   server1.ex.com IN SRV  0        2
   5060   server2.ex.com
```

Appendix C. Duplicate Detection

As described in section 9.4, accounting record duplicate detection is based on the session identifiers. Duplicates can appear for various reasons:

- Failover to an alternate server. Where we close to real-time performance is expected, failover tresholds need to be kept low and this may lead to a relatively large likelihood of duplicates.

- A crash of a client at the time it just had managed to send a record from a non-volatile memory would likely cause the same record to be sent soon after the client has rebooted.

- Duplicates received from RADIUS gateways.

- Implementation problems and misconfiguration.

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 User-Name contained in them and duplicate elimina¡ tion is easy in this case.

In other situations it may be necessary to perform real-time dupli¡ cate detection, e.g. when the credit limits or fraud attempts are being monitored in real time.

In general, only the duplicate generation at failover case is some¡

thing that can be reliably detected by the Diameter client. The

Diameter server is therefore responsible for the duplicate detection process. When real-time duplicate detection is required, this implies a database-like search functionality to find duplicate records. Implementors are advised, however, that there exists ways to avoid expensive all-record searches. For instance, it can be usually safely assumed that duplicates appear within a time window of longest imaginable network partition, perhaps a day as an example. So only records within this time window need to be looked at. Secondly, hash¡ ing techniques or other schemes may be used to eliminate the need to do a full search even in this set except for rare cases.